Sustainable Peer-Led Teaching Activities in Computing Sciences

Carol Binkerd
Texas A&M University-Corpus Christi
Carol.Binkerd@tamucc.edu

Mehrube "Ruby" Mehrubeoglu, Ph.D.
Texas A&M University-Corpus Christi
Ruby.Mehrubeoglu@tamucc.edu

Abstract

With the increasing decline of students entering computer science and related programs, it is essential that as many students as possible are retained. Consequently, computing sciences departments across the nation are implementing a variety of retention initiatives. Unfortunately, especially at some of the smaller institutions, some are not easily sustainable because of budget constraints. This paper explores some of the current initiatives being implemented and their low-cost alternatives.

Introduction

The number of students across the nation entering college as computer science undergraduates has declined at an alarming rate (TAMU-CC, online; Avery, online). Consequently, most institutions are implementing various retention initiatives. Some of the current popular initiatives are: discipline-specific tutoring programs, peer-to-peer mentoring programs, and peer-led learning teams.

At Texas A&M University-Corpus Christi (TAMU-CC), all three of these have been or are being tried. And, all three were or are grant-funded activities. Unfortunately, oftentimes, when the grant money is spent, the programs are not continued. Therefore, this paper explores low-cost alternatives that can be, at least financially, more easily sustained.

Background

It is clear through the teaching experiences of the authors and the current literature that active learning, defined as 'introducing activities into the traditional lecture and promoting student engagement' (Prince, 223-235) to be effective in keeping student interest and attention. Intuitively, an interested student is more likely to remain in the program than someone who is bored in the classroom; therefore, academic retention activities, both in and outside the classroom, emphasize effective learning. There is further evidence that teams that incorporate campus culture and social interactions, thereby establishing a sense of belonging among the students, will contribute to student retention (Tonso, 25-37). The challenge, of course, has to do

with designing activities that can be implemented in the absence of additional financial resources.

M. A. Mooney and P. J. Mooney describe a student teaching-based instructional model in engineering education (Mooney, 10-16), which we have seen easily adapted in computing sciences: The Mooneys' paper was motivated by their belief that "one never really learns something until one teaches it" (Mooney, 10-16). In this paper, the proposed sustainable solutions to the retention problem also encourage peer-led learning teams and peer-to-peer learning through tutoring as well as peer-to-peer mentoring.

Sustainable Solutions to the Retention Problem

Currently, the most popular peer-led teaching activities are peer-team leaders, peer tutors and peer mentors. These activities are usually funded through grant monies. And, as a consequence, are often not sustained past the duration of the grant's funding. In the following sections, the authors have supplied examples of specific activities that professors can implement in their current classes without need for further expenditures.

Peer-Led Learning Teams

Peer-Led Learning Teams can aid in making an introductory course more exciting and interesting, especially with traditional students. Having a peer explain a concept using interactive activities, allows the traditional student to more actively participate in the learning process (Gosser, 1-133; Roth, 1-167; Berke, online). On the other hand, non-traditional students tend to view these additional activities as more of interference than a bonus. And, most often, these peer leader positions are funded with grant money, which limits the number of semesters this can be done.

But, the professor can introduce peer-led learning using the students within the class itself. For example, computer literacy classes usually draw students with a very mixed background with extremes in degrees of computer experience. A large component of most literacy classes is learning a variety of software packages. Instead of having all students complete a prescribed set of tutorials, the professor can individualize for each student what he/she will learn and surreptitiously use peer-led learning. Each student sets a list of objectives for him/herself for what skills he/she needs to learn in the particular software project. Each student must then learn the material and present a guided, hands-on lesson for all class members in these areas.

Similarly, CS1 programming classes have students with a wide range of programming experience, from none to a high school class to several years. In a lecture class of 50, one of the authors used peer-led teams for an exam review. About a week before the exam, the professor met with the students who had already taken a similar course in high school. The professor supplied the students in advance with the review questions, and explained to these students that they would be the team leaders. Before the group review, the professor met with the team leaders to answer their questions. And, after the teams reviewed, the professor met with the entire class for questions.

In another programming course, as review for an exam, one of the authors gave samples of exam coding problems. She split the class into small groups and assigned each group a coding problem to solve. Each group then presented, explained, and answered questions on the coding problem.

Usually, the CS1 course does have some students with prior programming experience. During the lab portion of the class, one of the authors has found it to be a good practice to seat the experienced students in between the less experienced. The goal, of course, is that the more experienced students will aid the less experienced. Left to their own devices, students generally choose to sit with their acquaintances. Unfortunately, their acquaintances are usually at the same skill level they are.

In order to foster a collaborative environment, within the first two weeks a team-building exercise is performed. First, the professor spends time discussing the proven effectiveness of study groups. Then, the class brainstorms about personality traits desired in study group members. Half the class is told that they will stay seated, and every other student will move. Students are then given approximately 2 minutes to "interview" each other before the movers advance to the next seated student. (Yes, this is based upon speed dating.)

Another team-building exercise in an introductory programming class is to have teams reconstruct a program. First, write a simple program and reproduce enough copies to have one per group. Cut each program into slips of paper, with one instruction per slip of paper. Have each group compete against the others seeing which group can reconstruct, compile and execute the program.

As a team-building and peer-led exercise, in a CS1 or CS2 class, again break the class into teams. Each team is assigned a section of code needed to complete a programming problem. Each team must present and explain its section of code. The goal is that once all sections of code are completed, the program will compile and execute. For example, have each team write one or two functions or methods.

A separate introductory class aimed at engineering technology students uses peer-to-peer learning teams as part of several laboratory assignments. One example of such activity involves the coding of a digit onto a 7-segment display after decoding a hidden code, and finding a digit that this code represents. The students are asked to pair up in groups of two, and are then given the cipher. They are to solve the cipher together, and come up with a word of their choice to code. The codes are then swapped among teams, and decoded. Students help each other with encoding and decoding the hidden words. The digit to be displayed is then determined according to a mathematical formula provided by the professor.

The students are encouraged to find partners such that their skills strengthen the team; students with good computer skills are asked to find partners who are good with hands-on exercises, thus encouraging students to learn from each other's use of skills and experiences. A report is prepared as team effort, a digital picture of the lit-up 7-segment display is uploaded to the computer, and included in the 'Results' section of the report (based on a collaborative

learning model (Mehrubeoglu, 1-13)). In the meantime, the students practice the intended skills, such as creativity, teamwork, working with technology, and effective communication skills.

Discipline-Specific Tutoring Programs

Most institutions have a tutoring center, at which students can receive assistance in a variety of subjects such as composition, algebra and so on. Most institutions do not, however, supply tutoring for subjects in technical areas such as computer science and engineering.

Some institutions have funding to hire students to specifically tutor in the technical areas. At TAMU-CC, during a 3-year period, students were hired to specifically tutor students in the introductory computer science courses, (CS1 and CS2). Unfortunately, at the end of the grant period, no more funding was available. Moreover, the tutoring was available in the university's central location for tutoring, rather than at the labs where the students spend most of their time and where they experience difficulty.

One low-cost idea to implement discipline-specific tutoring is to use student volunteers. The authors of the paper are currently aiding students in organizing a departmental honors society. A requirement of continued membership is that each member must contribute a minimum number of hours per week to aid in tutoring others. Student tutors will be required to be present in one of the labs during a specific time period. The on-duty tutor will be able to continue with his/her own assignments unless assistance is needed. Currently, the engineering technology students have informally instituted this practice. More senior members of the Engineering Technology Society voluntarily assist lower-level students.

Peer-to-Peer Mentoring Programs

Again, a low-cost solution is to employ the assistance of the students in the honor society. For the most part, this can be handled along with the tutoring. The honors tutor (or tutors) will be available during the prescribed time period for academic tutoring and mentoring. (Most students do not seek in-depth advising from peer advisors (Mottarella, 48-61)).

The lead author has instituted voluntary mentoring programs. The disadvantage to these is that they are voluntary. As people's schedules become hectic, voluntary activities are often put at the bottom of the priority list.

Tutoring will become part of the requirements to remain an active member of the honor society. Thus, this will no longer be a completely voluntary activity. And, the hope is that students will value their membership in the honor society sufficiently to continue with the tutoring sessions.

Discussion and Conclusions

Although some of these low-cost ideas and their implementation may not be new, their adaptation at TAMU-CC, and its contribution to diversity will be unique. TAMU-CC is classified as Hispanic Serving Institution (HSI), with a diverse group of students from the

surrounding areas. Students generally work from 20-40 hours a week while attempting to maintain a full course load, (minimum 12 credit hours). Since students are taking time-intensive computing science courses, retention is difficult.

Furthermore, at smaller institutions, funding is always an issue. If an institution is willing and able to fund student positions for peer team leaders, peer tutors (in the discipline), peer mentors, and faculty release-time to supervise the students, all to the better. Supplying student jobs is itself a means to retention. In the more likely event that this is not possible, faculty can creatively and covertly employ teaching methodologies to provide peer-led learning activities. As faculty sponsors, faculty can strongly encourage the members of student clubs to assist in tutoring and mentoring their fellow students.

References

1. Texas A&M University-Corpus Christi, Factbook Head Count Enrollment by College Major Level and Average Age, Planning and Institutional Effectiveness. Available: http://pie.tamucc.edu/
2. Avery, G. "Not adding up: Schools faced with an enrollment drop in computer-science students" The Daily Camera [online], 9 April 2007. Available: http://www.dailycamera.com/news/2007/apr/09/not-adding-up/
3. Prince, Michael "Does Active Learning Work? A Review of the Research" Journal of Engineering Education, Vol. 93, No 3, July 2004, pp. 223-235.
4. Tonso, Karen L "Teams that Work: Campus Culture, Engineer Identity, and Social Interactions" Journal of Engineering Education, January 2006, pp. 25-37.
5. Mooney, Michael A. and Mooney, Paul J. "A Student Teaching-Based Instructional Model" International Journal of Engineering Education, Vol. 17, No. 1, 2001, pp.10-16.
6. Gosser, David K., Cracolice, Mark S., Kampmeier, J. A., Strozak, Victor S., Roth, Vicki and Varma-Nelson, Pratibha "A Peer-Led Team Learning: A Guidebook" Prentice Hall, New Jersey, 2001, 1-133.
7. Roth, Vicki, Goldstein, Ellen, and Marcus, Gretchen "Peer-Led Team Learning: A Handbook for Team Leaders" Prentice Hall, New Jersey, 2001, pp. 1-167.
8. Berke, Thomas "Peer-Led Team Learning, An Active Learning Strategy that Works" National Education Association, Advocates Online [online], No. 39, Spring 2003. Available: http://www2.nea.org/he/advo03/advo1003/front.html
9. Mehrubeoglu, Mehrube and McLauchlan, Lifford "A Cooperative Learning Model in Multi-Disciplines across Universities in Freshmen Courses" Proceedings of 2007 ASEE Annual Conf. and Exposition, AC 2007-2494, June 24-27 2007, pp. 1-13.
10. Mottarella, Karen. E., Fritzsche, Barbara A., and Cerabino, Kara C. "What Do Students Want in Advising? A Policy Capturing Study" NACADA Journal, Vol. 24, No.s 1 & 2, Spring & Fall 2004, pp. 48-61.