

Learning with In-Class Technology: The 15-Minute Learning Module Approach Litre Project Report

George N. Rouskas, Carolyn S. Miller
Department of Computer Science, North Carolina State University
{rouskas,miller}@ncsu.edu

ABSTRACT

The PIs designed, implemented, and incorporated in their CSC 216/316 courses a set of learning modules (14 modules for CSC 216 and 10 for CSC 316) to demonstrate fundamental programming concepts. Each module provides a new abstraction, concrete connections to demonstrate the relevance of the material, and an exercise to be done by the student. Implementation and assessment of CSC 216 (respectively, CSC 316) took place in Fall 2006 (respectively, Spring 2007). Results to assessment-specific questions from two sections of CSC 216, one technology-based and one non-technology-based, were collected and compared. The comparison did not provide evidence that the availability of laptops made any appreciable difference in student scores; however, we speculate that the interactive learning experience from the learning modules is more important than the underlying technology.

The use of the learning modules has increased the degree of interactivity between instructor and students. Although there was no substantial difference in the amount of material covered, preparing a well-defined learning module and making sure that the relevant JAVA code works correctly is a time-consuming task. We estimate that preparing for a technology-based section takes approximately 20% more time than a non-technology-based one, with a similar increase on TA effort.

Our experience shows that it is easy to overestimate what can be accomplished in 15 minutes. We have also found that it is important to provide students with incentives (i.e., course credit) for completing the in-class assignments. Also, it was difficult to determine if laptops were being used in any productive way when exercises were not being done. For programming projects, it is important to use standardized software that is readily accessible to students. Otherwise, it is likely to spend a large amount of time in-class to identify and resolve software incompatibilities. We have also come to realize the importance of infrastructure in terms of wireless access and availability of electrical outlets in the classroom.

The learning modules for CSC 216/316 are available for all CSC faculty to use in future semesters.

INTRODUCTION

CSC 216 (Programming Concepts) and CSC 316 (Data Structures) are core Computer Science courses with an approximate combined enrollment of 300 students per school year. The PIs have recently taught, and will continue to teach, laptop sections of CSC 216 (Miller, Fall 2004 through Spring 2007) and CSC 316 (Rouskas, Spring 2005 through Spring 2007) using technology in the classroom. These sections were sponsored originally by the College of Engineering (CoE) Laptop Initiative program. The goal of this project was to enhance student learning by introducing technology into the classroom experience. Each student and the instructor used laptops during each class meeting. Based on student surveys of the laptop sections conducted by Dr. Rebecca Brent, students were receptive of the new format; her report to one of the PIs (Rouskas) at the end of the Spring 2005 semester states: "Looking over the data, I think there was a very positive response to the technology. Most students seemed to feel that it helped their learning." Similar observations applied to the laptop section of CSC 216.

Given that the vast majority (well over 90%) of incoming CoE freshmen bring their own computer (typically a laptop) to campus, it is envisioned that in the near future all sections of CSC 216 and CSC 316 will eventually transition to the laptop format. While isolated efforts by individual instructors to introduce technology in relatively small, separate sections of a class may yield good results, extending this concept to all sections of a sequence of core courses (such as CSC 216/316), including sections with a large number of students, is an ambitious and challenging task. When implementing such a transition to technology-based instruction on a broader basis, one has to pay attention to the impact on courses that follow the laptop course, as well as on the curriculum overall. Because the CSC 216/316 course sequence provides students with the fundamental concepts of programming, data structures, and algorithms, it plays a central role in the Computer Science curriculum. Therefore, this course sequence is an ideal candidate for introducing students to technology-based instruction that complements traditional class lectures.

We used LITRE funding to design, implement, and incorporate in our CSC 216/316 courses a set of learning modules to demonstrate fundamental programming concepts. The PIs were responsible for designing the learning modules and integrating them in their lectures, and they supervised TAs who implemented the modules, tested them, and created a web-based interface. Students with laptops are able to download and use the modules during the lecture period to gain hands-on experience with the corresponding concepts. The learning modules were designed to complement the lecture material and reinforce important concepts (refer to "Project Objectives" below).

Objectives

Developing learning environments that incorporate technology in the classroom is a challenging task. By adapting learning materials to use the technology, the instructor can provide a pedagogy utilizing more learning styles and provide an interactive learning experience for each student. For instructors, we hope to raise awareness of how technology can be used in the classroom, support them in developing a new style of classroom learning environment, and provide them the means to move beyond (and "unlearn") the one-way communication typical of most college lectures. For students, we aim to provide a personalized learning experience within the classroom, improve student confidence with new learning, and convey to learners exactly what needs to be mastered.

Outcomes

The following are expected outcomes for students taking our laptop sections:

1. Students will acquire core concepts in programming and will apply JAVA code to illustrate core concepts in the discipline. This outcome relates to LITRE goal d) performance in the discipline.
2. Students will be able to apply data structures and algorithms to solve a wide range of realistic problems. This outcome relates to LITRE goal a) problem solving, and d) performance in the discipline.
3. Students will evaluate alternative approaches and assess their strengths and weaknesses with respect to given applications. This outcome relates to LITRE goal a) problem solving, and d) performance in the discipline.
4. Students will have a positive attitude towards programming by using computers in the classroom.

More specifically, in coordination with Dr. Dianne Raubenheimer, we developed two outcomes for the project:

1. Students will be able to apply Linked List and Tree data structures to solve realistic problems.
2. Students will construct Linked List and Tree data structures and implement recursive and non-recursive algorithms for managing these data structures.

Outcomes 1 and 2 are currently outcomes of CSC 216 and CSC 316. CSC 216 covers Linked List data structures. CSC 316 covers both Linked List and Tree data structures.

METHOD

We organized the classroom experience into modules with each module providing a new abstraction, concrete connections to explain the relevance of the material, and an exercise for the student. The exercise is meant to make the learner an active learner, stimulating him/her to apply concepts and explore the if/why/how/what of the topic. The results of the exercise provide the instructor with feedback to guide the next stage of learning. The

modules provide short (10-15 minute) cycles to be repeated for many/most of the lecture periods.

Each learning module is available to students through a dedicated web page, and contains:

1. a section on the abstraction;
2. a section discussing the concrete relevance; and
3. an exercise to be downloaded and used to apply the knowledge of the module.

Students receive the entire module or portion of the module in class and return the completed exercise to the instructor, all electronically.

The exercise component of each module is specific to the concept/abstraction. For the CSC 216/316 courses that involve programming, the exercise generally includes the following three parts (refer also to sample modules attached):

- a) A software component (Java code) which manipulates the input or implements visualization and other auxiliary functions. This piece of code may not be directly related to the concept emphasized in the module, and may be time consuming to implement and debug. Therefore, it will be provided to the students in advance.
- b) Code which is directly related to the concept to be reinforced. This piece of code will not be provided in advance; instead, students will have to implement this code in class. For instance, students may be asked to write the code for binary search or for inorder traversal of a tree structure.
- c) Input data, typically in a separate file, that will be used by the software; e.g., a file with n names to fill the array to be searched.

Timeline

The PIs jointly implemented this project in three phases:

1. *Design phase, Summer 2006.* The PIs jointly identified the important abstractions and concepts to be emphasized, coordinated the sequence of modules between the two courses, and determined appropriate in-class exercises for each abstraction/concept.
2. *Implementation and assessment of CSC 216, Fall 2006.* Prof. Miller taught two sections of CSC 216, one technology-based and one non-technology-based during Fall 2006, and she implemented 14 learning modules.
3. *Implementation and assessment of CSC 316, Spring 2007.* Prof. Rouskas taught a technology-based section of CSC 316 during Spring 2006, and he implemented 10 learning modules.

Evaluation, Assessment, Data Collection and Analysis Plan

The two project outcomes were assessed using the following plan:

1. Special questions at mid-term and final exams were developed to assess the students' ability to solve realistic problems using Linked List data structures.

- Results from a non-technology-based class and a technology-based class were collected and compared.
2. Students completed programming projects common to a non-technology based class and a technology-based class. A portion of these projects focused on Linked List data structures. Results from the grading rubric were collected and analyzed for the non-technology-based and technology-based classes.
 3. Dr. Dianne Raubenheimer, COE Director of Assessment, observed two lectures of CSC 216 in Fall 2006, and one lecture of the technology-based section of CSC 316 in Spring 2007.

For Fall 2006, the Department of Computer Science offered 5 sections of CSC 216. Two of these sections used the modules and were assessed. These two sections were comparable in the following ways:

- Both sections were taught by the same instructor (Miller) who administered the same exams, the same homework, and the same programming assignments.
- The sections were of comparable size and were composed of students primarily from CSC and ECE departments.
- The sections were both taught in 1 hour 15 minute time periods on the same days.

The sections were not comparable with respect to technology in the classroom. One section required all students to bring and use laptop computers to each class. These computers were used for the exercises in the modules. The other section did similar exercises but used paper and pencil. A few of these students brought their own laptops to class but did not use them for the exercises.

Special questions were developed and used on the final exam to assess the students' ability to solve realistic problems using Linked List data structures. Results from the technology-based and non-technology-based classes were collected and compared.

Responses to three questions were assessed. One question dealt with developing Java code to implement a linked-list data structure. A second question required the use of the linked-list to solve a real-world programming problem. The third question required a recursive solution to a linked-list operation.

For each question, the technology-based section had 23 respondents. The non-technology-based section had 25 respondents. Laptops were not used during the exam by students in either section.

For Fall 2007, the Department of Computer Science offered 3 sections of CSC 316. One section, taught by Rouskas, was technology-based and used the modules developed under this project. The other two sections were non-technology-based and were taught by

different instructors. These other sections were sufficiently different in terms of both content and order of coverage that it was not possible to coordinate the use identical questions in exams.

RESULTS

Effect on Student Learning

The performance of CSC 216 students with respect to the three assessment questions is summarized below.

Question 1: Provide a Java program that implements a linked-list structure.
This question was worth 25 points.

	Mean	Standard Deviation
Laptop	20.83	5.94
No Laptop	20.92	6.59

Question 2: Use the linked-list to solve a real-world problem
This question was worth 15 points.

	Mean	Standard Deviation
Laptop	10.69	3.42
No Laptop	11.12	5.01

Question 3: Provide a recursive solution to a linked-list operation
This question was worth 10 points.

	Mean	Standard Deviation
Laptop	7.74	2.45
No Laptop	7.32	3.42

The main conclusion from the above results is that technology in the classroom provided no appreciable differences in learning. However, these results do not necessarily imply that the learning module approach does not have merit. Since both sections used the learning modules, it is possible that the learning modules benefited both groups of students regardless of the technology used (laptops versus paper and pencil). Therefore, it is possible that the interactive learning experience *per se* is more important than the

underlying technology. However, without the support from COE's laptop initiative and this LITRE grant, it is unlikely that the PIs would have undertaken the task of creating the modules and making classroom learning more interactive.

Effect on Faculty/Pedagogy

- **Pedagogy.** The main difference has been in the degree of interactivity between instructor and students. Students are much more likely to ask questions when working on the in-class exercises and the instructor can immediately find out about the student's level of understanding of the concept at hand. As a result, the instructor can more effectively target problem areas during the lectures.
- **Workload.** Preparing a well-defined learning module and making sure that the relevant JAVA code works correctly is a time-consuming task. We estimate that preparing for a technology-based section takes approximately 20% more time than a non-technology-based one, with a similar increase on TA effort.
- **Content.** There were no appreciable differences in the amount of material covered, but we felt that we were able to cover the concepts treated in the learning modules in more depth than in non-technology-based sections we had previously taught.
- **Technology.** It was difficult to determine if laptops were being used in any productive way when exercises were not being done. In fact, unproductive use of laptops during class time has been an issue nationally (see, for instance, <http://www.washingtonpost.com/wp-dyn/content/article/2007/04/06/AR2007040601544.html>)
- **Satisfaction.** The PIs have been satisfied with student use of laptops for in-class exercises.
- **Training/faculty development.** As CSC faculty, we did not need any special training for developing the learning modules or using the laptops in class.
- **Support.** The main technical support we received were from our TAs, who were well-qualified for these tasks.

CONCLUSIONS

Specific lessons have you learned about student learning and technology.

Our experience shows that it is easy to overestimate what can be accomplished in 15 minutes. In our early attempts in defining the learning modules, students took significantly longer than expected to complete some of the in-class exercises. In these

cases, it took several iterations and refinements to the exercise to accomplish the desired objective within our self-imposed 15-minute limit.

We have also found that it is important to provide students with incentives for completing the in-class assignments. During the first semester one of the PIs (Rouskas) introduced the exercises, it was obvious that several students did not put serious effort in completing them as doing so did not contribute to their final grade. In Spring 2007, in-class assignments accounted for 5% of the students' grade, and the vast majority of students was actively engaged in this effort.

Specific lessons have you learned about teaching with technology.

Assessment of Technical Challenges

For programming projects, it is important to use standardized software that is readily accessible to students. Otherwise, it is likely to spend a large amount of time in-class to identify and resolve software incompatibilities.

We have also come to realize the importance of infrastructure in terms of wireless access and availability of electrical outlets in the classroom. Teaching a technology-based class in older classrooms (e.g., in Withers Hall before the renovation) was challenging and sometimes disruptive (e.g., students had to move to seats near the walls to access electrical outlets). Our experience in the well-equipped classrooms in the EB-I and EB-II buildings has been much more positive, especially as it pertains to the ability of students to access the Internet so as to download the learning modules.

Potential Applications for Others on Campus

The learning modules for CSC 216/316 are available for all CSC faculty to use in future semesters. In fact, the CSC 216 modules were used again in Spring 2007, and another faculty will be using the CSC 316 modules in Summer 2007.