Active Learning in Technical Courses

David Cordes
Department of Computer Science
The University of Alabama
Tuscaloosa, AL 35487
205.348.6363
cordes@cs.ua.edu

Allen Parrish
Department of Computer Science
The University of Alabama
Tuscaloosa, AL 35487
205.348.6363
parrish@cs.ua.edu

Key Words: active learning, teaming, cooperative learning

Abstract

The University of Alabama is one of seven schools participating in the Foundation Coalition, an NSF-sponsored partnership looking at educational reform within the undergraduate engineering curriculum. In addition to active involvement in the Foundation Coalition, the Department has re-structured several of its own undergraduate courses around the lines of cooperative learning and technology-enabled education. This includes the re-design of classrooms as well as a shift in focus from a traditional lecture to more of an "active learning" environment.

Introduction

The University of Alabama is participating in an NSF-sponsored engineering education reform project, the Foundation Coalition. The primary goals of the Foundation Coalition are to re-vitalize undergraduate engineering education through the incorporation of curriculum integration, human-interface development (active and cooperative learning), and technology-enabled education. This paper looks at the issue of human-interface development, specifically the use of teams and the teaming processes utilized in the classroom.

The use of techniques such as teaming and cooperative learning have been recognized by a number of researchers as beneficial to the classroom environment [1,2]. These techniques are fairly well-known today, and will not be re-stated in this paper. Instead, the reader is referred [3,4,5] for information on the benefits of teaming and cooperative learning.

Teaming within the Computer Science Department

While teaming is being used in a variety of courses within the discipline, this paper will focus on one specific course within the department. Briefly, the other courses that integrate teaming are senior-level courses in areas such as operating systems, database, and software engineering. While teaming plays an important role in these courses, it is quite common to see teaming in these courses in any curriculum around the country. Thus, potential audience interest in examining one of these courses seemed minimal.

The course in question is a sophomore/junior level course that is designed as part of the “bridge” between the freshman year and the upper-level courses within the major. The freshman year at Alabama is not unlike that found in any number of other institutions, focusing on fundamental programming skills and an introduction to the discipline itself. In addition, our freshman year focuses a significant amount of time during the second semester on basic algorithms and data structures.

The course under examination, CS 325 (Software Development and Systems), is designed to meet two basic objectives of the Department: (1) introduce the students to the concepts associated with object-oriented programming using Ada95 and C++; and (2) transition the students from a PC environment to the Unix workstation environment.

In the past, the main complaint regarding this course has been the fact that it is one of the more “intense” courses within the major. The amount of material within it is significant, and it has often been viewed as a “race to see if we can cover everything this semester” by the instructor. However, we were also receiving signals from the senior-level courses that some of the (what we believed to be) more fundamental issues associated with the course were not being mastered by the students. This fact, coupled with the Foundation Coalition’s emphasis on teaming and active learning within the classroom, seemed to provide an ideal testbed for integrating this new approach to the instruction of CS 325.

The remainder of this paper will focus on the use of teams in CS 325 (Software Development and Systems). During the initial semester of “active learning,” the basic philosophy utilized was to spend the first half of the course going over the basics/fundamentals of a given topic(s). At this point the lecture would stop and they would get into teams (of three or four) and work on some exercises developed around the “mini-lecture.” The course was taught in this fashion during the Fall 1994 semester. This was not true “active leaning” in its purest sense, but rather a combination of lecture and guided recitation. Nevertheless, results were positive, and it was decided to pursue active learning further the next Fall.

During the Fall of 1995, the CS 325 course is being taught in a true “active learning” environment. The class is divided into teams of four individuals, and is located in a room that the Department renovated over the summer to promote the concept of student teams. Instead of individual desks, the students now sit at tables (holding two people each). A team of four will sit at two adjacent tables, and simply turn to face each other during team exercises.

The course utilizes a combination of “lecturettes” and group exercises. Paying attention to the fact that the average adult attention span is approximately 18 minutes, the course proceeds with 15-minute lecturettes and then guided discussions or exercises by each of the groups. The example below demonstrates a class session in which students are introduced to packages in Ada95.

- Five minutes: Introduction/motivation: Review the concept of an ADT. Discuss how conventional programming languages permit operations that might violate the integrity of an abstract data type.
- Five minutes: Using a STACK, have the teams identify five different things a user might do to destroy the stack that cannot be prevented in conventional languages.
- Ten minutes: Identify the basic operations that you want users to perform with respect to STACKS (guided discussion with the class)
- Ten minutes: Have each team identify the basic operations they should consider for the following data types: QUEUE, LINKED_LIST, TREE, DIRECTED GRAPH

- Ten minutes: Overview the syntax used for operations in an Ada package specification
- Ten minutes: Have each team generate a partial Ada package specification for the data types they had identified operations for previously (QUEUE, LINKED_LIST, etc.)
- Ten minutes: Discuss the implementation of the actual data type for STACKS; overview both a static and dynamic approach to the STACK
- Ten minutes: Have the teams construct static and dynamic representations for the data stored in each of their data structures
- Five minutes: Review and summarize the day (package concept, operations visible to user, representation of internal data); What's ahead for next time: package body (implementation)

A few basic observations should now be made regarding the specific organization and structure of the teams in CS 325:

- First, the teams are selected by the students themselves. Each team lasts for approximately four weeks. However, when it comes time to pick a new team, they can not select any of the team members that they had been on a team with before. This way, initially they all know the other members on their team, and as they get more familiar with the class they became more comfortable moving to new teams.
- Second, each team is given a specific task (or, more commonly, a set of tasks) to discuss and complete each day (the same set of tasks is assigned to all teams). Students get credit for each day in which they participate, as a sheet is turned in at the end of the day containing the work done by the team during the class session that identifies the team members present. It is important for the team to have specific deliverables at the end of each day, otherwise student participation is reduced during the class exercises.
- Third, a "reading list" is handed out about every 3–4 weeks (roughly the same time frame as changing teams). On this list are specific topics for specific days, along with what material this corresponds to in the book. This way students have a reminder regarding the specifics they are suppose to be covering in the book on a regular basis. Since approximately half the time in class is spent in teams, it is not possible to cover all the material one would normally cover in the class (all the details of a concept). The students soon pick up on this, and utilize the reading list so that they will be prepared for the daily class activities.
- Fourth, the instructor must make a strong effort to keep moving between teams as much as possible during the teaming. The basic strategy is to go over to a team, listen for about 15–30 seconds, interrupt with a question or two (sometimes directed at a specific person, sometimes to the whole team), and if they are on track then move on to the next group.
- Fifth, time management is much more important than it is in a traditional course. Since the time is broken down into such small increments, it is vital that the instructor keep close track of his/her "lecturettes." The ability to talk at great length on any subject is common among faculty, and keeping focused on a given topic and covering it within the time specified can be quite a challenge.
- Finally, it is important to design the activities such that even the best of teams cannot complete them within the time allocated. The instructor should stress that the teams are not expected to complete the task, but rather to work on it for the duration of the allotted time. Idle teams can be very disruptive (as other

teams realize they are working when others are not), and so these “sponge activities” serve a very useful role in the teaming process.

Class performance in these exercises has been outstanding (to date). It should be noted that this class meets for 75 minutes twice a week (8 a.m.–9:15 a.m. Tuesday/Thursday). Most 8 a.m. classes that have been taught by the authors are a real struggle. This class is actually in their seats and ready to work every day. The change in student attitude is remarkable.

Lessons Learned

The authors are firm believers in the potential benefits of utilizing teaming in their courses. However, they have also identified several key issues that cannot be ignored when you incorporate teaming into a classroom environment. These issues are discussed below.

- It is vital that each team have some identifiable “deliverable” that is due at the end of the session. If the students do not have a specific deliverable, it is impossible to keep the team focused on the task at hand. It does not have to be a final, polished product, but the teams must turn in something to indicate the progress they made during the session.
- It is also important for the teams to realize you are interested in their efforts. This normally means that the instructor spends most of the time walking around the room and listening into the team’s conversations. Feel free to interrupt and ask the teams questions. Watch a team for a few seconds, ask some (pretty obvious) questions and (occasionally) some more difficult ones to make sure they knew what they were doing.
- Only distribute one copy of the task per team. Handing out multiple copies often led to four people dividing up the work and each doing 25%, instead of working as a team on the entire set of problems. It is a good idea of have extra copies available for all students as they leave the class, but only let the team have one copy while they are working.

Looking over the changes that a faculty member faces during their first semester in a true “active learning” environment, the following two facts seem to emerge as the most fundamental concepts it is necessary to grasp in order to operate in this new environment.

First, it is *extremely* hard for the instructor to yield control of the learning process, and let the students take responsibility for their own education. In an environment such as this, the amount of lecture time is drastically reduced from traditional classes. As a result, it is impossible for the instructor to cover the material to the depth that which he/she is accustomed. Instead, the instructor must focus on the fundamentals, provide accurate and timely reading assignments for the students, and rely on peer pressure (by the other members of the team) to help ensure that each student keeps up with the material. It is very easy to get caught in the trap of “I’ve got to tell them this, otherwise where will they ever learn it?” However, if you give the students the opportunity, they will often surprise you with their own initiative and abilities. If you expect students to develop an interest in (and ability to perform) life-long learning, you must provide them with this opportunity during their formal educational process.

Second, parallels between the format of a given class session and the 4-quadrant Learning Cycle [5] should be emphasized. The four phases of the traditional learning cycle: (1) motivate; (2) instruct: (3) develop competency; and (4) apply the principle in question. Any of the classes utilizing active learning should also incorporate these basic concepts. For example, when introducing the students to dynamic memory allocation in

Ada95, the class session was broken into four major components (with both "lecturettes" and team exercises found in each of the four segments):

- Ten minutes: Motivation of the material (teams identified why dynamic allocation of memory was sometimes needed).
- Twenty-five minutes: Instruction on the concepts. 15 minutes on allocation of memory (with team exercises). 10 minutes on the built-in Ada routines for memory deallocation.
- Twenty minutes: Applications to ensure competency on the part of the students. Students traced through a number of Ada examples (identifying the output produced) developed sample algorithms for a few simple dynamic memory allocation problems.
- Twenty minutes: Students applied the concepts of dynamic memory allocation to the development of a linked list in Ada, complete with procedures to Add, Delete, Print, and Initialize.

This procedure works very well for most (if not all) of the material presented in CS 325. In addition to providing a sense of "closure" for each lecture, the students were able to recall the basic concepts associated with each day, as it usually focused on a single "theme."

Summary

We believe that the implementation of active learning techniques into CS 325 has provided tremendous benefits to the course. The use of teams permits students within the course to not only begin to learn how to interactive effectively with other students (people), but also begins to place the responsibility for learning on the shoulders of the student, instead of the faculty member.

Future plans call for the movement of some fundamental teaming applications into the freshman year of the major. Studies will also be conducted to attempt to identify any changes in the retention rate of majors during the first two years of the curriculum. Student retention rates are at their lowest in the first two years of the major, and it is hoped that the adoption of more active-learning based techniques will help to increase overall student retention.

References

Johnson, D., Johnson, R., Smith, K., "Cooperative learning: An active learning strategy for the college classroom," *Baylor Educator*, Winter 1990, Volume 15, Number 2, pp. 11–16.

Katzenback, J.R. and Smith, D.K., *The wisdom of teams: Creating the high performance of organizations*, Harper Business Publishers, Inc., New York

Smith, K., "The craft of teaching cooperative learning: An active learning strategy," *Proceedings of the 1989 Frontiers in Education Conference*, pp. 188–192.

Bellamy, L., Evans, D.L., Linder, D.E., McNeill, B.W., Raupp, G., "Teams in engineering education," final report for NSF Grant SE-9156176, Arizona State University, Tempe, AZ.

Felder, R.M. and Silverman, L.K., "Learning and teaching styles in engineering education," *Engineering Education*, Volume 78, Number 7, April 1988, pp. 674–681.