7-1-2013

Experience Report: A Sustainable Serious Educational Game Capstone Project

C. Shaun Longstreet
Marquette University, christopher.longstreet@marquette.edu

Kendra Cooper
University of Texas at Dallas

Accepted version. Published as part of the proceedings of the conference, *18th International Conference on Computer Games: AI, Animation, Mobile, Interactive Multimedia, Educational & Serious Games*, 2013: 217-221. DOI.

Experience Report: A Sustainable Serious Educational Game Capstone Project

C. Shaun Longstreet
Center for Teaching and Learning, Marquette University
Milwaukee, WI
Kendra Cooper
Department of Computer Science, The University of Texas at Dallas
Richardson, TX

Abstract: Capstone courses play a key role in many Computer Science/Software Engineering curricula. They offer a summative opportunity for SE students to apply their skills and knowledge in a single experience and prepare them for work in industry. Capstones have many attributes that make them a valuable high-impact practice, yet there are several challenges that can be associated with them. These challenges include the general nature of a capstone that prevents deeper applications of skills, not to mention the difficulty of creating an interesting and engaging design project upon which students can make meaningful contributions and engage in extensive team dynamics. This experience report outlines an innovative approach to a senior design capstone course that addresses common limitations of capstone courses. The SimSYS capstone course is unique in that it involved a mixed team organization involving a more senior design team who led a development team over the course of the semester, thereby leveraging the diverse experience of capstone students completing their CS/SE degree. The results point to solutions for continuing a capstone project successfully in subsequent semesters that could be of interest to other SE curriculum designers looking to develop effective capstone courses.

2013 18th International Conference on Computer Games: AI, Animation, Mobile, Interactive Multimedia, Educational & Serious Games (CGAMES), (July/August 2013): pg. 217-221. DOI.

SECTION I.

Introduction

Capstone course experiences usually promote some applied integration of knowledge and skills of a single discipline. Conversations about capstone courses became popular in higher education during the 1990G. In many disciplines, and in science, technology, engineering and math (STEM) curricula especially, the senior capstone course has become commonplace. Taken as a culmination of a major, there are many purposes for offering a capstone experience. One reason is that it can provide a chance for students to demonstrate their understanding and appreciation of their major. Another reason is to provide a Oull-circle experienceOwherein students may draw from their junior level or general education experiences and apply them to real-world problems. Yet another reason is that a capstone may act as a professionalization opportunity, preparing students for work beyond their academic environment.[1] Within the software engineering (SE) curricula, traditional capstone projects are often one term, cover requirements, design (architecture, component design), implementation, testing, user documentation, and various umbrella activities (see related works below). Students will typically work as individuals, in small groups, or as a whole class on singular projects with the faculty assessing the deliverables at the end of the semester.

Given the benefits of capstone courses as noted above, there are still significant issues that faculty and curriculum designers face to effectively introduce and sustain the capstone experience. These hurdles can discourage some and prevent successful program developments that could benefit majors. The first stumbling block is the limited evidence for significant application and integration of knowledge. Capstones, by their nature, assess students in generalities rather than on individual strengths, aptitudes, and ambition. Since capstones typically offer a culminating application experience, new knowledge is rarely introduced, which ill-serves underprepared students and creating uneven levels of student participation. Moreover, capstones are often centered on a senior design project; this in itself can be challenging in that developing a significant learning opportunity that can serve as a culmination of studentsOabilities and knowledge. Another difficulty comes in creating a semester-long project that is simply not a rehearsal of practices that merely add up to a collected series of activities, the end of which is already known and success is measured by whether or not the deliverable functions. At the same time, it is not easy to create a project that encourages consistent team interaction over the progression of the semester. Without

2013 18th International Conference on Computer Games: AI, Animation, Mobile, Interactive Multimedia, Educational & Serious Games (CGAMES), (July/August 2013): pg. 217-221. DOI.

a significant amount of faculty mentoring, students can meet sparingly and parse tasks out to individuals.

Thus, students may complete a capstone course without gaining highly desirable skills in team cooperation, effective communication and interpersonal skills.

This paper is an experience report, briefly describing how we first developed a course around a senior design project with a mixed team, which we then channeled what we learned into an ongoing capstone course that both advances research into serious educational games and provides essential industry valued skills. The intent of the mixed team was to address common capstone limitations while capitalizing on students interest in contributing towards a research project on developing a serious educational game. The structure of the remainder of this paper is as follows: Section 2 describes our development of a split team senior capstone course; Section 3 summarizes the outcomes of our experience and lessons learned; Section 4 is a review of related work, indicating the range of work around capstones and mixed team courses; and Section 5 ends with summary and future work.

SECTION II.

A New Capstone Course

As noted in the limitations above, we strove to develop a senior design capstone course that would better leverage a broader range of studentsabilities interests, and to prepare them more effectively for professional expectations in industry. In part, this required flexibility because the prerequisite courses emphasized different components of the course content widely depending on who was teaching the course.

A. Prerequisites

The senior design capstone required several prerequisites which are outlined below to provide some scope of incoming studentsabilities.

In SE *3354/5354* students must demonstrate an applied understanding of the following: *software lifecycle development models, requirements engineering techniques, design principles, testing techniques; metrics in software engineering; formal methods in software development; establish, participate in an ethical software development team; software project management tools and techniques; CASE tools for software development*.

2013 18th International Conference on Computer Games: AI, Animation, Mobile, Interactive Multimedia, Educational & Serious Games (CGAMES), (July/August 2013): pg. 217-221. DOI.

In *SE 6361 Requirements Engineering*, successful students would be able to demonstrate critical awareness and application of: *digital filters using S/W tools such as LabVIEW or Matlab; sampling rate conversion methods using polyphase structures for decimation and interpolation filters; linear prediction, optimum linear filtering, and adaptive filtering techniques; parametric and non-parametric spectral estimation methods*.

In *SE 6362 Advanced Software Architecture and Design*, the learning objectives included: concepts and methodologies for the development, evolution, and reuse of software architecture and design, with an emphasis on object-orientation; analysis, synthesis of system data, process, communication, and control components; decomposition, assignment, and composition of functionality to design elements and connectors; using non-functional requirements for analyzing trade-offs and selecting among design alternatives; transition from requirements to software architecture, design, and implementation.

Finally, one other pre-/co-requisite course is, *SE 6367 Software Testing and Verification*, in which class participants have to show the can: generate tests from requirements; apply test adequacy measurements and enhancements; perform functional testing, control flow-based testing, dataflow-based testing, regression testing, mutation-based testing, model-based test generation; perform effective program debugging; perform static and dynamic source code analysis; identify fault-prone software components and security vulnerabilities; understand and apply software testing tools.

The split team capstone course drew upon the traditional learning objectives for SE 6387. These included demonstration of their ability to comprehend and critically apply: requirements engineering techniques; design (architecture, component) techniques to realize the requirements; programming techniques to realize the design; SQA techniques to ensure the quality of the artifacts (models, code); project management techniques; configuration management techniques.

In addition, we added another objective; that students would need to demonstrate an ability to apply professional teamwork skills and leadership techniques. The intent of this was to address what we felt were some of the weaknesses in traditional capstone courses. This last objective became all the more important we found, as the project continued into subsequent semesters.

B. The Project

2013 18th International Conference on Computer Games: AI, Animation, Mobile, Interactive Multimedia, Educational & Serious Games (CGAMES), (July/August 2013): pg. 217-221. DOI.

The senior design project comes out a larger research project: the development of educational games as next generation of e-learning tools. Because games have a reputation for being fun and engaging, while also being immersive, requiring deep thinking and encourage more complex problem solving, this capstone course project was the beginning of a SE educational Game Development Platform (GDP), called SimSys. The long term of SimSys vision is to provide an approach that offers: intelligent, semi-automated game script generation; repositories of re-usable game assets (graphics, audio, learning objectives, challenge content); automated dynamic player assessment and automated game adaptation; and an agent-oriented game play engine that executes scripts with some interesting, non-deterministic behavior to keep the game fresh for the player.

At the time, SimSys was intended to be a simulation game that would have players take on the role of a junior software engineer. In the game world, the player was challenged to demonstrate their understanding and ability to apply SE concepts in a variety of scenarios. We began the project by first identifying students who would take the lead in design and those who would develop the game engine and the user interface. This then allowed the project to move forward in the coming semester.

C. The Team Design

Initially, the design team consisted of one M.Sc. student and one B.Sc. student, with an average industry experience of 3.5 years. They both had very strong GPAs and were (fast trackt) students who were pre-admitted into the SE graduate program. They both expressed an interest in software architecture and agent-oriented software design. The development team consisted of five M.Sc students working on completing a capstone experience in SE, had an average industry experience of one year.

The faculty consisted of the faculty of record, a tenured faculty member with several years teaching experience in the capstone course. There was also an external advisor from the university faculty and graduate student professional development office. And, on occasion, an international guest faculty member contributed as opportunity and time allowed. The average length of experience was approximately ten years. The role of the faculty and advisor was to offer feedback and guidance to the teams on the overall project, the software design and development, the game aesthetics, its assessment capabilities, and ability to engage learning. The instructors were not allowed to create or modify the design, they would only ask questions, elicit the rationale, and comment on pedagogical design details.

2013 18th International Conference on Computer Games: AI, Animation, Mobile, Interactive Multimedia, Educational & Serious Games (CGAMES), (July/August 2013): pg. 217-221. DOI.

2013 18th International Conference on Computer Games: AI, Animation, Mobile, Interactive Multimedia, Educational & Serious Games (CGAMES), (July/August 2013): pg. 217-221. DOI.

D. Development Process

In order to facilitate the course, the instructor and educational development staff member met weekly with the students. In-class time primarily consisted of a two-part meeting every week. The instructors would first meet with the design team. The primary responsibilities of the design team consisted of listening to the desired parameters of the game, develop the software architecture, present this information to the development team, and answer their questions. Prior to the beginning of the semester, a series of orientation and design meetings were held to give the design team more time to prepare for the course. When the semester began, at the weekly class meetings, the design team presented their strategies for implementation and discussed how they would engage the development team for the coming week. They were required to review their progress with the project and summarize their communications with the development team. They explained any alterations to the software architecture/design, answered instructors questions, and prepared to brief the development team who would be joining the team meeting.

After the design team consulted with the instructors, the development team joined the meeting. The responsibilities of the development team included attending to the design specifications outlined during the meeting and to develop relevant components of either the Game Engine or the UI.

They were briefed by the design team and then asked if they had specific questions about the architecture or the design process. The development team then provided a demonstration of their work from the previous week. This was done sometimes as individuals, at others as an entire group, with each team member taking a turn explaining their specific contributions to that week development.

E. Transition to Next Capstone

While there was a developed product at the end of the semester, it contained several flaws within the code that did not meet specification requirements. Reflecting upon the course strengths and shortcomings, we were able to make significant improvements to the capstone course that has led to a fruitful rebooting of the project in four subsequent semesters since then. We improved the work output of the student teams while providing an even better learning experience for subsequent capstone teams by taking several measures that we believe can be transferable to capstone courses at other institutions.

2013 18th International Conference on Computer Games: AI, Animation, Mobile, Interactive Multimedia, Educational & Serious Games (CGAMES), (July/August 2013): pg. 217-221. DOI.

The first step was to assess the project as a whole and identify smaller components that could serve specific tasks in a manageable amount of time over the course of a semester. The second step was to redesign the initial first few class sessions to more quickly identify various studentsknowledge and skills shortcomings so that we could provide them with specific, concrete, assignments given to insure they would follow through with remedial tasks.

In following semesters, we were able to build upon this structure so that students were provided calibration exercises that fed into the overall project. This allows the project to maintain its forward momentum while insuring that all students are quickly on-board as a team. To address issues about teamwork and communication, a significant portion of early class meeting time was devoted to encouraging communication, and acknowledging productive interfacing.

This occurred by tracking communications with copying instructors on emails, regular mid-week check-ins via Skype, as well as by tracking logs of development teamsOwork in the repository. Moreover, highlighting the function of specific roles with development subgroups also prompted students to value frequent communications with clear expectations and timeframes. To facilitate a better sense of responsibility and accountability, there was increased time devoted to professional development over the entire course of the semester, as well as to providing feedback on how individual contributions were impacting the growth of the larger project.

Providing clear, high professional expectations with an increased context and positive feedback helped prevent subsequent project slippage and countered occasional complacency that can arise in student-led projects.

SECTION III.

Results

By the end of the first semester, there was something of a game prototype. To some extent, the game did allow for a player to start and enter the simulation, choose a character and play a couple of scenarios towards completion. Upon assessment, however, the prototype had a quite limited function. The designs and the code were not easy to maintain, evolve, or pass on to the subsequent development teams. Much of the project knowledge was too complicated; there was too little documentation, the software did not rely on existing components/libraries; standard libraries used were not well understood, and the

2013 18th International Conference on Computer Games: AI, Animation, Mobile, Interactive Multimedia, Educational & Serious Games (CGAMES), (July/August 2013): pg. 217-221. DOI.

design depended on advanced programming language features (e.g. reflection). In the end, it was decided that the initial prototype was to be discarded. Given the complexity of the initial development teams product, as developed, we realized that it would be too difficult for subsequent teams to build around the initial project.

There were several lessons learned in the initial iteration of the course for the faculty involved. The first was that the development and implementation of a mixed team course in a senior design capstone was very time intensive. The mentoring of the design team and subsequent facilitation of the meetings tended to take several hours a week. Working with offsite collaborators, while workable, did create difficulties for the faculty of record and for the students who were not used to receiving immediate feedback in class time.

As this was a senior capstone course with several prerequisites, the instructors had made assumptions about the relative preparedness of the students. There was still some expectation that individuals within the development team would need to calibrate their skills. Subsequently, we set aside time at the beginning of the course for students to prepare for the project. At the opening of the semester, we held an initial briefing of the project requirements with both the design and development teams in attendance; the development team was provided two weeks to do any specific remedial work that would prepare them for the initial all-hands meeting. Even given a list of requirements, plus time at the beginning of the semester to work on remedial skills, we found that the development team had difficulties assessing their own abilities.

By the end of the first semester, it was clear that the approach was too agile for the skill level of the students. Both the design team and the development team would have benefitted from more structure, more comprehensive preliminary write-ups to ensure common understanding of the solution and goals of the project. It was clear that in order to insure a common starting point, better use of existing libraries, components and development tools would have greatly assisted everyone. Another issue that became apparent at the end of the course was the scope and complexity of the project was beyond the means of students learning to work effectively in teams and learning how to use new development tools and repositories. Furthermore, given the breadth of the project, we believe that prioritizing immediate applicable abilities as well as less frequently emphasized soft-skills meant that it was less feasible to address architecture/design skills in this project as currently structured. Students within the development team did not realize until it was too late that their progress depended on cooperation and steady contributions to each otherG work. At the end of the semester several bottlenecks occurred

2013 18th International Conference on Computer Games: AI, Animation, Mobile, Interactive Multimedia, Educational & Serious Games (CGAMES), (July/August 2013): pg. 217-221. DOI.

and students began to code in parallel making integration of their work extremely difficult and poorly executed. This led to an unsatisfactory prototype that could not be used in subsequent semesters.

Students within the design team, even though more advanced than their peers, also required professional development skills. In addition to having the need to provide more succinct, clear presentations to the development team, the design team needed to develop a greater awareness of the need to encourage frequent communication channels between face to face meetings. Moreover, the design team, while fairly aware of their responsibilities as designers, did not fully appreciate the context of the course within a larger research project and within a continuing series of senior capstone courses in semesters to come.

SECTION IV.

Related Works

A rich collection of reports on capstone project-oriented courses is available, representing a variety of experiences. These include, for example, reports on projects using different organizations (intra-class, inter-class), industrial partners as customers (Christensen 2003, Vanhanen 2012), alternative development methodologies ranging from agile to plan-driven (Smith 2011, Roggio 2006), projects organized a student competitions (Wong 2012); and those that emphasize project management skills (Parrish 1995).

With respect to organization, the majority of reports consider an intra-class organization, in which one class of students are organized into sub-teams to systematically engineer a project from a preliminary project description through to demonstrating a prototype to their customer (e.g., Bullard 1988, Dascalu 2005, Demurjian 2009, Malinowski 2009). They consistently include established software engineering activities: requirements engineering, architecture, design, implementation, testing, and demonstrations in addition to project and configuration management, with corresponding milestones. Within this collection many of the details vary, such as the duration of the course (e.g., one quarter or term, two terms), who plays the role of the customer (instructor, external representative), source of the project (team, instructor, external representative), team organizational structure (flat, chief programmer, mixed control), responsibilities of the sub-teams (each develops an entire project or a subsystem for integration), and the provision of more senior students to technical or management guidance.

2013 18th International Conference on Computer Games: AI, Animation, Mobile, Interactive Multimedia, Educational & Serious Games (CGAMES), (July/August 2013): pg. 217-221. DOI.

Relatively few reports are available on capstone project-oriented courses that are organized across different classes (inter-class). For example, (Daigle 1997) has organized projects that involve students from the sophomore level software engineer course with students in the senior capstone project course. The sophomore students work closely with the senior project teams to produce realistic schedules and estimates for cost/effort for the senior teams to use (instead of toy examples); sophomores review the design models created by the senior team against those produced by the previous yearG teams (providing them with an external, objective review of the design); and sophomores use the code developed by the seniors for reverse engineering exercises (instead of randomly selected code). In other work, (Fenwick 2005) reports on three variations in organizing a project across classes: projects that span the capstone project course and a lower level data structure course; an upper level specialized elective course (HCI); and both the data structure and a junior level (database) course. After trying all three variations, the authors present four lessons learned. The first is to mandate collaboration with lower level students. If given the option to collaborate or not, the senior students may not voluntarily involve the less experienced students. The second and third relate to component deliver: prepare for the failure of teams to deliver a component and manage the component develop progress carefully. Lastly, the fourth lesson is to beware of final project dependencies – students may not be able to complete their projects.

SECTION V.

Conclusions and Future Work

A significant consequence of our work was the realization of a capstone course that leveraged limited resources to offer a deep learning opportunity for a group of students with broad technical and experiential backgrounds. This learning opportunity not only allowed for significantly improved CS/SE technical skills, it created several semester-long opportunities for students to work on a concrete, time-on-task heavy projects that drew upon a variety of abilities and knowledge. It has provided students with prospects several new skills upon which they were able to scaffold what they had learned in previous courses.

While there are several benefits to the mixed team capstone course, we identified a few significant limitations over the course of the capstone. The first, and most significant, was the broad range of disparate skill and knowledge levels within the development team. The second limitation was the issue of team interaction and level of studentsOabilities to develop channels of communication. Students in the program were not used to working in

2013 18th International Conference on Computer Games: AI, Animation, Mobile, Interactive Multimedia, Educational & Serious Games (CGAMES), (July/August 2013): pg. 217-221. DOI.

a distributed team environment where they depended on peer contributions to the project. A third issue that we faced was the level of professionalization the students had had prior to coming to the course.

While the discovery of our studentsO knowledge shortcomings may sound like a problem, it did outline where the department could improve the SE curriculum. This exercise in developing a split team capstone course confirmed particular identified curriculum needs being introduced in the revised edition of SWEBOK. At the same time, it indicated the need for greater attention to catching lapses in fundamental skills, either earlier in the curriculum or within the early stages of the capstone course.

The success of the course remains in that it gave the design team invaluable experience with leading a diverse group of people and offered the development team the chance to work in a highly variable environment where project needs necessitated close and frequent interaction and increased time on task. For the instructors, the process did advance the larger project in unexpected ways. When it was clear that there was a greater need for increased definition of the development process, a line of research opened when we outlined a new meta-model for serious educational games for developing and measuring pedagogically sound training and education as well as other areas in serious educational game design.

The SimSYS senior design capstone course has continued successfully three times a year since first offered in 2010. On average, there are 4–6 graduate students working on the project each semester. The project has led to one MasterG thesis on intelligent adaptation and four student co-authored pieces in peer-reviewed proceedings. Not only has the capstone project furthered serious game research, it has provided many students with valuable experience that industry feedback suggests graduates need. Even further, the course indicates how other CS/SE curricula can successfully develop sustainable projects for senior design capstone courses around the subject of game software development.

References

[1]Jayne E. Brownell and Lynn E. Swaner (2010), *Five High-Impact Practices: Research on Learning Outcomes, Completion and Quality*. Washington DC: Association of American Colleges &Universities.

[2]Catherine Bullard, Inez Caldwell, James Harrell, Cis Hinkle, and A. Jefferson Offutt, 6natomy of a software engineering projectO *SiGCSE Bull*. 20, I (February 1988),129-133.

2013 18th International Conference on Computer Games: AI, Animation, Mobile, Interactive Multimedia, Educational & Serious Games (CGAMES), (July/August 2013): pg. 217-221. DOI.

[3]Ken Christensen, Dewey Rundus, and Zornitza Genova Prodanoff, (Partnering with Industry for a Computer Science and Engineering Capstone Senior Design CourseO In *Proceedings of the ASEE Southeast Section Annual Conference, 2003.*

[4]Roy J. Daigle and Marino J. Niccolai, Gnter-class synergy by designO In *Proceedings of the twenty-eighth SIGCSE technical symposium on Computer science education, 1997*, pp. 92-95

[5]Sergiu Dascalu, Y. Varol, Fred Harris, and B. Westphal, (computer science capstone course senior projects: from project idea to prototype implementationQ in *Proceedings of the 35lh Annual Conference on Frontiers in Education, 2005.*

[6]Steven A. Demurjian, *Experiences in Project-Based Software Engineering: What Works, What Doesn't, Software Engineering: Effective Teaching and Learning Approaches and Practices,* IGI Global, 2009 (191-211).

[7]James B. Fenwick, Jr. and Barry L. Kurtz. Intra-curriculum software engineering education. In *Proceedings of the 36th SiGCSE technical symposium on Computer science education, 2005*, pp. 540-544.

[8]Christopher Malinowski and Richard E. Noble, (capstone Projects: Putting the Pieces TogetherO information *Systems Education Journal*. 7 (24), pp. 1-9.

[9]Allen Parrish, David Brown, and David Cordes, (Experience in teachin? a management-oriented capstone software engineering courseQ Software Engineering Education, *Lecture Notes in Computer Science* Volume 895,1995, pp 277-293.

[10]Robert F. Roggio, 6 model for the software engineering capstone sequence using the Rational Unified ProcessO In *Proceedings of the 44th annual Southeast regional conference, 2006*, pp. 306-311.

[11]Tucker Smith, Kendra M.L. Cooper, and C. Shaun Longstreet. 2011. Software engineering senior design course: experiences with agile game development in a capstone project. In *Proceedings of the 1st international Workshop on Games and Software Engineering (GAS 'II),* pp. 9-12.

[12]Jari Vanhanen, Timo Lehtinen, and Casper Lassenius, "Teaching realworld software engineering through a capstone project course with industrial customers," in *Proceedings of the First international Workshop on Software Engineering Education based on Real-World Experiences, 2012*, vol., no., pp. 29-32.

[13]Wilson Wong, James Pepe, James Stahl, and Irv Englander, 6 Collaborative Capstone to Develop a Mobile Hospital Clinic Application Through a Student Team CompetitionO In *Proceedings of the Information Systems Educators Conference,* v29 nI972, 2012, pp. 1-12.

[14]Hadar Ziv and Sameer Pati, (capstone Project: From Software Engineering to InforrnaticsO In *Proceedings 2010 23rd IEEE Conference on Software Engineering Education and Training*, pp. 140 E〉 146.

2013 18th International Conference on Computer Games: AI, Animation, Mobile, Interactive Multimedia, Educational & Serious Games (CGAMES), (July/August 2013): pg. 217-221. DOI.