ASEE's 123rd Annual • Conference & Exposition • New Orleans, LA • June 26-29, 2016

Paper ID #14470

Encouraging Student Innovation in a Freshman-Level Computer Science Course

Ms. Cynthia C. Fry, Baylor University

Cynthia C. Fry is a Senior Lecturer of Computer Science and the Director of the Computer Science Fellows program at Baylor University. She teaches a wide variety of engineering and computer science courses, deploys a series of faculty development seminars focused on Curiosity, Connections, and Creating Value, and works collaboratively and remotely with a series of colleagues on the development of EML-based courses. She is a KEEN Fellow.

Dr. Kenneth W. Van Treuren, Baylor University

Ken Van Treuren is an Associate Professor in the Department of Engineering at Baylor University. He received his B. S. in Aeronautical Engineering from the USAF Academy in Colorado Springs, Colorado and his M. S. in Engineering from Princeton University in Princeton, New Jersey. After serving as USAF pilot in KC-135 and KC-10 aircraft, he completed his DPhil in Engineering Sciences at the University of Oxford, United Kingdom and returned to the USAF Academy to teach heat transfer and propulsion systems. At Baylor University, he teaches courses in laboratory techniques, fluid mechanics, energy systems, and propulsion systems, as well as freshman engineering. Research interests include renewable energy to include small wind turbine aerodynamics, UAS propeller design and experimental convective heat transfer as applied to HVAC and gas turbine systems.

Encouraging Student Innovation in a Freshman-Level Computer Science Course

Abstract:

In a world where the demand is high for employees who can think creatively and apply entrepreneurial behaviors and thought processes to their work, it is critically important for engineering and computer science programs to provide more educational opportunities that take the essential basics of the disciplines and add to that content the experiences that will also encourage the development of entrepreneurial behaviors in students' development of solutions to the challenges they face. In a second-semester project-based learning course in computer science at Baylor University, the students were introduced to an idea-generation technique called Painstorming chosen to encourage opportunity recognition, and asked to develop their own idea for a semester project. This paper will cover the success of project-based learning in engineering and computer science courses, show a method of idea generation called Painstorming, the application of Painstorming to software applications as a means to generate group project ideas, the adjustments necessary for the successful implementation of this approach in an already busy course, and the preliminary results of the experiment.

An Introduction to Problem-Based Learning

Problem-Based Learning is an "instructional (and curricular) learner-centered approach that empowers learners to conduct research, integrate theory, and practice, and apply knowledge and skills to develop a viable solution."[1] Figure 1 compares traditional learning to PBL. Instead of the traditional lecture, memorize, and test, PBL is more of a discovery knowledge process which results in application of this knowledge to solve problems. PBL is an opportunity for the instructor to be a "coach" and the student to take charge of their learning. With all the technology available today to access knowledge, using technology is increasingly becoming the desired method of learning. These days it is frequently possible to observe students who ask a question and then immediately seek the answer on their smart phones. Students seem empowered with knowledge, having the world at their fingertips. Now, students need to understand how to use this knowledge and PBL offers a way to shape how students learn and apply this knowledge to carefully crafted problems in the classroom. It is thought that PBL does the following[2]:

1. Develops critical thinking and creative skills.
2. Improves problem-solving skills.
3. Increases motivation.
4. Helps students learn to transfer knowledge to new situations.

Critical thinking and creative skills refer "to the ability to analyze, synthesize, and evaluate information, as well as, to apply that information to a given context."[3] This is the heart and soul of PBL.

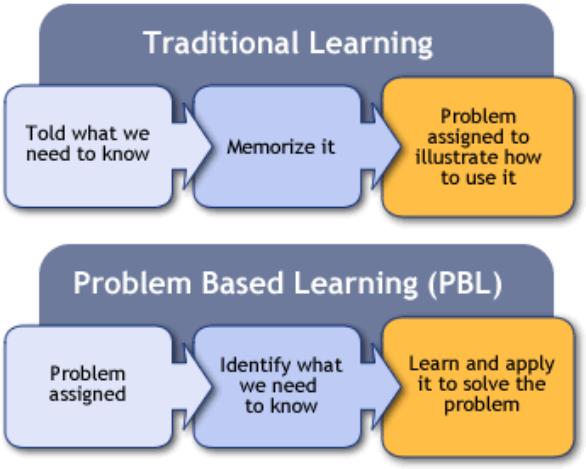

Figure 1 Traditional vs Problem-based Learning[4]

The Problem-based Learning Initiative (PBLI) identifies some generic essentials of PBL[5]:

1. Students must have the responsibility for their own learning
2. Problems must be ill-structured and allow for free inquiry.
3. Learning should cover a wide range of disciplines or subjects.
4. Collaboration is essential.
5. Self-directed learning must be applied back to the problem.
6. Closing analysis is essential to reinforce learned principles and concepts.
7. Self and peer assessment should be accomplished.
8. PBP activities must be valued in the "real" world.
9. Student examinations should measure PBL progress
10. PBL should be the basis for the entire curriculum not just one course.

The last statement is part of the challenge that faces PBL. The majority of the large number of papers presented at ASEE conferences concerning PBL highlight application of PBL for a particular classroom scenario. Typically, PBL is placed into a course by a professor, assessed, and then refined. While the students in that course are exposed to PBL, unless it is part of the entire curriculum, the skills learned with PBL are not adequately reinforced. Very few curriculums are based solely on PBL. If curriculums were based more on PBL there would be improved critical thinking and problems solving by the students, skills valued by industry.

Instructor Role in Problem-based Learning

In PBL the instructor changes from the knowledge expert to one of a coach or guide. This puts the instructor in the often uncomfortable position of allowing students the

freedom to plan their direction. Relinquishing this control is something instructors who use PBL struggle with the most.[6] Another challenge for the instructor is to develop appropriate ill-structured problems. Stanford University uses the following guidelines for ill-structured problems[7]:

1. Require more information for understanding the problem than is initially available
2. Contain multiple solution paths
3. Change as new information is obtained
4. Prevent students from knowing that they have made the right decision.
5. Generate interest and controversy and cause the learner to ask questions.
6. Are open-ended and complex enough to require collaboration and thinking beyond recall.
7. Contain content that is authentic to the discipline.

Assessing Problem-based Learning

Because this is a much different learning strategy, traditional assessment tools are not always useful. Tools that measure knowledge do not measure abilities to solve problems. The assessment technique will ultimately depend on the problem/project and the instructor's experience.[6] Gentry lists the following as possible assessment techniques[8]:

1. Portfolio of completed assignments
2. Journal containing reflections, summaries, etc.
3. Peer review
4. Scoring rubric
5. Team self-evaluations
6. Teacher observation and monitoring
7. Periodic presentations and updates
8. Written reports
9. Skills tests
10. Tests or quizzes
11. Final presentations, papers, or displays

Assessment needs to be flexible, fair and equitable, timely, and focused on the process rather than the topic.[8]

Application of Problem-Based Learning to an Intro to Computer Science Class

In computer science, as in most disciplines, group projects introduced into the curriculum early, can help students develop a host of skills that are increasingly important in the professional world.[9] A challenge, however, is in where and how to integrate the experience into the semester. In the introductory courses in computer science, the curriculum is full of necessary and essential topics – problem-solving, the basic structures of a program, the syntax and semantics of a programming language – making it difficult to find the time in lecture to include a group programming project.

Our second introductory course (CS2), CSI 1440, "Introduction to Computer Science II with Laboratory," provides an opportunity to take advantage of several of the benefits of problem-based learning, namely the tackling of larger tasks, demanding the power of an object-oriented programming paradigm.[10] Our CS2 course picks up from the first introductory course (CS1) course, CSI 1430, "Introduction to Computer Science I with Laboratory," where the basic syntax and semantics of C++ are taught, along with sequence, branch, loop, objects, classes, arrays, and searching and sorting. As such, we start with the basic tenants of dynamic memory allocation and then move into a deeper understanding of classes and object-oriented programming (string class, advanced file I/O, recursion, polymorphism, virtual functions, exceptions, templates, the standard template library (STL), linked lists, stacks, queues, and binary trees)[11], and a group project provides an excellent environment to apply what is being learned to a problem requiring these design tools.

Normally, the group project is determined for the class by the instructor, who plans the scope of the project, as well as the requirements of the project, in a way that best fits the constraints of the course and the learning objectives of the students. However, in the spring of 2015, a new approach was taken in the design and development of the group programming project, and this approach was tested in two of the six sections taught in spring of 2015 (30 of the 97 students enrolled in CSI 1440 during that semester).

"Painstorming" as an Ideation Methodology

In order to push the students' understanding of the total software lifecycle of a project, we forced student teams to select their own software design project by introducing them to an idea generation technique known as "Painstorming," as the front end of the design process. In so doing, the student had to develop a much higher-level understanding of the design challenge, along with the details required to execute the project in the fixed amount of time given. Instead of merely responding to the design criterion identified by an instructor, they had to evaluate the feasibility of various design changes based on the pains identified with an existing software application. In this way, they learned to practice some of the essentials of PBL mentioned earlier in this paper, including

- taking responsibility for their own learning,
- requiring collaboration among project team members from the beginning,
- formative and summative individual and peer assessment, and
- identifying new features of an existing software application by identifying the next pains or frustrations to be redesigned.

Painstorming as a method of ideation has been used extensively in engineering design, as a means of detecting daily hardships that might be mitigated by innovation in design.[12] It is the process of uncovering pain points to drive breakthrough innovation.[13] Instead of jumping to solutions, Painstorming uncovers

"pains" or irritations or frustrations in existing designs, providing an opportunity to redesign functionality to alleviate the source of pain. As an ideation methodology, it helps students to focus on a true pain/opportunity in the market place, increasing the likelihood of developing a high-value solution.[14]

Painstorming as an ideation methodology came out of the disadvantages of brainstorming:

- The "right idea" may not come at the right time
- Group convention may inhibit original, innovative ideas
- Team may be distracted by misdirected focus
- Domination of discussion by a few of the group members
- Aside from encouraging unconstrained thinking, there is little to actively stimulate new ideas[15]

In the spring of 2015, CSI 1440 students were introduced to the critical skill set for a successful computer scientist. These include:

- *Curiosity* - In a world of accelerating change, today's solutions are often obsolete tomorrow. Since discoveries are made by the curious, we must begin to learn how to investigate a rapidly changing world with an insatiable curiosity.
- *Connections* - Discoveries, however, are not enough. Information only yields insight when connected with other information. We must learn to habitually pursue knowledge and integrate it with their own discoveries to reveal innovative solutions.
- *Creating Value* - Innovative solutions are most meaningful when they create extraordinary value for others. Therefore, we must become champions of value creation. Part of the objective of this group project is to help you to persistently anticipate and meet the needs of a changing world.[16]

In the first two-hour lab, the students were introduced to Painstorming,[17] and were led through a brief workshop. The class was divided in half, where one half was asked to list as many pains regarding a typical school chair/desk unit (chair with book rack below, small table top connected to right side) as possible, and the other to list as many pains regarding a shopping basket (plastic, with metal handles; meant for small trips to grocery store) as possible. The teams were given 15 minutes to list as many pains as they could. Once they had finished, the two groups were switched (those who had painstormed the school desk were now tasked with the shopping basket, and vice versa) and spent another 15 minutes developing as many pains on the other item as possible. The lists for each item were combined, eliminating duplicates, and the class spent a few minutes determining which of the pains were feasible (in terms of cost, value to society, manufacturability, etc.).

Students in the class were assigned to small groups, and asked to go through a painstorming exercise where they could choose any software application, develop a list of pains, and then choose the top two or three pains they thought were feasible.

The students did this individually, then shared their chosen software application and list of pains with their teammates. As a team, they were then asked to rank each team member's idea, and either choose one as the team project, or combine aspects of several (where there was convergence in the software applications chosen).

Determining the Scope of the Projects

Probably the biggest challenge in allowing students to design their own group projects is the time involved in the determination of whether the project's scope will fit within the content of the course as well as the timeline for the semester. During weeks 3 and 4 of the semester, the small groups were required to meet with the instructor for 15 minutes. At this first "progress report," teams had to submit a preliminary scope of work and project plan. Each team presented their top idea (and their alternate project ideas), answering questions about design approach and feasibility. Based on this meeting, if the project was well thought out and feasible, given the experience of the students and the time remaining in the semester, the scope of each project was either approved or revised.

During week 9 project teams had to schedule a second progress report to finalize the design and planning for their projects. Student teams submitted a refined scope of work and project plan, discussing work completed, challenges anticipated, and their plan for completion.

At the end of the semester, each team presented their final projects to the entire class. At the beginning of their presentation, each team presented the client/instructor with the final report. Required elements of this final report were:

- Software Application
    - Design documentation
    - Source code
    - Executable
- Documentation
    - Statement of Work
    - Purpose
    - Benefits
    - Project Plan
    - User's manual
    - Systems requirements
- Presentation
    - Oral presentation (using tools like powerpoint, prezi, etc.)
    - Demonstration of code
- Final Report
    - Description of application
    - Discussion of how the team arrived at final design
    - Design
    - Test plan
- Appendix A – Each member's original design

- Appendix B – Source Code

These items form the assessment instruments that were measured and compared to the students in the control group (sections 3-6 of the spring 2015 term).

Assessment of Student-Chosen Projects
Several assessments were conducted throughout the semester to measure the effectiveness of student-chosen projects, including a pre- and post-survey on the fundamentals of project idea generation, formative assessment of peers, summative assessment of peers, and rubrics used for both the final design and the final presentation.

A pre- and post-survey were conducted to measure basic knowledge regarding curiosity, making connections, and creating value with regard to generating their own ideas for a CS2 software project. In the pre-survey, conducted during the first week of class, questions included:

1. On a scale of 1 (not curious at all) to 10 (extremely curious), rate your curiosity:
2. On a scale of 1 (I'm not curious at all) to 10 (I use curiosity in my work all the time), how often do you consciously use curiosity in the work that you do as a student?
3. On a scale of 1 (not at all) to 10 (all the time), how often have you connected the content you are learning to the world around you?
4. On a scale of 1 (not at all) to 10 (all the time), how often have you realized your own potential to create value through what you do as a computer scientist?

The same set of questions were asked, along with some open-ended questions regarding improvements, during the last week of classes. The results showed an encouraging improvement, in the students' own perception, in their awareness of curiosity, making connections, and creating value within their discipline.

CSI 1440 Spring 2015 Pre-/Post-Survey Results

Questions	Pre-Survey Class Average	Post-Survey Class Average
On a scale of 1 (I'm not curious at all) to 10 (extremely curious), rate your curiosity.	6.1	8.2
On a scale of 1 (I'm not curious at all) to 10 (I use curiosity in my work all the time), how often do you consciously use curiosity in the work that you do as a student?	4.9	5.5
On a scale of 1 (not at all) to 10 (all the time), how often have you connected the content you are learning to the world around you?	7.4	8.7
On a scale of 1 (not at all) to 10(all the time), how often have you realized your own potential to create value through what you do as a computer scientist?	4.6	6.8

The formative peer assessment, the summative peer assessment, as well as the rubrics for the presentation and the final report can be found on the course website.[11]

Future of Problem-based Learning

Problem-based learning is a pedagogical technique that will be part of the academic process for years to come however, change is slow. For PBL to be successful requires a mindset be developed from elementary school through the university. Unfortunately, many public school curriculums are driven by state-sponsored mandatory testing preparation for these tests is addressed by schools using traditional lecture and memorization techniques. These are the skills and experiences that form the foundation for students and that are brought to the university instead of a foundation in creative problem solving and independent learning. A creative approach to solving problems and the ability to use all the technological resources available at one's disposal will be essential for the future success of today's university students. No one knows what the future will hold and students will need to be flexible to adapt to new technologies and methods. This will lead to life-long learning, something that is necessary for the student to stay viable for the future.

References

1. Savery, J. R., 2006, "Overview of Problem Based Learning: Definitions and Distinctions, Journal of Interdisciplinary Journal of Problem-Based Learning, 1 (1), pp. 8-20.

2. Problem-based Leaning, www.learning-theories.com/problem-based-learning-pbl.html, accessed on January 29, 2016.

3. Gallow, D., "What is Problem-based Learning?" www.pbl.uci.edu/whatsipbl.html accessed on January 29, 2016.

4. Graphic from www.presentlygifted.weebly.com/problem-based-learning.html, accessed on January 29, 2016.

5. Barrows, H., "Problem Based Learning Initiative," http://www.siumed.edu/dme/PBL-Home.html, accessed on January 29, 2016.

6. Utecht, J.R., 2003, "Problem-Based Learning in the Student Centered Classroom," www.jeffutecht.com/docs/pbl.pdf accessed on January 29, 2016.

7. "Problem-based Learning," Speaking of Teaching, Stanford University Newsletter on Teaching, Winter 2001, 11 (1).

8. Gentry, E., 2000, "Creating Student-centered, Problem-based Classrooms," ASPIRE, Huntsville: University of Alabama, www.aspire.cs.uah.edu, accessed on January 29, 2016.

9. Caruso, H.M., and Wooley, A.W., 2008. "Harnessing the power of emergent interdependence to Promote Diverse Team Collaboration," Diversity and Groups, 11, pp. 245-266. https://www.cmu.edu/teaching/designteach/design/instructionalstrategies/groupprojects/benefits.html, accessed on January 31, 2016.

10. Drury, H., Kay, J., and Losberg, W., "Student Satisfaction with Groupwork in Undergraduate Computer Science: Do Things Get Better?" Australasian Comuting Education Conference (ACE2003), Adelaide, Australia. http://sydney.edu.au/engineering/it/~judy/Homec/Pubs/ACE_Drury_120.pdf, accessed on January 31, 2016.

11. Fry, C.C., Course materials for CSI 1440, "Introduction to Computer Science II with Laboratory," https://classnotes.ecs.baylor.edu/wiki/CSI_1440_Fry_Spring_2015, accessed on January 31, 2016.

12. Kaplan, S., "Forget Brainstorming. Use Painstorming for breakthrough innovation." http://www.leapfrogging.com/2013/06/20/painstorming-for-innovation/, accessed on January 30, 2016.

13. Ryckman, P., "Fostering entrepreneurs, and trying to revive a city." http://www.nytimes.com/2010/06/24/business/smallbusiness/24sbiz.html, accessed on January 31, 2016.

14. Daimi, K., "Strengthening Elements of Teamwork, Innovation, and Creativity in a Software Engineering Program," The Journal of Engineering Entrepreneurship, Volume 3, Number 1, 2012.

15. Weaver, J.M., "Brainstorming and Painstorming," Presented at End-of-Semester workshop at Baylor University, Waco, Texas, December 2010.

16. "Entrepreneurial Mindset Defined," the Kern Engineering Entrepreneurship Network, http://www.kffdn.org/files/keen-entrepreneurial-mindset-defined.pdf, accessed on January 31, 2016.

17. Fry, C.C., "Brainstorming and Painstorming," Powerpoint adapted from Jonathan Weaver's presentation in 2010, designed as a module for CSI 1440 lab, spring 2015, https://classnotes.ecs.baylor.edu/w/images/4/48/PainStorming.pdf, accessed on January 31, 2016.