Seamless Integration of Group Communication into an Adaptive Online Exercise System

H. C. Liebig, W. Effelsberg
{cliebig,effelsberg}@informatik.uni-mannheim.de
Praktische Informatik IV, Institut für Informatik
L15,16
University of Mannheim
68131 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 immediate group communication with other distance learners using Internet videoconferencing technology. To find an 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, 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 CATS. In this paper we explain our pedagogical approach 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 a student has to do something himself/herself helps to better 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 ([4], [16], [19], [13]).

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 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.
- If the student has difficulties with a particular exercise, no immediate, individual help is provided, which is a considerable drawback when compared with a traditional presence scenario.
- As Allen shows in ([1], [2]) there is no communication with peers if the distance is too far. However, 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. The basis for all our algorithms is a continuous measurement of the performance of the learner. We use the measurement results in different ways: to adapt the exercise problems to the proficiency level of the learner, to create virtual learning groups with the same proficiency level, and to integrate seamless communication with a videoconferencing facility.

Our key idea is to *automatically create online communities of learners at the same level of proficiency*. Related work in the literature has addressed some but not all of these issues, and not in an integrated fashion.

Early work on automated computer-supported exercises began in the 1950s with the first simple training systems. A much more advanced integration with artificial intelligence and a tutoring concept with a domain, student and tutor model respectively is the basis of Intelligent Tutoring Systems (ITSs); for an overview the reader is referred to [6]. Training systems for symbolic calculation and problem generation are described in [12] and [15]. An automatic adaptation of multiple-choice questions is described in [18]. Schulmeister addresses the psychological importance of feedback to the learner from the online system and the opportunity to somehow control the program[17]. He argues that control by the student leads to induced benefits such as the development of certain meta-cognitive skills. Also, since artificial intelligence techniques alone are often unable to fully adapt the behavior of the system to that of the learner, he proposes a hybrid model with system feedback to the learner as well as 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 insights 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. We know from our experiments with video games how much an increasing level of difficulty keeps users fascinated with their activity over a long time.

Let us now consider related work at the systems level. Whereas many of today's learning management systems (LMS) have a built-in exercise system, mostly based on simple multiple choice questions (WebAssign[5], Im-c's CLIX-Campus [10], Blackboard [3]) they do not allow the student to easily start a videoconferencing session. A self-assessment system for a physics course is described in [8]. This system diagnoses individual knowledge and skills at the transition from secondary to higher education but again offers no videoconferencing facility. The integration of videoconferencing into an online tutoring system is addressed in [7] and [11], but in these projects the tutoring system is not adaptive nor is performance of the learners measured.

Our main contribution in this paper is the online measurement of the proficiency level of the learners, and the *automatic* creation of an online videoconferencing community of learners at the same level of proficiency. This reduces leaner isolation and enhances motivation and peer feedback.

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 and the integration of the videoconferencing facility. Section 4 explains the generation of adaptive exercise problems. Section 5 describes the implementation of the CATS System. Section 6 presents an example of an exercise problem. 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 feedback on a students current level of understanding, 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 become stabilized and thus part of a person's problem-solving skills. For example, Mayes et al. include a ´´dialog phase´´ in their learning cycle [20], or Nonakas "Externalization-Process" [9]. Also, new research on 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 [21]). 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 online exercise scenario; "lone wolves" in front of their PCs don't learn in an optimal fashion. As a consequence, we address the automatic creation of online peer communities in our work in contrast to the help desk approach ("ask the expert") taken in other research projects.

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, and do not communicate 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 means of 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 LMSs 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 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 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 who might benefit from working together. If a student wishes to contact other competent persons 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 peers.

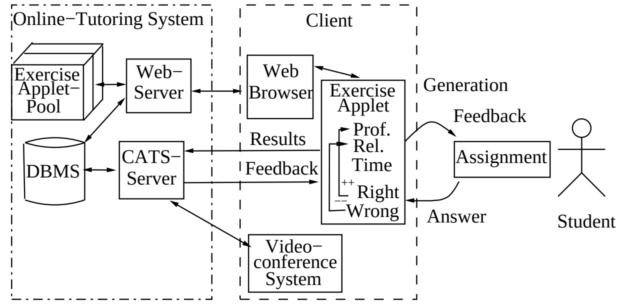

Figure 1: Architecture of the CATS system

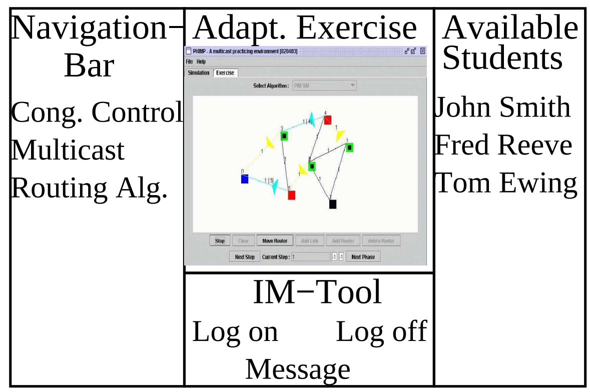

Figure 2: User interface of the CATS Client

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 and implemented with Web technology. An overview is shown in Figure 1, with the client on the right and the server on the left.

3.1. The CATS Client

The CATS client provides the user interface to the distance learner. Figure 2 shows a screenshot of the client. 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 practice 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 has completed an exercise, he/she transmits the results back to the CATS server.

Dependent on meassured performance indicators, CATS periodically calculates a new student ranking list for each group of exercises.

On the right side of the client screen, a ranking list of those students is shown who 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. CATS fully integrates Audio-/Video Communication with seamless group and peer communication without a media break between computer and telephony.

3.2. Indicators of Learning Performance

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

- Proficiency. Proficiency indicates the level of difficulty up to which the student is able to solve the exercise problems. CATS has no limit to proficiency levels.

- Reliability. Reliability (sometimes called "confidence level") is a indicator of 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, measured in milliseconds.

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

$$\text{Performance} = \frac{\text{Proficiency}}{\text{Time}} \cdot \text{Reliability} \cdot 1000$$

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 one 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 [18]. 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 of two main steps: First the content area is selected at random. Then a question from 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 can 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 are true; all other alternatives must remain unchecked. The scoring is not obvious. An example might be: for each alternative answer correctly checked or unchecked the student gets one point, for incorrect checkmarks he/she loses one point.

- Fill-In-the-blank statement. This is a type of quiz where the question is formulated as a statement with one word missing. This word has to be filled in by the student. The student's answer is matched against a set of provided keywords that are considered correct (and score fully) and optionally against a second set of "almost correct" keywords that yields only half of the points.

- A mathematical problem for which the answer results in a numerical value. The student's response is directly matched against the correct result. To allow for the automatic analysis of more complex problems, several alternative results may be defined and assessed that arise from predictable mistakes made by the students.

- An ordering problem. For example, the quiz presents several steps in a certain process that have been randomly shuffled. The student has to properly sort these steps. Assigning scores is easy: the correct answer yields the maximum number of possible points, one error (switching two steps or putting one step in the wrong position) yields 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 is 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 our WILD tools for in-lecture quizzes[14].

The database approach is currently under implementation in CATS.

4.2. Dynamically Generating Questions

The concept to dynamically generate exercises is based on the item response theory (IRT) [22]. 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 discrimination 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 statistics, 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 is calculated by the model the developer of an applet uses. We found out 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 is 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 proceed to the next higher level for the next set of problems. In some exercises a new algorithm or model will be tested. If the solution of the learner is not correct, CATS provides a solution with an explanation (in some exercises as an animation) and a new question with the same proficiency level.

We note that the psychological literature discusses the question of whether the learner or the system should select the level of difficulty for the next set of problems controversially [17][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 Java applet implement these different styles.

5. Implementation

Our current implementation includes the CATS Server, which will be used together with some template applets as a framework, independent of the subject, and the CATS clients.

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 and logins. The communication server and the exercise server are Java applications, implemented with Sun's Java Version 1.4.1. They use TCP/IP to communicate 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.

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. 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. With the high bandwidth of the Internet, available to our students today (768 kbit/s to 100 MBit/s) the video quality is usually very good.

5.2. Implementation of the CATS Client

The exercise applets are Java-based. Thus they can take advantage of the full computational power of the 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. The applets will start at the lowest difficulty level and move up to the highest level of problems the student can solve. During this process the reliability parameter is evaluated (+1 for every right solution and -1 for every wrong solution). 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 solution. In many cases this solution will be shown in a graphical or even animated way.

5.3. Status of the CATS System

Today exists a prototype of CATS with nine different exercises in three subjects (multimedia technology, computer networks and political science). The videoconference facility is tested with H323 and SIP. We haven planned a detailed evaluation during the next winter term 2003/04 in the lecture “multimedia technology”. Main objectives in the evaluation is the acceptance and the learning effectiveness of the students, who are using CATS.

6. Example

We use a simple example to illustrate the preparation of questions by the teacher and the use of the system by students. The idea is to produce a simple trainer for fractional arithmetic. The teacher has to perform the following tasks:

1. Define the learning objectives
   The learner should learn simple calculations with fractions, i.e., fractional calculations with addition, subtraction, multiplication and division.

2. Design the exercise problems at different levels of complexity
   The teacher designs a template of fractional calculations. At runtime the actual measurements to be used in the calculation will be generated automatically as shown below:

$$\frac{30}{5} = ? \tag{1}$$

$$\frac{3+17}{2\cdot 5} = ? \tag{2}$$

$$\frac{4\cdot(3+6)}{2\cdot(3+5)} = ? \tag{3}$$

$$\frac{4\cdot(5+7)}{3\cdot(12+2)} + \frac{6\cdot(8+4)}{7\cdot(9+5)} = ? \tag{4}$$

$$\frac{\frac{(12+45)}{7}}{6} + \frac{13+7}{12+9} = ? \tag{5}$$

3. Assign an integer value to each level of complexity
   In our example (1) stands for a complexity level of 1, (2) a complexity level of 2, etc.

4. Implement the JAVA applet
   The teacher implements the JAVA applet that presents the problems to the student.

5. Integrate the JAVA Class “results” into the applet
   This class is part of the CATS framework. It allows to upload the results of the exercises to the CATS server, together with the measured performance parameters of the student. The class is subject-independent.

6. Register the applet
   To open up the applet to the students it is necessary to register the applet in CATS. The teacher uses the CATS administration tool to register the new applet on the server side.

Bob and Alice are two students taking a WWW-based math course. Bob is currently in New York and Alice is in Paris. Bob and Alice register in CATS as students in the math class. Now they can train computations with fractions. Let us assume that Bob is able to solve problems up to level 3 without any problems. But to reach level 4, he must understand how to handle multiple fractions. Alice is a better student and knows how to handle those; she is currently online. Bob clicks on her name on the right side of the user-screen and is connected via the videoconference tool to Alice. Alice now explains to Bob how to do multiple fractions. Both students benefit from the cooperation: Bob gains knowledge, and Alice deepens her understanding (“learning by teaching“).

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 a learner to establish a videoconference with a tutor or other learners in an integrated fashion. It shows a list of other “relevant” learners and allows one 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 through increased motivation by contact to peer 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).

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.

Acknowledgments

CATS is part of the project “Universitärer Lehrverbund Informatik (ULI)”, which is financed by the Federal Ministry of Education and Research (Germany).

References

[1] Thomas John Allen. *Managing the flow of technology*. MIT Press, Cambridge, 1977.

[2] Allen, Thomas John. Architecture and Communication: Among Product Development Engineers. Technical Report 165-97, MIT, Cambridge, September 1997.

[3] Blackboard Inc. Washington D.C. http://www.blackboard.com, 2002. Last visit: 20.05.2003.

[4] Bransford, J.D., Brown, A.L., and Cocking, R.R. *How People Learn. Brain, Mind, Experience and School*. National Academic Press, Washington D.C., 2000.

[5] Brunsmann, J., Voss, J., Homrighausen, A., and Six, H.-W. Assignments in a Virtual University -The Webassign System-. In *Proceedings of the 19th World Conference on Open Learning and Distance Education ICDE*, Viena, Austria, 1999.

[6] Claude Frasson and Gilles Gauthier. *Intelligent Tutoring Systems*. Ablex Publishing Corporation, 1. edition, 1990.

[7] Ferreira Martins and Gustavo J.M. and Alves, R.C. and Ricardo Costa and Nick Hine. Collaborative Learning in a Web-Accessible Workbench. In *LNCS*, volume 2440, Berlin, Heidelberg, New York, Barcelona, Hong Kong, London, Milan, Paris, Tokyo, 2002. Springer.

[8] Günther Kurz and Heide Hübner. Electronic Interactive Self Assessment in the Introductory Physics Course. In *Proceedings of the 4th International Conference on New Educational Enviroment*, page 1.1/19, Lugano, Switzerland, 2002. SUSPI.

[9] Ikujiro Nonaka and Hirotaka Takeuchi. *Die Organisation des Wissens*. Campus-Verlag, Frankfurt am Main, 1997.

[10] im-c GmbH. http://im-c.de. Last visit: 20.05.2003.

[11] Liang A.H. and Maguire, B. Ziarko, W. The Application of a Distance Learning Algorithm in Web-Based Course Delivery. In *Lecture Notes in Artifi cial Intelligence*, volume 2005, page 338, Berlin, Heidelberg, New York, Barcelona, Hong Kong, London, Milan, Paris, Tokyo, 2001. Springer.

[12] Mao Yibin and Lin Jianxiang. Intelligent Tutoring System for Symbolic Calculation. In *Proceedings of the Second International Conference ITS '92*, pages 132–139, Berlin, Heidelberg, 1992. Springer.

[13] Mitsuru Ikeda, Shogo Go, and Riichiro Mizoguchi. Opportunistic Group Formation -A Theory for Intelligent Support in Collaborative Learning-. In *Proceedings of AIED-97*, pages 167–174, Kobe, Japan, 1997.

[14] Nicolai Scheele, Martin Mauve, Wolfgang Effelsberg, Anja Wessels, Holger Horz, and Stephan Fries. The Interactive Lecture: A New Teaching Paradigm Based on Ubiquitous Computing. In *CSCL 2003 Poster Proceedings*, Bergen, 2003. CSCL 2003.

[15] Ossi Nykänen. A Framework for Generating Non-trivial Interactive Mathematical Exercises in the Web: Dynamic Exercises. In *World Conference on Educational Multimedia, Hypermedia and Telecommunications*, pages 1473–1474, Norfolk, VA, USA, 2000. Association for the Advancement of Computing in Education (AACE).

[16] Qin Z., Johnson D.W., and Johnson R. T. Cooperative Versus Competitive Efforts and Problem Solving. In *Review of Educational Research*, volume 65 No. 2, pages 129–143. Review of Educational Research, 1995.

[17] Rolf Schulmeister. *Grundlagen hypermedialer Lernsysteme*. R. Oldenbourg Verlag, München, 2. edition, 1997.

[18] Sherman X. Huang. A Content-Balanced Adaptive Testing Algorithm for Computer-Based Training Systems. In *Proceedings of the Third International Conference ITS '96*, pages 306–314, Berlin, Heidelberg, New York, 1996. Springer LNCS.

[19] R.E. Slavin. Cooperative Learning. *Review of Educational Research*, 50:313–342, 1980.

[20] T. Mayes and L. Coventry and A. Thomson and R.Mason. Learning through Telematics: A Learning Framework for Telecommunication Applications in Higher Education. Technical report, British Telecom, Martlesham Heath, 1994.

[21] V. M., Kern and J.M., Pernigotti and M.M., Calegaro and M., Bento. Peer review in engineering education: speeding up learning, looking for a paradigm shift. In *Proceedings of the 7th International Conference On Engineering and Technology Education- INTERTECH*, page 1.1/19, Santos, 2002. SUSPI.

[22] Wainer, H. *Computerized Adaptive Testing: A Primer*. Lawrence Erlbaum Associates Inc., 1. edition, 1990.