u2b00559: MARRYING ON-CAMPUS TEACHING TO DISTANCE TEACHING

Lukosch, Stephan; Roth, Jörg; Unger, Claus

University of Hagen, Department for Computer Science, 58084 Hagen, Germany
{Stephan.Lukosch, Joerg.Roth, Claus.Unger}@Fernuni-Hagen.de

Abstract Hyper-lectures support distance learning and traditional lecturing based learning. They can be used in asynchronous and synchronous, collaborative learning scenarios. Three scenarios and their implementation with DreamTeam, a general groupware development and runtime environment, are addressed.

1 Introduction

Traditionally, campus universities and distance teaching universities played different roles, attracted different audiences and followed different teaching paradigms. This situation has significantly changed during the past few years:

- campus universities and distance teaching universities are anxious to attract new audiences;
- distance teaching universities transcend time and space constraints by means of telecommunication and multimedia;
- campus universities distribute their lectures via satellite online to remote sites.

Differences between both kinds of universities are blurring, new concepts like Virtual Universities are emerging. In our contribution, we present an integrated approach for different kinds of teaching and learning.

Traditional on-campus teaching is based upon lectures, which beside oral presentations include slides, experiments, etc. Presentations and public online discussions are unique in the sense that nothing is recorded for playback or future use. In contrast, traditional distance teaching is based upon written courses, discussions are offline via telephone, e-mail, news groups, etc.

In our approach, we introduce the notion of hyper-lecture, which is a hypermedia document containing

- a set of slides;
- a text version of the lecture;
- relevant references to other learning materials, Web pages, etc.;
- textual, graphical, audio or video annotations.

Annotations can dynamically be added to the hyper-lecture, either individually by a student or publicly by, e.g., the lecturer. Annotations can be linked to slides, texts, graphics, formulas, etc.; they can be chosen from a variety of artifacts like texts, freehand sketches, diagrams, Web pages, audio clips, video clips, etc. Audio annotations provide a convenient and fast method to 'record' a real lecture.

Most of their time, students work asynchronously, i.e. independent of each other; occasionally they co-operate synchronously with other students, tutors and lecturers. Thus our hyper-lecture learning environment supports asynchronous as well as synchronous learning.

Via our approach, a complete hyper-lecture can gradually emerge from a traditional, recorded lecture, which is stepwise enriched by a slide show, textual documents, etc. Such a hyper-lecture can then be used for traditional lecturing, distributed online lecturing, distance teaching, information retrieval, etc. Depending on the learning scenario, the same hyper-lecture can be presented in different forms.

A first prototype for a hyper-lecture learning environment has been developed on our DreamTeam platform; it allows students and lecturers to co-operatively read, discuss and annotate hyper-lectures online or offline in a user-friendly CSCW environment. Online discussions are supported by a built-in audio conferencing component. DreamTeam and its applications are realised in Java; thus, beside some minor restrictions with regard to audio and video annotations, which are not yet fully supported in Java, our hyper-lecture environment is platform independent and does not contain any native code.

In the following, we sketch some example learning scenarios, briefly describe the implementation of our hyper-lecture learning environment with DreamTeam and finally mention some other relevant projects.

2 Learning Scenarios

2.1 Single Learner Environments

In an asynchronous learning environment, a student starts her private viewer in the DreamTeam environment and gets access to courses, slides, annotations and links. While a student may not change the original hyper-lecture and its links, she may add annotations; since an annotation may contain a link, a student can indirectly extend the link structure of a hyper-lecture as well. Annotations can either be permanent, i.e. integrated into the learning materials, or private, i.e. individually added and edited by the student. For distance students, the basic layout of the hyper-lecture is defined via a course document (fig. 1), while for traditional students, the hyper-lecture usually follows a slide show (fig. 2). Beside DreamTeam components, an asynchronous learning environment may contain non-DreamTeam components as well, like simulation environments, Applet viewers, etc.

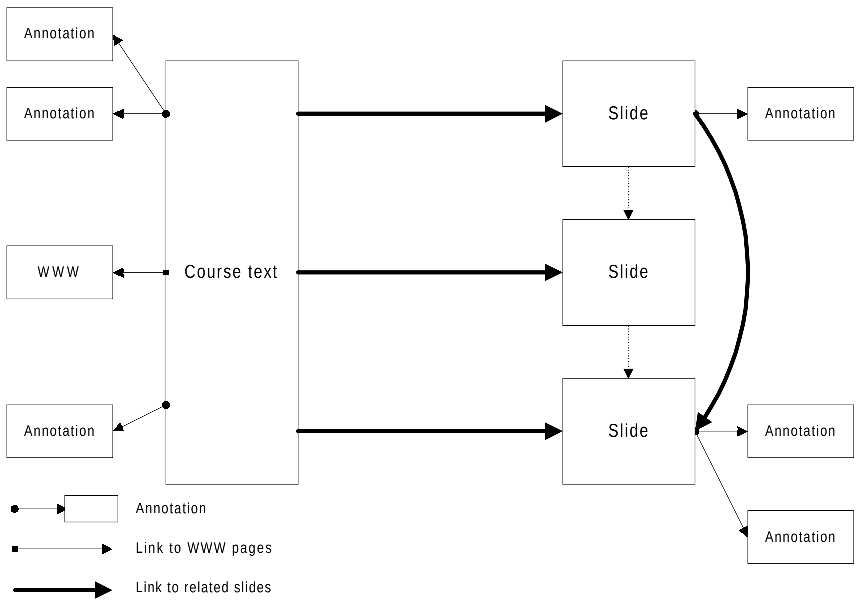


Fig. 1: Hyper-lecture for distance students

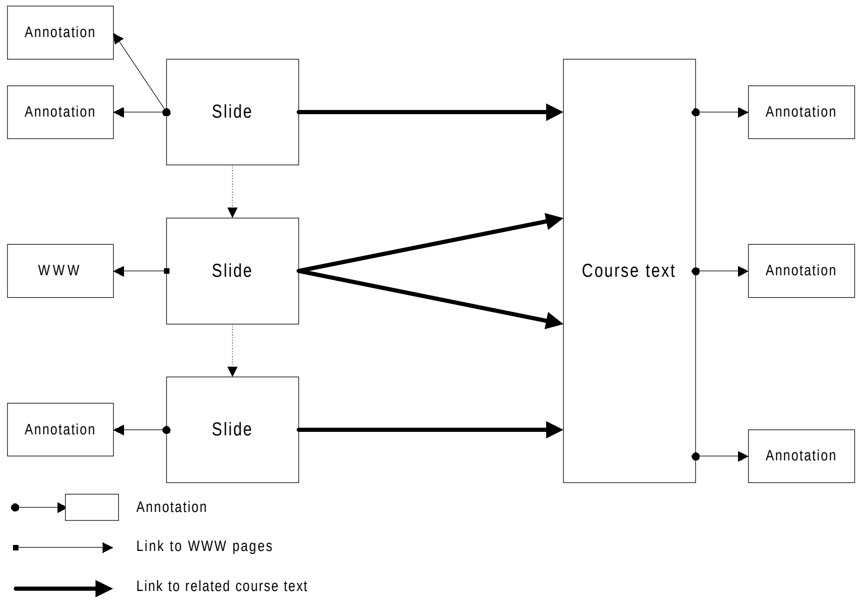


Fig. 2: Hyper-lecture for traditional students

2.2 Collaborative Learning Environments

In a collaborative learning environment (fig. 3), each participant starts a DreamTeam session, which may contain the following collaborative components:

- DreamView (a collaborative Browser and Annotator)
- DreamChat
- Portholes
- DreamAudio conference

Depending on their connection bandwidth and their hardware equipment students may have to restrict their individual environments. As minimal support for a session, each student has to run the DreamView browser and the DreamChat tool. If there is enough bandwidth available, Portholes of all participants as well as a moderated DreamAudio conference can be added to individual environments. Since Java so far does not sufficiently support video applications, Portholes are not fully integrated in the DreamTeam environments: at each site, a camera regularly takes a picture of the participant and stores it to a local file, which is then distributed to all session participants via DreamTeam.

During a session, all participants can collaboratively work on the hyper-lecture, can add annotations and use their individual, shared pointing devices (mouse pointers). One participant can dynamically take the role of a tutor or lecturer, run a slide show and control the audio conference. Other participants can join in via annotations as well as chat and audio contributions.

DreamView provides two *trace modes* which can individually be selected by each session participant:

1. in *individual mode*, the student does not follow the slide show, but individually navigates through the hyper-lecture, follows individual links, etc.;
2. in *trace mode*, the student follows the lecturer's slide show.

A *follow command* allows a student to switch from individual mode to trace mode.

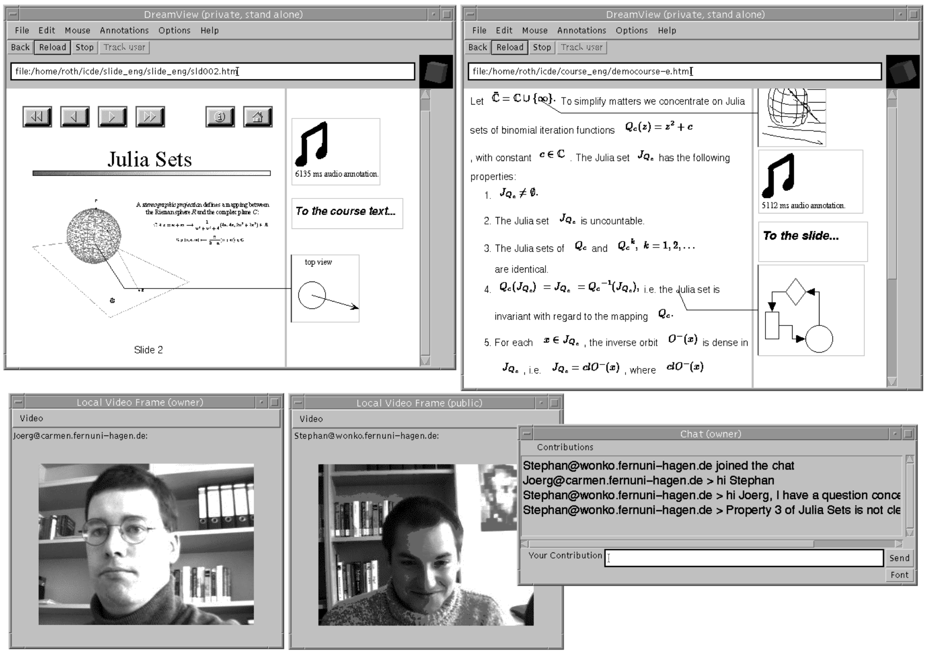
DreamView supports three kinds of annotations:

1. *global, public annotations* are permanent, integral parts of a hyper-lecture. They are included when the hyper-lecture is developed or can be added by the lecturer during a session. As soon as a new annotation is added, it becomes visible to all participants and is saved at all sites when the session terminates.
2. *local, public annotations* can be added by each session participant. They become visible to all session participants but, when the session terminates, are only saved with the owner's version of the hyper-lecture. During a session, each participant can add another participant's local annotation to his own annotations and let it survive the session in his version of the hyper-lecture.
3. *local, private annotations* are only visible to their owner and are only saved with their owner's version of the hyper-lecture.

DreamTeam supports the following platform independent types of annotations:

- WWW documents
- diagrams
- freehand sketches
- texts

DreamTeam, via the Java Native Interface (JNI), supports audio annotations under Windows 95 and Solaris. Via the Java Media Framework (JMF), externally recorded audio and video annotations in various formats like AIFF, AVI, MIDI, MPEG-1, Quicktime, Sun Audio, Video:H.263, Wave (see [JMF99]) can be attached and played.


DreamView (private, stand alone)
File Edit Mouse Annotations Options Help
Back Reload Stop Track user
file:/home/roth/icde/slide_eng/slide_eng/sld002.htm
Julia Sets
6135 ms audio annotation.
To the course text...
top view
Slide 2
DreamView (private, stand alone)
File Edit Mouse Annotations Options Help
Back Reload Stop Track user
file:/home/roth/icde/course_eng/democourse-e.htm
5112 ms audio annotation.
To the slide...
Local Video Frame (owner)
Video
Joerg@carmen.fernuni-hagen.de:
Local Video Frame (public)
Video
Stephan@wonko.fernuni-hagen.de:
Chat (owner)
Contributions
Your Contribution
Send
Font

A *stereographic projection* defines a mapping between the Riemann sphere R and the complex plane C:

Let $\bar{\mathbb{C}} = \mathbb{C} \cup \{\infty\}$. To simplify matters we concentrate on Julia sets of binomial iteration functions $Q_c(z) = z^2 + c$, with constant $c \in \mathbb{C}$. The Julia set J_{Q_c} has the following properties:

1. $J_{Q_c} \neq \emptyset$.
2. The Julia set J_{Q_c} is uncountable.
3. The Julia sets of Q_c and $Q_c{}^k,\ k = 1, 2, \ldots$ are identical.
4. $Q_c(J_{Q_c}) = J_{Q_c} = Q_c{}^{-1}(J_{Q_c})$, i.e. the Julia set is invariant with regard to the mapping Q_c.
5. For each $x \in J_{Q_c}$, the inverse orbit $O^-(x)$ is dense in J_{Q_c}, i.e. $J_{Q_c} = clO^-(x)$, where $clO^-(x)$

Stephan@wonko.fernuni-hagen.de joined the chat
Joerg@carmen.fernuni-hagen.de > hi Stephan
Stephan@wonko.fernuni-hagen.de > hi Joerg, I have a question conce
Stephan@wonko.fernuni-hagen.de > Property 3 of Julia Sets is not cle

Fig. 3: Collaborative Learning Environment

3 Creating Learning Environments with DreamTeam

When developing groupware for education scenarios, one has to take into account various students' needs and requirements:

- *User interfaces:* especially for remote students it should be as easy as possible to install and run a co-operative application. Different applications should provide consistent interfaces, at least for all aspects of installation and general collaboration.
- *System platforms:* most of our students are using PCs with a variety of different operating systems. A co-operative applications should make as few assumptions as possible about the system platform with regard to hardware requirements (screen resolution, system performance etc.) and software requirements (operating system version, installed software etc.). PCs tend to be unstable. A co-operative application should be tolerant against local breakdowns and should easily allow to restart a system and rejoin an existing session.
- *Network:* Most of our students are using the Internet for communication, mainly through dial-up modem connections, connecting them to the university either directly or via a service provider.

These connections are exposed to network delays, bandwidth degradations, as well as sudden, unexpected disconnection. Usually, students are off-line most of the time and their Internet addresses may change between consecutive dial-ins. It is the task of a rendezvous component [RU99] to find out which students are actually available for a meeting and how they actually can be reached via the Internet.

DreamTeam ([RU98a], [RU98b]) is a platform for synchronous collaboration which offers predefined solutions for the above requirements; it offers a variety of services for application developers as well as end-users. The DreamTeam environment allows the developer to develop co-operative applications like single user applications, without struggling with network details, synchronisation algorithms, etc. DreamTeam has been realised in Java [SUN99] without the need for any native code, i.e. it can be run on many platforms. It consists of a development environment, a runtime environment and a simulation environment. The development environment [Roth99] mainly consists of a huge Java class library which contains groupware specific problem solutions as building blocks. The runtime environment provides an infrastructure with special groupware facilities. A front-end on top of the runtime environment allows end-users to control and configure the system. Finally, collaborative applications can be tested in the simulation environment, which allows to simulate network characteristics on a single computer.

A common paradigm for shared workspaces is WYSIWIS (What You See Is What I See) [SBF+87]: the strict WYSIWIS policy enforces identical workspace displays on every participating system, whereas a relaxed WYSIWIS policy allows private areas and tolerates slight layout differences. Since DreamTeam is designed for heterogeneous environments, it supports relaxed WYSIWIS.

For an end-user it may be meaningful to use a shared application in a private environment as well, without having to learn a new user interface. With minor additional effort, DreamTeam applications can be developed for both, shared as well as private environments. An application can ‘ask’ the environment in which mode it is currently running and can enable or disable specific functions.

DreamTeam is based upon a completely decentralised architecture, thus there is no central server for storing session states. Though the decentralised architecture leads to more complex algorithms, it avoids performance bottlenecks and ensures higher reliability. Because of the fully replicated architecture of DreamTeam all component states have to be replicated as well.

To develop groupware applications in a more component-based manner, DreamTeam comes along with a component concept called *TeamComponents*. TeamComponents allow to efficiently develop synchronous groupware applications with the help of software components, each of which is responsible for a specific artefact, offers its own user interface and manages its own internal data. Using components helps to divide a software project into well defined parts. Well documented interfaces help to reduce integration efforts and improve software quality. TeamComponents can be developed separately from an application, thus the component developer and the application developer can be different people. A set of classes and interfaces guarantees the proper integration of components into an application, even if the corresponding component source codes are not available. Among other modules, we efficiently developed the annotation feature of the DreamView browser with the help of TeamComponents.

Usually, students can join collaborative sessions with a minimum of hardware equipment, whereas the teacher may use several hardware components simultaneously, e.g. cameras, special pointing devices like sketch pads, and audio devices. To support different, separate hardware devices at a single site, DreamTeam allows to distribute an application at runtime across a set of computers inside a conference room, e.g. on the teacher's site, one computer may be responsible for camera and audio input, another may handle the sketch pad and a third one may display the current slide show.

4 Related Work

DreamTeam has been developed for easily producing and running general, complex groupware applications. Collaborative learning environments are just being used to test the validity and completeness of the DreamTeam concept and prove its applicability for complex applications. From the big variety of learning environments, the following should be mentioned:

The project *Classroom 2000* [AAF+96] addresses traditional classroom teaching; it helps the lecturer and her students to thoroughly study and append a given lecture. It emphasises the use of LiveBoards and uses the WorldWideWeb and its browsers as presentation platform. It contains tools which support the pre-production, the in-class/live and the post-production phase of a lecture. [AAB+98] carefully reports about real experiences with Classroom 2000 and provides a long list of ambitious future extensions, including fine-grained media integration and synchronisation, searching in audio and video sources, automatic adaptation to a broad scale of student environments, sophisticated tools for annotations, automatic generation of summaries, etc.

Within the VirtUE (Virtual University for Europe) project three interrelated kinds of scenarios for virtual classrooms, virtual campuses and learning on demand have been developed; various prototypes of these scenarios have been sucessfully implemented at several universities, mainly based upon satellite video and audio conferences. An overview can be found in ([VDP97], [EuroPACE98]).

IFIP Working Group WG2.7 on User Interface Engineering developed a set of scenarios for various roles at a virtual university [IFIP98].

5 Future Work

Future versions of learning environments will benefit from general extensions of the DreamTeam environment; the following are of particular interest:

- Video and audio clips, interactive applets, online simulations and laboratories, etc., form important components of interactive courseware. To fuse all these media is a real challenge. It would further be cost and bandwith efficient if such interactive components can locally be loaded and run simultaneously at participating sites, centrally controlled by, e.g., the lecturer.
- To avoid online bandwidth bottlenecks, it should be possible to transfer big audio and video files offline before a session and automatically integrate them individually into the operating sytems environments of participating sites.
- Student environments heavily differ with regard to hardware and software equipment. An automatic, optimal adjustment of a given environment to the overall requirements of a specific session would be helpful.
- Automatic versioning and updating will help lecturers and students.

References

[AAB+98] Abowd G.D., Atkeson C.G., Brotherton J.A., Enqvist T., Gulley P., Lemon J.: *Investigating the capture, integration and access problem of ubiquitous computing in an educational setting*, in Proceedings of CHI, 440-447, May1998.

[AAF+96] Abowd G.D., Atkeson C., Feinstein A., Goolamabbas Y., Hmelo C., Register S., Sawhney N., Tani M.: *Classroom 2000: Enhancing Classroom Interaction and Review*, Technical Report GIT-GVU-96-21, GVU, September 1996.

[EuroPACE98] Implementation Models for a Virtual University for Europe: http://www.europace.be/oldies/info/ideas/models.htm

[IFIP98] IFIP Virtual University Case Study http://osiris.sund.ac.uk/~cs0gco/IFIP/ifip_index.htm

[JMF99] Java(TM) Media Framework API Home Page: http://www.javasoft.com/products/java-media/jmf/index.html

[Roth99] Roth, J.: *How to write shared applications with „DreamTeam“*, Technical Reference, FernUniversität Hagen, Feb. 1999

[RU98a] Roth J., Unger C.: *Dream Team - a synchronous CSCW environment for distance education*, Proc. of the ED-MEDIA / ED-TELECOM 98, Freiburg, Jun. 1998, 1185-1190

[RU98b] Roth J., Unger C.: *DreamTeam - a platform for synchronous collaborative applications*, in Th. Herrmann, K. Just-Hahn (eds): Groupware und organisatorische Innovation (D-CSCW'98), B. G. Teubner Stuttgart 1998, 153-165

[RU99] Roth, J., Unger C.: *Group Rendezyous in a Synchronous, Collaborative Environment*, in R. Steinmetz (ed): Kommunikation in Verteilten Systemen (KiVS'99), 11. ITG/VDE Fachtagung, Springer, March 1999, 114-127

[SBF+87] Stefik, M.; Bobrow, D.G.; Foster, G.; Lanning, S.; Tatar, D.: *WYSIWIS revised: early experiences with multiuser interfaces*, ACM Transactions on Office Information Systems, Vol. 5, No. 2, Apr. 1987, 147-167

[SUN99] *JavaSoft Home Page*, http://java.sun.com, 1999

[VDP97] Georges Van der Perre: *Higher education and lifelong learning: matching the needs of the knowledge society with the tools of the information society*, in proceedings of the 3rd International Conference on Information Technologies in Occupational safety and Health Information, Training and Education, 13-15 November 1996, Brussels, 80-91

Biographies

Stephan Lukosch and Jörg Roth are full-time researchers in the area of groupware development environments and are both working on their PhD theses. Claus Unger is full professor at the Department for Computer Science, his research area is interactive systems, including CSCW and mobile agents. He has chaired IFIP WG2.7 User Interface Engineering for six years and is presently Prorector for Continuing Education at the University of Hagen, and elected Secretary and Treasurer of the Association for Computing Machinery (ACM).