
PRACTICAL COURSES IN DISTANCE EDUCATION SUPPORTED BY AN INTERACTIVE TUTORING COMPONENT

Rainer Lütticke FernUniversität Hagen, Hermann Helbig FernUniversität Hagen

Abstract

Preventing the time- and cost-consuming travel to practical courses in computer science for students at the FernUniversität (Open and Distance University) Hagen we have developed an Internet-based virtual laboratory (VILAB). There the students have access to complex software-tools and have to solve problems in different domains. During the problem solving processes the students get support by an interactive tutoring component. Based on the actions of the students during their work in VILAB, on the analyses of solutions or parts of them, and on a student model this tutoring component gives at once individual feedback. Additional support can be given by newsgroup, emails, or manuals. For regular teaching VILAB is used since summer 2002. The first results of our evaluation have shown that our learning environment has motivated the students, the solutions of the students are improved, and learning inside of VILAB was effective.

1 Introduction

In former times it was common at the FernUniversität that text courses were taught in distance and seminars and practical courses took place at our university. However, due to the personal situation of our students (work, family, disability, etc.) and economic aspects (travel costs, unproductive time of travelling) they have the demand to reduce the time in which they have to be present on the campus of the university. On the other side there is the problem of the sometimes feeled isolation during self-studies and the request for more communication and feedback during their learning and working on exercises.

Answering these demands we have realised a new concept of practical courses using the virtual mobility created by the Internet, new ways of communication, and modern technologies for teaching. To reach this aim for the field of applied computer science we have developed a virtual learning environment. It based on the concepts of laboratories used for the education in natural sciences. In these labs the students are guided by a curriculum to several different experimental stations. There they can work on exercises and additionally, can test and deepen their theoretical and methodical knowledge getting support by an assistant. This paradigm is used for the realisation of our Internet-based learning environment for the education in applied computer science at our university. Therefore we have called it “virtual electronic laboratory” (VILAB¹). Its functionality is described in section 2. To integrate functions of a personal human assistant of a real lab into our learning environment, to support the students, and not to leave them alone with problems during a session in the virtual lab our interactive tutoring component described in section 3 is implemented in VILAB. Analogue to traditional practical courses the learning content in VILAB used for our practical courses in distance is divided in several stations inside of the lab. There the students have access to several software tools, on which they work on their experiments and problems (section 4). The learner perspective is given in section 5. In section 6 we present our evaluation results and close with a conclusion about the kinds of support for learning we offer in VILAB.

¹ <http://pi7.fernuni-hagen.de/vilab/>

2 Working with VILAB

2.1 Architecture

VILAB is realised in a client-server architecture in which the Linux-based server is operated at the FernUniversität in Hagen (s. Fig 1). The server saves the content of the lab and manages the display of content via a navigation tool for the students. The server also harbours the tutoring component (section 3) and the software-tools and provides the communication infrastructure to databases and WebAssign². The access to the server is managed by decentral user clients (Windows- or Unix-based computers) which must be connected with the Internet. Via remote-login to the server the students are directly routed to a navigation tool which is one of the two user interfaces. After a login the server starts a new instance of the lab and the send/receive system of the tutoring component. Thereby the students cannot get access (intended or not) to the Linux-system of the server. For the display of the content of the lab the student has to open a browser on his client with an individual URL as second user interface. Via navigation tool local hypertext or PDF documents as well as documents out of the WWW can be selected by buttons. An action which shall lead to a display of a hypertext document result in an automatic reload in the browser of the user. This is realised by loading local documents in the individual directory of the user inside of VILAB and copying them to the URL of the user. PHP scripts inside of the hypertext document notice the change of the content in the user's URL and start the reload. In this way it is possible that actions on the server indirectly control the browser of the client. Documents of the WWW are embedded in a frame so that higher-level PHP-scripts can manage also the display of this kind of documents in the browser of the user. Via navigation tool the software-tools installed on the server can be activated. These tools can communicate via sockets with the tutoring component leading to the fact that every user have individual ports for this socket communication. Since every user possesses individual URLs, directories and ports inside of VILAB it is possible for the system to manage handling, saving, and display of data of many users at the same time.

2.2 User access

Realising the access to the learning environment to the students it is important to support the different client operating systems. The remote-login to the server of VILAB (via ssh to the navigation tool) is for students using Unix-like systems (e.g. Linux) trivial. Windows user have to install the open source packages *cygwin* and *xfree86.4.x* to get the analogous Unix functions needed for the access. The students can either download these packages from our web pages or ask for a CD produced at our institute and containing all needed software packages. Additionally, we demand for the client only a x86 processor and any standard browser (e.g. Netscape or Explorer).

The exclusive use of a browser environment seems to be simpler at the first glance, especially for the Windows user, but there are some large disadvantages. Since one of our main purposes is the integration of software tools to get practical experience, in this alternative variant we had to rewrite the tools as Java-Applet (in fact this is not possible due to the complexity of tools) or had to run the tools within commercial systems redirecting in- and output interfaces of the tools to a Java-Applet. On the other side the access to VILAB via ssh has the advantages that the reaction time of the tools is much smaller, all existing tools for Unix can easily be integrated in our learning environment, and we have large technical freedom for the realisation of our didactical concept. Offline working with a CD-ROM is out of discussion since some tools need the Internet access (communication with other systems), collaborative work would be excluded and access to databases for generation of adaptive feedback would not be possible. If students like to work offline they have also in the realised remote-login variant the possibility to download all text documents of VILAB.

² WebAssign is an established online system for standard exercises (e.g. multiple choice) with modules for automatical correction of such standard exercises. Additionally it allows the integration of further modules for the analysis of more complicated problems [2], [7]. This system is integrated in VILAB in such a way that users of the lab do not notice when they work with functions of WebAssign.

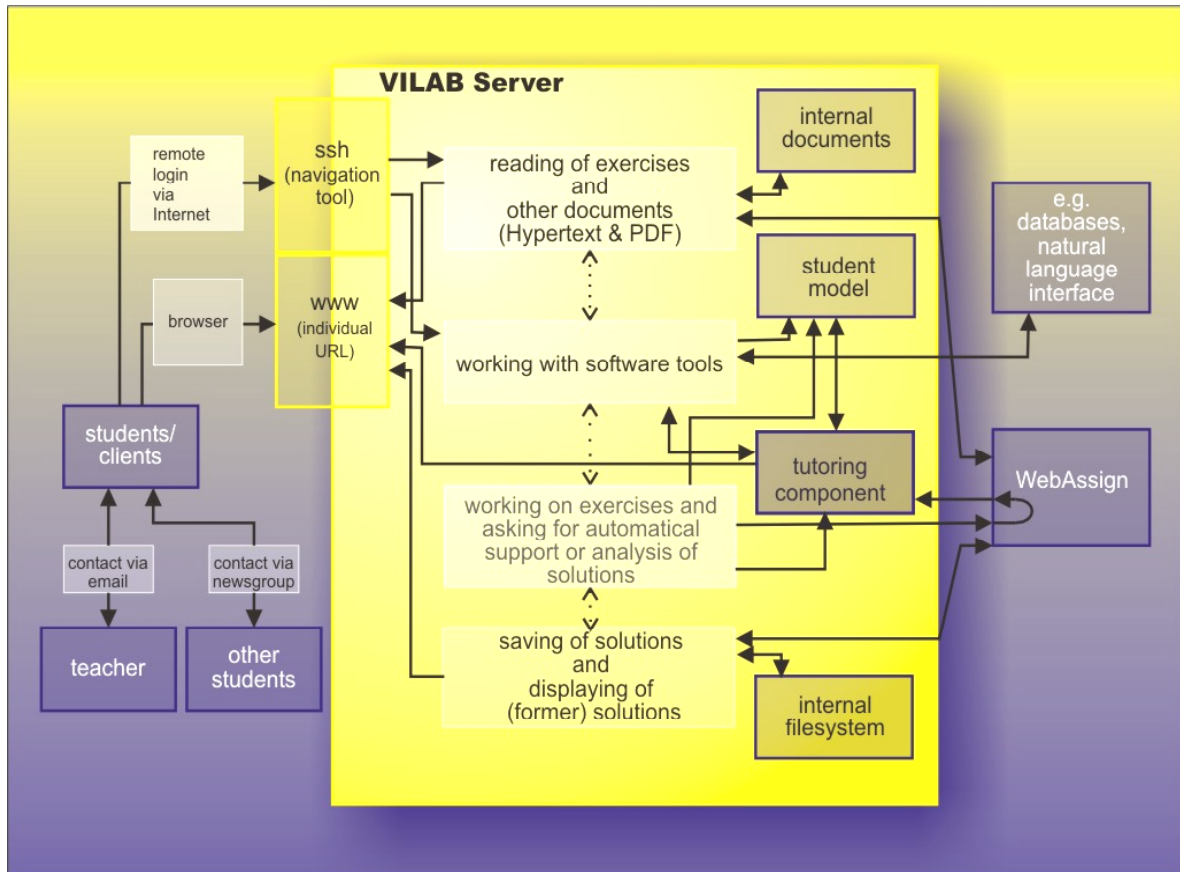


Fig. 1: Architecture and functionalities in VILAB

3 Tutoring component

While a student is working on an exercise or in a software tool, the send-receive system of the tutoring component has the functions of the error diagnosis and correction. Additionally, the tutor saves the student's result of an exercise, which was requested for correction, via the WebAssign system or the software tools. In this way the student starts every new editing of an exercise with his last result and a teacher has the possibility to examine results of the students. Depending on the learning target a possibly existing model result can be presented to the student. However, in most cases an analysis of the student's solution based on tests, comparison(s) with model solution(s), or methods using the artificial intelligence (e.g. rule based systems or inferences) is didactically better.

3.1 Error Diagnosis

Diagnosis means in this context the decision whether a solution is correct or not and to find out what and why something is wrong or incomplete. The tutoring component has a passive and an active error modus for two different kinds of problem solving support which are discussed in analyses of ITSs (Intelligent Tutoring Systems) [3]. In the passive modus ("intelligent analysis of student solutions") the send-receive system of the tutoring component has to identify errors from the student's final result after he has requested for a correction. This modus is needed for the non interactive (therefore the term "passive") programs (e.g. SQL, Java) in which errors are only detectable by a complete analysis of the student's input. This analysis is realised by correction modules and possibly following processes. The output of such an analysis integrating also an student model (Fig. 1) is an error code with accompanying parameters. The student model is based on data collected during the student's work in the laboratory and saved in a MySQL database. In the active modus ("interactive problem solving support") the send-receive system of the tutoring component does not wait for the student's final solution. If an interactive software tool detects an error, it sends at once the specific

error code with accompanying parameters to the send-receive system of the tutoring component. Thereby the student is provided with intelligent help on each step of problem solving.

3.2 Error Correction

The error correction has the goal to show the student how to remove the errors asserted in the diagnosis by identification of missing or incorrect knowledge which is responsible for errors. The overriding goal of the tutor is the finding of the logical errors and not the complete listing of syntax errors as in standard program diagnosis systems. The error correction and the proposals for the problem solving are also embedded in the error codes and the accompanied parameters. These codes are processed by the send-receive system of the tutoring component with an error table. Every error code leads to the display of hypertext documents which are individually modified by Javascript variables extracted out of the parameters. The content of the documents includes several elements (not always all elements): a statement about the correctness of the solution, an error list, a description and explanation of possible errors, hints for improvement of solution and for the avoidance of errors, a listing of useful literature or lectures to derive lacking knowledge, example of similar exercises, possibly graphical information (in software-tools), an assessment of the student's performance, and a motivation [10]. In this way the students get adaptive feedback to their individual solution.

4 Practical courses with VILAB

Conception and platform are nearly independently developed from the teaching content. In this way new laboratory stations, which contain the content, can be added and further Unix-based software tools can easily be integrated. Currently there are five stations in the laboratory: Programming with the languages Java, C, Prolog, and Scheme (procedural, object orientated, functional, and logical), neuronal networks, relative databases, as rule based systems (semantic networks and natural language transformation in SQL), and computer linguistics. Every station has a main page giving information about purpose of the station, background, reference to other stations, and an overview about the exercises and software tools (e.g. request on databases, constructing neuronal networks with the SNNS³ [12], or transform natural language information in semantic networks using the graphic knowledge engineering tool MWR⁴ [4]) in the station. The exercises are divided in learning targets, description of the task, relevance of the exercise for the field of knowledge, knowledge background for solving the exercise, helpful literature, description of the required software tools, characteristics of the solution, and kind of control/help by the tutoring component. The solutions can be standard tests (e.g. MC, numbers, correlations, etc.), free texts (programme code, SQL-queries, rules, etc.), or graphical solutions (e.g. networks created with the software tools).

In a practical course the students have to solve certain exercises of different stations in several month. How they reach the solution depends on themselves. The students can either read the given literature to the problems or get practical experience by free work with the software tools or directly work on the given exercises and problems with the interactive tutoring help. After the deadline for the work in VILAB is over the teacher controls with the help of the tutoring component the solutions and gives a final feedback. However, since the students know the comments of the automatic tutoring component they will almost know the final remarks of the teacher.

³ SNNS: Stuttgart Neuronal Network Simulator

⁴ MWR (MultiNet-Wissensrepräsentation) is a software tool for the graphic representation of MultiNet terms. In this graphic environment the terms can be produced, displayed, and manipulated. MultiNet (multilayered extended semantic networks) is a paradigm of knowledge representation created for knowledge processing and specially for the semantic representation of natural language information [6].

5 Learner perspective

During practical courses with VILAB students are supported in different ways depending on their kind of questions and problems. The access to VILAB is described in a manual (PDF) step by step. If there still exist problems students can write emails to the administrator. For the navigation inside the learning environment they can use also the help of the manual, however, the navigation is very simple so that in fact the students do not need help. Every exercise and its background is explained in detail (s. previous section) so that there is no misunderstanding. Nevertheless it is possible to ask the teacher via email for more information. If students have not enough knowledge to solve an exercise there are links to literature in the system or in the WWW. For the use of the required software tools there are embedded manuals and FAQ lists. The format of the solutions of exercises is explained and examples given so that there is no problem to construct an adequate solution. A request of an analysis of a solution results in the immediate (few seconds) display of the feedback of the tutoring component (s. section 3). Wrong solutions lead to automatically produced and adaptive suggestions concerning the improvement of the solution:

- links to literature dealing with knowledge which seems to lack
- links to examples of similar exercises
- information about errors in the solution

Adopting these suggestions students can enhance their solution and resend it afterwards. Subsequent feedback (based on individual student models including the former feedback) is again the result. A few or a lot of steps of such an interaction are possible until the solution is correct. If students want to have other support than the interactive tutoring component is giving, they can use a special newsgroup to ask for help or send an email to the teacher of the practical course. The last two kinds of the communications can also used for questions concerning functionalities inside of VILAB. However, since the students can only select between navigating, reading of exercises, working on them, opening of documents, working with software tools, asking for automatical analysis of solutions, and saving of solutions (Fig. 1) the functionalities are straightforward. Therefore there is no need for automatical help inside of VILAB regarding functionalities. The core of VILAB is its tutoring component for the analysis of exercises. Since its feedback includes all actions of the learning system which are possible it is assured that the students take full advantage of the options of the system.

6 Evaluation

The first use and the didactical testing took place in summer 2002 with concrete content of the field applied computer science. Since this summer term VILAB is used for teaching at the FernUniversität Hagen. However, VILAB is not only used in practical courses but also in seminars to deepen the students' knowledge about the individual themes (e.g. about themes correlated with software-tools, the software-tools themselves, elearning, or tutoring systems) and for self-assessment in addition to textual courses.

Results of evaluations (questions to students) show that the system works stabile, the installation of the free software for Windows user and the access to VILAB is easy and the reaction times of the systems are very small (even with 56k modem). The students like the kind of learning with VILAB, since they have a lot of freedom. The design and navigation inside of the learning environment is "good" and functional. As supposed from the view of the students the interactive tutoring component was very important because of

- its help during the finding of the correct solution
- its motivation to find the right solution
- the possibility to learn with its fast automatic feedback

The tutoring component was so effective that newsgroup or email to the teacher of a practical course were almost not used. These communication methods were also not used because functionalities were not understood. From the view of the teacher the most important point is that the students have learned more compared to former courses without VILAB: The students' solutions of the exercises are very good. There are more students who finished their practical course successful. Self-assessment was connected with better results in examinations and working in VILAB resulted in better seminar talks. Additionally, a lot of students wanted to write their diploma thesis in the field of VILAB.

7 Conclusion

VILAB is an Internet-based learning environment training practical skills in computer science by working with complex software tools. The students get practical experience by the problem solving of different exercises and the help of an interactive tutoring component, which motivate them to search for the right solution of a given problem.

The students get different kind of support for learning by VILAB:

- Solving of problems is supported by the interactive tutoring component.
- The individual style of learning is supported because the way of learning is not determined. Reading of knowledge background, collecting practical experience, and solving of problems can be mixed and build many learning circles.
- Saving of knowledge is supported. Coached learning by doing on practical exercises leads to a transfer of theoretical knowledge into practice and strategies for solving of concrete problems. Together with the monitoring of the learning success this is more successful than pure reading of texts [9], [5], [1].
- The motivation of learning is supported by help and comments of the tutoring component.
- VILAB supports the access to software-tools and databases which otherwise they could not use from home (installation to complicated, no rights of use, tools need too much disk space, wrong operating system, etc.).
- The individual demand for self-determination is supported. Via Internet the students can work independently of space and time.

An integration of collaborative software tools with corresponding exercises and tutoring feedback in the future will also train the social skills during problem solving in groups. This training is a very important educational aim inside of practical courses since collaborative working is mostly used in companies.

References :

1. BALLSTAEDT, S.-P. ET AL. (2000) Planung, Entwicklung, Durchführung von Fernstudienangeboten. Deutsches Institut für Fernstudienforschung an der Universität Tübingen
2. BRUNSMANN, J.; HOMRIGHAUSEN, A.; SIX, H.-W.; VOSS, J. (1999) *Assignments in a Virtual University – The WebAssign-System*, Proc. 19th World Conference on Open Learning and Distance Education, Vienna, Austria
3. BRUSILOVSKY, P. (1999) *Adaptive and Intelligent Technologies for Web-based Education*, *Künstliche Intelligenz*, 13(4), pp. 19-25
4. GNÖRLICH, C. (2002) *MultiNet/WR: A Knowledge Engineering Toolkit for Natural Language Information*, In Technical-Report 278, FernUniversität Hagen, Germany

5. HAYNIE, W.J. (1997) *Effects of Anticipation of Tests on Delayed Retention Learning*, Journal of Technology Education, 9
6. HELBIG, H. (2001) *Die semantische Struktur natürlicher Sprache: Wissensrepräsentation mit MultiNet*, Springer, Berlin
7. LÜTTICKE, R.; GNÖRLICH, C.; HELBIG, H. (2002) *Internet-basierte Lehre im Rahmen von VILAB*, Schubert, S., Reusch, B.; Jesse, N. (ed.), *GI-Lectures Notes in Informatics (LNI)*, P-19, "Informatik bewegt", 32. Jahrestagung der Gesellschaft für Informatik, Bonner Köllen Verlag, pp.293-296.
8. LÜTTICKE, R., HELBIG, H. (2003) *Das Virtuelle Informatik Labor als Teil des Lernraums Virtuelle Universität der FU Hagen*, von Knop, J.; Haverkamp, W. (ed.), *GI-Edition Lectures Notes in Informatics (LNI)*, P-44, "Security, E-Learning, E-Services", 17. DFN-Arbeitstagung über Kommunikationsnetze, Bonner Köllen Verlag, pp. 363-370
9. MORRISON, G.R.; ROSS, S.M.; GOPALAKRISHNAN, M.; CASEY, J. (1995) *The Effects of Feedback and Incentives on Achievement in Computer-Based Instruction*, Contemporary Educational Psychology, 20:32-50
10. MUSCH, J. (2000) *Die Gestaltung von Feedback in computerunterstützten Lernumgebungen: Modelle und Befunde*, Zeitschrift für Pädagogische Psychologie, 13:148-160
11. SCHLAGETER, G., MITTRACH, S. (1998) *Virtuelle Universität*, Informatik Forschung und Entwicklung, Volume 13, pp. 159-162.
12. ZELL, A. (1997) *Simulation neuronaler Netze*, Oldenbourg, München

Authors :

Dr. Rainer Lütticke
Prof. Dr. Hermann Helbig
FernUniversität Hagen, Department of Computer Science,
Intelligent Information and Communication Systems
Universitätsstr. 1, 58084 Hagen, Germany
rainer.luetticke@fernuni-hagen.de