LABASSISTANT: A WEB-BASED GENERAL-PURPOSE SOFTWARE FOR THE DELIVERY AND ADMINISTRATION OF COMPUTER BASED LABORATORY SESSIONS

E. Kehris, D. Paschaloudis, C. David, G. Fragidis

ABSTRACT
The design of a web-based general-purpose software system, named LabAssistant, that supports computer-based laboratory sessions is presented. The aim of LabAssistant is to provide support to the preparation, conduct, evaluation and administration of the laboratory sessions offered by academic institutions. LabAssistant may support both the traditional and distance-learning educational delivery model. LabAssistant is used in collaboration with other domain-specific software such as mathematical modeling tools, simulation software, database management systems, etc. It adopts a step-wise approach to laboratory sessions: at each step a domain-specific task is suggested to the students while the corresponding theory is presented, the appropriate domain-specific software features are explained and useful hints/suggestions related to the suggested task are made available for inspection. All this information (i.e. the domain-specific tasks, software features, related theory, suggestions) and its sequencing is determined by the academic staff during the preparation of the laboratory session and are input to LabAssistant. The students employ LabAssistant to initiate a new laboratory session or continue a previously suspended one and utilize the domain-specific software to accomplish the tasks suggested to them, while LabAssistant keeps track of the student progress. LabAssistant also provides support to administrative tasks related to the regulations and the assessment methods adopted by academic institutions.

KEYWORDS
Web-based learning environment, Unified Modelling Language, Java

INTRODUCTION

The ever increasing processing capacity of personal computers, the popularity of Internet with the open technological standards and the widely available Internet browsers caused in the 1990s the re-evaluation of adapted practices in a wide range of disciplines (Erkes et. al, 1996). In this framework, the traditional teaching methods and approaches employed by the academic community are reconsidered (Byrd and Harman, 1998; Charles, 1995). However, educational instruction still retains the traditional methods of lecture, note taking, reading and writing assignments, and laboratory training. These rather conservative pedagogical methods, while undoubtedly effective for some students, are now being challenged by more recent approaches that attempt to exploit the capabilities of new instructional and learning technologies (Hativa and Becker, 1994; Piccoli, et. al., 2001). Modern instructional material that is visually and aurally augmented while providing navigational control of the hypermedia context allows students to explore subject material at their own pace. Educational research shows that student learning improves when students are offered the capacity to interact with the instructors, the subject or instructional materials (e.g. to get immediate feedback) and co-operate with other students (e.g. to accomplish group-work tasks).

The active learning - as contrasted with the passive learning of lectures and other authority-to-student methods - can be increased with the proper use of interactive, multimedia courseware that has been competently designed and developed. Such instructional software naturally uses graphics, sound, video, and hypertext in addition to text.

Multimedia lecture presentations with colourful graphics, sound, video, and hypertext are good so far, but the addition of interactive study questions and self-assessment quizzes with feedback, allow them to deliver more efficient the educational content. Such computer-mediated courses must be web-based for several reasons: first, they are easily accessed in any classroom equipped with Internet access and computer video projection; second, since the material is permanently on the Web, it can be used by individual students on their own schedule (and place), as well as by the instructor in lecture. Web-based presentations are cross-platform and server-delivered, so the instructional materials are always and everywhere available; third, once the Web-based hypermedia courseware is designed and developed for use in the classroom, computer lab, and student desktop, it is also available for use in a distance learning framework.

The ultimate goal of a computer based learning procedure should be to create a situation in which the corresponding discipline content and the important critical thinking skills are imparted to students by hypermedia courseware, so that the professor or instructor can use class time to interact with students, ask and answer questions, offer and deliver personal help, and generally motivate and encourage students to learn on their own.

It is evident that hypertext/multimedia educational courseware, designed using instructional methods validated by research, and built and delivered by the new computer/network/web-based technologies, improves higher education and makes such instruction more successful, efficient and cost-effective. This paper describes an attempt to develop a web-based general-purpose software that will support the delivery and administration of the computer-based laboratory sessions that are offered by the Department of Business Administration of Technological and Education Institute (TEI) of Serres.

COMPUTER-BASED LABORATORY SESSIONS

The department of Business Administration of TEI of Serres offers seven computer-based laboratory modules: four of them are compulsory and three are electives. The compulsory modules are all taught during the first two semesters of study and contain two modules related to computer technology (introduction to computers, database systems), one to statistics and one to business communications. The computer-based laboratory modules are taught to small groups of students in appropriately equipped computer laboratories. Each lab is equipped with 20 PCs that are connected to a network and have access to Internet.

The number of students that attend the compulsory computer-based laboratory modules each semester is nearly 200 and as a result the number of student groups formed for these modules ranges from 35 to 40. The majority of the computer-based laboratory modules are delivered by part-time staff who are employed as laboratory associates. Normally, four to five part-time laboratory associates are employed for delivering a computer-based laboratory module each semester.

According to the regulations, each student registered to a laboratory module has to participate to a pre-determined minimum number of laboratory sessions. The laboratory associate at the beginning of each laboratory session records the students who are present. The students that have participated in the appropriate number of laboratory sessions are allowed to take part in the final exams of the laboratory module.

Although the broad subject areas covered in each laboratory module are described the module's syllabus, the detailed topics that should be taught in each laboratory session is not recorded (documented). As a result, the majority of the laboratory sessions are carried out under the instruction of the laboratory associates who orally present to the students the concepts, software tools, examples and tasks. Due to this approach (a) different laboratory associates cover the same material in different

levels of depth, using different approaches, case studies and examples (b) best practices (e.g. tasks given to the students as exercises, cases studies, examples) that are identified to work well are not adopted by all the laboratory associates that teach the same module (c) it is difficult for the group of students to synchronise. Thus, students that complete a task submitted by a laboratory assistant before their fellow students have to remain idle until the next task is presented to them (the laboratory assistants usually wait for the whole group to complete the current task before presenting the next one). In order to address these issues, detailed documentation for one laboratory module (the Database module) has been developed. The documentation was designed upon the following principles:
the laboratory associates should mainly act as tutors and not as lecturers: the laboratory associate should provide individual support to students so that they overcome the difficulties they face. This means that activities such as the provision of technical information related to the specific software or the description of the task that should be attempted by the students should not be the responsibility of the laboratory associates but rather a responsibility of the proposed software system.

The students should learn by attempting to carry out appropriate tasks: i.e. the students should be encouraged to explore the software used in a systematic way, relate software technicalities to underlying theoretical concepts and interpret computer responses generated by student actions. The graphical user interfaces adopted by current software besides their intuitive functionality provide rich information (e.g. through the help). As a result, students do not face major difficulties in understanding the general principles of current software packages. However, they do have difficulties in identifying the appropriate steps that are necessary for the solution of problem-oriented tasks as well as in transforming the information provided by the software in the form of numbers to "answers" to specific problems.

The tasks presented to students should require a small number of steps. The students should understand the proposed task easily, so that their effort should be directed on relating the task to the required steps. Most of the steps required for the completion of the task require the use of the computer. However, some steps (e.g. the interpretation of the computer result) may be carried out without the use of the computer.

Based on these principles, the documentation produced included the necessary theoretical background together with a number of tasks that had to be carried out by the students during the laboratory sessions. Introductory tasks in each session aim at familiarising the student with the theoretical issues and the technicalities related to the software and usually may be carried out in a single step, while more advance tasks aim at allowing the students to solve specific problems using the software and usually require a number of steps. For example one of the introductory tasks required the student to "develop a query that displays the address of the customers having surname starting with the letters So". This task may be carried out in a single step: the user develops an appropriate query using the software facilities. More advanced tasks represent more complex problem-oriented cases and require multiple steps. For example, an advanced task states that "A new customer named Iason Filon (address Main Str. 4) orders 35 CDs and 20 folders. Carry out the appropriate actions to represent this event in the database".

The documentation was employed for one year and was well accepted both by the laboratory assistants and by the students. However, a number of limitations have been identified. More specifically, the written documentation:

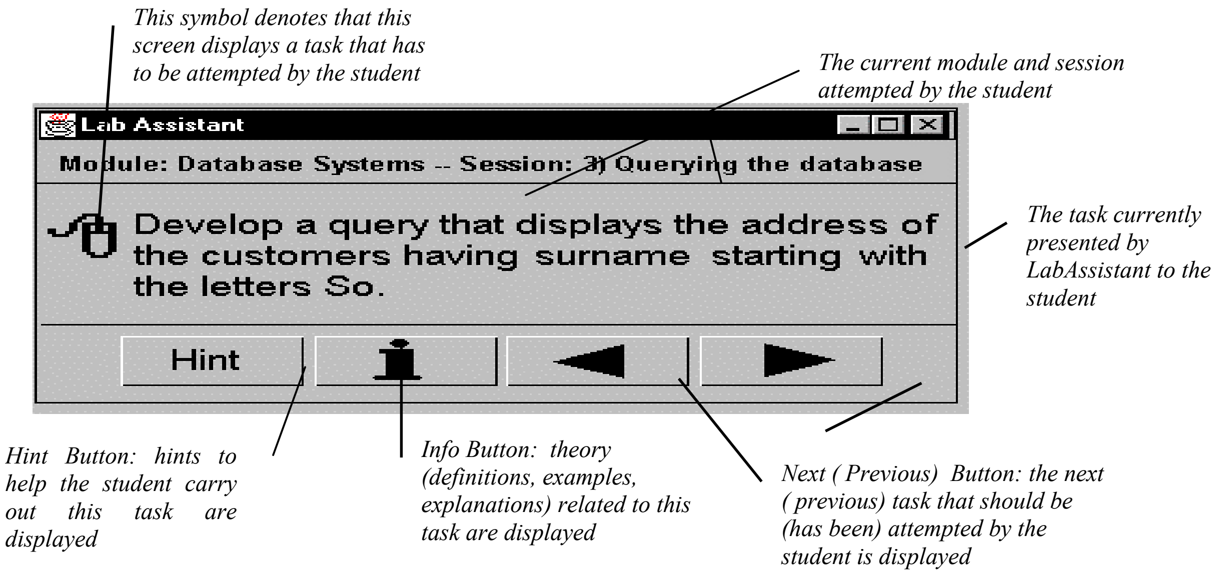
- is difficult to modify. Procedures for the collection of suggested modifications have to be established. Furthermore, these modifications have be agreed by the people involved in the teaching of the laboratory sessions and finally they have to be carried out.
- is time consuming to modify. The written documentation is distributed to the students at the beginning of each semester. As a result, even simple modifications have to be postponed for a long time
- is expensive. The written documentation has to be reproduced to a large number of copies and distributed to the students.

- is difficult to synchronise a whole group of students - given that the laboratory groups usually consist of students with significantly different computing knowledge, some students complete the tasks much faster than the others.
- places unnecessary low-level responsibilities to the participants. The students have to bring with them the documentation, they have to remember the last task they have completed during the previous laboratory session so that they continue from that point. The laboratory assistants need to have a small number of extra copies of the written documentation in the case that some students do not bring their copy, they have to assist students in locating the point they stopped last time, etc.

The software system (called LabAssistant) presented in this paper aims to eliminate the above mentioned limitations. It could be considered as an extended electronic version of the printed documentation that has been developed and used in the department. It is therefore clear that the principles adopted by the written documentation also apply to LabAssistant.

DESIGN AND PROTOTYPE IMPLEMENTATION

LabAssistant is a domain-independent software environment that may be used for the development of any computer-based laboratory module that adopts its step-wise approach. LabAssistant is executed on a window that always remains open together with the window that executes the software used in the laboratory module and presents to the students theory-related concepts, problem-oriented tasks that they have to attempt during a laboratory session and hints that may help the students to carry out successfully the proposed tasks (see Fig. 1). In LabAssistant terminology the theory-related concepts, the tasks and the associated hints are referred to as LabItems. A different symbol is associated with each LabItem as shown in table 1.


Lab Assistant

Module: Database Systems -- Session: 3) Querying the database

Develop a query that displays the address of the customers having surname starting with the letters So.

This symbol denotes that this screen displays a task that has to be attempted by the student
The current module and session attempted by the student
The task currently presented by LabAssistant to the student
Hint
Hint Button: hints to help the student carry out this task are displayed
Info Button: theory (definitions, examples, explanations) related to this task are displayed
Next (Previous) Button: the next (previous) task that should be (has been) attempted by the student is displayed

Figure 1. LabAssistant display window

The order in which the LabItems are presented to the students needs to be defined to LabAssistant. Next and previous LabItems are presented to students through the corresponding (arrow) buttons.

Table 1. The symbols used in LabAssistant and their meaning

Symbol	Meaning
🖱	A task that has to be accomplished by the student using the computer
✍	A task that has to be accomplished by the student without using the computer
👓	An observation
🕮	Theory-related concepts
◎	The aim of the laboratory session

The design of LabAssistant is shown as a class diagram in Fig. 2 using the UML notation (Fowler and Scott 1977). In UML the classes (concepts) are represented by rectangles. The LabAssistant classes and their description is shown in Table 2. Associations between the classes are shown by lines. Three types of associations are supported:

- inheritance (is-a) associations that define superclass-subclass relationships and are represented by an arrow; e.g. Task (as well as Hint and Theory) is-a LabItem;
- aggregation (has-a) associations denote a whole-part association and are represented by a rhombus; e.g. each LabModule has a number of LabSessions.
- general type relationships represented by a line having a label to improve readability; e.g. The Student is registered to a LabModule.

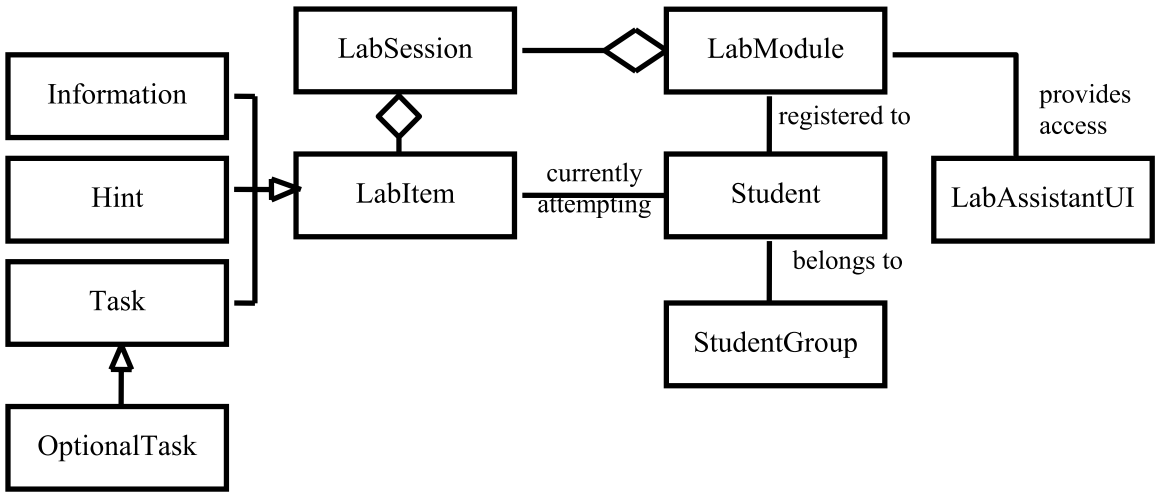


Figure 2. UML class diagram for LabAssistant

Table 2. LabAssistant classes

Class	Description
LabAssistantUI	the user interface that allows a user to access the laboratory modules that are available
Student	models a student who is using LabAssistant
StudentGroup	models a group of students (max 20 students). Each group attends a laboratory session in one of the department's PC laboratories according to a weekly schedule.
LabModule	a laboratory module that is available in LabAssistant
LabSession	a laboratory session
LabItem	text displayed on the LabAssistant's window.
Information	text providing theory-related information
Task (or Normal Task)	text describing a problem that has to be solved by the student using the software employed in the module
Hint	text providing hints that may help the student to solve a specific task
OptionalTask	additional tasks that are attempted only by the students who have completed the normal tasks.

As it is shown in the UML class diagram, LabAssistantUI provides access to a number of laboratory modules (LabModules), each of which consists of a number of laboratory sessions (LabSessions). In turn, each laboratory session contains a number of laboratory items. As it has been explained earlier, a laboratory item may be either a theory-related concept (called Information) or a problem-related task (called Task) or a hint related to a task (called Hint). It should be noted that each laboratory session contains a number of (normal) tasks together with a relatively small number of optional tasks. Optional tasks are problems that are not necessary to be carried out by the students; they are attempted only by the students that have completed successfully the normal tasks of the session.

LabAssistantUI allows students to be registered as users in any of the available laboratory modules of LabAssistant. During the user registration procedure the students provides his/her group and password. After being registered to a laboratory module, the student starts to explore the contents of the first session. Typically, the student is firstly presented to some theoretical issues as well as with short descriptions related to the software employed. Then the student is required to carry out specific tasks using the module-specific software.

LabAssistant monitors and stores the progress of each student. More specifically, for each student the most recent task accomplished is stored while the time it took him/her to accomplish each task is calculated and stored. The most recent task accomplished by each student is used by LabAssistant to determine for each student the next task he/she has to attempt and to estimate the overall group performance. In cases where LabAssistant identifies students that have advanced significantly in relation to the rest of their group, it proposes to them optional tasks.

The time required by each student to accomplish a task may also provide important information related to the difficulty of the laboratory sessions. For example, relatively short accomplishment time for specific tasks may suggest that these tasks are easy for the students and thus their number may be reduced. On the contrary, relatively long accomplishment time for other tasks may suggest that these tasks are more difficult for the students and thus possibly it is sensible to allow the students to attempt more of them. LabAssistant determines the time it takes a student to accomplish a task by calculating the time elapsed between two successive presses of the Next button. The data manipulated by LabAssistant (i.e. the registered users, the available laboratory modules with their corresponding sessions and items, etc) are stored in a relational database. The schema of the relational database is shown in Figure 3.

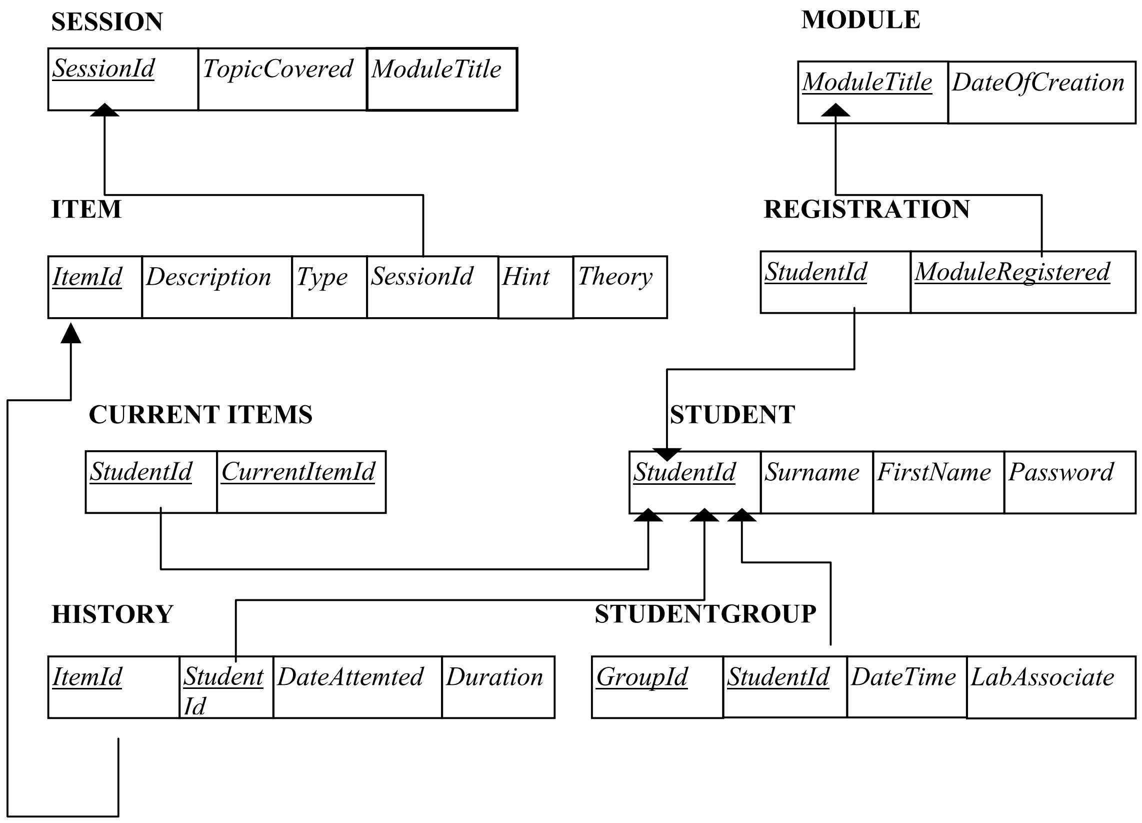


Figure 3: The relational database used by LabAssistant to store data

A prototype version of LabAssistant that supports a single user has been implemented based on the client-server architecture. The client is responsible for implementing the user interface while the server is dedicated to storing the related data. Thus, the client is implemented using the Java programming language while the server is implemented by a database management system (dbms) that is accessed by the Java client whenever the user presses one of the available buttons. The content (LabItems) of a laboratory module structured into laboratory sessions needs to be stored appropriately in the database using the standard facilities provided by the dbms. Experimentation with the prototype implementation demonstrates that it allows the quick and easy modification of the content since standard update facilities are provided by the dbms. Furthermore, the material is distributed electronically eliminating the cost of the printed version.

FUTURE WORK

Future work is going to focus in a number of directions. Initially, the effort will be directed towards the development of a fully functional version of the software. In its fully functional version LabAssistant will support multiple users, monitor their progress and keep them synchronised by presenting to the students belonging in the same group tasks of the same laboratory session. The developed fully functional version of the software will be evaluated. Students will be asked to evaluate the functionality, user-friendliness and the quality of the content by filling-in structured questionnaires.

In addition, software tools to facilitate the production of the laboratory material is also an immediate objective. The material of a laboratory module may be easily developed in parallel by a number of people if the participants agree upon a common basis and appropriate software tools are available. For example, if the scientific and the laboratory associates who deliver the laboratory module Database Systems agree upon the example (case study) and the structure of the module (i.e. the topic of each laboratory session) then each laboratory associate may be assigned the development of a laboratory session. At the final stage, the independently developed laboratory sessions may be evaluated by the laboratory associates who may propose modifications. Special-purpose software designed to support the collaboration among the people involved in the production of laboratory modules will have immediate effects on the development lead time and the quality of the content.

Finally a much more challenging improvement is the development of module-specific software facilities that evaluate the answers provided by the students to the suggested tasks.

REFERENCES

Byrd, A. K. and Harman Y. S., 1998, «Collaborative Learning in the College Classroom: Building Knowledge with Teamwork and Technology». Journal of Educational Media & Library Sciences 35.3

Charles, C.M, 1995. «Introduction to educational research», (2nd ed.). White Plains, NY: Longman Publishers USA.

Erkes J. W., Kenny K. B., Lewis L. W., Sarachan B. D., Solobewski M. W., Sum R. N., 1996, «Implementing Shared Manufacturing Services on the World Wide Web», Communications of the ACM, 39(2), 34-35.

Fowler M., Scott K., 1977, «UML Distilled: Applying the Standard Object Modelling Language», Addison Wesley.

Hativa, N. and Becker, H. J., 1994. «Computer based integrated learning systems: Research and theory». International Journal of Educational Research, 21(1), 1-119.

Piccoli, G., et. al., 2001. «Web-based Virtual Learning Environments: A Research Framework and a Preliminary Assessment of Effectiveness in Basic IT Skills Training». MIS Quarterly, 25(4), 401-426.

E. Kehris, D. Paschaloudis, C. David, G. Fragidis
Technological Education Institute of Serres
Greece
Email: {kehris, dpasch, david}@teiser.gr