John Fernandez, Texas A&M University-Corpus Christi
Dr. Fernandez is Assistant Professor of Computer Science in the Department of Computing and Mathematical Sciences. Having served 20 years in the U.S. Air Force and 10 years in private industry, Dr. Fernandez brings real-world experiences into the classroom for his students. His research interests are in HCI, information assurance, and software engineering.
Computer Science Capstone Courses Supported By Industry-Based Software Tools

Introduction

Service learning or civic engagement is a goal being pursued by many institutions of higher education. This goal is addressed by computer science (CS) and information technology (IT) programs which encourage or require some form of real world experience. However, students in computer science and in other science fields are not typically attracted by community or real world projects. Faculty and staff in these disciplines have a responsibility to connect students with the community and the world that they will support upon graduation. This paper describes a paradigm for community-based capstone courses that uses industry-sanctioned software engineering support tools. A discussion of the supporting pedagogical approaches used for very successful capstone projects that have been completed by computer science students at Texas A&M University-Corpus Christi (A&M-CC) are presented. Challenges associated with the software products are presented as well as solutions that use industry grade software development support tools for the entire life cycle. The capstone course experience can truly be the capstone of a student’s education, while also having a beneficial impact on the community.

Conflict of Mindsets: Traditional Versus Engaging

The author has several years of experience developing and managing the development of systems for many organizations, and can attest to the existence of a certain mindset among CS professionals. Even while teaching software engineering courses, the author has heard comments from students, such as, “users are idiots.” Obviously, these comments are meant as jokes, but it is clear that many students carry a bit of an attitude against the non-computer-oriented community. In the past 25 years, little has changed in the attitudes of students pursuing degrees in computer science or software engineering. In fact, little has changed since the NATO conference which was the genesis of software engineering.

It is a fact that the majority of computer science graduates will not be developing large NATO-type projects, but rather much smaller systems with interactive components. Therefore, a user-centered development methodology like human-computer interaction (HCI) is a much needed discipline within all computer science programs. This discipline is the basis for an engaging mindset. A detailed discussion of this topic may be found in Fernandez. However, one should mention some of the most commonly cited authors in this area: Preece et al., Shneiderman & Plaisant, McCracken and Wolfe and Rosson & Carroll.

Conflict of Mindsets: Programming Versus Analysis & Design

Alan Cooper relates his experience with programming as such a difficult and absorbing task that it dominates all other considerations, including the concerns of the user. Fernandez provides an approach to close the gap with user-centered software engineering or HCI education. However, the challenges are bigger than this. In fact we can safely state that the concerns are international in scope.

From the author’s experience, most computer science students and professionals believe that programming is the most important skill that should be emphasized. One student commented, after
receiving a poor grade in a capstone course requirements analysis report, that he thought one only had to be a good programmer to be a good computer science professional. In fact, some senior faculty members at the author’s institution have commented that programming is the most significant skill required of computer science majors. The author contends that programming is important, but the skills of analysis and design are the most significant skills required in the real world.

**Industry needs are changing**

The field of computer science is constantly evolving and needs are ever-changing. The skill set that once provided students with the proper amount of competence to perform their jobs is inadequate in industry today. Most universities are not able to adapt to these changes by molding computer science curricula to advance the student’s skill set in order to produce graduates who meet employer needs. Researchers and industry leaders are beginning to recognize a deficiency in computer science curricula. It appears that, on the average, computer science graduates are not being sufficiently equipped for the business world.

Researchers who studied software practices around the world concluded that India and Japan rated much better than the U.S. in defining software specifications, developing detailed designs and conducting design and code reviews. The table below is an excerpt from this study and illustrates the practices used by companies in various projects. The differences highlight the concerns of industry leaders.

<table>
<thead>
<tr>
<th>Number of Projects Submitted</th>
<th>India</th>
<th>Japan</th>
<th>Europe</th>
<th>US</th>
</tr>
</thead>
<tbody>
<tr>
<td>24</td>
<td>27</td>
<td>22</td>
<td>31</td>
<td></td>
</tr>
<tr>
<td>Architectural specifications (%)</td>
<td>83.3</td>
<td>70.4</td>
<td>72.7</td>
<td>54.8</td>
</tr>
<tr>
<td>Functional specifications (%)</td>
<td>95.8</td>
<td>92.6</td>
<td>81.8</td>
<td>74.2</td>
</tr>
<tr>
<td>Detailed designs (%)</td>
<td>100.0</td>
<td>85.2</td>
<td>68.2</td>
<td>32.3</td>
</tr>
<tr>
<td>Design reviews (%)</td>
<td>100.0</td>
<td>100.0</td>
<td>77.3</td>
<td>77.4</td>
</tr>
<tr>
<td>Code reviews (%)</td>
<td>95.8</td>
<td>74.1</td>
<td>81.8</td>
<td>71.0</td>
</tr>
</tbody>
</table>

Table 1. Projects (%) of Various Worldwide Companies Using Designated Practices

By reviewing the data in Table 1, it seems reasonable to see why U.S. companies may feel comfortable with outsourcing of software development to other countries. Outsourcing research appears to indicate that computer science and information technology curricula lack sufficient experience and education in the area of software design. Programmers graduating from off shore universities which appear to emphasize this experience and education are an appealing resource to industry leaders.

No one is naïve enough to argue that U.S. students’ lack of these skills is the only reason for outsourcing. Outsourcing is also attractive to industry because of the cost benefit. In a study done by Gartner Research in 2005, it is reported that five percent of information technology jobs in developed countries have been sent offshore, and by 2015 that number will increase to an estimated thirty percent. As shown by other studies, depending on the type of project and level of difficulty of the tasks, programmers sometimes produce as few as sixty lines of code per month and rarely produce more than a few thousand. Given that average programmer salaries in the U.S. are about $66,000 and that employees in offshore companies receive far smaller pay compensation, an estimated thirty percent cut in development costs can be achieved by outsourcing. Due to the basics of business, it is obvious that a
lower cost to the employer for the same amount of work drives outsourcing as the solution that best benefits the employer.

To combat these issues, universities must place more emphasis on software design and development in computer science curricula. This will result in more productive software developers, thereby reducing development cost, and providing graduates with the skill set needed in industry. In fact, industry members of Texas A&M University-Corpus Christi’s Computer Science Advisory Board have commented on the need for graduates to have strong skills in systems analysis and design, which means using software engineering CASE tools. However, maintaining a curriculum in line with employer needs that requires industry tools can be an expensive process.

**Shared Software Infrastructure Project**

The Open Standards SSI Hub Project at Texas A&M University alleviates the expense of incorporating additional material into the curriculum. At no cost to participating universities, SSI Hub supplies resources that facilitate the task of empowering students with the experience of using industry tools to develop cutting edge industrial strength software, while still educating students in software design theory. The SSI Hub is able to provide these resources through partnerships with industry leading companies, such as IBM, who appreciate the need for cohesion between academia and industry. These sponsors contribute to the information pool by providing tools for the following areas: requirement tracking tools, integrated development environment, relational database management systems, project tracking, web server technology, configuration management systems, defect tracking and change management, testing tools, visual monitoring, and visual modeling.

In addition to supplying software to participating universities, the SSI Hub also provides technical assistance and tutorials for the software, suggestions and sample assignments on incorporating the software into the curriculum, and links to sites relevant to the software. Universities, aware of employer needs, are using these resources to provide students with experience of real world practices. This translates to graduates equipped with the knowledge and skills needed to smoothly and competently transition from the role of student to the role of employee. The method of education through experience is most effective if students are able to use tools similar to or the same as those used in industry. The tools made available by the SSI Hub are the industry strength tools students need. The vision of the SSI Hub project is to continually make advanced knowledge available to students through a partnership of industry and university leaders. Students, upon entering the corporate world, will be active and effective contributors to the evolving industry of computer science.

A&M-CC is using some of these tools, and planning to use others, for graduate and undergraduate software engineering courses, as well as senior capstone courses. The SSI Hub website, as well as other sources provide excellent examples of realistic experiences for students. Several institutions have become members of the SSI project and are using SSI tools to enhance graduate and undergraduate software engineering and associated courses. Information about the individual university programs can be found on the website.
It should be noted that the theoretical basis for most of the SSI Hub tool sets can be attributed to the Software Engineering Institute’s CMMI model and associated research. Information about this work may be found at http://www.sei.cmu.edu/cmmi./17

Tools for Software Engineering Industry Experience

Many of the tools provided by the SSI Hub are IBM products. Table 2 below contains a listing of the major tools, most of which are part of IBM’s Rational Suite of products. Several other tools are available on the SSI web site15, but Eclipse, Rational Software Architect, and Rational ClearCase appear to be the most commonly used for software projects. Notice that IBM’s DB2 is also available for use within the university curriculum. Rational Rose may be familiar to some readers and it should be noted that the name has been changed to Rational Software Architect.

<table>
<thead>
<tr>
<th>Tool</th>
<th>Purpose</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requisite Pro</td>
<td>Requirements Tracking</td>
<td>Rational</td>
</tr>
<tr>
<td>Eclipse</td>
<td>Integrated Development Environment</td>
<td>Open Source</td>
</tr>
<tr>
<td>DB2</td>
<td>Relational DBMS</td>
<td>IBM</td>
</tr>
<tr>
<td>ClearCase</td>
<td>Configuration Management</td>
<td>Rational</td>
</tr>
<tr>
<td>ClearQuest</td>
<td>Defect Tracking &amp; Change Management</td>
<td>Rational</td>
</tr>
<tr>
<td>Robot</td>
<td>Automated Regression Test Tool</td>
<td>Rational</td>
</tr>
<tr>
<td>PurifyPlus</td>
<td>Error Detection Tool</td>
<td>Rational</td>
</tr>
<tr>
<td>Project Console</td>
<td>Measurement Monitoring</td>
<td>Rational</td>
</tr>
<tr>
<td>Rational Software Architect</td>
<td>Visual Modeling</td>
<td>Rational</td>
</tr>
</tbody>
</table>

Table 2. Sample Tools on SSI Hub Web Site

When leaving a university, on average, many college graduates have an inadequate grasp of the theory of computing and a lack of practice and practical experience utilizing the science and technologies of computing. Unfortunately, this deficiency in real-world experience is often times true. Universities concentrate on teaching the core subjects associated with computer science without adequately reacting to the dynamic nature of real world computing and the evolutionary aspects of technology. Although students are being taught logic and methods to design programs, most computer science curricula have not evolved sufficiently to incorporate new technologies. Just as students are no longer being taught to program using punch cards, design concepts must no longer be taught with concepts alone. The advancements in software engineering tools and techniques, such as those in Table 2, must become part of the core of computer science curricula.

IBM’s Ishigaki shows the integration of Rational tools across the life cycle of a typical software system in industry in the figure below. This also indicates how a university may be able to incorporate Rational tools across the curricula.
Incorporation of tools, such as those offered in the SSI Hub, into computer science curriculum is critical in the process of educating students of the highest caliber. The industrial strength power of Eclipse, IBM Rational Software Architect, and IBM Rational ClearCase and other tools empower students with the knowledge and experience needed for a productive future in industry. Not only does the use of these products leave more time for mastering critical concepts in the classroom, the hands-on experience of these products intimately familiarizes students with tools being used to develop real-world applications. Few, if any, companies use simple text editors or development environments solely for one particular language, which is typically of many universities which teach programming concepts. Taking advantage of such tools emphasizes software engineering analysis and design, resulting in students who are more marketable and more confident when they make the transition from academia to industry.

Outsourcing is a part of the global environment and it will continue as a fact of doing business. However, students who develop expertise with software engineering industrial tools will be assured of jobs throughout their careers. Working with clients to define requirements, specifications, and design cannot be outsourced. Professional software engineers will be the architects of future software systems. Programming, like home construction, can be done by just about anyone. What is really critical is to get the design blue print done right. This is the key to a successful software engineering industry in the U.S. The SSI Hub is an important step supporting this effort.

**Community-Based Capstone Education**

The author has been the instructor in four capstone courses at A&M-CC in which all students work on projects for non-profit organizations and a few that are university-related. Before initiating a capstone course, contacts are made with city, school, and university organizations in order to find CS type requirements that are real and can provide the basis for student projects. The types of projects that seem
to be most common are Web-based database systems with interactive components. The principals in each of the project offices are visited in order to get a better understanding of the needs and to explain to the users the process students would use to complete their projects.

Students complete a formal requirements document which captures all of the client’s needs with uses cases, data flow diagrams, E-R diagrams and other modeling approaches. The requirements are presented in class for other students to ask questions and participate in an informal review of the system. After visiting with the clients a second time and presenting the use cases to verify that the requirements have truly been captured, the project teams move to the next step of designing a solution. Use cases are enhanced, the DFDs are refined, structure charts are developed, main screen shots are drafted and other design models are included in a formal design document that the student teams present in class for an informal review of the design. The users are shown draft main screen shots and are asked for feedback on the functionality that is being incorporated and how the design views the system at this point.

Software engineering is a prerequisite for the capstone course and the author experimented with his graduate software engineering class to incorporate RSA into the course. Students were provided copies of the RSA software using portable hard drives and brief installation instructions. Since students chose to do the work on Windows platforms, only Windows installation instructions were provided. The installation guide was provided to make the process easier and clearer for the students because they were provided no in-class instruction on RSA beyond a 30 minute lecture introducing SSI and its tutorial and other resources. The students were greatly challenged while learning to use RSA. All students thought the experience was very good. Some recommended a two to three hour lecture introduction to the tool would have been useful to cut the learning time. However, the graduate students completed the assignment so the experiment was considered a success.

Future undergraduate classes will incorporate RSA into the software engineering course with the instructor spending time to explain the package and its capabilities. This will provide students the facility to complete the analysis and design segments of the capstone course with greater efficiency. The ultimate objective is a well-designed project that will lead to an effective implementation for the real-world client.

The author has started to use the ClearCase repository to capture all capstone project code and documentation. Although no client has requested modifications of the systems built by capstone students, modification should be part of the services offered by the department through student projects or internships. At the moment, the author has over 20 binders with code and documentation on his book shelves. The digital images of the code are on CD, but not in a formal repository. With RSA, Clearcase, and the other SSI tools, the code may be managed like a software development enterprise intending to have a long-term relationship with its clients.

One final note on the capstone courses is important. After the system is in semi-final form, students are asked to prepare test scenarios using the major use cases in order to conduct usability evaluations with class members. Using a large computer lab, students evaluate each other’s work and provide written feedback to the developers. Some of the students expressed that this had been a valuable experience for them because they could see how what seemed obvious to a developer may not be that obvious to a user.
Examples and details of student-led projects which have satisfied community organizations are presented by Fernandez (2004).

Conclusions

Capstone courses that focus on performing real-world community projects for users with real problems can greatly help the student to see his or her role in satisfying the needs of users. Using industrial strength software tools for capstone courses bring all of the education together to support the activity which is the cornerstone of a service oriented field like computer science or information technology. Community leaders have built relationships with local and international students. The benefits continue today as students have obtained employment or moved on to complete their degree and the applications they built are serving the community. Student surveys indicate that students received very positive results from the experience of working for real world clients. This work was partially funded by NSF Minority Institutions Infrastructure Program grant #EIA-0330822.

References

15. Shared Software Infrastructure Hub, Texas A&M University, web site at http://ssi7.cs.tamu.edu/ssi/