Transforming Experience of Computer Science Software Development Through Developing a Usable Multiplayer Online Game in One Semester

Ilmi Yoon[1] and Eun-young Elaine Kang[2]
[1]*Computer Science Dept, San Francisco State Univ. 1600 Holloway Ave, SF, U.S.A.*
[2]*Department of Computer Science, California State University, Los Angeles, U.S.A.*

Keywords: Multiplayer Online Game, Game Development, Industry Style Team Work, Project-based Learning.

Abstract: We present an instructional design of computer science project-based course to transform students' experiences of acquiring software development skills. In a collaborative classroom emulating a typical industry work setting, students will collectively create and build a Multiplayer Online Game using a variety of complex software components. A course was taught to design and develop a working Multiplayer Online Game within one semester: building a ready-to-usable game in one semester with whole classmates presents significant challenges to cope with and stimulate students to realize the important aspects of teamwork and software engineering principles. Students present their progress, discuss future milestones and trouble shoots, update documents for clearer communication and utilize source control tool throughout the semester. Unlike usual class setting, all students worked collaboratively together like one company to achieve the goal. In the class, students started from concept design and developed specific components of working Multiplayer Online Game, while broadly learning game design, 3D graphics, Game Engine, Server-client architecture, Game Protocol, network programming, database, Software Engineering principles, and large application development as a team project. The course was successfully transferred to CSULA in Fall Quarter, 2013.

1 INTRODUCTION

Over the past decade, the software development process has become exceedingly complex (Fuggetta, 2000). The industry now expects software engineers and computer scientists to excel not only in the technical aspects of software development, but also in self-learning of advances in new technology, the fluent communication of their ideas in presentations and documentation, often produced in a collaborative environment (Dutoit and Bruegge, 1998). Because software development today cannot be accomplished without collaboration across teams, experience in this area is highly desirable in computer science students entering the job market (Mencher, 2003; Parberry et al., 2005). Given the rapidly changing nature of technology, software developers can no longer rely exclusively on concrete knowledge previously attained. Rather, they must have the ability and adaptability to adjust swiftly to changing practices and emerging platforms. These abilities are seldom addressed in an academic setting, leaving new graduates minimally prepared and struggling in these important areas. There is an enormous need for courses that provide students with the training and experience to meet the changing expectations of industry and to successfully transform knowledge they have acquired in the classroom into the ability to produce complex, innovative, high-quality software products in a collaborative environment.

The pedagogy currently used in most Computer Science classes with team projects employs team projects of a few weeks to a whole semester in duration with several team members, involving minimal or no collaboration across teams. While such an approach is essential and effective for teaching specific content knowledge and achieving particular learning outcomes, it is also limited in that it focuses on the development of relatively simple programs to teach specific content knowledge and on the completion of classroom-specific assignments

Yoon I. and Kang E..
Transforming Experience of Computer Science Software Development Through Developing a Usable Multiplayer Online Game in One Semester.
DOI: 10.5220/0004963703290335
In *Proceedings of the 6th International Conference on Computer Supported Education* (CSEDU-2014), pages 329-335
ISBN: 978-989-758-021-5

that are not readily transferrable to an industry setting and do not adequately prepare students to work on large-scale programming projects in a highly synchronized work environment that demands complex contributions across teams. Students are rarely exposed to inter-team interactions or to working collaboratively across teams to develop a common product. Ultimately, the scope of the software developed in the typical CS classroom and the integration of its components tend to be simplistic as a result of the aforementioned limitations (Parberry et al., 2005). This paper presents an alternative approach to prepare students for the challenges they will face in industry: a course designed around a complex project to be completed by the class as a whole, giving students practical experience with inter-team collaboration and an understanding of the complexities of the software development process.

Objectives of the course are to: (1) teach students effective communication, presentation and collaboration skills that are often neglected in the traditional classroom; (2) motivate students to learn essential CS core content through peer interaction and encouragement; (3) enable students to produce a robust end product; and (4) give students practical insights into the real-world work environment to prepare them to master the challenges of a career in CS.

In next section, we discuss backgrounds on selecting Multiplayer game as instructional material. In section 3, we discuss the course design and structure to teach students to build a working MMORPG game within a semester (15 weeks). In section 4, we present the results and learning outcome. Conclusion and on-going work are discussed at section 5.

2 BACKGROUND - INSTRUCTIONAL MATERIAL

Massively Multiplayer Online Role Playing Game (MMORPG) is a type of online multiplayer game that allows thousands of players to play concurrently in a virtual world. Even after players log out or disconnect, the virtual world persists and the updates made by player remain. Such games form an online community of players who interact with one another and progress through the game over time. The quality of social interactions and character progression within the game radically increase the excitement of the game; popular games like WoW (World of Warcraft) and Second Life easily develop communities of multimillions of players, to the point that they have become significant trends in global youth culture (Song, S., Lee, J., 2007). It is the author's experience that being able to understand the behind-the-scenes work that goes into the creation of such games and then developing a game themselves strongly motivates students to focus on and excel in their CS coursework.

Social gaming and MMORPG are relatively recent game genres as they bloom over the availability of Internet. Unlike stand-alone game, a MMORPG game keeps virtual world persistent even after players log out, so game server has to run infinitely connecting all the clients and updating game status at database. Therefore, building a MMORPG game covers broad spectrum of computer science technology from computer graphics, 3D modeling, game engine, network programming, client-server architecture, and database, so it helps students exposed to diverse technical components comprehensively. Also a Multiplayer Online game is indeed a large application with a lot more challenges in debugging and quality assurance. Succeeding in building a reliable one ensures students to learn the qualities that software companies want.

3 IMPLEMENTATION

This class was designed for Computer Science senior and graduate students who completed programming languages, data structures and Software Development Principles but not necessarily have taken elective courses such Software Engineering, Network nor Database courses. Considering that majority of students find jobs at industry after graduation, the class was organized in more of industrial flavor; a team of students (1) receive tasks and milestones, (2) achieve the milestones by actively looking for solutions from any available resources, (3) interact with other teams collaboratively, (4) and produce documents for clear communications between teams and future extensions.

The class was organized into Game Concept Design team, Game Server team, Game Client team, Game protocol team, Database team, Art Support team, Game Content team, Testing team, and Launching team. Within first 2 weeks, objectives and the responsibility of each team were discussed and then students were assigned into one or two teams based on their interest and backgrounds. From

then on, teams worked on in parallel to achieve their milestones, while learning all the required technology for the given task and sharing their learning with other teams.

Sample Summary of team tasks is as below.

Game Concept Design Team: this team has most important initial responsibility. The objective of the whole class was to build an educational MMORPG for Computer Science students, especially for beginners who are taking first programming courses. The concept design team should consider the limitations such as time (10 weeks for actual implementation), lack of artists, and unknown factors like the potential of other team's performance. The team also has to collect everyone's idea, finalize the game concept into several milestones and produce the documentation.

Game Client Team: this team is responsible to implement 3D game client using Panda3D. Students have to learn 3D graphics, Panda3D game engine and Python. Also client communicates with server using the game protocol to update the status (position, motions, and levels, etc) of other clients concurrently in the same virtual space.

Game Server/Protocol Team: this team is responsible to implement game server that connects to all the clients, synchronize every client status and update the Database. Team did not start from scratch as there was a JAVA game server written for a similar MMORPG game called "NurseTown" by students of Internet Application course. The server team had to read and revise the game server to work for "deBugger" game. Also the server team has to develop an effective set of protocols used between clients and server. Students in this team have to learn socket programming, client-server architecture, and Network protocols.

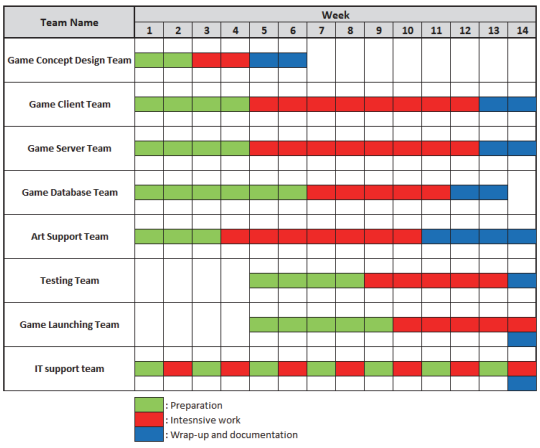

Figure 1: Timelines for each team over weeks during the semester.

Art Support Team: this team is responsible to provide 3D models to create the virtual world and characters in the game. As there are no artists in the class, this team has to find freely available 3D models and convert the format to work with Panda3D. Students in this team have to learn basic of 3D modeling tools such as Blender or Maya and converting 3D model formats, and real time rendering performance trade-off (visual quality vs. rendering speed).

Game Content Team: "debugger" game is a unique MMORPG game for educational purpose. Therefore, the educational game contents are as important as other game assets. This team develops educational contents. As senior or graduate students of computer science, this team is already familiar with educational contents that beginner programmers learn which is good and bad. This team needs to interview beginner students, observe the troubles that they probably forgot and produce effective educational contents.

There were a little over 20 students in the class, so each student choose one primary team and one secondary team, so each team has at least 2 to 3 students. By having students in a few different teams, students could learn more than one technical area and teams can communicate and update the progress easily. Each team has to work very collaboratively as the whole class works for one project together.

Important concepts and technologies were taught to the whole class by several lectures, but most of class meetings were used as presentation of milestones, progress, trouble shootings of problems on hand, so the whole class stays in a same page and gets exposed to the problem and solutions.

Software Engineering principles have been practiced throughout the whole course as teams realized the importance of the inter-team communications due to the rapid developments in parallel. Changes in one team did propagate changes in milestones or requirements in other teams. SVN was actively used.

4 RESULTS

The whole class successfully developed a working MMORPG game at both 2009 Fall (DeBugger - http://smurf.sfsu.edu/~debugger/), 2011 Fall (World of Balance - http://smurf.sfsu.edu/~wob/) when the courses were offered in this style, 2013 Spring (extension of World of Balance game), and 2013 Fall (extension of DeBugger game, California State Univ. Los Angeles). The games produced by the

class had intention to be used for educational game research or science discovery game research which added substantial challenges (entertaining while achieving another purpose) compared to pure entertainment games. As a whole class, students have constantly struggled together to validate the game design for achieving its intended purposes. Developed games are currently being used as intended (Yoon et al, 2011; Yoon et al 2013)

DEBUGGER GAME (Educational MMORPG For Computer Science Students)

Each team achieved milestones close to the initial (very ambitious) goal. **Game concept team** came up with the title, "deBugger" and the theme of the game; fighting against bugs within inside of computer, inspired by the origin of the word and the significance of the word to computer science students. Team also prioritized concepts in to several milestones, so kept the milestones doable for one semester, but left many interesting ideas for future extensions.

Primary milestone for Fall 2009 was to build virtual world where players can explore and fight with bugs by solving multiple choice questions. Health drops under bug's attack and game items can be used to shield from the bug's attack (delay the health damage or clear out options from the multiple selections). Players can level up by solving questions required by each level. Also player should be able to chat, create friends' list and maintain their inventory (trading or giving gifts).

Art Support Team achieved beyond the requirement. Students learned 3D modeling tools and created 3D environments for the game. We used 3D character models from Panda3D repository, but all the environments (figure 2) were created by art support team and theme was inspired by movie Matrix and Tron during inspirational discussions at class.

Client Team was the largest team and they were divided into sub teams to be responsible for specific tasks in parallel. List of such tasks are Registration Process (Avatar Selection), Login Process, Chat, Bubbles (translucent panel), Chat Bubbles, Character Name and Damage Bubbles, Friends, Inventory, Character Info, Hot keys, Camera control, Character Movements, Interface (mouse, Keyboard), Collision, Battle System, NPCs (Non Player Character), Bug (with AI), Mini maps Client team also achieved impressive progress of completing all the requirements and the resulting game client was reliable and capable of very smooth rendering. Each task listed above took intensive effort to understand the technology (3D graphics) to manipulate Panda3D game engine.

Server team worked hard to make MMORPG game alive. Server team was able to reuse the majority of server code built for a game called, "Nursetown" by Prof. Ilmi Yoon and her graduate students. Server team had to understand the communication mechanism and modified DB schema and added lots of new game protocols.

Game Contents team focused on adding educational contents into the game. Game contents team structured pedagogy of computer programming learning and created quizzes in different levels for such purpose.

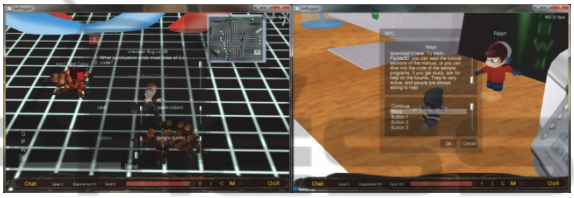

Figure 2: Left: A Scene from the game: NPC (Non Player Character) requires simple AI to interact with players; guide players to follow rules of game or inform about events to participate. Right: A scene from the game: Bugs are attacking players. Bugs have AI to chase the player, handle collision and re-spawn.

Data Base team studied design of efficient DB schema and developed it for debugger game. Team also installed mySQL server and deploy the data sets.

Testing team was developing a test client that has a minimal AI to wander around, acts like a usual client and save the communication log, so we can use the test client to measure the scalability of the server. In the middle of the semester, we found that the debugger game needs a bug server that controls all the bugs that has AI to attack the player and wander around the debugger world. Testing team took the role and successfully completed the creation of the bug server with the close collaboration with the client team.

Launching team studied popular MMORPG hosting company's web site and designed a web site for the possible full version of the debugger web site and implemented core set of it to be able to launch the game for programming class students (figure 13). The launching team made a simple web site (http://smurf.sfsu.edu/~debugger) with link to download client application (self-extracting installer for windows), game rules, information of how to get started, and screen captures.

The class as one whole team completed a working game (version 0.9) by the end of semester. The game contained core MMORPG game features.

The client and server connect and communicate in stable manner. DB stores persistent data. User can create an account with options of a few characters. And then user can log in at the location where they logged out last time. Player can explore the virtual world starting from a desktop space into the inside computer world. User can find if friends are logged in and chat with friends or other players within the same space. Players can check their game items at the inventory box, health bar and the current level as well as their current position at the mini-map. Most importantly, players can battle with bugs using the multiple choice questions and the health bar and level were properly updated.

At the same time, whole class produced one giant document (437 pages, available at http://smurf.sfsu.edu/~debugger/download_documentation.php) of each team's work for future developers who will extend this game to the next level.

At the Fall 2013, this course was adapted at CSULA for the first time and the students extended the DeBugger game by adding multiple mini games (http://csproject.calstatela.edu:8080/debugger/#/). Due to the transition from the semester system to quarter system, there were additional challenges. Even so, the students' evaluation shows the effective learning in large scale projects and communication skills.

WORLD OF BALANC (Science Discovery Game for Computational Ecology Research)

This game focuses and extends the scope of game from simple entertainment or education further into Gamification or crowd computing that enables scientific data collection through game play. While players are playing an ecosystem nurturing game, they are also producing data that can be meaningfully used by ecologists (Yoon 2013). Details of game can be found at the given web site (http://smurf.sfsu.edu/~wob/). This game project presented additional challenge of understanding computational ecologists' research agenda to be able to develop effective science discovery game. The content team also worked extra hard to connect the game with computational engine that scientists use for their research. Over two semesters, game engine (client side) has been changed from Panda3D to Unity3D to support multiple platform (iPad, Windows, etc). The game has been used for learning/training impact study and play test with 20 players for a week.

Figure 3: A Scene from World of Balance Game (Unity3D Client), showing the biomass of the species within the game.

5 CONCLUSION AND ON-GOING WORK

Students were proud of what they achieved in one semester (15 weeks). Considering that the first few weeks were used to the introduction to the class and setting up the team, students had about 2.5 months to work on the game. To be able to make the MMORPG game work, each team has to deliver what they were expected in milestones. It serves as a good pressure on each team and students collaborated across team enthusiastically. Most students put efforts beyond requirements, resulting in good learning outcome.

This class can serve as a good team project course for senior students in Computer Science, so they can taste the industry style problem solving approach and team work in academic environment. Also as a by-product, the class produced a usable game that is currently used in CSc 210 Introduction to Computer Programming course at SFSU. The objective of using this game for educating CS freshman seemed to give pressure to students to deliver a smoothly working game as their clients are sitting right next to them, so it lets them alert about their end user.

The project's focus is not only producing the end product, but assessing/evaluating students' learning from the experience. We have created rubrics to measure students learning (Table 1) and plan to assess and evaluate them thoroughly in Spring 2014.

To be able to replicate the same class in the future, course material needs to be better collected and organized. There are hundreds of discussion threads at iLearn news group that were effective while being used, but not so organized, so may not be transferrable to next year class nicely. Inter-team communications were well captured to be used in the class for each team progress presentation and documentations. But inner team communication and resources were not saved. Each team created very

useful resources to be shared within team to learn the required technology for each team for example, client team learned Panda3D, Python, collision handling together helping each other. Same applied for each team. Vast amount of self study materials were created by each team. These can be better managed to be created at wiki, so they can produce more re-usable study resources.

ACKNOWLEDGEMENTS

This project is being funded by National Science Foundation Div. Of Biological Infrastructure, Biological Databases and Information, NSF DBI- 0543614. Also this project is recently funded by NSF Transform Undergraduate Education (TUES) 1140939 for 3 years to prove the learning efficacy quantitatively and spread widely.

REFERENCES

Dutoit, A. H., Bruegge, B, 1998 "Communication metrics for software development," Software Engineering, IEEE Transactions on, Vol. 24 Issue: 8, pp. 615 – 628.

Fuggetta, 2000 "Software process: a roadmap," ICSE '00 Proceedings of the Conference on the Future of Software Engineering, ISBN:1-58113-253-0.

Jones, R. M., 2000 Design and implementation of computer games: A capstone course for undergraduate computer science education. In Proceedings of the 31st SIGCSE Technical Symposium on Computer Science Education, pages 260–264. ACM Press, 2000.

Mencher, M., 2003 "Get in the Game!," New Riders Publishing.

Moser, 1997 "A fantasy adventure game as a learning environment: Why learning to program is so difficult and what can be done about it." In Proceedings of the 2nd Conference on Integrating Technology into Computer Science Education, pages 114–116. ACM Press, 1997.

Parberry, 2001 "Introduction to Computer Game Programming with DirectX 8.0." Wordware Publishing, 2001.

Parberry, I., Roden, T., Kazemzadeh M.B., 2005 "Experience with an industry-driven capstone course on game programming: extended abstract," Proceedings of the 36th SIGCSE technical symposium on Computer Science Education, Vol. 37 Issue 1.

Song, S., Lee, J., 2007 "Key factors of heuristic evaluation for game design: Towards massively multi-player online role-playing game, International Journal of Human-Computer Studies, no. 65, 709-723.

Yoon et al. 2011 Ilmi Yoon, Gary Ng*, Zoran Millic*, Supakit Kiatrungrit*, Yiyi Miao*, and Sunggye Hong, "Educational Multiuser Online Game, 'DeDugger' Game for Introductory Computer Science Class," International Conference on Frontiers in Education: Computer Science and Computer Engineering, July, 2011, pg. 393-398.

Yoon et al. 2013 Yoon, I., Ng, G., Rodrigues, H., Nguyen, T., Paik, J., Yoon, S., Williams, R., Martinez, N., "Iterative Design and Development of the 'World of Balance' Game: From Ecosystem Education to Scientific Discovery," IEEE International Games Innovation Conference (IGIC), Sept. 2013, Vancouver, British Columbia, Canada, Pg. 283-290.

Yoon et al. 2013b Ilmi Yoon, Arno Puder, Gary Ng*, Manori Thakur, Hunvil Rodrigues, Jae H. Paik, Eun-Young Kang, "Educational MMORPG for Computer Science: DeBugger, a Virtual Lab on PC and Smart Phones," International Workshop on Distance Education Technologies, Brighton, United Kingdom, August, 2013, pg. 96-10.

APPENDIX

Table 1: Project goals and corresponding assessment mechanism.

Goal 1: Teach students effective communication, presentation and collaboration skills that are often neglected in the traditional classroom.	
Assessment Rubric	Assessment Mechanism
• Students presentation skills both oral and written • Ability to communicate complex CS concepts • Students ability to work in groups • Organizational skills • Focus of task • Students ability to communicate clearly via exchange of shared documents and protocols with team members	• Students evaluation of skills and competencies via survey • Interviews and focus group discussion • Classroom observations (via a protocol) • Records of public presentations/journal publications at conferences and professional meetings

Table 1: Project goals and corresponding assessment mechanism. (Cont.)

Goal 2: Motivate students to learn essential CS course content through peer interaction and encouragement, thereby increasing course engagement and making learning experience relevant and fun	
Assessment Rubric	Assessment Mechanism
• Student reflections of their roles • Student reflections of their own learning process • Students skills and strategies to engage in the design and development process to create and engaging product.	• Students' course satisfaction survey • Interview and focus group discussion • Student self-evaluation
Goal 3: Enable students to produce a robust end product, i.e., a game that can be played by the public and that can be continuously expanded by other students in successive classes	
Assessment Rubric	Assessment Mechanism
• Students design and development skills • Students critical thinking skills applied to the development of the game • Student content knowledge • Originality of ideas	• Performance evaluation of the game design process • Interview and focus group discussion • Protocol to measure effectiveness of the product (MMORPG game) developed • Student self-evaluation
Goal 4: Give students insights into the real-world industrial work environment to prepare them to master the challenges of a CS career	
Assessment Rubric	Assessment Mechanism
• Effectiveness of the class in providing hands-on experience and skills to succeed in CS career. • Students' determination to pursue a CS degree • Long-terms career goals in CS or other STEM field	• Interview and focus group discussion • Student on-line surveys