Seamless Integration of Group Communication into an Adaptive Online Exercise System

Hans Christian Liebig, Wolfgang Effelsberg
University of Mannheim, Germany

Abstract. Distance learners in traditional online exercise and tutoring systems often get stuck with questions for which they need the help of a tutor or colleague. Learning alone can also be frustrating. In our Communication And Tutoring System CATS we have integrated the possibility to dial up a tutor and/or to setup an immediate group communication with other distance learners using Internet videoconferencing technology. To find the appropriate partner, we have implemented a measurement algorithm that keeps track of the performance level of a learner by measuring the percentage of correct answers at the current level, the reliability with which the learner answers the questions and the time he/she takes. From these measures we derive a unified performance parameter that controls the presentation of the next set of questions. These are then generated dynamically by the exercise applet. The CATS system *automatically* selects the most appropriate communication partner(s) based on the exercises the learners are currently working on, and on their skill levels. We motivate this approach from a pedagogical point of view and present the architecture and implementation of the CATS system.

Keywords. Collaborative learning, online tutoring, distance learning, peer communication

Contact information.
Hans Christian Liebig
Praktische Informatik IV
University of Mannheim
L 15, 16
68131 Mannheim
Germany
cliebig@informatik.uni-mannheim.de
Phone: +49 621 181 2606
Fax: +49 621 181 2601

HANS CHRISTIAN LIEBIG, WOLFGANG EFFELSBERG

SEAMLESS INTEGRATION OF GROUP COMMUNICATION INTO AN ADAPTIVE ONLINE EXERCISE SYSTEM

Abstract. Distance learners in traditional online exercise and tutoring systems often get stuck with questions for which they need the help of a tutor or colleague. Learning alone can also be frustrating. In our Communication And Tutoring System CATS we have integrated the possibility to dial up a tutor and/or to setup an immediate group communication with other distance learners using Internet videoconferencing technology. To find the appropriate partner, we have implemented a measurement algorithm that keeps track of the performance level of a learner by measuring the percentage of correct answers at the current level, the reliability with which the learner answers the questions and the time he/she takes. From these measures we derive a unified performance parameter that controls the presentation of the next set of questions. These are then generated dynamically by the exercise applet. The CATS system *automatically* selects the most appropriate communication partner(s) based on the exercises the learners are currently working on, and on their skill levels. We motivate this approach from a pedagogical point of view and present the architecture and implementation of the CATS system.

1. INTRODUCTION

Exercises and tutorials in small groups are an important component of learning at all levels of education. The fact that the student has to do something himself/herself helps to deeply understand the material. It is also well known that communication with peers and with the tutor in small group settings is very helpful for the acquisition and internalization of knowledge (Bransford et al., 2000, Qin et al., 1995).

In distance learning scenarios, individual work by students is often supported with online exercise tools. The state of the art is online retrieval of exercises/problems from the teacher's Web server, individual work on those problems, and submission of the solution, either on paper or electronically to the server or by email, for manual evaluation by the teacher. Such exercise systems have several major drawbacks:

- All students receive the same exercises, independent of their individual knowledge and learning style.
- The manual grading of all the individual submissions is very time-consuming for the teacher.
- When the student has difficulties with a particular exercise, immediate, individual help is not provided which is a considerable drawback when compared with a traditional presence scenario.
- As Allen shows in (Allen, 1997, Allen 1977) there is no communication with peers, if the distance is to high. This is desirable from a pedagogical point of view, CATS might also help to increase the student's motivation to do the exercises with the help of communication.

CATS, the Communication and Tutoring System under development at the University of Mannheim, integrates solutions to these issues into an online tutoring system.

Related work in the literature has addressed some of these issues, but not all of them, and not in an integrated fashion. The integration of videoconferencing into an online tutoring system is addressed in (Martins Ferreira et. al. 2002) and (Liang et. al., 2001). Many learning management systems have a built-in exercise system, mostly based on simple multiple choice (Brunsmann et al., 1999, imc's CLIX-Campus, Blackboard) but they do not allow the student to start a videoconferencing session. A self-assessment system for a physics course is described in (Kurz & Hübner, 2001). This system diagnoses individual knowledge and skills at the transition from secondary to higher education but again has no videoconferencing facility.

We claim that an *automatic adaptation of the level of difficulty of such questions* is desirable in order to maintain the right level of challenge for the learner. We know from video games how much an increasing level of difficulty keeps users fascinated with their activity over a long time.

The history of computer-supported learning began in the 1950s with the first simple training systems. A much more advanced integration with artificial intelligence and an tutoring concept with a domain model, a student model and a tutor model is the basis of Intelligent Tutoring Systems (ITSs); for an overview the reader is referred to (Frasson, 1990). Training systems for symbolic calculation and problem generation are described in (Yibin, 1992 and Nykänen 2000). An automatic adaptation of multiple-choice questions is also described in (Sherman, 1996). Schulmeister addresses the importance for a learner to get feedback from the online system and to have an opportunity to somehow control the program from a psychological point of view (Schulmeister, 1997). He argues that control by the student leads to induced benefits such as the development of certain meta-cognitive skills. Also, artificial intelligence techniques alone are often unable to fully adapt the behavior of the system to the behavior of the learner, so he proposes a hybrid model with system feedback to the learner and some control on his/her side.

We also consider a feedback-based control loop to be a promising compromise between the behaviorist principle of exercising by drill-and-practice and the cognitivist principle of self-controlled learning.

These pedagogical and psychological thoughts have motivated us to integrate algorithms for the measurement of the proficiency level of the learner, feedback to the learner, and automatic adaptation of the level of difficulty of the problems into the Communication and Tutoring System CATS that we are developing at the University of Mannheim

The remainder of this paper is structured as follows. In Section 2 we present the pedagogical motivation for our approach. Section 3 describes the design of the CATS system. Section 4 explains the generation of adaptive exercise problems. Section 5 describes the implementation of the CATS System. Section 6 presents examples of exercise problems. Section 7 concludes the paper and gives an outlook on future work.

2. PEDAGOGICAL MOTIVATION FOR ONLINE GROUPWORK

2.1. The Role of Exercises in Traditional Teaching

Group exercises play an important role in traditional teaching at all levels of the educational system. They serve many purposes: the student has to become active himself/herself, the teacher or tutor gets a feedback on the current level of understanding of the students, and group work is motivating to most students: they enjoy meeting with other people and doing work together. Also, when results are turned in to the teacher, exercises are an important basis for grading individual performance.

Many theoretical models of human learning include a phase in which newly acquired knowledge has to be explained to others in order to be stabilized and to become part of a person's problem solving skills. For example, Mayes at al. include a "dialog phase" in their learning cycle (Mayes et al., 1994). Also, new research of peer reviewing among students shows that this type of interaction greatly improves the memorization of facts and skills, increases motivation and is well accepted by students (see for example Kern et al., 2002). We conclude that easy communication between a learner and his/her tutor and fellow learners, at any time, is a very desirable component of any exercise scenario; "lone wolves" in front of their PCs don't learn in an optimal fashion.

2.2. Exercises in Today's Distance Learning Scenarios

While in traditional settings group exercises work sufficiently well, this is not the case in distance learning scenarios. The current state-of-the-art is that the lecturer offers static exercises as homework that can be downloaded from the Web by the students. They do their work on their own and mail the solutions back to the teacher. In classical Open Universities this is often still done on paper. In slightly more modern settings, the students send back their solutions electronically, by email or by uploading them onto the teacher's Web server. Correcting and grading is again done manually.

In these very typical scenarios the students are isolated, they have no communication with other students involved in the exercises or with the teacher. They have no means of knowing about other students and their proficiency. Interpersonal communication is rare.

These problems are well known. Current research attempts to counter them by improved electronic exercise systems. These can be divided into two classes. The first class aims at accelerating the traditional method of feedback; these systems use an integrated exercise management system for rapid electronic distribution of traditional exercises and return of the students' solutions. The second class offers fully automated exercises for a limited type of problems. For example, most learning management systems contain a simple authoring system for multiple-choice questions; the students submit their answers to the multiple-choice questions via the Web to the LMS server and get an immediate feedback. In these systems the exercise (set of problems) is the same for all students, independent of individual performance. The second class provides better feedback to the students, but it does not solve the problem of lack of interpersonal communication while working on exercises.

2.3 Integrating Interpersonal Communication into the CATS Exercise System

CATS allows individualized problem generation to match the level of the student: the better the student performs, the harder the next question, generated by the software, will be. The student also gets immediate individual feedback.

Since the CATS server always knows who is working on what and also keeps track of the current levels of performance it can *automatically* form groups of students that might profit from working together. If a student wishes to contact a competent person about the topic on which he/she is currently working or just enjoys communicating with other learners he/she can hit a button, and CATS will automatically set up a videoconference with the right person(s).

3. DESIGN OF CATS

CATS is a Web-based system that supports online exercises for distance learners. It is designed to be used in various subject areas. The information CATS derives about the proficiency level of the learners is used for different purposes: to give the learners feedback on their current ranking in the class, to automatically set up videoconferences with other students studying the same subject at a similar level of proficiency, and to automatically generate the next set of problems for each student. Knowing the proficiency level of each student also gives important feedback to the teacher.

The CATS system is based on a client -/server architecture, implemented with Web technology. An overview is shown in Figure 1, with the client on the right and the server on the left.

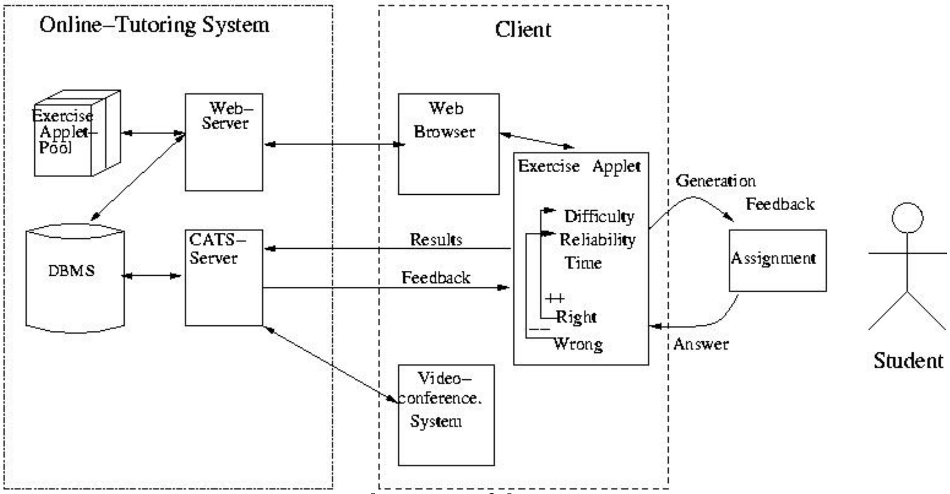

Figure 1: Architecture of the CATS system

3.1 The CATS Client

The CATS client provides the user interface for the distance learner. Figure 2 shows a screenshot of the client user interface. On the left side of the client screen, a navigation bar is provided to allow a direct jump to specific exercises. The main exercise frame in the middle of the screen contains the Java applet with the dynamically created exercise. The student has the possibility to train a topic as often as he/she likes. In each round, a new exercise is offered, with the proficiency level adapted to the student's performance. When the student completes an exercise he/she transmits the results back to the CATS server.

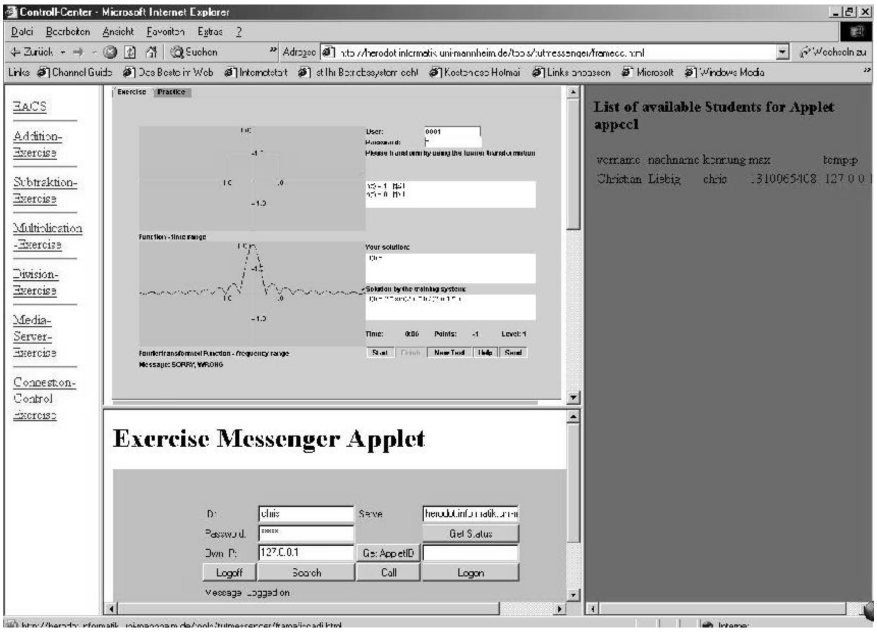

Figure 2: User interface of the CATS Client

Three performance parameters are evaluated by the applet: *proficiency* which indicates up to what complexity the student is able to solve the problems, *reliability* which indicates how reliable he/she is with his solutions, and *time*, i.e., how fast he/she can solve the problem. These parameters are also transmitted back to the exercise server. Periodically, CATS calculates a new ranking list for each group of exercises.

On the right side of the client screen, a ranking list of those students is shown that are currently working on the same exercise. A "callto" URL is integrated to call the selected student directly with MS Netmeeting, another H.323-based tool or a SIP-based tool (Session Initiation Protocol). When the student hits this button an IP videoconferencing session is automatically initiated. Thus it is easy to communicate with another student or a tutor who is familiar with the topic and has the right level of competence.

3.1.1 INDICATORS FOR LEARNING PERFORMANCE

In the CATS system we have implemented the following Key Performance Indicators (KPIs):

- **Proficiency.** Proficiency indicates up to which difficulty level the student is able to solve the exercise problems. CATS has no limit of proficiency levels.
- **Reliability.** Reliability (sometimes called "confidence level") is a indicator how consistent the student is in performing at a given proficiency level. It is the variance of the observed proficiency.
- **Time.** The time a student takes to solve a problem is obviously another important indicator, in milliseconds.

To calculate a unified level of performance we need to combine these parameters into one value. We use the following formula:

$$Performance = \frac{Proficiency}{Time} * Reliability * 1000$$

This value is stored in the database and used to create the ranking lists of the students, define the set of group members for the videoconferencing tool and dynamically select the next set of problems to be presented to the student.

4. GENERATION OF ADAPTED EXERCISE PROBLEMS

There are two basic ways to implement adaptive exercises. The first is to create a pool of questions, assign a level of difficulty to each and select appropriate questions at runtime. The second is to generate questions dynamically with a program.

4.1 Selecting Questions From a Database

Large-scale assessment tests make it necessary to provide a large number of questions and select them automatically from a database. Huang describes a method to create a pool of tests and use an algorithm he calls CBAT-2, an adaptive algorithm to evaluate the difficulty level (Huang, 1996). Questions in CBAT-2 are indexed by two parameters: a difficulty level and a guessing factor. The guessing factor describes the probability with which a student can guess the correct answer. The question selection procedure consists two main steps: First the content area is selected at random. Then a question among those associated with the content area is chosen. This is done based on the amount of information that a question may provide for the student's assessment, i.e., the appropriate level of difficulty. This algorithm needs a large pool of questions to avoid guessing and cheating.

The learning context of CATS is different from Huang's. We could also create a database of questions for which automatic evaluation by the server is possible. Examples of such types of questions are:

- **Multiple choice quiz with one correct answer.** This is the most obvious type of question for automatic analysis.
- **Multiple choice quiz with a variable number of correct answers.** This is very similar, but the question can have more than one correct answer. The student has to select the answers he/she thinks to be true; all other alternatives must remain unchecked. The scoring is not obvious. An example could be: for each alternative answer correctly checked or unchecked the student gets one points, for incorrect checkmarks he/she loses one point.
- **Fill-In statement**. This is a type of quiz where the question is formulated as a statement with one word missing. This word has to be provided by the student. The student's answer is matched with a set of provided keywords that are considered correct (and score fully) and optionally against a second set of ``almost correct'' keywords that only give half of the points.
- **A mathematical problem** for which that results in a numerical value. The student's response is directly matched against the preset correct result. To allow for the automatic analysis of more complex problems, several alternative results may be defined and assessed that arise from small and predictable mistakes made by the students.

- **An ordering problem.** For example, the quiz presents several steps of a certain process that are randomly shuffled. The student has to properly sort these steps. Assigning scores is easy: the correct answer gives the maximum amount of possible points, one error (switching two steps or putting one step in the wrong position) gives half these points, etc.

Many other types of automatically gradable questions are possible.

When creating the database we define fine-grain categories for the knowledge areas of the courses and label all questions accordingly. We also manually assign a level of difficulty to each question/problem. When a learner works on a specific category, the next set of questions to be presented to him/her by CATS will be selected from the current category, at the level of performance computed with the formula above. A generic Java applet will be used to handle the retrieval of the question from the database, present it to the learner, record his/her performance parameters and communicate the results and performance back to the CATs server.

A major advantage of the database approach is that it can be applied easily to all kinds of content domains. Also, a simple editor can be written to help teachers from all subject domains to create questions. It could be similar to the editor provided with the WILD tools for in-lecture quizzes (Scheele et al, 2002).

The database approach has not yet been implemented in CATS.

4.2 Dynamically Generating Exercises

The concept we use is based on the item response theory (IRT) (Wainer, 1990). The IRT uses three parameters to characterize each item: Difficulty level, discrimination factor and guessing factor. The difficulty level describes how difficult the question is. The discriminatory factor describes how well the question can discriminate students of different proficiency. The guessing factor is the probability that a student can answer the question correctly by guessing. The parameters are necessary to calibrate the evaluation statistic, especially for multiple choice questions. It would be an improvement to develop an exercise generator capable of creating an unlimited number of problems for each proficiency level.

In our system proficiency is the most important parameter. That factor will be calculated by the model the developer of an applet uses. What we found out is that some of the models use a general principle. That means that they are often not very difficult, and on the other hand more sophisticated models use the ideas of the basic ones.

In the first step such a basic algorithm will be assigned to a low difficulty level. The next steps differs for each algorithm. If the algorithm is parameterized the influence of each parameter on the complexity of the solution must be evaluated. In many cases the developer will find a linear, squared or exponential influence. In that relation he needs to calculate the difficulty level as a degree of the proficiency. If the solution of the learner is correct he will go up to the next higher level for the next set of problems. With some exercises a new algorithm or model will be tested.

We note that in the psychological literature, the question whether the learner or the system should select the level of difficulty for the next set of problems is discussed controversially (Schulmeister, 1997, p. 153). In CATS we allow the learner to decide; the system just offers him/her different levels. It also allows him/her to indicate a preference for the style of the exercises (textural, graphical or analytical); of course that requires that the problem applet implements these different styles.

5. IMPLEMENTATION

Our current implementation includes the CATS Server, which will be used together with some template applets as a framework, independently from the subject and the CATS clients. The number of clients increases every month.

5.1 Implementation of the CATS Server

The CATS server runs on a Linux system (SuSE 8.0), with the relational DBMS Postgres. We use a simple table structure for the relations *students*, *results*, *groups*, *logins*. The communication server and the exercise server are Java applications, implemented with Sun's Java Version 1.4.1. They use TCP/IP for communication with the exercise applets.

For the Web-based administration interface, we use the Apache web server with a PHP module. The registration processes, performance analysis, ranking lists and help lists are realized with PHP. The main components of the CATS server are shown in Figure 1. For storing the exercises and results, the user data, group information, etc., we use the relational DBMS Postgres. A simple schema represents the entities *students*, a*pplets, results, groups and logins.*

The *exercise server* is a Java application that uses jdbc to access the database. Communication with the applets is done over a special serializable object called "results" which has a communication method implemented.

The *conference server* is also a Java application that uses jdbc for database access and for the communication processes within the CATS server as well as for peer-to-peer communication.

The *administration tool* is a PHP-based script that allows to administrate CATS. Available functions include a registration process, ranking lists, help lists, group-matching, etc.

To set up a videoconference with a *group* of co-learners rather than a single person, CATS can use two possibilities: With an integrated H.323-based MCU (Multicasting Unit) we open a separate communication "room" for every group. Each instance of an H.323 program at a student site then connects to that MCU. The second techniques is to initiate a videoconferencing tool that runs over multicast-IP (e.g., JMF-Studio, vic, rat), but that requires the availability of multicast IP at all participating sites.

We are currently preparing a detailed user study to investigate the acceptance of the CATS services by the students, to improve the user interface of the client based on the students' feedback, and to see what the effect of adaptive question generation and automatic setup of videoconferences on learning efficiency is.

5.2 Implementation of the CATS Client

The exercise applets are Java-based. Thus they can take advantage of the full computational power of a programming language. During the development process each problem is analyzed to identify what makes the exercise "difficult" or "easy". In the domain of computer science, amazingly often, it is possible to adjust the level of difficulty with a simple parameter.

All the exercises are session-based. That the applets will start with the lowest difficulty level and go up to the highest level the student can solve. During this process the *reliability* parameter is evaluated (+1 for every solution right and –1 for every solution wrong). In addition, the time the student needs to solve the problem is recorded.

After the assignment is completed the student can ask CATS to present the correct ("official") solution. In many cases this solution will be shown in a graphical or even animated way.

6. EXAMPLES

A first example we have implemented in CATS is from the mathematical domain: the Fourier transform. In the first round the student is asked to manually transform a randomised, parameterised simple cosine function from the time domain into the frequency domain. If his/her answer is correct the next function to be transformed will be more complicated. At the third level of difficulty the student must know the general rules for the transformation of combined functions (e.g., f*g).

Another example would be from the domain of tax law: The first question could be a simple calculation between the different kinds of taxes and the limited compensation between them. In the highest level, a exercise can be created with international tax law relations. The tax cases, at different levels of complexity, are known, but the values will be randomly selected.

7. CONCLUSIONS AND OUTLOOK

We have presented CATS, a Web-based communication and tutoring system under development at the University of Mannheim. The features that distinguish CATS from similar systems are:

- Exercises are generated dynamically by the Java applet, depending on the level of performance of the learner.
- The server keeps track of the current activities of all online students and of their current level of performance. It maintains ranking lists.
- The client software allows to establish a videoconference with a tutor or other learners in an integrated fashion. It shows a list of other "relevant" learners and allows to contact them by audio/video with the click of a button.

We believe that such a system improves the learning success through instantaneous help when a learner runs into a problem, and trough increased motivation by contact to other learners.

A first prototype of the CATS system is operational and currently in the debugging phase. We are planning to gain practical experience with it in two virtual university projects: ULI (www.uli-campus.de) and POLITIKON (www.politikon.org).

8. REFERENCES

Allen, T.J. (1977). Managing the Flow of Technology.MIT Press, Cambridge, 1977.

Allen, T.J. (1997). Architecture and Communication: Among Product Development Engineers, Technical Report 165-97, MIT, Cambridge, September 1997.

Blackboard Inc. Washington D.C. http://www.blackboard.com, 2002. Last visit: 04.12.2002

Bransford, J., D., Brown, A. L., Cocking, R. R. (2000). How People Learn. Brain, Mind, Experience and School. National Academy Press, Washington D.C., 2000.

Brunsmann, J, Voss, J, Homrighausen, A, Six, H.-W. (1999). Assignments in a Virtual University – The Webassign System, Proceedings of the 19th World Conference on Open Learning and Distance Education, ICDE, Vienna, Austria.

FRASSON, C Gauthier, G. (1990), Intelligent Tutoring Systems, Ablex Publishing Corporation, 1990
Im-c GmbH. http://im-c.de. Last visit: 04.12.2002
Kern, V. M., Pernigotti, J. M., Calegaro, M. M., Bento, M. (2002). Peer review in engineering education: spee ding up learning, looking for a paradigm shift. Proc. 7th International Conf. On Engineering and Technology Education – INTERTECH 2002, Santos
Kurz, G., Hübner, H. (2001). Electronic interactive self assessment in the introductory physics course. Proc. 4th International Conf. on New Educational Environments, Lugano, 2001, Flückiger, F., Jutz, C., Schulz, P, Cantoni (eds.), pp. L 1.1-19 – 1.1-22.
Liang, A.H., Ziarko, W. Maguire, B. (2001) The Application of a Distance Learning Algorithm in Web-Based Course Delivery, Lecture Notes in Artificial Intelligence, Volume 2005, pp. 338
Martins Ferreira, J.M., Gustavo, R.C. Alves, Costa, Ricardo, Hine, Nick. (2002), Collaborative Learning in a Web-Accessible Workbench, Springer Lecture Notes in Computer Science, Volume 2440, pp. 0025
Mayes, T., Thomson, A., Mason, R., Coventry, L. (1994). Lear ning through Telematics: A learning framework for telecommunication applications in higher education. Technical Report, British Telecom, Martlesham Heath.
NYKÄNEN, O, 2000 A Framework Generating Non-Trivial Interactive Mathematical Exercises in the Web: Dynamic Exercises, World Conference on Educational Multimedia, Hypermedia and Telecommunications, AACE, 2000, p. 1473-1474
Qin, Z, Johnson, D.W., and Johnson, R.T. (1995). Cooperative Versus Competitive Efforts and Problem Solving, Review of Educational Research, Vol. 65, No. 2, 1995, pp 129-143.
SCHEELE, N., MAUVE, M., EFELSBERG, W., WESSELS, A., HORZ, H., FRIES, S.: The Interactive Lecture. A New Teaching Paradigm Based on Ubiquitous Computing, 2002. University of Mannheim, submitted for publication.
SCHULMEISTER, R., 1997, Grundlagen hypermedialer Lernsysteme, 2. Edition, R. Olde nburg Verlag, München, Wien, pp. 153-161
SHERMAN X. HUANG, (1996), A Content-Balanced Adaptive Testing Algorithm for Computer-Based Training Systems, Proceedings of the Third International Conference, ITS '96, 1996, Frasson, Gauthier, Lesgold, Lecture Notes in Computer Science, Springer, Berlin Heidelberg, New York, Vol. 1086, pp.306-314
WAINER, H, (1990), Computerized Adaptive Testing: A Primer, Lawrence Erlbaum Associates, Inc., Publisher
YIBIN, Mao, Jianxiang, Lin, (1992), Proceedings of the Second International Conference, ITS '92, 1992, Frasson, Gauthier, Lesgold, Lecture Notes in Computer Science, Springer, Berlin Heidelberg, New York, Vol. 608, pp.132-139

9. AFFILIATIONS

Hans Christian Liebig and Wolfgang Effelsberg are with Praktische Informatik 4, Institut für Informatik, University of Mannheim, Germany, www.informatik.uni-mannheim.de/pi4