Developing User Interface Computation and Peer Instruction: Assessing Results on Engagement, Retention, and Failure Rate for CSCI 1300 Introduction to Computer Programming

Submitted by Jeffrey La Marche, doctoral student in Department of Computer Science (CSCI)
Faculty Advisor: Dr. Tom Yeh, Assistant Professor in Department of CSCI

Goals & Objectives

Research Question: How does the inclusion of a novel contextualized teaching and learning method and peer instruction impact the engagement, retention, and failure rate of the CU Computer Science (CS) Department’s CSCI 1300 Introduction to Computer Programming course?

The goal of my PhD dissertation research is to understand what effects introducing peer instruction (PI) and a novel contextualized teaching and learning method called user interface computation (UIC) will have on levels of student engagement, retention rate, and failure rate of CSCI 1300. One objective of this research is a major revision of the CU Boulder CS Department’s CSCI 1300 Introduction to Computer Programming course. The CS department undergraduate curriculum committee currently has future plans to update CSCI 1300 to teach the Python and Java programming languages, which are two languages CS students will encounter throughout their required coursework. UIC is based on a scripting language called Sikuli (http://www.sikuli.org/) that works with the Python programming language (http://www.jython.org/), which incorporates both the Python and Java languages. Additional objectives include increasing engagement and retention in CSCI 1300 as well as reducing student failure rates in the course.

Motivation

Enrollment rates for computer science degrees have primarily experienced decreases since the Dot-Com Bust of 2001, when computer jobs were drastically eliminated, the economy entered a recession, and available CS jobs began to be sent overseas to save money [8]. By 2020 there is expected to be a 30% increase in the number of jobs available for software developers in the US [1]. The number of computer science students has not risen to meet projected needs and it is expected that there will be a deficit of qualified computer scientists to meet demands. By 2018 the prediction is that only 61% of the newly available computing jobs can be filled by graduating students. The situation improves slightly by 2020 where 72% of the available jobs can be filled by college graduates [9, Fig 1]. A further disheartening statistic is the number of women and minorities that are computer science majors. Based on a Computing Research Association report, during 2010 - 2011, of all the CS degrees awarded 11.7 % were awarded to females and 33.1% were awarded to minority students [16].

If female and minority enrollment could be bolstered this could help improve the future deficits of qualified computer workers.

One method of attempting to increase enrollments and diversity in CS is by making the introductory computer science courses more engaging for students. At CU Boulder, CSCI 1300 is the first course that computer science majors take and the course can fulfill computing requirements in some other engineering departments, such as Aerospace Engineering. Enrollment for CSCI 1300 regularly exceeds 200 students per semester with the combined majors and non-majors sections. Non-majors that take this course come from engineering departments primarily and during the summer there have been graduate students, high school students, continuing education students, and students from other areas outside of engineering (e.g., linguistics and molecular and cellular
biology). These students take the course because they might need to learn to program for an aspect of their research, for their job, as a requirement for another class, or possibly because they are merely interested and are exploring different potential areas they could major in. CSCI 1300 may also be the first time a student has ever attempted to program a computer and combining this fact with the course being taken by a wide range of students it is therefore a crucial gateway for enticing students to become a CS major, to stay in the major, or for non-majors to take additional CS courses. Currently CSCI 1300 is taught using many different methods depending on who is teaching the course. Sometimes the course is taught with only one programming language, other times there are two languages, PowerPoint may be the primary lecture method, and occasionally iClickers are used. Overall the class is lecture focused with a short 50 minute recitation period once a week.

Revising and updating the CSCI 1300 course is particularly salient now as in November 2012 a new bachelor of arts (BA) degree was created at CU Boulder for students in the Arts & Sciences (A&S) program. Current projection is that about 40 students will be enrolled in the program for fall 2013 and within five years the enrollment could exceed 150 students. Students in the A&S CS BA program will potentially be majoring in CS for very different reasons than someone in the Engineering program. It is also likely that the student population from A&S will be more diverse than from Engineering. What these students find engaging will be different than someone coming from a purely engineering perspective and offering content that is not primarily math and textually based programming could be more enticing and engaging.

**Theoretical Framework**

The main theoretical grounding for this research is Peer Instruction (PI) and Contextualized Teaching and Learning (CTL), both of which use constructivism for their frameworks. In constructivist theory the learner actively constructs their own knowledge about topics through experiences, which means that knowledge is not merely imparted on the learner in a non-active manner, such as by lecture. Constructivism has several originators including John Dewey and Lev Vygotsky. Dewey’s theory regarding experience as being a key component to successful education and Vygotsky’s Zones of Proximal Development provide major contributions to the constructivist framework [5, 12, 15]. Constructivism contains four key components [12]:

1. Provide students with material of a familiar context
2. Guide students in their learning to help them expand their skill levels
3. Decrease reliance on the instructor
4. Include collaborative learning activities

Peer Instruction (PI) is a pedagogical technique, invented by Harvard University physics professor Eric Mazur, that is designed to promote and increase student engagement in the classroom [4]. In PI, there is assigned reading before each class, there is a short content knowledge assessment (such as a quiz), and several clicker questions are asked during class. These clicker questions focus on more challenging course material that can easily be misunderstood or is difficult to master and the questions are designed to promote student critical thinking and discussion. PI directly addresses constructivism components (2) by providing students with exploration questions that require students to use their knowledge and also to expand it to a deeper level, (3) by having students rely on each other and their own preparations to formulate knowledge that they require for success in a course, and (4) emphasizing collaboration to create a shared understanding of the material with discussion of topic questions. PI has been implemented and shown to have amazing results for introductory CS courses at numerous other universities [10, 11] including the University of California San Diego, University of California Davis, University of Toronto, Harvard University, and the Technical University of Denmark. At CU, some introductory sciences, particularly in the Physics Department, have used PI and this has never been tried for CS. My research will be the first serious attempt to apply PI for CS at CU.

Contextualized Teaching and Learning (CTL) is a technique that relates relevant real world material to what a student is learning [7], thus invoking the first component of constructivist theory. One form of contextual learning that is being used for computer science education at other academic institutions is media computation (MediaComp), which was created by faculty of the Computer Science Department at Georgia Tech in response to concerns about student retention in introductory computer science courses [13]. MediaComp uses the familiar context of computer media to teach computer programming and some examples are digital images, music, and video. As with PI, Media Computation has been shown to have excellent results in introductory CS courses at other universities [6] and is used at places such as Georgia Tech, Duke University, and University of New Mexico. As part of my research I will be developing a novel CTL called User Interface Computation (UIC) and I believe it will be more effective than MediaComp.
Methodology
Implementing peer instruction for CSCI 1300 will primarily model the methods employed successfully in the Computer Science Department at the University of California San Diego [14] and it will also contain elements from Mazur's implementation [4]. The CSCI 1300 course implementation of peer instruction follows these steps:

1. Students are assigned readings that need to be completed before the next class.
2. Before class students have a reading assessment administered through Desire2Learn.
3. During class students are asked questions that challenge and explore topics that they read before class and that were shown to be problematic based on the reading assessment. The question format is as follows:
   a. Students answer a question individually.
   b. Students have small group discussion emphasizing students justifying their answers to each other.¹
   c. Students re-vote based on their discussions.
   d. Students are asked to share their answer justifications and explanations with the whole class.
   e. Instructor leads discussion of the question topic.

The students can be shown the pre-discussion and post-discussion voting results at different times in the question sequence, with doing so after 2c and 2d being the most common, and a mixture of both methods will be used in CSCI 1300. Lectures do still occur in a PI environment and are greatly reduced from the traditional lecture sage on stage model. In PI there will be one short (15 minutes) lecture segment or several smaller ones interspersed into each course. Developing the actual clicker questions is a key component to making PI successful and to aid in this best practices will be used [3] as well as utilizing content already available to act as templates for questions specific to the CSCI 1300 UIC implementation [2].

In addition to creating PI resources I will create the new User Interface Computation curriculum and I will utilize UIC when teaching the CSCI 1300 course. As UIC is a novel teaching method I will create all of the new curriculum and this will include developing a series of new lab exercises and projects, developing examples and mini-lectures, generating PI materials, and producing supplementary readings for helping students learn UIC concepts. With MediaComp, computer media is used to teach programming and by having students explore how to create and modify media such as using loops to edit an image pixel-by-pixel. UIC further changes the way a student learns to program by using the computer graphical user interface (GUI) to contextualize computer programming. The GUI provides an even more familiar context for students than media as all students have used a GUI, while not all students may have edited a digital image. Using UIC the actions that a programmer would take to manually complete a task on a computer strongly correlates to how the computer will be programmed to complete the same task. This strong correlation exists because the Sikuli programming language uses GUI images to take actions that correspond to the same graphical features a programmer would see and interact with to complete the same task. For instance, Table 1² below shows a mapping of a user’s actions and the corresponding Sikuli Code they would need to write to click every visible “Like” link on the user’s Facebook News Feed. Facebook and liking posts is a highly familiar context for student’s today.

<table>
<thead>
<tr>
<th>Context</th>
<th>User Action</th>
<th>Sikuli Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>On Facebook’s user interface (UI), a student wants to click “Like” on every post currently visible on her News Feed.</td>
<td>Find each Like on the screen.</td>
<td><code>all_likes = findAll(&quot;facebook_like.png&quot;)</code></td>
</tr>
</tbody>
</table>
|                              | Left-click on each Like that is found. | `for like in all_likes:`
|                              |                                          | `click( like)` |

Table 1: Facebook Example of User Actions Mapped to Sikuli Code

¹ In some classes discussion groups and seating are assigned and currently this is not a planned research method.
² In the User Action column the “Like” is a graphic named facebook_like.png that was captured from the actual Facebook UI.
Analysis
Analysis of the integration of PI and CTL into CSCI 1300 will consist of three aspects: engagement, failure rate, and retention. Engagement is the most difficult aspect to measure as it is highly subjective. Qualitative and quantitative engagement data will be gathered by anonymous student surveys using the Desire2Learn online course system or a survey website such as Survey Monkey (http://www.surveymonkey.com). Question types on the surveys will include short answer, multiple choice, and Likert scale. These surveys will be sent to students and questions will ask about how much they believe that they are learning and their engagement level from clicker questions, talking with their peers, reading class materials, course assignments, and from lecture. Further quantitative data will be obtained from class grades and from clicker question answers for the individual and group responses. This data will be used to revise the course and also be compared to results obtained between teaching in the fall and spring semesters.

Failure rate will be assessed on two criteria: (1) students that receive a final grade of F and (2) students that receive anything below a C, as typically anything below a C does not count toward requirements and a student would need to take the course again. Retention rate will be measured by collecting the CU reported current class enrollment data at the end of each day. When a student requests to be added or dropped after deadlines this information will also be kept. Additionally, at the end of the course, students that clearly have not been a part of the class and still appear on the enrollment documents will be counted as having dropped the course. An example of this would be when a student has no grades since the first month of the course. To provide further insight into reasons for dropping CSCI 1300 students will be asked to provide a short explanation. The data collected for failure rate and retention rate will be analyzed with previously taught course iterations.

Proposed Timeline
This research project will start in the summer 2013 and continue through the 2013-2014 academic year. Summer 2013 will consist of a pilot trial of the revised CSCI 1300 course (PI + UIC) during summer term A. The data from the summer will not be used for overall research results because the summer course is highly intensive and will offer different results than during the normal academic year. The summer data will be used for examining how PI and UIC work in the course and what types of revisions will help with teaching in the fall and spring. During the summer all surveys and initial curriculum will be finalized and data collection documentation will be submitted to the CU IRB for approval. After the course I will officially propose my dissertation topic.

The fall 2013 semester will be my first teaching using the new PI and UIC curriculum I develop. Data will be collected and evaluated on an ongoing basis. Changes will be made to tweak the course content if problems arise or if trying something new may be beneficial. At the end of the fall semester and during winter break all data will be analyzed and changes to the curriculum for the spring semester will be made.

The spring 2013 semester will be my second teaching of CSCI 1300 using the PI and UIC methods I’ve developed and will consist of revised content up to this point. Again data will be collected and analyzed during the course and changes will be made if needed to the course content. Toward the end of the semester I will begin writing papers to discuss the results and they will be submitted to appropriate journals and conferences including the Association for Computing Machinery (ACM) Special Interest Group on Computer Science Education (SIGCSE), Innovation and Technology in Computer Science Education, and the International Computing Education Research (ICER). Research findings will also be disseminated to the CU STEM community.
**Project Outcomes**

**Personal Development**
The completion of this research project will benefit my personal development by providing data that is necessary for finishing my PhD thesis dissertation. Additionally, publications based off of this research will help me become known in the CS education research field, researching peer and contextualized instruction will benefit my pedagogic skills and help me engage students more fully, and lastly, increasing my teaching exposure will strengthen my teaching portfolio when applying for teaching focused professor positions after I graduate.

**Home Department Benefits**
The Computer Science Department will benefit from this research by creation of a new curriculum that aligns with the current goal of redesigning the course to make use of the Python and Java programming languages and the PI and UIC methodologies can be adapted and incorporated into other CS courses. Additional potential benefits include improving retention, increasing enrollment, reducing failure rates, and increasing engagement. The course will also be more inviting to students that come from the College of Arts & Sciences and students that do not have a highly technical background. It is believed that the redesign of the course will appeal more to women and minorities and could lead to an increase in computer science majors in both groups at CU Boulder.

**The CU STEM Community and Beyond**
The CU Community will benefit by this research as computational thinking and programming skills are needed by a broad audience of students and having an engaging and accessible course available will provide an enticing option for students of any major to gain academic and life skills that they will need for the 21st century. The broader worldwide academic community will also benefit from this research as course materials are going to be made freely available via the Learn Programming with User Interface Computation website (http://csci1300sikuli.weebly.com/index.html).