

## Supporting Collaborative Exercises for Distance Education

Jörg M. Haake, Till Schümmer, and Anja Haake

*FernUniversität Gesamthochschule in Hagen*

*Computer Science VI – Distributed Systems*

*Universitaetsstr. 1, D-58084 Hagen, Germany*

*Email: {joerg.haake, till.schuemmer, anja.haake}@fernuni-hagen.de*

### Abstract

*At the German Distance Learning University, collaborative synchronous exercises have been recently identified by students and teachers as an important future form of collaborative learning in the university's virtual learning space. The main requirements of collaborative exercises in a distance learning university include support for preparation of exercises, learning group management, collaborative learning sessions, and learning management. We support such collaborative exercises in the FUB system by providing groupware tools for each phase of collaborative exercises. Especially important is the support of complex problem solving processes. Results of a trial use indicate that our approach works, and identify some needs further improvement. The implications of our approach and our experiences for the design of next generation learning platforms are discussed.*

### 1. Introduction

The FernUniversität Hagen is the German distance learning university. Teaching at the FernUniversität includes different forms of learning: courses, seminars, and different forms of practical problem solving (in a so called practicum). Course material and accompanying individual exercises are sent to distributed students via surface mail or the Internet. Course-specific newsgroups and direct e-mail communication with professors and teaching assistants support asynchronous discussions. Students might also use the telephone to contact staff directly. In addition, students can meet tutors and other students at one of the 60 learning centers, if one exist near their location.

Students primarily learn individually. It is difficult for them to find appropriate co-learners and to learn together. As a result, students feel isolated, lack practice of collaboration, and miss the motivation that teamwork and team members may provide.

Current approaches in distance education trying to address some of these problems include:

- Facilitated communication through a newsgroup or forum, or in chat rooms,
- Broadcast of live lectures with limited ability to ask questions, and
- Asynchronous team-work in practical-oriented courses (e.g. lab experiments or seminar work) or other forms of problem-based learning (PBL) [7].

Typically, these approaches focus either on group discussions (asynchronous or synchronous) or on asynchronous collaboration. Some approaches support spontaneous short-term question-related collaboration. Others support long-term collaboration for PBL.

Our approach aims on facilitating medium-term, planned collaborative exercises in the context of a distance learning course with distributed students. We argue that synchronous exercises are beneficial because of direct feedback among students. Students correct themselves during the exercise, and because of shared experiences students can begin to feel as a group and have colleagues, who motivate them. These exercises are prepared by the teacher, and provide guidance to the problem solving and learning process through carefully designed learning material. While the teacher heavily determines the task and the problem solving process, the students are free to organize their actual learning. In order to support this type of learning, we developed a prototype of a collaborative learning environment called FUB.

The reminder of this paper is organized as follows. It begins with a problem analysis of collaborative exercises. The main body of this paper describes our approach to design and implement a collaborative learning environment called FUB for collaborative exercises. After presenting our experiences with using FUB in a course, we analyze the implications of our approach for next generation learning platforms. Then, we compare our approach with existing collaborative learning and working environments and learning platforms. Finally, we present our conclusions and future work.

## 2. Problem Analysis

Since the early 90's, the FernUniversität Hagen uses computer-supported learning to support distributed learning scenarios, such as virtual seminars and virtual lab experiments. In 1998, these experiences and developments led to the ongoing development of the FernUniversität virtual learning space. As a basis for this learning space, a series of learning platforms were developed, of which the next release is planned for 2003.

Recently, an analysis of potential collaborative learning scenarios identified collaborative exercises as one of the most wanted learning situations among teachers and students. Consequently, work started on analyzing the requirements of supporting distributed groups of students working on joint exercises.

We illustrate our understanding of collaborative exercises using a story, which was developed while performing a use case analysis:

During distance learning courses, e.g. the operating systems course in the school for computer science, course units and accompanying exercises are sent to the students every two weeks. Students then solve these exercises and send their solutions back for evaluation/correction. In our scenario, students want to solve these exercises collaboratively in small groups

Initially, a student who is enrolled in the operating systems course must join a learning group. To do so, the student must first discuss when enough partners are interested in performing the exercise. In a second step, interested students enter the system at the negotiated date and time and select the exercise, which he/she wants to solve next. The system in response shows all students, who are currently online and interested in this exercise. This pool of students is the set from which learning groups can be built. One student can then select one to three prospective peer learners and start a new learning group with them.

The group members now jointly solve the exercise. In our example, the teacher designed an exercise, in which the students were asked to compare two concepts explained in the course material. He structured the exercise into two phases: first, the students are asked to perform a brainstorming on properties useful for comparing the concepts. In a second step, the students are asked to construct a semantic network, which should characterize both concepts and show the relationships between the two concepts.

When they finish the exercise the learning environment forwards their solution (i.e. the final semantic network) to an evaluation/correction step.

Now, the group members either start another cycle (i.e. negotiate the next exercise, date and time) or students

decide to leave the group and join another learning group (e.g. when they feel unhappy with the current team).

The benefits of this scenario include:

- Students can learn in teams (as opposed to usual isolated learning). The learning environment provides a medium for all group activities, such as scheduling, coordination, communication, and joint problem solving (e.g. joint drawing, writing).
- Students are free to choose their learning groups. Students can change group membership after an exercise is finished. Students can continue collaboration. Thus, communities of interest can emerge.
- Students are free to schedule and structure their group problem solving. However, teachers can guide students through the problem solving process by providing the material, hints, and explicit steps of the problem solving process (such as brainstorming and semantic network construction in the above scenario).

Based on this scenario, we can identify the following requirements for a collaborative learning environment facilitating such collaborative exercises:

**Preparation of exercises:** Teachers must be able to construct exercises with a varying degree of didactic structure. An exercise can be characterized by task description, initial data to work with, a prescribed problem solving process (incl. roles and steps), a characterization of the result, and a recommended group size and time frame.

**Learning group management:** Students must be able to browse and join open groups before they work on an exercise. The system should facilitate negotiation of exercises, dates and times among the group members. In case of synchronous work, latecomers must be supported.

**Collaborative learning:** We use a constructivist learning approach. Students must be able to jointly construct the solution to the task. Thus, the system must support joint viewing and editing of task material. For each required step in a problem solving process, dedicated support should be offered (e.g. a brainstorming tool and a semantic network construction tool in our scenario). Transitions between steps (i.e. control and data flow between tools) must be supported. Especially important is to support going back and forth between steps, since only then exploratory and iterative learning can happen. For a structured problem solving process, the system should provide some guidance through the learning process.

**Learning management:** Students need support for selecting exercises and submitting joint solutions, and for checking the status of exercises (e.g. not yet solved, submitted, evaluated) and of their own progress (e.g. how many points were achieved in the course, how many are still needed). Subsequently, results of an

evaluation/correction need to be communicated back to the group. Further support for group discussions of the corrected solution with or without tutors would be useful.

### 3. Approach

This section describes our approach to develop a collaborative learning environment supporting collaborative exercises. Based on a representation of collaborative exercises, which can be used by the learning environment to guide distributed students through the learning experience, we provide tools for the different phases of a collaborative exercise and show how they support the four groups of requirements identified in the previous section.

#### 3.1. Supporting the preparation of collaborative exercises

A collaborative exercise can be characterized by a task description (goals and objectives, result), some initial data to work with, a prescribed problem solving process (incl. roles and steps), a characterization of the result and the evaluation/correction method (e.g. by a human, or by a software agent), and a recommended group size and time frame. It is associated with a course unit, which should contain the necessary information to understand and solve the task. It is important to note that not all properties of a collaborative exercise are needed in all circumstances: only task description, characterization of result and the evaluation/correction method are mandatory.

The current collaborative learning environment provides the following exercise management services for the teacher:

- Create, copy, and delete an exercise representation
- Edit an exercise representation in order to manipulate the individual properties such as the
  - Task description
  - Initial data: optional
  - Problem solving process: required is a one step process associated with a specific tool; more complex processes with multiple steps are optional
  - Characterization of the result (document type) and the evaluation/correction method (manual or automatic, by whom)
  - Group size: required to define the minimum and maximum number of participants
  - Associated to course unit: required

The problem solving process representation can be used to guide students through the learning process. The learning environment can show the steps in the user

interface and enable students to select a step or switch back and forth between steps.

Figure 1 shows the FUB task editor for teachers. In this example, it displays an exercise called “Übung 5”. The tool shows a textual description of the task under the heading “Anweisung” (task description), the group size (min, max) under heading “Gruppengröße”, the e-mail address and IP address of the mail server to be used to submit a solution for evaluation/correction under the headings “Korrektor” and “Mailserver”, and a list of three instances of this exercise. The interface elements in the right part of the window (the list with time-stamped exercise solutions and the buttons labeled “Export” and “Löschen”) allow the teacher to manage the submitted results of the task.

Note that new tasks are not created with this tool, but within the learning group management tool. In this tool, all tasks are listed and have an attached context menu that allows the creation, removal and change of exercises. The list also serves as a starting point for students, who want to work on a specific task. We will explain this in the next section.

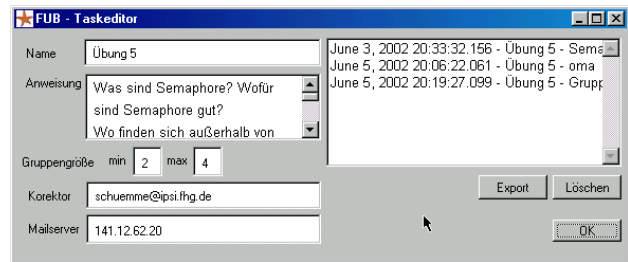


Figure 1: The Task Editor in FUB

#### 3.2. Supporting learning group management

In order to allow students to join learning groups, the following services are supported by the FUB Group Management Tool (see Fig. 2):

- Students can see a list of possible exercises. When they select an exercise (from the list of “possible exercises” (“Mögliche Aufgaben”), the system shows all other students, who also want to work on this task in the list of “possible partners” (“Mögliche Partner”).
- Students can start a new learning group. For this, they select the exercise from the list of “possible exercises” and the prospective group members from the list of “possible partners”. Pressing the “start” button will then create a new group that works on the selected exercise. All prospective group members will receive invitation messages for this new group. Accepting the invitation will open the collaborative learning session for them.

- Students can browse open learning groups (see list of “active groups” (“Aktive Gruppen”) in Fig. 2).
- Students can join a learning group. If a student wants to join a group, he selects the group from the list of open learning groups and presses the “participate” (“Teilnehmen”) button (see Fig. 2). All current group members are asked by the system, whether or not they want to allow this person to join them. Based on a poll, the system then decides if the new user can enter the running collaborative application (e.g. the brainstorming tool).

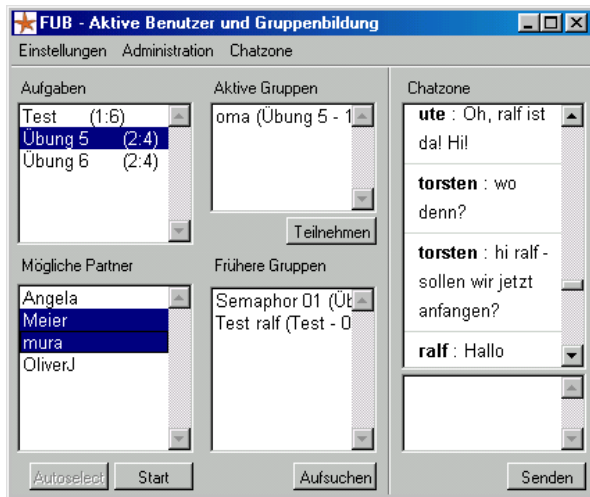


Figure 2: The FUB Group Management Tool

- Members of a learning group can keep the group closed by rejecting new members.
- The learning environment facilitates the negotiation of exercises, dates and times among the group members. In the current version, a newsgroup is provided for this purpose.
- In case of synchronous work, late comers for synchronous learning sessions are supported by: allowing them to join the desired group (as described above) or providing the rest of the group sufficient awareness, that the latecomer is not yet logged on at the system. In the latter case, the group may either start the exercise and invite the late user when he arrives, or wait in the group management tool, until the late user arrives. If they decide to wait, the other members may do some social chat in the chat zone and when they notice that everyone is there, they may start the collaborative exercise as described above.
- Students can revisit groups. All episodes (i.e. previous sessions) of collaborative exercises are stored in the list of “former groups” (“Frühere

Gruppen”). This allows the students to revisit the learning session and have a look at the results.

### 3.3. Supporting collaborative learning sessions

In our scenario, students must be able to jointly construct the solution to the task. Thus, upon starting an exercise, the learning environment provides access to a shared workspace containing the task description and initial material. The learning process is embodied in the connection of the tools, which are linked according to the control and data flow defined by the learning process. The latter contains a list of the steps, which lead upon activation to the appropriate tool (as identified in the problem solving process defined in the exercise). In the simple case, this is just the tool defined in a one step learning process.

For each required step in a problem solving process, dedicated support is offered by a specific tool (e.g. a brainstorming tool and a semantic network construction tool in our scenario). Upon activation, the tool loads the specified input data and displays it to all group members. The tool supports joint viewing and editing of the material. In order to facilitate collaboration, each tool provides group awareness in terms of who is currently present.

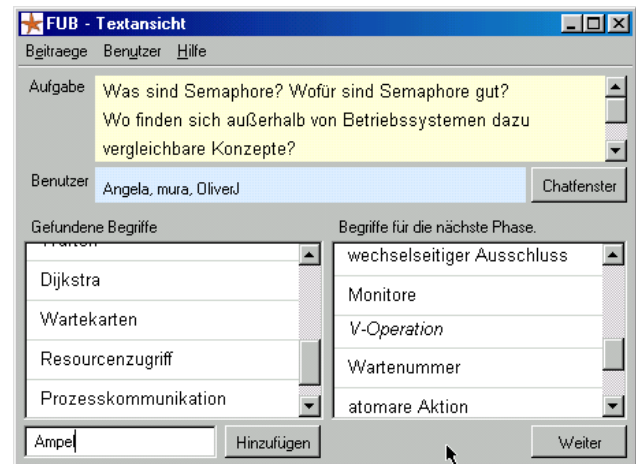


Figure 3: The Tool to perform the first phase of the collaborative brainstorming task

Figure 3 shows the cooperative brainstorming tool, which displays the task description (“Aufgabe”), the current users (“Benutzer”), and two jointly editable lists to the students. The first list (“Gefundene Begriffe”) is intended for brainstorming, while the second list (“Begriffe für die nächste Phase”) is used to collect those brainstorming results selected as the starting point for the next phase (i.e. semantic network construction).

Transitions between steps in the problem solving process are supported by a two-phase procedure. When one group member presses the “next” (“Weiter”) or “finish” (“Fertig”) button of a tool, the final state of its content is regarded as the result of the current step. When users move from one task to another, the system ensures that the whole group follows. Thus, only one user has to act as a leader when navigating through the subtasks. At each transition between tasks, the domain models of the specific source-tool and the target-tool are synchronized. When a student in Fig. 3 presses the “next” (“Weiter”) button, the brainstorming tool is closed and the semantic network construction tool (see Fig. 4) is opened. Automatically, the result of the previous tool is displayed as a starting point. Now, students can jointly construct the semantic network. Using the “back” (“Zurück”) button allows them to go back to the previous phase (i.e. to the brainstorming tool), while with the “finish” (“Fertig”) button they can finish work and submit their result to the evaluation/correction phase.

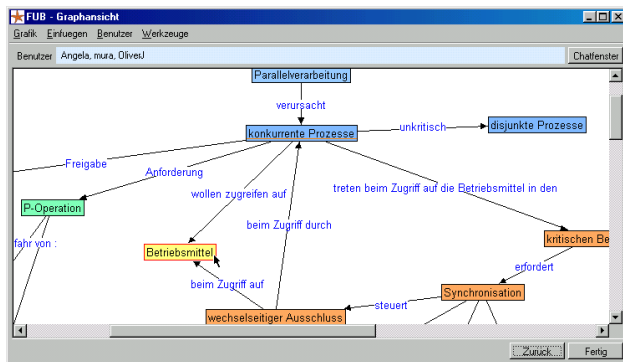


Figure 4: The Semantic Network Tool

This way, going back and forth between steps is easily supported. Groups can notice a shortcoming of an earlier step later in the process, and can go back to repair the deficit. Consequently, the following steps should be re-examined in order to make the solution consistent. Thus, exploratory and iterative learning can happen.

At this time, the system does not show where students are in the problem solving process. However, it would be possible for the system to provide some guidance through the learning process by displaying the steps in the proposed order and by showing steps which are still to be (re-)executed. In addition, the teacher may add explanations and hints to the description of the problem solving process, thus providing further guidance.

### 3.4. Supporting learning management

Students need support for selecting exercises and submitting joint solutions, and for checking the status of

exercises and of their own progress (e.g. how many points were achieved in the course, how many are still needed).

This is achieved by providing the shared group exercise workspace: Students can browse, negotiate and select exercises. Students can submit their solutions as the final step of the problem solving process. The learning environment then packages the solution according to the specification of the exercise, tags it with group and member IDs, and forwards it to the evaluation/correction phase. This can either be a human tutor receiving the solution via e-mail, or a software agent checking the correctness of the solution. At the FernUniversität, we developed and use the WebAssign system [3, 19] for many types of exercises.

Subsequently, the result of an evaluation/correction needs to be communicated back to the group. Again, this is mediated by the learning environment, which notifies the students about availability of the correction. Further support for group discussions of the corrected solution with or without tutors could be implemented, using the services and mechanisms already provided by the learning environment.

## 4. Implementation

The core of the FUB learning platform is the provision of synchronous exercises. Distributed groups of students use special synchronous groupware applications that provide shared workspaces and guide their progress through these exercises. In this section, we will first briefly describe the COAST groupware framework that was used to implement FUB. Then we will show how exercises with different phases are modeled in FUB. Finally, we will describe the architecture and show how FUB can be integrated within the FernUniversität learning platform.

### 4.1 The COAST framework

To implement such tools, we used the open source COAST groupware framework [15, 17]. This framework

- offers a data description language to express the application’s domain model – in our case for example the brainstorming entries,
- provides mechanisms for a set of users to synchronously manipulate the shared domain model,
- keeps the shared model and its visualizations at a consistent state,
- supports the provision of clues about other user’s activities (group awareness),
- includes a pre-defined extensible model of users and their work environments, and
- assists the developer in modeling user interaction and collaborative sessions.

Applications built with COAST basically contain four parts [16]: the domain model, the user model, the shared application model, and the local view-controller pair. We will explain each part in a separate paragraph.

The *domain model* represents the problem specific data as shared objects. In COAST, all shared objects are (on demand) replicated to the clients. Clients can manipulate shared objects and the resulting changes are automatically propagated to the other clients. In FUB, the resulting semantic network is an example for the shared domain model.

The *user model* describes the users working with the application. This user model is much more important in groupware applications than it is in single user applications, because groupware applications are used by more than one user.

Developers use the *shared application model* to implement logical session management. To open a collaborative tool for a specific user, the developer adds this user's user model to the set of users of the shared application model that represents this collaborative tool. When users interact with the groupware their user objects are added as interested parties to the shared application model. When they work with the application they manipulate the content of a shared application model. Developers thus explicitly model, who is collaborating with whom using what kind of tools.

The COAST framework monitors this information to keep *local views and controllers* consistent. If a user for instance uses a brainstorming tool, then his user representation is included in the set of users of the shared application model that represents the brainstorming tool. COAST will automatically open a window (the view-controller pair) at this user's machine.

## 4.2 Modeling exercises

The current FUB implementation includes three different forms of cooperative tools, which are all modeled as shared application models: A chat, a textual brainstorming, and a semantic network creation tool.

The teacher can specify an exercise by providing templates of the shared application models together with an initial shared domain model. In addition, he has to relate the created templates to model the transition between the different phases. Currently, an application developer is needed who composes the problem solving process representation out of predefined components. In a future version, we will provide a graphical composition environment using representations as shown in figure 5.

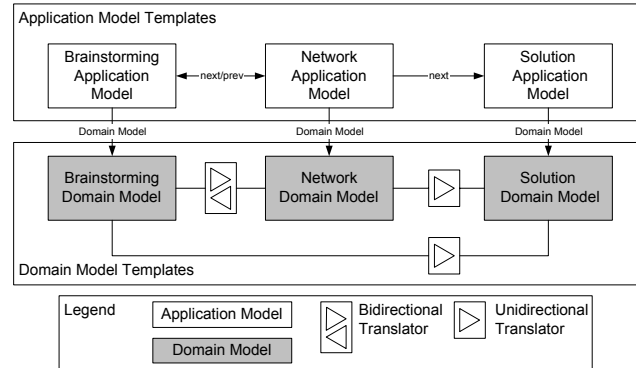


Figure 5: Application Models and Domain models in the current FUB exercises.

Figure 5 shows the composition that we used to model the exercises that we use in our current course. It shows the three shared application models that are sequentially used by the students. The student enters the exercise by opening the shared brainstorming application model. The brainstorming's domain model contains two word lists (not shown in Fig. 5): the words that were collected in the brainstorming and the words, which were selected for further use in the consolidation phase of the brainstorming.

When the student decides to move to the next phase, this is reflected in the shared application models of the tools (the user representations are moved from one to another shared application model). This change triggers a translation of the brainstorming data to the semantic network editor's data and to the solution domain model (cf. the translator icons in figure 5). In our case, the entries that were collected in the word list for the next phase are translated into nodes of the semantic network and put to a random position. Both word-lists are added to the solution's domain model, since they shall be submitted when the task is solved. Translator objects perform this translation, which gives the system high flexibility and adaptability. When students move back to the Brainstorming application model, the translator incremental updates the word lists of the brainstorming domain model to reflect the semantic network's state. Note that figure 5 shows an additional translator that creates a submittable solution out of each phase's results, whenever the users exits this phase.

When the student enters the FUB system, it presents the set of created exercises. By creating a new learning group for one of the exercises, the student enters the workflow of the selected exercise. Technically, this means that the application model templates (cf. fig. 5) are copied and that the user representations of the group's users are added this newly created brainstorming application model.

### 4.3 Architecture

COAST applications use a client-server architecture: A central server – called mediator – is responsible for replicating the shared data to the clients, maintaining consistency between the clients shared data, and keeping the data persistent.

Clients locally perform all user actions and send information on state changes to the mediator, who in response distributes change notifications to other clients participating in the collaboration. The application logic is implemented completely in the clients. Thus, the server can be kept independent from any specific application.

Prospective users of the FUB system have to copy the client software to their local machines. Since we didn't know many details about the end-users system configuration (especially whether or not the user is allowed to install any software), we decided to ship the client as a single executable, which does not require any system changes or installation.

The integration of the FUB platform with the rest of the FernUniversität learning platform was achieved by using special monitoring clients. These clients also connect to the FUB mediator and monitor the activities that take place. Whenever the students for instance decide to submit a solution, the monitoring client resends the solution by mail to the correctors of the course. With the same technology, one can add the solution to the existing WebAssign system [3, 19], which then automatically checks the solution.

## 5. Experiences

We currently use FUB in a course on operating systems. The course has 7 sections (learning units) and 7 corresponding assignments. Each assignment has between 4 and 6 tasks. In assignments 5 and 6, we included one task, which had to be done in a virtual group using FUB.

FUB was introduced in the course because students tended to learn facts without understanding the larger context and because they were not able to discuss about these relationships. Asking them to create a semantic network using FUB should help them to draw connections between the different topics covered by the single course units. Cooperative learning was required in order to force the students to discuss the different concepts, while they add them to the concept map.

This course had about 400 registered students. Initially, 80% of them participated actively in the way that they sent in their exercises and request corrections. This percentage naturally goes down during the semester. When we started our experiment in the last third of the semester approximately 80 students still sent in assignments.

We asked the students to use the course's newsgroup to check for time slots, where groups between 2 and 4 students could meet with in the FUB system. News traffic started in the last three days before the official days, where the assignment should run. Thus, we could observe that the students started organizing themselves as late as possible.

11 students did actively seek for peer learners. Three of them unfortunately didn't want to use the packaged windows version, but asked for a Linux version of the tool. Although this would have been technically possible (FUB is implemented in VisualWorks Smalltalk, which provides virtual machines for all commonly used platforms), the request was too late to package a new version.

So, 8 students actively ran the exercises. This number seems to be quite low, but we see several reasons that explains this:

- The exercises were voluntary for all students. Thus, there was no need to participate in the virtual exercise to successfully finish the course.
- Because only two of the exercises used the virtual learning technology, it was possible to solve most of the exercises without using the tool.
- FUB came with a long (10 pages) user's manual, which might have shocked some students.
- The exercise demanded that the students find a shared time-slot where they could solve the task. Although this worked fine for all groups that participated, it requires additional coordination, which might have hindered several students from participating.

The usage experience that we have so far is thus limited to the three groups. Instead of providing any statistical usage results, we will therefore describe how one group solved the task using the tool. Our description is based on chat logs and the final solution that the group submitted. The group members learned to know each other by means of the news group. They came up with dates, where they wanted to solve the task.

At the agreed date, two of the three students arrived in time. They did social chat until the third student arrived. After 22 minutes, one of the first students noticed the third student's arrival and they went on to the task.

Throughout the whole task, they used a chat tool in addition to the brainstorming and semantic network tool, which was globally available in FUB. The group first worked with the brainstorming tool for 11 minutes. In this time, 16 (47%) of 24 chat messages regarded the solution of the exercise. 4 (12%) messages regarded the understanding of the assignment, and 14 (43%) messages regarded coordination and social chat, which was not related to the task.

After this first part, two students encountered network problems (caused by their internet provider). They

managed to reconnect, but the connection was no more stable. Thus, they discussed for 49 minutes (49 messages), how to go on. Finally, they agreed to work on at the next day.

At the next day, they met again. They spent the first nine minutes of their interaction discussing the events of the previous day (19 messages) and how to re-enter the collaborative task (15 messages). After they reentered the brainstorming view, they continued to find and discuss concepts for half an hour. In this time, 41 messages (48%) concentrated on the task's topic, 23 (27%) messages were sent to clarify the assignment, and 21 (25%) messages were used for coordination.

One student mainly initiated the coordination messages. This student asked four times, whether or not the group would like to switch to the next phase (after 16, 17, 20, and 24 minutes). After the first two questions the group did not answer, but continued to discuss the concepts. After the third question, they agreed to go on, but then they started a discussion again and the requesting user did not move to the next phase. Finally, after the fourth question, the group moved on.

They spent 30 minutes working with the semantic network tool. Since they used this tool the first time, most of the communication (25 (35%) of 72 messages) regarded the use of the tool. They assisted one another to learn the functions that the network tool offered. 22 messages (31%) regarded coordination, when the users for instance split the control in drawing area between them. When the semantic network was almost finished, they started discussing the content again. They discovered that one important concept was missing and discussed how they could integrate it in the picture. At the end, they discussed whether the task was solved or not and if they had answered all aspects of the exercise (7 messages – 10%).

Regardless of the network problems, the group reflected some days later in the newsgroup on the tool use for doing the exercise. They stated that they could learn well during the discussion and that they had fun doing the collaborative exercise.

The other two groups acted in a comparable way, although they did not encounter any network problems. Considering the first uses of students of the FUB platform, we can identify several implications for future exercises with FUB:

First of all, the system can be used successfully to solve exercises in groups. All groups that used the system liked the way, how they could come together and have a group experience. This encourages us to continue offering collaborative exercises.

The student's solutions have been of high quality. We noticed that the students started to reflect on the individual facts and transferred these facts to their personal environment. Thus, we can say that the goal of

understanding facts in a larger context was reached. The discussion logs also reveal that the students started to explain and defend their understanding of the courses subjects – an ability, which as we think, is very important.

Several findings suggest improvements of our approach: Most important, we did overestimate the willingness of students to use CSCL systems on a voluntary basis. We are therefore considering to introduce some mandatory collaborative exercises.

From the current experience, all groups who did collaborative exercises agreed to do another collaborative exercise. Thus, it might be sufficient, if the first collaborative exercise is mandatory. This will require the students to have a look at the software and gain a first-time experience with collaborative exercises. We assume that a high percentage of the groups will then continue with perhaps voluntary collaborative exercises.

Another issue is the scheduling of sessions. This seemed to produce much overhead for the students. Having an integrated scheduling mechanism (such as a shared calendar tool for marking intended sessions) would ease the process of group formation and thus lower the initial effort for starting a collaborative exercise.

As mentioned before, several students asked for a software version for other operating systems than Microsoft Windows. We will therefore ship runtime versions of the platform for all major operating systems in the next term. We also gained positive experience with shipping a stand-alone tool (as one single executable that does not require any installation). This keeps us independent of different browsers, different Java installations or several plug-in installations. Using FUB only requires low bandwidth. However, downloading the FUB installation with approximately 2.65 MB may take a while. Shipping the platform on CD rather than offering it for download may also help to get FUB used by students who have a low bandwidth connection to the Internet.

## 6. Implications for the Design of Next Generation Learning Platforms (NGLP)

Our current prototypical implementation of the FUB system was aimed at gathering early experience with our approach. In this section, we discuss how collaborative exercises should be supported in a next generation learning platform, such as the next version of the FernUniversität learning platform 2003. FUB services that are planned to be integrated into the platform include:

- Registration, storage and retrieval of exercise representations;
- Exercise management services for creating and manipulating exercises, which support reuse and improvement of exercises;

- Learning group management services, which facilitate group building and scheduling of sessions;
- Collaborative learning services including basic session management, execution support for complex problem solving processes, translation services between tools of different phases within an exercise, and orientation and guidance services supporting students in complex learning processes;
- Learning management services including monitoring status of courses and exercises, and submitting results and discussing corrections.

While these services are partially back-end services to be used by application developers, we will also supply generic components that can be directly included by tool builders (e.g. learning process control panels, group building component). This will make the integration of collaborative exercises into courses easier. In addition, we will integrate our components with existing services of the FernUniversität virtual learning space, such as the single sign-on portal, the course administration, and the semi-automatic correction systems.

## 7. Related work

Related work can be categorized into three groups: collaborative learning environments, general CSCW environments, and learning platforms.

Collaborative learning environments usually support specific collaborative learning methods. Most closely related to our approach are problem-based learning environments (PBL) [7], such as CNB [5, 13], CSILE [14], Belvedere [18], and Web-SMILE [5]. Generally speaking, these PBL support systems have been developed to support collaborative PBL between homogenous learners of high schools or of universities. These systems focus mainly on supporting science inquiry. In addition, most of them are implemented based on the Web, so that a large number of users can construct a shared knowledge representation primarily in an asynchronous collaboration mode. To our knowledge, only the CROCODILE system [11] supports primarily synchronous collaboration. However, none of these systems support collaborative exercises.

General CSCW environments, such as Groove [4], BSCW [1], Lotus Notes [9], application sharing systems such as Netmeeting [12], and shared whiteboards, support synchronous and asynchronous forms of collaboration. However, they fail to support all the functionality required to support synchronous collaborative exercises. While it is possible to build a similar environment to ours in Groove or other groupware development environments, this has to our knowledge not been done.

Finally, current learning platform such as WebCT [19], Blackboard [2] and others primarily aim at asynchronous learning scenarios. Systems such as Interwise [6] or Lotus Notes Learning Space [8] support lecture broadcast and limited discussions, but do not offer dedicated support for collaborative synchronous exercises and group building. Only the L-3 cooperative learning environment [10] offers dedicated support for synchronous group learning (using the so-called PoC concept [21]). However, while L-3 did so far only support ad-hoc group building [22, 23] we are supporting longer-term and intentional group building and community support. Furthermore, our focus on integrating exercises into the FernUniversität's virtual learning environment led us to explicitly address the issue of semi-automatic evaluation/correction and to analyze how learning platforms should support collaborative exercises.

## 8. Conclusions

In this paper, we analyzed the main requirements of collaborative exercises in a distance learning university, which include support for preparation of exercises, learning group management, collaborative learning sessions, and learning management. Then we presented our approach to support such collaborative exercises in the FUB system by providing groupware tools for each phase of collaborative exercises. Especially important is the support of complex problem solving processes. We discussed the current implementation of FUB and reported about a trial use during this summer semester, indicating that our approach basically works – but needs further improvements.

Our approach exceeds related work in three ways: it provides dedicated support for mid-term collaborative exercises, it suggests necessary features of next generation learning platforms aiming at supporting this class of collaborative learning situation, and it reports about the results of a trial use.

Currently, we finish the first round of evaluation in the context of the operating systems course in the school of computer science. While our initial experience indicates some success, we plan a follow-up study with more and longer usage. Here, we plan to improve our system and use it in the winter term for a distributed systems course. In addition, we are currently working on integrating our approach into the FernUniversität's learning platform.

## References

- [1] Bentley, R.; Appelt, W.; Busbach, U.; Hinrichs, E.; Kerr, D.; Sikkil, K.; Trevor, J.; Woetzel, G.: "Basic Support for Cooperative Work on the World-Wide Web", in: *International Journal of Human-Computer Studies: Special issue on Innovative Applications of the World-Wide Web*, 1997.
- [2] <http://www.blackboard.com/>
- [3] Brunsmann, J., Homrighausen, A., Six, H.-W., Voss, J. Assignments in a Virtual University: The WebAssign-System. *Proceedings of the 19th World Conference on Open Learning and Distance Education*, Vienna/Austria, June 1999.
- [4] <http://www.groove.net/>
- [5] Guzdial, M., Hmelo, C., Hübscher, R., Nagel, K., Newstetter, W., Puntembakar, S., Shabo, A., Turns, J., and Kolodner J.L. "Integrating and Guiding Collaboration: Lessons learned in computer-supported collaboration learning research at Georgia Tech.", In: *Proceedings of CSCL'97*, 91-100. Toronto, Ontario, Canada, 1997.
- [6] <http://www.interwise.com/>
- [7] Koschmann, T., Kelson, A.C., Feltovich, P.J., Barrows, H.S. (1996). Computer-Supported Problem-Based Learning: A Principled Approach to the Use of Computers in Collaborative Learning. In T. Koschmann (Ed.), *CSCL: Theory and Practice of an Emerging Paradigm* (pp. 83-124). Mahwah, NJ: Lawrence Erlbaum Associates.
- [8] <http://www.lotus.com/home.nsf/welcome/learnspace>
- [9] <http://www.lotus.com/>
- [10] <http://www.l-3.de/en/index.html>
- [11] Miao, Y., Haake, J. M. Supporting Problem Based Learning by a Collaborative Virtual Environment: A Cooperative Hypermedia Approach. In: R. H. Sprague Jr. (Ed.): *Proceedings of the 34th Annual Hawaii International Conference on System Sciences (HICSS-34)*, Maui, Hawaii, USA, January 3-6, 2001, CD-ROM.
- [12] *Microsoft Corporation*: "NetMeeting Home", <http://www.microsoft.com/windows/netmeeting/>: 2001.
- [13] O'Neill, D. K. "The Collaboratory Notebook: A Networked Knowledge-Building Environment for Project Learning", In: T. Ottmann & I. Tomek (Eds.), *Educational Multimedia and Hypermedia*, pp. 416-423, VA: AACE, 1994.
- [14] Scardamalia, M., Bereiter, C., and Lamon, M. "The CSILE project: Trying to bring the classroom into World 3", In: K. McGilly (ed.), *Classroom lessons - Integrating cognitive theory and classroom practice*, pp. 201-228. Cambridge, MA: MIT Press, 1994.
- [15] Schuckmann, C., Kircher, L., Schuemmer, J., Haake, M.J. "Designing Object-Oriented Synchronous Groupware with COAST", In: *Proceedings of ACM CSCW'96*, pp. 30-38, 1996.
- [16] Schuckmann, C.; Schümmer, J.; Seitz, P.: "Modelling collaboration using shared objects", *Proc. of the Conference on Supporting Group Work*, 1999, 189-198.
- [17] Schümmer, T.; Schümmer, J.; Schuckmann, C.: "COAST - An Open Source Framework to Build Synchronous Groupware with Smalltalk", OpenCoast Development Group, 2001, available online at <http://www.opencoast.org/documentation/CoastSTBook.pdf>.
- [18] Suthers, D.D. Toth, E., and Weiner, A. "An Integrated Approach to Implementing Collaborative Inquiry in the Classroom", In: the *Proceedings of CSCL'97*, pp. 272-279, Toronto, December 10-14, 1997.
- [19] <http://niobe.fernuni-hagen.de/WebAssign/> (in German)
- [20] <http://www.webct.com/>
- [21] Wessner, M.; Pfister, H.: "Points of Cooperation: Integrating Cooperative Learning into Web-Based Courses", *Proceedings of the NTCL2000, The International Workshop for New Technologies for Collaborative Learning*, Hyogo, Japan, 2000.
- [22] Wessner, M.; Pfister, H.: "Group formation in computer-supported collaborative learning", *Proceedings of the 2001 International ACM SIGGROUP Conference on Supporting Group Work (GROUP'01)*, ACM Press: Boulder, CO, USA, 2001, 24-31.
- [23] Wessner, M., Haake, J. M., Tietze, D. (2002). An infrastructure for Collaborative Lifelong Learning. *Proceedings of the 35th Hawaii International Conference on System Science. HICSS 2002*, Hawaii, January 2002.