International Review of Research in Open and Distributed Learning

Simulating the Computer Science Closed Laboratory in an Asynchronous Learning Network

Timothy Margush

Volume 3, numéro 2, octobre 2002

URI : https://id.erudit.org/iderudit/1072883ar
DOI : https://doi.org/10.19173/irrodl.v3i2.94

Aller au sommaire du numéro

Éditeur(s)

Athabasca University Press (AU Press)

ISSN

1492-3831 (numérique)

Découvrir la revue

Citer cette note

Margush, T. (2002). Simulating the Computer Science Closed Laboratory in an Asynchronous Learning Network. *International Review of Research in Open and Distributed Learning*, *3*(2), 1–8. https://doi.org/10.19173/irrodl.v3i2.94

Résumé de l'article

The quiz tool that is part of the WebCT package is used to provide an asynchronous simulation of a (synchronous) closed laboratory experience to beginner computer science students. The technique suggested in this paper and which may be applied to other disciplines, offers students a guided path through an exploratory, discovery based learning experience.

International Review of Research in Open and Distance Learning
Volume 3, Number 2. *ISSN: 1492-3831*

October - 2002

Research Notes – Vol. 3, No. 2

Simulating the Computer Science Closed Laboratory in an Asynchronous Learning Network

Timothy Margush
The University of Akron
USA

Abstract

The quiz tool that is part of the *WebCT* package is used to provide an asynchronous simulation of a (synchronous) closed laboratory experience to beginner computer science students. The technique suggested in this paper and which may be applied to other disciplines, offers students a guided path through an exploratory, discovery based learning experience.

Keywords: *WebCT*; programmed learning; computer science; programming; teaching programming; closed laboratory

Introduction

In the 1991 Report of the Association for Computing Machinery and the Institute for Electrical and Electronics Engineers-Computer Science (ACM/IEEE-CS) Joint Curriculum Task Force (Tucker, 1991), closed laboratory experiences are recommended as an essential part of the undergraduate computer science program. In that report the closed laboratory model is defined as:

> *. . . a scheduled, structured, and supervised assignment that involves the use of computing hardware, software, or instrumentation for its completion. Students complete a closed lab by attending a scheduled session, usually 2-3 hours long, at a specific facility. Supervision is provided by the instructor or a qualified assistant who is familiar with the details of the assignment. The specialized equipment, software, and supervision offered by closed laboratories makes them more desirable than open laboratories in certain situations. Closed laboratories are particularly important in situations where the assignment relies on instructor-student interaction or a team effort among students to complete the work.*

The most recent report from the ACM/IEEE-CS Joint Curriculum Task Force stipulates: "Most courses in a computer science program must include a laboratory component that requires students to develop their technical skills and acquire an understanding of

effective professional practice" (Chang, 2001). This report reinforces the importance of including laboratory experiences to the learning process.

Closed laboratory experiences were first introduced to the computer science curriculum in the mid to late 1980s. This mode of instruction offered a controlled and supervised setting where students could design and test small programs, or use prewritten programs to support experimental inquiries into various issues related to the course material. By definition, the closed laboratory environment requires the presence of instructors or assistants knowledgeable enough to help students through difficult parts of the assignment. Closed laboratory assignments are typically designed to be completed within a two or three hour time frame.

As more universities try to meet the needs of an increasingly diverse and mobile student population, faculty have been encouraged to develop class offerings in distance learning formats or at least to provide online support for traditional classes. These online components are offered in both synchronous and asynchronous modes.

Synchronous distance learning implies person-to-person communication in real time. The communication medium can be as simple as the telephone or may involve Internet based technologies such as chat rooms, whiteboard, text messages, audio/video presentations, or virtual classroom simulations. The important theme is that students and instructor communicate in real-time. Because this format combines particular demands on both student and instructor schedules, this mode may be impractical for students who may live in widely different time zones.

Asynchronous distance learning means that students and instructors may access course information at different times. Support for the asynchronous model is generally through CD-ROM, or online accessible content, message boards, and email. One disadvantage of the asynchronous model is the delay between questions and answers. Instructors may compose elaborate email responses to student questions, only to learn that students either answered questions for themselves shortly after submitting their questions or have lost interest due to the lack of any immediate response. Delays in obtaining answers to questions may prevent students from completing assignments in a timely manner.

Because all students gather at an appointed time and place to work together on a prepared assignment, the closed laboratory model is, by definition, synchronous. Thus the need to provide this important mode of instruction in an online environment brings special challenges. A synchronous online environment can meet the requirements outlined in the Task Force's definition of closed laboratory if a "specific facility" is interpreted to mean a common chat room, and if supervision can be accomplished in an online format. Team efforts and instructor-led experiments can be facilitated through the use of whiteboard, chat, and even voice and video tools.

At the time the ACM 1991 guidelines were written, the specialized software mentioned in the definition was either expensive or required machine resources were not commonly available to many students in their homes. As system and software prices have dropped, and as home computers have become more powerful and ubiquitous, it is now probable that more students will have the required resources in their homes to support all of the activities typically performed in closed lab settings. In fact, for the introductory programming classes, it is possible to provide free, yet sophisticated, integrated development environments to support programming activities.

Closed laboratory components have been a standard part of introductory courses in many undergraduate computer science programs. Advances in technology and the readily

available hardware and software needed to decentralize the computer laboratory make it feasible to explore distance learning implementations of the closed laboratory model.

Using interactive voice or text messaging, remote desktop viewing or control, and possibly chat or whiteboard tools, synchronous techniques are ideal in that they allow immediate feedback to questions. Students are able to progress through assignments without unnecessary delays, obtaining help when needed. However, due to conflicts between student and instructor schedules, this mode of instruction may sometimes be impractical.

As asynchronous implementation increase flexibility in scheduling, students around the world are able to work at their own convenience. However, such techniques deviate from some of the strict wording of the definition of a closed laboratory experience. However, by relaxing some of the constraints in the definition, many of the benefits afforded by the closed laboratory setting can be retained even in an asynchronous learning network.

This paper documents the use of the *WebCT* quiz tool to provide a programmed learning style of presentation that combines tutorial, hands-on practice, and assessment to simulate the closed laboratory experience. This approach is a compromise between the closed laboratory setting and an open laboratory style (where students are provided little or no help in completing an assignment) that can easily be implemented in an asynchronous learning network. The use of the *WebCT* quiz tool as a programmed learning guide allows students to work through an assignment in a step-by-step fashion, at their own time and pace, without the frustration of being unable to complete a critical step that would prevent them from finishing the assignment.

Background

Programmed learning techniques were popularized in the 1960s, based largely on the work of B. F. Skinner. In this approach to education, a cognitive domain is divided into small presentation units, sometimes called frames. The frames are sequenced so knowledge units required for subsequent frames are mastered before they are needed. Two modes of programmed learning are generally identified, linear and branching. Both Personalized System of Instruction (PSI) and Computer Aided Instruction (CAI) have made use of both of these techniques for many years (Pettijohn, 1998).

The *WebCT* platform's quiz tool provides a simple and effective tool to implement the linear model of programmed learning. The *WebCT* quiz can be configured to present one question at a time, allowing small portions of a learning unit to be presented. This fulfills the first requirement of the programmed learning technique. Each question may contain a description of a concept, or simply present a task to be completed. Each frame of the presentation ends with some type of assessment. This could be a simple yes/no, multiple choice, short answer, or even paragraph style question that would allow a complete program or code fragment to be pasted into the answer area.

Once a question has been answered, a typical CAI system calls for analysis of the answer and perhaps employing branching techniques to provide custom feedback to students. In the linear model, the system simply moves on to the next question. Since the *WebCT* quiz module does not analyze the responses to individual questions until the entire quiz is completed and submitted for grading, immediate feedback can only be general in nature and must be supplied as part of the next question to be presented. This supplies reinforcement when correct answers are given, and offers an opportunity to present essential information to facilitate progress through subsequent frames.

Simulating the Closed Laboratory using Programmed Learning Techniques in *WebCT*

Closed laboratory exercises designed to supplement introductory programming classes typically involve the following activities: detection and correction of basic programming errors (syntax), locating and correcting errors in logic (semantics), synthesis and modification of code (program design and maintenance). Successful completion of these tasks often requires a multiple step solution. To ensure students can progress through each step, instructors are present in the closed laboratory setting to provide assistance when required. The major barrier to converting a traditional closed laboratory assignment to an asynchronous format is the absence of this immediate help. Without the availability of an instructor, students may be unable to complete a critical task and thus to progress through subsequent stages of the assignment.

One possible solution is to employ programmed learning techniques with feedback and reinforcement. Using this approach, the individual steps of a task are presented in sequential learning frames. Each frame includes the presentation of some new idea or a task to be performed. Each frame ends with a question that must be answered before proceeding to the next. By anticipating the critical points in an assignment, solutions needed to progress through the multi step task can be included at the beginning of the next frame, thus providing immediate encouragement for those students who successfully complete each step. More importantly, it allows a student who has answered a question incorrectly (or not at all) to view a correct response and make the appropriate adjustments to their program, allowing them to continue with the next step. Although a branching model of programmed learning would provide more individualized feedback, the linear model is sufficient in most cases to replace the role of the instructor in the closed laboratory setting.

Although designed to administer quizzes via the World Wide Web, the *WebCT* quiz tool can be used to implement a closed laboratory assignment in a linear style programmed learning module. It also enables quizzes to be presented in two basic modes: one question may be presented at a time, or all questions may be viewed as a single webpage. To use this tool to create a programmed learning module, the questions must be viewed one at a time. It is also desirable to control the order in which the frames are viewed. The quiz tool can be configured to prevent students from revisiting a question once it has been skipped or answered. This feature also allows the module to be used as an evaluation tool.

The one shortcoming of the *WebCT* quiz tool as a platform for the presentation of a programmed learning module, is its inability to immediately evaluate each response and provide customized feedback. Quizzes administrated via the *WebCT* quiz tool are not graded until all questions are answered and the student submits the quiz for grading. This means that feedback must be of a general nature and must be included at the start of the next frame.

The *WebCT* quiz module provides automated grading facilities for many of the question types. This makes grading lab assignments an easy task. For one thing, all of the assignments are submitted in the same format; and since submissions are electronic, all responses are clearly legible, thus the often time consuming process of grading lab assignments is reduced.

The quiz tool provides several mechanisms for providing students with comprehensive feedback after the assignment is graded. *WebCT* quizzes allow the grader to supply a written comment after each response. Automatic feedback can also be built into the

questions themselves and can be made available to student after the completed assignment has been submitted. Provision is made for general feedback for each question, as well as custom feedback for each possible response (for some question types). The quiz can also be configured to display different levels of feedback.

Although it would be possible to overcome many of these shortcomings through the use of external or custom software, integration of such a product within the *WebCT* environment is difficult. *WebCT*-Vista advertises the inclusion of: "a development toolkit for integration with either custom or commercial learning software, giving institutions the ability to seamlessly incorporate external e-learning applications" (*WebCT*, 2002). This facility may allow integration with the *WebCT* internal email and grade book, making the development of a custom programmed learning and assessment tool feasible.

The basic ideas elaborated above may be applied to create programmed learning modules in a variety of disciplines. The rest of this paper examines some of the techniques I have used in developing an asynchronous learning network version of a typical closed laboratory assignment for an introductory programming course. In this case, sample questions from a laboratory exercise covering basic repetition structures are used to illustrate the concepts. In this lab assignment, students needed to predict the behavior of existing loops, modify the behavior of loops, and design their own loops.

Most laboratory assignments are accompanied with some pre-written source code files. In our course, all students use the same compiler (provided with the textbook), so required assignment files are provided in a format used by that compiler. This zipped collection of files is made available for download. Prior to beginning the assignment, the zip archive must be retrieved and expanded on the students' local disks. For some students, the process of downloading and unzipping an archive is new, and appropriate instructions and software are provided.

Lab exercises begin with a list of objectives. Traditionally, I include some background material presented in small informational blocks. Each block ends with a question that can usually be answered by reading the material. In general, 10 to 20 percent of the assignment comprises questions such as Sample Question 1, which is an example of a pre-lab style question:

Sample Question 1: This Hands On assignment focuses on repetition structures. The objectives are: (1) to write a simple counter-controlled loop using the for structure; (2) to become more familiar with the different repetition structures provided in C; (3) to understand the importance of proper initialization of variables; and (4) to learn that more than one solution can be produced for a given problem. The C++ programming language supports three distinct repetition structures: while, for, and do/while. Each of these structures is generally used for specific situations, but the programmer should recognize that the common usage does not mean another structure will not work. By exploring the use of an alternative structure in a given situation, the programmer should be able to recognize similarities and differences in the structures.

Name the three looping structures supported by the C++ language: ___

Once the background material has been introduced, the student is instructed to open the project for the first task. One of the important skills programmers need to develop is the ability to read and understand programs written by others. Most lab exercises include activities that develop this skill. Sample Question 2, provided below, is an example of a question that encourages learners to develop program reading skills:

Sample Question 2: You will need to open the workspace for part A of the assignment. Look at the source code and read the documentation. The program is supposed to compute the sum of a sequence of integers based on user input. If the user enters 3 and 5, what should the program display (according to the program specifications) as the sum?
A) 3 B) 5 C) 12 D) some large negative value E) none of the above

If students simply run the program, they will probably give the wrong answer to this question, as the provided program (by design) does not meet the stated specifications. Other similar questions might ask students to examine segments of code and predict the results, or execute the program with particular test data and report the results.

So far, students can work through the lab without facing critical programming tasks that might cause frustration and possibly prevent them from completing the assignment. When changes to the program are required, subsequent tasks often depend on students making correct modifications. After asking students to make a change to the program, the subsequent question should include a correct solution. If students are unable to complete the prior step, they may easily copy and paste the suggested solution into their program and proceed to the next question. Sample Question 3 illustrates how a possible answer might be included at the beginning of a question:

Sample Question 3: Hopefully you moved the cout statement to the location shown below. This will allow the original value of the variable start to be printed before it is changed by the body of the loop. Be sure your program resembles the one below before going on with this question.

```
int main()
{
int sum, start, stop;
cout << "Enter the starting and stopping value (eg. 3 10): ";
cin >> start >> stop;
cout << "The sum from " << start << " to " << stop << " is ";
while(start <= stop)
sum += start++;
cout << sum << endl;
return 0;
}
```

This change has not corrected all of the problems. You should still be concerned about the incorrect sum being displayed. What is the cause of the problem?

When students encounter Sample Question 3, they would have just completed the previous frame in which they were asked to correct an observed error related to the value of the variable start. The solution displayed in the question shows the correct modification required by the previous task. If students' answers are incorrect, they have the opportunity to correct them before going on. Having correct solutions at this point is essential to continuing. At the end of each frame, students are asked to identify the cause of the remaining logical error. In the next question, they will be asked to correct the error. If they made an inappropriate change to answer the preceding question, or if they incorrectly identify the error at this stage, they will not have much chance at supplying the appropriate correction at the next stage of the assignment; therefore, the correct answer to this question will again be supplied in the next.

Sample Question 4 includes a brief analysis of the problem the student was asked to identify in Sample Question 3 and suggests a solution. This will usually be sufficient to

get a student back on track if they did not see this problem and tried to make some other change.

Sample Question 4: As you probably determined, the problem with the loop is that the variable sum is not initialized. Add a statement to set sum to zero before the start of the loop. Verify that the program gives the correct output for several input cases.
There is one final change that you need to make to complete this portion of the lab. Modify the looping structure completely, using a for loop in place of the while loop. You can change the body of the loop, but do not change any of the surrounding code. Specifically, replace the two lines (starting with the keyword while) with a for loop that correctly computes the sum.

Test, and when satisfied, paste a copy of the modified main function here:

Successful completion of this task again requires modification of the existing code. The solution is then pasted into a paragraph style answer box in the *WebCT* quiz. Since this is the last task to be performed on this part of the assignment, no solution needs to be provided in the next frame. However, students will benefit from eventually seeing a correct solution, so it is important to provide feedback once the assignment has been graded. I generally use the feedback option in *WebCT* to provide feedback for each question, especially if the correct answer is not given in the next question. When grading assignments, there is a place to add comments to each response, so this makes it easy to customize feedback.

Conclusions

Introductory programming courses are typically taken by students who have little or no programming experience. Prior to taking our introductory computer programming course in an online, asynchronous format, many students expressed concern about the lack of in-person help. Indeed, knowing from experience that students often needed help at several stages of the assignment in order to complete it successfully, I also initially had doubts about the effectiveness of using the closed lab assignments as a component of the asynchronous course. The programmed learning style of presentation seems to have successfully addressed these difficulties. Student evaluations have provided positive feedback about the HandsOn assignments (my terminology for the closed laboratory assignments). Three students wrote:

> *"In regards to the HandsOn experience, these exercises are one of the most helpful tools to learn programming."*
>
> *"I am most impressed with the hands-on assignments in particular. If you recall, I was in this course back in Fall 2000 and the hands-on assignments are very similar to the in-class labs we had then. They are very interactive and they illustrate some fundamental concepts in the course."*
>
> *"Firstly, I am very happy with the hands on assignments because I took this course at another university last semester and needless to say I am retaking it because I did not have enough practice."*

When building an asynchronous version of a course that traditionally includes a closed lab experience, it is well worth the time to develop a programmed style interactive quiz that simulates the closed laboratory session. I generally give students an entire day to complete a HandsOn assignment. This allows students adequate time to think about

problems they encounter, and perhaps get help through the discussion board, email, or chat (if available) from the instructor or other students. Most students have little difficulty completing the assignments within this time frame.

As long as specialized equipment is not required, the application of programmed learning techniques to the problem of implementing closed laboratory experiences in an asynchronous learning network may be extended to other traditional laboratory settings. The ready availability of computers and software required for introductory computer science laboratories makes the distance learning format possible at this time. For medical or scientific labs, this may not be the case. Even in the field of computer science, access to special input devices such as virtual reality equipment or digitizing tablets may not be feasible at remote locations. Such laboratories would still likely require on-site facilities or a simulation of the equipment.

References

Chang, C. (2001). Computing Curricula 2001 *Computer Science: The Joint Task Force on Computing Curricula, IEEE Computer Society and Association for Computing Machinery*. Retrieved June 15, 2002 from: http://www.acm.org/sigcse/cc2001/cc2001.pdf

Pettijohn, T. F. (1998). *Psychology: A ConnecText, Fourth Edition.* Hightstown, NJ.: Dushkin-McGraw Hill.

Tucker, A. B. (1991). *Computing Curricula 1991: Report of the ACM/IEEE-CS Joint Curriculum Task Force*. New York: ACM Press.

WebCT (n.d.) *Vista Extensible, Enterprise-Class Architecture.* Retrieved October 1, 2002 from: http://www.WebCT.com/products/viewpage?name=products_vista_architecture.