Institut für Informatik
der Technischen Universität München

Interaction and Collaboration Mechanisms for Distributed Communities and Groups in Educational Settings

Chengmao Xu

Institut für Informatik
der Technischen Universität München

Interaction and Collaboration Mechanisms for Distributed Communities and Groups in Educational Settings

Chengmao Xu

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender:		Univ.-Prof. Dr. E. Jessen
Prüfer der Dissertation:	1.	Univ.-Prof. Dr. J. Schlichter
	2.	Univ.-Prof. Dr. A Brüggemann-Klein

Die Dissertation wurde am 25.04.2000 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 04.12.2000 angenommen.

Abstract

In the past few years, the dramatical technological advances, in particular the Internet and the Web, have changed the way that knowledge is conveyed between tutors and students. Distance learning systems are supposed to facilitate this knowledge transfer by harnessing potentials of these new technologies. However, there is no integrated learning environment available up to now which can support various teaching and learning scenarios of students and tutors in a consistent way. Most existing distance learning systems only support limited teaching and learning activities. Students and tutors must move between different tools in their different learning and teaching phases. In particular, current distance learning systems often put great emphasis on information sharing (i.e. preparation, distribution and utilization of electronic learning material) and have not offered effective approaches to interaction and collaboration among students as well as between students and tutors who are dispersed across time and space.

Communityware and groupware are expected to support interaction and collaboration between people. But studies have shown that they have developed separately in the past. While communityware is primarily concerned with creating, maintaining, and evolving social interaction in communities, groupware is aimed at supporting groups of people engaged in a common task and pursuing a common goal.

The task of the thesis is then to incorporate both communityware and groupware to design and develop an integrated learning system. The research method adopted in this thesis is to have each chapter focus on a few key issues from CSCW and then apply these discussion results to develop the targeted system model. First of all, a pyramid model consisting of lectures, discussion groups and learning groups in educational settings will be defined by investigating issues remaining in existing CSCW and distance learning systems. To support community interaction and groupwork seamlessly, we will extensively examine spatial approaches to communities and groups. Such a discussion will lead us to a spatial framework comprised of an entrance hall, a discussion group, a tutor studio and several learning groups. Then an artifact-based model used to design group rooms will be discussed, and a set of key components facilitating both interaction and collaboration between people within the spatial world will be presented.

In addition, a notification server will be extensively investigated because both synchronous and asynchronous awareness play a key role in facilitating interaction and collaboration between people. A set of new technologies for distance learning such as Web-Based Information Systems, and XML/XSL technologies will be addressed in great detail as well.

Though this work is oriented at supporting interaction and collaboration in educational settings, it can, of course, be applied for other working environments as long as both interaction and collaboration have to be supported because most organizations work in communities, groups and teams.

Danksagung

Die vorliegende Arbeit entstand während meiner Tätigkeit an dem Lehrstuhl für Angewandte Informatik – Verteilte Systeme der Technischen Universität München als Stipendiat des DAAD (Deutschen Akademischen Austauchdients).

Mein Dank gilt zuallererst Herrn Prof. Johann Schlichter, der mir diese Dissertation ermöglichte. Neben seiner akademischen Betreuung hat er in den letzen drei Jahren meine Entwürfe sowohl in Englisch als auch in Deutsch Wort für Wort korrigiert. Dies hat auf mich einen unvergeßlichen Eindruck gemacht.

Dr. Michael Koch und Dr. Gunnar Teege möchte ich dafür danken, daß Sie mit Ihren zusammen durchgeführten Diskussionen und Kooperationen zu meiner Dissertation beigetragen haben. Insbesondere danke ich Dr. Martin Bürger, der mir in vieler Hinsicht geholfen hat, als ich gerade in Deutschland ankam.

Frau Evelyn Gemkow hat die meisten Kapitel der Dissertation sprachlich korrigiert. Ich bedanke mich herzlich dafür, dass sie keine Mühe gescheut hat, mir zu helfen.

Darüber hinaus danke ich Frau Schädlich des DAAD, die ich noch nie getroffen habe, aber die mir durch zahlreiche Schreiben und Mails geholfen hat.

Herzlichen Dank gebührt auch meinen Kolleginnen und Kollegen am Lehrstuhl von Herrn Prof. Schlichter für die freundliche Hilfe und das ganze Arbeitsklima.

Nicht zuletzt danke ich meiner Frau, die mich während meiner Promotion in Deutschland begleitet hat. Ohne ihre Anstrengungen hätte ich nicht mit dem Schreiben der Dissertation zu Ende kommen können.

Contents

1 Introduction

1.1 Background

1.1.1 Distance Learning

"*Today's university is at a turning point, and turn it must. The time has come to recognize that education is a business and students are the consumers*," argues Tsichritzis, the chairman of the board at the GMD, the German National Research Center for Information Technology [Tsichr99]. After centuries of evolutionary changes, the predominant model in current universities is still the combination of traditional *forschung und lehre* (research and teaching) as proposed by Wilhelm von Humboldt in the last century.

In the past few years, the dramatical technological advances, for example, the exponential development of the Internet and the Web (World Wide Web), have changed the way that knowledge is conveyed in universities. As people are trying to take advantage of new technologies to shape a new educational era, universities are undergoing tremendous changes. For example, a very bold concept has recently emerged from literature, i.e. *education brokerage*. With the help of education brokerage, universities do not have to produce all their lectures and seminars locally. Rather, learning material is offered in a just-in-time and on-demand way [Hamala96]. This includes not only books, CD-ROMs, and other off-line materials, but live lectures and discussions through networks from other professors or specialists. In some cases, agents can be used to locate required information or the best experts who can offer it. In short, it is expected that learning material can be delivered to groups or individually, off-line or on-line, and at different quality and cost levels [Tsichr99]. There is no doubt that more efforts will be made to develop the education brokerage in the near future.

The dramatic advances in technologies do not only influence the teaching and learning processes in traditional universities, but also lead to the emergence of other forms of educational environments. *Virtual universities* [Chella97], for example, do not have traditional campuses, professionals' offices or libraries, but electronic workplaces and global digital libraries.

Broadly speaking, *distance learning systems* refer to various computer and telecommunication systems facilitating knowledge transfer between tutors and students. Distance learning systems do not only include systems supporting knowledge transfer between students and tutors who are distributed in space and time, but also those used in traditional classes where students and tutors are face-to-face.

1.1.2 New Technologies for Distance Learning

Schlichter has summarized technologies which already have a great impact on the development of distance learning systems [Schlic97a]:

Hypermedia The hypertext concept allows improved student orientation because learning materials (e.g. lecture notes, research reports, books, and seminar papers) and administrative information are better structured. Hypertext course material may be used either for home study and lecture preparation by students or for knowledge presentation by the lecturer.

Multimedia Multimedia enables new possibilities of using various methods to represent complex structures and arguments. Depending on the degree of supported interactivity, the range of multimedia-supported course material extends from learning through advice, passive tutoring (self-controlled learning) and training (learning by exercise) to active tutoring (supervised learning), games (entertainment), simulation (discovering while learning) and problem-solving (learning by doing).

Telecommunication The improved bandwidth of local and wide area networks opens up a new range of opportunities both for the distribution of teaching material electronically and for the communication between students and lecturers. Telecommunication may help to reduce the space dependencies in the university environment. Lectures may be broadcasted via networks between universities. Remote students can ask questions to the lecturer via back channel.

Computer-supported cooperative work CSCW (Computer-Supported Cooperative Work) has played a key role in facilitating collaboration between people in distance learning. Many distance learning systems make use of CSCW to support the interaction between students and tutors who are distributed in time and space. As far as universities are concerned, concepts and tools developed in CSCW can be used to support a collaborative teaching and learning environment which facilitates group learning, the execution of team projects, and the establishment of discussion groups. CSCW might also facilitate the paradigm shift from instructionalism to constructivism.

In spite of these remarkable advances in technologies, traditional distance learning systems put great emphasis on independent study, supported by well developed learning material and the superconnectivity of the Internet and the Web. However, knowledge acquisition in educational environments must involve interaction and collaboration. Research on learning from the constructive and interactive perspectives shows that interaction and collaboration among students, tutors and experts of the educational process are one of the most important aspects in knowledge acquisition. Previous studies about educational systems have indicated that the current support for interaction and collaboration is still far from people's actual need because computer networks are primarily designed for distributing information [Chella97]. Most applications of technology to distance learning have involved expensive solutions to the problem of information delivery and have ignored the important aspect of interaction and collaboration among the many participants.

1.1.3 Communityware and Groupware

CSCW (Computer-Supported Cooperative Work) systems are supposed to support interaction and collaboration between people. The workshop organized by Irene Greif of MIT and Paul Cashman of DEC in 1984 [Grudin94] ignited the boom of the research area of CSCW. Tackling issues like coordination, collaboration and cooperation, CSCW reflected a change in emphasis from using the computer to solve problems to using the computer to facilitate human interaction [Ellis91]. According to Wilson:

> *"CSCW is a generic term which combines the understanding of the way people work in groups with the enabling technologies of computer networking, and associated hardware, software, services and techniques."* [Wilson91]

While the word *group* in Wilson's definition is just a generic term, research and development of CSCW systems typically support collaborative work of well-organized people who are dispersed across time and space. Compared with Wilson's definition, Teufel's definition of CSCW is based on the specific concepts of *workgroups* and *teams* whose members are involved in common tasks or willing to reach common goals:

> *"CSCW is the notion of an interdisciplinary research area which focuses on how individuals work together in workgroups or teams, and how they can be supported by information and communication technologies."* [Teufel95a]

While CSCW refers to a multidisciplinary research field, *groupware* is software supporting collaborative work in groups and teams. A typical example is that of a group of people working together synchronously or asynchronously using computers connected via local area networks.

Meanwhile, the term *communityware* has appeared in the past few years to support more diverse and amorphous groups of people [Ishida98]. According to Ishida, communityware refers to the methodologies and tools for creating, maintaining, and evolving social interaction in communities. In other words, compared with groupware, communityware focuses on an earlier stage of collaboration, i.e. group formation from a wide variety of people. Apparently, communityware developed independent of CSCW. However, there are two reasons for us to discuss both communityware and groupware in the context of CSCW:

- Some experts have proposed a wider view of CSCW systems, suggesting that CSCW does not need to only focus on narrowly-defined groups [Ellis91].
- There should be a close relationship between communityware and groupware because different levels of communication supported by communityware and groupware are actually a continuum.

Though we can discuss both groupware and communityware in the context of CSCW, the independent development of communityware and groupware has affected the consistent support for interaction and collaboration in distance learning. This thesis aims at tackling both interaction and collaboration mechanisms in distance learning by harnessing the potentials of both communityware and groupware.

1.2 Problems

With the help of CSCW, a wide variety of distance learning systems have been designed and developed for supporting diverse education scenarios. Especially, they are supposed to support interaction and collaboration among tutors, students and specialists in a flexible way. Unfortunately, a lot of problems remain unresolved. Instead of only concentrating on issues appearing from existing CSCW systems such as those pointed out by Grudin ten years ago [Grudin88], we tackle issues which arise from the application of CSCW systems for distance learning.

1.2.1 Support for Interaction and Collaboration

Currently, a typical description about the support for interaction and collaboration in distance learning looks as follows:

> *"Interaction and collaboration among students as well as between students and tutors are not constrained in classes. Interaction can be offered at many levels. Interaction can take the form of an email-based service giving a reply within a few hours, or the form of a continual online question/answering chat room. Interaction can be offered asynchronously, live at the end of the lecture, or even preemptively at any point during the lecture."* [Tsichr99]

While a wide variety of CSCW systems have been developed in the past few years, only a small number of them have been successfully applied for distance learning. With respect to the support for interaction and collaboration, most current distance learning assert that they have provided people with synchronous or/and asynchronous communication mechanisms.

Asynchronous Collaboration

Typical systems for asynchronous interaction and collaboration in distance learning include:

- World Wide Web (Web)

  The Web [Berner94] merges the concepts of hypertext and networked information to provide an easy but powerful global information system based on two public and simple standards: HTTP (HyperText Transfer Protocol) and HTML (HyperText Markup Language). In distance learning, the Web has dramatically changed the way that educational information is produced, distributed and used. Nowadays, a wide variety of educational information from lecture scripts, seminar reports, course information, to notifications are available in the Web. Tutors are more willing to move their course material from traditional transparencies into the Web with different available files (e.g. PostScript, HTML and PDF). Depending on their different learning phases, students can either browse them on-line or print them.

- Electronic Mail (email)

  Currently, email is thought of as the most successful CSCW system. In educational settings, email can support interaction between students and tutors in many ways. Students can use email to contact their tutors and supervisors, posing questions about their lectures and seminars, discussing issues relevant to their theses and reports. They can send mails to unknown specialists around the world who can provide valuable information for their study. Among students, email is also an important communication tool for them to talk about various issues with each other, ranging from private interests to specific issues emerging from their learning process. For tutors, electronic mail is a convenient and inexpensive tool to distribute information and to discuss issues or answer questions concerning learning and teaching activities.

- Bulletin Board System (BBS)

  BBS allows participants to create discussion groups for connecting messages related to a certain subject. In distance learning, BBS can be seen as an information exchange market, where some people raise questions and others give answers. Such an information exchange process usually involves only students. Few tutors have time or interest in engaging in this kind of discussions.

Synchronous Collaboration

Compared with asynchronous systems, synchronous systems enable students and tutors to work together in real-time. So far, the following systems have been widely used in distance learning systems:

- MBone (Multi-cast Backbone)

  While the Web has significantly changed the way that learning material is distributed and used, it separates information providers and consumers (i.e. tutors and students). The improved bandwidth of both local and wide area networks opens up a new range of opportunities for the communication between students and tutors. Being able to relay lectures and seminars in real time, MBone allows students and tutors who are distributed geographically to appear in virtual class rooms “face-to-face”. Remote students can ask questions via back channel. If a tutor is using a Web browser to present his lecture, some systems allow remote students to use their local Web browsers to follow the presentation in the same pace as their tutor. This technology helps to overcome the problem that remote students in MBone usually cannot get a clear view of transferred images of scripts used by their lecturers.

- Text Chat

  With more and more Web sites supporting chat rooms, text chat is evolving into one of the most popular interaction facilities in the Internet. In distance learning, students are increasingly using chat tools to exchange their learning experiences and to get to

know more people with some common interests and backgrounds. Experiences with chat rooms show that participants in a chat are more willing to use their self chosen nicknames. Indeed, the principle of anonymity has a positive influence on dynamical interactive environment.

1.2.2 Problems in Interaction and Collaboration

In spite of these various systems available today, studies about distance learning indicate that these mechanisms are insufficient enough to meet the needs of students and tutors.

Discreteness of CSCW Applications

In the previous section, we have introduced several CSCW systems used to support interaction and collaboration between students and tutors in educational settings. One of the basic problems is that most existing distance learning systems just provide a set of discrete CSCW applications.

As introduced above, most groupware systems support either synchronous and or asynchronous systems, but not both [Greenb98]. There are several major categories of asynchronous systems such as email and BBS. Similarly, we have different categories of synchronous systems supporting real-time communication ranging from chat tools to video/audio conferencing systems. It is therefore difficult for users to move fluidly across the synchronous/asynchronous barrier. For a group to switch between different groupware tools, they must start applications, establish communication channels and import documents from one system to another. This is a heavyweight transition. Ideally, it should be very easy for students and tutors to switch between applications.

Separation of Different Phases of Collaborative Learning

This discreteness of CSCW applications makes it difficult to implement an integrated environment where students can study individually or interact with each other easily and conveniently. Because most existing distance learning systems directly make use of these discrete CSCW applications, the phases of a collaborative learning process are separated. Generally speaking, we can divide a collaborative learning process into several phases, which can be observed in terms of information sharing and interaction respectively:

Information Sharing A typical computer-supported knowledge transfer process may include the following phases:

- *production*: Tutors produce learning material such as lecture scripts and exercises for students. As discussed before, universities do not need to produce all their material locally.

- *distribution*: The learning material produced by tutors or experts is normally maintained in various on-line systems such as Web-Based Information Systems [Isakow98]. Correspondingly, students can directly get and make use of learning material through the Internet.
- *acquisition*: Students can find needed knowledge sources with the aid of course information published by tutors, searching engines, information filtering systems and agent systems. Agents can automatically search information on behalf of a user according to certain searching criteria defined by the user.
- *manipulation (private study)*: After acquiring needed information, students can use it for their study. They can use it for preparing their lessons before classes, reviewing courses after classes, completing tasks and preparing their exams.

Interaction On the other hand, interaction is needed in a collaborative learning process. To set up an interaction channel, people are normally required to go through the following necessary steps:

- *locating partners*: The first step for interaction is to find potential partners. In traditional campus, it should not be a problem because interaction usually occurs among students who visit the same class. In distance learning, locating desired partners to interact with may be more complicated than expected.
- *group formation*: Interaction between people often happens in some organizational form. After finding appropriate partners, people need to set up some organizational form to interact with each other, e.g. discussing issues emerging from the learning process. Typical organizations may be communities, groups or teams.
- *interaction*: Within an organization, communication facilities (e.g. email and chat tools) help students exchange learning experience and solve problems.

The problem is that most existing distance learning systems may support one or several of these phases, but not all. BBS, for example, allows people to exchange experience or to express desire to collaborate with others, but does not support a learning group in which they can work together for a common goal. To set up a group, people must make use of other tools like Teamrooms [Rosema96b]. Likewise, recommender systems and database retrieval systems are typical systems used to retrieve and filter information about individuals, but do not allow close collaboration. On the other hand, chat tools, video/audio conferencing systems can better support close collaboration, but they do not support the manipulation of learning material.

Inconsistent Support for Different Levels of Communications

The fact that the phases of a collaborative learning process are separated in existing distance learning systems tells us that distance learning should not simply make use of discrete CSCW

applications. Rather, an integrated learning environment must be provided. To achieve this goal, it is indispensable to thoroughly analyze the need of both students and tutors in distance learning. Behind the failure of existing distance learning systems is the fact that most of them have not successfully recognized both students and tutors' need for different levels of communications during their whole learning and teaching process and then have not successfully supported them in a consistent way.

Basically, there are four different levels of communications: informing, coordination, collaboration and cooperation [Bair89]. Experiences with distance learning show that these different levels of communications are required by people in different phases discussed above.

Firstly, informing plays a key role in educational settings. Informing is a basic activity in teaching and learning. It is often a one way communication channel. The Web, for example, has evolved into one of the most important infrastructure for tutors to distribute learning materials. Students have increasingly become accustomed to getting what they want for their study. A somewhat higher level of communication is coordination between people. When a work involves shared use of information and other resources, some kind of coordination is required. For example, a group of students who participate in a practical course may need to coordinate their activities. Further, a group of students who are engaged in the design and development of a project need to collaborate or cooperate together.

An ideal learning environment should support these different levels of communications among students and between students and tutors smoothly and consistently; this means that all phases of a collaborative learning, from conducting private study, to locating a partner, to interacting with each other, and to working together closely, have to be supported in a consistent way.

1.2.3 Other Issues in Distance Learning

In addition to problems regarding interaction and collaboration, there are also other issues in distance learning which will more or less influence the design and development of distance learning systems. Let us have a look at these issues respectively.

Pedagogical Issues

Because new learning and teaching paradigms are emerging, traditional learning and teaching models are facing more and more challenges and are due to changes. The use of distance learning systems offers opportunities to ask two basic questions:

- What is the new role of tutors?
- What is the new role of students?

Both tutor and student roles are changing as learning and teaching methods in distance learning undergo a dramatic shift. Clearly the tutor remains a source of knowledge and specifies

what should be learned for students. Designing and presenting content to students continue to be the responsibility of the tutor. However, as learning activities are not restrained in conventional classes any more and students have various opportunities to acquire needed knowledge, tutors role will change from primarily producing and explaining teaching material to supervising the learning process and answering their questions.

On the other hand, with the aid of the Web/Internet, students can study independently as well as in groups. They can work at their own pace in their own way. Individuals become more and more responsible for their own acquisition of knowledge instead of being passive as in traditional classes.

The role changes of students and tutors show that the electronic support for interaction and collaboration among students as well as between students and tutors has become unprecedentedly important. Distance learning systems must pay more attention to this concern. It also explains why this thesis is devoted to the discussion of interaction and collaboration.

Quality Issues

Since the advent of distance learning, serious concern about qualities has been raised at once. In addition to decreased interaction in distance learning, quality issues have also caused sharp critics from people, including:

- Content Quality

  As learning material in distance learning may be provided in various ways, a natural question to ask is how to ensure that this material can remain at a higher quality level and who or which organization should take the responsibility for the quality control. Though various measures have been proposed in the past to tackle this issue, it has remained as a serious concern of people.

- Delivery Quality

  Although there should be currently no technical difficulties in delivering learning material through the Internet, students will quickly become frustrated if systems used to deliver it are poorly developed or the systems crash too often. Because most learning material is delivered as hypermedia documents, designers of distance learning systems must consider issues relating to hypermedia management and use of hypermedia, including necessary hypermedia navigation tools.

These concerns teach us that the construction and use of electronic learning material continue to be key issues which this thesis will cover. Without the strong and flexible support from well-defined learning material, a distance learning system can not well support basic learning activities, let alone the support for interaction and collaboration among geographically distributed students and tutors.

Ethical Considerations

As in traditional education, ethical issues also exist in distance learning [Lawhea97]. However, distance learning causes new concerns. With the rapid development of distance learning, people raise questions about whether degrees earned primarily by distance learning are equivalent to conventional degrees offered by the same type of institution, and whether learning materials and methods have been professionally reviewed. Meanwhile, there are also additional ethical issues, such as how to deal with copyright and intellectual property in distance learning. The relationship between professors who offer materials for distance learning and the institutions hiring them also becomes complicated. It is not expected to discuss all of these issues in this thesis, but some of them, for example, the ethical issues concerning behaviors of students in virtual learning environments will be extensively discussed. Likewise, how to protect users' privacy in distance learning will be one of our key concerns in this thesis.

1.3 Goals

This thesis focuses on the design and development of an integrated learning environment providing consistent support for all phases of distance learning, ranging from informing, coordination, collaboration, to cooperation. In a broad sense, these different levels of communications can be classified into *interaction* and *collaboration*. Given the fact that communityware and groupware are primarily aimed at supporting interaction and collaboration respectively, this thesis is intended to harness the potentials of both communityware and groupware for supporting community interaction and groupwork between students as well as between students and tutors.

According to this basic goal, this thesis is supposed to address three issues as follows:

- **Investigation of both communityware and groupware**

  The goal to support all phases of collaborative learning in an integrated environment makes it necessary to thoroughly investigate existing communityware and groupware systems. Such an investigation should lead us to recognize which problems still remain unresolved, and helps us design an integrated learning environment.

- **Examination of distance learning systems**

  Meanwhile, it is indispensable to examine existing distance learning systems for supporting interaction and collaboration in educational settings. This study should clarify what is actually needed by students and tutors, what has been achieved, as well as what problems remain unresolved.

- **Integrated support of communityware and groupware for communities and groups in educational settings**

> Drawing on these studies about CSCW and distance learning, I will focus on developing a system integrating communityware and groupware aspects to support community interaction and groupwork in educational settings.

Further, CSCW systems traditionally put great emphasis on issues such as synchronous/asynchronous cooperation, awareness information, concurrency control, information management etc. Although other issues such as privacy control, history management and version management have been extensively discussed in literature, few distance learning systems have emphasized their importance in supporting collaborative work. This thesis will pay more attention to these issues and describe how to implement them in educational settings.

1.4 Project: Lecture 2000

The work done in this thesis is part of project *Lecture 2000* [Schlic97b, Schlic97a]. The geographical distribution of a university often restricts and encumbers the collaboration between students, professors and research assistants. For example, the fourteen chairs in the Informatics department of the Technische Universität München (TUM) are located at six different sites requiring up to half an hour travel time by public transportation to go from one site to another. Students must shuttle between dormitories, chair locations and the main campus of the university to attend classes and to interact face-to-face with tutors and fellow students. The project Lecture 2000 aims at supporting and enriching common university lecturing by applying the potential of new technologies and by designing and implementing an integrated multimedia based learning and teaching environment. Lecture 2000 provides a comprehensive information space and supports collaboration facilities which supports the communication and collaboration among students as well as between students and lecturers even when they are distributed across several locations.

The information space supports both students in their studies and lectures in their attempts to create, present and distribute courses. It consists of informations as follows:

- lecture and seminar papers,
- exercises and solutions,
- administrative information such as announcements of courses, available theses or projects, curricula and general study regulations,
- domain specific dictionaries and glossaries as well as references to external information sources.

In addition to functionality to access and manipulate the information space, Lecture 2000 supports a collaborative work environment which enhances the collaboration between participating users. Though the information space of Lecture 2000 will be discussed in the thesis, great emphasis of this thesis will be put on the collaboration mechanisms in educational settings.

1.5 Overview of the Thesis

For achieving the goals listed in the preceding section, I try to apply spatial approaches to solve problems inherent with distance learning systems and CSCW systems. The development of concepts and the orientation of the work can be illustrated by Figure 1.1.

In **chapter 2**, I will first examine two groups of concepts of CSCW systems: *interaction, coordination, collaboration* and *cooperation* as well as *community, group,* and *team*. This discussion leads us to define two key concepts which serve as the cornerstones of my discussion through the whole thesis: *community interaction* and *groupwork*, which are supposed to be supported by communityware and groupware respectively.

Communityware, a burgeoning research area in the past few years, refers to systems which are intended to provide support for people who do not know each other personally but are willing to share some mutual understanding and experiences. In contrast, *groupware* are computer-based systems which aim at supporting a group of people who know each other and are engaged in a common task (or goal).

The discussion in this chapter concludes that communityware and groupware, which have developed separately in the past few years, are concerned with distinct stages of social activities of people. However, there is an increased need to have a seamless transition between communityware and groupware. Virtual communities and groups in educational settings, for example, necessitate an integrated support from both communityware and groupware.

In light of *distance* separating students and tutors, **chapter 3** first shows a taxonomy of various distance learning systems, for which two aspects are of particular importance: *information sharing* and *collaboration*. I am primarily concerned with the deficiency of their support for collaboration among students as well as between students and tutors. The main reasons for the failure are that these systems have poorly recognized different levels of communications needed by participants involved in learning and teaching processes, and that these systems lack effective mechanisms to organize and to facilitate these communications.

Based on the investigation, I will turn the attention to the project Lecture 2000, one of whose major goals is to support collaboration in distributed educational settings. Instead of depending on discrete synchronous and asynchronous mechanisms, Lecture 2000 attempts to harness the potentials of both communityware and groupware for supporting both interaction and collaboration in educational settings. Such a goal motivates the development of *the pyramid model* which integrates various collaborative entities (i.e. communities, groups and teams) in educational settings.

Because of their strong relationship to physical reality and therefore their highly intuitive nature, *spatial approaches* have recently often been used for software systems supporting community interaction and groupwork. In **chapter 4**, a wide various spatial approaches to communityware (e.g. Media Spaces, Virtual Reality Systems, MUDs and MOOs) and groupware (e.g. TeamRooms) will be examined in great detail. The conclusion of such a examination is that in spite of their tremendous potentials for both community interaction and groupwork, spatial approaches to communityware and groupware have been evolved independently in the past.

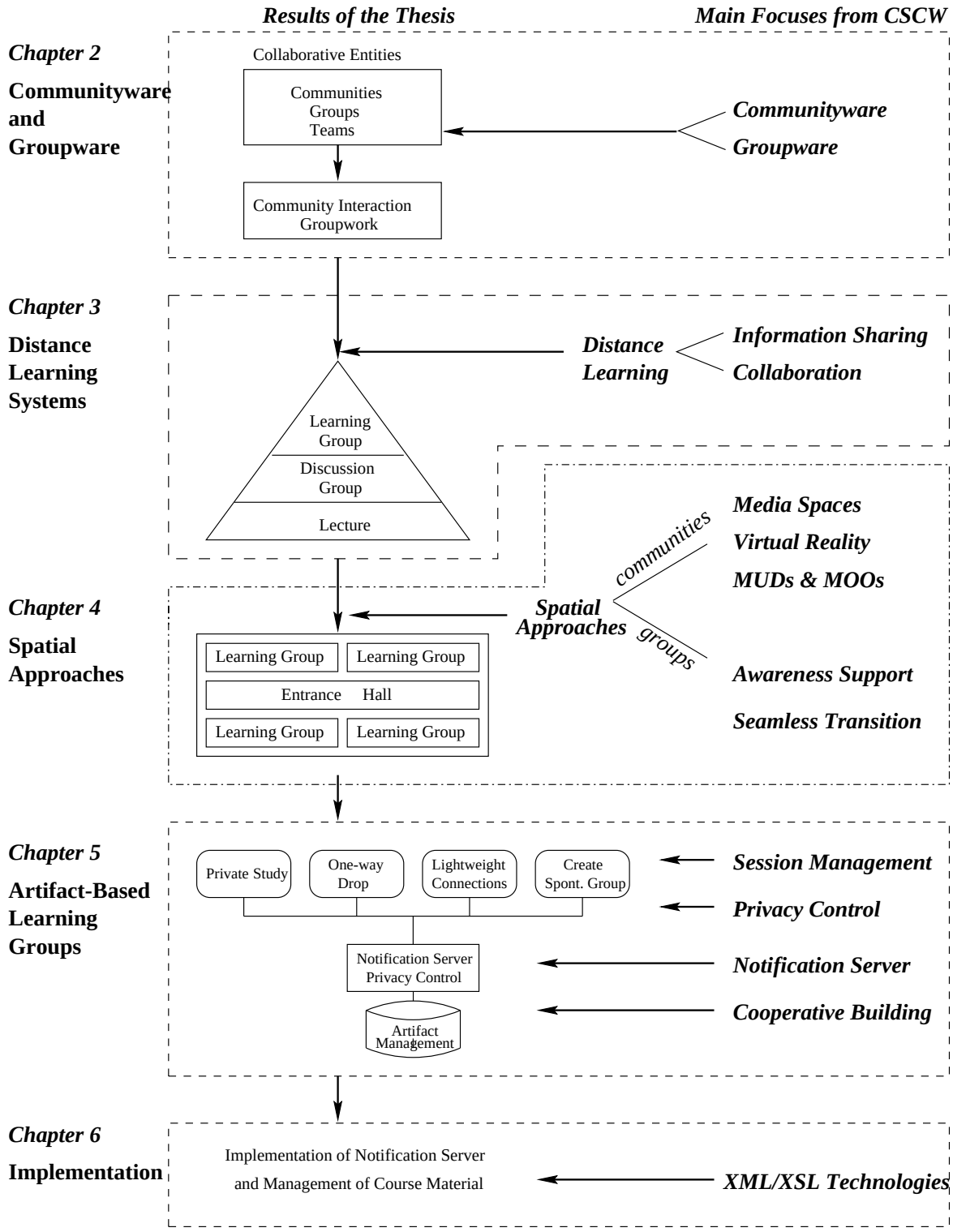


Figure 1.1: Overview of thesis

The need of supporting the pyramid model and the study of spatial approaches make it necessary and possible to find a way to develop a spatial framework for educational settings which supports both community interaction and groupwork. The spatial framework in Lecture 2000 is comprised of an entrance hall, a discussion group, a tutor studio and several learning groups.

Instead of covering all components present in the spatial framework, **chapter 5** puts great emphasis on the support of learning groups. After briefly introducing learning groups from different perspectives, I will first summarize seven key components which play key roles in facilitating interaction and collaboration within learning groups. Among them, *artifact management* is responsible for managing artifacts like *rooms*, *workplaces*, *containers* and *documents*; *private study* focuses on how to support students' learning activities; *creation of spontaneous groups* enables community members who have developed a closer relationship to create their own learning groups; *one-way drop* makes it possible for a user to leave a brief message in group rooms and then to facilitate asynchronous collaboration; *lightweight connection* realizes synchronous collaboration implicitly; *privacy control* combines spatial approaches and role-based access control models to avoid intrusiveness caused by a community member entering a group room abruptly and to protect users' privacy; finally, *history management* addresses issues concerning investigation of session information and version management in hypermedia.

These seven key components are implemented on the basis of *an artifact model*, which offers a conceptual model for defining various artifacts within learning groups, and *a notification server*, which is concerned with the provision of both synchronous and asynchronous awareness for collaboration. Both of them will be discussed in great detail.

Chapter 6 primarily focus on how to implement the notification services needed by components defined in the last chapter and management of learning materials. To simplify the implementation of the notification services, we choose to extend NSTP which is a notification server mainly supporting synchronous applications. Through the introduction of a persistent state repository, both synchronous and asynchronous awareness are supported by the notification server.

XML/XSL will play a key role in managing electronic learning materials in Lecture 2000. Lecture scripts in Lecture 2000 are XML-based documents. Combined with XSL processors, these XML documents can be converted into different file formats and support learning and teaching activities in different phases of a collaborative learning process.

Chapter 7 will give a overall summary about the whole work. The emphasis will be put on the review of the taxonomy of distance learning systems depicted in chapter 3. I will point out that Lecture 2000 provides an integrated learning environment, where both interaction and collaboration among students as well as between students and tutors have been supported.

2 Communityware and Groupware

After examining the key concepts of CSCW systems like interaction, coordination, collaboration and cooperation as well as communities, groups, and teams, I define the two terms "community interaction" and "groupwork" in this chapter, paving the way for further discussion in the thesis.

In terms of the definitions of community interaction and groupwork, I focus on the description of the burgeoning research area "communityware", followed by a taxonomy of communityware systems. Subsequently, a brief examination of groupware is presented.

Finally, this chapter concludes that communityware and groupware should be combined to support community interaction and groupwork in educational settings, eliminating the problems of separate development in the past.

2.1 Social Networks

The dramatic technological advances substantially changed the way people interact and work. One of these outstanding advances is the emergence and development of telecommunication and network technologies, making it possible for people to exchange information and get themselves involved in collaborative work despite being separated by space and time.

In 1794, the first telegraph line, which realized the concept of optical telegraph by mounting visible wing (called "semaphore") in regular distance atop of buildings or specially-constructed towers, was established between Paris and Lille in France [Reichw98]. The wing positions represented symbols of an alphabet for sending enciphered messages. In the current world, most computers are connected to the Internet, which, as the most important result of telecommunication and network development, has drawn special attention of people with different backgrounds and given a tremendous impetus to the development of information technology. In the beginning of its use, the Internet was primarily used to distribute information between people. However, much work has been done in the past to seek ways to enable people dispersed in space and time to work together. This research area is called *Computer Supported Cooperative Work* (CSCW), which is originally intended to support people working collaboratively on well-structured tasks like conducting a project together. The term *groupwork* is used to refer to this kind of common work.

While professionals continue their efforts to overcome various barriers in supporting groupwork, which has been challenged by diverse difficulties, new applications stemming from

the research area are sprouting up in Internet rapidly. As opposed to traditional systems supporting groupwork, these systems focus on enabling various *community interactions* in the Internet, like expressing some kind of feelings, stimulating social contacts etc. An investigation into how the Internet has an impact on communities, conducted by Microsoft through setting up a community street in the Northern London and published in New York Times on Oct. 13, 1998, revealed that even the lives of ordinary families and neighborhoods may be considerably affected by the Internet and its relevant technologies such as Email, Bulletin Board, NetNews and Chat Systems etc.

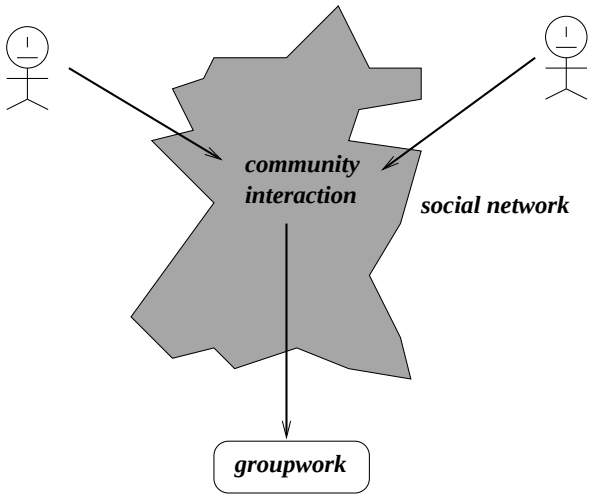


Figure 2.1: Social activities consist of community interaction and groupwork

At present, many people like to refer to the Internet as *social networks*. They argue that when computer networks link people as well as machines, they become social networks, or the basic building blocks of societies. This suggests that an era of supporting social activities through the Internet has emerged. Both the support for "traditional" groupwork and for community interaction have become a substantial part of the social networks around us. Furthermore, people involved in groupwork are no longer confined to a single location or organization, the social networks enables groupwork across many organizations (see Figure 2.1), which will be illustrated by our work for the project *Lecture 2000*.

2.2 Community Interaction and Groupwork

2.2.1 Interaction and Collaboration

The premise for this work, which is aimed at supporting both community interaction and groupwork in educational settings, is first of all the understanding of interaction and collaboration.

To distinguish between *interaction* and *collaboration*, let us have a look at the term *cooperative work*. There is no doubt that CSCW supports cooperative work. Unfortunately, for

this term, we can find many distinct definitions in literature. For instance, Ehn thinks that *all work* is essentially cooperative, in the sense that it depends upon others for its successful performance [Bodker88]. According to this view, it seems to imply that it's absolutely dispensable to add the term *cooperative* to that of *work*. At another extreme, Sorgaard has a very specific set of criteria for what would count as cooperative work [Bannon89]. Both views suggest that *cooperation* take place at different levels.

Bair suggests that the degree of cooperative needs can be represented by four levels of human communication (see figure 2.2) . Briefly, the characteristics of each level of human communication are as follows [Bair89, Koch97]:

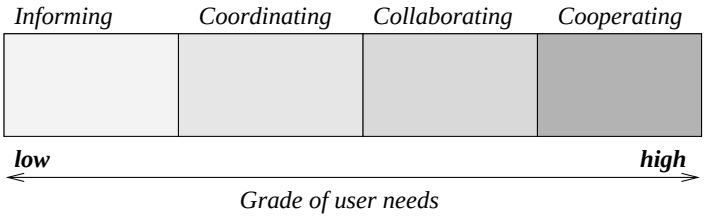


Figure 2.2: Different levels of communication

Informing (no acquaintance) Information is communicated anonymously, through mass media or local resources. There is often only a one way communication channel.

Coordinating (some acquaintance) The emphasis of coordination is primarily put on the synchronization of the shared use of information and other resources. It is not necessary for participants to pursue a common goal although common interests and organizational affiliations are likely. People communicate to share information and resources. According to Malone, "*Coordination is managing dependences between activities.*" [Malone94]

Collaborating (working relationship) Participants are engaged in the same work process. However, the interaction between participants is loose. Although there may be common output, each individual is evaluated independently. At the end of the collaborative process, all partial results serve as the result of the collaboration.

Cooperating (goals are common) People are involved in a common work process. Sublimation of individual goals is in favor of the team's goals. The team is evaluated as a whole, and competition is minimized. Decisions are made by group consensus, especially for collective action.

As Bair suggested that the degree of communication is essentially a continuum, I will use two terms to denote relatively lower and higher communication level and user needs of this spectrum. One is the term *interaction* which is referred to as the lower levels of human communication. Of course, it also represents relatively lower user needs. Another is the term *collaboration* used to denote higher levels of human communication and user needs.

Collaboration is the task-focused communication of teams and groups towards a common goal. The relationship between the different communication levels defined by Bair and the terms used throughout the thesis is illustrated by Figure 2.3.

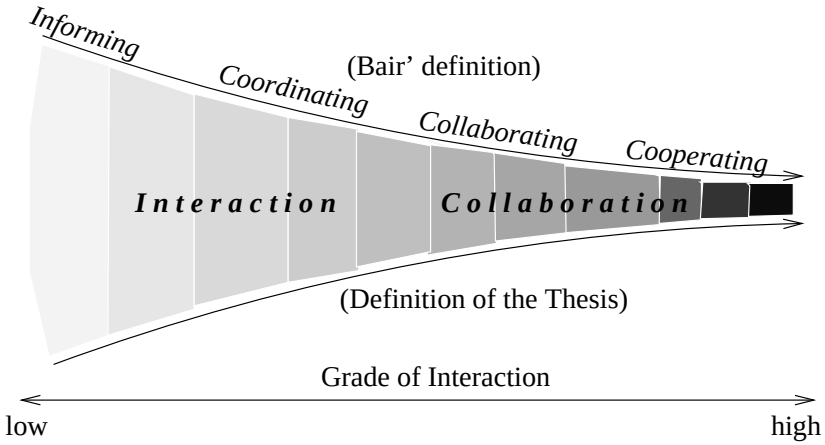


Figure 2.3: Abstraction of interaction and collaboration

Definition 2.1 (Interaction)
Interaction refers to the lower levels of human communication. Participants in an interaction process are loosely or temporarily organized together and are not expected to accomplish a common task.

Definition 2.2 (Collaboration)
Collaboration denotes the higher levels of human communication. People engaged in a collaboration process are tightly coupled and work for a common task and toward a common goal.

It is noteworthy that different communication media[1] support different user needs of cooperation. For example, if a team used electronic messaging exclusively, it would be difficult that they would ever achieve the "cooperation level". The higher the level of required user needs, the higher the mutual understanding among participants, which requires richer interaction.

2.2.2 Communities, Groups and Teams

In general, we can distinguish between three collaborative entities, namely *communities*, *groups* and *teams*.

[1]Bair classified communication media into two major categories: face-to-face and substitutions for face-to-face. Face-to-face meeting is the "first medium" of human interaction. Messaging systems, file sharing and computer conferencing etc are substitutes for face-to-face communication which can be divided into real-time interaction among people and asynchronous communication. A detailed taxonomy of communication media can be found in [Bair89]

Communities

Even in the early 1950s, Hillery found that there were at least 94 definitions for the term "community" [Ishida98]. For the sake of brevity, let us focus on only two of them in an effort to characterize this concept.

The 19th-century German sociologist Ferdinand Tonnies defined *"gemeinschaft"*, or *community*, as

> *"small, geographically distinct, kinship-interwoven groupings characterized by intimate, overlapping, and stable relationships."*

The term of "community" has also a detailed definition in Webster's English Dictionary [webste96]:

> *"1)social group of any size whose members reside in a specific locality, share government, and often have a cultural and historical heritage; 2) a social, religious, occupational, or other group sharing common characteristics or interests and perceiving itself as distinct in some respect from the larger society within which it exists."*

These definitions have some shared characteristics despite the small disparities between them:

- **Locality**

  Firstly, both the definitions highlight the importance of *locality* implied by the term *community*. Historically, community has been defined in terms of geographical boundaries. In fact, communities have long been confined to being in some localities because of, for example, the deficiencies of necessary transportation and communication facilities. However, technological development, in particular those of network systems in recent years, has diminished the role of locality in communities. In the modern society, people form a community independently of where they live and work. Mynatt has addressed such a diminished role of locality for communities as follows:

  > *"Certain technologies reshape notions of physical space and proximity, A car moves faster than a person walking and hence redefines the idea of how far a person can get easily in 10 minutes – which redefines what might be considered local. 'spanning technologies' such as bikes, radio, TV, telephones and telegraphs, and autos have each, in turn, led to reconceptualization of time and space, and hence of what it means to be local and the possibilities for human interconnection." [Mynatt98]*

  With the help of the rapid development of these "spanning technologies", more communities are emerging from great distances. As stated by Hiltz, "*communities are now defined in terms of social relationships, rather than in terms of space*" [Hiltz97].

- **Social Interaction**

  Secondly, social interaction is a key concept of communities. Without certain social interaction, a group of people who even live or work in a locality can not be considered as community. Suppose that a group of students always eat at the same mess hall but have no contact with each other, it makes no sense in this case to think of this group as a community. Within a community, there are always some kind of social interaction occurring even though no intimate personal contacts happen between its members.

- **Shared Characteristics and Interests**

  Finally, the most important factors to bind people into a community are the common characteristics and interests existing between community members. Shared interests often serve as the basis for the formation of community. If a group of people share common characteristics and interests, though having no social interactions temporarily and being separated by geographical distances, chances still exist to get them participating in a community. In fact, common characteristics and interests foster the formation and evolution of communities.

Given the fact that traditional factors of communities are becoming less important, communities can be seen as a set of people who do not necessarily know each other or interact on a personal basis but share common characteristics and interests. Furthermore, community members keep some kind of social interaction. Communities may be the starting point for identifying a set of people one could interact with, e.g. to find some help for solving problems or to share experiences [Schlic98].

Definition 2.3 (Community)
A community is a group of people who share common characteristics and interests and keep some of loose social interactions, but do not necessarily know each other or maintain personal contacts.

Groups and Teams

Certainly, groups and teams can arise from communities. According to the definition of community, community members do not necessarily know each other. Then it makes sense to apply a different term for possible groups whose members do know each other. Schlichter et al refer to such groupings as *groups* [Schlic98]. Compared to communities, building contacts within groups for cooperation on a project or a task is easier because there is already a certain level of mutual knowledge and understanding.

The most advanced form of a community is a *team*. Team members know each other and are working together to achieve a common goal. This description corresponds with a definition found in Webster's Dictionary: "*team: a number of persons associated in some joint action*" [webste96].

According to Teufel [Teufel95a], we can give the following definitions:

Definition 2.4 (Group)
A group emerges when two or several persons interact and affect each other.

Definition 2.5 (Workgroup)
a workgroup is a group with common task.

Definition 2.6 (Team)
A team is a workgroup whose members are willing to reach a common goal.

Teams differ from groups in that groups do not necessarily cooperate. Thus, the interaction in groups is loose because there is no shared task or common goal. Examples of groups are a group of friends or the members of a research institute.

2.2.3 Community Interaction and Groupwork

In the preceding sections, I have examined two concepts: levels of communication and collaborative entities which lead to the definition of two terms: *community interaction* and *groupwork*.

Apparently, there are some inherent correlation between different levels of communications and collaborative entities. Lower levels of communication usually occur in communities while higher levels of communication are indispensable for groups and teams to accomplish the common task and to achieve common goals (see figure 2.4) .

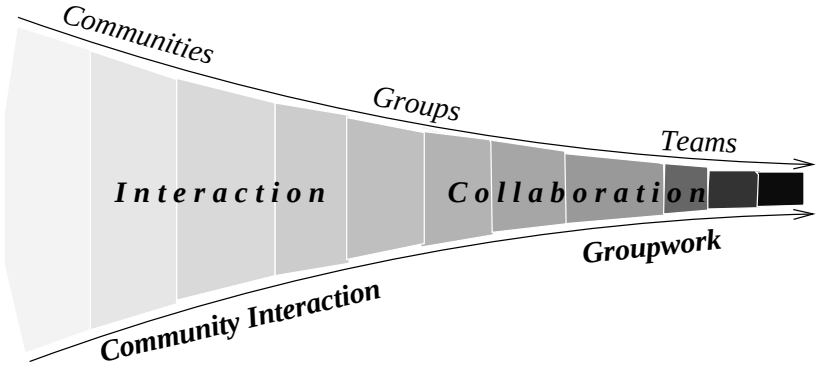


Figure 2.4: The relationship between collaborative entities and interaction

Definition 2.7 (Community Interaction)
Community interaction is the sum of activities between community members expressing their concerns, feelings, interests etc. These activities, which do not necessarily get community members involved in a common work or aiming at common goal, may involve:

- *emotional support and sociability, which reflect a sense of belonging and participation*

- *exchanges of information and services of interest to members, or common-interest discussions etc.*

Compared with the definition of community interaction, the definition of the term *groupwork* is not so straightforward but somewhat confusing in literature. As Bannon *et al* reported, distinctions between such terms as cooperative work, collaborative work, collective work, and group work are not well established in the CSCW community [Bannon89]. Some people argue that CSCW systems primarily address the support of cooperative work or collaborative work while others use the term groupwork to represent the same meaning. On the other hand, it is recognized that cooperative work can occur within workgroups or teams [Teufel95b]. Throughout this thesis, I use the term *groupwork* to denote cooperative work occurring either in groups or in teams.

Definition 2.8 (Groupwork)
"Groupwork is the amount of activities relating to tasks done by a group of members with a common goal and to fulfill this goal." [Teufel95b]

Naturally, there is an overlap of community interaction and groupwork in the sense that either of them can include the other in most cases, largely because there are no clear boundaries between interaction, coordination, collaboration and cooperation, which can be depicted by Figure 2.4.

2.3 Communityware

2.3.1 Virtual Communities

In recent years, *virtual* is one of the most popular words appearing in computer literature. Virtuality has painted a picture of mystical and spectacular future of computer systems. At the heel of *Virtual Memory*, developed in 1960s to "expand" computer memory to accommodate large programs, people seem to be overwhelmed by a variety of something virtual like *Virtual Reality, Virtual Organization, Virtual University* etc. To exactly define the term *Virtual Community*, one must first of all make it clear what the word *virtual* or *virtuality* means.

Turoff defined *virtuality* as "*the property of computer system with the potential for enabling a virtual system (operating inside the computer) to become a real system by encouraging the real world to behave according to the template dictated by the virtual system*" [Turoff97]. Compared with the somewhat tortuous definition, the following provides a more direct and impressive explanation of virtuality:

> *"Virtual memory enables programmers to write code referring to storage not actually available in the computer. Virtual reality enables a user to experience visual, auditory, and tactile sensations that do not exist in the normal human environment.... Virtual offices allow employees to operate in dynamically changing work environments."[Mowsho97]*

No matter what this word means, virtuality certainly enables computer systems to create environments in which people feel that they operate, interact or work as in a world dictated by the concept itself (e.g. virtual memory).

Perhaps, Hill's understanding about a virtual community can facilitate my discussion [Hill98]:

> *"The term community means 'a group of people who share characteristics and interact'. The term virtual means 'in essence or effect only'. Thus, by virtual community we mean 'a group of people who share characteristics and interact in essence or effect only'. In other words, people in virtual community influence each other as though they interacted but they did not interact."*

Definition 2.9 (Virtual Community)
Virtual Community is a group of people who share common interests and characteristics but are distributed in space and time. This group of people are linked by computer systems.

Some definitions of virtual community still cling to the importance of locality in communities. For example, Mynatt defines *network communities* as:

> *"Network communities are robust and persistent communities based on a sense of locality that spans both the virtual and physical worlds of their users."*

2.3.2 Characteristics of Virtual Communities

Virtual communities distinguish themselves from other communities and groups in some ways:

- **Technologically mediated**

  First, technology has played a key role in bringing up and sustaining new forms of communities by dramatically diminishing the importance of geographical proximity. Apparently, network technologies help overcome spatial distance to achieve social cohesion. Historically, media such as radio, television, and print media, as well as transportation, messenger, telephone, or telegraph are used to traverse these distances. Virtual communities, by contrast, rely on relatively immediate computer network communications to span these distances.

  The following technological factors have a great impact on the forming and evolving of virtual communities.

  - *Telecommunication*

    At present, it is widely recognized that the improved bandwidth of local and wide area networks may help to reduce the space dependencies in traditional communities. The combination of wireless communication, cable communication, and satellite service will increase the access speed to the Internet, which will stimulate the formation and growth of virtual communities across great distances and attract more people to be engaged in virtual communities.

- *Computer-Supported Cooperative Work*

  The research in Computer-Supported Cooperative Work has built a relatively solid basis for supporting virtual communities. A lot of software systems originally developed to facilitate well-organized groups have been widely used to support the communication between community members.

- *Information Management*

  A prerequisitive to community interaction is to identify and to select the relevant individuals of a community. In order to support this process, attributes describing the individuals are required. This information can be obtained along different ways, many of which are facilitated by information systems containing information about people's skills, competencies and interests. Certainly, the development of Web-based Information Systems [Isakow98] is expected to play a more important role in facilitating virtual communities.

- *Multimedia Technology*

  The widespread use of multimedia technologies, for example, audio/video conferencing, enables community members to communicate more efficiently and to create an illusion of talking face-to-face. This helps to draw more people to participate in virtual communities.

- *Agent Systems*

  Finally, an extensive study of applying agent systems to virtual communities has been conducted in the past few years. Later, we will make some brief introduction to agent-based community systems. Generally, agent systems have been used for information filtering and finding potential community members etc.

- **Larger, more dispersed in space and time and more heterogeneous**

  The technology development has made virtual communities larger, more dispersed in space and time. The ease of contacting other people promotes the growth of relatively large virtual communities and the evolution of densely knit relationships among community members. On the other hand, virtual communities are also more dynamic. Community members have more heterogeneous social characteristics [Hiltz97].

- **Multiple interaction styles**

  Mynatt *et al* point out that network communities (virtual communities) enable participants to communicate in different ways [Mynatt98]. Virtual communities have the ability to engage in many different kinds of interaction, such as "face-to-face" conversation, "hallway" meetings and greetings, or peripheral or ambient awareness of "distant" noise or conversation. Interaction in virtual communities is not tightly tied to a particular task or channel, but allows for different kinds of participation, informal, formal, or serendipitous.

2.3.3 Communityware

As mentioned before, computer networks become *social networks* when they link people as well as machines. As the largest network of the world, the Internet, swiftly evolving from an infrastructure primarily used to distributing information into a social network encouraging social interaction between people, is fostering more and more *communityware* in the past few years. In general, people refer to communityware as computer systems which sustain the formation of virtual communities and further support community interactions. So far, there is, however, no strict and unanimous definition about the new research area. Even the name denoting this research area is still pending.

Recently, more people are inclined to use the term "communityware" to refer to systems supporting the process of organizing people who are willing to share some mutual understanding and experiences. Not only dedicated to organizing people who share some common interests and preferences, communityware is also used by its other proponents to recognize the importance of *emotional support and sociability* between networked people.

Ishida defined "communityware" as *the methodologies and tools for creating, maintaining, and evolving social interaction in communities* [Ishida98]. Here, the "social interaction in communities" amounts to *community interaction* defined in section 2.2.3. Drawing on my understanding of virtual communities and community interaction, I define the term *communityware* as follows:

Definition 2.10 (Communityware)
Communityware represents the methodologies and tools for creating, maintaining, and evolving community interaction. Typically, a communityware system is an environment on the Internet that hosts virtual communities and helps community members reap the benefits of establishing online connections. In general, communityware is primarily concerned with the following issues:

- *facilitating contacts of a wide variety of people,*
- *supporting emotion and information exchanges, and*
- *fostering group formation.*

Recalling the example about an electronical community street in London (see section 2.1), we can find the enormous potentials of networked communities in facilitating contacts of people, supporting emotional and information exchanges like gardening tips, recommending a decent plumber or electrician, finding a trustworthy babysitter, or swapping notes on local shops and restaurants. Common-interest discussions included debating on a proposed municipal plan that would introduce controlled parking on nearby streets, which has created concerns that their street would be flooded by cars.

2.3.4 Taxonomy of Communityware

So far, there is no publicly recognized taxonomies about communityware. In this section, I try to classify communityware systems in two ways: *application-level taxonomy* and *function-based taxonomy*.

Application-Level Taxonomy

Recommender Systems Recommender systems can be thought of as communityware because they are used to find appropriate persons to contact, which is one of tasks deemed to be tackled by communityware. To find somebody, it is often necessary to make choices without sufficient personal experience of the alternatives. In everyday life, we rely on recommendations from other people either by word of mouth, recommendation letters, movie or book reviews in newspapers, or general surveys. Recommender systems assist and augment this natural social process, helping people find relevant information as well as potential partners.

Recommender systems, also called *collaborative filters* coined by the developers of the first recommender system Tapestry [Goldbe92], can be divided into *content-based recommendation* and *collaborative recommendation*. In content-based recommendation one tries to recommend items similar to those a given user has liked in the past, whereas in collaborative recommendation one identifies users whose tasks are similar to those of the given user and recommends items they have liked [Balaba97].

Content-based systems are based on building a user profile. These systems attempt to extract patterns from the observed behavior of the user to predict which items would be selected or rejected. Examples of this form of filtering include LyricTime [Loeb92] and INFOSCOPE [Fische91]. Maltz *et al* reported that these systems, however, suffer from a "cold-start" problem. New users start off with nothing in their profile and must train a profile from scratch. Even with a default profile at the start, there is still a training period before the profile accurately reflects the user's preferences. Moreover, the user's searches can become limited by the profile.

Collaborative-based recommendation systems give to users others' prior experiences with an information source. Tapestry was the first system to support collaborative (-based) filtering in that it allows its users to annotate the documents they read. Other Tapestry users can then retrieve documents based not only on the content of the documents themselves, but also on what other users have said about them. Tapestry provides free text annotations as well as explicit "likeit" and "hateit" annotations so users can pass judgments on the documents which they read [Maltz95]. The combination of high volume and personal taste made Usenet news a promising candidate for collaborative filtering [Konsta97]. In GroupLens, a collaborative filtering system for Usenet news, communities of users rank the articles they read on a numerical scale. The system then finds correlations between the ratings different users have given the articles.

Agent Systems Autonomous agents are computational systems that inhabit some complex dynamic environment, sense and act autonomously in this environment, and by doing so realize a set of goals or tasks for which they are designed [Maes95]. Further, agents can take many different forms, depending on the nature of the environment they inhabit. So-called knowbots, software agents or interface agents which inhabit the digital world of computers and the Internet are those of great interest to communityware developers. Agent systems can leverage the development of support for community interaction in many ways. At present, people give special attention to the thriving field.

People can find diverse application potentials of agents in many fields of communityware, including for recommendation systems. For instance, Maes proposes to develop the infrastructure which will allow the students to, without much efforts, locate an expert who is available to help the student remotely at that moment in time. This infrastructure will help build *"an electronic market for knowledge, and to have agents that can match up students and experts and make deals for an on-the-fly assistance relationship"* [2].

To participate in this knowledge marketplace, a person can create selling agents that know what expertise that person has and when that person is available to help someone else. Someone in need of help can create a buying agent that knows what kinds of expertise that person needs and by when. The buying agent may also have information on how to compare different people offering the expertise using criteria such as: price charged, reputation of the expert, level of expertise and availability. The market would automatically match up these different buying and selling agents. Once a match has been made, other media are used to implement the tutoring relationship.

Email and Discussion Lists Email and discussion list are the oldest and most popular form of interaction in the Internet. Email allows an individual to send a message directly to another person. However, email is often used to go beyond an one-to-one interaction. In an email discussion list a message sent to a group address is then copied and sent to all mail addresses on a list. When people direct a series of messages and responses to the list, a group discussion can develop.

Usenet and Bulletin Board Systems Bulletin Board Systems (BBSs) are another form of asynchronous communication as email discussion list that, however, refine it in a number of ways. Most BBSs allow participants to create topical groups in which a series of messages, similar to email messages, can be bound together on after another. Moreover, email is a "push" media [Kolloc98], messages are sent to people without them necessarily doing anything. In contrast, BBSs are "pulling" media, people must select groups and messages they want to read and actively request them.

The Usenet is the largest conferencing system in the world and has a unique form of social organization. The Usenet consists of several millions of helpful people all over the world. They use a variety of computer networks and software tools to communicate with each other.

[2] http://pattie.www.media.mit.edu/people/pattie/

The Usenet is divided into many thousands of newsgroups, which means groups of people who share the interest in a certain range of topics. Whittaker et al report that in 1998 there are over 17,000 newsgroups and 3 million users in the Usenet [Whitta98]. A new site joins the Usenet simply by finding any existing site that is willing to pass along a copy of the collection of messages it receives.

World Wide Web While the World Wide Web has been very popular for some time, it's only recently that it has also become a site for interaction. In its original incarnation, the Web served as a powerful way of distributing, accessing and linking documents, and helping people find suitable and potential persons to interact with and furthermore developing a collaborative ties. Web sites can now support both synchronous and asynchronous communication. Through the use of various software tools, web sites can host asynchronous discussion groups as well as real-time text chat.

Text Chat Systems Text chat differs from email and Bulletin Boards in that it supports synchronous communication by which a number of people can chat in real time by sending lines of text to one another. With a large number of Web sites fostering chat like yahoo, chat is evolving to one of the most popular forms of interaction on the Internet, and accounts for a considerable proportion of the revenue of the commercial online providers such as America Online. Most chat systems support a great number of "channels" dedicated to a vast array of subjects and interests. In some systems, people can set up private "rooms" in keeping their participants secluded from others.

Media Space Systems Media spaces which use integrated video, audio and computers to allow geographically distributed individuals and groups to work together can also be viewed as a community systems, because the support for peripheral awareness was perhaps their most powerful facet [Bly93], and such awareness is mainly used to support informal communication among people. The best known examples of media spaces are the Cruiser [Root88] and RAVE [Gaver92b] systems.

MUDs and MOOs With respect to the support for community interaction, spatial metaphors are used for informal communication in communities. In the past, people have paid special attention to MUDs and MOOs. MUDs and MOOs are a kind of Internet-accessible virtual world with their own geography, characters, and objects of various kinds [Curtis92]. In other words, they are computationally-based environments, providing access to a persistent, online "world" which can generate a sense of a known virtual place that can be inhabited and shaped by an emerging community. Typical examples include Pueblo [ODay96], a MOO supporting an elementary school-centered learning community, and Jupiter [Curtis95], a hybrid MUD and media space which was developed at Xerox PARC and is available to members of Xerox research and development community. In these systems, when people join the community, they typically start by creating a character and building a home for themselves. With simple

commands, participants can move from one locale to another, talk to other people, manipulate objects they encounter in each place, and extend the world by creating and describing new places and objects with behavior.

Apart from Media Spaces and MUDs and MOOs, there are other systems which adopt space-based approaches to support community interaction. For example, FreeWalk [Nakani96] uses a common three-dimensional space just like a real life park or lobby to enable people to experience accidental encounters in a network. In fact, due to the inherent feature of locality in communities, spatial approaches are of particular importance to develop systems supporting community interaction.

Function-Based Taxonomy

Alternatively, one can examine communityware systems in light of what communityware is supposed to support. Ishida painted a somewhat detailed picture of functions supported by communityware [Ishida98]:

- knowing each other
- sharing preferences and knowledge
- generating consensus
- supporting everyday life
- assisting social events

Here I adopt a more direct and easily-understood classification of communityware systems. According to Schlichter *et al* [Schlic98], communityware can be classified into three categories: systems used to find partners from a wide variety of people to interact; systems supporting contact between partners; and systems fostering the common understanding between partners.

Find someone to interact with In contrast to groupware systems supporting well-organized people to work together on a common task, communityware systems usually do not target any specific groups of people. More specifically, people supported by communityware have more distinct backgrounds and usually do not know each other personally. Thus, communityware first of all has to tackle the issue of helping people find potential partners to interact with. To pursue this goal, these systems must be capable of:

- managing and providing information about potential partners

  As introduced before, WWW, Database Retrieval Systems, Recommender Systems are typical computer systems which are used to distribute, access, filter and retrieve informations about organizations as well as individuals.

The application of XML/XSL (Extensible Markup Language/Extensible Stylesheet Language[3]) technologies to Web-based information systems will make the management and searching of electronical information over the Internet more powerful and flexible. XML specifies neither semantics nor a tag set. In fact, XML is really a meta-language for describing markup languages. With XML, individuals could be easily categorized in a standard way by research area, title, specific skill or other criteria. Agents can then search these identified computer professionals in a consistent way.

- reaching or being found by a wide variety of people

  One of the key features of virtual communities is that their members may be dispersed across the whole Internet. A system which is supposed to help find partners must be capable of reaching people in far-away places or being popular enough to draw a wide variety of people to join in. We know that some agent systems, called softbots by Mueller [Muelle97], can move through the Internet to find individuals with whom their designer are supposed to interact. On the other hand, the Usenet and Bulletin Board involves tens of thousands of people to view news organized according to shared interests. Meanwhile, some MUDs and MOOs are also growing rapidly to serve the same function to locate desired partners for community members.

- providing a lenient, generous interactive environment for unfamiliar faces

  Inherent with communities is that their members do not know each other at least personally, which determines that the interactive environment for communities must be "lenient" and "generous". People can, for example, talk with each other anonymously. Chat Systems, MUDs and MOOs can be classified into this group of systems.

With the help of these systems, community members can find people with some shared preferences and interests or characteristics, which paves a way for their further contact.

Make contact with the selected people The next step is to contact individuals found in the previous step. Similarly, a wide variety of systems such as Email, Chat, MOOs and MUDs etc are available to enable people to build and maintain the contacts. Typically, we divide them into synchronous and asynchronous. Later in this thesis, we will come back to this topic in more detail.

Build a common understanding Finally, the very shallow understanding and confidence between them initiated by the first two steps can be promoted by further and more intimate contacts. Some systems can support community members to extend this relationship. For example, Chat supporting private "rooms", Mail, Teleconferencing systems are of great interest to this step.

Table 2.1 summarizes various systems in terms of their underlying functions. Note that many systems appear more than once in the table because they can support distinct issues pertinent

[3] http://www.w3.org/XML/

Find someone to collaborate with	World Wide Web NetNews, Bulletin Boards Recommender Systems Agent Systems MUDs and MOOs Email Chat Tools
Make contact with the selected people	Email Chat Tools MUDs and MOOs
Build a common understanding	Chat Tools Teleconferencing Systems Email MUDs and MOOs Media Spaces

Table 2.1: Function-based taxonomy of communityware

to community interaction. In particular, some space-based approaches such as MUDs and MOOs draw more and more attention in supporting community interaction as a whole.

2.4 Groupware

2.4.1 Definitions

Groupware is supposed to support "cooperative work" in groups and teams, which has been sketched in previous sections. Bannon and Schmidt investigated diverse definitions of the term "cooperative work" in more detail [Bannon89]. They contributed to the discussion as well by their own definition of cooperative work:

> *"In sum, the term 'cooperative work' is the general and neutral designation of multiple persons working together to produce a product or service. It does not imply specific forms of interaction or organization such as comradely feelings, equality of status, formation of a distinct group identity. " [Bannon89, P. 362]*

The definition emphasizes two aspects of cooperative work:

- several persons work together
- the work must lead to a product or service

Koch argued that this definition by Bannon and Schmidt can not identify the characteristics of cooperative work because every type of work can be seen as cooperative work with enough explanation [Koch97] in the sense that conducting a task always involve other persons. Instead, he defined the term as follows:

Definition 2.11 (Cooperative Work)
Cooperative work is the work situation in which several persons work together in order to achieve a (common) result (product or service). The special organization form or structure of the group can be arbitrary.

The following features can characterize these work situations:

- *at least consensus of partial goals*
- *possibility of mutual adaptation of action plans*
- *group members do not necessarily have equal rights, however, the group structure is usually lean using a flat organizational hierarchy.*

In my opinion, the substantial difference between the definitions by Bannon and Koch is how to understand the phrase "work together". Many products and services are results of a work which indirectly involve people who have never contacted each other or talked together about the work. In fact, they are results of distinct and unrelated work processes. Under these circumstances, it does not make sense to speak of a cooperative relationship between them. The definition by Koch excludes these possibilities from "cooperative work" by emphasizing "consensus of partial goals" and "mutual adaptation of action plans", which means that this group of people *really work together*.

Definition 2.12 (Groupware after Ellis)
Groupware are computer-based systems that support groups of people engaged in a common task (or goal) and that provide an interface to a shared environment [Ellis91].

The notions of *common task* and *shared environment* are crucial to this definition, which excludes multiuser systems, such as time-sharing systems, whose users may not share a common task. In particular, *shared environment* precludes all situations which could be thought of as cooperative work by the definition of Bannon and eliminated by Koch. The shared environment confines participants of a cooperative work to being involved in the same work process.

2.4.2 Taxonomy of Groupware

In contrast to communityware, there are well-established criteria for classifying groupware systems. As Figure 2.5 suggests [Teufel95b, P. 11], communication serves as the basis of coordination and cooperation. In terms of characteristics of communication technologies like applied media types, distribution of space and time, number of communication partners

as well as communication type, CSCW applications can be classified according to different models, among of which are the time-space-matrix model by Johansen [Johans88] and the function-based taxonomy by Teufel [Teufel95b, P. 27].

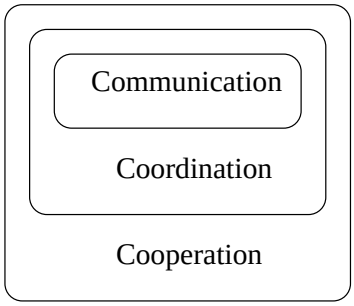


Figure 2.5: Communication, coordination and cooperation

Function-Based Taxonomy

According to supported functions, groupware systems can be divided into four categories:

Communication System Communication systems enable people to explicitly exchange information. Typical examples of communication systems include Email, Bulletin Boards and Video/Audio Conferencing Systems.

Shared Information Space In this class, we find systems which allow people to maintain and manipulate shared information. The information exchange in this case is implicit. Distributed Hypermedia Systems, certain Database Systems supporting multiuser operations belong to this category.

Workflow Management McCarthy and Bluestein argue that workflow management software is a proactive computer system which manages the flow of work among participants according to a defined procedure consisting of a number of tasks. It coordinates user and system participants, together with appropriate data resources which may be accessible directly by the system or off-line to achieve defined objectives by set deadlines. The coordination involves the passing of tasks from participant to participant in correct sequence ensuring that all fulfill their required contributions taking default actions when necessary [4].

Workgroup Computing Workgroup Computing Systems support the cooperation of persons who work in groups or teams and are supposed to solve tasks which have average or low grades of structures and repetition frequencies. Planning Systems, Multiuser Editors and Decision Supporting Systems can be considered as Workgroup Computing Systems.

[4] http://www11.informatik.tu-muenchen.de/lehre/lectures/vp-SS99/vorlesung-ss1999

Time-Space Taxonomy

As far as this thesis is concerned, the time-space taxonomy of groupware systems will be of great interest because it will provide a basis for the discussion of distance learning systems in the next chapter, where a slightly different time-space-matrix model for educational systems will be presented. Likewise, based on the same classification model, we will consider approaches in chapter 3 which demonstrate the capabilities of eliminating seams or gaps in CSCW systems.

	Same Time	**Different Times**
Same Place	face−to−face Interaction	asynchronous Interaction
Different Places	synchronous distributed interaction	asynchronous distributed interaction

Figure 2.6: Time-space taxonomy of groupware systems

Groupware can be designed to help a face-to-face group, or a group which is distributed across the Internet. Moreover, a groupware systems can be used to support a group in real-time or in non-real-time. These considerations of time and space suggest the four categories of groupware represented by the 2x2 matrix shown in Figure 2.6. Systems which conform to the classification will be discussed in section 4.3.3.

2.4.3 Artifact-Based Communication

Artifact is *an artificial product*, or *something made by human beings*. In computer systems, a document is a typical artifact. Certainly, an artifact can facilitate synchronous and asynchronous collaboration, but it can also enable direct and indirect collaboration.

Communication mechanisms are critical to coordination and thus essentially needed for collaborative work. To perform communication, participants typically use two fundamental human skills [Schlic97b]:

- direct communication with other participants and
- manipulation of shared artifacts.

Because manipulation of shared artifacts can be observed by other participants, it thus constitutes a form of indirect communication. These skills are often used in combination. For example, when communicating directly, participants often use references to shared artifacts

as an easy way of establishing referential identity [Miles93]. Similarly, when working with shared artifacts, participants often communicate directly with each other.

Definition 2.13 (Direct and indirect communication)
With direct communication a sender transfers the information specially to the receiver. With indirect communication the sender performs "actions" by which the receiver obtains knowledge without the sender explicitly taking care of the obtaining process of the receiver. [Koch97, P. 33]

The substantial difference between direct and indirect communication is whether the receiver (instead of the sender) actively does something in order to get the information. A sender always adopts some measures to send the information (for instance, *go* to post office to *send* a letter, or *save* a document in a shared database system). In an indirect communication, the information can not directly reach the receiver unless he actively seeks to get it. Figure 2.7 explains the differences between direct and indirect communication.

Definition 2.14 (Synchronous and Asynchronous Communication)
A synchronous communication occurs if group members communicate with each other at the same time, otherwise it is asynchronous communication [Borgho95, P. 106]

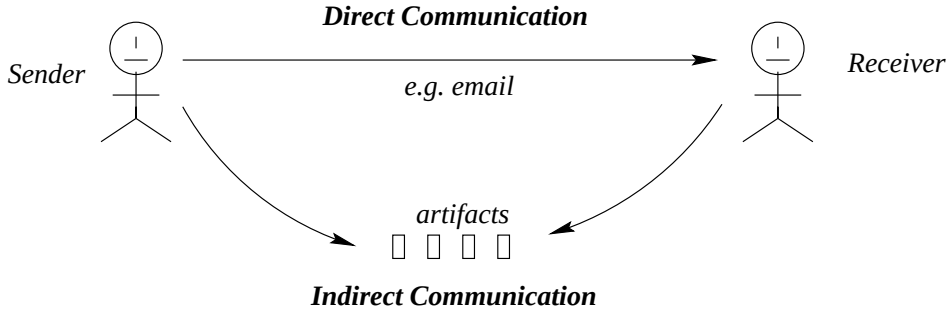


Figure 2.7: Indirect and direct communication.

It's straightforward that asynchronous communication can be direct (e.g. Email) or indirect (e.g. through shared documents in database systems etc). However, synchronous communication should be mostly direct (e.g. face-to-face meeting). Thus, groupware systems can be classified in terms of synchronous and asynchronous as well as direct and indirect communication (see figure 2.8) .

2.5 Integrated Support for Communities and Groups

So far, I have examined the definitions of communityware and groupware together with their respective classifications. While communityware is concerned with supporting earlier stages of collaborative work, groupware provides the electronic support for a group of people who

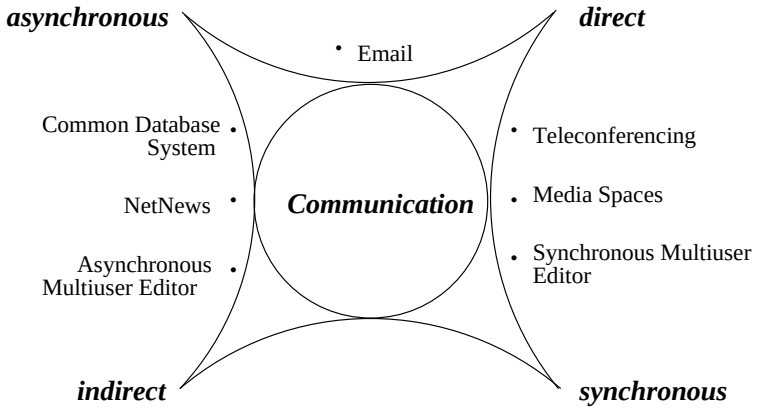


Figure 2.8: Classification of groupware systems in terms of synchronous and asynchronous, direct and indirect communication

are engaged in a common task, focusing on later stages of interaction and collaboration between people. Hiltz puts the substantial differences between communityware and groupware as follows:

> *"Computer Supported Cooperative Work relationships are generally narrowly focused and geared to accomplishing tasks through coordinating activities and providing information. Most studies of computer-supported social networks have looked at computer-supported cooperative work, or how people work together online despite being separated in space (usually) and in time (often). ... By contrast, virtual, or electronic, communities involve sociability, emotional support, and a sense of belonging as important ends in themselves, though they are often accompanied by exchanges of information and services." [Hiltz97]*

In spite of overlaps in their functions, communityware and groupware supporting distinct stages of social activities of people have evolved separately in the past few years. However, there is an increased need to have a seamless transition between communityware and groupware. Virtual communities and groups in educational settings which will be tackled in the next chapter, for example, necessitate an integrated support from both communityware and groupware. This work is intended to support them in an effort to overcome the separate development of communityware and groupware.

As suggested by Figure 2.9, the distinction between communities and groups becomes blurred in some situations, which means community interaction can lead to groupwork, and groupwork can also benefit and buttress up communities. In summary, the following two points constitute key issues to be addressed by the integrated support of community interaction and groupwork:

- group formation from a wide variety of people
- support for their groupwork

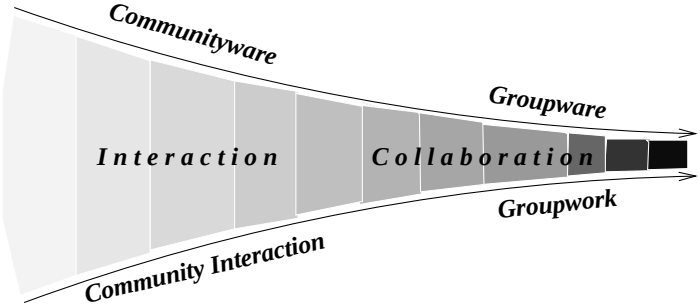


Figure 2.9: Integrated support for community interaction and groupwork

In the next chapter, I will have an insight into distance learning systems, taking into account community interaction and groupwork in educational settings. Based on the analysis, an approach to harness the capabilities of both communityware and groupware for supporting communities and groups will be worked out.

3 Virtual Communities and Groups in Educational Settings

This chapter is devoted to the discussion of distance learning systems for which two aspects are of particular importance: information sharing and collaboration. After examining current distance learning systems with respect to support for these two aspects, I am primarily concerned with the deficiency of their support for collaboration among students as well as between students and tutors. The main reasons for this deficiency are that these systems still poorly recognize various communication levels needed by participants involved in the learning and teaching processes, and that these systems lack effective mechanisms to organize and to facilitate these communications.

Based on the investigations, I will turn the attention to the project Lecture 2000 which aims at tackling these problems. Instead of depending on discrete synchronous and asynchronous mechanisms, Lecture 2000 attempts to harness the potentials of both communityware and groupware for the support of both interaction and collaborations in educational settings. Motivated by the goal, a model depicting communities, groups and teams in educational settings will be introduced.

3.1 Current State of Distance Learning

Recently[1], two news from the United States show the growing interest and prospect of distance learning systems. One is the "New Learning Anytime, Anywhere Initiative" which is intended to *"enhance and promote distance learning opportunities for all adult learners"*, demonstrating new high-quality usages of technology for distance learning in post-secondary education and training. Another one is a new federal legislation that establishes *"a five-year program to allow selected distance learning ventures to participate in major federal student aid programs"*, which puts an end to restriction of federal aid to students enrolled in conventional colleges and universities. Now, students in many newly created virtual classrooms can be just as lucky and benefit from federal aid. Both programs serve as some of the boldest of a number of recent experiments using high tech for higher education, a trend prompted at least in part by the emergence of the Internet and WWW.

However, the discord about distance learning will remain even though a lot of new advances

[1] The most information of this section comes from a series of special reports on Education and Cyberspace by New York Times dating from March 1998.

like the adoption of these measures can surely encourage its further development. Supporters of distance education, who are hoping these new programs can help boost the popularity of what they believe is a promising new educational enterprise, argue that distance education is important *"for those who simply cannot, for whatever reason, attend conventional classrooms, and it is a way to encourage and offer additional opportunities to people"*. Moreover, it will save future education costs, bringing college-level coursework, workforce training and degrees to a far wider range of people.

Still, some questions are raised about whether electronic distance learning is a sufficient replacement for the campus variety. Opponents argue that *"there is no substitute for the student actually witnessing a mind at work in a classroom."* Perhaps more importantly, critics say that they are concerned about the quality of distance education programs, and question whether they merit support. Some tutors who have experienced distance education complained that they must spend more time with students than those in their conventional classes. On the whole, it must be suggested that distance learning is at least currently only an alternative to, but not a replacement of, the traditional university.

Meanwhile, a number of educational systems were developed to integrate distance learning and traditional on-site courses. With the help of distance learning technologies, some universities will go global and become available within organizations, regions and even countries. In this context, distance learning extends traditional approaches to education, rather than replacing it.

In spite of these different arguments, it is true that distance learning is actually taking root across the world. The motor behind the thriving development of distance learning is new technologies, in particular the Internet and WWW. In this chapter, I will first give an overview of various distance learning systems, positioning them in a paradigm of classification according to the distance in time and space between students and tutors. After summarizing remaining problems of these systems, Lecture 2000, a project intended to incorporate both Computer Supported Cooperative Work and distance learning, will be discussed. Motivated by the wish to harness the potentials of both communityware and groupware, a model depicting virtual communities, groups and teams in educational settings will be presented.

3.2 Distance Learning Systems

3.2.1 Distance Learning

In April 1993, a NATO Advanced Research Workshop on "Collaborative Dialogue Technologies in Distance Learning" proposed a definition of the term *distance learning*:

- a group of people including students, teachers as well as teaching assistants performing different kinds of cooperative learning activities such as discussion or problem solving, using conventional or multimedia-based material;
- a group of teachers, organizing, facilitating and controlling cooperative learning;

- a group of authors creating courseware which may be used by individuals as well as by a group during the learning and teaching process;
- a computer environment supporting all these activities.

However, this proposal could not help distinguish between various forms of distance learning systems but rather complicate the understanding of this term itself, especially in the face of a dozen of dazzling terms used to denote the field like *Computer-Aided Instruction (Learning)*, *TeleTeaching*, *Telelearning*, *TeleTutoring*, *Distance Education*, *Education Information Brokerage*, *Virtual Classrooms* and *Virtual Universities* etc. Thus, the first task we face is to investigate distance learning systems from a special and consistent perspective, positioning all distance learning systems into a single paradigm.

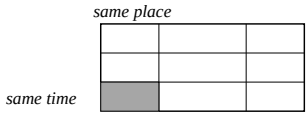
3.2.2 Taxonomy of Distance Learning Systems

The key for distance learning is, indeed, the word *distance* which differentiates it from other possible educational forms. In section 2.4.2 on page 34, a time-space taxonomy of groupware systems has been illustrated. Similarly, using distance in time and space between students and tutors suggested by Lawhead *et al* [Lawhea97] as distinctive elements, we can better investigate a wide variety of distance learning systems. Figure 3.1 shows such a taxonomy.

	same place	***partly distant in place***	***distant in place***
distant in time			Courseware Products
partly distant in time		Distance Learning at Universities	Virtual Universities
same time	Traditional Classrooms Augmented Classrooms		TV Universities

Figure 3.1: Taxonomy of distance learning systems in terms of the distance in time and space between students and tutors

Let us have a look at some typical systems which appear in this paradigm.



same time and same space

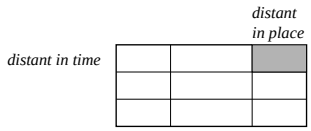
First of all, conventional education, where tutors and students appear in the same place (classrooms) and at the same time, belongs to this category. So far, most pedagogical experts still think of face-to-face instruction as indispensable.

Even within traditional classrooms, some innovations are occurring in an effort to better support the learning and teaching process at universities. A distinguished example is the

project *Classroom 2000* [Abowd96b] , which was initiated at Georgia Tech in July 1995 and intended to investigate the impact of ubiquitous computing [Abowd96a, Bellot93, Streit96, Weiser93] on education. As the developers state, the philosophy underlying the project is to view *"college classroom teaching and learning as a multimedia authoring activity"*. Based on this design idea, the Classroom 2000 project applies a variety of ubiquitous computing technologies like electronic whiteboards, personal pen-based interfaces and the World Wide Web together with software to facilitate automatic capture and content-based access of multimedia information in the university classrooms.

The initial approach is to augment the actual classroom by augmenting existing media such as chalkboards and paper with electronic media. Lecturers are complemented with a presentation on a high-resolution, large, interactive display device. The lecturers or students can share prepared materials during the class, and are also able to write or draw on the electronic display. Moreover, for increasing student productivity, each seat in this classroom will have an inexpensive pen sensitive display device. A student will be able to take electronic notes, either by writing on "blank pages" or by annotating an electronic copy of the presentation on the main display. Students can review their notes to recreate what happened in class at any time.

Nevertheless, in addition to people's concerns about the prospect of these luxurious classrooms, the presumption of viewing college classroom teaching and learning as a multimedia authoring activity will also pose many disputes. After all, lecturing is much more than authoring.



distant in time and distant in space

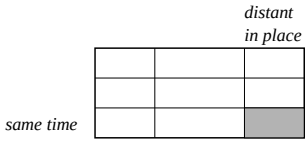
In these systems, tutors and students will never meet face-to-face. A typical example is "courseware" which in the beginning of 1990's appeared in supermarkets, bookstores of some countries, like the USA and some Asian countries such as China and Singapore. Many ventures have been working together with schools and universities to develop Computer-Aided Instruction Systems. Consisting of *lectures, exercises and examinations etc.*, these systems are supposed to help pupils or students privately studying before and after classes or to train professionals of companies. On the other hand, *Exercise Library Systems* provided learners with thousands of exercises in order to help students consolidate the acquired knowledges from classes. Normally, these systems can also automatically generate examination sheets according to users' preferences and needs (e.g. levels of exams) thus enabling learners to test their achievements.

An example of this perspective is the project *Virtual Classroom[TM]* [Turoff95]. Developed in New Jersey Institute of Technology, USA, the Virtual Classroom is offering an entire degree program, the B.A. in Information Systems, via Videotapes plus the Virtual Classroom.

Rather than being built of steel and concrete, the Virtual Classroom consists of a set of group communication and work "spaces" and facilities that are constructed in software. According to Turoff, the software structures incorporated in this system were *"specifically designed to support collaborative learning, including discussions, student presentations, joint projects,*

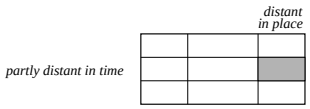
debates, role-playing games, etc." However, all these activities are based on *asynchronous* participation of students and instructors, which causes me to position the project into the category of systems distant in time and distant in space.

same time and distant in space



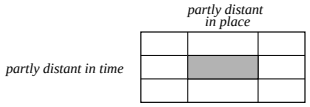
Systems which adopt synchronous communication to overcome the physical distance between tutors and students can be classified into this category. TV universities and some distance learning systems only using synchronous communication facilities like video conferencing to relay lectures between tutors and students dispersed in space fall into this category. Instructors and students separated physically appear in a lecture mediated by telecommunication (e.g. TV meeting systems) and network technologies at the same time. In some distance learning systems of this category, students can ask questions via back channel. But teachers and students never meet face-to-face to hold discussions as in conventional classrooms.

partly distant in time but distant in space



In this category, *Virtual Universities* have certainly dazzled people with many possible advantages of using the Internet for education. Other concepts for describing the similar stuff include *Education Brokerages in the Internet* [Hamala96], *Electronic Knowledge Marketplace* supported by agents [Maes97] etc. The shared characteristic of these systems is that teachers and students never meet face-to-face but in Cyberspace either in real-time or asynchronously. Because of these systems' remarkable importance for distance learning systems, the next paragraph will be dedicated to this topic.

partly distant in time and partly distant in space



Apart from systems depicted above, a wide variety of distance learning systems are intended to improve, enhance and facilitate learning and teaching activities at traditional universities where students and tutor are located in a geographically distributed campus or even distributed across several universities. An example is the TeleTeaching project of the universities of Mannheim and Heidelberg in Germany [effels96], where students can appear in the conventional classrooms or attend a lecture through communication systems like MBone, Videoconferencing Systems, or TV Systems. Even in a centralized campus, distance learning systems can be used to promote flexibility of learning and teaching activities. Normally, developers of these systems insists that distance learning systems can not be a substitute of, but a complement to, conventional classrooms.

Trying to support and enrich common university lecturing by applying the potential of the new technologies and by designing and implementing an integrated multimedia-based learning and teaching environment, *Lecture 2000*, which serves as the cornerstone of this thesis, is also such a system. Indeed, the work done in this thesis is part of the project Lecture-2000.

3.2.3 Virtual Universities

Since virtual universities have very specific features, I will pay more attention to the underlying concepts.

Certainly, virtual universities are utterly unorthodox educational "organizations". According to Chellappa [Chella97], a virtual university *"does not have a traditional campus, professionals' offices, or a library; instead, there are electronic workspaces and global libraries that provide richer functionality and features than physical analogs"*.

Despite the feature of being virtual, a virtual university usually consists of an administrative body, instructors, content providers/validators, and students connected through an open electronic infrastructure with appropriate control mechanisms.

The specific features of virtual universities can be characterized by answering the question: *" What has instigated people to the idea of developing virtual universities?"* In general, we can tackle the question from three aspects.

Reengineering of Education Process

A remarkable rationale for developing virtual universities is the need to reengineer the education process. Chellappa *et al* argue that besides delivering customized educational content and linking far-flung professors and students, a virtual university is a *"reengineered vision of a university' educational process"*. Hämäläinen puts an emphasis on the necessity to reengineer education in the face of technological advances as follows:

> *"New technology alone will not make education more efficient. If we continue to re-implement conventional models borrowed from classroom-based or distance education focusing on passive transmittal, we can expect only marginal improvements and may well simply escalate costs" [Hamala96].*

Behind the motivation of reengineering of education process is the pervasiveness of the Internet and the Web technologies, which provide the platform for innovative forms of teaching and learning. On the other hand, electronic commerce is growing exponentially [Riggin98], which I believe will probably have a strong impact on the further development of electronic education.

Promising to reengineer the education process, the focus of virtual universities seems to shift from simply supporting people who can not appear in conventional classes, as stated in section 3.1, to conducting a thorough reform in the education field.

Just-in-time and On-demand Approach

The challenge facing the reengineering of the education process is to develop *demand-led, rather than product-driven and just-in-time rather than in-advance* learning approach, which

means that students are potentially able to learn what they need when and where they want to and in the format most appropriate to their needs.

One of the most striking features of the current educational model is the enormous duplicated effort that goes into the production and presentation of the same courses in thousands of locations with marginal reuse or sharing of efforts [Hamala96]. Some experts believe that new infrastructure of virtual universities will leverage the ability to mass-market products to customers around the world, achieving an unprecedented economy of scale that should dramatically reduce the unit cost to a mere fraction of what universities commonly charge today.

Enhancement of Collaboration between Students and Tutors

A problem haunting distance learning is the deficiency of interaction between students and tutors. In spite of the promise to greatly enhance communication and interaction between students and tutors by using Internet, telecommunication and wireless computing etc, the issue remains unresolved. Virtual universities still face the challenge to develop more efficient and promising mechanisms. To some extent, it is also one of the main reasons why many people remain skeptical about the future of virtual universities. In fact, this thesis also intends to overcome the obstacles caused by the physical distribution between students and tutors. Some of the problems posed by the distance between students and tutors will be briefly discussed later in the chapter [2].

3.3 Information Sharing and Collaboration in Distance Learning

In the face of various systems used in educational settings, I intend to concentrate on recognizing their common features, highlighting their key points.

Traditional distance learning systems put strong emphasis on independent study, supported by well developed learning materials. This model can be characterized as one-way media. However, the potential of CSCW has quickly recognized at least two kind of activities: *information sharing* and *collaboration*, which embodies an important change of distance learning systems focusing mainly on private study, the trend is towards supporting a more cooperative situation.

Recalling the different levels of communication depicted by Figure 2.2, informing is in the lowest level of interaction between people. In terms of information sharing, informing refers to anonymous communication through the mass media or local resources. In this case,

[2] Some other reasons for developing virtual universities are addressed by Carswell, stating that there is the euphoria associated with new ideas and technologies, and naturally universities want to be part of this [Carswe98]. Moreover, there are pressures on universities, both financial and social, such as supporting the physical infrastructure of a university, which are causing them to explore the effective use of technology in teaching.

there is often only a one way communication channel. With the help of CSCW, interaction and collaboration among students as well as between students and tutors will play a more important role in learning process.

3.3.1 Information Sharing

Traditionally, universities are content providers. Professors generate teaching materials for courses and seminars. However, universities are due to a radical restructuring in terms of content providing because of the rapid technological development in the Internet and the Web as well as the deteriorating financial and structural crisis in universities [Tsichr99].

In the face of these challenges, traditional concepts about information management at universities need to be overhauled. For distance learning, the following issues concerning information sharing must be investigated:

- *Information:* What does the information consist of?
- *Information Source:* Where does the information come from?
- *Information Acquisition:* How can the information be acquired?
- *Information Management and Use:* How is the information organized and used?

This section will mainly focus on the last issue, i.e. the management and use of electronic learning material.

Information

Firstly, information relevant to distance learning consists of *learning materials* such as lectures, seminar papers which constitute the basis for learning activities of students. In some systems, *auxiliary materials* such as exercises and solutions as well as annotations taken by students are also included.

Information Source

As introduced before, the emergence and rapid evolution of the Internet and the Web have a tremendous impact on the teaching and learning activities at universities. Universities as providers of quality materials are facing challenges. More and more students do not longer consider information from courses and seminars as their unique information source. Instead, they can get highly qualified information from the Internet. Basically, universities can import content from specialists or distinguished scientists around the world.

Briefly, students can get useful information from the following sources (see Figure 3.2):

- course scripts, papers and other on-line materials

- books, CD-ROMs and other off-line materials
- live lectures and discussions imported through the networks from professors and scientists elsewhere
- content specially prepared by remote specialists and distinguished scientists

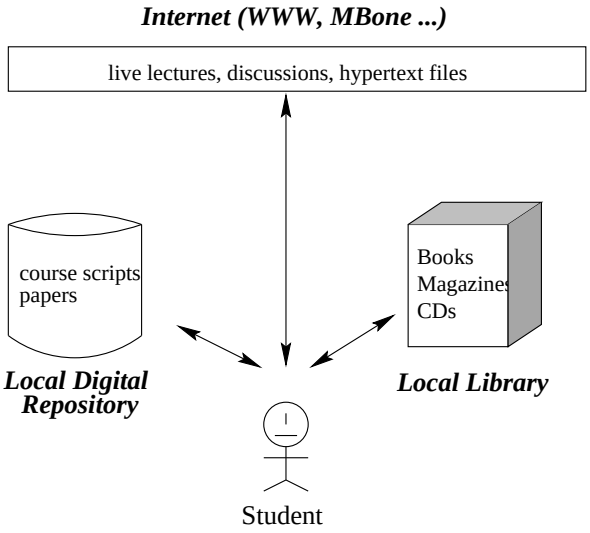


Figure 3.2: Information sources

Information Acquisition

With the various information sources, different approaches are available for people to get their needed information. Firstly, there are already a wide variety of search tools used to locate on-line materials around the world. Most traditional libraries provide electronic retrieval systems for users to easily identify and retrieve off-line materials according to user specified authors, titles, published dates and key words. MBone has long been used to relay live lectures and discussions with preannounced lecturing or discussion date, duration and frequency. Finally, agents can automatically search for specialists and scientists distributed all over the world who can provide knowledges and experiences for these agents' owners.

Recalling the example about agent systems in communityware (see section 2.3.4), which proposes to use selling agents and buying agents to facilitate the learning and teaching activities through the electronic knowledge marketplace. In this case, students can directly find course materials and persons who can offer them according to their own needs. This approach differs from traditional distance learning systems in the way that here learning and teaching become utter buying-selling relationships, which may be the boldest idea for distance learning up to now. In this case, there will be no centralized information repository managed by an organization. Instead, the learning process is distributed. More specifically,

the learning and teaching activities will take place directly among one person who seeks knowledge and another who can offer it.

However, there are some issues remaining unresolved with this approach to brokering information between specialists and students, for example:

- Who provides a degree to the students?
- Who is in charge of this kind of "knowledge business" once problems emerge?

Because students can now get learning material from diverse sources, there are increased concerns about the quality of these materials. In order to guarantee the quality of distance learning, the collected learning materials must undergo some quality assurance procedures; that is, learning materials should be evaluated and accredited. Ideally, the evaluation and accreditation process has to be carried out by an organization endowed by the law with the authority to validate learning materials. But there are currently still no practical measures to assure the enforcement of such a process.

On the other hand, the customization of learning material is of great importance to distance learning as well. Figure 3.3, for example, demonstrates the customization process of learning materials. To collect learning material which best meets the requirements of students, one must first conduct some surveys about the demands of students. For example, the demands and preferences of students can be obtained through questionnaires filled out by students. This kind of survey will lead to the establishment of a repository of course materials. During the use of this repository, the system searches through the material repository according to students' needs and preferences, and generate courseware fulfilling the requirements of the students.

Information Management and Use (Web-based Information Systems)

For a long time, most learning materials in distance learning have been distributed on CDs, video tapes and/or through TVs. Due to the development of the Internet, the Web has increasingly become the main platform for students to access learning materials. Already, several architectural approaches have been proposed to manage electronic learning materials. In this section, Web-based Information Systems (WISs), which combine a global addressing system, network protocol, document mark-up language and client-server architecture, will be discussed. We will show how they provide a simple method for users to *search*, *browse* and *retrieve* information, influencing the further development of distance learning systems.

According to Isakowitz [Isakow98], Web-based applications encompass four general kinds of Web-based systems: *Intranets*, to support internal work, *Web-presence* sites that are marketing tools designed to reach consumers outside a firm, *electronic commerce* systems that support consumer interaction such as online shopping, and a blend of internal and external systems to support business-to-business communication, commonly called *extranets*. The Web's standard interface and inexpensive global access enable almost any organization to implement an interorganisational system. With WISs increasingly applied for electronic

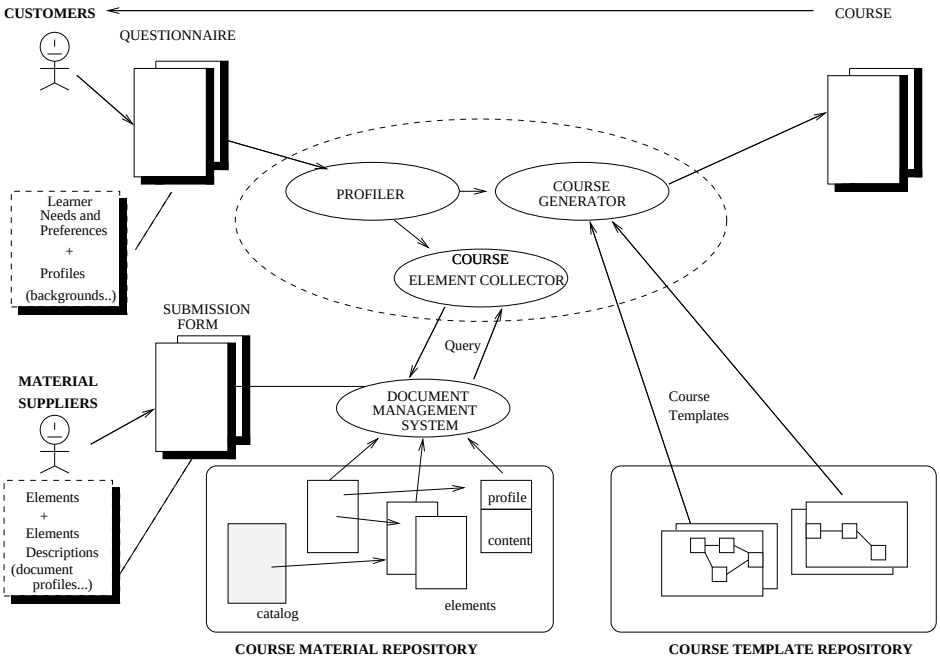


Figure 3.3: Customization of learning materials according to Hämäläinen [Hamala96]

commerce like electronic shopping [Lohse98] and electronic auction [Tenenb98, Heck98], I foresee an increasingly important role which WISs can play in information distribution for distance learning.

Architecturally, WISs are usually tightly integrated with other non-WISs such as databases and transaction processing systems. Common Gateway Interface (CGI) applications link databases and Web servers together. As Dennis states: “WISs are information systems first and Web systems second” [Dennis98]. However, WISs are sufficiently different from traditional information system (IS) in that it requires new approaches to software engineering.

WIS handles multimedia documents. Thus, a WIS must address issues like creation, management, delivery and retrieval of documents, as well as issues like structuring, linking and navigating with respect to multimedia. In general, the development of WISs involves the following issues [Balasu98]:

- information architecture,
- user interface and navigation design,
- content creation and authoring,
- workflow and document management,
- publishing,
- document review and link management, and

- search and retrieval.

Some of these issues such as content authoring and navigation will be discussed in chapter 6. Here, we focus on architectural approaches to WISs. There are at least two different approaches to implementing Web-based Information Systems. This discussion will serve as a basis for the implementation of our targeted system described in chapter 6.

Firstly, take the Web DataBlade Module of the Informix Universal Server which provides a complete set of tools, functions, and examples for developing complex database driven Web sites. Typically, Web database applications require a Common Gateway Interface (CGI) application written in Perl, C, shell, or PHP to access data. The *Webdriver* application offered by Informix replaces the need to write CGI programs. With the aid of the Webdriver application, the process of accessing data stored in an Informix database through the Web looks as follows (see Figure 3.4):

1) When a URL contains a Webdriver request in the form of *http://host/cgi-bin/webdriver?...*, the Web browser makes a request to the Web server to invoke the Webdriver.

2) The Webdriver composes an SQL statement to retrieve the requested AppPage (i.e. a HTML page with dynamic tags performing data queries to database) and then expands it by executing the WebExplode function.

3) The WebExplode takes the selected AppPage from the Web application table stored in an Informix Server database, executes the SQL statements within the AppPage, expands the Web DataBlade module tages and formats the results in HTML.

4) The WebExplode returns the resulting HTML to the Webdriver.

5) The Webdriver returns the HTML to the Web server.

6) The Web server returns the HTML to be rendered by the Web browser.

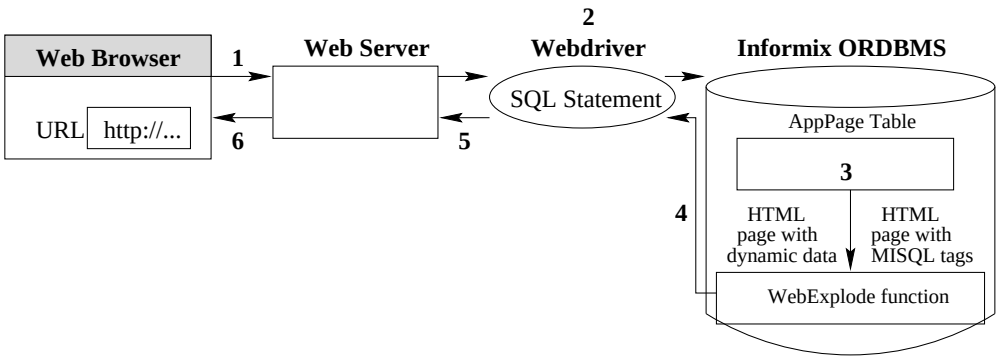


Figure 3.4: Architecture of Web-based Information Systems with Informix Universal Server

While many Web applications follow this architectural approach, the architecture of WISs is currently undogoing dramatic changes with the advances of Java and CORBA (The Common

Object Request Broker Architecture) technologies. A CORBA object, using the client ORB (Object Request Broker) and IIOP (Internet Inter-Orb Protocol), can access the services of an object on a server, which in turn can access one or more objects of relational databases or legacy systems as long as each system includes an ORB.

A client object requests the services of other objects through an ORB that resides on the client system. Using IIOP, the client ORB then reaches across the network, looking for ORBs on other systems and the server objects which can provide the requested services. Each object has a unique name, provided according to the CORBA naming service, which identifies it and its services to other objects. Once the ORB has found the requested service or object, the client object communicates with the server object, still using IIOP. Each object's IDL interface tells other objects how to use its services and the results those services generate.

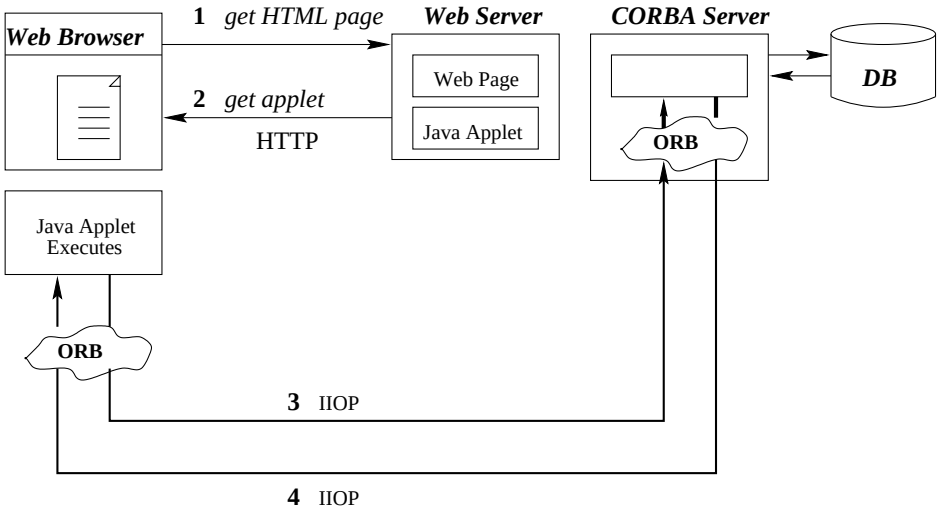


Figure 3.5: Web-based Information Systems using CORBA technologies

With a Java-enabled browser and CORBA technologies, a user can access services of objects from a multitude of servers and hosts. Such a process works as follows (see Figure 3.5):

1) In a Web browser, the user enters a URL that points to a page on the Web server.

2) The Web server receives the user's request and uses HTTP to return a Web page to the client's browser. The Web page contains the Java applet that makes up the client component of a Web-based Information System.

3) With the Java applet running, the applet on the client side sends an IIOP message through the Object Request Broker to a server-side software object, invoking the object's methods and passing the user's request. The server-side object queries the database for the client-side object.

4) Upon receiving query results from the database, the server-side object returns the results to the client-side Java applet through the ORB. The applet receives the results and displays them in the user's Web browser.

The client might be a PC running Windows, a workstation running Unix or a Macintosh. Server objects may be written in Java, C/C++, or other languages and may include new and legacy code. Objects do not need information about each other's implementation's details. They communicate only through their defined interfaces.

In chapter 6, we will see that the current implementation of Lecture 2000 is the combination of the above two approaches. Course Composer, which enables tutors to compose lecture scripts with reusable multimedia modules in the system repository, makes use of CORBA technology to allow the communication between client and server objects. Course Navigator, on the other hand, uses Informix Webdriver to access pages in the database.

3.3.2 Collaboration

So far, distance learning systems have attempted to make use of a wide variety of communication mechanisms to support the collaborative learning and teaching activities in educational settings. This collaborative support is of particular importance to distance learning because of the inherent physical distance among participants of the learning and teaching processes. On the other hand, the deficiency of providing this kind of collaborative supports in distance learning has already posed serious concerns about the quality of the learning process and the prospect of distance learning as a whole. In this section, I am mainly concerned with a synchronous and an asynchronous collaboration mechanism respectively: MBone and BSCW, both of which can be seen as distinguished examples of approaches to supporting collaboration in distance learning systems.

Multicast Backbone (MBone)

As an infrastructure supporting various synchronous collaborations, Multicast Backbone is playing a more and more important role in distance learning systems, from relaying lectures to bringing people to real-time discussions.

Short for Multicast Backbone, MBone has been in existence since early in 1992, when people attempted to multicast audio and video from a meeting of the Internet Engineering Task Force (IETF) [Macedo94, Bauer95, Casner92]. Since then, more application examples about MBone applied for meeting and education have appeared, including the pioneering Jason Project developed by the Jason Foundation for Education in the USA, which used MBone during 1993 to transmit experimental data from underwater science and exploration and to support shore-based models [Macedo94]. Recently, universities are increasingly taking advantage of the technology to transmit their lectures and to support collaboration between students and tutors. As Hardman states, MBone is now moving from the pilot stage to a usable service in countries like the U.K. and the U.S. [Hardma98]. In Germany, people have already begun using this facility in education for several years. Our chair, for example, already began relaying the lecture CSCW for students two years ago.

Basically, MBone is a virtual network running on "top" of the Internet [Erikss94]. Mbone is composed of networks (*islands*) that support multicast transmission. On each of these is-

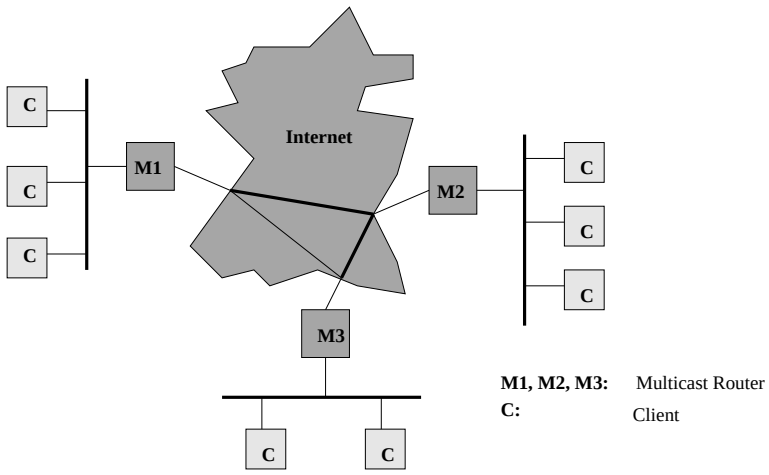


Figure 3.6: MBone topology [Erikss94]

lands, there is a host (Multicast Router) that is running the *mrouted* multicast routing demon. The mrouters are connected with one another via tunnels. Figure 3.6 shows islands, each of which consists of a few clients, mrouters denoted by M1, M2 and M3, as well as tunnels between mrouters.

I will now address some issues relevant to MBone, including some MBone-based application scenarios for distance learning.

Multicast The way data packets are sent on the Internet is divided into three types. With *unicasting*, one host sends information packets to another specific single host. *Broadcasting* sends packets from one host to all hosts on the same subnet. Normally, the routers between one subnet and another subnet will not let broadcast packets pass through. Finally, *multicasting* sends packets from one host to a group of hosts. MBone utilizes the technology of multicasting to transfer data including video/audio streams to a group of users who have expressed desire to receive the multicasted data from the Internet.

Media Tools Up to now, a number of applications have been developed in the context of MBone. Typically, these tools include audio- and videoconferencing, whiteboard, text editor and session directory. An overview about these tools is shown by table 3.1.

Audioconferencing	VAT, RAT, IVS ...
Videoconferencing	VIC, NV, IVS ...
Shared Whiteboard	WB, TeleCanvas ...
Shared Text Editor	NT/NTE ...
Session Decription Directory	SDR (SD)

Table 3.1: Typical media tools used in MBone

In contrast to HTTP used for the Web, media tools on the MBone are built by using the protocol RTP (Real-Time Transport Protocol[3]), which provides end-to-end delivery services for data with real-time characteristics, such as interactive audio and video. Moreover, applications typically run RTP on top of UDP rather than TCP mainly because occasional loss of an audio or video packet is usually acceptable, while the delay for retransmission using TCP is unacceptable in an interactive conference.

Session Management As one of the main concerns of the thesis, session management will be investigated in more detail in chapter 5. Due to the importance of session management in facilitating collaboration, it is valuable to take SDR as example to show how session management is realized in MBone. Basically, the session management in SDR allows users to publish and to find sessions in MBone.

For announcing and scheduling multimedia conferences on the MBone, SDR is currently a popular tool [4]. Its main features regarding session management include:

- *Session Announcement*

  SDR uses a session announcement protocol called SDAP (Session Directory Announcement Protocol). It periodically multicasts a session announcement packet describing a particular session. To receive SDAP, a receiver simply listens on a well-known multicast address and port. After getting a session announcement packet, the receiver simply decodes the SDAP message, and then displays the session information for users.

- *Scheduling Sessions*

  New sessions in SDR can be created by pressing the "New" button on the SDR main window (see figure 3.7) . Every session must have a name, a short description, a scope (i.e. how far packets are distributed), contact information, a start time and a stop time. In addition, sessions may optionally possess links to further information in the Web, a number of detailed media descriptions, and more detailed timing information.

- *Media Descriptions (Specification of Tools)*

  Of course, media descriptions are the main reason for describing a session. SDR allows users to create conferences with one or more media tools like audio- and video-conferencing, whiteboard and shared text editor.

- *Secure Announcements and Private Sessions*

  Although the purpose of SDR is to publish sessions, not all sessions need to be advertised publicly. Secure sessions are advertised twice - once unencrypted with just the bandwidth, contact information and multicast addresses, and once encrypted with

[3] ftp://ftp.isi.edu/internet-drafts/1id-abstracts.txt

[4] SDR is based on sd, a Session Directory developed by LBL (Lawrence Berkeley Laboratory, California, USA)

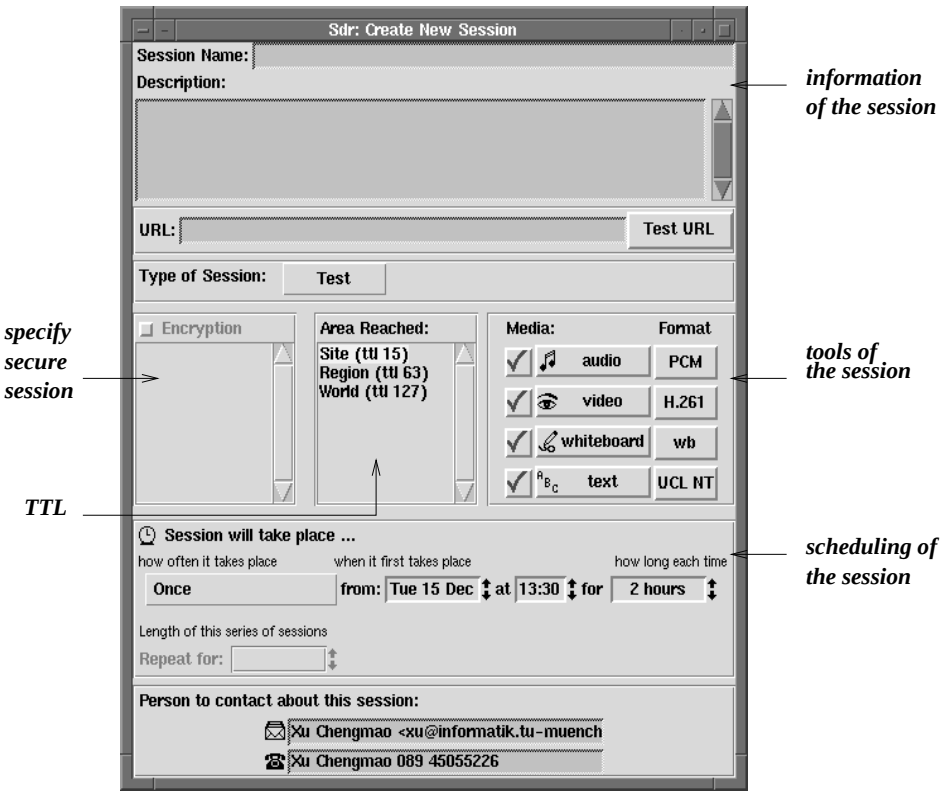


Figure 3.7: Create a session in SDR

all of the information including encryption keys for the multimedia tools. Meanwhile, SDR allows a session to be created as “public” or “private”.

To create a private session, SDR must be pre-configured with a set of private group names and their encryption keys. When the user selects “private”, the list of the groups the user has keys for is displayed.

Undoubtedly, MBone will become more and more important in distance learning. Firstly, MBone greatly benefits distance education because multicast conferencing costs a fraction of the cost of other solutions like cable TVs and other forms of teleconferencing. Secondly, multicast conferencing allows real-time multiway audio and video communication over the Internet, which can give remote students a feeling of physical presence in classrooms. Application scenarios of MBone for distance learning include:

Relaying of Lectures Certainly, media tools of MBone such as audio- and videoconferencing systems can transmit lectures in real-time. Remote participants can see and hear what the tutor is doing and saying. In addition, the lecture material can also be transmitted to all session participants. Finally, unlike listening to a radio broadcast, the remote participants could also talk back. Thus participants can ask questions to tutors and get answers from them.

One issue to be tackled during the relaying of lectures through MBone is how to solve the problem that transmitted pictures of lecture scripts, which are usually shown in a Web browser and projected to a screen in a classroom, are too small to be discerned by receivers. A possibility to overcome this difficulty is to let MBone transmit URLs of Web pages which the lectuere is explaining instead of their projected pictures. On the receiver side, one can clearly view the lecture scripts being shown by tutors through the receiver' own Web browser according to the received URLs.

Groupwork In the meantime, media tools including shared workplaces such as a shared whiteboard enable people to be engaged in groupwork. SDR integrates methods for the initiator of a session to invite other colleagues or friends to join it by clicking the button "Invite" and then giving the string "username@hostname" to specify those to be invited. The latter can then press the button "Accept Invitation" to attend the session.

The World Wide Web

In addition to supporting information sharing as illustrated before, the Web has been increasingly used to allow people for conducting cooperative work. Although Email is still the most popular tool in the Internet used by people for communication, many systems are trying to exploit the tremendous potentials of the Web to do it better. In this section, I will discuss some issues of the Web as a platform supporting collaborative work, especially paying more attention to the system BSCW, which is primarily used to support asynchronous cooperative work.

WWW for Collaboration Berners-Lee, one of the inventors of the Web, said that the Web was developed as a pool of human knowledge, which would allow collaborators at remote sites to share their ideas and all aspects of a common project [Berner94]. However, the widespread adoption of the Web offers enormous potentials in supporting cross-platform cooperative work within widely-dispersed work groups. A lot of projects have investigated this kind of potentials.

The potential and viability of the Web for cooperative work can be seen from several distinguished features [Bentle97, Dix96, Woo94]:

- Web browsers are available for all popular computing platforms and operating systems, providing platform-independent access to information.
- Web browsers offer a simple user interface and consistent information presentation across these platforms.
- Web browsers are already part of the computing environment in an increasing number of organizations, requiring no additional installation or maintenance of software for users who cooperate through the Web.

It seems that the Web is currently best suited for asynchronous communication between people because of the following critical requirements for synchronous communication which are, however, "insurmountable" in the current Web implementation:

- Transmission of continuous media:

  HTTP does not directly support the specification of a guaranteed transmission rate between server and client. Thus, the Web is unsuitable for real-time continuous media like audio and video, which will certainly make it difficult for people to develop systems based on continuous media, such as audio- and videoconferencing.

- Peer-peer communication:

  In the Web, there is no support for server-server, (server initiated) server-client or client-client communication. Then, the Web is problematic for synchronous applications where the server usually needs to play an active role to transmit information between cooperative partners.

Of course, people have attempted to solve these problems by taking advantages of technologies like plug-ins or java applets. The Web in its current form is, however, as Bentley states, more suited for centralized cooperation tools which have no strong requirements for highly-interactive user interfaces, rapid feedback or a high degree of synchronous notification [Bentle97]. Therefore, much of the work in this area has focused on more 'asynchronous' forms of cooperation support. BSCW is a typical example of this type of work.

Basic Support for Cooperative Work (BSCW) The BSCW project at GMD[5], Germany, is concerned with the integration of collaboration services for existing environments, supporting widely-dispersed work groups with different computing, network and infrastructures. The BSCW shared workspace system is the basis for these services, including features for uploading documents, version management, group administration. All of these are accessible from different platforms using *unmodified* Web browsers. At present, uploading documents is no longer a problem in most browsers. The main contribution of BSCW is to enable a group of people to share documents through common Web-based pages.

The BSCW system is based on the notion of a 'shared workspace' which the members of a group establish for organizing and coordinating their work. A shared workspace is a repository for shared information, accessible to group members using a simple user name and password. A shared workspace can contain information such as documents, images, links to other Web pages or FTP sites, threaded discussions, members contact information and so on. The contents of a workspace are represented as information objects arranged in a folder hierarchy. Members can transfer (upload) information from their machines to a workspace and set access rights to control the visibility of this information and the operations which can be performed by others. In addition, members can download, modify and request more

[5] http://bscw.gmd.de

details on the information objects. After each operation the server returns a new HTML page showing the new state of the workplace.

In fact, present Web technologies plus CGI (Common Gateway Interface) have overcome all obstacles for implementing BSCW shared workspaces. The BSCW system provides an extension of the Web's client-server architecture without modification of Web clients, servers or protocols. The core is a standard Web server extended by the BSCW software through CGI.

Extend WWW for Synchronous Collaboration Although BSCW supports synchronous collaboration as well, it mainly serves as an asynchronous collaborative tool for people who share documents. As discussed above, it requires more extensive innovation and development of proprietary mechanisms to develop Web-based synchronous collaboration systems, which often do not run across all platforms and require more complex installation of additional client and server software [Palfre96, Kirby95, Woo94]. Dix investigates various possibilities for using the Web to support synchronous collaboration [Dix96].

Because of the benefits brought by the Web-based interface, the standard platform and the simplicity of use, more and more distance learning systems will be developed on the basis of Web technologies.

Of course, there are other mechanisms to support collaboration among students and between students and tutors, some of which are developed based on spatial approaches. In fact, spatial approaches are the main focus of this thesis.

3.4 Problems in Existing Distance Learning Systems

Certainly, there are a group of critical problems yet unresolved in distance learning systems, for example, the quality of learning materials. However, this thesis is supposed to primarily tackle issues concerning interaction and collaboration among students as well as between students and tutors. After introducing various distance learning systems including their classification, key issues of information sharing and collaboration, it is necessary to summarize problems which remain unresolved by existing distance learning systems in terms of interaction and collaboration support.

With respect to interaction and collaboration, most current distance learning systems have provided people with both synchronous and asynchronous communication mechanisms in an effort to compensate the lack of face-to-face contact in conventional classrooms. But a number of open issues still remain, some of which are not only common to most existing distance learning systems, but have been problematic for many CSCW systems for a long time:

- *Problem 1:* They have failed to recognize and to fulfill people's needs in educational settings for different levels of communications, ranging from informing, coordination, collaboration, to cooperation;

- *Problem 2:* They have failed to recognize the role and importance of various collaborative entities (i.e., communities, groups and teams) in educational settings. Indeed, these entities are needed to foster and to enable various communications at universities;

- *Problem 3:* They have failed to support seamless transition among these communication levels as well as among these collaborative entities.

- *Problem 4:* They have ignored or have not effectively handled a number of issues concerning the learning process such as informal interaction, lightweight connection, privacy control, history management etc.

I first address the first three problems while the last will be dealt with in chapter 5.

3.4.1 Communication Levels

The study in section 2.2.1 on page 16 has shown that there are various communication levels needed by people, including *informing*, *coordination*, *collaboration* and *cooperation*. Further, I have defined two terms *interaction* and *collaboration* to denote relatively lower and higher communication levels. Both interaction and collaboration could play a key role in promoting the exchange of knowledge between people and enhancing the efficiency of learning and teaching processes at universities. Let us investigate the role of each of them briefly.

Interaction Being at the lowest level of communication, informing can be thought of as the basic form of interaction. Informing refers to anonymous communication through electronic material. In the simplest circumstance, tutors provide students with high-quality learning material. Most distance learning systems support a repository for the storage and use of learning material. In the past few years, people are increasingly assured that there is a need for modernizing the form of information production and distribution at universities [Tsichr99].

Apart from this kind of basic learning activities, there are other forms of interactions needed by students and tutors:

- answer questions arising from lectures;

- exchange learning experiences;

- engage in some common tasks such as practical courses, seminars or projects, and

- attend other discussions.

Indeed, interaction is a ubiquitous activity at universities. Basically, all people need to communicate with each other. Common to all these learning activities listed above is that students and tutors are usually involved in very short, informal and spontaneous interactions. They often interact with each other in a very loosely and often temporarily built group. More specifically, there are in communities. Further, people's behaviors are not restrained by obligations, duties or goals. They may have some partial common goals. However, they do not work for a definite common goal.

Collaboration In addition, close collaboration among students can be seen everywhere at universities as well, from attending the same practical course or seminar to being engaged in a project. The difference is that they do not only share common interests and preferences, but also are involved in a cooperative process for achieving a common goal. Therefore, they need to contact each other much more often than in interaction.

In addition to being involved in a common work such as a project, there are other opportunities for people to work cooperatively. For example, students often attend a small group consisting of 3 - 5 persons and learn together. Learning is easier as a group activity. In the next section, I will come back to this topic and give more detailed explanations.

Conclusion 1: *Most existing systems have failed to recognize and fulfill the need for various communication levels by people in educational settings. In other words, people in educational settings do not only need interaction but also collaboration. The problem in most distance learning systems is that they support either interaction or collaboration but not both of them.*

3.4.2 Collaborative Entity Hierarchy

In many distance learning systems, interaction and collaboration seems to occur only between students and tutors. Under this circumstance, it is not difficult to understand why some tutors who have experienced distance education complained that they must spend more time with students than those in their conventional classes. Indeed, we need a new spectrum of collaboration with a wide variety of collaborative possibilities used by students and tutors in distance learning.

In chapter 2, different collaborative entities (e.g. *communities*, *groups* and *teams*) have been illustrated. Each of these entities prefers a certain communication level. While people in communities like to exchange feelings, experiences and knowledges, groups and teams aim at supporting collaborative work. In distance learning, these different entities have to be supported because they can *foster* and *frame* interaction and collaboration among students as well as between students and tutors. Nevertheless, many systems have failed to come to this point, making it difficult for people to smoothly share their information and knowledge.

In light of the need for different levels of communication between people, distance learning systems should not simply stop with the provision of synchronous or asynchronous tools. They must encourage collaboration in different hierarchies between students as well as be-

tween students and tutors, and foster and enable various groups to conduct their learning activities with them.

Conclusion 2: *The philosophy underlying this work is that we do need diverse collaborative entities that can foster and support different collaborative work among students as well as between students and tutors.*

3.4.3 Seamless Transitions

Many existing systems have failed to support different types of seamless transitions such as *transition between information sharing and collaboration*, and *transition between collaborative entities*, which are of great importance to collaborative work between people.

Transition between information sharing and collaboration Existing distance learning systems have separated information sharing and collaboration in the learning and teaching process. Due to this separation, information sharing in most cases only means that tutors offer learning materials and students utilize them. In these systems, a repository of learning materials is available for students to make use of. Meanwhile, a variety of tools independent of the information repository is supposed to provide collaboration support for people.

However, there is no doubt that information sharing and collaboration are two inseparable components of learning and teaching processes. More generally, the seamless transitions between all communication levels depicted above are indispensable.

Transition between collaborative entities As stated above, we need diverse collaborative entities – communities, groups and teams – to support interaction and collaboration among students as well as between students and tutors. However, these virtual entities should be dynamic, which means there will be seamless transitions between them. For example, groups and teams can be established out of communities, and teams can emerge from groups as well.

This kind of transitions between collaborative entities is supposed to reflect the dynamic and evolving collaborative relationships between people. When people in a community, for example, get to know each other and share more common interests working together, they should be allowed to set up an entity for themselves which can better support their closer collaborative work.

Transition between synchronous and asynchronous work Greenberg [Greenb98] illustrates various gaps which exist in many groupware systems and hinder or block social interaction or that do not let people easily move between different styles of work. One of the gaps exists between synchronous and asynchronous work. In the next chapter, I will discuss this in more detail.

Conclusion 3: *The seamless transitions between information sharing and collaboration transition, and between collaborative entities is of great importance to distance learning. One of the main goals of modern CSCW research is the support of seamlessness.*

In summary, these problems discussed above lead to the separated support for the phases of a collaborative learning, which has been extensively addressed in chapter 1. An integrated learning environment must be able to support different levels of communication and relevant collaborative entities. Only in such an integrated environment, the transition between interaction and collaboration, between different collaborative entities can become possible.

3.5 Collaborative Entities in Lecture 2000

After analyzing problems remaining in current distance learning systems, we recognize that it is critical to have an integrated learning environment supporting all three collaborative entities in educational settings, i.e. communities, groups and teams. With these three collaborative entities supported, all phases of a collaborative learning process, from private study, forming of learning groups, to close collaboration, will be consistently supported in the learning environment. The question to ask is what collaborative entities there are in educational settings.

At a university, interaction and collaboration are a ubiquitous work form between students, professors and research assistants. Both learning and teaching can be regarded as a kind of interactive and collaborative process. Lectures, seminars and practical courses serve as places where interaction and collaboration are organized and take place.

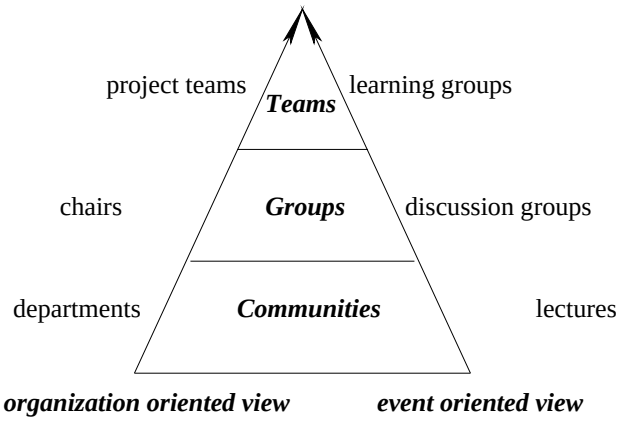


Figure 3.8: Different views of communities, groups, and teams in educational systems

The interaction and collaboration occurring at different levels of abstractions lead to the forming of a variety of communities, groups and teams, whose relationship can be depicted by the pyramid model shown in Figure 3.8. We can observe the model from two different views, i.e. *the organizational view* and *the event-oriented view*. According to the organizational view, a university is organized into departments, chairs and project teams which respectively correspond to community, group and team as defined above. On the other hand,

we can also observe a university in terms of the event-oriented view. Collaborative entities derived from the event-oriented view will serve as the basis for all discussions in the remaining chapters of the thesis. Basically, lecture 2000 distinguishes between the following collaboration entities:

- the class of a lecture as a community;
- learning groups for the lecture as teams;
- a discussion group and spontaneous groups facilitating the transition from the community to the teams inside the community.

Both views reflect the same relationships between communities, groups and teams. Interpreting the differences between the pyramid bottom and its top, one can observe the following trends from bottom to top: less members, but closer cooperative relationships between the members; informal spontaneous interactions dominate within a community (at the bottom of the pyramid) whereas the work in a team (at the pyramid top) is usually planned and goal-oriented.

3.5.1 Class as a Community attending a Lecture

People who come to attend a lecture may come from different departments of a university, from other cities or other states of a country, and even from other continents. At present, such a group of people can either sit together in a classroom or attend the lecture remotely. As an effort to remove the geographical obstacle some lectures at the TUM are propagated to the wide area network via Mbone. The common interest in the lecture serves as a “glue” to unite people who often do not know each other personally. They constitute a community called “class”. Notably, members of the community also include those who never actually make their appearance in class but are nevertheless interested in the content of the course.

3.5.2 Discussion Group for asynchronous Contact Facilitation

Associated with each lecture there is a discussion group available to all attending students providing asynchronous contact facilitation. The discussion group can be regarded as a public service which facilitates communication between lecture participants. Normally, the members of such a group share common interests in exchanging learning experiences and discussing problems connected with the learning process.

Under these circumstances people contact each other asynchronously and often anonymously using nick names. In practice, a discussion group of a lecture plays a role similar to that of the Usenet newsgroups mentioned before. As interactions mature people can establish a team for conducting a more efficient and closer collaboration. A discussion group represents a subgroup in a small community.

Spontaneous Groups for synchronous Contact Facilitation

Informal spontaneous interactions have proven to be very important in communities. In a distributed community, where members do not know each other, this kind of unplanned meeting plays a very important role in identifying potential collaboration partners.

Four categories of interactions have been distinguished [Isaacs96]:

- planned (prearranged meeting)
- intended (explicitly sought by one person)
- opportunistic (anticipated by one party but occurring only when the parties happened to see each other)
- spontaneous (unanticipated by either party).

The latter two types of interactions are called “unintended”. The term “spontaneous (temporary) group” specifies a group formed by people when they meet each other in an “unintended” manner. These kinds of meetings lead to synchronous interactions. Previous research has shown that synchronous interactions provide people with better opportunities for developing a common understanding than asynchronous interactions.

3.5.3 Learning Groups

The class exists only as a very loose and often only virtual community. Experience shows that students prefer to participate in discussions in small groups composed of 2 to 5 persons in order to discuss course related issues because in a small group, the common goals and responsibilities can be better achieved and assumed than in a large one. Therefore, the interactions within loosely coupled discussion and spontaneous groups often lead to the creation of small groups for preparing and reviewing the lecture content in more detail. These small groups, called learning groups, distinguish themselves from the class and the discussion groups by fewer members, closer relationships and clearer objectives. The members of learning groups communicate with each other both synchronously and asynchronously. Learning groups are examples of teams as defined above. Normally, the work of a small group leads to the answers to issues arising from the learning process which can be stored as histories for future studies and examinations.

3.6 Summary – Integrated Support of Groupware and Communityware for Distance Learning

Drawing on the discussion of distance learning systems in this chapter and the investigation of CSCW systems in the preceding chapter, a 3C-model can be given to integrate my

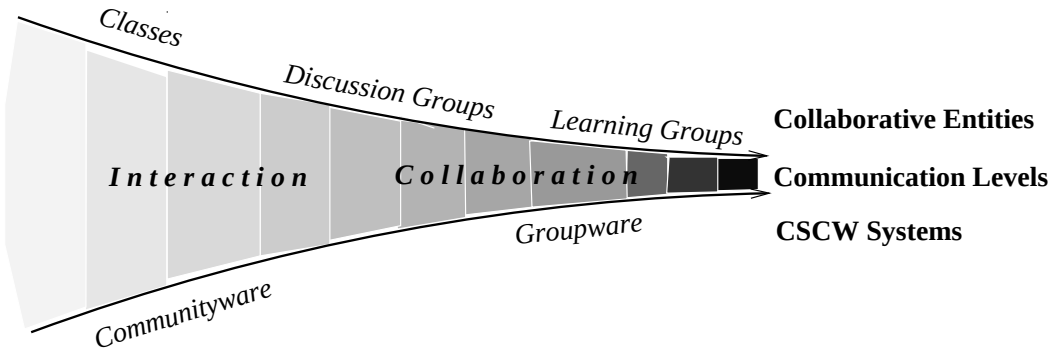


Figure 3.9: Support for collaborative entities from both communityware and groupware in Lecture 2000

discussion results about the support of learning activities in educational settings (see Figure 3.9):

- Communication Levels: There is a need for different levels of communication in educational settings, ranging from informing, coordination, collaboration to cooperation, which can be abstracted into interaction and collaboration.
- Collaborative Entities: Different levels of communication must be fostered and supported in different collaborative entities, i.e. communities, groups and teams. According to our event-oriented view, we have classes as communities, discussion groups as groups and learning groups as teams.
- CSCW Systems: Communityware and groupware can be integrated to provide support for interaction and collaboration within these collaborative entities.

Note that in the remaining chapters, we often mention only communities and groups, thinking of teams only as a more specific form of groups. Moreover, we also often use the terms community interaction and groupwork to denote the general terms interaction and collaboration as defined before.

The remaining task of this thesis deals with the implementation of this model on the basis of spatial approaches.

4 Spatial Approaches to Communities and Groups

Because of their strong relationship to physical reality and therefore their highly intuitive nature, spatial approaches have recently often been used for software systems supporting communities and groups. After showing their general features, I primarily concentrate on spatial approaches to communityware and groupware respectively. In spite of their tremendous potentials for facilitating interaction and collaboration, spatial approaches to communityware and groupware have, however, been evolved independently in the past.

Incorporating the results of last chapter, a spatial framework for educational settings supporting both community interaction and groupwork will be presented in this chapter. Finally, a general view of this framework will be given.

4.1 What is Spatial Approach

In the past few years, spatial approaches supporting the collaboration of groups of people across geographical barriers have become increasingly popular. Both communityware and groupware systems have attempted to use spatial metaphors to structure virtual environments in which people can interact with each other or with various objects within these environments. The reason for this development is that the strong relationship to physical reality of such metaphors and therefore their highly intuitive nature [Benfor93b].

Since all of us have a great deal of rich experiences in interacting with various objects located in everyday spatial environments, these experiences can be exploited for designing computer systems. Harrison writes:

> *"... we live in a three-dimensional world, and the structure of the space around us moulds and guides our actions and interactions. With years of experience, people are all highly skilled at structuring and interpreting space for their individual or interactive purposes" [Harris96].*

Table 4.1 lists common spatial metaphors and their highly intuitive natures [Greenb98, Kuhn96]. Almost all of these spatial metaphors have been used in areas such as interface design [Kuhn96], spatial video conferencing systems [Ishii92], media spaces [Gaver92a,

Spatial Metaphors	• personal offices; • breakout rooms and conference rooms; • live-in dedicated project rooms; • non live-in dedicated project rooms; • public spaces for social interaction and casual work e.g., coffee rooms, foyers, and commons; • cities, landscapes, desktops.
Intuitive Natures	• living and acting in space is a common experience for all people; • spatial structures and artifacts offer familiar affordances and operations; • human memory relies on spatial arrangements and layouts of items; • human spatial experience is tightly linked to visual and auditory perception; • spatial structures convey a wide range of auxiliary information in subtle but intuitive ways, including way finding assistance, visualizing choices for actions, and tracing past operations.

Table 4.1: Spatial metaphors and their highly intuitive nature

Gaver92b, Bly93], CSCW environments [Lee96, Greenb98, Rosema96b] and virtual reality [Greenh95, Carlss93].

Basically, spatial approaches can be defined as follows:

Definition 4.1 (Spatial Approach)
Spatial approach is a kind of methodology which makes use of the highly intuitive nature of spatial metaphors to design computer applications. A computer application based on spatial approaches can be seen as a set of spaces through which people can move, interacting with each other and with various objects which they find there.

In general, spatial approaches help to tackle a range of common issues undoubtedly important to cooperative work. These issues which will be discussed in this chapter include [Benfor96]:

1) **Persistence and on-going activity**

   People's cooperation may take place over long periods of time and may involve may inter-related instances of different cooperative activities. Spatial approaches can provide a persistent work place within which cooperative work can be situated.

2) **Peripheral awareness**

   People in a space can establish a general awareness of what others are doing beyond their current main activity.

3) **Navigation and chance encounters**

   Informal communication plays a vital role in establishing and maintaining co-operating relationships. Spatial approaches attempt to support this issue through the notion of navigation within a shared spatial setting.

4) **Usability through natural metaphors**

   CSCW systems using spatial approaches can exploit people's natural understanding of the physical world, including spatial factors in perception and navigation, as well as general familiarity with common spatial environments, in order to construct cooperative systems which can be more easily learned and used.

Reviewing existing CSCW systems, there are two main groups of systems which utilize spatial metaphors: *information spaces* like TeamRooms [Lee96, Greenb98, Rosema96b] directly supporting groupwork by providing a place to share partial results, and *social virtual worlds* like media spaces, MUDs and MOOs and virtual reality systems mainly supporting community interaction of people. These two groups have developed independently. In this chapter, I will first examine these two different kinds of spatial approaches and then try to find a way to incorporate them so that they support both community interaction and groupwork in educational settings.

4.2 Spatial Approaches to Community Interaction

In chapter 2, I defined *community interaction* as the sum of activities between community members expressing their concerns, feelings, interests etc. Moreover, some mechanisms to facilitate community interaction have been discussed, including recommender systems, agent-based systems, and so on. The following is intended to show how to use spatial approaches for facilitating community interaction.

4.2.1 Social Worlds for Communities

The pervasiveness of Web and Internet technologies has enabled a wide range of communities emerging from Cyberspace, some of which are nurtured, for example, by email systems

or agent technologies (see section 2.3.4 on page 26). In contrast, social worlds, which inherit the sense of locality of communities and are emerging as a kind of network places in the Internet, exploit the potentials of spatial approaches to support informal, awareness-rich, and serendipitous interactions between community members.

As discussed in chapter 2, community is traditionally seen as a local form of social group in the sense that it is based on bounded and relatively small-scale set of relationships [Mynatt98]. New technologies have fostered new forms of communities by dramatically diminishing the importance of the old local boundaries. Social worlds are persistent virtual spaces in the Internet for communities.

By the notion of *social worlds*, I mean systems based on spatial approaches for community interactions. In Strauss' Theory of Action/Interaction [Fitzpa95], a social world is

> *"an interactive unit that arises when a number of individuals strive to act in some collective way, often requiring the coordination of separate perspectives and the sharing of resources. It has at least one primary activity site where activities occur..."*.

While this definition makes clear that a social world has at least one primary activity site, it does not differentiate the worlds of communities from those of groups and teams. O'Day gives more specific definition of *social (virtual) worlds* [ODay96] with the emphasis on their distinguishing roles for community interactions. I will also refer to this kind of social world as community places or community worlds in the thesis.

Definition 4.2 (Social Worlds after O'Day)
Social virtual worlds do not emphasize support for specific group activities or tasks and often are not particularly intended for work. Instead, they offer places for people to be together. They support informal communication, shared construction and use of objects, etc.

Here both the terms *places* and *sites* are used to create a sense of being together for community members who are physically distributed across the real world. For example, media spaces are multimedia environments connecting geographically dispersed spaces which create a new sense of locality by bridging these separate spaces. And MUDs are computation-based environments that provide access to a persistent online *world*. These social worlds are also called *network communities* [Mynatt98], *network places* [Nichol95] etc.

4.2.2 Facilitation of Interaction

Given these social worlds, I will show how they are used for community interactions by attempting to answer the following three questions:

- How can they offer awareness information?
- How can they tackle privacy issues?
- How can they afford informal interactions?

Awareness

Information about community members, such as who is around, what sorts of things they are doing, whether they are relatively busy or can be interrupted, will play an essential role in facilitating community interaction. Such information is called *awareness* in CSCW systems.

Awareness implies "*knowledge of something through alertness in observing or in interpreting what one sees, hears, feels etc.*" [webste96]. For example, people maintain awareness of an association of people, their reasons for being together and their shared knowledge, which is called *organizational awareness* [Conkli88]. Another example is *task awareness*, all of which involves understanding the purposes of a task, the specific goals and requirements of the group in pursuing the task, and how the task on hand fits into a larger plan.

In groupware systems, awareness has been extensively discussed as knowledge with respect to other people's activities in collaborative environments [Douris92b, Gutwin96c, Gutwin96d, Greenb96a]. According to Dourish:

Definition 4.3 (Awareness)
Awareness is an understanding of the activities of others, which provides a context for your own activity [Douris92b].

Dourish explains further that for groupware systems, this context is used to ensure that individual contributions are relevant to the group's activity as a whole, and to evaluate individual actions with respect to group goals and progress. The information, then, allows groups to manage the process of collaborative working [Douris92b]. Apparently, Dourish defines a generic term of awareness for groupware systems where a group of people are involved in a collaborative working process.

To understand what awareness means in the context of communities, it is helpful to look at a fine-grained categorization of awareness. Gutwin *at al* distinguish four categories of awareness in CSCW systems, which include [Gutwin96d]:

Informal awareness of a work community is the general sense of who is around and what they are up to – the kinds of things that people know when they work together in the same office. Informal awareness is the glue that facilitates casual interaction.

Social awareness is the information that a person maintains about others in a social or conversational context: things like whether another person is paying attention, their emotional state, or their level of interest. Social awareness is maintained through conversational cues such as back-channel feedback, and through non-verbal cues like eye contact, facial expression, and body language.

Group-structured awareness involves knowledge about such things as people's roles and responsibilities, their positions on an issue, their status, and group processes.

Workspace awareness is defined as the up-to-the minute knowledge a person requires about another group member's interaction with a shared workspace if they are to collaborate effectively.

Therefore, the first two kinds of awareness (i.e., informal awareness and social awareness) are, then, extraordinarily vital to community interaction because they do not involve a cooperative working process but rather exchange of the concerns, feelings, and interests of community members. As stated in [Douris92c], *" Awareness may lead to informal interactions, spontaneous connections, and the development of shared cultures"*. In other words, awareness helps to build a community.

Two kinds of spatial approaches, *Media Spaces* and *Virtual Reality Systems*, should help to demonstrate the potentials of awareness for community interactions.

Media Spaces

A media space is a system that uses integrated video, audio, and computers to allow individuals and groups to work together despite being distributed spatially and temporally. In contrast to video-conferencing systems, media spaces use long-lasting connections that show continuous video or snapshots of offices and common areas at a remote site. That is that a media space is usually constantly in use, functioning like an extension of physical space. Bly *et al* argue that media space *"was not something that was turned 'off' or 'on' during the day, but was continually available"* and define media space as:

> *"an electronic setting in which groups of people can work together, even when they are not resident in the same place or present at the same time. In a media space, people can create real-time visual and acoustic environments that span physically separate areas. They can control the recording, accessing and replaying of images and sounds from those environments."* *[Bly93]*

The best known examples of media spaces include Xerox Parc Media Space [Bly93], CRUISER [Root88], RAVE [Gaver92b] and CAVECAT [Mantei91] systems.

Bly *et al* summarize potential functionalities which media spaces can provide, such as peripheral awareness, chance encounter, locating colleagues, video phone conversations, group discussion, presentation, and social activities [Bly93]. In spite of these numerous application potentials, it is suggested that the affordance of *peripheral awareness* was perhaps their most powerful use because media spaces provided an overview of who was around and what was happening, and thus offered the possibility of joining in [Bly93, Douris92c]. Let us take RAVE as an example to show how peripheral awareness is provided in media spaces.

In RAVE, five buttons are designed to allow diverse levels of engagement, from general awareness to a remarkable degree of focused collaboration. The first three buttons are primarily used to offer peripheral awareness:

- *background* button allows people to select a view from one of public areas for displaying on their monitor;
- *sweep* button makes short one-way connections to various nodes (i.e., offices) in the environment (i.e., one can sweep all nodes or a subset of relevant ones);
- *glance* button allows more focused attention to particular colleagues by making a single 3-second one-way connection to a selected node;
- *vphone* and *office share* support more focused interactions, both of which use a two-way audio and video connection which allows colleagues to engage in the video equivalent of a face-to-face conversation. Vphone serves short conversation while office share connection lasts longer.

The rationales underlying these buttons can be illustrated from two points. Firstly, video/audio connections in media spaces provide users with more direct and reliable awareness information about who is around and what is happening. This awareness serves as the basis for community interaction. Without the aid of this awareness, community interaction between geographically distributed people should be supported through other indirect methods. Nevertheless, with a camera in the office, use of media spaces has raised serious issues concerning privacy. Therefore, an important design concern underlying these buttons is how to constrain the use of media spaces, i.e. both to support and encourage their use in ways that enhance existing work practices and to discourage possible misuse (e.g., spying and monitoring, etc) [Gaver92b]. Later in this section, several key issues regarding privacy will be discussed in more detail.

Virtual Reality Systems

In contrast to media spaces, virtual reality systems offer awareness for facilitating community interactions by making use of some common properties of spaces like *position*, *direction* and *distance of objects*, which can be demonstrated by the systems DIVE and FreeWalk.

DIVE (Distributed Interactive Virtual Environment) In an attempt to manage interactions within heavily populated spaces which comply with natural social conventions, Benford *et al* introduce a spatial model by defining key concepts like *"aura"*, *"focus"* and *"nimbus"* [Benfor93b].

Aura is defined to be a sub-space which effectively bounds the presence of an object within a given medium and which acts as an enabler of potential interaction. Objects carry their auras with them when they move through space. As soon as two auras collide, interaction between objects in the medium becomes a possibility.

After an aura has been used to determine the potential for object interactions, the objects themselves are subsequently responsible for controlling these interactions. This is achieved on the basis of quantifiable levels of awareness between them. The awareness levels are

calculated from a combination of nimbus and focus. The concrete approach to calculation of awareness depends on particular applications. This spatial model has also been applied to a text conferencing system CyCo.

Apparently, DIVE tries to simulate how people initiate a conversation in everyday life. When two persons approach more closer towards each other, a conversation between them is probably about to start. To get the right feeling about an impending conversation, however, does not only depend on spatial factors of people. For example, the expressions in people's eyes may contain the key for them to interpret the spatial meanings correctly, which might pose a challenge to these systems merely based on spatial properties.

FreeWalk FreeWalk is a desktop meeting environment to support informal communication [Nakani96]. As stated before, most community interaction is informal communication. FreeWalk provides a 3-D *community common* where people can meet with each other and can behave just as they do in real life. In these kinds of community commons, casual interaction is supported. This kind of community commons are used to primarily support casual interaction which can be characterized by accidental encounters, unlimited participants, and unpredictable topics of conversation.

To enter this 3-D community common, people just need to specify the IP address of the systems server. Within this system, each participant is represented as a pyramid of 3-D polygons. His live video is mapped on one rectangular plane of the pyramid. The most remarkable features of FreeWalk are that participants standing far away appear smaller and those near are larger, and that voices are also transfered under the same policy, i.e. voice volume is proportional to the distance between sender and recipient.

The decisive spatial property used in FreeWalk is *distance* between people. Distance facilitates conversations, and distance is used to group conversations.

4.2.3 Privacy Control and Spatial Solution

In a community environment, people must depend on some mechanisms to get awareness about the others. In media spaces, video is, for example, used to learn who is around and what someone is doing, which, however, might raise serious privacy concerns. Privacy is a very important issue of CSCW systems, and this thesis is expected to pay more attention to it although it is difficult to define the notion in more than an intuitive fashion [Bellot93]. To begin with, I discuss this issue by showing how media spaces have addressed this issue and how spatial factors help prevent violations of privacy in media spaces. In the remaining parts of the thesis, this topic will be taken up again.

Privacy in Media Spaces

Research of media spaces has raised a number of issues concerning privacy, protection and control. A typical privacy problem in media spaces is *intrusiveness* [Douris93, Mantei91, Gaver92a].

Reviewing existing literature regarding privacy issues in media spaces [Obata98, Douris92c, Bornin91], I find that there are four major models intended to diminish the negative impact of using audio/video connections on users' privacy: *open model*, *protecting model*, *reciprocal model* and *approaching model*. For the convenience of discussion, I first define a *caller* as the one who wants to contact someone by using video or audio connections, and the *recipient* is the one who is to be contacted by the caller.

Open Model Xerox PARC's media space was probably the earliest example of media spaces [Bly93]. This system takes no explicit measure to protect privacy of users (see Figure 4.1). A recipient who did not want to be disturbed could only point the camera out of a window, or simply turn off the microphone [Douris93]. Users of this media space are exposed to the "monitoring" of cameras.

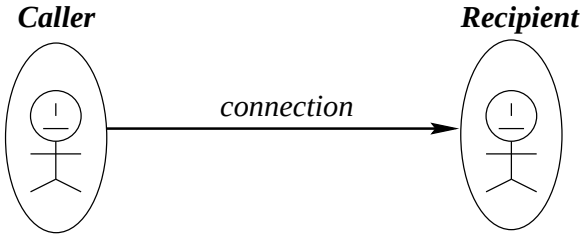


Figure 4.1: No protection of privacy for recipients

Protecting Model Meanwhile, numerous mechanisms for users of media spaces to keep them from intrusiveness have emerged (see Figure 4.2). Typical mechanisms of this kind include:

- image protection of recipients in CRUISER [Root88]

  The members of the CRUISER media space have a simple way of indicating their level of accessibility through the use of computer-generated bars which appear on their video image. These bars (called "blinds" by Root) can overlay the whole or partial image of the recipients, indicating to the caller whether or not the recipient is now accessible for conversation.

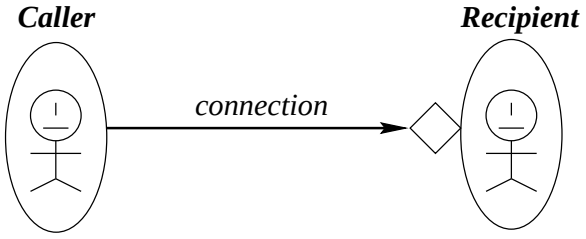


Figure 4.2: There are protection mechanisms for recipients

- accessibility rules and door states in CAVECAT [Mantei91]

CAVECAT users can create rules for connections which define, for example, from whom a recipient wants to accept what connections (short-term, one-way, video). Moreover, certain common rules are represented by accessibility states through an iconic interface which draws on the metaphor of office doors. Full accessibility is suggested by an open door; a door which is ajar makes the recipient less accessible; a closed door requires more permission, and a locked or barred door indicates that the recipient is completely inaccessible.

- agent and auditory cues in RAVE [Gaver92a]

  In RAVE, an agent component is used as a gatekeeper for recipients which responds to requests for audio and video connections made by callers. The agent embodies recipients' privacy within the system. A caller cannot glance at a recipient unless the agent has been consulted. Rules for the agent are set by control panels which enable users to select those who will or will not be given permission to glance or to use vphone, office share connections etc.

  Further, there is a mechanism in RAVE to provide feedback to recipients on the state of connections. This feedback takes the form of non-speech audio cues or pop-up messages on the screen. Normally there are three feedback parameters: *before*, *after* and *inform*. For example, *before parameter* specifies some action which should take place before the actual video connection is made. A typical action for before parameter might generate the sound of an opening door.

Reciprocal (Symmetrical) Model In this model, all connections are reciprocal; that is, when a caller can see or hear a recipient, the recipient can see or hear the caller at the same time (see Figure 4.3). This principle does not mean that the recipient *must* see the caller if the caller sees the recipient, only that the recipient has that option if he so desires. This model is used by CRUISER, POLYSCOPE etc [Bornin91, Gaver92a]. In fact, no protection measure is taken for recipients' privacy in this model. Users of the reciprocal model seek to be treated "fairly".

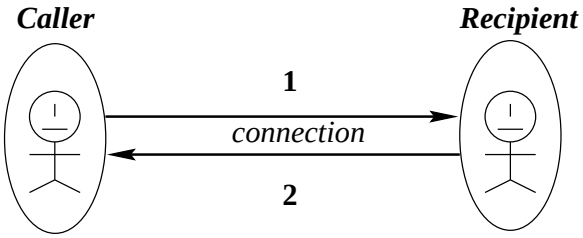


Figure 4.3: Connections are reciprocal or symmetry to recipients and callers

Approaching Model A very natural way to deal with intrusiveness in media spaces is to enable a caller to approach a recipient from public place instead of directly jumping into the private place of the recipient (see Figure 4.4). The sense of distance plays a key role in this model. OfficeWalker [Obata98] is such a system. As an example to use spatial factors to protect privacy, this model will be discussed subsequently in more detail.

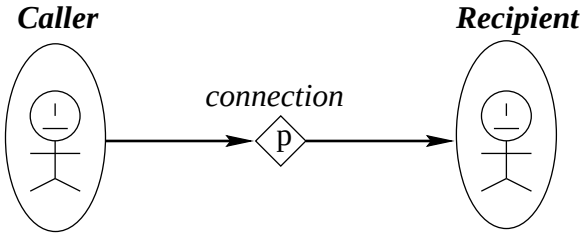


Figure 4.4: A caller approaches a recipient from public place to private place

Note that many media spaces use more than one of these mechanisms listed above. Although these mechanisms of protecting privacy stems from media spaces, they can certainly give us reasonable hints on protecting privacy in CSCW systems.

Approaching Model as a Spatial Approach to Privacy Protection

Taking the approaching model as an example, I turn to examine how spatial factors can help prevent violations of privacy in media spaces and in communityware.

Obata *et al* claim that the problem of intrusiveness is caused by *"eliminating the sense of distance that afford appropriate behavior and social norms since people usually communicate with others from a distance that is appropriate to their mutual relationship"* [Obata98]. Indeed, people possess unspoken proximity rules in everyday life that specify the amount of distance which is considered to be appropriate in a relationship.

Physical proximity has long been known to be an important factor in the development and maintenance of social relationships. In general, the shorter the distance between two people the greater the probability that they will form a social relationship, communicate, or enter into a working collaboration [Root88]. Nishide categorizes the distance into five zones according to the possibilities for starting a conversation. Due to the particular interest in spatial approaches, I cite these categories as follows [Obata98]:

1) The zone of exclusion, which ranges from body contact to a distance of about 50 cm. Others are generally not allowed to enter this zone. Usual conversations do not occur at this distance.

2) The zone of conversation, which ranges from approximately 50 cm to 150 cm. In this zone, conversations take place. Formal conversations occur in a range of more than 80 cm. When this zone is entered, a conversation is mandatory.

3) The zone of Proximity, which ranges from approximately 1.5 m to 3 m. This is the zone to approach a recipient for a conversation. However, it is possible to enter this zone without starting a conversation for the time being.

4) The zone of mutual recognition, which ranges from approximately 3 m to 20 m. It is possible to recognize acquaintance and their facial expression at this distance. Greetings usually occur in this zone. Ignoring an acquaintant is difficult when the range between 3 m and 7 m is entered.

5) The zone of recognition, which ranges from approximately 20 m to 50 m. It is possible to recognize acquaintance, but greetings rarely occur at this distance.

In spite of the significance of distance in real world, social virtual worlds have appeared to intentionally ignore it. For example, Root recognized the importance of spatial distance, but ignored it completely in the design of the CRUISER system, arguing that *"the notion of spatial proximity is meaningless in the virtual world, all users being equally proximate"* [Root88]. In contrast, OfficeWalker exploits the sense of distance to tackle intrusiveness in communities. Let us have a close look at the social world of OfficeWalker and the procedure of initiating a conversation.

A community world in OfficeWalker consists of the following components:

- *a hallway* is a public (virtual) place shared by neighbors, who can be defined independent from their physical locations;
- *private places* are used by these neighbors as their own offices.

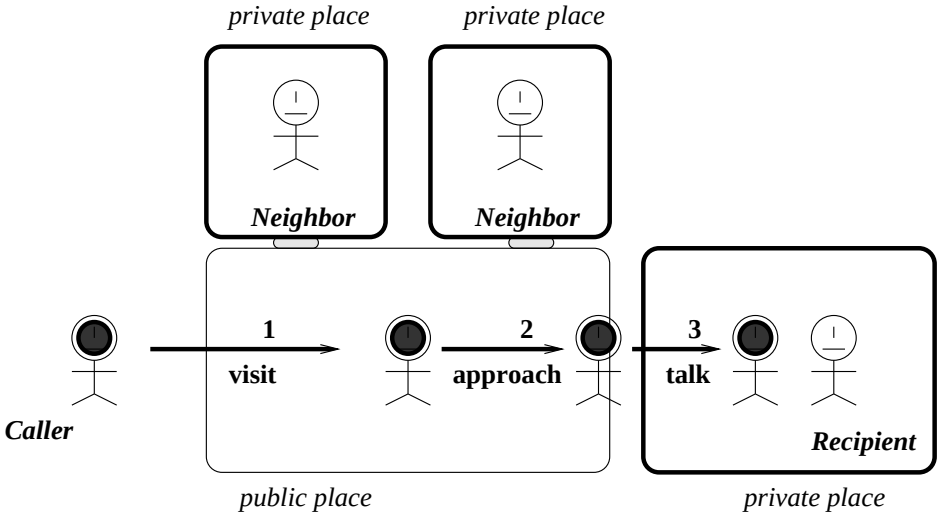


Figure 4.5: A caller approaches a recipient from a public place in OfficeWalker

The procedure of a caller approaching a recipient, then, can be described as follows (see Figure 4.5):

1) A caller first visits a recipient's hallway (the public place), and the recipient and other neighbors can see the caller's video image from their private places. In this phase, the recipient does not know whom the caller wants to contact. There is no social need to respond to the caller. The caller can wait until the recipient is available without interrupting his activities if the recipient looks busy.

2) As the caller decides to contact the recipient, the caller approaches the private place of the recipient. The caller and the recipient can see now each other on *closer* view.

3) At this time, the recipient is aware of the coming caller. Staying at this distance, it is uncomfortable for both of them to refuse a conversation.

Drawing on this example, I find that there are two spatial factors which play essential roles in protecting privacy of community members. One is *place* (public place and private place). The implicit rules people apply to public places differ from those applied to private places. When a person enters into a public place already occupied by another person, all they have to do is either greeting each other or ignoring each other completely. But in a private place, the host should say something to one who makes a visit to him in the office. Another spatial factor is *distance* between community members. As discussed above, there are generally recognized distance rules needed by partners of conversations. Violating these rules will cause intrusiveness.

4.2.4 Informal Interaction

There is a large variety of notions like *intended* and *unintended, planned* and *unplanned, formal* and *informal, explicit* and *implicit,* for describing interactions between people, mostly according to how these interactions are initiated. In addition, we can also find terms like *casual, lightweight, spontaneous, opportunistic* to serve the same intention [Nunama91, Root88, Isaacs96, Edward94, Whitta97]. Thus, it is necessary for us to exactly discern their essential meanings with regard to community interactions in spatial approaches.

Community Interaction in Social Virtual World: Explicit, Unplanned, Opportunistic and Spontaneous

For the convenience of discussion, I first introduce several terms to characterize interactions. Kraut distinguishes between four categories of interactions [Isaacs96]:

- *planned*: prearranged meetings;
- *intended*: explicitly sought by one person;
- *opportunistic*: anticipated by one party but occurring only when the parties happen to see each other; and
- *spontaneous*: unanticipated by either party.

According to these categories, most community interactions in spatial approaches are *explicit, unplanned, opportunistic* and *spontaneous*. Let me examine them respectively.

explicit Firstly, attending community interactions in spatial approaches involves an explicit process. For example, people in media spaces can notice what the others are doing with the help of audio and video connections and then find an appropriate chance to contact them. This is an explicit process. Both Abel and Kraut have observed the main weakness of these explicit attempts, saying that *"this kind of explicit processes has brought together people who do not necessarily have a context or need to interact, and placed them in a situation where it is awkward to ignore each other"* [Abel90, Kraut90]. Likewise, interactions in virtual reality systems and MUDs are usually explicit, too, because users of these systems must explicitly attempt to enter these places which are *disjoint* from their real workplaces. Users of FreeWalk must explicitly take steps to enter into the virtual world through a relevant IP address.

Thus, to some extent, the spatial approaches' ability to support community interactions are limited by the inherent feature of spaces. In contrast, agent systems can bring together people who have not taken explicit steps to contact each other.

unplanned Secondly, in spite of the explicit attempts, community interactions occurring in spatial approaches are usually unplanned, which is in keeping with the nature of communities that community members do not know each other personally. Thus, they usually can neither foresee nor plan activities in advance.

opportunistic and spontaneous Further, community interactions comply with the definition of opportunistic and spontaneous ones by Kraut. On the one hand, a person could anticipate to meet someone in spaces. This anticipation can become reality only when they happen to see each other in the spaces. So it is opportunistic. On the other hand, many users enter into community worlds with no anticipations to see any specific person. Under this circumstance, interaction was not expected by either party. According to Kraut's definition, it is then spontaneous. In this thesis, I do not differentiate the opportunistic from the spontaneous interaction. Instead, I use another term, namely *casual interactions*, to refer to situations where persons happen to run into each other without considering whether it has been anticipated by either party of the interaction.

Definition 4.4 (Casual Interactions)
Casual interactions occur when people happen to run into each other without an intention in advance.

In summary, community interactions in spatial approaches are mostly explicit, unplanned, opportunistic and spontaneous. However, the initiation of sessions within community spaces (e.g., within MUDs) might be implicit and lightweight, which means that two or more persons could be involved in a conversation triggered by artifacts they are using at the same time. In next chapter, lightweight session management will be discussed.

Community Interactions are Informal

One of the key features shown by spatial approaches to communities like media spaces and virtual reality systems is that community interactions occurring there are informal. However, two questions need to be raised regarding informal interactions:

- What does the notion of *informal* exactly mean in CSCW systems?
- How important is informal interaction to communities?

The existing literature on organizational communication indicates that interactions in organizations are usually informal. According to Fish:

> *"Formal communication flows through official organizational channels, is often prearranged, and is typically conducted in a formal style. In contrast, informal communication cuts across these organizational boundaries and often happens spontaneously. In terms of styles, informal communication is often more frequent, expressive and interactive than formal communication." [Fish93]*

Whittaker *et al* put an emphasis on the *synchronous* character of informal interactions, saying that informal communications are those *"taking place synchronously in face-to-face settings"* [Whitta94]. No matter what the word *informal* means, most studies of informal interactions have recognized their common characteristics as being brief, unplanned, frequent and intermittent [Whitta94, Tang94, Fish93, Bornin91]. Thus, the last section's discussion leads us to the conclusion that community interactions in media spaces and virtual reality systems are informal, too.

Additionally, research of CSCW systems has shown the importance of informal interaction in everyday life. Isaacs' studies show that 92% of the interactions in two-office settings were not pre-arranged, although some of those were intended. These interactions were frequent and very short; 31% of workers' time was spent in informal interactions, and they lasted under two minutes on average [Isaacs96]. Kraut *et al* point out the role of informal interaction for generating collaborative relationships. Fish *et al* demonstrate that informal interactions are especially important to the less directly task-oriented aspects of organizational membership (e.g., learning the organizational culture, becoming loyal to an organization, making judgment of others, and forming relationships) [Fish93].

Drawing on all of these discussions, there are three critical points which can be concluded:

- Informal interactions dominate everyday communications between people;
- Community interaction in spatial approaches are mostly informal;
- Informal interactions help to develop collaborative relationships.

Spatial approaches can facilitate these kinds of informal interactions because of the specific roles of objects' proximity in them. When people are physically close to one another, there are many opportunities for them to engage in interactions. It suggests that spatial approaches which just create a sense of physical proximity are appropriate to informal and then to community interactions.

4.2.5 MUDs and MOOs

In contrast to media spaces and virtual reality systems which offer users fancy multimedia environments, MUDs and MOOs support community interactions in a very different way. Firstly, most MUDs are entirely text-based. Secondly, in addition to interactions among users, users' interactions with objects play an important role in MUDs, too.

A MUD's World

For historical reasons, MUDs stands for "Multi-User Dungeon" mainly because the first MUDs were networked multi-player dungeons and dragons games. Since many current applications of MUD technology have moved far from their origins as violent games, some people prefer to say that the "D" stands for some other word such as "Domain" or "Dimension" [Bruckm97, Douris98, Curtis92]. MOOs are object-oriented MUDs (MUD, Object Oriented). MOOs normally have an object-oriented style and allow for varying degrees of programmability, modification of rooms, interconnections, furnishings and virtual artifacts in the systems.

A MUD consists of three primary components: *Personas*, *rooms* and *objects*.

- Personas

  When one (a player) first connects to a MUD, he must pick a name. "Players" may also choose their gender and describe what they look like. Curtis observed that the most significant social factor in MUDs is the perfect anonymity provided to the players, which would encourage players to actively attend conversations in MUDs.

- Places

  The virtual world of MUD is structured into different *"rooms"*. When one says something, it is heard by others in the same room. *"Exits"* are used to connect rooms together.

- Objects

  MUDs are filled with objects. Players can create their own objects, or manipulate objects created by others.

In general, MUDs are a kind of Internet-accessible virtual world with its own geography, characters, and objects of all kinds [Curtis92]. In other words, they are computation-based

environments, providing access to a persistent, online "world" which can generate a sense of a known virtual place that can be inhabited and shaped by an emerging community. In these systems, when people join the community, they typically start by creating a character and building a home for themselves. With simple commands, participants can move from one locale to another, talk to other people, manipulate objects they encounter in each place, and extend the world by creating and describing new places and objects as well as behavior. In summary, MUDs:

- divide the universe into collections of essentially independent rooms
- provide facilities for tailoring these rooms in various ways
- provide facilities for navigating among these rooms
- support communication (text-based or multimedia) between participants
- support manipulations of objects located in the rooms

According to the definition of spatial approaches (see section 4.1 on page 67), all of these critical features of MUDs suggest that they belong to typical social worlds.

MUDs: Evolving Social Worlds

MUDs are a kind of continuously evolving social world. On the one hand, MUDs themselves are technologically evolving. On the other hand, relationships between their players in MUDs are evolving as well. A MUD is originally an entirely text-based virtual reality system that accepts connections from multiple users across the network and provides to each user access to a shared database of *rooms, exits* and other *objects* [Curtis92]. MUD players talk with each other by using simple commands like "say", "emote", "whisper" etc. In the past few years, MUDs have evolved continuously from these kinds of entirely text-based systems to those supporting multimedia communications. These typical examples include Jupiter [Curtis95] and wOrlds [Kaplan97]. Jupiter is a hybrid MUD and media space which was developed at Xerox PARC and is available to members of Xerox research and development community. Likewise, wOrlds also supports audio/video links between participants. A difference between them is that Jupiter supports graphical representations of MUD objects as well.

MUDs are also evolving in architectures. Most MUDs like Jupiter are generally implemented on a central server which maintains the state of the MUD world and the objects in it. In addition, the server normally includes an interpreter for the internal programming language, and a core set of objects that provides the basic functionality of network places, communication facilities, and programming tools. In contrast to Jupiter's centralized architecture, wOrlds is a distributed MUD [Kaplan97], based on the Object Request Broker (ORB) technology [Orfali96].

Furthermore, the relationships between players of MUDs are evolving, which is perhaps the most striking characteristic of MUDs as a social virtual world. MUDs are proven to be typical social worlds because some social phenomena were observed in them. For example, commands which are made available in MUDs, such as “whisper”, “smile” , are primarily designed to enable people to develop a kind of social relationships, expressing their feelings and exchanging their experiences, and so on. This kind of relationship development, Curtis reports, has led to wedding ceremonies held in MUDs. Note that these kinds of intimate relationships grow up from early anonymous conversations between players who have very different backgrounds.

MUDs for Education

MUDs have been introduced for education in the past few years. Typical examples include Pueblo [ODay96], a MOO supporting an elementary school-centered learning community, and MOOSE Crossing [Bruckm97], a text-based virtual world developed by MIT Media Lab for kids. The latter is of more interest to me because this system depends on a theory called “Constructionism” which emphasizes the role of construction activities for learners in the *learning process*.

Constructionism, a term coined by Seymour Papert [Bruckm97], argues that knowledge is *“built by the learner, not supplied by the teacher. ... this happens especially felicitously when the learner is engaged in the construction of something external or at least sharable...”*. More importantly, the construction activities should happen in communities. Bruckman writes:

> *“... community and construction activities are mutually reinforcing. Working within a community helps people to become better dancers/programmers/designers and better learners. Conversely, working on design and construction projects together helps to form a strong, supportive community.” [Bruckm97]*

Thus, MUDs are much more than chat rooms. A constructionist approach to virtual community design seeks to maximize each individual’s opportunities for creative expression and active participation and provides well-designed software tools which have a low initial barrier to use. The critical part to the software tools is a language which makes these creative activities possible.

4.2.6 Summary

In conclusion, I find that there are two classes of spatial approaches intended to support community interactions.

The first class includes systems which use video/audio to connect physically distributed workplaces. There are two types of these systems: *media spaces*, which have been illustrated in the preceding section, and *desktop video systems* like Piazza [Isaacs96]. In contrast

to media spaces, Piazza is integrated into the desktop applications. In particular, Piazza provides opportunities to encounter others through tasks and activities *accomplished on-line*. The *Encounter* component of Piazza enables people to be aware of and easily contact other people who are conceptually "nearby", which means that they are looking at the same data at the same time using the same application. Such "lightweight" interaction will be discussed in more detail later in this chapter.

The second class of systems for community interaction make use of virtual places which are *disjoint* from people's real workplaces: *virtual reality systems* and *MUDs and MOOs*. In these systems, users temporarily leave their real workplaces and enter a visual social world to communicate with others.

The common characteristics of these spatial approaches are that spaces and spatial properties of objects (e.g., position, direction and distance) can offer awareness. Awareness plays an important role in facilitating community interaction. Dourish *et al* put it in the following way:

> *"... awareness may be a useful basis for community access (an information tool, especially for locating colleagues) and for community building (a shared space for 'sightings' and personal snippets). In particular, this second usage helps maintain working relationships in a group which would otherwise have few direct interactions." [Douris92c]*

Moreover, community interactions in spatial approaches are mostly explicit, casual and informal.

4.3 Spatial Approaches to Groupwork

Spatial approaches to groups are primarily intended to support collaborative processes which involve close coordination between group members. Typical spaces for groups are *teamrooms*.

4.3.1 Teamrooms for Groupwork

Johansen describes how team rooms have become an important tool used by groups to organize their work [Johans91]. Teamrooms provide a permanent shared space used by the group, which can serve as:

- a meeting room,
- a work area,
- a place to store documents that are needed by the group's projects, and more generally,

- as a venue for communication within the group [Rosema96b].

Certainly, teamrooms share many similarities to social worlds for communities. For example, both of them support casual interactions; teamrooms are also populated by persons and artifacts, and so on. Nevertheless, teamrooms are distinct from community social worlds in the sense that teamrooms put great emphasis on collaborative work of groups whose members know each other and work closely towards a common goal. In such teamrooms, a specific group culture dominates activities within the environment. Group members abide by elementary rules of group conduct. Their work results are to be protected from others. In general, teamrooms are not expected to foster exchanging of concerns, feelings and experiences between group members but rather provide direct support for collaborative process such as collaborative document editing.

A spatial approach for groups consists of the following major components:

- Rooms

  First, a spatial approach usually consists of several separate rooms, each of which is intended to support a group's work and provides shared data objects and tools to manipulate the objects in collaborative sessions. When a room is created, some basic artifacts are made available by the system to offer elementary facilities to users.

- Tools

  Typically, both generic communication tools (e.g. chat tool, shared whiteboard) and a number of applets are contained in the rooms to support groupwork. Popular applets in teamrooms include diagram tools, outline tools, brainstorming tools, browsers for information such as Web pages, notification functions, as well as items such as games [Rosema96b]. Some systems have also incorporated facilities for communication over audio and video links. People connected to the place can see and manipulate artifacts within that place.

- Users

  Many systems divide users into administrators and members. Others support also user roles such as observers [Lee96]. The administrator of a room is allowed to change the access control to the room and to the objects within the room.

- Data

  Data are objects which can be manipulated by group members in a shared workspace such as documents.

The components in groupware systems differentiate themselves from those in communityware systems. For example, users in social worlds for communities have no specific roles. Rather, they usually have equal rights. A detailed comparison between teamrooms and community worlds will be given in 4.4 on page 93.

4.3.2 Awareness Support for Groupware Systems

Awareness information is always required to coordinate group activities, whatever task domain is tackled. As shown before, informal awareness and social awareness are critical to facilitation of community interactions while group-structured awareness and especially workspace awareness play a particular role in enabling groupwork between group members.

One major focus of groupware development has been the creation of *virtual shared workspaces* in distributed computer environments. Ellis *et al* defines groupware as the *"computer-based systems... that provide an interface to a shared environment."* Whiteboards and overhead projections are examples of shared workspaces in face-to-face meetings. According to Ishii, shared workspace activities include sharing information, pointing to specific items, marking, annotating and editing [Ishii93a].

Workspace awareness is defined as the up-to-the minute knowledge a person requires about another group member's interaction within a shared workspace if they are to collaborate effectively, which means that workspace awareness provides people with perspectives on the current state of their collaborative work. Workspace awareness falls into two types: synchronous and asynchronous. While synchronous awareness enables real-time collaboration distributed over space, asynchronous awareness enables collaboration distributed over time as well as space. Asynchronous workspace awareness allows shared workspace systems to support flexible and dynamic work styles [Nomura98].

Synchronous Groupware Systems

In synchronous groupware systems, a user must be able to keep track of others' current activities in order to get a context for coordinating his own work. Table 4.2 lists elements and answers to questions which Gutwin think are of particular importance to collaborative work in share workspaces [Gutwin96b]. Note that these questions relate to states of collaborative activities which *are occurring* in shared workspaces.

Mechanisms to offer these kinds of awareness information have been extensively discussed in CSCW systems. Typical mechanisms include:

- Audio/video connections

  Certainly, audio/video connections offer direct communication channels among participants of collaborative tasks. In audio/video conferencing, participants can easily get information about who are present in the meeting and what the others are doing.

- WYSIWIS (What You See Is what I See)

  Another obvious way to take care of the need of awareness is to build a WYSIWIS environment. Because the WYSIWIS concept attempts to enable all participants of a shared workspace to have the same view of objects (the same widgets, the same telepointers) manipulated by any of them, the presence and action of participants are visible. However, WYSIWIS environments are rather limiting, in that, for example,

Element	Relevant Questions
Identity	Who is participating in the activity?
Location	Where are they?
Activity Level	Are they active in the workspace?
Actions	What are they working on? What are their current activities and tasks?
Intentions	What are they going to do? Where are they going to be?
Changes	What changes are they making? Where are changes being made?
Objects	What objects are they using?
Extents	What can they see?
Abilities	What can they do?
Sphere of Influence	What effects can they have?
Expectations	What do they need me to do next?

Table 4.2: Elements of Workspace Awareness for Synchronous Groupware

they prevent customized or private views. Consequently, there has been an interest in how to offer awareness in "relaxed WYSIWIS" or even "non-WYSIWIS" environments [Stefik87]. In general, strict WYSIWIS emphasizes identical views of all shared objects, while relaxed WYSIWIS requires only identical views of public objects.

- Radar Views

  In some systems, a mechanism called *Radar* [Gutwin96a] helps to provide information about who is currently working within the shared workspace, or offers information about other people's interaction with the workspace.

- Telepointers

  Telepointers [baecke93a, Loge94, Gutwin98] provide a fine-grained sense of awareness of other users' actions. Normally, synchronous groupware systems can include one or several distinguished cursors which are used as reference to information in public windows.

Asynchronous Groupware Systems

The elements listed in table 4.2 also constitute the critical points of awareness information in asynchronous groupware systems. However, asynchronous systems put great emphasis on what *has been* done and what *is to be* done with regard to shared workspaces.

In existing CSCW systems, two approaches to the provision of awareness information for asynchronous groupware can be identified: *informational approach* and *role-restrictive approach* [douris92a]:

Informational approach This mechanism offers awareness information by providing *explicit facilities* through which collaborators can inform each other of their activities. Reviewing existing CSCW systems, I find that there are two types of such explicit facilities for supporting awareness in asynchronous systems:

- *Auxiliary Tools*
  For example, the primary mechanism available to Quilt users are a hypermedia system which represents the document along with *annotations*, *audit trail recording*, and *integrated electronic mail* and *conferencing systems* [Fish88]. With annotation, users can comment on each others activities. Electronic mail and conferencing systems enable users to discuss issues relevant to their common work and distribute information about their current and planned activities.
- *Event Services*
  As introduced in chapter 2, a BSCW workspace maintains a set of documents and records actions performed on documents in the space. The records of actions are accomplished through the *event service* which provides information of what has happened lately in each document space, as well as the current status of the space. Events are triggered whenever a user performs an action in the workspace. The system records the events and presents the recent events to each user as *event icons* in the workspace listing [Bentle97]. The current version of BSCW distinguishes between *new events*, *read events*, *change events*, *move events* and *touch events*.

Role-restrictive approach This second mechanism makes use of group members' roles in collaborative systems. It is well-known that a role describes an individual's relationships to the shared work objects and to other participants, and is typically linked to a set of operations which can be performed. For example, an "author" role in a shared authoring system might be associated with the read, write, create, delete and edit operations, while a "reviewer" might be limited to read and annotate. Then, people can obtain awareness information of who might have done certain operations to shared objects. The problem is that if roles defined in a system are associated with many common operations, the effect of this mechanism used to provide awareness will be decreased. Apparently, information provided by this mechanism relates to the *character* of the activity, rather than the *content*.

Artifacts for Awareness Information

In principle, all mechanisms discussed above for provision of awareness information can be exploited in spatial approaches to groupware systems. However, artifacts in spatial approaches can help to provide more intuitive information for collaborators to coordinate their

collaborative work. *Memos* can, for example, be used to note what has been done in shared workspaces; plans for future work can be observed by all group members with help of *a to-do* list in teamrooms. I will discuss the enormous potentials of various artifacts in facilitating group work in spatial approaches in the next chapter.

4.3.3 Seamless Transitions in Spatial Approaches

Seamlessness of CSCW Systems

In chapter 2, the space/time matrix of CSCW systems suggested by Johansen has been introduced. To tackle the issue of seamless transition in spatial approaches, it is helpful to fill the matrix with CSCW applications. The new space/time matrix illustrated by Table 4.3 shows that many groupware systems were designed to support *only limited* collaborative activities or situations. Consequently, when people move between styles of collaboration, they must switch from one groupware application to another. People refer to these physical or perceptual boundaries within CSCW systems as *barriers* or *seams* [Ishii93b, Ishii90, Greenb98].

Greenberg *et al* call these seams "gaps", arguing that these gaps in CSCW systems force people now to make fairly *heavy-weight* and *disruptive transitions* within and between software to move across these gaps. Some gaps identified in literature include [Baecke93b, Greenb98]:

- the gap between individual and shared work, where people have difficulty moving their working styles, and their artifacts between a personal working area and the group's working area;
- the technology gap that exists when groups use both conventional software and groupware, and when groups work in heterogeneous computer environments;
- the gap between synchronous and asynchronous work, where either different time or same time interaction is supported, but not both;
- the gap between different phases of a collaborative activity, where people need to move between different work tasks;
- the gap between same place and different places, where part of a group that is trying to meet are co-located in a single room, another part in another room, and the rest in their own offices;
- the gap between informal and formal activities;

In face of these seams or gaps, one major concern in CSCW systems is how to eliminate them to enable people to work *seamlessly*. The concept of *seamlessness* is raised to tackle such problems. According to Ishii:

	Same Time	**Different Time**
Same Place	*Face to face interactions* • conference tables with embedded computers • public displays • dedicated tools for e.g., voting and brainstorming	*Ongoing tasks* • teamrooms • group displays • shift work • project management
Different Places	*Distributed real time interactions* • chat systems • transparent sharing of single user applications • video conferencing • media spaces	*Communication and coordination* • unstructured or semi-structured electronic mail • electronic bulletin boards • asynchronous conferencing • list servers • schedulers • collaborative hypertext

Table 4.3: CSCW systems are divided into synchronous and asynchronous ones [Johans88]

> *"'Seamlessness', in the sense of eliminating unnecessary obstructing perceptual seams, has been a key concept of our evolving media design. Collaborative work, in particular, is marketed by spatial, temporal, and functional constraints that force us to shift among a variety of spaces or modes of operations. Seamless design undertakes to decrease the cognitive loads of users as they move dynamically across different aspects of their work." "*

The most notable obstacle which spatial approaches can help to remove is the gap between synchronous and asynchronous groupware systems. Spatial approaches can support both asynchronous and synchronous collaboration among group members in the sense that spaces and their contained artifacts are persistent[Greenb98]. In general, any object in the place continues to exist even after all users have left the place. Later visits to the place by one or more users will find the objects unchanged. In following sections, I illustrate how spatial approaches tackle the gap between synchronous and asynchronous work and the gap between individual and shared work.

Support Synchronous and Asynchronous Groupwork

Most groupware systems support either synchronous or asynchronous collaboration, but not both. From the typical space/time matrix of CSCW systems suggested by Johansen, we can see that there are several major categories of asynchronous systems. Typical examples include electronic mail, list servers and bulletin boards. Similarly, there are different categories of synchronous systems like video conferencing, media spaces, chat systems etc. It is therefore difficult for users to move fluidly across the synchronous/asynchronous barrier.

The room metaphor, which is widely used by groupware systems, offers a simple way for a group to move between asynchronous and synchronous work, which can be observed from two aspects:

- Connection Channel

  By using the room metaphor, group members who are present in a room at the same time, are automatically connected together. Unlike groupware applications that require people to create and establish separate connections between each tool and communication channel, a room acts as a single connection point. Systems like TeamWave [1] and CBE [Lee96] assume that the act of entering a room immediately connects all people within it, both for communication and for work.

- Container of Artifacts

  On the other hand, a room is also a container of artifacts. When people work in the room at different times, they can work asynchronously just by leaving their work in the room. In everyday life, some specific-purpose artifacts such as to-do lists, action items, telephone logs, and vacation schedules can coordinate activities among group members [Covi98]. Persistence allows working artifacts and annotations to be left in a room for others to review or change at a later time. Here spatial proximity can link these messages and artifacts.

Support Individual and Shared Work

Everyday working styles of people shift regularly and easily between individual and group activities, suggesting that CSCW systems should support different working styles, too. However, most existing systems distinguish between single-user application and multi-user groupware. By forcing them to switch tools, this introduces a barrier for people to move between individual and group activities.

In contrast, a room can be occupied by a single person or by groups. People can bring artifacts into a room, either for their own individual use, to leave them for others. They can also work on them together with others. If a person enters an already occupied room, the room and its tools are available for all persons present in the room. The feature that a room serves as a connection channel and a container of artifacts helps remove the barrier

[1] http://www.teamwave.com/

between individual work and shared work. Of course, one premise for the transition between individual and shared work is that all tools in the room must not only support group activity, but may also be used to individual work.

4.4 A Spatial Framework for Educational Settings

So far, the discussions in the first three chapters have led to two important results:

- The 3-C model depicted by Figure 3.9, which integrates the requirement to provide consistent support for different levels of communication within various collaborative entities by harnessing the potentials of both communityware and groupware.
- The examination of spatial approaches to communityware and groupware. Spatial approaches include social worlds fostering informal, opportunistic and unplanned community interactions, and teamrooms supporting groupwork and helping to overcome various seams inherent in groupware systems. Both in social worlds and teamrooms, awareness support and privacy protection are two critical issues.

The problem we face now is that spatial approaches to communities and groups have developed independently, which has diminished the powers of spatial approaches to support both community interaction and groupwork.

4.4.1 Independent Development of Spatial Approaches to Communities and Groups

The study in the preceding sections shows that there are two entirely different classes of spatial approaches. Spatial approaches to communities and groups have been developed with no apparent connection between each other in the past:

- **Spatial Approaches to Communities**
  - Spatial approaches to communities are referred to as *social worlds*. Typical examples include media spaces, virtual realities, MUDs and MOOs etc.
  - In these social worlds, informal, opportunistic and unplanned community interactions dominate all activities, fostering social relationship and supporting the exchange of feelings, interests and experiences among community members.
  - Community members normally do not know each other personally. In fact, they are quite willing to communicate with each other anonymously.
  - Informal and social awareness play a key role in facilitating community interactions within these social worlds.

  - Community interactions may lead to the forming of groups as soon as a preliminary confidence between community members has been generated and a closer working relationship is desired.

- **Spatial Approaches to Groups**

  On the other hand, spatial approaches to groups put great emphasis on support of a concrete collaborative process among group members:

  - Typically, a *teamroom* is used as a spatial metaphor for structuring the groupwork. In teamrooms, group members work closely towards a common goal, for example, editing of a shared document. Shared workspaces bring people together as they work either synchronously or asynchronously.
  - Persons in teamrooms usually know each other and share more common interests than in social worlds for communities.
  - Spatial approaches can be used to remove some barriers (or gaps) in existing groupware systems, especially the barrier between synchronous and asynchronous groupwork.
  - In teamrooms, group-structural awareness and workspace awareness are of particular importance to groupwork. Being involved in a collaborative process, group or team members need to learn more about the work of others in the same group or team.

Due to the independent development of spatial approaches to communities and groups, it is difficult to use existing spatial approaches to meet our pursuing goals embodied by the 3-C model described at the end of last chapter, i.e. providing consistent and seamless support for students and tutors in educational settings. Thus a new spatial framework has to be proposed to incorporate spatial approaches to both communities and groups.

4.4.2 A Spatial Framework

The new spatial framework should fulfill the following requirements:

- First of all, the new spatial framework should be a social world fostering informal, opportunistic and unplanned community interactions.
- This kind of community interactions can lead to the forming of groups.
- Newly created groups in the spatial framework can do their collaborative work.

Figure 4.6 presents a spatial framework for this intention. Apparently, the environment is expected to support:

- communities, groups and teams, and

- community interaction and groupwork

A very distinguished feature of this framework is that it can be used by diverse educational applications. In this dissertation, I take a lecture as a typical example of communities to explore how to support both community interaction and groupwork by using spatial approaches and artifacts located in them.

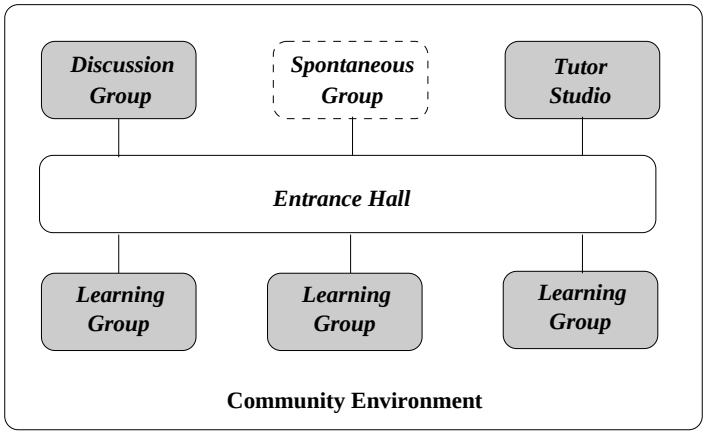


Figure 4.6: A spatial framework in educational settings consisting of Discussion Group, Learning Group, Tutor Studio and Entrance Hall

A Brief Tour of the Environment

The class community environment shown in Figure 4.6 consists of:

- an entrance hall;
- a discussion group;
- a tutor studio, and
- some learning groups.

As the lecture progresses new learning groups and spontaneous groups might be established. The sum of spatial units in the environment represents a lecture community. From the entrance hall, a student can go into the discussion group, any learning group, the tutor studio, or initiate a spontaneous group with other people whom he encountered in the entrance hall.

Discussion Group A discussion group provides an asynchronous interaction mode for lecture participants to discuss and solve lecture related issues.

Learning Groups A learning group is a team that is composed of several students who meet regularly for preparing and reviewing the lecture. Entrance into a learning group is protected by a password selected by the owner of the group. However, the tutor of the lecture is allowed to take part in team discussions without providing a password.

A learning group exports the following information about itself: (1) current room state representing who can enter the room; (2) participant list of all users currently present in the room; (3) date list of future sessions. Based on the exported information members of the lecture community might interact with learning group members and thus participate in team discussions. It can even lead to a change in team structure.

Tutor Studio It is assumed that a tutor occasionally participates in learning group discussions. Additionally there is a tutor studio which aims at increasing the collaboration possibilities between students and the tutor. In practice a tutor studio is similar to an office. Entrance into the studio is restricted to pre-determined consulting hours.

Like a learning group, a tutor studio exports information to provide people with the current state of the studio, data lists and contact information such as the secretary's telephone number.

Entrance Hall The entrance hall is furnished with a variety of information about activities occurring in other places on the server, such as in learning groups and the tutor studio. In particular, one can learn about others who are also in the entrance hall at the same time. Moreover, the entrance hall provides a medium for awareness and it facilitates all possible interactions in the lecture community:

- Taking advantage of the exported information of a learning group people in the entrance hall can attempt to contact team participants. If possible, they can join a team and take part in its discussions.
- People in a learning group can notice those who appear in the entrance hall. When a user has found a potential collaboration partner in the entrance hall, he can invite him to join the group.
- People who happen to meet each other in the entrance hall can initiate a spontaneous (temporary) group. If they agree to form a team for close and regular collaboration, they can also do so. Thus, the spontaneous group may change into a team (learning group).
- From the entrance hall one can also judge whether or not entrance to the tutor studio is permitted by checking the information on the door of the studio.
- Finally, one can enter into the discussion group to seek possible collaboration with others.

4.4.3 General Features of the Framework

Community Interaction

All spatial units in the framework constitute the community environment for a class, which means that community members are usually (*but not always*) allowed to enter any space of the environment, including entrance hall, discussion group and learning groups. They can encounter other persons in the entrance hall; they could initiate short lived (spontaneous) groups for themselves to conduct a private conversation; in learning groups they could view documents made available by the members of the group, and so on. In general, community interaction can take place anywhere in the environment.

Now, let us briefly have a look at what kinds of community interactions can happen in the entrance hall, in the discussion group and in learning groups.

- In the spatial framework shown above, the entrance hall is the most active space for informal interaction, where there is no restriction to activities of community members. The entrance hall serves as a chat room similar to those in many systems. More specifically, it also contains many specific artifacts to facilitate interactions. It can be seen as a facilitator of social relationships. Because of its feature as a "buffer", the entrance hall can help prevent intrusiveness while enabling casual interactions.
- In the discussion group, people communicate with each other through discussion of common subjects. This free discussion is a kind of community interaction.
- In learning groups, community members can take part in group discussions occurring there if they are welcome by members of the group. Even if no team member is in the room, community members can still have a look at the work made public to the community by the team. In this case, community members can also make contributions to groupwork by leaving notes in the room, such as answers to questions. In general, community interactions within teamrooms are asynchronous.

Forming of Spontaneous Groups

People have the chance, through contacts in the entrance hall, in the discussion group or in other learning groups, to get to know others who have share their interest in keeping a more intimate relationship. Often, they need to build short lived spaces for them to conduct private activity. Probably, the short lived group can bring them into a long-term collaborative entity which inhabits the community environment as well. A spontaneous group will disappear from the environment after its members leave.

Groupwork

Learning groups are primarily used for groups to conduct collaborative work. Even so, the work done there might be of interest to those outside these teams. Therefore, it is increasingly important that workplaces for teams should not hinder community members from

interacting with those teams. This consideration is particularly important in educational settings because of the central role knowledge sharing has within educational systems. In fact, the policy of availability to communities will effectively intensify this kind of knowledge sharing especially in small communities where people are more likely to share common interests.

5 Artifact-Based Learning Groups

In this chapter, I turn my attention to artifact-based learning groups. I will first present them from different perspectives, summarizing key components of the system such as artifact management, private study, one-way drop, lightweight connection, creation of spontaneous group rooms, flexible privacy control, and history management.

After introducing an artifact model for learning groups, I will primarily concentrate on addressing how these key components can be realized, tackling a series of relevant issues like notification server, session management, and role-based access control model.

5.1 A Visit to Lecture 2000

The spatial framework presented in the preceding chapter is comprised of an entrance hall, a discussion group, a tutor studio and several learning groups. Instead of covering all of these components, this chapter is primarily devoted to the design of learning groups in Lecture 2000.

Like most CSCW systems, Lecture 2000 is a client-server system. Any client has to log on to the server before anything useful can be accomplished. Once a user starts up the client part of the system, a log-on window will appear to request the user to enter his user name and password for identifying the user's member status. Unlike many other CSCW systems, the purpose of this step in Lecture 2000 is not to limit users' access to the system, but rather to differentiate community members from group members. Later in this chapter, we will show how a user's member status will be determined. Basically, any user can enter the system by just giving a user name (usually a pseudonym). For users who have once logged on to the system's server in the past, related profile records consisting of information such as user's name, password, member status will be created and preserved in the system's repository.

After logging into the system, community and group members may continue differently. While a community member will first enter the entrance hall, a group member can directly enter his own group room. Further, a community member who is present in the entrance hall can enter a learning group later. Thus, activities within the spatial environment can be discussed from three different perspectives:

1) Community members in the entrance hall,

2) Group members in their own learning groups, and

3) Both community members and group members in a learning group.

5.1.1 Community Members in the Entrance Hall

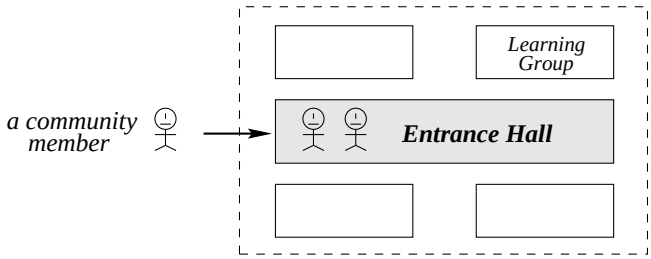


Figure 5.1: Community members in the entrance hall

The entrance hall is the starting point for community members (see Figure 5.1). As a typical workplace for communities, the entrance hall affords community interactions in many ways. Since community members usually do not know each other and mostly interact informally, unplanned and spontaneously (see section 4.2.4), the entrance hall must be a friendly social environment in order to foster interactions among them as easily as possible. The facilitation of community interaction in the entrance hall can be observed from the following aspects (see Figure 5.2).

Information

For example in Figure 5.2, once the user Koehler enters the entrance hall as a community member, he finds some useful information, including:

- **Information about Learning Groups**

  One of the most important principles for the system design is that privacy control should not stifle but rather encourage interaction and collaboration. To pursue this goal, some information about a group has to be exported to community members. The door state of each learning group, for example, shows whether this group is currently open to others (e.g., community members in the entrance hall or persons from other groups) for avoiding intrusiveness, an issue which has been extensively addressed in section 4.2.3.

  1) *door state*

     A door may have one of the following three states:

     – open
     – closed
     – ajar

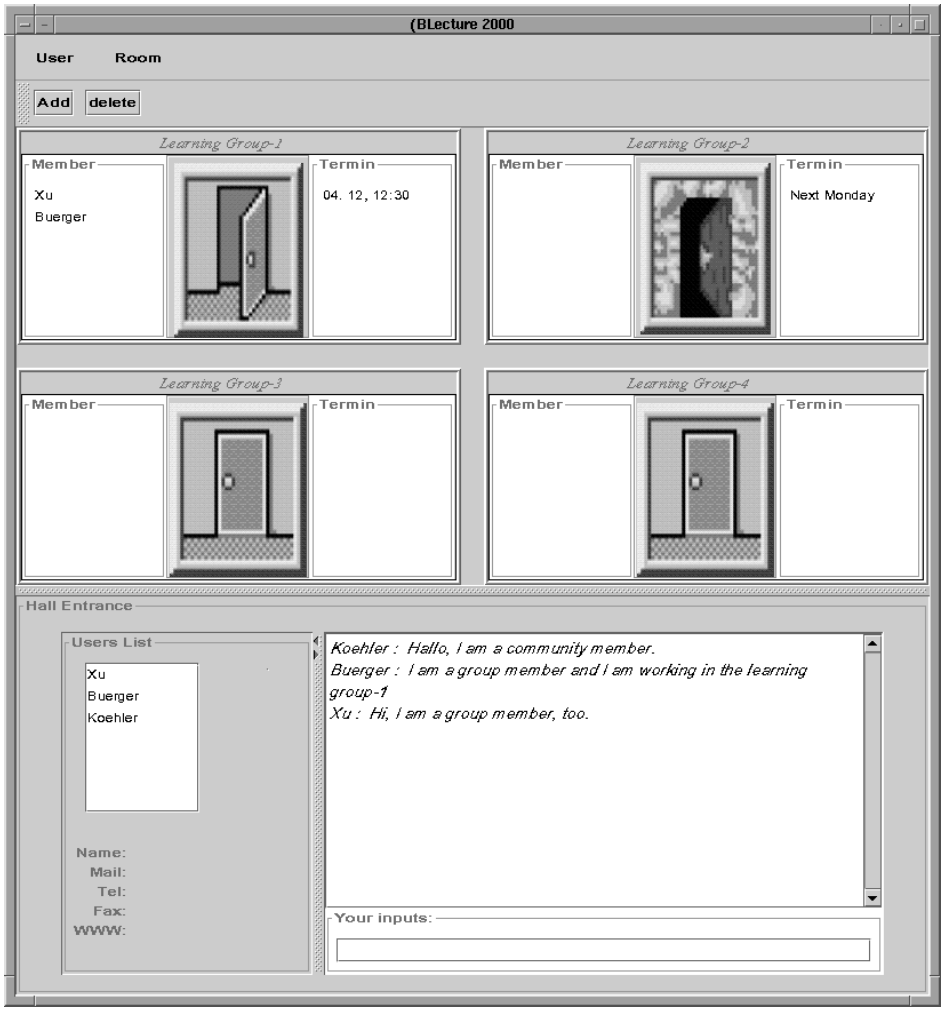


Figure 5.2: A community member enters the entrance hall

In the example shown in Figure 5.2, the community member Koehler finds that the door to Group 1 is open, while the door to Group 2 is ajar and the remaining doors are closed. These door signs represent three different states of these learning groups: anybody may currently join Group 1; Group 2 is not available at this very moment, but people can later join this group; Both Group 3 and Group 4 are closed without offering further information. Perhaps these groups are holding a meeting, or the members simply do not want to be disturbed. A group can close its door at any time for preventing others from learning what is happening within the room.

2) *meeting date*

   A group which has scheduled future meetings expresses a signal of welcome for other participants in the group meetings. In this example, Group 1 and Group 2 have published the meeting date for others to join.

3) *participant list*

A participant list of a group includes those who are currently working within the group room. People appearing in the list may be not group members of this room but just visitors to the room. The list of official group members of a room can be obtained by clicking the menu item “room”. This example shows that two users are working in Group 1 while nobody is in other group rooms.

Briefly, the information like door state, meeting date, and participant list of a group helps people to identify opportunities for collaboration. At the same time, it can greatly diminish possible intrusiveness between community and group members as well as among members from different groups.

- **Information about Partners**

  In addition to information exported by learning groups, community members in the entrance hall can get to know each other through information about themselves. Clicking a user’s name in the user list will bring up information such as the user’s name, mail, and WWW address. Like group members, community members can also edit their own user profiles. A user’s profiles can serve as a glue to bring people together even after they have left the environment. A detailed description of user profiles can be found in section 6.2.1.

Communication between Partners

In the entrance hall, community members communicate with each other through some tools such as chat, whiteboard. Figure 5.2 shows that a chat tool is chosen for communication among community members present in the entrance hall.

Entering into Learning Groups

Community interaction happens not only among people within the entrance hall, but also between those in the entrance hall and learning groups. Moreover, community interaction can also happen within learning groups once community members enter the learning groups. In this example, the community member Koehler can enter Group 1 if he likes. What happens within learning groups will be discussed later.

5.1.2 Group Members in a Learning Group

An Artifact-Based World

After logging on to the system’s server, a group member directly enters his own learning group (see Figure 5.3). As illustrated by Figure 5.4, a learning group is an artifact-based working environment. As a learning group is created, some basic artifacts are generated and made available:

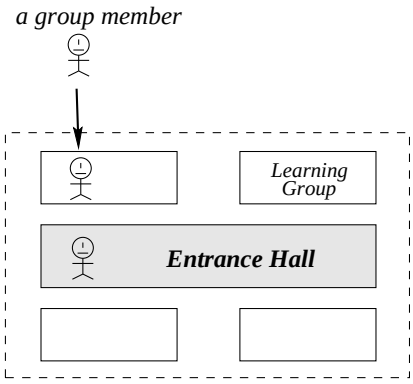


Figure 5.3: Group members in a learning group

- Room

  Firstly, each learning group owns a room artifact, affording various activities of the group. A room accommodates workplaces of the group.

- Workplace

  A workplace in a group room can be either *private* for a group member or *public* for the whole group. A private workplace holds artifacts created by the owner of the workplace. On the other hand, a public workplace is used to hold artifacts available to all group members inhabiting this room. In general, each group member will possess a private workplace, and there may be one or more public workplaces in a learning group. Figure 5.4 shows that Group 1 has four personal workplaces and a public workplace.

- Container

  Containers are designed for group members to hold their smaller artifacts and to better structure their working environment. Of course, one can set up his own containers at will. But it is reasonable to allocate a container to a kind of documents. In Lecture 2000, some special containers are defined to hold similar artifacts:

  – Script Container

    The artifacts in this container consist of materials such as lecture scripts, tasks, and annotations etc, which directly support the user's learning activities. They can be either references to course material provided by tutors or documents prepared by students according to their own preferences and learning advances.

  – Coordination Container

    This container is used to structure artifacts such as to-do list, memos and notices etc, which help to coordinate learning activities among group members.

  – Notification Container

    Artifacts which enable a user to send information to other group members will be assigned to this container. For example, a group member can send a memo

Figure 5.4: A group member enters directly his own learning group

information to his colleagues' coordination container. Further, mails are also typical artifacts present in a notification container.

- Documents

  Finally, the artifacts stored in containers are called documents. Learning scripts in script containers are documents, and do-to lists and mail lists in a coordination and notification container are also documents. Note that some documents can be just references to other course material which is provided by tutors or by the system and maintained in the system's repository.

The relationship among these artifacts is shown by Figure 5.5. It is noteworthy that both rooms, workplaces, containers and documents are referred to as artifacts. Later on, we will find that rooms, workplaces, and containers are all instances of organizational artifacts whereas documents are instances of functional artifacts. This working environment for a

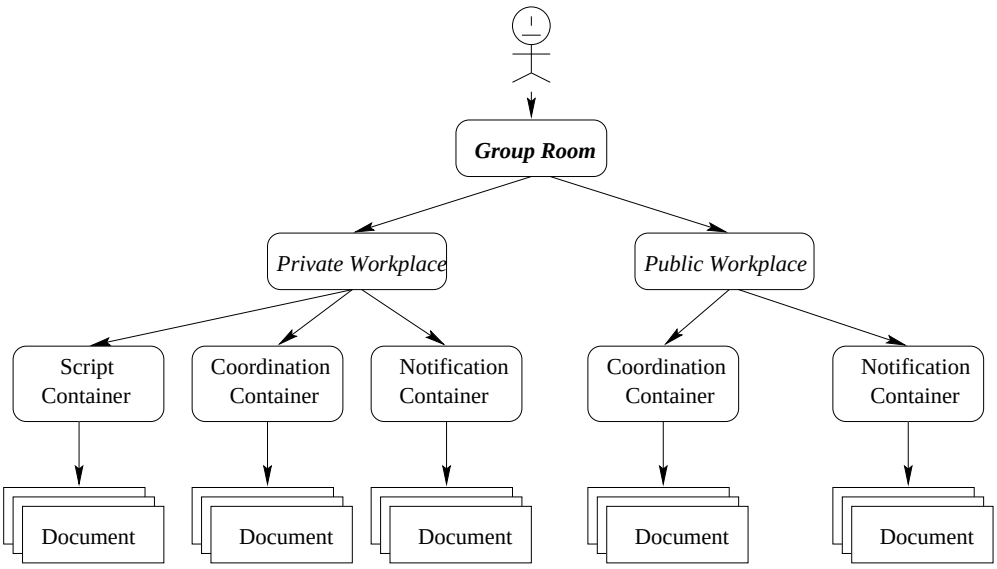


Figure 5.5: An artifact-based world in learning groups

learning group is just an example of our artifact-based model, which will be shown in the next section.

Collaboration between Group Members

With the help of these artifacts, group members in a learning group can collaborate in the following ways:

- **Asynchronous Collaboration**

  A group member normally works on documents within his own private workplace. This process is referred to as *private study*. If he wants to share a document with others, he can simply do that either by setting a "lenient" privacy control on the document or by moving it from his private workplace to the public workplace of the room.

  The organization of a learning group into workplaces, containers and documents is intended to better support asynchronous collaboration by enabling users to easily locate information useful to them. Various possibilities for asynchronous collaborations will be presented in section 5.4 on page 125.

- **Synchronous Collaboration**

  Group members within a group room may engage in a synchronous collaboration in two ways. They may actively choose to attend a session; that is, they may either be invited by someone in the session or find and join the session by themselves. This is an explicit process. Alternatively, they can participate in a session implicitly; that is, people who are "working" on the same artifact at the same time will be brought into

a session under certain circumstances. Note that the word “working” may represent “entering”, “opening”, or “editing” etc. with different artifacts. A detailed description of implicit sessions can be found in section 5.5.3 on page 135.

Connection to Entrance Hall

While working in a group room, group members can keep aware of what is happening in the entrance hall. When a group member finds a familiar appearance in the entrance hall or he finds the conversation there interesting, he can simply choose to join.

5.1.3 Community and Group Members in a Learning Group

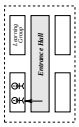


Figure 5.6: Both community and group members in a learning group

If a group room’s door is left open, community members currently in the entrance hall can choose to enter the learning group. Thus, a learning group in Lecture 2000 may sometimes have both community and group members. Seemingly, a community member who has joined a group does not feel that he is a stranger. If permitted, he can do what a group member can do. For example, he may visit a private workplace located in the group and attend meetings with other users present in the room. However, what a community member is allowed to do is well defined by a role-based privacy control mechanism, which is aimed at protecting the privacy of the group’s work, and discussed in section 5.6.

A role is a category of users within the user population of a given application; all users in a certain role inherit a set of access control rights to objects within the application [Edward96]. A community member usually takes the role *visitor* while a group member has the role *collaborator*. Because of the differences between the access rights assigned to the roles visitor and collaborator, the actions of community members are strictly constrained. As a

rule, a community member will be allowed to view what group members have done and annotate the group member's work.

5.1.4 Key Components

All analysis in the previous chapters about supporting learning activities of both community and group members indicates that we need the following key components in Lecture 2000:

1) *Artifact Management:*

   We must first of all develop an artifact repository to manage artifacts like rooms, workplaces, containers and documents. In addition, information about users, sessions, histories has to be preserved in the repository, too. The basic function regarding artifact management is to implement basic operations on artifacts such as creating, updating and deleting artifacts in the repository.

2) *Private Study:*

   Instead of focusing on ordinary communication tools like chat and whiteboard, the system is primarily aimed at supporting students' learning activities, which serve as the basis for further interaction and collaboration among group members, as well as between community and group members. Typical learning activities may include:

   - working on lecture scripts;
   - completing tasks;
   - preparing for exams;
   - other activities.

   To support these learning activities, the system integrates a series of applications such as *course composer* which originally makes teachers to create lecture scripts with reusable material modules stored in a repository. On the other hand, students can create personalized lecture scripts from these building blocs according to their own preferences such as learning goals and knowledge levels.

3) *Creation of Spontaneous Groups:*

   Community members might feel it necessary to have their own groups either after a certain level of trust and confidence between them has been created or when they have common interest to discuss a problem without delay and privately. These groups are called *casual* or *spontaneous groups*. Their rooms are referred to as *casual* or *spontaneous rooms*. Spontaneous groups can evolve into learning groups as their members develop a closer relationship.

4) *One-Way Drop:*

I borrow the term *one-way drop* from TeleNotes [Whitta97] to denote a set of functions enabling people to leave a brief message in group rooms. When a user enters an empty room, synchronous collaboration is impossible to occur within the room. Even in a room where several users are present, they may concentrate on their individual work. In these contexts, asynchronous collaborations are required to build and to maintain a communication channel between people. Key functions for one-way drop include *leaving a message*, *annotating* and *commenting* of documents etc.

5) *Lightweight Connection:*

For synchronous collaboration, I put great emphasis on connections which do not require participants to issue an invitation or to create a session explicitly. Rather, people can engage in meetings with little initial overhead. The discussions in the preceding chapter have concluded that community interactions are normally informal, brief, spontaneous, as well as one-to-one. Further, group members are often involved in such informal interactions unless they are present in an official meeting. Characteristics of informal interactions suggest that people should be allowed to connect with each other quickly and easily. Our artifact-based model can make it easier to pursue this goal.

6) *Flexible Privacy Control:*

We firmly believe that a flexible privacy control mechanism will not stifle collaboration but rather make a system more open, i.e. people are more willing to use and to work with the system. We protect privacies of community and group members by combining spatial approaches and role-based access control mechanisms. While the spatial approaches help to avoid intrusiveness, the role-based access control model assures that both community and group members will behave within learning groups in a predictable and manageable way. For realizing the role-based access control model, three types of roles are defined: *Owner*, *Collaborator*, and *Visitor*. Correspondingly, we support three types of access privileges to artifacts within a group room: *ADMINISTRATION*, *COLLABORATION*, and *VISITING*. Users who assume a certain role will be granted the associated access privileges.

7) *History Management:*

Finally, the system must provide a component managing document versions and recording the access history of artifacts in group rooms. In addition, it will enable users to track past sessions and to explore session information such as participants, durations of a session etc.

5.1.5 System Architecture

In order to implement these seven components, we need another two cornerstones to build the whole system:

- an artifact-based model and

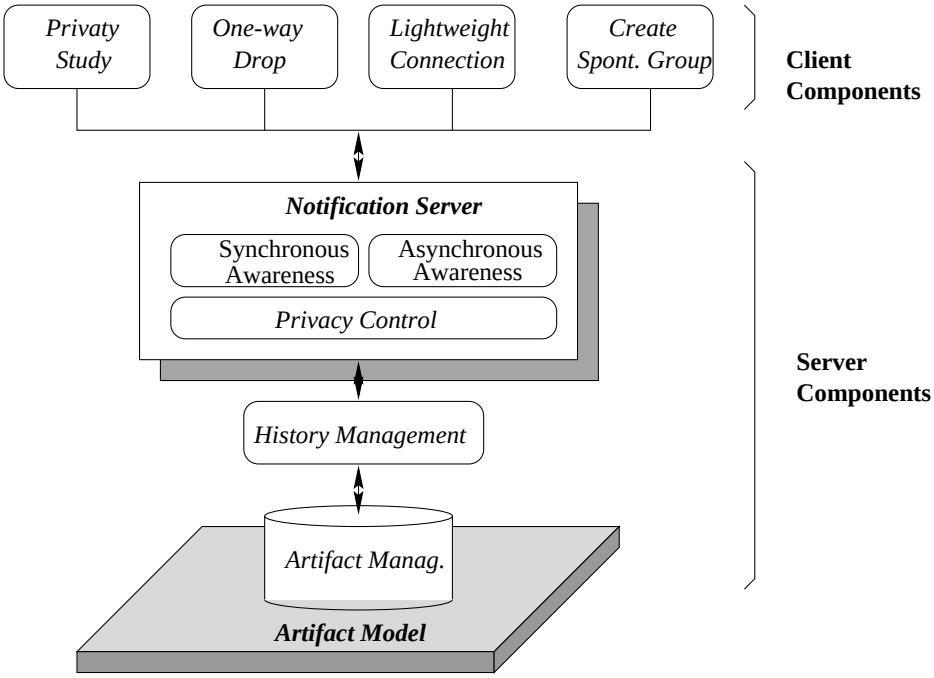


Figure 5.7: System architecture: key components are based on the artifact model and the notification server

- a notification server.

As shown by figure 5.7, the artifact model offers a conceptual model for defining various artifacts within learning groups, and the notification server is concerned with the provision of both synchronous and asynchronous awareness for collaboration. A great deal of discussions in the last chapter has involved the support for awareness information.

As the discussion in this chapter continues, we will find that one of key features of the notification server is to support so-called *centralized state sharing*; that is, states of all artifacts are maintained in the centralized notification server. As such a feature simplifies the implementation of notification services, the server can also integrate the function of privacy control (i.e. control of access rights to artifacts). Once an access request to artifacts (i.e. shared states in the notification server) is issued by a client, the notification server will first of all check this client's required access right. Only when a positive result is returned from the checking process, the shared state in the server can be just changed and a notification about such a change will be generated and delivered to relevant clients.

Figure 5.7 classifies the seven key components listed above into two groups:

- Client Components

  Components like private study, one-way drop, lightweight connection and creating spontaneous groups reside on client sides and directly support community and group member's learning activities, including synchronous and asynchronous collaborations in learning groups.

- Server Components

  On the other hand, components such as privacy control, history management and artifact management are located on the server side. It is noteworthy that the history management component also exploits advantages of the notification server to make history records of what was changed, when, and by whom. As the notification server generates an update-type event, a call to the history management component is made to record relevant information into the system's sole repository managed by the artifact management component.

I will first introduce how to set up spontaneous groups. The artifact model, the notification server and the seven key components will be discussed in more detail in the following sections.

Set up Spontaneous Rooms

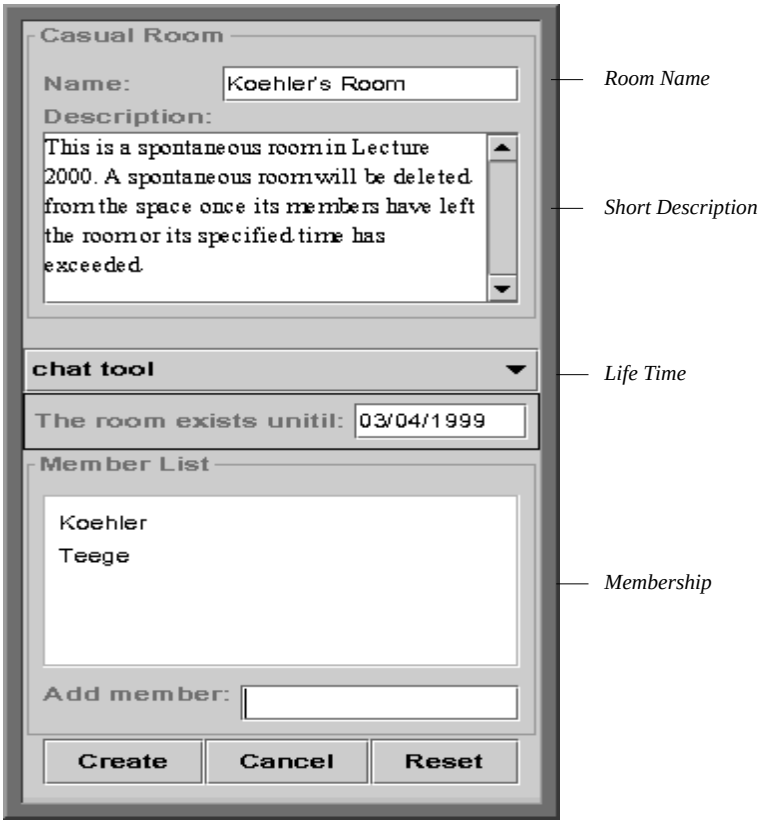


Figure 5.8: Create a spontaneous room

By clicking the menu *room*, community or group members can set up their spontaneous rooms. Attributes needed to define a spontaneous room include:

- room name

- short description
- initial membership
- life time

Figure 5.8 shows how to create a spontaneous room. By default, a room will automatically disappear from the server once its inhabitants have left. Alternatively, users can also specify a life time for the room. The spontaneous room will disappear from the system once the specified time for the room has exceeded. However, the maximum life time of a spontaneous room is limited by the system. The door of the spontaneous room is closed at the beginning to keep the room's new inhabitants from being disturbed.

Once a spontaneous room has been created, some basic artifacts will be automatically made available within the room, including:

- private workplaces for group members
- public workplace(s) for the group
- containers

Like normal learning groups, spontaneous groups can be browsed and joined by users. Related discussions about session management can be found in 5.5 on page 131.

5.2 Artifact-Based Learning Group

5.2.1 Artifact Model

In this section we will first introduce our artifact model used to build group rooms. The characteristics of the artifact-based model will be examined in greater detail later in this section.

The artifact-based model comprises the following three classes of artifacts (altogether six types of artifacts):

1) **Function Artifacts** directly support collaboration among group members:

   - *Information Artifacts*

     Objects which directly support the learning process of students in a learning group such as electronic educational material, exercises, annotations, etc. These artifacts constitute the information space [Schlic97b] of Lecture 2000.

- *Coordination Artifacts*

  Objects used to coordinate learning activities among group members, such as whiteboards, time tables, to-do lists, memos, notices, etc. In general, coordination artifacts are used to leave a message in a workplace for others. They may contain references (links) to artifacts being coordinated in the workplace.

- *Notification Artifacts*

  Objects which directly provide users with notification means, i.e. allowing users to notify others of their own work, such as telephone list, mail list, etc. More specifically, these notification artifacts allow a group member to send a message to other rooms or workplaces from his own workplace.

2) **Action Artifacts** are associated with each function artifact defined above (i.e. information, coordination and notification artifacts), specifying user operations on them. Action artifacts are similar to Helpers specified for MIME files in WWW-browsers.

   The idea behind action artifacts is that users don't use traditional work methods for the workplace, i.e. open a program (e.g. an applet), then load objects in the program environment. By introducing action artifacts, a user may attach his favorite program as an action artifact. If the user selects the artifact, the attached program is automatically invoked.

3) **Organization Artifacts** are responsible for grouping and organizing function artifacts, including

   - *Container Artifacts*

     Container artifacts organize and structure function artifacts. Physical counterparts for container artifacts are likely to be drawers, cabinets, tables etc. in an everyday office environment. Container Artifacts can contain other containers. A group room, for example, is a special container artifact which is used to group both private and public workplaces in this room.

   - *Layout Artifacts*

     Layout artifacts define the location and proximity of artifacts in a container artifact. Artifacts within a learning group will be so arranged that it can satisfy the user's desire to "rummage through chests and cupboards". Every room can define certain default layout values for users to select among them. Users can change the appearance and arrangement of their workplaces and containers in accordance with their own preferences and customs by assigning different values to these Layout artifacts.

Figure 5.9 illustrates a generalized artifact-based group room, where each user can own at least one container for organizing his function artifacts. In addition, the group can specify several public containers shared by the whole group which mainly include artifacts facilitating coordination and collaboration of group activities. All function artifacts located in

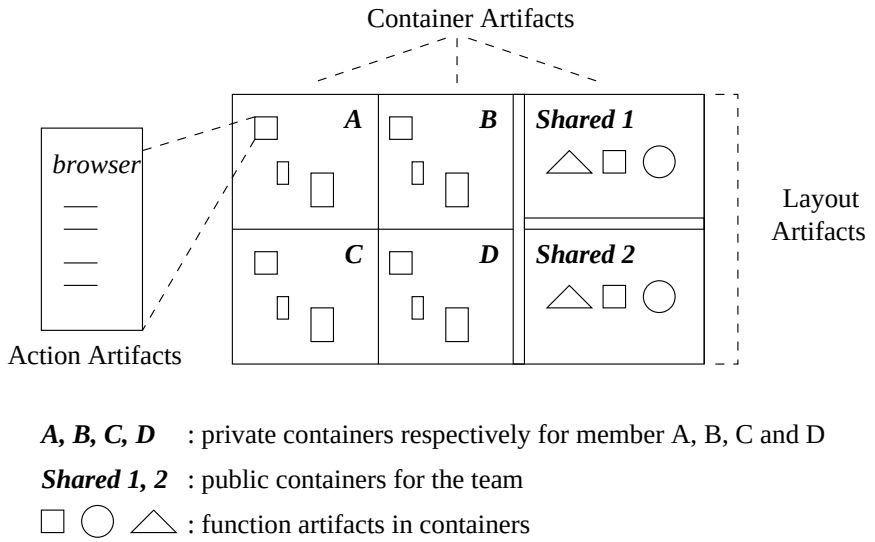


Figure 5.9: A generalized artifact-based group room

containers have associated with their action artifacts specified by their owners. In this example, a browser is an action artifact selected for the lecture script which enables the user to view the electronic educational material and also to complete his exercises with the help of forms embedded in the browser[1]. Finally, the layout artifacts define the general appearance of the room by specifying proximities of artifacts.

For simplifying the realization of the spatial environment for Lecture 2000, I here apply the artifact model in a concise way. As depicted in section 5.1.2, learning groups in Lecture 2000 consist of room, workplace, container and document. Apparently, rooms, workplaces and containers are instances of defined container artifacts. Various documents are examples of functional artifacts including lecture materials, to-do lists, memos, annotations, and so forth. Action artifacts implement programs which relate to a set of documents (functional artifacts). Finally, menus which allow users to set up the appearances of their rooms are linked to corresponding layout artifacts.

5.2.2 Characteristics of Artifact-Based Group Rooms

The artifact-based learning groups differ from other existing shared workplaces of groupware systems in many ways. I will illustrate several important characteristics of these kind of workplaces with respect to support for community interaction and groupwork.

Artifact-based

The most striking feature of our learning groups is that they no longer follow the traditional tool-based method for supporting collaboration, they are instead artifact-based. Most

[1] Such examples can be seen on our homepage *http://www11.informatik.tu-muenchen.de/lehre/*

existing groupware systems have introduced a wide variety of tools in supporting users' collaboration, such as chat tool, whiteboard, desktop videoconferencing, multi-user editor, and so forth. In these traditional systems, a user opens a tool, loads objects into it, and attempts to collaborate with others. In artifact-based learning groups, people can work like in an everyday environment. Opening a lecture script, they can read and add annotation. Clicking on the mail list, they can process mail with the tool specified by the associated action artifact.

Effective support for asynchronous collaboration

Traditionally, people can let workplaces simply persist across sessions to support asynchronous collaboration [Lee96, Rosema96b]. When one enters into such a space in which no other persons are currently present, he gets only the final result. However, confronted with a jumble of lifeless signs left in tools (e.g., characters and figures on a whiteboard), it is certainly difficult for people to later understand and keep track of activities which took place within that place. On the other hand, studies show that in everyday working environments people are usually skilled in "communicating" with some artifacts around them (e.g. to-do lists, memos etc.). Thus, workplaces consisting of artifacts which can be assigned a practical meaning and whose usage conforms to recognized everyday usage can better support asynchronous collaboration among users than traditional ones (e.g. TeamRooms). For instance, looking at a memo left on the desk, they can learn what the others have done or have requested. Directly browsing through work of colleagues (when allowed), one can find something without going through the traditional asynchronous procedures (i.e. describing the problem, contacting one or several colleagues and waiting for a reply). Of course, the prerequisite for efficiently browsing needed information within group rooms is that the system provides awareness information for asynchronous collaboration, a topic discussed in section 5.4.

Support for variable lightweight connections

Normally, most systems enable synchronous collaboration when group members are present in a room at the same time [Benfor93a, Lee96, Greenb98]. In fact, one should be allowed to continue his own work even though someone enters the place. With an artifact-based mechanism, a group can control under which conditions a real-time collaboration occurs. If an artifact is set to be a *trigger* of lightweight connections, a session will be initiated when two or more people are working on that artifact at the same time. The trigger could be a room, a workplace, a container, or even a document in the room. When specifying the whole room as a trigger, group members will automatically work together in real-time once they enter the room at the same time. Considering a container as a trigger, real-time collaboration takes place when people just attempt to access the specific container. In short, the workplace allows individual and collaborative work simultaneously.

Customization and Flexibility

Groupware requires fine tuning to meet the needs of all group members [Grudin91]. Supporting customization is one prominent feature of this kind of workplace. With the given six types of artifacts, group members can choose to set up their own workplaces. For example, a group member can choose to set up several private workplaces for himself. As the discussion in this thesis unfolds, more features supporting the customization will be found.

5.2.3 Ubiquitous Computing, Augmented Reality and Cooperative Buildings

The idea to develop an artifact-based model arises from the investigation of a variety of relevant work done in the past few years.

We took our inspiration from everyday working environments, especially from objects found in offices or at homes, the purpose of which is to capture and convey information, and thus implicitly or explicitly to support people's collaboration. In addition, the work in the research areas of *ubiquitous computing* [Weiser93, Weiser98], *augmented reality* [Mackay93] as well as *cooperative buildings* [Streit98, Kirsh98, Covi98] helped us to identify the artifacts for Lecture 2000.

In fact, people are increasingly trying to model their computer environments after true everyday working environments. For example, we find that today's technology, rather than replacing some office artifacts (e.g. paper), has increased our use of them. Furthermore office artifacts have been used as models, e.g. tabs, pads and boards in *ubiquitous computing*, and Digital Drawing Board and Mosaic in *augmented reality*. The research area of *cooperative buildings* has further confirmed our view. To provide virtual environments (Adaptive Rooms) able to dynamically adapt to users' needs, Kirsh [Kirsh98] distinguishes among four types of virtual objects, i.e. *active objects*, *reactive objects*, *passive objects*, as well as *information objects*, for all of which we can find physical counterparts around us, e.g. desks, chairs, drawers, walls, ceilings and floors.

With the help of these ideas, I get a more detailed insight into real group rooms. Covi *et al* conducted a detailed investigation of dedicated project rooms within companies [Covi98], finding that ten different types of shared artifacts are used in these rooms. The most interesting finding of the study is the usage of cognitive artifacts to codify procedures and to make the project's progress visible. A number of artifacts were used in these rooms which are rarely used in shared meeting rooms. Two kinds which are relevant to groupwork support are of particular interest:

- *Shared Visual Displays:* Flip charts, whiteboards, tack boards and walls often hold important artifacts, displaying work in progress, current status of tasks, reference materials or past accomplishments. These shared visual displays help the group to make intangible work visible and editable via representations;

- *Coordination Documents:* The most important coordination document is a to-do list consisting of objects to be built. Similarly, action items are used for reminding group members of agreed-upon group obligations and priorities for which they are accountable.

Covi's study emphasizes the importance of artifacts used in everyday working environments. Furthermore, it leads us to a consideration of an artifact-based spatial model since it enables us to achieve goals previously elaborated.

5.3 Notification Server

For synchronous collaboration (e.g. lightweight connections), a user must be able to keep track of others' current activities in order to get a context for coordinating his own work. For asynchronous collaboration (e.g. private study and one-way drop), records of actions performed on shared documents must be maintained for providing information of what has happened lately in the shared workspace as well as the current status of the space. A notification server is supposed to provide this kind of awareness information for the realization of major key components listed in the last section.

5.3.1 Overview of Notification Server

Notification Server

In client/server computing, a client initiates the client/server interactions by sending a request to its server. A server is a logical process that provides services to requesting processes. The functions that a server should perform are determined in large part by the types of requests that clients can send to the servers [Borgho95, P.21].

While a CSCW system may encompass a variety of servers supporting group work, a notification server is one primarily used to distribute and inform about updates and changes made by users to other participants involved in the collaborative process. A notification server works very simple. As shown by Figure 5.10, once a shared state on the server has been changed, the server notifies all clients of such a change.

Definition 5.1 (Notification Service and Notification Server)
Notification services include functions managing shared states of a collection of clients and notifying the clients whenever one of the states changes. A notification server is the subsystem which implements the notification services[2].

[2]Notification service is sometimes called *awareness service* or *event service*.

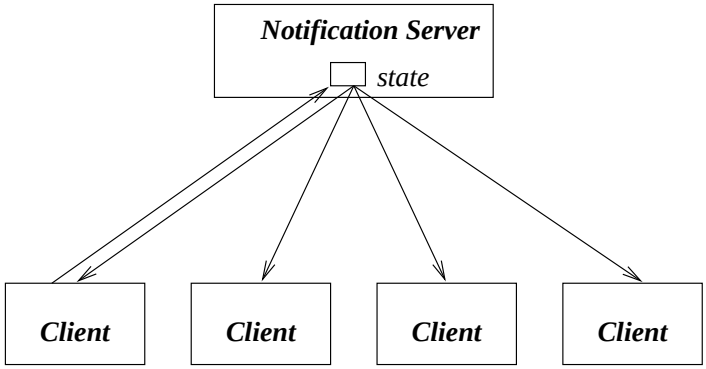


Figure 5.10: Notification server

Notification Server for both Synchronous and Asynchronous Awareness

Traditionally, a notification server is a primary concern of synchronous CSCW systems, providing *synchronous awareness* about events just happening [Patter96, Palfre96, Trevor97, Schuck96, Woo94]. In a learning environment, for example, this synchronous awareness offers the up-to-the-minute knowledge that a student requires about other students' interactions with the shared workspace. From users' perspective, a notification server is used for the need that user interfaces of the participants must be kept consistent to promote the impression of working together.

In asynchronous systems such as mail system and bulletin boards, there is no need for users to be aware of what is currently happening in the shared workspace. Instead, these systems must provide mechanisms to enable users to keep aware of what *has happened* with regard to their own work. This kind of *asynchronous awareness*, for example, is supported by BSCW, providing knowledge about past events. Various mechanisms for providing asynchronous awareness have been discussed in section 4.3.2 on page 88.

As issues of notification and awareness have become increasingly important in CSCW, notification servers are also concerned with mechanisms for providing awareness information for asynchronous groupware systems [Ramdun98]. For supporting asynchronous awareness, a notification server has to *maintain shared states*, which means that changes of shared states must be preserved for providing information of what has happened lately in the shared workspace as well as the current status of the space.

A notification server incorporating the support for both synchronous and asynchronous awareness plays a key role in implementing our integrated learning environment. It helps to remove the seam between synchronous and asynchronous CSCW systems. It is then easier for users to move fluidly across the synchronous/asynchronous barrier.

In summary, the notion of the notification server in Lecture 2000 covers awareness support for both synchronous and asynchronous components in the following ways:

- Once people are working in real-time, the notification server supports these synchronous collaborations by propagating awareness information between participants.

- If people work asynchronously, new states of artifacts will be maintained through the use of a state repository. Latecomers can get to know what has happened in the shared workplace through these shared states.

5.3.2 Notification Server Architecture

Centralized vs. Decentralized Architectures

A notification server is a subsystem providing synchronous and asynchronous awareness for collaboration participants. A question is then whether the subsystem should be centralized or decentralized? Figure 5.11 shows two possible architectures for notification servers:

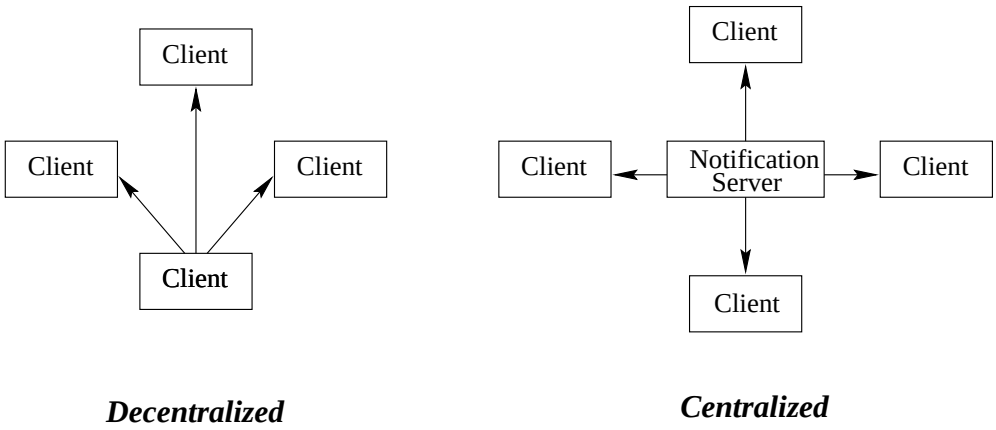


Figure 5.11: Notification server architectures

- Decentralized Architecture

  A decentralized architecture means that each client in the system sends notification messages to all other clients directly, which requires that each client has a copy of the server component. For example, clients in GroupKit communicate with each other directly without the aid of a centralized server [Rosema96a].

- Centralized Architecture

  However, it is sometimes necessary to ensure that requests sent to a notification server by clients are serialized, i.e., placed in a consistent order. Then, a site should be chosen to take the role of serializer. This site is then responsible for receiving awareness messages and redistributing them to all clients in a consistent order.

A distributed architecture implies parallel processing since each site in the system runs a copy of the server. Process bottlenecks are then less likely. There are, on the other hand, some disadvantages to this approach. Aside from the problem of serialization, this approach has a high overhead in algorithm complexity. For instance, every client should know about

all others in order to broadcast changes to them. Further, states of the server on each site must be kept up-to-date as users join and leave the system. Some problems of a decentralized architecture have been summarized by Borghoff and Schlichter [Borgho95, P.190].

It is common knowledge that a centralized server has its drawbacks, too. For example, a failure of the server can cause the whole system to break down. Moreover, such an architecture can lead to the so-called bottleneck phenomenon. On the other hand, it also offers advantages. For example, it can be easily realized, and it it simplifies serialization and synchronization.

Centralized State Sharing

We choose the centralized architecture for Lecture 2000 not only because of its simplicity and its capabilities for serialization and synchronization but also because of the importance of the concept *centralized state sharing*, i.e. shared states of artifacts are put in a centralized server. Briefly, some advantages provided by centralized state sharing can be identified as follows:

- introducing latecomers to a session

  Firstly, centralized state management makes it easy to introduce a latecomer to a session, i.e. to a synchronous application which is joined by several people at the same time. If all shared states in an application are kept by the server, then a latecomer must only contact the server without disturbing other clients in order to get current states of the artifacts in the application.

- asynchronous collaboration

  Secondly, the centralized state sharing makes it possible to support asynchronous collaboration. If shared states are distributed in clients, it will be somewhat difficult to manage them. With the centralized states, the management of shared states becomes easy. Clients can query most recent states of artifacts. These shared states serve as a glue to keep people in contact regardless of whether or not they work together in real-time. But the server needs a persistent repository to manage shared states.

- privacy control

  A client can access an artifact only when he owns relevant access rights to the artifact in the server. Because the notification server maintains all shared states of artifacts, it is easy to incorporate the function of privacy control into the server.

- concurrency control

  Finally, the control of concurrent access to shared artifacts is a critical issue of CSCW. Patterson *et al* argue that when clients are permitted to change the same state variables or collection of variables, it may be necessary to ensure that only one is doing it at

any given time[3]. To accomplish this, some form of locking is needed. Because of the centralization of artifact states, it is easy for the notification server to incorporate access control to artifacts. Jupiter, a MUD system based on a single sever, makes the same arguments [Nichol95].

Some notification servers do not support centralized state sharing. Zephyr [DellaF88], for example, provides a notification service without keeping any client-defined state at the server. Instead, the server accept messages from clients and multicast those messages back out to interested clients. This kind of notification servers have the difficulty to introducing latecomers to meetings. Asynchronous collaborations are impossible to be realized because shared states will be lost once clients have left.

Client- vs. Server-based Semantics

In terms of the flexibility of notification servers, another issue to be addressed here is which side of a notification system (i.e. the client side or the server side) should be responsible for interpreting the semantics of shared states. Generally, we can distinguish notification servers in CSCW between *server-based* and *client-based semantics*. A notification server with client-based semantics can be easily applied to other CSCW systems. In contrast, a notification server with server-based semantics is difficult to be used by other CSCW systems because semantics of the shared states have been predefined by the server in this case. The motivation behind this discussion is that we plan to make use an existing notification server to support notification services for Lecture 2000. A notification server with server-based semantics will make it difficult to implement this plan because the semantics of the shared states are predefined by the server and we can not explain them according to our own needs.

The feature of Jupiter's server-based semantics is one of its most striking characteristics. In section 4.2.5, I have briefly introduced that Jupiter's server consists of rooms, exits and objects. The Jupiter client programs are primarily responsible for managing the local details of the user interface (the presentation component). The client has no knowledge about these rooms and even their users. It is a more or less straightforward engine for manipulating the local machine's I/O devices under the direction of the server and the user. All of the semantics of the various actions it performs are carried out in the server's code [Curtis95]. Thus, the Jupiter server is substantially more complex and general than the client. Note that the Jupiter server connects to a database that maintains the state of the virtual world and the objects in it.

A very different example is NSTP (Notification Service for Transfer Protocol) with client-based semantics. NSTP, a notification server developed by Lotus, provides a common service for sharing state in synchronous multi-user applications [Patter96]. The model for NSTP encompasses the concepts *Place*, *Thing* and *Facade*. A Place simply contains the shared state for an application. Once a user is in a Place, he receives notifications about any state change

[3] In optimistic approaches to concurrency control, users can feel free to manipulate shared data at any time [Borgho95]. In Lecture 2000, a pessimistic approach is applied: a state in the server can be changed only when it has been successfully locked.

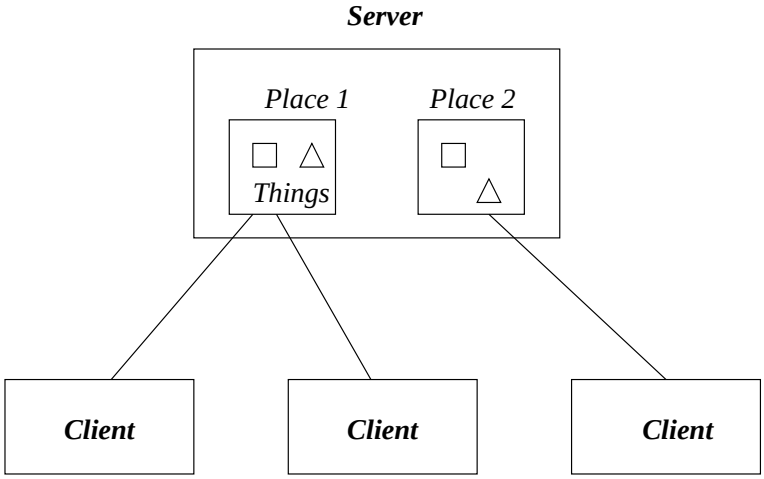


Figure 5.12: NSTP: Notification Service Transfer Protocol

of Things in that Place; they may also make changes of their own. Once the user leaves a Place, he no longer receives notifications from that Place. Things are the keepers of state within a Place. Things may be created, changed, locked, unlocked and deleted. Changes of Things will cause notifications to clients which are connected to the relevant Place. Finally, a Facade is the external view of a Place. Figure 5.12 shows a simple view of the notification server NSTP.

Because of its client-based semantics, this notification server then knows nothing about the semantics of its Places. Similarly, NSTP has no concept of different types of states. Thus, clients are responsible for explaining the semantics of Places and Things.

In short, characteristics of notification servers with client- or server-based semantics can be identified as follows:

- A system with server-based semantics imposes less requirements on the client side. Normally, the clients in the system are much simpler. The installation of client programs becomes easy, too. However, the server can normally not be used by other CSCW systems because the semantics of shared objects in this server have been predefined.

- In contrast, the benefit of client-based semantics is that the server could be used in a wide variety of CSCW applications. The server provides only notification services to clients. The semantics of the shared states in the server can be defined and interpreted by clients according to requirements of an application. Nevertheless, the client program of such a system is more complicated.

5.3.3 Notification Server Model

General Requirements

The discussion in the preceding section leads us to a centralized notification server architecture for Lecture 2000 with the support of centralized shared states. In summary, the notification server should fulfill the following design requirements:

- it should reflect the features of the artifact-based model.

  Certainly, artifacts in the system are hierarchically structured (see Figure 5.5). A room includes private and public workplaces. In workplaces, we find script, coordination, and notification containers. Finally, containers are filled with ordered documents. We should take advantage of the hierarchical structure of artifacts to simplify the realization of the notification server.

- it enables centralized state sharing

  The centralized state sharing can ease the problem of introducing a latecomer to a session. It allows the support for asynchronous collaboration. Finally, it simplifies the implementation of privacy control and concurrency control.

- it does not treat synchronous and asynchronous collaboration differently

  In section 4.3.3, I have addressed the issue of seamless transitions in CSCW systems. Especially, I put great emphasis on the removal of the barrier between synchronous and asynchronous systems. To pursue this goal, a notification server must be able to support both synchronous and asynchronous awareness.

Model

According to these requirements, I describe an abstract notification model providing required notification services for interaction and collaboration within the learning environment. Next chapter will discuss how to implement the notification model and related notification services. The implementation of the model will be based on NSTP with enhanced functions to support persistent states.

As shown by Figure 5.13, an artifact depicted in the artifact model (i.e. room, workplace, container and document) is represented as a *unit* in the notification server. A unit consists of two parts:

1) *a set of states*

   First of all, each unit is defined by a set of centralized states in the notification server. Each state includes the following fields:

   - *name*: The name is used to refer to the state. If the unit represents a group room, we can define states like $room\_name$, $door\_state$ and $group\_members$

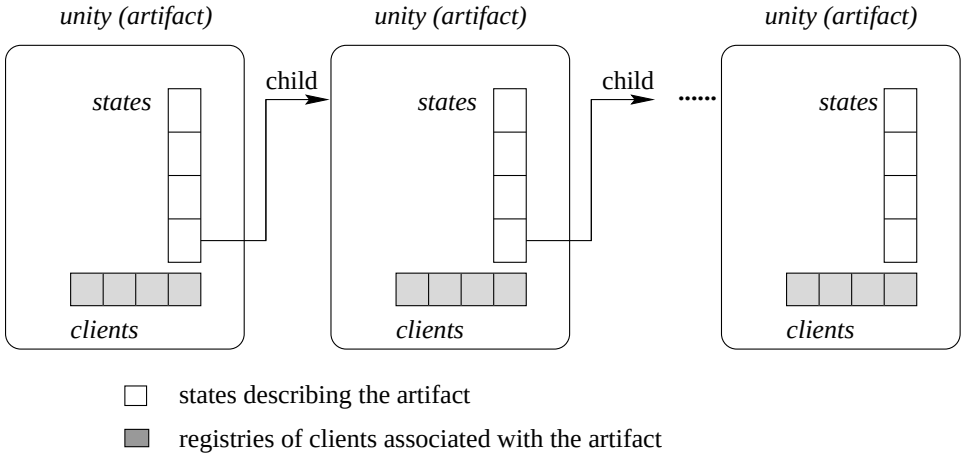


Figure 5.13: The notification server model for Lecture 2000

to respectively denote the room's name, its current door state (i.e. closed, open or ajar) and who are group members of the room. Because of the hierarchical structure of artifacts, each unit defines a special state to specify this artifact's children, which is illustrated by Figure 5.13.

- *value*: It specifies the value of the state. Basically, it may be an array or a string. For example, the value of the state $group\_members$ can be defined as an array of user ids. Of course, the state $room\_name$ has a value of string.
- *locked*: This boolean value specifies whether the state is currently locked by a client. A client attempting to update a state's value must first be granted the lock of the state.
- *attributes*: In addition, a set of user-defined attributes in the form of $(name, value)$ is associated with the state. A meaningful example of attributes is version information about the state.

2) *a list of clients*

Secondly, a list of clients is associated with the unit, specifying who are currently *working on the artifact* represented by the unit. Only clients included in this list can get notifications about what is happening with the artifact from the server. If the unit denotes a group room, the list contains ids of all users who are currently present in the room. The term *working on an artifact* will be accurately defined later in this chapter. Each client in this list is defined by two parts:

- *client*: It specifies the client, including information such as:
  - $user\_name$: user name and
  - $user\_id$: user identifier.
- *computer*: It specifies the computer on which the client is working, including information like:

  - IP_addr: IP address and
  - $port\_num$: port number.

According to these definitions, the whole artifact-based spatial learning environment will be represented by units and their associated states in the notification server. Thus the notification server supports two kinds of functions as follows:

1) *Artifact Management Services*

- *create, update and delete units and states:*
  First of all, the notification server must enable clients to create, update and delete units and their states. Because these units and their states should be kept persistently to support asynchronous awareness, the notification server will call functions in the component of artifact management to complete this process.

  Once a unit or its associated states have been created, updated or deleted, or a client has "entered" or "left" the unit, we say that an *event* has happened on the server side. This event will then cause the notification server to generate a *notification* (i.e. a message) sent to related clients to inform them of the event. Upon getting such a notification from the server, client programs will update their user interfaces, depending on how clients interpret the notification they have received.
- *access control to units and states:*
  Before a client can manipulate units and their states in the notification server, the server must first check whether the client has required access right. Only when the access right is satisfied, the manipulation can be then executed. In the system architecture, a component for privacy control has been defined. The access control will work together with this component to determine the access right of clients.

2) *Notification Services*

- *subscribe and unsubscribe:*
  To get notifications from the server about what is occurring or what has happened with a unit and its states, a client must register himself with the unit. A simple example is that once a user enters a learning room, the server will add the user into the client list of the unit representing the room artifact. Then the client can get all notifications regarding the room from the server.

  Once the client leaves the room, he will be deleted from the client list of the room. Then the client will be no longer informed of any changes regarding the room.
- *notification transfer:*
  The most important notification service is that the server is responsible for generating needed notifications according to what has happened with units and their states and sending them to related clients.

- *support for asynchronous awareness:*

  The notification services listed above are typically for synchronous communications. For asynchronous communications, knowledges about past events are needed. Thus the server must be able to record what has happened with units, their associated states and client lists, which requires the support for persistent state. We will discuss it in the next chapter.

5.4 One-Way Drop: Asynchronous Collaboration

5.4.1 Asynchronous Collaboration with Shared Artifacts

As the discussion continues, we will find that all components of this system are built on the basis of artifacts. Earlier in section 2.4.3 on page 34, I have discussed what purposes an artifact can serve. Briefly, an artifact can be used to support both synchronous and asynchronous collaboration. We know that inherent with synchronous applications is the existence of a shared context which consists of a shared environment as well as a multiuser interface to this environment. The environment incorporates a variety of shared artifacts which are managed and manipulated by the whole group. A simple example of a shared artifact is a document jointly authored by a group of co-authors. On the other hand, shared artifacts also allow a wide variety of working modes ranging from individual to collective work [Schlic97b]. The individual work on a shared artifact builds the basis for one-way drop.

A general process for asynchronous collaboration through the use of shared artifacts could be divided into three steps (see Figure 5.14):

1) Someone first produces an artifact.

2) The artifact is then shared with other collaborators who change and contribute to the artifact by annotating, commenting on it.

3) These changes, notes and comments are then gathered by one of the collaborators and are reconciled into a new version of the artifact.

In Lecture 2000, a similar process of asynchronous collaboration through the use of artifacts, illustrated also by Figure 5.14, looks as follows:

1) *Create:* Both students and tutors can contribute new artifacts to the learning environment. Firstly, tutors are expected to prepare learning materials like lecture scripts and tasks for a distance learning system which constitute a solid basis for further learning activities of students. With the aid of these learning materials, students can then conduct their private study. Moreover, students can also leave new artifacts there during their stay in the learning environment.

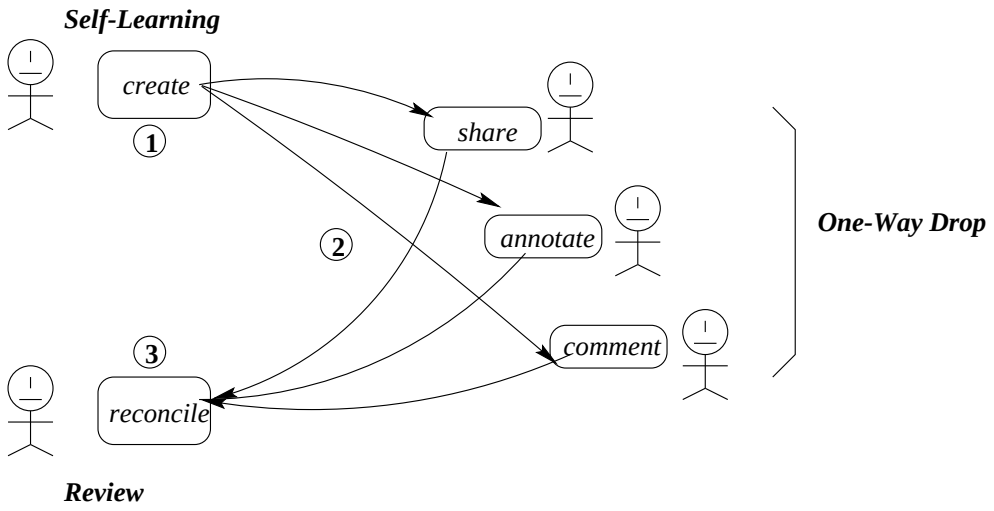


Figure 5.14: A asynchronous collaborative process through the use of shared artifacts

2) *Contribute:* Later, the one-way drop component allows others (i.e. collaborators) to comment and annotate these artifacts as long as such changes do not violate rules of privacy control. In this phase, new versions of artifacts may be generated.

3) *Reconcile:* Finally, with the help of awareness artifacts (e,g. memos, to-do lists), which collaborators use to provide context information about what they have done to shared artifacts, authorized users can reconcile these changes. Even without the context information, the system enables people to keep aware of what has happened to shared artifacts.

Drawing on this brief discussion, we find that the essence to support asynchronous collaborative process through the use of shared artifacts is the ability to understand what the others have contributed to these artifacts. According to Olson *et al*, essential prerequisitives to asynchronous work with shared artifacts include things such as

> *"an ability to effectively communicate information, an ability to understand the actions of collaborators, and an ability to integrate work from others." [Olson98]*

Unless we could find a way that allows people to easily know what has happened within the rooms, the artifact-based system would not make collaborations easier and more efficient but rather complicate the inherent dynamics of the working environment shared by both community and group members.

5.4.2 Asynchronous Awareness Support

Briefly, asynchronous awareness refers to information about what others have done within a learning group in the past. In an artifact-based learning group, information building asynchronous awareness includes:

- user who has initiated the change to the artifacts,
- type of the change, i.e. the artifacts have been newly created, viewed, updated or deleted?
- date when the change happened.

Because the notification server is supposed to record what has happened with shared artifacts (see section 6.1.3 on page 160), asynchronous awareness information can be provided by answering the following questions:

- Who has entered a group room and when? and when he left?
- Who has annotated or commented documents within a group room? and what he has contributed to these documents?
- Who has left new documents in a group room? and what these documents are?
- Who has destroyed a document and when?

When a user selects an artifact in the learning environment, he is allowed to learn this kind of information listed above.

In addition to this kind of asynchronous awareness which can be provided by the notification server, *awareness artifacts* are designed in Lecture 2000 to provide direct information for asynchronous collaboration between people. As introduced before, there are various awareness artifacts in coordination and notification containers. For example, coordination artifacts record what others have requested or done in a room while notification artifacts are used for a user to send an asynchronous message to other users' containers without entering the rooms or workplaces of the recipients.

More generally speaking, asynchronous awareness also includes information about where a document is. When one initiates a one-way drop to a learning group, it is critical to enable a user to locate documents quickly and easily because the asynchronous collaboration process normally happens with documents. For example, a user leaves a document in a room which can be later read by others to learn what the document tells. Rooms, workplaces and containers are primarily used to organize documents. To simplifying the process of locating targeted documents, the systems has adopted the following two mechanisms:

- *Guiding Path*

  A user can locate interesting documents through a guiding path which is comprised of a set of icons representing rooms, workplaces, containers and documents. As shown in Figure 5.15, such a guiding path can offer intuitive knowledges to lead a user to his targets.

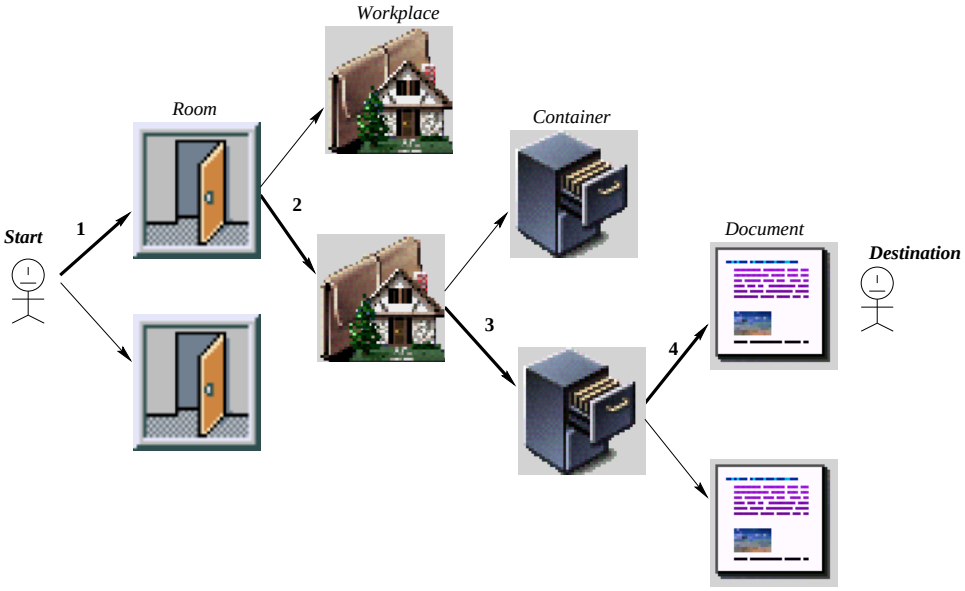


Figure 5.15: A guiding path consists of icons representing rooms, workplaces, containers and documents

- *Search Engine*

  Guiding paths may be sufficient for users who are familiar with the learning environment and then know where to find interesting documents. But a newcomer unfamiliar with the environment may be lost in the icons. A search engine which takes key words as inputs and returns direct paths to targeted documents will be helpful in this context.

  According to the analysis about what information constitutes asynchronous awareness, the search engine can take the following information as its parameters:

  - specification of artifacts: To directly locate an artifact, a user may provide the search engine with information such as:
    - group room in which the targeted artifact is located;
    - its name;
    - its owner;
    - date when the artifact is created or updated;
    - key words the artifact contains if it is a document;
  - specification of users: Who has created or updated the targeted artifact?
    - user name;
    - key words used to describe a user?

5.4.3 Awareness Artifacts

Awareness artifacts play a key role in enabling coordination among users. In section 5.2.3, a study covering ten different types of shared artifacts in rooms has been introduced. Indeed,

awareness artifacts such as whiteboards, time tables, to-do lists, memos, telephone lists, and mail lists etc. are often used in everyday working environments to facilitate and coordinate collaborative work. These artifacts are further classified into *coordination artifacts* and *notification artifacts* (see section 5.2.1):

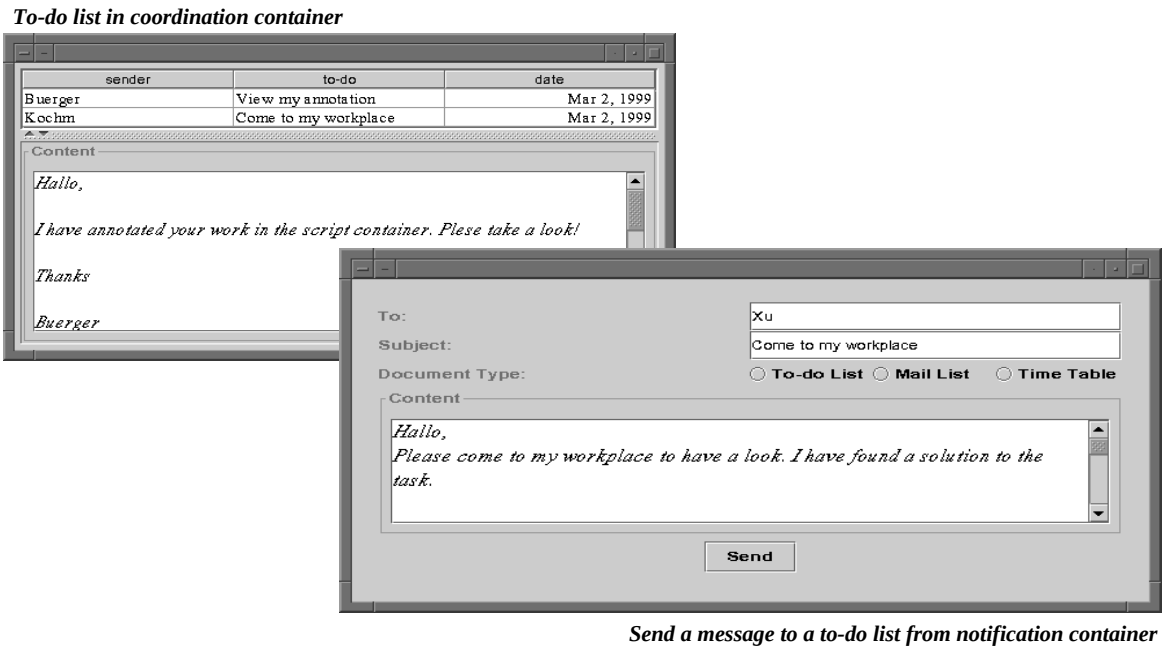


Figure 5.16: To-do list in a coordination container

The upper window in Figure 5.16 shows a to-do list located in a coordination container. User A, for example, can add an item into the to-do list of user B in two ways, provided that he has the related privileges as discussed in section 5.6.3 on page 147:

- User A opens user B's coordination container and then directly clicks the corresponding menu to add an item into the to-do list within the container.
- User A can send a to-do message from his own workplace to user B's to-do list, as illustrated by the second window of the same figure.

So far, the communication between coordination and notification artifacts is confined to the system environment. A user cannot send an artifact to another user's container from outside this system.

The difference between to-do lists and memos is very clear: users use a to-do list to expect further actions taken by readers of the list. In contrast, memos are primarily used to hold commentary or other kinds of messages dropped by visitors. Often, a message in memos does not request further actions by readers (In fact, these visitors usually use pseudonyms). Therefore, a user usually first looks through his to-do list and then memos.

Memo messages are organized in stacks, and each stack represents a specific topic or interest (see Figure 5.17):

- Messages in a stack are organized according to a natural order such as their arrival dates.
- Messages which have been read by a user will be put at the bottom of the stack so that he can continue to read.
- But, the view of the stack will be restored after the user has finished his reading; that is, people who come later in the room will find the same stack as before. Only the owner of the container can delete memos which may be saved into repository.

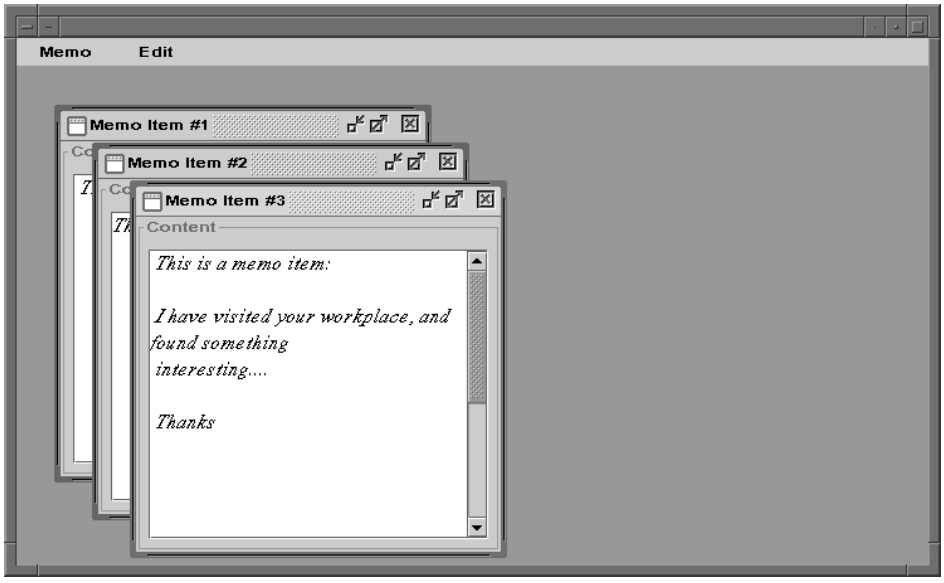


Figure 5.17: Memo stacks in coordination container

Annotating and Commenting Learning Materials

Apart from these coordination and notification activities supported within awareness containers, Lecture 2000 supports another kind of important one-way drops: annotating and commenting learning materials. This kind of asynchronous collaboration is primarily facilitated within script containers where lecture scripts, tasks etc. are located. I will discuss this topic in the next chapter, along with tools for learning activities.

5.5 Lightweight Connection: Synchronous Collaboration

5.5.1 Session Management

Groupware developers often concentrate on building applications such as multi-user editors. However, it is also important to provide groups with *session managers* [Greenb96b] for establishing connections between co-workers. These session managers must allow users to participate in sessions in flexible ways. Although the session management in MBone (see section 3.3.2 on page 52) has surely given us an informal knowledge about session management, a formal one is needed for continuing the discussion.

Patterson *et al* argue that a session is the duration of an application process from its invocation to its termination [Patter90]. Whereas this definition does not exclude the situation that a person interacts with a single-user's system, Ellis *et al* emphasize that a session normally involves several persons working together, such as jointly editing documents.

Definition 5.2 (Session)
A session is a period of synchronous interaction supported by a groupware system. Examples include formal meetings and informal work group discussions." [Ellis91]

Definition 5.3 (Session Management)
Session management means the process of starting, stopping, joining, leaving, and browsing collaborative situations across the network [Patter90].

This definition has specified some major but not all issues to be addressed. More specifically, tasks for session management can be identified as follows [Greenb96b]:

- creating, naming and deleting sessions,
- locating existing sessions,
- finding who is in a session,
- joining sessions,
- access control to sessions,
- allowing latecomers,
- allowing people to leave sessions early, and
- deciding whether sessions persist after all participants have left.

Subsequently, these issues will be discussed according to different models adopted for session management.

5.5.2 Models of Session Management

In general, we distinguish between *explicit* and *implicit* models of session management. While one must take actions to join session in the explicit model, the implicit model allows users to get into sessions with little effort.

Explicit Session Management

Some sessions are considered explicit largely because people need to take actions to join the sessions. Edwards argues that there are two explicit approaches to session management [Edward94]:

- **Initiator-based:**

  A user who has initiated a session, called an initiator, invites others to the collaborative session. As discussed in 3.3.2, the session management mechanism in MBone has also integrated such an approach, where the initiator can invite other colleagues or friends to join a session by clicking the button "Invite" and then giving the string $username@hostname$ to specify those to be invited. The latter can then press the button "Accept Invitation" to reply to the invitation.

- **Joiner-based:**

  In the joiner-based model, sessions have been created and started. Users are enabled to browse these sessions. Once they have found an appropriate session, they can attempt to join it. Again, MBone allows users to browse all sessions and then choose one to join.

In some literature, initiator-based is referred to as *calling-out* while joiner-based is called *calling-in*. A more general term for both approaches listed above is *heavyweight* session management because starting such a collaboration is *expensive*. The joining process usually requires some additional actions which are not a part of the task on which the participants are trying to collaborate. An example is that one who wants to chat with others (e.g. the chatting is now his task) must spend some additional time to locate a chatting group before he can start chatting.

Drawing on this brief introduction, we need four basic operations to support the concept of explicit or heavyweight session management:

- *Create:* Users must be able to create a new session explicitly; that means a synchronous application will be initiated. The creation process at least needs the following information:

  – Which synchronous application is chosen for the session? At present, two typical real-time applications, i.e. chat tool and whiteboard, have been realized on the basis of the notification server.

  - Who will be allowed to join the session? By default, users who belong to the same group as the initiator of the session can normally feel free to join the session. Detailed description will be discussed in terms of privacy control later in this chapter.
  - What is the purpose of the session? It amounts to a short description about the session.

  In addition, information such as the duration of a session has to be specified as well. Otherwise, the session will be deleted by the system once all participants in the session have left.

- *Browse:* Users must be allowed to browse existing sessions currently active in the system, which provides the following information to users who are interested in attending sessions:

  - initiator of a session;
  - a list of participants in the session;
  - a short description of sessions;
  - starting and ending dates of sessions.

- *Invite:* The initiator of a session or others who have the necessary permission can invite others to join in. At present, the invited persons must be registered in the system. Again, the needed information for the invitation includes $user\_name$ and $port\_num$ of the targeted users.

- *Join:* Finally, a user can join a session either by accepting an invitation or by browsing existing sessions, provided that he has the related privileges to join the session. This adds his name into the list of the session participants and causes the application for the session to appear on his display.

Figure 5.18 shows the relationship between sessions and their participants. Whereas participants of $session_1$ are from the same room, the other two sessions are joined by members from different rooms.

Implicit Session Management

From the description of heavyweight sessions, we know that sessions must first be created before people can join them. Apparently, these kinds of sessions are more appropriate for formal, planned collaborations than for informal and spontaneous ones.

Nevertheless, the analysis in the preceding chapter has shown that most interactions in an everyday working environment are informal, brief, and spontaneous. *Lightweight* or *implicit* connections are supposed to support this kind of interactions. For Lecture 2000, lightweight

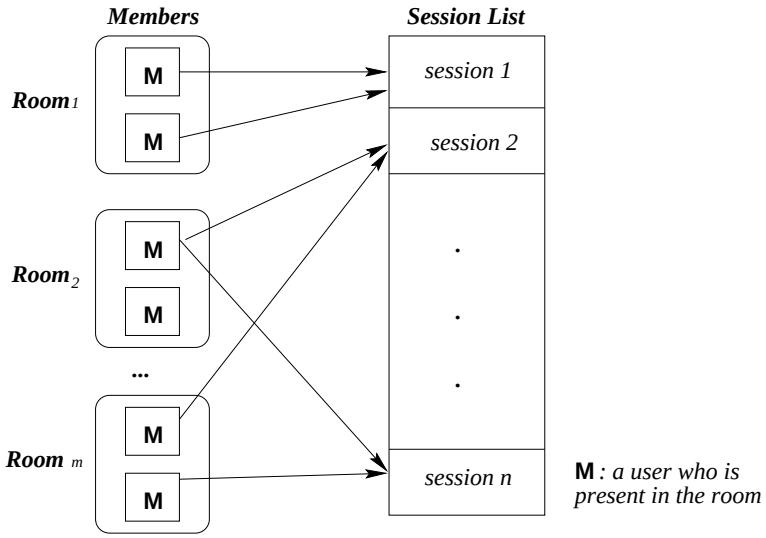


Figure 5.18: Relationship between sessions and members.

connections can play an important role in facilitating interactions among group members as well as between group and community members.

As opposed to heavyweight sessions, lightweight session management service will automatically detect potential opportunities for collaboration and then take appropriate measures to let related users engage in the collaborative process. There is no need for users to explicitly issue invitations or to browse existing sessions. At present, two mechanisms are primarily used to enable this kind of lightweight session management in CSCW:

- **Wandering** (or **Rendezvous**)

  Firstly, users in some systems wander through a virtual world and then have the chance to meet others who are already there. The best known examples are MUDs (see section 4.2.5). When a person connects to a MUD via a known Internet address and a port number, he enters one of several rooms where he can engage in a text-based chat dialog with all others in the same room. Another example is TeamRoom [Rosema96b] which is also based on the room metaphor. Similarly, users of TeamRoom can see what rooms are available, who is around, what rooms they are in, and how active they are. People can freely move between rooms. When they enter a room, they are connected with all the conferencing tools available in the room, and then join the session automatically.

  Similar systems of this kind also include other virtual worlds like media spaces (see 4.2.2). In principle, interactions in media spaces belong to lightweight connections because there is neither a session for people to invite others to join in nor a session for others to browse. Instead, people in media spaces often get into sessions spontaneously by scanning snapshots of offices or public places. The common feature is that all of them support informal and unplanned meetings. Sometimes, it is better to use the term *chance meeting* to refer to meetings of this kind.

- **Artifact-Based**

  Other systems supporting lightweight interactions are artifact-based [Edward94] or document-centric [Greenb96b]. In section 4.2.4, I have introduced the Piazza desktop system in which users who are working "*on the same data at the same time*" are brought into a session. A similar example is the system Intermezzo [Edward94], which detects potential collaborative situations by looking for overlaps or confluences in the activity information published by applications across the network. When two activity tuples contain the same object token, then the session management service can take action to allow the users to enter into a collaborative situation. In these systems, working on the same document means that they may have the same interest to exchange something. For example, when two persons read the same book, they may share the same education backgrounds and then there may be a potential collaboration between them.

Another example which may be classified into the artifact-based session management is lightweight connection in TeleNotes [Whitta97]. However, the use of documents for lightweight connections in TeleNotes is distinctive. Just like Piazza, Telenotes is also a desktop system intended to support brief, informal, unplanned and intermittent connections, and both systems exploit the presence of work-related artifacts such as a shared Web-page, an edited document etc. As opposed to Piazza, TeleNotes emphasizes the importance of two-person exchanges because the majority of office interactions are brief informal two-person interactions. The distinguishing feature of TeleNotes is that these work-related artifacts help manage the history and context of intermittent interactions between people. In everyday working environments, electronic documents and folders are sometimes left visible in the user's electronic workspace to serve as *reminders* and *context-holders* for urgent work in progress. Information in TeleNotes is therefore spatially arranged around the computer desktop in stacks, with each stack being relevant to a separate ongoing lightweight-communication task.

5.5.3 Lightweight Connections for Lecture 2000

Detection of Potentials

For supporting lightweight connections, Lecture 2000 takes advantage of the artifact-based virtual world. Firstly, it consists of a set of group rooms interconnected by the entrance hall. Secondly, each room is composed of various artifacts. Furthermore, Lecture 2000 integrates the wandering and artifact-based approaches described in the preceding section to support lightweight connections by defining a general term "*working on an artifact*". The term that a user is working on an artifact means that he is either already in the artifact (e.g. in the entrance hall, in a group room), or trying to open the artifact (e.g. open a container), or manipulating the artifact (e.g. a document). Typical scenarios of working on an artifact include:

- entering the entrance hall;

- entering a group room;
- entering a workplace;
- opening a container, and
- manipulating a document.

In terms of lightweight connections, entering a group room, for example, has now the same meaning as manipulating a document. For each artifact (room, workplace, container and document), an attribute *synGranu* is defined (see section 5.2.1). An artifact is called a *trigger* of lightweight connections if its *synGranu* has been set to be true. Once two or more persons are *working on such an artifact*, this system will sense the collaborative potential and attempt to initiate a session for this group of persons. If these persons do not oppose to engaging in a synchronous collaboration, a session will be created and initialized.

Just as described before, the artifact "entrance hall" is always set to be a trigger of synchronous collaborations (i.e. the attribute *synGranu* is always set for the entrance hall). The difference is that persons appearing in the entrance hall will not be asked about their willingness to participate in sessions in the hall. Instead, community members are automatically involved in sessions in the entrance hall.

For detecting opportunities of lightweight connections, we must check whether two or more users are working on the same artifact. For example, as a user tries to enter a group room, it must be possible for the system to check whether there are others present in this room at the same time. The system can get to know who are currently in the room by examining the client list associated with the room artifact in the notification server. Users who are working on the room artifact (e.g., they are currently entering the room, or they are already in the room) will appear in the client list of the room artifact.

Policies for Lightweight Connections

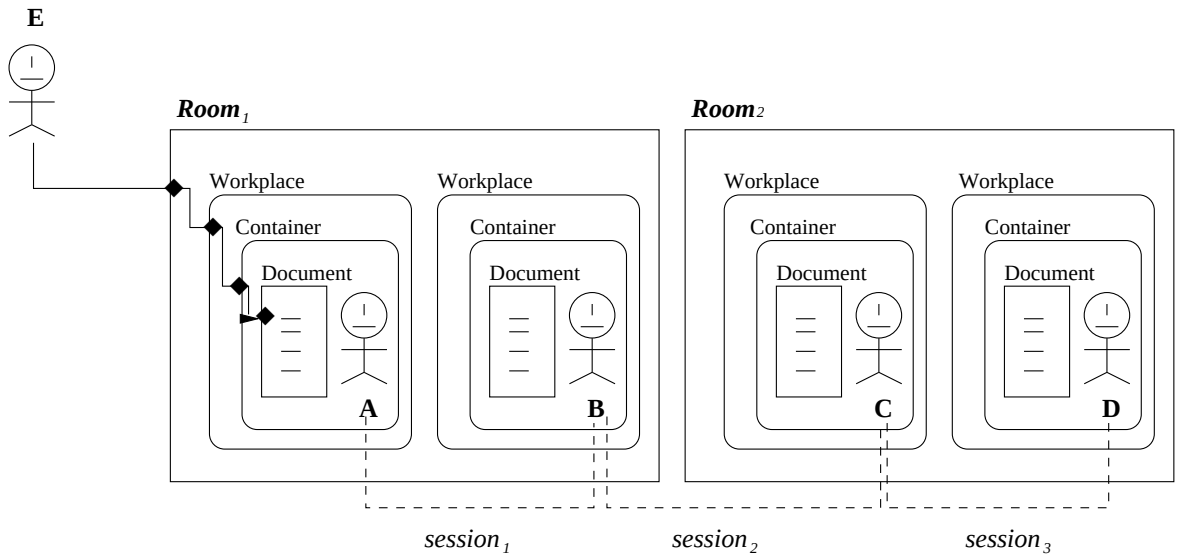
Initiating lightweight connections is sometimes more complicated than what we have introduced up to now. One reason is that artifacts in Lecture 2000 are embedded; that is, a room contains workplaces while a workplace contains containers and a container contains documents. The complexity of controlling the initiation of connections can be illustrated by a very simple example.

Suppose that user A enters a room where another user B is working in his own workplace (see Figure 5.19). If the room has been set to be a trigger, a session will be created between users A and B as user A enters the room. What will happen as a newcomer E tries to enter the same room, to open a workplace, to click a container, and finally to manipulate a document in the room? According to the definition of trigger, at least four same lightweight sessions will be created between users A, B and E. Of course, it should be avoided. Just one session is enough for the same group of people. Now we need to decide which artifact decides the initiation of lightweight sessions when the embedded artifacts are all triggers and a user "opens" them one after another.

For the convenience of the discussion, I define two terms. Artifact a is the *parent* of artifact b if a contains b. Conversely, b is the *child* of a. If room a contains workplace b, for example, we refer to a as the parent of workplace b. We have two policies for controlling the initiation of lightweight connections:

- **High-Level Control Policy**

  Users will be automatically linked together as long as one of the parents of the artifact on which they are working is set to be a trigger of lightweight connections.



Session 1 and 3: Lightweight connection **Session 2:** Heavyweight connection

Figure 5.19: Initiation of lightweight connections and management of multi-sessions

In Figure 5.19, users A and B are working on different documents in $Room_1$. Suppose that $Room_1$ has been set to be a trigger of lightweight connections. User E is trying to enter $Room_1$. Under high-level policy control for lightweight connections, a session has existed for user A and B because they are in the same room, provided that both of them have not refused to join in. As user E is trying to enter the room, he will also be automatically brought to this session regardless of whether the workplaces, containers or documents within this room are triggers or not.

- **Low-Level Control Policy**

  Users will be brought into a session only when they are working on the same artifact which is a trigger of lightweight connections.

user	$session_1$	$session_2$	$\cdots$	$session_n$
A	X		$\cdots$	
B	X	X	$\cdots$	
C		X	$\cdots$	
D			$\cdots$	
E	X		$\cdots$	

Table 5.1: List for both heavyweight and lightweight connections

Under the control of low-level policy, the parents of an artifact will not influence the initiation of lightweight connections. Suppose that all artifacts $Room_1$ are triggers. Since users A and B are in different containers and the low-level control policy has been chosen, no session has been created for users A and B. If user E entered user A's workplace, and is trying to manipulate on the same document on which user A is working, user E and A will engage in a lightweight session. Otherwise, there will be no session initiated for users A and E.

Avoiding Repetitions

No matter what policy is chosen, it is possible that the system sometimes attempts to initiate repeated sessions for the same group of users. Take the same example shown by Figure 5.19, and suppose the rooms adopt the high-level control policy and all artifacts are set to be triggers. Before user E comes to any room, there has been a session between user A and B, as well as one between user C and D. As user E tries to enter the first room, he will be brought into the $session_1$. If he continues to open user A's workplace, the system should not create another session for users E and A because users A and E have been engaged in a common session. If user E later enters into the second room, there should be a chance for him to engage in a lightweight connection with users C and D because E has no session with C and D up to now. The approach to solving this issue (i.e. avoiding repeated sessions) is to use the same session table mentioned in the last section to manage both heavyweight and lightweight connections. Table 5.1 shows the situation just discussed. Before a session is created, the system checks the session table to determine whether the latecomer should be brought into a session.

Summary

In summary, the steps to initiate lightweight connections in Lecture 2000 look as follows:

- *step 1:* A group room first chooses a control policy of lightweight connections (Who has the privilege to do so is determined by privacy control mechanism discussed in section 5.6.3 on page 147). Normally, the high-level policy is suitable for rooms where group members are more willing to join lightweight connections. Otherwise, the low-level policy should be selected.

- *step 2:* As a newcomer attempts to enter a group room, to open a workplace, to click a container or to manipulate a document, the system checks the attribute $synGranu$ of the artifact (i.e. either the room, the workplace, the container or the document) to see whether the artifact is a trigger.
- *step 3:* If yes, the system continues to check out who are now working on the artifact by examining the client list of the artifact. Note that people included in the client list may have not joined in a session. For example, they just do not like to talk about with each other at this moment.
- *step 4:* Check the session table to avoid repeated sessions for the newcomer, i.e. check whether a session between the newcomer and those in the client list has existed.
- *step 5:* If no session has existed between the newcomer and others included in the client list, initiate a session for them, provided that they do not refused to join in.

5.6 Privacy Control

5.6.1 Goals and Policies

The term *privacy control* has been mentioned in this thesis several times largely because the use of CSCW systems has raised a number of serious issues concerning protection of privacy in the past few years. As Edwards argues, collaborative systems provide "*a rich but potentially chaotic environment*" for their users [Edward96]. In Lecture 2000, such a problem would become more serious because both community members and group members are working in the same environment. More importantly, we encourage community members to enter group rooms. In contrast to Lecture 2000, many CSCW systems allow only authorized persons to enter and to participate in activities occurring there.

Any privacy control mechanism is supposed to pursue some goals. The goals of privacy control for Lecture 2000 can be identified as follows:

- Basically, the privacy control mechanism should help to *prevent intrusiveness*, an issue extensively tackled by systems such as media spaces (see section 4.2.2), and *protect users' artifacts from being misused*.
- Community members can work within the system *in a predictable and manageable way*. For example, when group members leave an artifact in their containers, they must be able to exactly know what may happen with the artifact. Only so, people can just feel free to leave their artifacts in the virtual world to share with others.
- This system should adopt *a number of default security policies* and require as few as possible active efforts by users to protect their privacy. Moreover, the potentials of the artifact-based model must be exploited to ease the task of privacy protection.

- Privacy control means to take measures to constrain accesses to artifacts. Our last goal is, then, that this mechanism should not stifle interactions between people. Instead, this mechanism should be *both strict and flexible enough* so that we can encourage interactions between participants and make the system more open. Making the system more open means that group members are more willing to use the system and to share their artifacts with others because the security mechanism in the system is believable. It also means that community members have still chances to enter these group rooms in spite of strict security measures.

In some CSCW systems, *policies* are introduced for overcoming their inherent dynamics. According to Edwards:

Definition 5.4 (Policy)
A policy is a general principle or plan that guides the actions taken by a person or a group. In other words, policies govern the particulars of how users and applications interact with one another [Edward96].

Users will then work and interact within the systems in a predictable and manageable way. In Lecture 2000, the general policy for controlling privacy protection is very simple:

> *Group members' work should be efficiently protected. Meanwhile, community members should be allowed to share group members' work regarding learning activities as much as possible.*

Here we have defined competing policies but not mutually exclusive. The above general policy involves the following concrete points:

- Group members can normally concentrate on their groupwork no matter who has entered their room. They will not be brought into sessions unless they like.
- Artifacts of a group member can be effectively protected from being misused by community members and other group members. Only users who are granted the associated access privileges can make certain access to them.
- What a specific user can do with respect to a specific artifact will become a common knowledge to users in the system for avoiding unnecessary attempts and misunderstandings. For example, a community member knows what operations are allowed and what operations are forbidden with respect to a to-do list.

Section 4.2.3 already discussed privacy control in media spaces, identifying four different types of control models. For Lecture 2000, the privacy of users will be protected by exploiting two basic approaches: one is the spatial approach, another is realized with role-based access control to artifacts.

5.6.2 Spatial Approaches

How spatial concepts work to prevent intrusiveness can be illustrated by the comparison between Lecture 2000 and OfficeWalker. In terms of privacy control, Lecture 2000 has some similarities with OfficeWalker although we have developed our system independently:

- Both systems distinguish between public places and private places (see section 4.2.3) for preventing intrusiveness. In OfficeWalker, a public place is a hallway shared by neighbors who are defined independent from their physical locations, and a private place is used by these neighbors as their offices. Similarly, Lecture 2000 uses the entrance hall as its single public place. Private places are learning groups inhabited by group members.

- In OfficeWalker, a caller visits a recipient (see Figure 4.2.3 on page 77) from a public place instead of directly jumping into the private place of the recipient. Similarly, in Lecture 2000, a community member first enters the entrance hall, where he can choose an appropriate room to enter according to exported information by learning groups.

However, there are essential differences between OfficeWalker and Lecture 2000:

- The public places in OfficeWalker and Lecture 2000 are intended to foster interactions. But OfficeWalker supports only interaction between public places and private places. Lecture 2000 also supports interactions among community members within the entrance hall, which means that users themselves in the entrance hall can talk with each other and even set up spontaneous rooms to in the system.

- In OfficeWalker, the distance between a caller and a recipient is used to determine when to initiate a conversation between them. It is assumed that staying at a short distance for a while without having a conversation is uncomfortable for both of them. In Lecture 200, even entering a room does not mean that a conversation will certainly happen. The initiation of a session depends on a lots factors, e.g. whether the room is a trigger of lightweight connections and whether involved users are willing to participate in a talking (see section 5.5).

Finally, private rooms in Lecture 2000 (i.e., learning groups) are not actually private. Indeed, they are used to structure groupwork. On the other hand, they are working places for people who share learning work and they facilitate collaboration.

Briefly, spatial approaches usually exploit two concepts to prevent intrusiveness:

- public and private places;

- distance between people.

5.6.3 Flexible Roles and Access Control

Role-Based Access Control

Role-based access control has been extensively addressed in areas like information systems [Gladney97, Sandhu96] and CSCW systems [Fish88, Edward96, Smith98]. The motivating impetus is the intuition that in most organizations access control decisions for performing actions are based on the appropriate role rather than on individual people [Moffet98]. With role-based access control, system administrators create roles according to job functions performed in a company or organization, grant permissions (access authorization) to those roles and then assign users to the roles on the basis of their specific job responsibilities and qualifications. There is usually a set of persons who may assume a certain role.

Definition 5.5 (Role)
A role is a category of users within the user population of a given application; all users in a certain role inherit a set of access control rights to objects within the application [Edward96]

Roles are particularly common in shared editors such as Quilt [Fish88]. Quilt supports different rights to the information depending on the relationship between user's social roles, the nature of the information, and the stage of the project. Each user of Quilt is assumed to have a specific role with regard to a particular document at a particular time, and categories of usage privileges can be associated with each role. Quilt supports roles for *writers* who are allowed only to change their own work, *readers* who are not allowed to modify the document, and *commentators* who can only add margin notes (i.e. comments and annotations) to the document.

In general, a role-based access model can be presented by a triple [Shen92]:

$$(S, O, A)$$

where S is a set of *subjects* (i.e., users wishing to access data), O is a set of *objects* (i.e., units of data that may be accessed), and A is a matrix with rows representing subjects, columns representing objects and $A[s, o]$ denoting the *access rights* (i.e., privileges that are needed to do certain operations on objects) of subject s with respect to object o. In other words, access control determines who can access what and in what manner [Ellis91].

Drawing on these discussions, we know that implementing a role-based access control model needs to go through the following steps:

1) define roles

2) define access privileges (or access rights)

3) grant roles to users (or specify membership of a role)

4) grant access privileges to roles

Static Roles

Definition 5.6 (Static Roles)
A role is static if its membership is specified in advance and rarely changed.

A static role is usually defined in terms of the users who are members of the role. In the beginning of this chapter, we have introduced that Lecture 2000 distinguishes between community members and group members. For controlling access rights to artifacts in rooms, we classify both community and group members into three sets, each of which represents a role. The three roles are *owner*, *collaborator*, and *visitor*, all of them are static because the membership of these roles can be, and usually, is predefined. There are three possibilities to specify members for these roles:

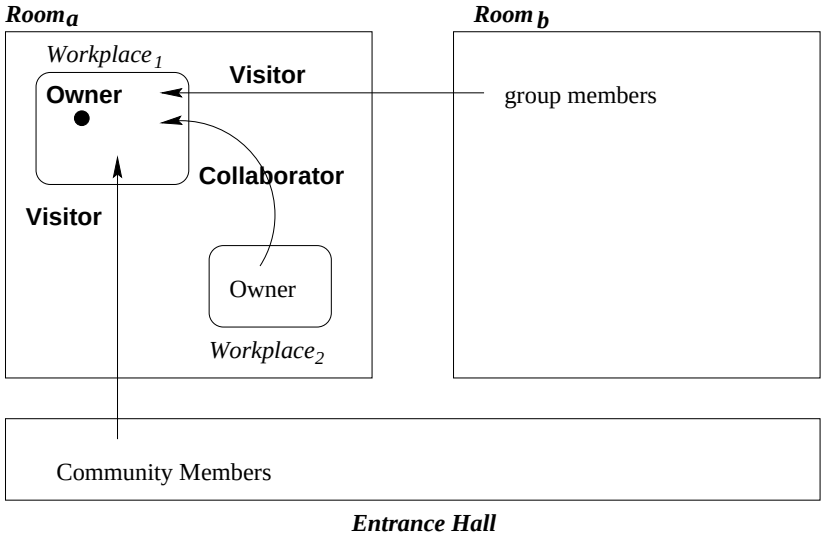


Figure 5.20: Role transformation with respect to a workplace

Default Specification: By default, the following rules govern the membership of static roles with respect to an artifact (see Figure 5.20):

- *owner*

  An artifact's owner is someone who has created the artifact. For example, a room owner has created the room and is in charge of this room's management. Similarly, a workplace owner is normally a group member in this room. As a room is created, the system will automatically generate a workplace for its every member. As illustrated by Figure 5.20, the owner of $Workplace_1$ is a group member of $Room_a$.

  The owner of documents and containers within a workplace is the same as the owner of this workplace in which these documents and containers are located, regardless of who has created these documents. For example, a community member can leave a note in a user's container as described before, but the owner of the note is the workplace owner.

- *collaborator*

  A collaborator with respect to an artifact is usually a group member of the room where the artifact resides. In other words, collaborators are "roommates" of a group. In Figure 5.20, the group member who owns the $Workplace_2$ is a collaborator with respect to the $Workplace_1$.

- *visitor*

  Visitors to an artifact are those who are not group members in the room in which the artifact is located. Thus, community members and also those from other groups are visitors to the artifact. In Figure 5.20, group members from the $Room_b$ and community members from the entrance hall are all visitors to $Workplace_1$.

Although the membership of a static role is specified in advance and changes rarely, the role of a person in the spatial environment may change over time. A person may assume different roles depending on the current context. For example, as Figure 5.20 indicates, group members in the $Room_b$ will become visitors to artifacts in the $Room_a$ when they have entered the $Room_a$.

These default rules enable us to implicitly specify members of a static role. But it is sometimes desirable to allow authorized users to explicitly specify memberships of static roles for an artifact. This can be achieved through the introduction of explicit rules: positive and negative rules.

Positive Specification: It explicitly lists user names for a role. The form for positive specification of membership is:

$$(user_1, user_2, \cdots, user_n)$$

A special value for the positive specification is all, which means anyone who has logged into Lecture 2000 is a member of the defined role.

Negative specification: specifies users who should not be a member of the role. The negative specification may have two forms:

- $not(user_1, user_2, \cdots, user_n)$: it explicitly defines that $user_1, user_2, \cdots, user_n$ can not assume this role.
- $null$: no user can take the role, meaning that nobody can access this artifact now.

Dynamic Roles

Studies of access-based privacy control show that a static role in CSCW systems is too rigid to support effective collaboration. Sometimes, dynamic roles are desirable in a collaborative environment.

Definition 5.7 (Dynamic Roles)
A role is dynamic if its membership is determined at runtime.

Therefore, dynamic roles are defined in terms of rules instead of their members. Here are some useful examples:

- *Example 1:* In a group room, a collaborator is normally a group member, and we see community members as visitors to artifacts within the room. However, one may want to specify collaborators dynamically as:

  "*All people who visit my room as I am there.*"

  According to this rule, normal visitors to the room, i.e. community members and users from other groups, will become collaborators to the user's artifacts as long as this user is currently in his own room. Thus community members will have the same access privileges to the user's artifacts as group members of the room.

- *Example 2:* Community members normally assume the role of visitor. A user wants to forbid all visits to his script container by visitors this Friday because he is going to prepare for an examination using documents in the container. Except for this Friday, visits to this container by users assuming the role visitor are allowed. Thus, he can define members of the role visitor to his script container dynamically as:

  "*All people who visit my script container except for this Friday.*"

  Thus community members can not assume the role "visitor" this Friday[4]. At that time, community members can not view learning scripts within the user's script container.

Thus, membership of a dynamic role will vary from time to time during the lifetime of a room. The essential difference between static and dynamic roles is that we *describe* rather than *specify* the membership for dynamic roles. In such a way, artifact owners can be relieved of some of the burden of tracking, updating, and anticipating role membership explicitly.

Lecture 2000 is developed on the top of the Informix database system. A dynamic role is associated with a predicate which consists of a set of clauses. The clauses for defining a dynamic role in Lecture 2000 are realized by using Informix SPL routines[5]. The reason for choosing SPL to realize the dynamic clauses is that the evaluations of SPL functions are faster than those written in other languages such as Java. Currently, we support the following SPL functions:

- check the current access date to the artifact

  $access(date)$: if the access date to the artifact is $date$, the function returns true.

- specify the user who accesses the artifact

  $subject(user)$: if $user$ accesses the artifact, the function returns true.

[4]To achieve this goal, one must set the static role's value of the script container as *null* because when the evaluation of a role's dynamic definition returns false, the system will check the role's static value. The evaluation order of roles will be discussed soon.

[5]A SPL routine is a user-defined routine written in Informix Stored Procedure Language(SPL). A SPL function is a routine written in SPL and SQL that returns a single value. A detailed description of SPL can be found at *http://www.informix.com*.

- specify whether the owner is in the group room

  $present(user)$: If $user$ is in the room, the function returns true.

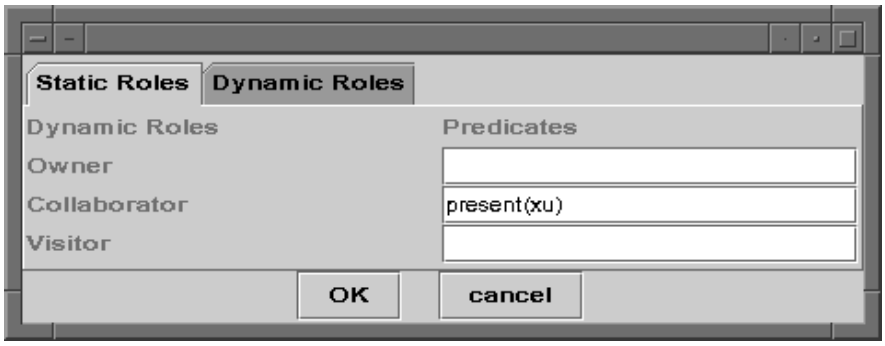


Figure 5.21: Define dynamic roles

Of course, more functions are expected to be realized in the future. At present, a valid predicate defining a dynamic role can simply have one of the following forms:

- AND formula: $clause\ and\ clause\ and\ \cdots$
- OR formula: $clause\ or\ clause\ or\ \cdots$
- NOT formula: $not(clause)$

With these clauses, we can now very easily define dynamic roles. Figure 5.21 defines the dynamic role "collaborator" for the room described in the example 1. According to this definition, all community members can take the role of collaborator when the user xu is currently present in the room.

In a system which supports both static roles and dynamic roles, we must determine a sequence for evaluating the roles of the user who issues an access request:

- first dynamic roles then static roles

  Dynamic roles of a user will be first evaluated because they may reflect the current privacy control requirements imposed on an artifact. Take the second example described above, all people can visit the user's script container except for this Friday, regardless of the static roles defined by this user.

- first senior roles then junior roles

  Later in this section, a role hierarchy which distinguishes between senior and junior roles will be discussed. For example, an owner is a senior role while a visitor is a junior. Senior roles of a user with respect to an artifact must be first evaluated. It is straightforward that a user takes the most senior role if several role predicates return true.

Artifact (ID)	Static Roles			Dynamic Roles		
	owner	*collaborator*	*visitor*	*owner*	*collaborator*	*visitor*
R0001					*present(xu)*	
R0002		*not(xu)*				
W0001						
...			...			...
D0001		*(buerger, teege)*	*null*			

Table 5.2: Management of static roles and dynamic roles

For simplifying the management of roles and facilitating role evaluation at runtime, a table is used to define and manage both static and dynamic roles. In table 5.2, the artifact $D0001$, for example, defines only user $buerger$ and $teege$ as its collaborator and its role definition for visitor is null. This means that no community members can share this artifact and only user $buerger$ and $teege$ can share the artifact as collaborators. Similarly, the artifact $R0001$ defines a dynamic role for a collaborator. The meaning of the predicate $present(xu)$ has been explained before. Note that if the predicate definition for a role is left vacant, it means that the corresponding dynamic role is not defined.

Access Privileges

After specifying the roles, we require a mechanism for defining, changing, and enforcing their associated access rights and responsibilities. To minimize the complexity of the role-based control model, we define three relatively fixed access privileges for roles:

- ADMINISTRATION,
- COLLABORATION, and
- VISITING.

Apparently, an owner normally has the access privilege of ADMINISTRATION, a collaborator that of COLLABORATION, and finally a visitor that of VISITING.

These access privileges represent different valid operations on artifacts:

◇ **Room-Level Privileges**

- VISITING
  A user with the VISITING privilege is authorized to:
  - Enter this room, provided that the room's door is open;
  - Participate in sessions held in this room, provided that he is invited or the room is a trigger of lightweight connections;

  - Enter workplaces within the room, provided the user has the necessary workplace-level privileges.

- COLLABORATION

  In addition to privileges of VISITING, a user with the COLLABORATION privilege can do the following things:

  - Enter this room, regardless of the status of the room's door;
  - Create, attend and invite others to sessions in this room;
  - Enter workplaces regardless of the access restrictions of workplaces.

- ADMINISTRATION

  In addition to the capabilities of the COLLABORATION privilege, a user with the ADMINISTRATION privilege can perform the following tasks:

  - Accept new group members, i.e., create a new workplace for the newcomer;
  - Remove a group member and delete its workplaces;
  - Set up room-level privacy control policies;
  - Grant room-level privilege ADMINISTRATION to other users.

◇ **Workplace-Level Privileges**

- VISITING and COLLABORATION

  A user with VISITING or COLLABORATION privilege can perform the following task within this workplace:

  - Open containers found within this workplace, provided that the user has the necessary container-level privileges.

- ADMINISTRATION

  In addition to the capabilities of the COLLABORATION privilege, a user with the ADMINISTRATION privilege can perform the following tasks:

  - Create or delete containers within this workplace;
  - Grant workplace-level privilege ADMINISTRATION to other users.

◇ **Container-Level Privileges**

- VISITING and COLLABORATION

  - View documents within this container, provided that the user has the necessary document-level privileges;
  - Leave a message (generate a document) in the coordination or notification container.

- ADMINISTRATION

  In addition to capabilities of a COLLABORATION privilege, a user with the ADMINISTRATION privilege can perform the following tasks:

  - Grant container-level privilege to other users;

  - Delete any documents within this container, regardless of the document owner.

◇ **Document-Level Privileges**

- VISITING
  - View the document.
- COLLABORATION
  - Comment, annotate this document;
  - Copy the document, and add it his own workplace if he is a group member in this room.
- ADMINISTRATION

  In addition to capabilities of a COLLABORATION privilege, a user with the ADMINISTRATION privilege can perform the following tasks:
  - Create a new document;
  - Update and delete a document.

Originally, these access privileges should be granted to roles. As mentioned before, we do not support directly granting access privileges to roles in Lecture 2000. Instead, each role is regularly associated with an access privilege defined above. The flexibility of access control to artifacts is realized by combining the abilities of static and dynamical roles specifying variable memberships.

The correspondences between roles and access privileges are described as follows:

- visitor: A visitor to a group room normally has the VISITING privilege of all artifacts within this room unless one or more of these privileges have been revoked from the visitor. Therefore, a visitor has the room-level, workplace-level, container-level, and document-level VISITING privileges. A user who is a member of the role visitor can usually enter a room, open a workplace, and view documents of containers. Such default privileges can be canceled by updating the role definitions with respect to an artifact.

- collaborator: A collaborator normally has the access privilege COLLABORATION to all artifacts in a room. In addition to what a visitor can do, a collaborator (they are group members in this room) can comment, annotate documents within a container, or leave a message in awareness containers (i.e., notification and coordination containers).

- owners: Being assigned the ADMINISTRATION privilege, an owner has all capabilities of a visitor as well as those of a collaborator. In addition, he can update the artifact, and create a new child (e.g., artifacts which are included, such as containers of a workplace) of the artifact.

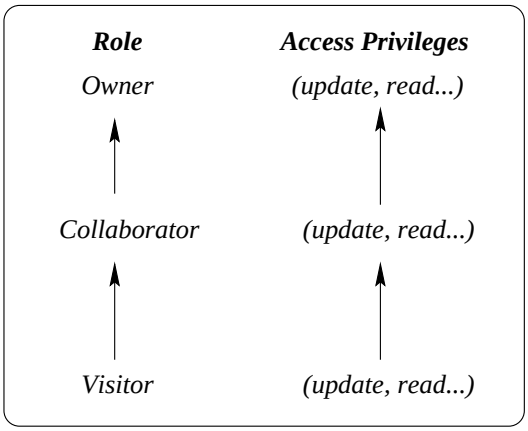


Figure 5.22: Role hierarchies and access privileges in Lecture 2000

Hierarchies and Inheritances of Roles and Access Privileges

In addition to the definitions of roles and access privileges, we must take into account two other issues:

- *role hierarchy* and
- *role inheritance*.

Both of these issues can ease the implementation of the role-based privacy control model. With role inheritance, there is no need for users to define a role for each artifact in a group room because roles for artifacts can be implicitly inherited and then the access privileges can be inherited as well.

For an organization, hierarchies are a natural means of structuring roles to reflect the organization's lines of authority and responsibility. In Lecture 2000, the roles also have a hierarchy as depicted by Figure 5.22. More powerful (senior) roles are shown at the top of this diagram and less powerful (junior) roles towards the bottom. Such a role hierarchy means that senior roles can inherit all access rights of junior roles. In this example, collaborators can automatically inherit all access privileges of visitors, and owners automatically have access rights of collaborators. Such inheritance has been illustrated as we tried to define access privileges for artifacts in the preceding section.

A remaining question is whether we need to define roles for each artifact within a group room. Fortunately, the answer to this question is "No". Because of the hierarchy of artifacts within a learning room (see Figure 5.23), a child artifact can inherit role definitions of its parent artifacts (i.e. its static and dynamic role definitions). For example, role definitions of a document are inherited from its parent container by default. Similarly, role definitions of a container are the same as that of its parent workplace. But an exception to the role inheritance between artifacts is that the role "owner" of a workplace must not be the same as the owner of its parent room. Instead, a workplace's owner is a group member of the room who possesses the workplace, but he may not have created the room for this group.

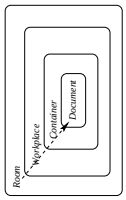


Figure 5.23: Role inheritance for artifacts

5.6.4 Summary

So far, we have defined the following concepts:

- *roles*: we have defined static roles and dynamic roles definitions for artifacts. And roles between artifacts can be inherited.
- *access privileges*: we have defined the three access privileges VISITING, COLLABORATING, and ADMINISTRATION, each of which is implicitly associated with a role.
- *objects*: we have defined four artifacts, namely room, workplace, container, and document.

Therefore, all the three elements given by the general role-based access model (S, O, A) (see section 5.6.3) have been discussed and defined. With the help of these definitions, let us see how the role-based access model is used to protect users' privacy. The process to protect a user's privacy can be observed in several steps:

- *step 1:* define roles

  As an artifact is created in Lecture 2000, it will inherit role definitions from its parent artifact. If the newly created artifact is a room, its static roles have the default values (e.g. the group members of the room assume the collaborator role of the room artifact

and all other users are visitors to the room). If needed, the owner of the room can change its static and dynamic role definitions as described before.

- *step 2:* evaluate roles

  When a user (e.g., a community member) attempts to work on an artifact, the system first of all determines the user's role. As discussed before, the system first checks dynamic role predicates defined for the artifact. If no predicates for dynamic roles have been defined or all the evaluations of predicates return false, the user will take the default role visitor because he is a community member, which is determined as the user logged on to this system (see 5.1 on page 99).

- *step 3:* determine access privileges

  Each role is assigned to corresponding access privilege. If the user is a community member, he assumes the role visitor. Then he is given the access privilege of VISITING. The earlier definitions of access privileges associated with VISITING have described what the user can do with rooms, workplaces, containers and documents respectively.

Finally, let us briefly see whether our goals of privacy control defined before (see section 5.6.1) have been satisfied. The first goal of privacy control is set for users to work in the environment in a predictable and manageable way. This goal is certainly fulfilled because users' behaviors have been confined to those which are explicitly defined by our access privileges. A group member knows, for example, what a community member can do in his workplace if he allows the role of visitor for his workplace. Moreover, both roles and access privileges could have default values. Often, the default settings are sufficient, and user customization of role privileges for individual artifacts are not required. Furthermore, the role definitions of artifacts can be inherited. This has certainly met our second defined goal; that is, we have defined a set of default security policies to easy the task of controlling privacy. Finally, the combination of static and dynamic roles makes privacy protection flexible.

6 Implementation of the System

This chapter is primarily concerned with the implementation of the notification services and the management of electronic learning material.

6.1 Notification Services

6.1.1 Artifact-Based Notification Model

In the last chapter, a notification server model and related notification services have been defined. The most remarkable features of the notification server are that it is based on the artifact model and that each artifact is defined by a set of states in the server. To implement the notification server for Lecture 2000, I choose to extend Lotus NSTP (see section 5.3.2 on page 120) because of the following obvious advantages of NSTP:

- NSTP provides a basic infrastructure for notification services;
- It allows centralized state sharing and then makes it possible to support both synchronous and asynchronous awareness;
- Its client-based semantics feature makes it possible to apply NSTP to Lecture 2000;
- The current version of NSTP is written in Java, which offers a lots of outstanding benefits for the development of distributed applications.

An Application Example of NSTP

To introduce NSTP's model to Lecture 2000, let us first see how NSTP implements a simple groupware system. Scribble is an applet of NSTP, which implements a shared whiteboard [Day97]. Each Scribble client can choose one out of several drawing and editing tools to modify the shared whiteboard. As introduced before, NSTP is based on the concepts of Place and Thing. A Place simply contains the shared state for an application. Things are the keepers of state within a Place. The shared whiteboard Scribble is implemented as a single Place. Each distinct drawing object in a shared whiteboard is represented as a Thing in that Place, whose value consists of the information needed to construct the drawing object, such as its type, size, position, and color. Once a user is in a Place, he receives notifications about

any state change of Things in that Place. Once the user leaves a Place, he no longer receives notifications from that Place.

In general, a NSTP Place encapsulate two parts. The first part consists of a set of threads, each of which maintains a connection between the notification server and a client who is connected to the Place. The thread handles requests sent on that connection, i.e. notifying the client of changes of Things in the Place. In NSTP's current version (i.e. PlaceHolder Version 1.0), connections between clients and the server are realized through sockets.

The second part of the Place encompasses all Things. In the Scribble example, if a rectangle is created in a user's board, a Thing representing the rectangle object will be created. The value of the Thing is a string with the format $colorfill\_patternx1y1x2y2$. Here, $(x1, y1)$ and $(x2, y2)$ specify the object's starting and ending positions respectively. And the Thing's type is "Rectangle". Once the Thing's state is changed, each thread in the first part notifies all related users of the change. Further, the attribute "type" of the Thing is used for clients to interpret the semantics of the value in a right way, i.e. drawing a rectangle in their own board using information represented by the string $colorfill\_patternx1y1x2y2$.

Implement the Artifact Model

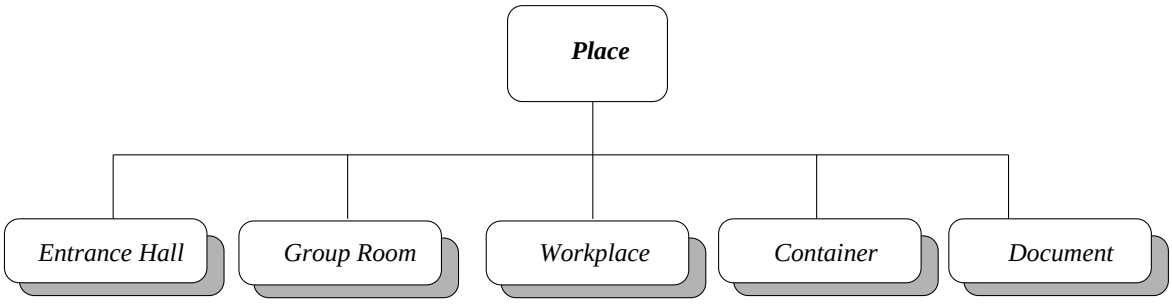


Figure 6.1: Each artifact is defined as a NSTP Place

Since NSTP has already implemented the basic notification functions for groupware (e.g., creating, changing, and deleting Things, and handling notifications sent by the server in response to changes of the status of Things), our task is to implement the notification server model defined in the last chapter (see Figure 5.13 on page 123) with NSTP's Places and Things. In Lecture 2000, the notification server looks as follows:

- All units in the notification model of last chapter are represented as a Place of NSTP, namely all artifacts (i.e. rooms, workplaces, containers and documents) are represented as a Place on the server side (see Figure 6.1).
- Features of an artifact are described by a set of Things of the Place. Since all artifacts have some shared attributes (e.g. artifacts' ID, name, and description), $Thing_1$, $\cdots$, and $Thing_7$ of a Place are used to specify these shared attributes. A detailed description about the allocation of shared Things can be found in Table 6.1. The purpose of

grouping all shared attributes together is to make it easy to implement some functions common to all artifacts, e.g. history management (see section 6.3.4).

- $Thing_8, \cdots,$ and $Thing_m$ are left for special use of each kind of artifacts. For example, workplace artifacts use $Thing_8$ to differentiate public workplaces from private ones.

Note that $Thing_7$ links all child artifacts in a room together. Figure 6.2 shows the relationship between Place, Thing and artifacts.

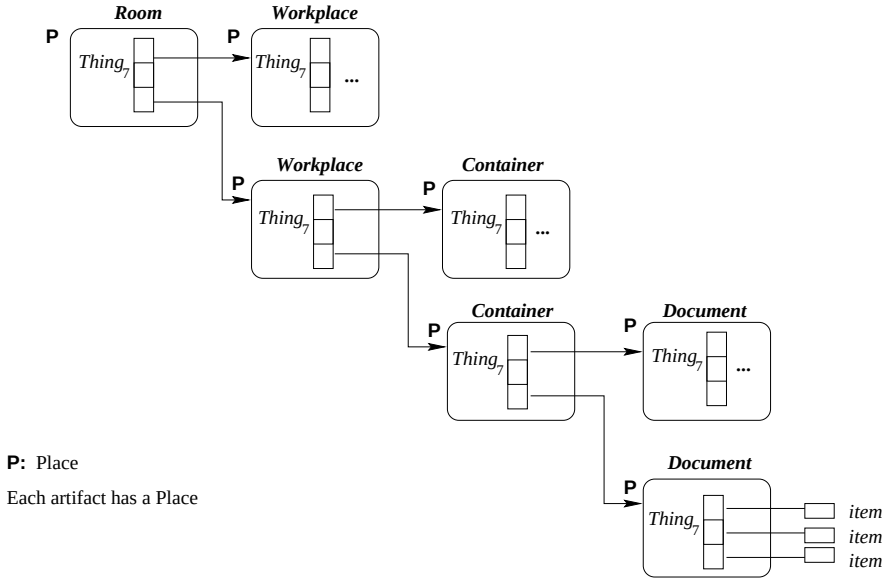


Figure 6.2: The relationship between Place, Thing, and Artifact

Let us briefly show how special Things of a Place are used to represent a group room. In addition to shared states represented by common Things (i.e. $Thing_1, \cdots, Thing_7$), a group room owns other shared states, some of which can be represented by special Things as follows:

- *group identifier* $\leftrightarrow Thing_8$: which group inhabits this group room? Its value is a string pointing to a learning group.
- *state of room door* $\leftrightarrow Thing_9$: is the group room allowed to enter now? It has three possible integer values, i.e. 0 representing an opened door, 1 a closed door, 2 a door which is ajar.
- *group meeting date* $\leftrightarrow Thing_{10}$: Its date type value tells community members when the next meeting in the room will be held?
- *lightweight connection policy* $\leftrightarrow Thing_{11}$: what policy for light connections (i.e. high-level or low-level control policy) is adopted by this room (see section 5.5.3)? Integer values 0 and 1 denote these two possible policies respectively.

	Thing	**State Name**	**State Description**
Group 1: Common Things	$Thing_1$	ID	The identifier of a Place on the server. Each Place on the server has a unique ID.
	$Thing_2$	$name$	Besides the unique ID, a Place may have a name. A Place's name can be used to locate the Place on the server-side if it is unique, or displaced in user interfaces.
	$Thing_3$	$description$	This attribute gives a brief description of the artifact. For a group room which the Place denotes, this Thing gives a brief introduction to the related learning group.
	$Thing_4$	$icon$	If necessary, an icon for the Place provides an image denoting the artifact. For example, a HomeFolder icon is used to represent a user's private workplace (see Figure 5.4). Three different door icons are used to represent three possible states of a learning group.
	$Thing_5$	$synGranu$	This Thing realizes the *trigger* concept. If the value is set, people who are currently working on the artifact will be brought into a session if desired.
	$Thing_6$	$creation\_date$	When was the artifact created?
	$Thing_7$	$children$	This Thing's value has the form "$children\_num\ child_1\ child_2\ \cdots\ child_n$". $child_i$ is the ith child's Place ID
Group 2: Special Things	$Thing_8$ $\cdots$ $Thing_m$	special Things	$Thing_9\ \cdots\ Thing_m$ left for each artifact. For example: document artifacts use these Things to represent the documents' content.

Table 6.1: Allocation of Things denoting shared states of artifacts

So far, we have implemented an artifact-based notification model with the concepts of Place and Thing from NSTP. The discussions in the preceding chapter have also summarized functions to be supported by the notification server. The next two sections will briefly introduce their implementations.

6.1.2 Artifact Management Services

Create and Delete Artifacts

As discussed before, a Place refers to an artifact (e.g. a room) in Lecture 2000. The server must first of all allow authorized clients to create and delete Places. Since the server has client-side semantics, i.e. the server does not know what kinds of shared states must be created for a Place and what these shared states really mean, we must provide server- and client-side functions respectively to enable user programs to manipulate a Place:

Client-side Functions	Server-side Functions

`place = new Place(name, type)`
`createPlace(server, place)`
`getPlace(server, placename)`
`removePlace(server, place)`
	`createPlace(place)`
	`getPlace(placename)`
	`removePlace(placename)`

Arguments used by these functions are as follows:

- `server:` Generally, a system may choose to run a server for a lecture or for all lectures of a tutor. In the first case, all students who visit the same lecture will build a community sharing sources in the server. In the second case, students who are lectured by the same tutor build a community. Lecture 2000 currently takes the first case and runs only a server. Even so, it is reasonable to use the `server` argument to specify a unique server for client programs.
- `placename:` It is desired for client programs to use a name to access places. If an duplicated names is used, the system will return its first match found in the specified server.
- `type:` Originally, NSTP supports only a Place type `NS:PlaceType`. For simplifying the implementation of our artifact model, we define four place types in Lecture 200 as follows:
  - `ROOM_PLACE`,
  - `WORK_PLACE`,
  - `CONTAINER_PLACE`, and
  - `DOCUMENT_PLACE`.

Thus, the argument `type` contains necessary semantic information so that client programs can interpret shared states of a Place easily and correctly.

Create, Update and Delete States

A Place exposes methods available for clients to manipulate Things within a Place. The following methods can create, delete Things in a Place:

```
createThing(attributes, value, place)
createDefaultThings()
createDefaultSharedThings()
createThing(thingSpec)
deleteThing(thingName, user)
getAttributes()
```

Access Control

The previous chapter has extensively addressed a role-based privacy control model. Briefly, each user in the system assumes a role who is granted a set of access rights with respect to an artifact. As summarized in section 5.6.4, it needs to go through some necessary steps to calculate a user's role and relevant access rights. We have also argued that as a client issues a request to access an artifact managed by the notification server, the server will first of all check whether the client has required access rights. Thus access rights in the role-based control model are finally transfered into *create*, *read*, *change* and *delete* defined by the notification server in section 5.3.3.

In summary, the following functions are required to determine whether a user is authorized to manipulate an artifact's Place and its Things:

```
getRole(place, client)
getCreate(place, role)
getRead(place, role)
getChange(place, role)
getDelete(place, role)
```

The `getRole` methods computes client's role with respect to a Place representing an artifact. As discussed before, the system will first of all check the user's dynamic roles and then static roles. According to the resulted role which the client assumes, the system will call related `getXXX` methods to check the client's access rights to the targeted Place and its Things.

6.1.3 Notification Services

Subscribe and Unsubscribe Notification Services

To get notifications about an artifact from the server, a client must first of all register himself with the Place representing the artifact on the server side. And only users who have subscribed for notification services of the place can manipulate Things within the Place. After the unsubscription, the client will not receive notifications about the Place any more. If the Place denotes a room, these two actions are just equal to the user entering and leaving the room. The subsequent two methods are needed for users to enter and then leave a Place:

```
enterPlace(connection, msg)
leavePlace(connection, msg)
```

The *connection* argument defines a communication channel between a client and the server, and *msg* shows a short message about the actions. Figure 6.3 shows that each Place encapsulates a set of threads, with each of them handling a single connection between the server and a client who has entered the Place.

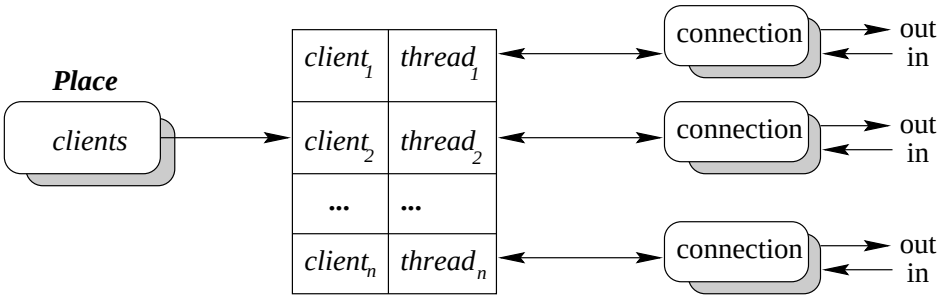


Figure 6.3: A Place encapsulates threads handling connection between clients and the notification server

Notification Transfer

In the current version, connections between the notification server and clients are implemented with Java sockets. A socket is one end-point of a two-way communication link between a client program and a server program. The `java.net` package provides two classes `Socket` and `ServerSocket` that implement the client side and the server side of the connection, respectively. Normally, a server's socket is bound to a specific port number. The server just waits and listens to the socket for a client to make a connection request. On the client side, the client knows the hostname of the machine on which the server is running and the port number to which the server is connected. To make a connection request, the client tries to rendezvous with the server on the server's machine and port.

```
connect(server, port)
connect(client, port)
sendMsg(outgoing)
receiveMsg(incoming)
```

The two `connect` functions are used by clients and the server respectively to create a connection. By using these two methods, two streams *in* and *out* illustrated by Figure 6.3 are created for transferring data between clients and the server. The `sendMsg` function uses the `outcoming` interface as its single argument which integrates the message information and possible methods to manipulate the message. Likewise, the `receiveMsg` methods is used to receive notification messages

Support for Asynchronous Awareness

NSTP is primarily concerned with synchronous awareness. As discussed before, asynchronous awareness which provides knowledge about past events is, however, of particular importance to Lecture 2000. Some mechanisms must be added into NSTP to support asynchronous collaboration.

Because the notification server has a centralized architecture, client's manipulation of artifacts will cause shared states in the server side to change. Thus, we can let the server record such changes to offer information about past events. While NSTP supports a large set of event (notification) types [1], only several types of events have to be recorded to support asynchronous collaboration in learning groups:

```
CREATE_PLACE          ENTER_PLACE
REMOVE_PLACE          LEAVE_PLACE
CREATE_THINGS         DELETE_THINGS       CHANGE_THINGS
```

Once these events have happened, we extend the server to catch them and save them into a repository which will be discussed soon. We know that each artifact in learning groups is defined as a Place of the server, and Things of the Place describe the features of the corresponding artifact. In this repository, the table *Event* is defined to record asynchronous awareness items:

- Event:
  - `user` who has initiated the event?
  - `place` where the event happened
  - `thing` which Thing has been created, read, changed or deleted?
  - `date` when did the event occur?
  - `event` what is the event type?

The following two functions are called to insert asynchronous events into the table Event:

[1] http://www.lotus.com/home.nsf/welcome/research

```
insertAwarenessItem(user, place, date, event)
insertAwarenessItem(user, place, thing, date, event)
```

The remaining issue is then how to exploit information recorded in the table to provide asynchronous awareness. In section 5.4.2 on page 126, questions about asynchronous awareness have been raised. The following functions can provide answers to these questions by making use of the table Event:

```
getUser(place, eventtype)
getDate(place, eventtype)
getPlace(client, date)
getPlace(date, eventtype)
```

6.2 Persistent States

As we have demonstrated in the preceding section, all artifacts in the system including rooms, workplaces, containers and documents are presented as a Place in the notification server. But NSTP is devoted to synchronous groupware, and asynchronous support is desired and must be added to NSTP, which means we should support persistent states. Once persistence has to be supported, the notification server needs some other infrastructure, e.g. a file system or a database system. Since NSTP has a centralized architecture, it will be relatively easy for us to add a data repository to the notification server. In Lecture 2000, a Informix Universal Server (version 9.2) is used to implement the repository with the management of artifacts and support for persistent states of these artifacts.

6.2.1 Artifact Management

We define the following entities to maintain the persistent states of the artifacts in the server:

- Room:

`ID`	identifier
`name`	room name
`description`	room description
`icon`	graphical representation of the room
`participant_list`	list of users currently working in the room
`synGranu`	value set for implicit session
`creation_date`	creation date
`children`	children Place IDs
`group`	ID of group which inhabits the room

`door_state`	showing the state of the room door
`dates`	information about next meetings

- Workplace:

`ID`	identifier
`name`	workplace name
`description`	workplace description
`icon`	graphical representation of the workplace
`participant_list`	list of users currently working with the artifact
`synGranu`	value set for implicit session
`creation_date`	creation date
`children`	children (i.e. container) Place IDs
`parent`	ID of parent artifact (i.e. room) in which the artifact is located
`owner`	who owns the workplace. If null, public workplace

- Container:

`ID`	identifier
`name`	container name
`description`	container description
`icon`	graphical representation of the container
`participant_list`	list of users currently working with the artifact
`synGranu`	value set for implicit session
`creation_date`	creation date
`children`	children (i.e. document) Place IDs
`parent`	ID of parent artifact (i.e. workplace) in which the container is located

- Document:

`ID`	identifier
`name`	document name
`description`	document description
`icon`	graphical representation of the document
`participant_list`	list of users currently working with the artifact
`synGranu`	value set for implicit session
`creation_date`	creation date
`children`	children (i.e. document) Place IDs
`parent`	ID of parent artifact (i.e. workplace) in which the document is located
`content`	document content

It is noteworthy that the `content` field in the table `Document` may have different forms of values depending on what the document is. If the document represents a memo, the `content` field contains a HTML text describing this memo's content. If the document refers to a learning script, this field will point to a root *Module*, a unit used to manage

learning materials. More discussion about the management of learning materials will come in the next section.

The initiation of a spatial environment for a user who has logged into the system does not depend on other users' current states. Each change of these artifacts will be written back to the repository. Thus, the server can set up the spatial learning environment for a new user with information stored in the system repository. The initiation of an artifact such as a room means that the server creates a related Place with all associated Things for the user by using functions listed in 6.1.2 on page 157. The client side just needs to create corresponding user interfaces according to values in the Place by calling functions such as:

```
getPlaces()
initiateEntrancePlace(entranceplace)
addRoomsIntoRoomPanel(roomplaces)
addWorkPlaces(workplaces)
addContainerPlaces(containerplaces)
```

6.2.2 System Information Management

In addition to artifacts which directly build the spatial learning environment, there are also other information which plays a key role in enabling or facilitating synchronous and asynchronous communications within group rooms.

User Information

CSCW systems are intended to facilitate collaborative work between people. In a CSCW system, information about its collaborative users is very important in enabling such interaction and collaboration because they need to learn something about their partners. User information does not only refer to personal information such as name, occupation, research or working area and hobby etc. It may also include other information such as what one is doing and how long he has worked in a shared workplace. Sometimes, it is not necessary or impossible for users to provide detailed personal information for a collaborative process. In chat rooms, for example, people prefer to use a pseudonym name to participate in chatting. They are often unwilling to let others know who they are. In other circumstances, this kind of personal information may be of particular importance. In an educational brokerage system (see section 3.3), a student may want to learn more about a tutor such as whether the tutor is an academic authority in the researching area before he can make a decision to request learning materials or advises from him. Lecture 2000 is expected to support the whole collaborative learning process from students' locating potential collaborators, forming learning groups to engaging in synchronous and asynchronous communication within the integrated learning environment. Thus the management of user information has become an important part of the learning system.

In Lecture 2000, we distinguish between *static* and *dynamic* user information. Static user information refers to a user's personal information which does not change over time, typically including a user's name, gender, occupation, working area, correspondence information (i.e. telephone number, email and WWW homepage) and so on. In addition, information like to which group the user belongs can be also seen as static information. In contrast, dynamic user information describes the relationship between a user and the learning environment which changes over time, such as on which artifacts the user is working, which sessions he is attending and whether he can be disturbed etc.

Static User Information To manage users' static information, the system maintains two tables defining learning groups and group members respectively as follows:

- User:

`ID`	user identifier
`name`	user name
`password`	password
`group_ID`	ID of learning group to which the user belongs
`tel`	telephone number
`fax`	fax number
`mail`	email address
`www`	WWW homepage
`key_words`	description of learning interests, research areas etc.
`self_intro`	a brief self-introduction to the user

- Group:

`ID`	group identifier
`name`	group name
`member_list`	group members' IDs
`description`	a brief description about the group
`room_ID`	in which room the group resides
`created_date`	when was the group built
`administrator`	who is the group's administrator

As a user logs on, we differentiate group members from community members by checking whether the user has owned a user profile with a valid password in the system's repository. A user will be considered to be a group member if and only if the following two conditions can be satisfied:

1) The user has a valid record in the $user$ table.

2) And there is a record in the $group$ table whose $member\_list$ contains the user's id.

For a user who logs onto the server for the first time, the system will automatically create a user profile in the system's repository. The user can complete or update his own personal information later. Once he has been accepted as a valid member of a group, his user id will be added into the group's $member\_list$. A user's status as a group member will help to determine his *role* in privacy control (see section 5.6.3 on page 142).

A user's static information is primarily used to filter and locate potential collaborative partners in the dynamic learning environment. For example, one can get a brief introduction to a user by clicking the user's name in the user list in the entrance hall (see section 5.1.1 on page 100). This information helps community members to get the first impression about others and to find appropriate partners to chat together or to set up their own learning groups in the server.

Dynamic User Information The dynamic information of a user records the user's activities with respect to the learning environment, including what the user is currently doing and what the user has done in the past. In fact, it is an inseparable part of awareness information discussed before. The purpose to discuss users' dynamic information separately is that it is often desired to provide an integrated approach to more detailed user information. In Lecture 2000, answers to the following questions constitute a user's dynamic information:

1. current status of a user
    a. In which room the user is currently working?
    b. Which sessions the user is currently attending?
    c. On which artifact the user is currently working?
    d. Is the user now accessible?
    e. What is the user's current role (i.e. owner, collaborator or visitor)?
2. past status of a user:
    a. When did the user log on to the server for the first time?
    b. How long the user has logged on to the server?
    c. How often has the user logged on to the server in the past?
    d. What is the history of his group member's status?
    e. What sessions has the user attended in the past?

According to this analysis, most of dynamic user information can be found in other tables. For example, the table Session, which will be discussed soon, records sessions in which a user has participated in the past. What we need is to set up a table recording the logging information of users:

- Logging:

`ID`	user identifier
`logging_date`	when the user logged in
`exit_date`	when the user terminated his session
`session_list`	sessions which the user has attended during this logging

A user's dynamic information can contribute to the facilitation of interaction and collaboration in the learning environment in various ways. Here are just several typical examples:

- First of all, it is sometimes important to learn whether a user has often logged on to the server in the past. A positive answer suggests at least that he is an active user of the system. One can then, for example, contact him to attend his group's discussion. Otherwise, it shows that the user perhaps does not like to work with the system although we can find his user profile from the system's repository.
- Secondly, it is also valuable for us to locate collaborative partners according to some specific needs. For example, one may need to look for a partner who has attended a discussion about the topic of "Security Problems in E-Commerce". Though we may find this kind of information in a user's static information, the dynamic user information can better reflect the user's current interests about this topic.
- Finally, as the spatial environment grows, there will exist more and more rooms and sessions. Then it may be difficult for users to locate a targeted person quickly. Aforementioned issues like 1a, 1b and 1c helps a user to overcome this kind of difficulties.

Session Information

In the preceding chapter, we have discussed a lot of issues regarding sessions. To support session management, we need to define the table $Session$ in the system's repository as follows:

- Session:

`ID`	session identifier
`name`	a user-defined session name
`description`	a short description about the session
`initiator`	initiator of the session if possible
`participant_list`	a list of all session participants
`starting_date`	starting date and time
`ending_date`	ending date and time

Of course, the $ending\_date$ is not defined while a session is still ongoing.

Based on this table, the following functions are provided to allow users to create new sessions, to browse existing sessions, and to check a specific session and so on:

```
createSession()                editSession(sessionid)
getSessions()                  deleteSession(sessionid)
getSession(userid)             getSession(username)
getSession(startingdate)       getSession(endingdate)
updateSession(name, initiator, description)
addParticipant(sessionid, userid)
```

The `createSession` function can be called to create either an implicit or explicit session record in the system. After creating a new session, one can call other functions to update the session's attributes such as its name, and its description. The `addParticipant` function is automatically called by the system to record who has attended the session before the session is closed. The `starting_date` and `ending_date` of a session is also automatically maintained by the system. The call to the function `editSession` will initiate a user interface allowing an authorized user to edit attributes of a session. A set of `getSession` functions allow to view existing sessions according to definite parameters.

6.3 Learning Material Management

Highly-qualified learning materials are the basis for most learning and teaching activities. They are not only needed by students in their private study, in learning groups, but also by tutors in lectures and seminars. It is either impractical to require tutors to prepare their learning scripts and tasks within our integrated learning environment. Instead, tutors are more willing to prepare them by using various authoring systems. In this section, we firstly make a thorough investigation about the creation, distribution and use of learning materials. Then we will discuss how students make use of learning materials within learning groups.

6.3.1 Scenarios of Using Learning Materials

According to Schlichter [Schlic97a], learning materials may include lecture scripts, seminar reports, tasks, and their solutions. In addition, administration information about courses, seminars can be seen as the part of learning materials. In some educational systems, domain specific dictionaries and glossaries are also made available to students.

Briefly, learning material progresses through several phases as follows:

Preparation Electronic learning materials such as lecture scripts and tasks are normally prepared by tutors;

Distribution File systems, database systems or Web-based Information Systems are used to manage electronic learning materials, which can be distributed through the Internet, viewed by using a Web browser, or printed out by students at home;

Utilization Both students and students can make use of these electronic materials in their different learning and teaching phases:

- Tutors use lecture scripts in lectures and seminars directly from the Web;
- Students prepare their lessons before classes;
- Students review their lessons after classes;
- Students complete tasks;
- Students prepare their exams;
- Both tutors and students use learning materials to support their discussions in learning groups.

In the preceding chapters, some of these issues such as how to use Web-based Information Systems to manage and distribute electronic data have been extensively addressed. Subsequently, we primarily focus on the preparation and utilization of learning materials, especially the lecture scripts.

6.3.2 Markup Languages for Lecture Scripts

XML-Enabling Technologies

It is well-known that the Web, the Hypertext Markup Language (HTML) and the Hypertext Transport Protocol (HTTP) have revolutionized the manner in which information is distributed, displayed and searched for. So far, most documents distributed across the Internet are HTML documents which are defined by almost the same set of tags. Indeed, there is a significant benefit to a fixed tag set with fixed semantics, namely *portability*. A Web page using the standard tags can be viewed by any browser and anywhere in the world. However, more and more people are recognizing the limitation of the fixed tag set with fixed semantics in HTML documents. Web designers want more control over the description and presentation of their documents. Many processes would benefit from more descriptive tagging.

Extensible Markup Language (XML) provides a significant advance in how data is described and exchanged by Web-based applications. In essential, HTML enables universal methods for *viewing* data while XML provides universal methods for *working* directly with data. As a subnet of Standard Generalized Markup Language (SGML), XML is optimized for delivery over the Web. Basically, XML allows developers to define an unlimited set of tags, providing authors of the documents with great flexibility. In other words, we can use any tags that we want in XML documents. We can write documents using our own tag names that are meaningful in the context of our subject matter and offer the possibility of far greater control over presentation. In contrast, HTML defines tags that describe how the data should be displayed, such as a bulleted list or a table.

While XML technologies will change the way that information is distributed and processed, the following aspects are of particular importance to the preparation, distribution and utilization of learning materials in Lecture 2000:

- More descriptive tagging

XML allows users to define their own tags to structure their documents. With XML, learning material module in Lecture 2000, for example, could be easily defined in a standard way by the following tags [2]:

`<module> </module>`	a unit of learning materials
`<issue> </issue>`	a single thematic entity
`<header> </header>`	header of the issue
`<kernel> </kernel>`	main statement about the issue
`<intro> </intro>`	introduction
`<details> </details>`	content of the issue
`<summary> </summary>`	summary
`<definition> </definition>`	definition of a term
`<note> </note>`	an annotation to a term

An example of using these tags to define a module in CSCW looks as follows:

```
<module>
   <issue>
      <header>Application-Level Taxonomy of CSCW Systems</header>
      <kernel>In this section, we discuss the classification of CSCW
         systems according to their application areas.
      </kernel>
      <details>
         <issue>
            <header>Communication System</header>
            <kernel>
                  ... ...
            </kernel>
            <details>
                  ... ...
            </details>
         </issue>
         <issue>
            <header>Shared Information Space</header>
            <kernel>
                  ... ...
            </kernel>
            <details>
                  ... ...
            </details>
         </issue>
           ... ...
      </details>
```

[2] http://www11.informatik.tu-muenchen.de/proj/targeteam

```
   </issue>
</module>
```

- More meaningful searching

  These descriptive tags enable more meaningful searches. Without XML, it is necessary for the search application to understand the schema of each database, which describes how it is built. Further, the search application can mostly just search though a document according to key words. With XML, applications can refine their searches and search through a set of documents in a consistent way. Based on the above defined tags, applications in Lecture 2000 can search through specific sections of a document such as introduction or summary.

  Moreover, we can specify a set of attributes for these tags. These attributes enable users to further optimize their searches. For example, the following code sets up attributes *title*, *author* and *date* for a module:

```
<attrs laststamp="934183739832">
   <attr name="title">
      <version stamp="934183739832">What is CSCW</version>
   </attr>
   <attr name="author">
      <version stamp="934183739835">Schlichter</version>
   </attr> <attr name="date">
      <version stamp="934183739832">11.11.1999</version>
   </attr>
</attrs>
```

- More flexible application development

  More importantly, XML-based documents can be easily exchanged and processed between different applications because they are self-describing. It enables users to process XML documents according to their own needs. For example, XML-based lecture scripts can be converted into different file formats by students and tutors in their different learning and teaching phases (see figure 6.4). In fact, these generated files are still XML documents. The generated HTML documents from XML can be shown in a Web browser and projected to a liveboard in a classroom. Students can print out the PostScript, PDF or Rich Text files of the scripts for their preparation or review of the lessons.

XML/XSL-based Document Processing

XML technologies involve a set of related standards developed by the World Wide Web Consortium (W3C) [3], an international industry consortium which leads the Web to its full

[3] http://www.w3c.org

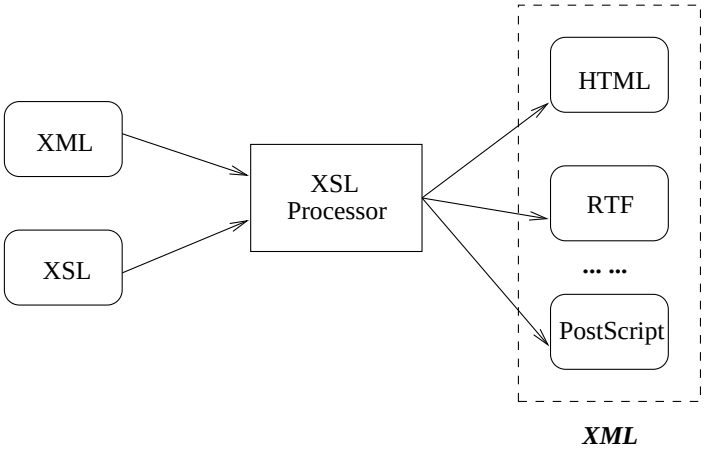


Figure 6.4: XML-based data can be converted into different data files

potential by developing common protocols. The XML initiative consists of the following related standards:

XML (Extensible Markup Language) : The current W3C Recommendations are XML 1.0.

XSL (Extensible Stylesheet Language) : XSL is a language for expressing stylesheets. It consists of two parts:

- XSLT (XSL Transformations) for transforming XML documents;
- an XML vocabulary for specifying formatting semantics.

As illustrated by Figure 6.4, the Extensible Style Language (XSL) is used as style sheet for describing how XML documents have to be presented because XML tag names have no predefined semantics. A style sheet contains instructions that tell a processor how to translate the logical structure of a source document into a presentational structure.

XLL (Extensible Linking Language) : XLL is an XML linking language that provides links in XML.

We use this section to briefly introduce how to use the stylesheet language for XML, especially the possible scenarios of applying XSL for Web-Based Information Systems. Just as other stylesheet languages such as CSS and DSSSL [4], an XML stylesheet specifies the presentation of a class of XML documents by describing how an instance of the class is

[4]CSS (Cascading Style Sheets) is a stylesheet from the W3C and has been designed for use with HTML. DSSSL (Document Style Semantics and Specification Language (DSSSL) is a standard from the ISO, which is more complex than CSS but also more powerful. When fully implemented, it can support both printing and online publishing from any SGML data.

transformed into an XML document that uses the formatting vocabulary. All XML documents can be presented as trees. An XSL processor begins at the root node in the source tree and processes it by finding the template in the style sheet that describes how that element should be displayed. Each node is then processed in turn until there are no more nodes left to be processed. The product of this processing is a result tree. If the result tree is composed of XSL formatting objects, then it describes how to present the source document.

For Web-based applications, XSL can be applied either at the server side or at the client side (see Figure 6.5 and 6.6). If an XSL processor resides on the server side to output HTML, one can make use of the flexibility and power of XML without having to worry about whether a particular client provides XSL support or not.

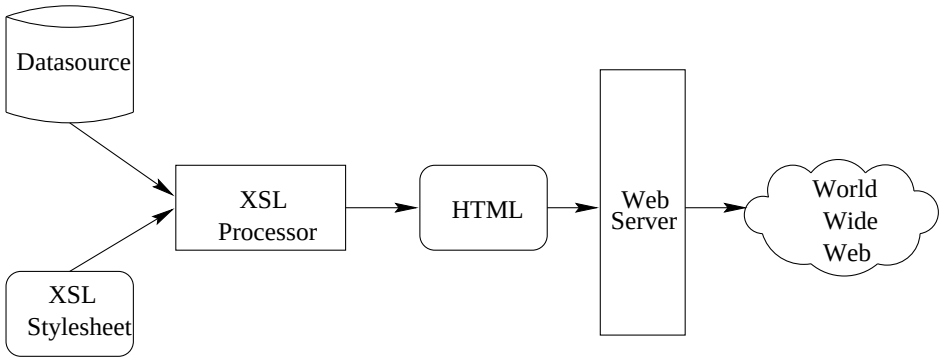


Figure 6.5: Server side-driven XSL

In order to take full advantage of XML and XSL, a scenario where the rendering happens on the client side, however, is the most appropriate one. In this case, the client can use different stylesheets based on user preferences. Another benefit is that a user is able to use a stylesheet that the server does not know about. This allows full customization of rendering based on a user's needs.

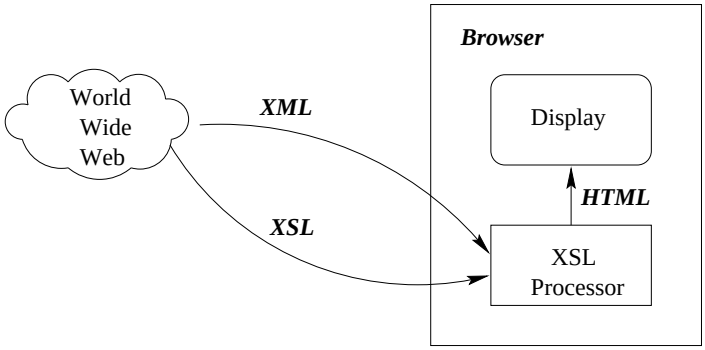


Figure 6.6: Client side-driven XSL

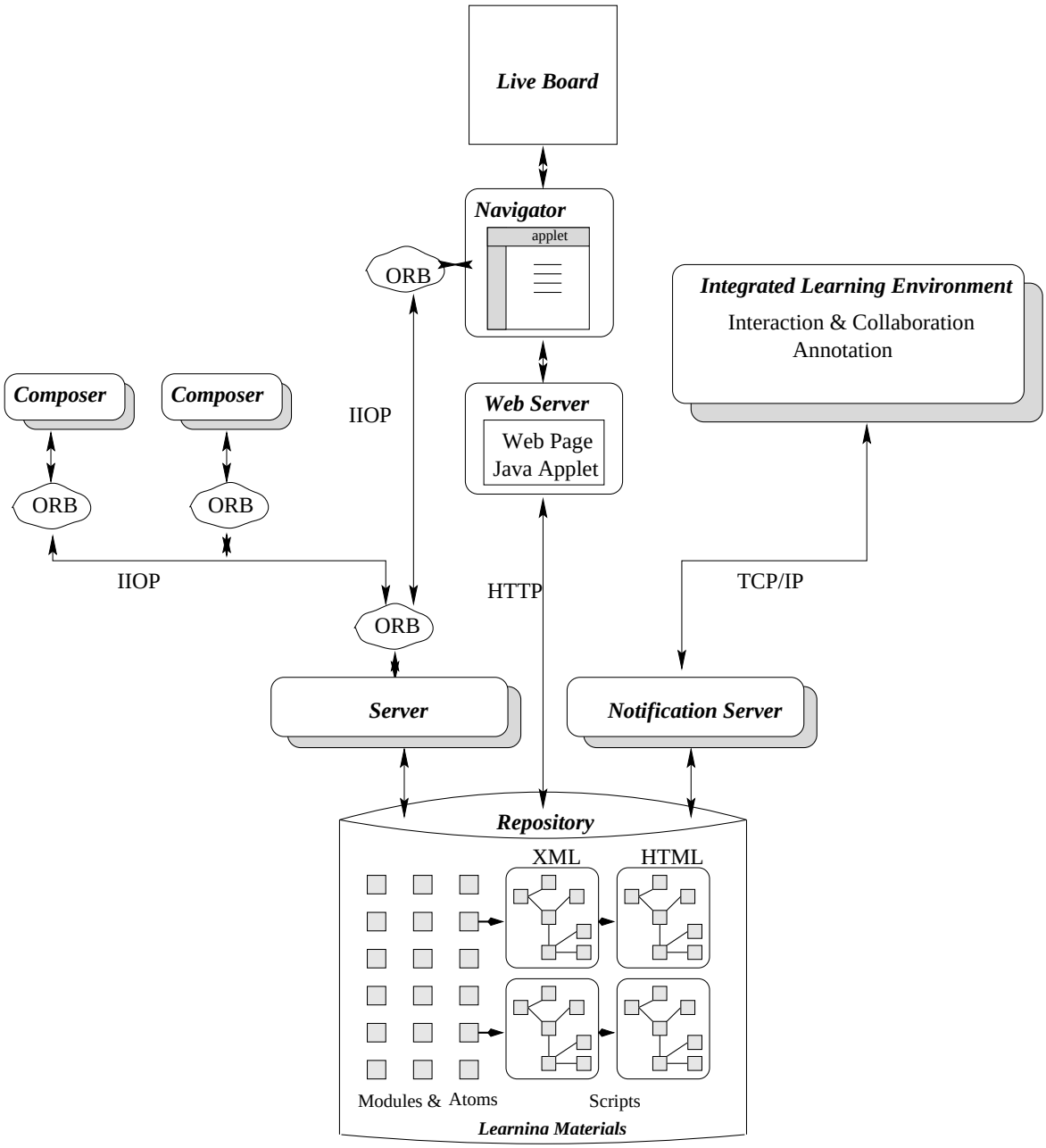


Figure 6.7: Application Scenarios using learning materials

6.3.3 Management of XML-Based Lecture Scripts

Lecture scripts in Lecture 2000 are XML-based documents stored in the same repository implementing the persistent states of artifacts discussed in the beginning of this chapter. In terms of the management of learning materials, the repository primarily consists of three types of elements:

- *modules*: A module is a reusable XML document which is defined by a DTD language (TeachML) specified in the chair.
- *atoms*: An atom are used to describe contents which can not be represented in XML such as multimedia objects.
- *course*: A lecture script is represented by a course element which is comprised of modules and atoms selected by a user according to his own preferences and needs.

The process of preparing and utilizing the XML-based lecture scripts involve the following activities (see Figure 6.7).

- Composing:

  The Composer tool enables tutors to create modules for learning materials. In the Composer, authorized users can add, update and delete modules into the systems' repository. The key function of the Composer is to enable users to select modules they need and to integrate them into a lecture script. In the future, students in learning groups should be also enabled to compose their own lecture scripts according to their own preferences and learning advances.

- Navigating:

  The lecture scripts stored in the repository can be used by tutors in classes. The integrated lecture scripts are still XML document. In order to directly use these scripts from the Web in classes, we can either use a Web browser which supports XML technologies or we translate them into HTML documents which can be viewed by using the Web-based tool *Navigator*. The Navigator is comprised of two Java Applets. One mainly lists siblings of the current node; the other shows the content list of the lecture script.

- Annotating:

  The learning materials created by tutors build the basis for interaction and collaboration in the integrated learning environment. In this environment, students can conduct their private study by using these learning materials. Likewise, they are enabled to annotate or comment these lecture scripts during their study.

As illustrated by Figure 6.7, the composer and navigator share the same repository as our integrated learning environment. XML-based lecture scripts composed from modules and

atoms stored in the repository can be further translated into HTML documents. Figure 6.7 also shows that the composer and navigator are connected to the repository through the protocol IIOP. It is then necessary for us to briefly see how distributed object technologies are used by Lecture 2000 to implement the connection between clients and servers.

Distributed Object Computing

As addressed in section 5.3.1, a distributed object application is often comprised of two separate programs: a server and a client. A typical server application creates some remote objects, makes references to them accessible, and waits for clients to invoke methods on these remote objects. A typical client application gets a remote reference to one or more remote objects in the server and then invokes methods on them.

Because current object-oriented languages such as C++ and Java do not have built-in language support for remote messaging, a separate facility must be used to allow programs to communicate with each other across the network. The Common Object Request Broker Architecture (CORBA) [5], for example, allows applications to communicate with one another no matter where they are located or who has designed them. It is a widely supported standard that allows objects to advertise their interfaces using Interface Definition Language (IDL) and communicate across networks using a language-neutral protocol called Internet Inter-Orb Protocol (IIOP). For example, CORBA allows a C++ clients running on Window NT to communicate with a Java object located on a Unix server.

In addition to CORBA, there are other widely adopted standards to support communications among remote objects. Table 6.2 lists these standards.

Standard	**Protocol**	**Interface Definition Language**	**Application Language**
CORBA (Common Object Broker Architecture)	IIOP (Internet Inter-Orb Protocol)	IDL (Interface Definition Language)	language neutral
DCOM (Distributed COM)	ORPC (Object Remote Procedure Call)	MIDL (Microsoft Interface Definition Language)	language neutral
RMI (Remote Method Invocation)	JRMP (Java Remote Method Protocol)	Java	Java

Table 6.2: Three standards for Distributed Object Computing

While applications using these different standards can not communicate with each other,

[5] http://www.omg.org

Voyager [6] is a state-of-the-art platform for distributed computing that includes a standard-neutral ORB. In other words, a Voyager program can simultaneously send messages to other CORBA, RMI and COM programs, and an object in Voyager program can simultaneously receive messages from other CORBA, RMI and COM programs.

In distributed object technologies, some kinds of stubs and skeletons usually must be generated for references to remote objects before a distributed object application can run. For example, a programmer must use the rmic compiler to create stubs for remote objects if he uses RMI in Java. RMI uses a remote object's stub class as a proxy in clients so that clients can communicate with a particular remote object. Without generating stubs and skeletons, Voyager supports the communication among remote objects very simply, which can be illustrated by its use in Lecture 2000, where a `Pool` is defined to manage elements of lecture scripts such as courses, modules and atoms discussed before. A `PoolManager` is responsible for all accesses to the pool. Of course, both `Pool` and `PoolManager` reside on the server side. A Voyager server is then initiated by the following codes:

```
// start up on port 8000
Voyager.startup("8000");
// define a Pool
Pool pool = new AbstractPool();
// define a PoolManager
PoolManager pm = new PoolManagerImpl(pool);
// bind into local naming service
Namespace.bind("PoolManager",pm);
System.out.println( "server ready" );
```

The client can simply get reference to the remote object `PoolManager` represented by `pm` in the server as follows:

```
// start up on random port
Voyager.startup();
PoolManager pm = null;
// look up remote object
pm = (PoolManager)Namespace.lookup("//host:8000/PoolManager");
```

Once a client gets a reference to its targeted remote object, it can use methods defined by the remote object to complete desired operations. In Lecture 2000, the needed methods to manipulate the `Pool` are defined as interfaces in the `PoolManager`, which are shown in Table 6.3.

Interface	Class	Function
`Pool`	`AbstractPool`	external persistent storage for a collection of learning materials.
`PoolManager`	`PoolManagerImpl`	management of the pool

`PMNode`	`PMModule`
	`PMAtom`
	`PMAtomAlt`
	`PMCourse`
		elements in the pool

`PMCourse`	`Course`	a course
`PMModule`	`Module`	a module
`PMAtom`	`Atom`	an atom

Table 6.3: Interfaces and their classes defined in the PoolManager

Course Composer

As illustrated by Figure 6.8, the composer consists of two parts:

- On the right is a table which contains all available modules, atoms and courses in the repository. All of these elements constitute the so-called `Pool` mentioned above;
- The tree on the left side is comprised of all available courses in the repository. Each course represents a lecture script. The subtree of a course node consists of all modules and atoms the course has used. And the structure of the subtree corresponds that of a lecture script in terms of chapters, sections, and paragraphs.

Thus the task to compose a lecture script is to select appropriate modules and atoms from the pool and then to add them into the subtree of a course.

Pool Management First of all, all modules and atoms in the repository are reusable, which means that they can be used by more than one courses. The task of the pool management is to import, export, and delete a module or an atom into or from the pool, as well as maintain attributes associated with these elements. For users' convenience, these functions are realized in a popup menu which can brought up by either clicking an item in the table or clicking a node in the tree. The popup menu includes the following items:

- *describe*: describe the element, providing information such as its children names and associated attributes. Further the element's title, author and created date are also included in the description.
- *import*: import the element from a file.
- *export*: export the element to a file.

[6]http://www.objectspace.com

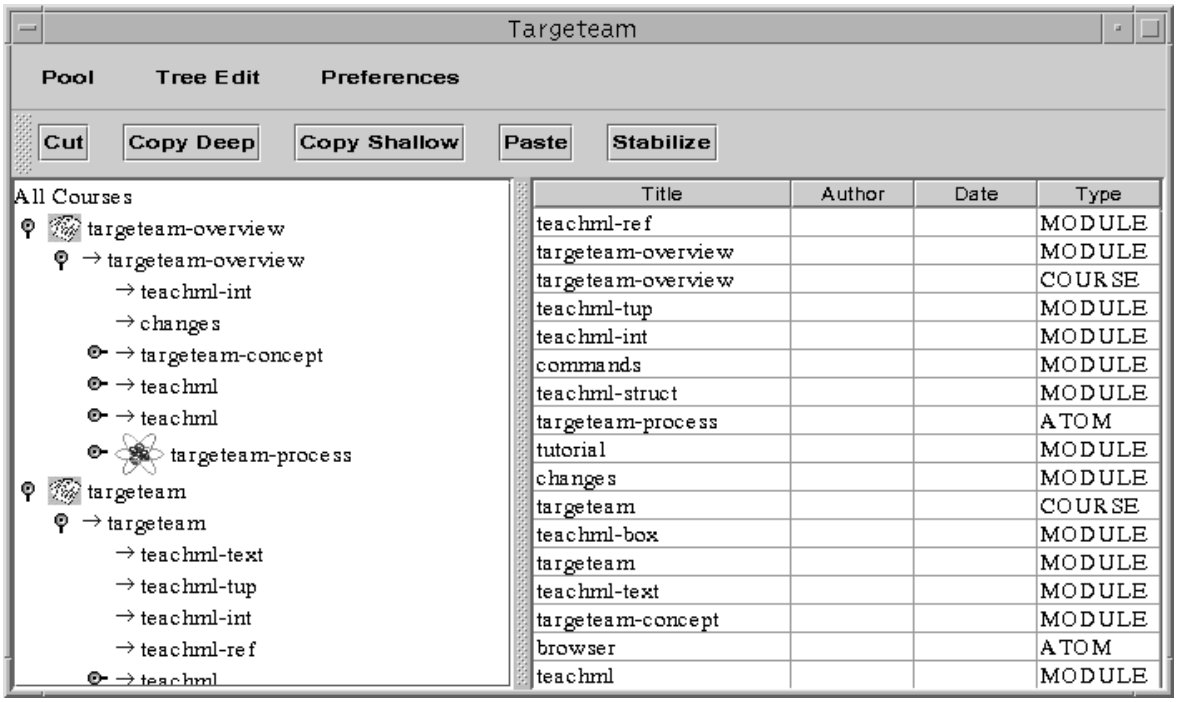


Figure 6.8: Composer

- *delete*: delete the element from the pool.
- *prepare*: prepare the content of a course element by integrating the course's root module and export the resulted XML document to the current directory.
- *attribute*: update attributes of the element. It can also list all available versions of an attribute. Detailed description about version management will be discussed in the context of version management.
- *create atom format*: create a new alternative format of an atom and import its initial content. Normally, an atom (e.g. an image) can be associated with several formats (e.g. gif, jpeg or ps).

In addition, the system allows users to define new elements directly by specifying their attributes such as title, and author. Later, these new elements' content can be imported from external files.

Composing The composing process is supported by the following buttons:

- *cut*: cut a module or an atom from the tree into the system clipboard ;
- *delete*: delete a module or an atom from the tree;
- *copy shallow*: just copy a module without its children;

- *copy deep*: copy a module with its all children;
- *paste*: paste a module or an atom from the system clipboard.
- *stabilize*: makes the current version of a node (i.e. course, module or atom) with its content, its attributes and its child structure persistent in the pool.

Moreover, the composer provides a powerful implementation of *drag and drop*. A user can drag an element from the table and the tree in the same composer or from other composer windows. Note that users can open more than one composer windows, each of which represents a version of the pool.

Integration Once a course has been composed, i.e. one has selected needed modules and atoms from the pool and inserted them into the tree as the course node's descendants, an integration process is required to generate the XML content of the course. That is a recursive process beginning with the root node of a course's subtree.

As Figure 6.4 shows, a XSL processor needs a XML and a XSL documents as its input and then the processor outputs a XML document. In the subtree of a course, each module's content is a XSL document which serves as the input of stylesheet. And all children's XML results of the module are merged to serve as the XML input of the XSL processor. For the XSL-processor to work also with leaf modules, we assume that it has an empty XML document. As Figure 6.9 shows, the integration process works as follows:

- The integration process begins at the root node of a course and processes its descendants recursively.
- For a module which is a leaf node in the tree, we assume that it has an empty XML document as its child with only an element *<reused-materials></reused-materials>*. The processor takes the module's XSL content as stylesheet and the empty XML as its inputs.
- For a non-leaf module node, we just need to merge all its children modules and children atoms together into a XML document wrapped with *<reused-materials>* and *</reused-materials>*. Such a merged result serves as the XML input of the XSL processor.

Finally, let us briefly see how a module specifies its children. For example, the *xsl:for-each* element contains a template which is instantiated for each node selected by a pattern. The pattern is specified by the *select* attribute. The following code repeatedly processes all *issue* children of a module which has an attribute *id* with the value "cscw-concepts":

```
<xsl:for-each select="tml:module[@id='cscw-concepts']/tml:issue">
    &tmlinc;
</xsl:for-each>
```

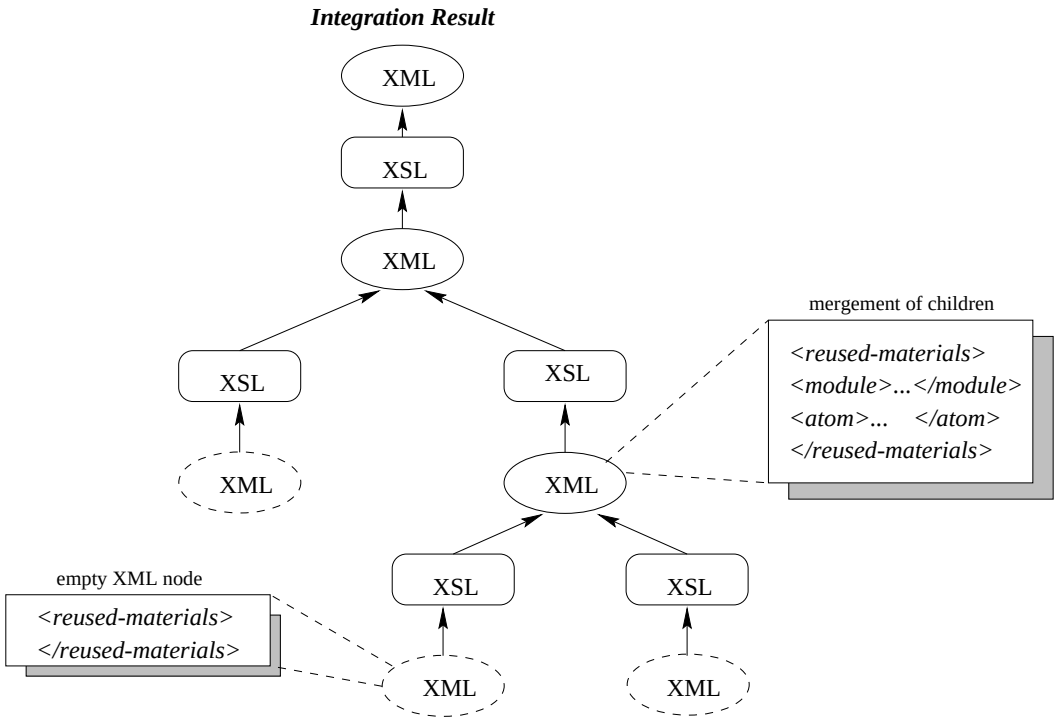


Figure 6.9: Integration process of a lecture script

This code demonstrates an important advantage of XML/XSL over HTML. Traditionally, a HTML document has to include its children as links specified through the <A> tag. In XML, one can dynamically specify children for a document in run-time. Moreover, one can change the order of elements, process elements more than once and so on.

Navigator

We translated the XML lecture scripts into HTML so that tutors can use them directly from the Web in their lectures. A navigator is then needed. The navigator shown in Figure 6.10 consists of three components. On the left is a Java Applet showing the subject list of the script. Another Applet shows siblings of the current node in the script. Finally, the current node's content is shown in HTML in the middle of the window.

The Applets work on a set of tables stored in the system repository. We know that the integration process discussed above outputs a single XML document which is comprised of all modules and atoms of a lecture script. To be viewed by any Web browser, the XML document must be divided into HTML documents in terms of chapters, sections and paragraphs. Such a transformation from a whole single document into HTML units can be easily implemented, because each XML document can be represented as a tree. Nodes in different levels of the tree can be classified into chapters, sections and paragraphs respectively.

Three tables are to be defined for the navigator to work:

- ScriptNode

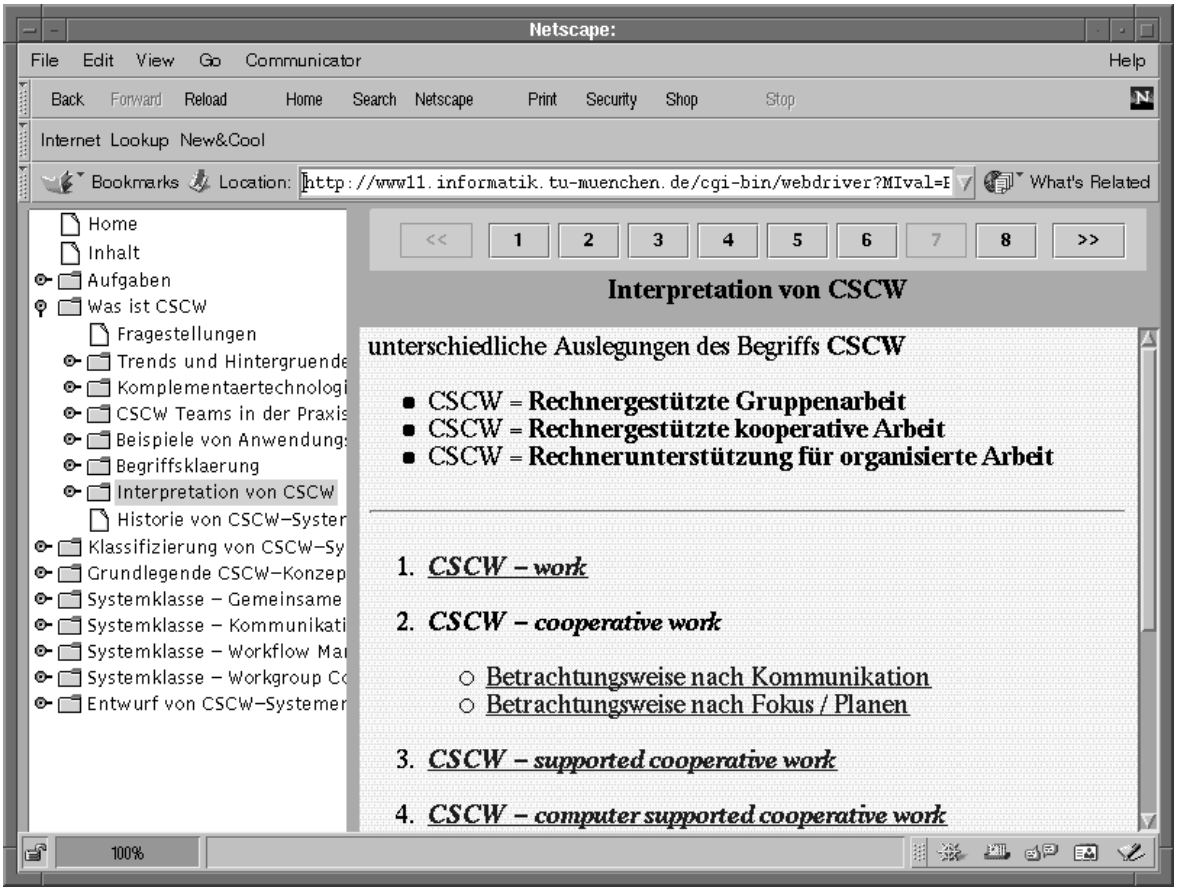


Figure 6.10: Course Navigator

`script_id`	script identifier
`node_id`	node identifier
`title`	title
`parent_id`	identifier of its parent
`index_pos`	position index with respect to siblings
`page_id`	page identifier

- Page

`page_id`	page identifier
`content`	HTML content of the page

- Graphic

`graphic_id`	graphic identifier
`description`	a short description of the object
`height`	height
`width`	width
`content`	a blob storing its content

The table $ScriptNode$ records structural information of a HTML page, such as the current node's title and its parent node. And the position of the current node with respect to its siblings is defined by *index_pos*. The HTML content of the HTML page, which is recorded in the table $Page$, is referenced by *page_id*. Atoms in the XML document are transferred into records in the table $Graphic$.

Annotating of Lecture Scripts in Learning Groups

There are a lots of discussions about the annotation of hypermedia documents such as those in Quilt [Fish88]. In Quilt, a document consists of a base and nodes linked to the base using a locally developed hypertext technology. These nodes act like the scraps of papers, notes and marginalia in traditional paper manuscripts. In this way, proofreading notations, comments etc. can all be incorporated as hypertext nodes.

With the advent of XML/XSL technologies, efforts have been increasingly made to define a standard for annotating hypermedia documents. Many argue that today's Web is a commercial one-to-many medium, namely authors actively publish information and browsers passively view it. It is desired to augment the Web with the basic capability for third party viewers to annotate content. In this way, it is possible to make the Web a mainstream many-to-many medium. However, despite several advances in this field, there is nowadays no widespread annotation service. In Lecture 2000, a simple approach to annotating lecture scripts is adopted. Group members in the integrated learning environment initiate a similar navigator as the one just discussed above to view lecture scripts. The difference is that each page in the navigator is embedded with a button `Annotate`. The process to annotate a HTML page then looks as follows:

- Group members use lecture scripts composed by tutors. As we have said, only tutors can use the composer to write their lecture scripts at this time. In the future, group members should be enabled to compose lecture scripts according to their own needs and learning advances.

- From his group room, a group member can call the navigator embedded with an `Annotate` button to view lecture scripts. The URL transferred to the browser contains information such as user's identifier, page identifier, and page title etc. Since the current repository is based on an Informix server, such a URL transfered to the navigator has the form "*http://host-name/cgi-bin/webdriver?userID=u&pageID=p...*".

- Clicking the button `Annotate` appearing on the bottom of each HTML page will bring up a new Web page which contains a form showing this page's annotation made by this group member. If no annotation has been made by the user, an empty form will appear.

- This group member can then enter or edit his annotations to the page at will. As the member submits the form, annotations in the form will be written back to the repository.

To support the annotation activity, a table `Annotation` is defined:

- Annotation

`user_id`	user identifier
`script_id`	identifier of lecture script
`page_id`	identifier of HTML page
`page_title`	title of HTML page
`annotation_content`	annotation made by the user to the page
`last_change`	last time when the annotation happened

Obviously, this table maintains all annotations, recording who has made what an annotation to which page. With the help of Informix webdriver, it is relatively easy to load relevant annotations made by a user into the Web browser and to write updated annotations back to the table. Of course, group members can directly make use of their annotations within their group room without the need to initiate a Web browser (the navigator). With annotations available in the table Annotation, one can easily search, view, and update his annotations in a Java application.

6.3.4 History Management

In CSCW systems, the term *history* can have different meanings in different contexts. For example, history in BSCW provides information about what was changed, when and by whom. It is primarily concerned with users' access activities to objects. The provision of such asynchronous awareness information in Lecture 2000 has been discussed in this thesis.

As we have shown, learning materials in Lecture 2000 are hypermedia documents. As a critical issue in hypermedia research, version management is primarily concerned with the process of organizing, coordinating, and managing the development of evolving multimedia objects. According to Hicks [Hicks98], this process involves:

- maintaining version history that supports access to previous revisions of hypermedia objects, and
- automatically tracking the derivation information of those objects.

Borghoff *et al* point out that the version management of documents is based on two principles [Borgho98]:

- Implicit version management: if a part of a document has changed, a new version of this part will be generated.

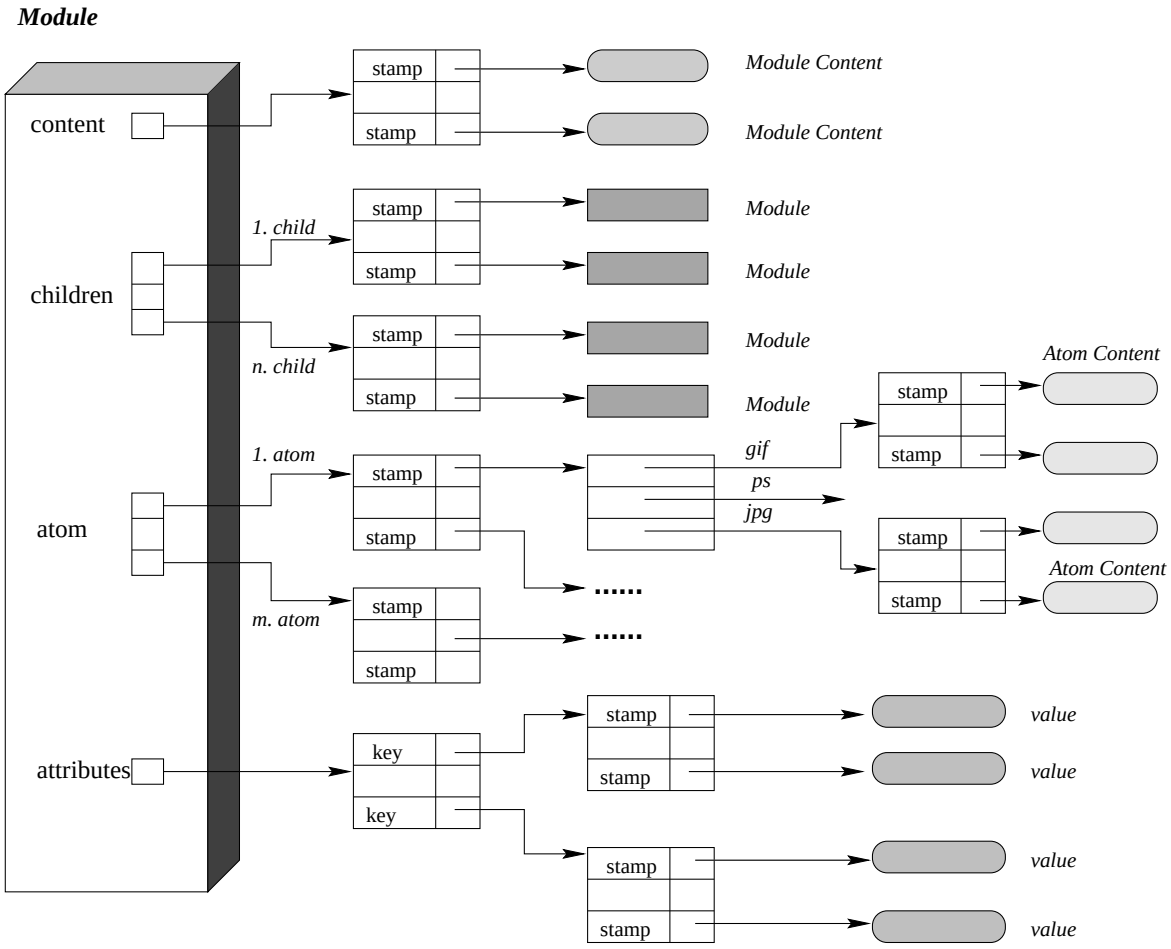


Figure 6.11: Version management of learning materials

- Hierarchical version management: if the version of a subordinate part of a document has changed, the version of the superior part will be updated as well.

The first principle is applied to the version management of modules, atoms, and courses [7]. Basically, a module is a reusable information unit and consists of the following major components:

- a versioned reference to the module content specified in XSL;
- an extensible list of versioned references to child modules;
- an extensible list of versioned references to child atoms;
- an attribute list with versioned attribute values.

[7] http://www11.informatik.tu-muenchen.de/proj/targeteam/

An atom is a MIME object and encompasses a content in several alternative formats, each of which is version-managed as well. Figure 6.11 shows a simplified view of version management of reusable modules. When a module's content, for example, is changed and the change is "stabilized", a new version of the content will be generated. The specific time when the change occurred is considered as the key (i.e. time stamp) used to access this version later. Likewise, its child modules, child atoms and attributes are also versioned. Because modules, atoms, module contents, atom contents and module attributes are all versioned, they are implemented as instances of versioned units which server as a super class of courses, modules and atoms. The following methods are offered to manage the version management of these units:

```
getModule(stamp)              getModule()
getModuleContent(stamp)       getModuleContent()
getAtom(stamp)                getAtom()
getAtomContent(stamp)         getAtomContent()
getAttrValue(name, stamp)     getAttrValue(name)
```

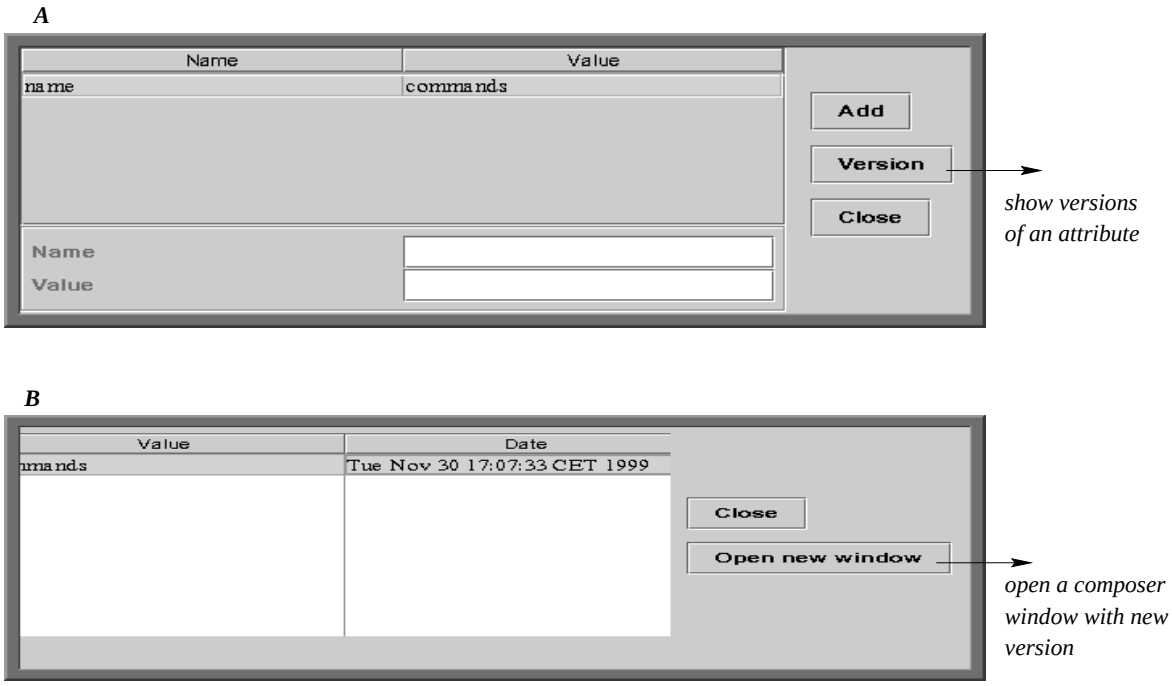


Figure 6.12: Version management in Composer

These methods can be classified into two groups. Methods with a time stamp get units from the repository which are valid at the specific time while methods with no parameters take the current time as their default time stamp and get units which are valid at the current time.

Finally, a course object also encompasses version information:

- a module which serves as the root of the course;

- a XML content of the last integration;
- a stamp denoting when the last integration happened.

The integration process is initiated by calling the function `Integrator` with the argument `stamp` used to select right element versions from the pool:

```
Integrator(rootModule, stamp)
```

Since all elements in the pool are versioned, it is then sensible to have several composer windows shown in the user interface at the same time, each of which represents a version of elements at a specific time. This can be implemented in two steps. Firstly, one selects an element from the pool and brings up a popup menu. Clicking the `Attribute` item in the popup menu will bring up a window which lists all attributes of the selected element.

As Figure 6.12 shows, one can then select an attribute from the list and click the `Version` button in the window. All available versions of the selected attribute will be shown. Select a valid version and click the `Open new window` button. A new composer window which contains all elements with the selected version will appear. As we have said above, one can use elements from any composer window to compose his lecture scripts.

7 Summary and Outlook

This chapter will summarize the contributions of this thesis and then present a number of open questions for further research.

7.1 Summary and Result

7.1.1 Summary

In the past few years, the dramatical technological advances, in particular the Internet and the Web, have changed the way that knowledge is conveyed between students and tutors. Distance learning systems are supposed to facilitate this knowledge transfer by harnessing potentials of these new technologies. However, there is no integrated learning environment available up to now which can support various teaching and learning scenarios of students and tutors in a consistent way. This thesis is intended to develop an integrated learning environment supporting both interaction and collaboration among students as well as between students and tutors who are distributed across time and space.

In light of such a goal, this thesis has extensively explored the following two issues:

- Distance Learning

  Though a wide variety of distance learning systems have been developed, they only support limited learning and teaching activities. Often, students and tutors must move between different tools in their different learning and teaching phases. In particular, most existing distance learning systems only emphasize the importance of information sharing and have not offered effective approaches to interaction and collaboration among students as well as between students and tutors.

- Communityware and Groupware

  Communityware and groupware are expected to support interaction and collaboration between people. But studies have shown that they have developed separately in the past. While communityware is primarily concerned with creating, maintaining, and evolving social interaction in communities, groupware system is aimed at supporting groups of people engaged in a common task and pursuing a common goal.

The task of the thesis is then to harness potentials of both communityware and groupware to design and develop an integrated learning system. The research method we have adopted in this thesis is to have each chapter focus on a few key issues from CSCW and then apply these discussion results to develop the targeted system model. First of all, a pyramid model consisting of lectures, discussion groups and learning groups in educational settings was defined by investigating issues remaining in existing CSCW and distance learning systems. To support community interaction and groupwork seamlessly, we have extensively examined spatial approaches to communities and groups. Such a discussion led us to a spatial framework comprised of an entrance hall, a discussion group, a tutor studio and several learning groups. Then an artifact-based model used to design group rooms was discussed, and a set of key components facilitating both interaction and collaboration between people within the spatial world were presented.

The notification server plays a key role in implementing the artifact-based learning environment. Based on NSTP, a notification server model supporting both synchronous and asynchronous communications was realized. In addition, the management of learning materials is the basis of all learning activities. Instead of building learning materials in the traditional HTML, we illustrated the huge potentials of XML/XSL in creating, distributing and utilizing electronic learning materials. And various scenarios of using learning material were presented.

7.1.2 Result

The result of this thesis is an integrated learning environment utilizing the power of both communityware and groupware. In particular, it explored the use of a wide variety of spatial metaphors and artifacts. In fact, the spatial framework just provides a collaborative environment while the use of various artifacts make possible both synchronous and asynchronous communications. In conclusion, the following results have been achieved:

- a detailed investigation of communityware, groupware and distance learning systems;
- a pyramid model depicting collaborative entities in organizations;
- a spatial framework supporting community interaction and groupwork in educational settings;
- an artifact-based model enabling both synchronous and asynchronous communications within learning groups;
- a set of key components implementing functions ranging from artifact management, lightweight connection, to privacy control and history management.

In the beginning of this thesis, we have emphasized that the phases of a collaborative learning process are separated in most existing distance learning systems (see section 1.2.2 on page 6). Moreover, section 3.2.2 on page 41 presented a taxonomy of distance learning by

using distance in time and space between students and tutors as distinctive elements, where Lecture 2000 was regarded as one supporting interaction and communication between students and tutors who may be partly distant in time and partly distant in space. It is then very valuable for us to review these two points in terms of what we have done in this thesis.

Separated Learning and Teaching Phases

A collaborative learning process can be divided into different phases, and most existing distance learning systems can just support one or more these phases, but not all. A natural question to ask is how these phases are currently supported by our integrated learning environment:

- Information Sharing
  - production
    The function of producing learning material can be easily integrated into the learning environment because the composer also has a client/server architecture and shares the same repository as the learning environment. A reasonable idea is to incorporate the composer application into the tutor studio. In the future, students in learning groups should be also enabled to use the composer to compose their own learning scripts according to their own preferences and learning advances.
  - distribution and acquisition
    The repository maintaining learning material is accessible by both tutors in classes and students in learning groups.
  - manipulation
    Students in learning groups can make use of this learning material in various way. They can conduct their private study, annotate lecture scripts and use them in their discussions.
- Interaction
  - locating partners
    Within the integrated learning environment, community members can feel free to enter the entrance hall. If allowed, they can enter learning groups to share learning results with group members and to participate in sessions held there. Likewise, group members can also attend chatting held in the entrance hall. In such a way, both group and community members are able to locate potential partners to interact or to collaborate with.
  - group formation
    Both in the entrance hall and in groups rooms, students who have built a trust with each other and have common interests in studies can create their own spontaneous groups, which can later become learning groups in the environment and utilize resources (i.e. learning material) made available by the environment.

– interaction

  Especially, interaction is enabled through various artifacts in learning groups. Both synchronous and asynchronous interactions are supported. With lightweight connections, people can often be brought into a session without the need to actively look for collaborative chances.

Lecture 2000 for People Partly Distant in Time and Space

In section 3.2.2, we assumed that Lecture 2000 supports interaction and collaboration between students and tutors in a consistent way regardless of the distance in time and space between students and tutors, (see Figure 7.1).

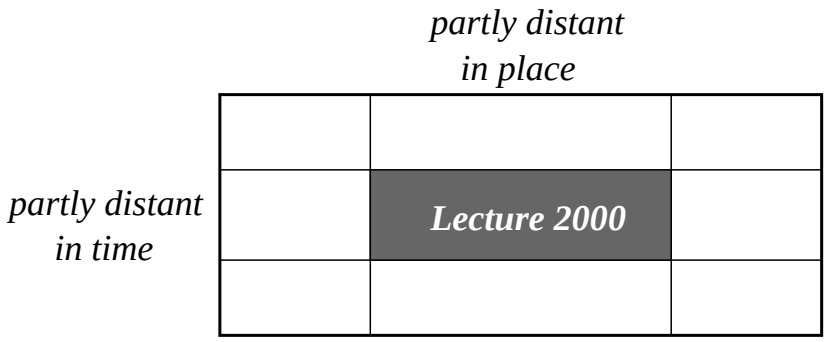


Figure 7.1: Lecture 2000: Partly distance in time and partly distance in space

When they are face-to-face as in classes, tutors can use learning material maintained by the system repository to present their lessons. When they are distributed in place, tutors and students can communicate with each other in two ways:

- with no distance in time (i.e. synchronously)

  Both explicit and implicit meetings are supported in the learning environment.

- with distance in time (i.e. asynchronously)

  The use of common artifacts make the asynchronous collaboration possible.

Though this work is oriented at supporting interaction and collaboration in educational settings, it can, of course, be applied for other working environments as long as both interaction and collaboration have to be supported. The reason is because most organizations work in communities, groups and teams.

7.2 Outlook

7.2.1 Implementation of Other Components in the Spatial Environment

This thesis has primarily focused on the design of learning groups. The discussion group and tutor studio are to be implemented in the future.

- Discussion Group

  The discussion group for a lecture community is intended to offer an opportunity for both community and group members to communicate asynchronously. But this kind of asynchronous communications can happen independent of where they are, which means that students who are not working within the learning environment can contribute to discussions in the discussion group as well. Especially, the discussions in the group are structured according to special interests and topics.

- Tutor Studio

  This studio first of all serves as a workplace for tutors who supervise the learning community. Within the tutor studio, a tutor can prepare his learning material for the learning community, correct tasks and accept students who come here to pay a visit.

7.2.2 Enhancement of the Notification Server

This thesis has paid more attention to the design and development of a notification server providing awareness support for both synchronous and asynchronous communications. NSTP, a Lotus software system, was extended to simplify its implementation. However, such a notification server faces limitations, and improvements of existing functions and implementation of new functions are obviously desired.

Support for Streaming Data

Video and audio are streaming data. Conferencing applications usually transmit continuous streams of information among users. Like most notification servers in multiuser editors, NSTP just transfer discrete state information among users. If video- and audio conferencing functions are needed, some additional infrastructures which are isolated from the learning environment will be required. Curtis *et al* [Curtis95] have pointed out, for example, that while virtual places provide a rich metaphor for a wide range of interactions, there are times when users need other communications. These users should be able to easily set up an audio/video channel between them.

Many systems try to utilize the power of MBone to support video- and audio conferencing among users. A lot of papers [Turlet96, Bauer95, Casner92, Macedo94] have discussed the use of MBone for real-time communications. Due to the inexpensiveness of multicast, it is

desired to integrate multicast functions into the virtual learning environment in the future. Key issue of applying multicast for supporting audio- and video conferencing in Lecture 2000 is that the streaming data must be encrypted so that only allowed users in a session can get information.

CORBA Compatible Architecture

The current NSTP version is implemented with the Socket classes of Java. It is not CORBA-compatible. Thus it can not communicate with other components of this project such as the course composer which is realized with Voyager. It is certainly desired to implement the whole system based on a common platform.

Because the connections between the notification server and clients are implemented with Sockets, NSTP defines a detailed protocol for how to explain the meanings of packages. If Voyager is used to unify the implementation of the system, a lot of interfaces must be defined to replace the role of the existing protocol. In a distributed object model, interfaces are used to realize the connections between servers and clients. A call to remote objects implements the desired functions for local objects.

7.2.3 Integrated Annotation of Learning Materials

Though people have attempted to make proposals [1] to allow annotating hypertext in a convenient way, there is no practical system available to support it. In Lecture 2000, the annotation to lecture scripts is realized through separated annotation nodes which are “attached” to a HTML page in lecture scripts. Two enhancements about the use of lecture scripts are expected:

- Due to the fact that few Web browsers can currently directly support the presentation of XML-based documents, lecture scripts in Lecture 2000 are to be converted into HTML documents to be viewed by tutors in lectures and students in group rooms. It is expected to directly make us of these XML-based learning material in the future.
- In such a way, the annotation can also directly made to XML-based lecture scripts. It is then not necessary to manage lecture scripts and annotations separately.

Of course, it depends on the endeavor of international organizations and some large companies such as the World Wide Web Consortium and Microsoft.

[1] http://www.w3c.org/

Bibliography

[Abel90] M. Abel, D. Corey, S. Bulick, J. Schmidt und S. Coffin. The US West Advanced Technologies TeleCollaboration research project. In G. Wagner (Hrsg.), *Computer Augmented Teamwork*. Van Nostrand Reinhold, 1990.

[Abowd96a] G. D. Abowd. Ubiquitous Computing: Research Themes and Open Issues from an Applications Perspective. Technischer Bericht 96-24. Graphics, Visualization and Usability Center, Georgia Institute of Technology, 1996.

[Abowd96b] G. D. Abowd, C. G. Atkeson, A. Feinstein, C. Hmelo, R. Kooper, S. Long, N. Sawhney und M. Tani. Teaching and Learning as Multimedia Authoring: The Classroom 2000 Project. *Proceedings of ACM Multimedia* (Boston, MA), S. 187–198. ACM Press, New York, NY, November 1996.

[baecke93a] R. M. Baecker, D. Nastos, I. R. Posner und K. L. Mawby. The user-centered iterative design of collaborative writing software. *InterChi'93: Conference on Human Factors in Computing Systems* (Amsterdam), S. 399–405, S. Ashlund, K. Mullet, A. Henderson, E. Hollnagel und R. White (Hrsg.). ACM SIGCHI, April 1993.

[Baecke93b] R. M. Baecker, D. Nastos, I. R. Posner und K. L. Mawby. The User-Centred Iterative Design of Collaborative Writing Software. *Proceedings of ACM SIGCHI Conference on Human Factors in Computing Systems* (Amsterdam, The Netherlands), S. 399–405, S. Ashlund, K. Mullet, A. Henderson, E. Hollnagel und T. White (Hrsg.). ACM Press, New York, NY, April 1993.

[Bair89] J. H. Bair. Supporting cooperative work with computers: Addressing meeting mania. *Proceedings of 34th IEEE Computer Society International Conference–CompCon Spring* (San Francisco, CA), S. 208–217, Februar 1989.

[Balaba97] M. Balabanović. Fab: content-based, collaborative recommendation. *Communications of the ACM*, **40**(3): S. 66–72. ACM Press, New York, NY, März 1997.

[Balasu98] V. Balasubramanian und A. Bashian. Document Management and Web Technologies: Alice Marries the Mad Hatter. *Communications of the ACM*, **41**(7): S. 107–114, Juli 1998.

[Bannon89] L. J. Bannon und K. Schmidt. CSCW: Four characters in search of a context. *Proceedings of 1st European Conference on Computer Supported Cooperative Work* (Gatwick, U.K.), S. 358–372. Computer Sciences House, Sloug, UK, September 1989.

[Bauer95] D. Bauer, E. Wilde und B. Plattner. Design Considerations for a Multicast Communication Framework. *Proceedings of 10th Annual Workshop on Computer Communications* (Eastsound, Washington), September 1995.

[Bellot93] V. M. E. Bellotti und A. Sellen. Design for Privacy in Ubiquitous Computing Environments. *Proceedings of 3rd European Conference on Computer Supported Cooperative Work* (Milan, Italy), S. 77–92, G. de Michelis, C. Simone und K. Schmidt (Hrsg.). Kluwer Academic Publishers, Dordrecht, September 1993.

[Benfor93a] S. Benford, A. Bullock, N. Cook, P. Harvey, R. Ingram und O. Lee. A Model of Conversation Management in Virtual Rooms. *Telepresence'93: 1st International Conference in Technologies and Theories for Human Cooperation, Collaboration and Coordination* (Lille, France), S. 12–23, 1993.

[Benfor93b] S. Benford und L. Fahlén. A Spatial Model of Interaction in Large Virtual Environments. *Proceedings of 3rd European Conference on Computer Supported Cooperative Work* (Milan, Italy), S. 109–124, G. de Michelis, C. Simone und K. Schmidt (Hrsg.). Kluwer Academic Publishers, Dordrecht, September 1993.

[Benfor96] S. Benford, C. Brown, G. Reynard und C. Greenhalgh. Shared Spaces: Transportation, Artificiality, and Spatiality. *Proceedings of Conference Computer Supported Cooperative Work* (Boston, MA), S. 77–86, M. S. Ackerman (Hrsg.). ACM Press, New York, NY, November 1996.

[Bentle97] R. Bentley, W. Appelt, U. Busbach, E. Hinrichs, D. Kerr, K. Sikkel, J. Trevor und G. Woetzel. Basic Support for Cooperative Work on the World Wide Web. *International Journal of Human-Computer Studies*, **46**(6): S. 827–846, Juni 1997.

[Berner94] T. Berners-Lee, R. Cailliau, H. F. Nielsen und A. Secret. The World Wide Web. *Communications of the ACM*, **37**(8): S. 76–82, August 1994.

[Bly93] S. A. Bly, S. R. Harrison und S. Irwin. Media Spaces: Bringing people together in a video, audio, and computing environment. *Communications of the ACM*, **26**(1): S. 28–47, Januar 1993.

[Bodker88] S. Bodker, J. L. Knudsen, M. Kyng, P. Ehn und K. H. Madsen. Computer support for cooperative design. *Proceedings of International Conference on Computer Supported Cooperative Work* (Portland, OR), S. 377–394. ACM Press, New York, NY, September 1988.

[Borgho95] U. M. Borghoff und J. H. Schlichter. *Rechnergestützte Gruppenarbeit — Eine Einführung in Verteilte Anwendungen*, Springer Lehrbuch. Springer Verlag, Berlin, 1995.

[Borgho98] U. M. Borghoff und J. H. Schlichter. *Rechnergestutzte Gruppenarbeit — Eine Einführung in Verteilte Anwendungen*. Springer, August 1998.

[Bornin91] A. Borning und M. Travers. Two approaches to casual interaction over computer and video networks. *Proceedings of ACM SIGCHI Conference on Human Factors in Computing Systems* (New Orleans, LA), S. 13–20, S. P. Robertson, G. M. Olson und J. S. Olson (Hrsg.). ACM Press, New York, NY, April 1991.

[Bruckm97] A. S. Bruckman. *MOOSE Crossing:Construction, Community, and Learning in a Networked Virtual World for Kids*. Dissertation. MIT Media Lab, Juni 1997.

[Carlss93] C. Carlson und O. Hagsand. Dive: A Platform for Multi-User Virtual Environments. *Computer Graphics*, **17**(6): S. 663–669, 1993.

[Carswe98] The 'Virtual University': Toward an Internet Paradigm? *ITiCSE'98* (Dublin, Ireland), S. 46–50, ACM (Hrsg.), 1998.

[Casner92] S. Casner und S. Deering. First IETF Internet Audiocast. *ACM SIGCOMM Computer Communications Review*, **22**(3), Juli 1992.

[Chella97] R. Chellappa, A. Barua und A. B. Whinston. An Electronic Infrastructure For a Virtual University. *CACM*, **40**(9): S. 56–58, September 1997.

[Conkli88] J. Conklin und M. L. Begeman. gIBIS: A hypertext tool for exploratory policy discussion. *Proceedings of International Conference on Computer Supported Cooperative Work* (Portland, OR), S. 140–152. ACM Press, New York, NY, September 1988.

[Covi98] L. M. Covi, J. S. Olson und E. Rocco. A Room of Your Own: What Do We Learn about Support of Teamwork from Assessing Teams in Dedicated Project Rooms? *CoBuild'98 First International Workshop on Cooperative Buildings: Integrating Information, Organization, and Architecture* (GMD, Darmstadt, Deutschland). Springer Verlag, Berlin, Februar 25-26 1998.

[Curtis92] P. Curtis. Mudding: Social Phenomena in Text-Based Virtual Realities. *Proceedings of Conference on the Directions and Implications of Advanced Computing* (Berkley), Mai 1992.

[Curtis95] P. Curtis, M. Dixon, R. Frederick und D. A. Nichols. The Jupiter audio/video architecture: secure multimedia in network places. *Proceedings of of ACM Multimedia'95* (San Francisco, CA), S. 79–90. ACM Press, November 1995.

[Day97] M. Day. What Synchronous Groupware Needs: Notification Services. *Proceedings of 6th Workshop on Hot Topics in Operating Systems (HotOS-VI)* (Cape Cod, MA). Computer Society Press, Mai 1997.

[DellaF88] C. A. DellaFera, M. W. Eichin, R. S. French, D. C. Jedlinsky, J. T. Kohl und W. E. Sommerfeld. The Zephyr Notification Service. *USENIX Conference Proceedings* (Dallas, TX), S. 213–219. USENIX, Winter 1988.

[Dennis98] A. R. Dennis. Lessons from Three Years of Web Development. *Communications of the ACM*, **41**(7): S. 112–113, Juli 1998.

[Dix96] A. Dix. Challenges and perspectives for Cooperative Work on the Web. *the ERCIM workshop on CSCW and the Web* (Sankt Augustin, Deutschland, February 7-9, 1996), Februar 1996.

[douris92a] P. Dourish und V. Bellotti. Awareness and coordination in shared workspaces. *CSCW'92: Sharing Perspectives* (Toronto), S. 107–114, J. Turner und R. Kraut (Hrsg.), November 1992.

[Douris92b] P. Dourish und V. Bellotti. Awareness and Coordination in Shared Workspaces. *Proceedings of International Conference on Computer Supported Cooperative Work* (Toronto, Canada), S. 107–114, J. Turner und R. E. Kraut (Hrsg.). ACM Press, New York, NY, Oktober 1992.

[Douris92c] P. Dourish und S. Bly. Portholes: Supporting Awareness in a Distributed Work Group. *Proceedings of ACM SIGCHI Conference on Human Factors in Computing Systems* (Monterey, CA), S. 541–547, P. Bauersfeld, J. Bennett und G. Lynch (Hrsg.). ACM Press, New York, NY, Mai 1992.

[Douris93] P. Dourish. Culture and control in media spaces. *Proceedings of 3rd European Conference on Computer Supported Cooperative Work* (Milan, Italy), S. 125–137, G. de Michelis, C. Simone und K. Schmidt (Hrsg.). Kluwer Academic Publishers, Dordrecht, September 1993.

[Douris98] P. Dourish. Introduction: The State of Play. *Computer Supported Cooperative Work*, **7**(1): S. 1–7. Kluwer Academic Publishers, Dordrecht, 1998.

[Edward94] W. K. Edwards. Session Management for Collaborative Applications. *Proceedings of International Conference on Computer Supported Cooperative Work* (Chapel Hill, NC), S. 323–330, R. Furuta und C. M. Neuwirth (Hrsg.). ACM Press, New York, NY, Oktober 1994.

[Edward96] W. K. Edwards. Policies and Roles in Collaborative Applications. *Proceedings of Conference Computer Supported Cooperative Work* (Boston, MA), S. 11–20, M. S. Ackerman (Hrsg.). ACM Press, New York, NY, November 1996.

[effels96] W. Effelsberg. Projekt TeleTeaching der UniversitŁten Heidelberg und Mannheim. *Praxis der Informationsverarbeitung und Kommunikation*, **18**(4): S. 205–208, Oktober 1995.

[Ellis91] C. A. Ellis, S. J. Gibbs und G. L. Rein. Groupware – Some Issues and Experiences. *Communications of the ACM*, **34**(1): S. 38–58, Januar 1991.

[Erikss94] H. Eriksson. MBONE: The Multicast Backbone. *Communications of the ACM*, **37**(8), August 1994.

[Fische91] G. Fischer und C. Stevens. Information Access in Complex, Poorly Structured Information Spaces. *Proceedings of ACM SIGCHI Conference on Human Factors in*

Computing Systems (New Orleans, LA), S. 63–70, S. P. Robertson, G. M. Olson und J. S. Olson (Hrsg.). ACM Press, New York, NY, April 1991.

[Fish88] R. S. Fish, R. E. Kraut und M. D. P. Leland. Quilt: A Collaborative Tool for Cooperative Writing. *Proceedings of ACM SIGOIS/IEEE TC-OA Conference on Office Information Systems* (Palo Alto, CA). Veröffentlicht in R. B. Allen (Hrsg.), *SIGOIS Bulletin*, **9**(2&3): S. 30–37, März 1988.

[Fish93] R. S. Fish, R. E. Kraut, R. W. Root und R. E. Rice. Video as a technology for informal communication. *Communications of the ACM*, **36**(1): S. 48–61, Januar 1993.

[Fitzpa95] G. Fitzpatrick, W. J. Tolone und S. M. Kaplan. Work, Locales and Distributed Social Worlds. *Proceedings of 4th European Conference on Computer Supported Cooperative Work* (Stockholm, Sweden), S. 1–16, H. Marmolin, Y. Sundblad und K. Schmidt (Hrsg.). Kluwer Academic Publishers, Dordrecht, September 1995.

[Gaver92a] W. W. Gaver. The Affordances of Media Spaces for Collaboration. *Proceedings of International Conference on Computer Supported Cooperative Work* (Toronto, Canada), S. 17–24, J. Turner und R. E. Kraut (Hrsg.). ACM Press, New York, NY, Oktober 1992.

[Gaver92b] W. Gaver. Realizing a video environment: EuroPARC's RAVE system. *Proceedings of ACM SIGCHI Conference on Human Factors in Computing Systems* (Monterey, CA), S. 27–35, P. Bauersfeld, J. Bennett und G. Lynch (Hrsg.), Mai 1992.

[Gladney97] H. M. Gladney. Access Control for Large Collections. *ACM Transactions on Information Systems*,**15**(2): S. 154–194, April 1997.

[Goldbe92] Y. Goldberg, M. Safran und E. Shapiro. Active mail — A framework for implementing groupware. *Proceedings of International Conference on Computer Supported Cooperative Work* (Toronto, Canada), S. 75–83, J. Turner und R. E. Kraut (Hrsg.). ACM Press, New York, NY, Oktober 1992.

[Greenb96a] S. Greenberg, C. Gutwin und A. Cockburn. Using Distortion-Oriented Displays to Support Workspace Awareness. Technischer Bericht 96/581/01. Department of Computer Science, University of Calgary, Canada, Januar 1996.

[Greenb96b] S. Greenberg und M. Roseman. Groupware Toolkits for Synchronous Work. Technischer Bericht 96/589/09. Department of Computer Science, University of Calgary, Canada, November 1996.

[Greenb98] S. Greenberg und M. Roseman. Using a Room Metaphor to Ease Transitions in Groupware. Technischer Bericht 98/611/02. Department of Computer Science, University of Calgary, Canada, Januar 1998.

[Greenh95] C. Greenhalgh und S. Benford. Virtual Reality Tele-conferencing: Implementation and Experience. *Proceedings of 4th European Conference on Computer Supported*

Cooperative Work (Stockholm, Sweden), S. 165–180, H. Marmolin, Y. Sundblad und K. Schmidt (Hrsg.). Kluwer Academic Publishers, Dordrecht, September 1995.

[Grudin88] J. Grudin. Why CSCW Applications Fail: Problems in the Design and Evaluation of Organizational Interfaces. *Proceedings of International Conference on Computer Supported Cooperative Work* (Portland, OR), S. 85–93. ACM Press, New York, NY, September 1988.

[Grudin91] J. Grudin. CSCW Introduction. *Communications of the ACM*, **34**(12): S. 30–34, Dezember 1991.

[Grudin94] J. Grudin. CSCW: history and focus. *IEEE Computer*, **27**(5): S. 19–26, Mai 1994.

[Gutwin96a] C. Gutwin, M. Roseman und S. Greenberg. A usability study of awareness widgets in a shared workspace groupware system. *ACM 1996 Conference on Computer Supported Cooperative Work* (Boston, MA), S. 258–267, M. S. Ackermann (Hrsg.). ACM Press, November 1996.

[Gutwin96b] C. Gutwin und S. Greenberg. Workspace Awareness for Groupware. *Proceedings of ACM SIGCHI Conference on Human Factors in Computing Systems, Companion Proceedings* (Vancouver, Canada), S. 208–209, April 1996.

[Gutwin96c] C. Gutwin, S. Greenberg und M. Roseman. Workspace Awareness Support With Radar Views. *Proceedings of ACM SIGCHI Conference on Human Factors in Computing Systems, Companion Proceedings* (Vancouver, Canada), S. 210–211, April 1996.

[Gutwin96d] C. Gutwin, S. Greenberg und M. Roseman. Workspace Awareness in Real-Time Distributed Groupware: Framework, Widgets, and Evaluation. *Proceedings of HCI'96* (London, UK), S. 281–298, A. Sasse, R. J. Cunningham und R. Winder (Hrsg.). Springer Verlag, London, August 1996.

[Gutwin98] C. Gutwin und S. Greenberg. Effects of Awareness Support on Groupware Usability. *Proceedings of ACM SIGCHI Conference on Human Factors in Computing Systems* (Los Angeles, CA), S. 511–518, C.-M. Karat, A. Lund, J. Coutaz, J. Karat und S. Pemberton (Hrsg.). ACM Press, New York, NY, April 1998.

[Hamala96] M. Hämäläinen, A. B. Whinston und S. Vishik. Electronic Markets for Learning: Education Brokerages on the Internet. *Communications of the ACM*, **39**(6): S. 51–58, Juni 1996.

[Hardma98] V. Hardman, M. A. Sasse und I. Kouvelas. Successful Multiparty Audio Communication over the Internet. *Communications of the ACM*, **41**(5): S. 74–80, May 1998.

[Harris96] S. Harrison und P. Dourish. Re-Place-ing Space: The Roles of Place and Space in Collaborative Systems. *Proceedings of International Conference on Computer Supported Cooperative Work* (Boston, MA), S. 67–76, M. S. Ackerman (Hrsg.). ACM Press, New York, NY, November 1996.

[Heck98] E. V. Heck und P. Vervest. How should CIOs deal with Web-based auctions? *Communications of the ACM*, **41**(7): S. 99–100, Juli 1998.

[Hicks98] D. L. Hicks, J. J. Leggett, P. J. Nuernberg und J. L. Schnase. A Hypermedia Version Control Framework. *ACM Transactions on Information Systems,* **16**(2): S. 127–160, April 1998.

[Hill98] W. Hill, L. Stead, M. Rosenstein und G. Furnas. Rocommending And Evaluating Choices In A Virtual Community Of Use. Technischer Bericht. ACM Press, New York, NY, 1998.

[Hiltz97] S. R. Hiltz und B. Wellman. Asynchronous Learning Networks as a Virtual Classroom. *Communications of the ACM*, **40**(9): S. 44–49, September 1997.

[Isaacs96] E. Isaacs, J. C. Tang und T. Morris. Piazza: A Desktop Environment Supporting Impromptu and Planned Interactions. *Proceedings of International Conference on Computer Supported Cooperative Work* (Boston, MA), S. 315–324, M. S. Ackerman (Hrsg.). ACM Press, New York, NY, November 1996.

[Isakow98] T. Isakowitz, M. Bieber und F. Vitali. Web Information Systems. *Communications of the ACM*, **41**(7): S. 78–80, Juli 1998.

[Isakow98] T. Isakowitz, M. Bieber und F. Vitall. Web Information Systems. *Communications of the ACM*, **41**(7): S. 78–80, Juli 1998.

[Ishida98] T. Ishida, T. Nishida und F. Hattori. *Community Computing*. John Wiley & Sons, 1998.

[Ishii90] H. Ishii. TeamWorkStation: Towards a seamless shared workspace. *Proceedings of International Conference on Computer Supported Cooperative Work* (Los Angeles, CA), S. 13–26. ACM Press, New York, NY, Oktober 1990.

[Ishii92] H. Ishii, M. Kobayashi und J. Grudin. Integration of inter-personal space and shared workspace: ClearBoard design and experiments. *Proceedings of International Conference on Computer Supported Cooperative Work* (Toronto, Canada), S. 33–42, J. Turner und R. E. Kraut (Hrsg.). ACM Press, New York, NY, Oktober 1992.

[Ishii93a] H. Ishii, K. Arita und T. Yagi. Beyond videophones: teamworkstation-2 for narrowband ISDN. *Proceedings of of the 3rd European Conference on Computer Supp. Coop. Work* (Milano), S. 325–340, G. d.Michelis, C. Simone und K. Schmidt (Hrsg.), September 1993.

[Ishii93b] H. Ishii, M. Kobayaschi und J. Grudin. Integration of interpersonal space and shared workspace: clearboard design and experiments. *ACM Transactions on Information Systems*, **11**(4): S. 349–375, Oktober 1993.

[Johans88] R. Johansen. *Groupware: Computer Support for Business Teams*. The Free Press, Macmillan Inc, NY, 1988.

[Johans91] R. Johansen, D. Sibbet, S. Benson, A. Martin, R. Mittman und P. Saffo. *Leading Business Teams*. Addision Wesley, New York, 1991.

[Kaplan97] S. M. Kaplan, G. Fitzpatrick, T. Mansfield und W. J. Tolone. MUDdling Through. *Proceedings of 30th Annual Hawaii International Conference on System Sciences (HICSS'97)* (Hawaii), S. 539–548. IEEE Computer Society Press, Januar 1997.

[Kirby95] A. Kirby und T. Rodden. Contact: Support for Distributed Cooperative Writing. *Proceedings of 4th European Conference on Computer Supported Cooperative Work* (Stockholm, Sweden), S. 101–116, H. Marmolin, Y. Sundblad und K. Schmidt (Hrsg.). Kluwer Academic Publishers, Dordrecht, September 1995.

[Kirsh98] D. Kirsh. Adaptive Rooms, Virtual Collaboration and Cognitive Workflow. *CoBuild'98 First International Workshop on Cooperative Buildings: Integrating Information, Organization, and Architecture* (GMD, Darmstadt, Deutschland), S. 94–114. Springer Verlag, Berlin, Februar 25-26 1998.

[Koch97] M. Koch. *Unterstützung kooperativer Dokumentenbearbeitung in Weitverkehrsnetzen*. Dissertation. Institut für Informatik, Technische Universität München, München, 1997.

[Kolloc98] P. Kollock und M. Smith. Communities in Cyberspace. *Working draft, University of California, Los Angeles* (1998), 1998.

[Konsta97] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon und J. Riedl. GroupLens: applying collaborative filtering to usenet news. *Communications of the ACM*, **40**(3): S. 77–87. ACM Press, New York, NY, März 1997.

[Kraut90] R. E. Kraut, C. Egido und J. Galegher. 6: Patterns of Contact and Communication in Scientific Research Collaboration. In J. Galegher, R. E. Kraut und C. Egido (Hrsg.), *Intellectual Teamwork*, S. 149–171. Lawrence Erlbaum Publ., Hillsdale, NJ, 1990.

[Kuhn96] W. Kuhn und B. Blumenthal. Spatialization: Spatial Metaphors for User Interfaces. *CHI96*, S. 346–347. ACM, April 1996.

[Lawhea97] P. B. Lawhead, E. Alpert, C. G. Bland und J. DeWitt. The Web and distance learning: what is appropriate and what is not. *SIGCSE/SIGCUE ITiCSE'97*, S. 27–37, 1997.

[Lee96] J. H. Lee, A. Prakash und T. Jaeger. A Software Architecture to Support Open Distributed Collaboratories. *Proceedings of International Conference on Computer Supported Cooperative Work* (Boston, MA), S. 344–353, M. S. Ackerman (Hrsg.). ACM Press, New York, NY, November 1996.

[Loeb92] S. Loeb. Architecting Personal Delivery of Multimedia Information. *Communications of the ACM*, **35**(12): S. 39–48, Dezember 1992.

[Loge94] C. Logé, V. Gay und E. Horlait. Computational Components for Synchronous Cooperation on Multimedia Information. *Int. COST 237 Workshop on Multimedia Transport and Teleservices* (Vienna, Austria), Lecture Notes in Computer Science **882**, S. 89–102, D. Hutchison, A. Danthine, H. Leopold und G. Coulson (Hrsg.). Springer Verlag, Berlin, November 1994.

[Lohse98] G. L. Lohse und P. Spiller. Electronic Shopping. *Communications of the ACM*, **41**(7): S. 81–86, Juli 1998.

[Macedo94] M. R. Macedonia und D. P. Brutzman. MBone Provides Audio and Video Across the Internet. *IEEE Computer*, S. 30–36, April 1994.

[Mackay93] W. Mackay, G. Velay, K. Carter, C. Ma und D. Pagani. Augmenting Reality: Adding Computational Dimentions to Paper. *Communications of the ACM*, **36**(7): S. 96–97, Juli 1993.

[Maes95] P. Maes. Artificial Life Meets Entertainment: Lifelike Autonomous Agents. *ACAM.*, **38**(11): S. 108–114, November 1995.

[Maes97] P. Maes. http://lcs.www.media.mit.edu/people/pattie/. MIT, 1997.

[Malone94] T. W. Malone und K. Crowston. The interdisciplinary study of coordination. *ACM Computing Surveys*, **26**(1): S. 87–119, März 1994.

[Maltz95] D. Maltz und K. Ehrlich. Pointing the way: active collaborative filtering. *Proceedings of ACM SIGCHI Conference on Human Factors in Computing Systems* (Denver, USA), S. 202–209, I. R. Katz, R. Mack, L. Marks, M. B. Rosson und J. Nielsen (Hrsg.). ACM Press, New York, NY, Mai 1995.

[Mantei91] M. M. Mantei, R. M. Baecker, A. J. Sellen, W. A. S. Buxton, T. Milligan und B. Wellman. Experiences in the Use of a Media Space. *Proceedings of ACM SIGCHI Conference on Human Factors in Computing Systems* (New Orleans, LA), S. 203–208. ACM Press, New York, NY, April 1991.

[Miles93] V. C. Miles, J. C. McCarthy, A. J. Dix, M. D. Harrison und A. F. Monk. Reviewing Designs for a Synchronous-Asynchronous Group Editing Environment. In M. Sharples (Hrsg.), *Computer Supported Collaborative Writing*, S. 137–160. Springer Verlag, London, 1993.

[Moffet98] J. D. Moffett. Control Principles and Role Hierarchies. *3rd ACM Workshop on Role-Based Access Fairfax VA*, S. 63–69. ACM Press, New York, NY, 1998.

[Mowsho97] A. Mowshowitz. Virtual Organization. *Communications of the ACM*, **40**(9): S. 30–37, September 1997.

[Muelle97] P. Mueller. Control Architectures for Autonomous and Interacting Agents. In L. C. A. R. W. Wobcke(Eds.), *Intelligent Agent Systems (Based on a Workshop Held at PRICAI'96 Cairns, Australia, Aug. 1996*, Lecture Notes in Artificial Intelligence **1209**, S. 1–26. Springer Verlag, Berlin, 1997.

[Mynatt98] E. D. Mynatt, A. Adler, M. Ito und V. L. O'Day. Network Communities: Something Old, Something New, Something Borrowed, Something Blue. *Computer Supported Cooperative Work*, **7**(1). Kluwer Academic Publishers, Dordrecht, Januar 1998.

[Nakani96] H. Nakanishi, C. Yoshida, T. Nashimura und T. Ishida. FreeWalk: Supporting Casual Meetings in a Network. *Proceedings of International Conference on Computer Supported Cooperative Work* (Boston, MA), S. 308–314, M. S. Ackerman (Hrsg.). ACM Press, New York, NY, November 1996.

[Nichol95] D. A. Nichols, P. Curtis, M. Dixon und J. Lamping. High-latency, low bandwidth windowing in the Jupiter collaboration system. *8th Annual Symposium on User Interface Software and Technology, UIST'95* (Pittsburgh, PA), S. 111–120. ACM Press, November 1995.

[Nomura98] T. Nomura, K. Hayashi, T. Hazama und S. Gudmundson. Interlocus: Workspace Configuration Mechanisms for Activity Awareness. *CSCW 98* (Seattle Washington USA), S. 19–28, ACM (Hrsg.), November 1998.

[Nunama91] J. F. Nunamaker, A. R. Dennis, J. S. Valacich, D. R. Vogel und J. F. George. Electronic Meeting Systems to Support Group Work. *Communications of the ACM*, **34**(7): S. 40–61, Juli 1991.

[Obata98] A. Obata und K. Sasaki. OfficeWalker: A Vitual Visiting System Based On Proxemics. *Proceedings of International Conference on Computer Supported Cooperative Work* (Seattle Washington USA), S. 1–10. ACM Press, New York, NY, November 1998.

[ODay96] V. L. O'Day, D. G. Bobrow und M. Shirley. The Social-Technical Design Circle. *Proceedings of International Conference on Computer Supported Cooperative Work* (Boston, MA), S. 160–169, M. S. Ackerman (Hrsg.). ACM Press, New York, NY, November 1996.

[Olson98] Dan. R. Olson, S. E. Hudson, M. Phelps und J. Heiner. Ubiquitous Collaboration via Surface Representations. *Proceedings of International Conference on Computer Supported Cooperative Work* (Seattle, Washington), S. 129–138, M. S. Ackerman (Hrsg.). ACM Press, Seattle, Washington, November 1998.

[Orfali96] R. Orfali, D. Harkey und J. Edwards. *The Essential Distributed Objects Survival Guide*. John Wiley Incorporated, New York, NY, 1996.

[Palfre96] K. Palfreyman und T. Rodden. A Protocol for User Awareness on the World Wide Web. *Proceedings of International Conference on Computer Supported Cooperative Work* (Boston, MA), S. 130–139, M. S. Ackerman (Hrsg.). ACM Press, New York, NY, November 1996.

[Patter90] J. F. Patterson, R. D. Hill, S. L. Rohall und W. S. Meeks. Rendezvous: An Architecture for Synchronous Multi-User Applications. *Proceedings of International Conference on Computer Supported Cooperative Work* (Los Angeles, CA), S. 317–328. ACM Press, New York, NY, Oktober 1990.

[Patter96] J. F. Patterson, M. Day und J. Kucan. Notification Servers for Synchronous Groupware. *Proceedings of International Conference on Computer Supported Cooperative Work* (Boston, MA), S. 122–129, M. S. Ackerman (Hrsg.). ACM Press, New York, NY, November 1996.

[Ramdun98] D. Ramduny, A. Dix und T. Rodden. Exploring the Design Space for Notification Servers. *Proceedings of International Conference on Computer Supported Cooperative Work* (Seattle, WA), S. 227–235. ACM Press, New York, NY, November 1998.

[Reichw98] R. Reichwald, K. Möslein, H. Sachenbacher, H. Englberger und S. Oldenburg. *Telekooperation — Verteilte Arbeits- und Organistationsformen*. Springer, Heidelberg, 1998.

[Riggin98] F. J. Riggins und H.-S. (Sue)Rhee. Toward a Unified View of Electronic Commerce. *Communications of the ACM*, **41**(10): S. 88–95, Oktober 1998.

[Root88] R. W. Root. Design of a multi-media vehicle for social browsing. *Proceedings of International Conference on Computer Supported Cooperative Work* (Portland, OR), S. 25–38. ACM Press, New York, NY, September 1988.

[Rosema96a] M. Roseman und S. Greenberg. Building Real-Time Groupware with GroupKit, A Groupware Toolkit. *ACM Transactions on Computer-Human Interaction*, **3**(1): S. 66–106, März 1996.

[Rosema96b] M. Roseman und S. Greenberg. TeamRooms: Network Places for Collaboration. *Proceedings of International Conference on Computer Supported Cooperative Work* (Boston, MA), S. 325–333, M. S. Ackerman (Hrsg.). ACM Press, New York, NY, November 1996.

[Sandhu96] Ravi. J. Sandhu und H. LFeinstein. Role-Based Access Control Models. *IEEE Computer*, **29**(2): S. 38–47, February 1996.

[Schlic97a] J. Schlichter. Lecture 2000: More than a course across wires. *Teleconference - The Business Communications Magazine*, **16**(6): S. 18–21, 1997.

[Schlic97b] J. Schlichter, M. Koch und M. Bürger. Workspace Awareness for Distributed Teams. *Proceedings of Coordination Technology for Collaborative Applications - Organizations, Processes, and Agents* (Singapore), Lecture Notes in Computer Science **1364**, S. 199–218, W. Conen und G. Neumann (Hrsg.). Springer Verlag, Berlin, 1997.

[Schlic98] J. Schlichter, M. Koch und C. Xu. Awareness - The Common Link Between Groupware and Community Support Systems. *Proceedings of First Kyoto Meeting on Social Informatics* (Kyoto, Japan), Juni 1998.

[Schuck96] C. Schuckmann, L. Kirchner, J. Schümmer und J. M. Haake. Designing Object-Oriented Synchronous Groupware with COAST. *Proceedings of Conference Computer Supported Cooperative Work* (Boston, MA), S. 30–38, M. S. Ackerman (Hrsg.). ACM Press, New York, NY, November 1996.

[Shen92] H. Shen und P. Dewan. Access Control for Collaborative Environments. *Proceedings of International Conference on Computer Supported Cooperative Work* (Toronto, Canada), S. 51–58, J. Turner und R. E. Kraut (Hrsg.). ACM Press, New York, NY, Oktober 1992.

[Smith98] R. B. Smith, R. Hixon und B. Horan. Supporting Flexible Roles in a Shared Space. *Proceedings of International Conference on Computer Supported Cooperative Work* (Seattle Washington USA), S. 197–206. ACM Press, New York, NY, November 1998.

[Stefik87] M. Stefik, D. G. Bobrow, G. Foster, S. Lanning und D. Tatar. WYSIWIS Revised: Early Experiences with Multiuser Interfaces. *ACM Transactions on Office Information Systems*, **5**(2): S. 147–167, 1987.

[Streit96] N. A. Streitz. Ubiquitous Collaboration and Virtual Organisations. *Proceedings of Workshop Telekooperations-Systeme in dezentralen Organisationen* (Berlin, Deutschland), S. 1–6, K. Sandkuhl und H. Weber (Hrsg.). Fraunhofer-Gesellschaft, ISST, Berlin, Februar 1996.

[Streit98] N. A. Streitz, J. Geissler und T. Holmer. Roomware for Cooperative Buildings: Integrated Design of Architectual Spaces and Information Spaces. *CoBuild'98 First International Workshop on Cooperative Buildings: Integrating Information, Organization, and Architecture* (GMD, Darmstadt, Deutschland). Springer Verlag, Berlin, Februar 25 1998.

[Tang94] J. C. Tang und M. Rua. Montage: Providing Teleproximity for Distributed Groups. *Proceedings of ACM SIGCHI Conference on Human Factors in Computing Systems* (Boston, MA), S. 37–43. ACM Press, New York, NY, April 1994.

[Tenenb98] J. M. Tenenbaum. WISs and Electronic Commerce. *Communications of the ACM*, **41**(7): S. 89–90, Juli 1997.

[Teufel95a] S. Teufel, C. Sauter, T. Mühlherr und K. Bauknecht. *Computerunterstützung für die Gruppenarbeit*. Addison-Wesley Publishing Company, 1995.

[Teufel95b] S. Teufel und B. Teufel. Bridging Information Technology and Business — Some Modelling Aspects. *ACM SIGOIS Bulletin*, **16**(1): S. 13–17. ACM Press, New York, NY, August 1995.

[Trevor97] J. Trevor, T. Koch und G. Woetzel. MetaWeb: Bringing synchronous groupware to the World Wide Web. *Proceedings of 5th European Conference on Computer Supported Cooperative Work* (Lancaster, UK), J. A. Hughes, W. Prinz, T. Rodden und K. Schmidt (Hrsg.). Kluwer Academic Publishers, Dordrecht, September 1997.

[Tsichr99] D. Tsichritzis. Reengineering the University. *Communications of the ACM*, **42**(6): S. 93–100, Juni 1999.

[Turlet96] T. Turletti und C. Huitema. Videoconferencing in the Internet. *IEEE/ACM TRANSACTION ON NETWORKING*, **4**(3): S. 340–351, Juni 1996.

[Turoff95] M. Turoff. Designing a Virtual Classroom[TM]. *1995 International Conference on Computer Assisted Instruction* (National Chiao Tung University, Hsinchu, Taiwan), März 1995.

[Turoff97] M. Turoff. Virtuality. *Communications of the ACM*, **40**(9): S. 38–49, September 1997.

[webste96] *Webster's encyclopedic unabridged dictionary of the English language*. Gramercy Books, New York, 1996.

[Weiser93] M. Weiser. Some computer science issues in ubiquitous computing. *Communications of the ACM*, **36**(7): S. 75–84, Juli 1993.

[Weiser98] M. Weiser. The Future of Ubiquitous Computing on Campus. *Communications of the ACM*, **41**(1): S. 41–42, Januar 1998.

[Whitta94] S. Whittaker, D. Frohlich und O. Daly-Jones. Informal Workplace Communication: What is it like and how might we support it? *Proceedings of ACM SIGCHI Conference on Human Factors in Computing Systems* (Boston, MA), S. 131–137, April 1994.

[Whitta97] S. Whittaker, J. Swanson, J. Kucan und C. Sidner. TeleNotes: Managing lightweight interactions in the desktop. *ACM Transactions on computer-human interaction*, **4**(2): S. 137–168, Juni 1997.

[Whitta98] S. Whittaker, L. Terveen, W. Hill und L. Cherny. The dynamics of mass interaction. *Proceedings of International Conference on Computer Supported Cooperative Work* (Seattle, USA), S. 257–264, S. Poltrock und J. Grudin (Hrsg.). ACM Press, New York, NY, November 1998.

[Wilson91] P. Wilson. Introducing CSCW - what is it and why we need it. In P. Wilson (Hrsg.), *Computer supported cooperative work: the multimedia and networking paradigm*, Juli 1991.

[Woo94] T. K. Woo und M. Rees. A Synchronous Collaboration Tool for World-Wide Web. Distributed Systems Technology Center, The University of Queensland, Queensland 4072, 1994.

List of Figures

List of Tables

Index