2006-800: GAME DEVELOPMENT IS MORE THAN PROGRAMMING

Bruce Maxim, University of Michigan

Professor Maxim is Associate Professor of Computer and Information Science at the University of Michigan -Dearborn. He has taught game design, artificial intelligence, and software engineering courses for 20 years. His current research interests include software usability, accessibility issues, and software quality assurance.

Game Development is More Than Programming

Abstract

Game development generates a great deal of excitement among undergraduate computing students. Many students are disappointed to find that they will not learn how to build computer games in their required computing courses. The author created a two-semester sequence of courses focusing on the application of software engineering principles to the design and implementation of computer games. These courses build on the material covered in the first course on software engineering taken by junior level students. This paper summarizes the content of these courses and the author's experiences in teaching game design during the past six years.

Introduction

The annual revenue generated from the sales of computer games in the United States alone exceeds $7 billion dollars. Statistics indicate that the revenue generated by the computer game industry will continue to exceed that of the motion picture industry.[7] Computer game development is big business.

The development of computer games is labor-intensive. Today, game developers rarely build computer games on their own, as they did 15 years ago. Many best-selling computer games contain thousands of lines of code and have multi-million dollar development budgets. Modern game development requires the effort of a team of skilled professionals to integrate multimedia content and complex computer software. Game development projects have a reputation for late delivery times and cost over runs. In December 2005, consumers observed hardware failures in the first Xbox 360 consoles delivered to consumers and the recall of a popular Nintendo Game Cube software product. The minimum costs incurred by a failed game development project ranges between $150,000 and $750,000.[13] Producing high-quality software products by large teams requires high levels of communication, organization, and planning to avoid costly delays and failures.

Game developers are beginning to understand that it is important to treat computer game design in the same way that other software engineers approach projects involving a large number of people and a significant investment of time.[13] Game developers are likely to benefit from using evolutionary software process models to mange their development risks and reduce their project completion times. The process of determining the technical requirements for a game software product is similar to that used to specify any other type of software product. However, unlike most software products, games have an entertainment dimension. People play computer games because games are fun.[8]

The International Game Developers Association (IGDA) proposed a curriculum framework for university level training in game development.[5] The core topic areas from the IGDA recommendations appear in Table 1. Many of these topics involve the application of skills taught in software engineering courses.

Table 1: IGDA Curriculum Framework

Core Topic	Key Elements
Critical Game Studies	game evaluation, usability assessment, genre analysis, media history
Games and Society	player demographics, game cultures, game affect, game applications
Game Design	game/play mechanics, game theory, interface design, user task analysis, design tradeoffs, quality assurance
Game Programming	physics, information design, artificial intelligence, networking, multimedia programming, implementation tools, prototyping, testing
Visual Design	architectural design, information visualization, computer graphics, animation, cinematography, 3D hardware
Audio Design	audio theory, audio aesthetics, music, sound effects, 3D audio
Interactive Storytelling	hypertext design, interactive fiction, plot exposition, character development
Game Production	project management, team dynamics, documentation, communications skills, quality assurance
Business of Gaming	industry economics, marketing, contracts, intellectual property

Textbooks emphasizing software engineering principles for game development are beginning to appear.[8, 12, 13, 14] The SE2004 curriculum recommendation suggests multi-media software production as a possible application area in undergraduate software engineering degree programs.[15] It is difficult to add courses to already crowded curricula so new courses need to make multiple contributions to the student's educational experience.

Recent educational computing conference proceedings have included descriptions of capstone courses where students implement computer games as a means of demonstrating their ability to apply their knowledge of computer science principles.[3, 6, 9, 10] Professors have argued that game applications provide fun and realistic projects that can help to motivate students to achieve better performance on software engineering class projects.[1, 2, 4, 6, 11] Studying game development can prepare students for entry into the game programming industry and high performance application areas.[10, 16]

Students just learning software engineering principles and practices find it difficult to apply them in the development of complex software projects. It is important to do more than simply use a game for the term project of a software engineering course, as some authors have suggested.[1, 6, 11] Real software engineering involves acquiring application domain knowledge in order to understand the client's needs. Adding game topics to crowded software engineering courses, as some authors have suggested,[2, 6] requires sacrificing important software engineering topics. Focusing on one application area in the first software engineering class is not fair to all students. Not every software engineering student wants to become a game developer. The author believes that the capstone design course should not be the only opportunity for students to manage complex software development projects. This suggests the use of elective courses as a means of providing additional software engineering experiences.

The author created a two-semester course sequence focusing on the application of software engineering principles in the development of computer games. Students take these courses after they complete the junior level course on software engineering required of all software engineering majors. This course sequence can be used by students to satisfy the application area requirement of the department's ABET accredited software engineering (BSSE) degree. The author's institution also offers an undergraduate game design minor and a graduate level certificate in game design. This paper summarizes the content of the game design courses and the author's experience in teaching game design during the past six years.

Courses

CIS 487 (Computer Game Design and Implementation I) deals with the study of the technology, science, and art involved in the creation of computer games. The focus of this course is on the application of software engineering methods in the hands-on development of computer games. Students study a variety of software technologies relevant to computer game design, including: simulation and modeling, computer graphics, artificial intelligence (AI), real-time processing, game theory, software engineering, human computer interaction, graphic design, game aesthetics and multi-media system design. The sequence of course topics appears in Table 2.

Table 2: CIS 487 Course Topics

Course Topic	Software Engineering Elements
Games and Society	player demographics, use case scenarios, game cultures, game applications, media history
Game Evaluation Criteria	game evaluation, usability assessment, game genre analysis
Game Design Principles	game play mechanics, game theory, design tradeoffs, information design, design documentation
Interactive Fiction	hypertext design, interactive fiction, plot exposition, character development
User Interface Design	user task analysis, documentation, interaction devices, information visualization, audio aesthetics
Game Production Methods	project management, team dynamics, written and oral communications skills
Game Architecture	software architectural design, graphics hardware organization
Game Programming	game physics simulation, heuristic game AI, networking, graphics and multimedia programming, tool use
Game Testing	quality assurance, testing, usability assessment, formal technical reviews
Game Asset Creation	audio theory, sound effects, computer graphics, animation
Business of Game Development	industry economics, marketing, contracts, intellectual property

The lectures for this course focus on game design issues, implementation tradeoffs, and project management practices needed for every successful software development project. Discussion of unsuccessful game products helps to illustrate the value of careful project management and the importance of understanding the characteristics of a software product's target audience. Game development projects have large budgets, short timelines, and high risks of failure. Managing these risks is possible only by adopting good software engineering practices as part of the game development process. Discussion of agile software process models and software quality practices as they apply to game development is an important part of this course.

The student work for this course includes the completion of several projects. All projects include design activities and students make use of several existing programming tools. Making use of existing programming tools and libraries allows students to focus on software engineering design rather writing all source code from scratch. The final project requires students to go through all phases of system life cycle: specification, design, implementation, testing, and evaluation. A description of the each CIS 487 project appears in the next section of this paper.

Grades assigned to software engineering documents determine 60% of the student's course grade. Grades on student presentations, peer reviews, and software determine the remaining 40% of the course grade. The assessment outcomes for this course are:

- Students demonstrate their ability to determine the requirements for computerized solution to a real-world problem.
- Students create analysis models for a game software product.
- Students demonstrate their ability to design a computerized solution to a real-world problem.
- Students use software tools and specialized game development environments to implement solutions to a real world problem.
- Students participate in the peer review of software engineering documents and software products.
- Students assess the quality of game products.
- Students apply techniques for testing computer games.

CIS 488 (Computer Game Design and Implementation II) continues study of the material covered in CIS 487. The course focuses on the use of team software engineering process in development of computer games and the use of game development tools (e.g. game engines). Students study a variety of software technologies relevant to computer game design, including: 3D graphics, computer animation, data-driven game design, multiplayer game programming, game AI, game theory, software engineering, and game content development. The sequence of the course topics appears in Table 3.

Even though this course appears to contain more implementation than design topics, presentation of these topics is from the perspective of a software engineer. The material associated with 3D graphics and game AI emphasizes the design tradeoffs required in selecting implementation mechanisms. One important topic of discussion in this course is managing the process of outsourcing game asset creation to students attending college in another city. Students quickly learn to appreciate the importance of up to date written documentation, the early establishment of software requirements and standards, and the value of version control repositories. Students

experience firsthand the benefits and compromises that accompany working with common off the shelf software (COTS) products such as game engines. Much of the software design discussion centers on techniques to facilitate the creation of reusable software components that can be included in future game products without modification.

Table 3: CIS 488 Course Topics

Course Topic	Software Engineering Elements
Game Design Principles	game analysis models, design tradeoffs, information design, algorithm design, math engine design, game agent design, user interface design
Game Production Methods	project management, team organization, feasibility study, design documentation, software configuration management
Game Asset Creation	music, sound effects, computer graphics, animation, cinematography, asset version control
Game Programming	3D graphics, 3D clipping, depth buffering, texturing, lighting, shadows, 3D animation, collision detection, game engine use (COTS)
Game AI	reactive agents, rule-based systems, state-based reasoning, navigation, obstacle avoidance, tactics, weapon selection, fuzzy logic, learning
Game Testing	quality assurance, prototyping, testing, usability assessment, formal technical reviews

The term project for this course requires each student to participate as a member of a multi-disciplinary team to develop a 3D multimedia computer game. Teams have team members assigned duties related to project management, software quality assurance, documentation, and version control as well as game asset creation. Teams meet frequently to ensure that the software engineering and game asset creation processed are proceeding according to the project schedule. The term project requires students to work through all phases of system life cycle: specification, design, implementation, testing, and evaluation. Students are required to deliver several project milestones as part of the incremental development of the final game product. Each milestone delivery requires an oral presentation and significant peer review of the accompanying work products. Descriptions of the CIS 488 project milestone assignments appear in the next section of this paper.

Grades assigned to software engineering documents determine 70% of the student's course grade. Grades on student presentations, peer reviews, and software determine the remaining 30% of the course grade. The assessment outcomes for this course are:

- Develop the requirements for a 3D multi-media computer game.
- Students create UML models for a game software product.
- Design a 3D multi-media computer game and create design documents for the game.
- Develop the requirements for an intelligent opponent for a computer game.
- Design an intelligent opponent and integrate it into an existing game prototype.

- Students participate in the peer review of software engineering documents and software products.
- Participate on a multidisciplinary team to implement a software system to meet the needs of a real-world problem.

Projects

The projects completed by students in these courses reinforce the software design and project management skills taught in their junior level software engineering courses. Brief descriptions of the course projects appear in this paper. The detailed assignment sheets are available from the author's personal web site.[17] Copies of the actual student work products from CIS 487 and CIS 488 appear on course web pages.[18, 19]

Students taking CIS 487 complete four projects and participate in the technical reviews of work products produced by their classmates. The focus of the projects assigned to CIS 487 students is on user profiling, user task analysis, game design, and software quality assessment.

Project 1 (CIS 487): Game Quality Assessment. (2 weeks)
Students select a commercially produced computer game and evaluate its quality using criteria discussed in class. Each student prepares a written report summarizing the findings of his or her review and makes an oral presentation of the review to the class using MS PowerPoint.

Project 2 (CIS 487): Text Adventure Game Design. (4 weeks)
Students develop an analysis model and design document for an original text-based adventure game. The game implementation involves using Inform (an existing commercial text-based game engine). Using text-based game engine forces students to concentrate on developing storyline, user interaction design, and game play without relying on graphical pyrotechnics to hide a weak game design. Reviewers use a checklist to assess the quality of the complete game. Several students assess each game and the game designer has an opportunity to revise the game based on peer feedback.

Project 3 (CIS 487): Multimedia Game Design Treatment (4 weeks)
Students create the requirements, analysis model, and a preliminary design document (game treatment) for an original multimedia computer game. A formal technical walkthrough of the design document is part of the peer review process. Several students acting as reviewers provide feedback to each game author following the presentation of the game design document. The reviewers make use of a checklist to assess the quality of the design document. The game designer uses the reviewer feedback to revise the preliminary design document.

Project 4 (CIS 487): Multimedia Game Development (8 weeks)
Students create a complete design document for a multi-media computer game and implement the game using specialized game development tools (Visual C++ and the DirectX SDK). Several students play test each final game product and provide constructive feedback to the author regarding its over all quality.

Students taking CIS 488 participate as part of an interdisciplinary team in the semester long development of a 3D computer game. CIS 488 students handle the software engineering and project management aspects of a game. Students from the near by College of Creative Studies create the multimedia assets required to implement the game. CIS 488 students complete several assignments as project milestones in the process of developing the game.

Project 1 (CIS 488): 3D Game Pitch Document (1 weeks)
Each student completes a short game pitch document for an original 3D multimedia game and presents it to the class. The students assess each game for feasibility. The instructor and students consult in selecting two or three games for further development.

Project 2 (CIS 488): 3D Game Design Treatment (3 weeks)
Students organize themselves into teams consisting of five software engineers and five game asset developers. Students are free to structure their teams using any model they wish. Students refine the requirements and create a preliminary design document for the first incremental prototype of a 3D computer game. The game teams use the peer reviewer feedback to revise the preliminary design document.

Project 3 (CIS 488): Alpha Release Prototype and UML Modeling (4 weeks)
Teams create UML models for their complete games and create working game prototypes using specialized game development tools (Visual C++ and the DirectX SDK or Torque). A formal technical walkthrough of the UML model is part of the peer review process. Students outside of the development team play test the game prototype. The teams use reviewer feedback to revise their design documents.

Project 4 (CIS 488): Game AI Quality Assessment (2 weeks)
Each student selects a commercially produced computer game and evaluates the nature and quality of the game AI techniques using criteria discussed in class. Each student prepares a written report summarizing the findings of his or her review and makes an oral presentation of the review to the class using MS PowerPoint. The intent of this assignment is to help students determine the feasibility of incorporating specific types of game AI in the next releases of their evolving game products.

Project 5 (CIS 488): Beta Release Prototype and Design Document (4 weeks)
Teams develop the requirements for an intelligent agent or NPC (non-playing character) to add to their game. The implementation of the intelligent agent becomes part of an incremental release of the game product. Revision of design document reflects this change and regression testing occurs to ensure that implementation of the intelligent agent has not broken the game. Students outside of the development team play test the new prototype. A formal technical walkthrough of the design document is part of the peer review process. The team uses reviewer feedback to revise the game design.

Project 6 (CIS 488): Gold Release and Product Testing (14 weeks)
Students deliver the final design document for a multi-media computer game and the final game software. Several students play test each of the final game products and provide constructive feedback regarding the over all quality of each to the development teams.

Summary

The attrition rate from the author's game design courses is less than that of many other 400 level CIS courses (typically only 1 or 2 students from a class of 24). Table 4 contains the average scores for the course evaluation items for the game design courses (4 = completely agree, 0= completely disagree) and the overall course rating (4=highest, 0=lowest). The course evaluation forms for each course indicate above average levels of satisfaction on the part of the students.

Table 4: Student Course Evaluations

	CIS 487 Fall 2004	CIS 488 Winter 2005
Course fulfilled my needs	4.0	3.3
Course objectives were clear	3.9	3.6
Course prerequisites are adequate	3.1	3.7
Course was challenging and interesting	4.0	3.3
Course material is up to date	4.0	3.6
Course never repeats material from other courses	3.9	2.9
Overall course rating	3.9	3.6

Most students taking CIS 487 were undergraduate students. The author has taught CIS 487 seven times. The only course prerequisite for CIS 487 is the department's first Software Engineering course. Some students felt that a course in Windows graphics programming would be a useful course prerequisite for CIS 487.

Winter 2005 was the first offering of CIS 488. A majority of the students taking CIS 488 were graduate students who took CIS 487 as undergraduates. Many students wanted to work on larger projects and to work on team projects. Students wanted to work with commercial game engines. One common complaint in both CIS 487 and 488 was that creation of game art took time away from their software development and documentation efforts. Some of the graduate students felt that the game AI content repeated content covered in the department's Artificial Intelligence course (which is not a prerequisite for CIS 488).

Student course feedback from this initial offering proved valuable in making modifications to version of CIS 488 appearing in this paper. The author applied to the University for funding to purchase the Torque game engine for student use. The CIS department negotiated an agreement to coordinate course offerings with the Animation and Digital Media department at the College of Creative Studies. The course projects were redesigned to become part of a multidisciplinary team-based term long project. The new version of CIS 488 is part of our Winter 2006 semester course offerings.

Many prospective students inquire about availability of game design courses as part of their degree work in Computer Science, Information Systems, or Software Engineering. Our department offers ABET accredited degrees in each of these majors. CIS 487 and 488 satisfy part of the degree requirements for any of these degrees. Our ABET reviewers were satisfied that our game design courses cover a sufficient level of software engineering to justify their inclusion as

an application area in our BSSE degree. Eight of the author's students have gone on to complete game-based masters projects. Three of the author's former students are working full-time as software engineers in the game industry. Even students who do not plan to work in the game industry appreciate the opportunity to create larger and more complex programs as part of the game design projects. All students acknowledge that creating these games within the time allowed in each course is only possible because of their software engineering training.

Bibliography

1. Becker, K. Teaching with games: the minesweeper and asteroids experience. *Journal of Computing Sciences in Colleges* 17 2 (December 2001), 23-33.
2. Claypool, K. and Claypool, M. Software engineering design: teaching software engineering through game design. In *Proceedings of 10th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education* (Capprica, Portugal, June, 2005), ACM Press, New York, NY, 2005, 123-127.
3. Coleman, R., Krebs, M. Labouseur, A., and Weir, J. Game design and programming concentration within the computer science curriculum. In *Proceedings of 36th SIGCSE Technical Symposium* (St. Louis, MO, February, 2005), ACM Press, New York, NY, 2005, 454-500.
4. Coulton, P. and Edwards, R. Could game design become a core subject for engineering? In *Proceedings of 33rd Annual Frontiers in Education* (vol. 1, 2003), IEEE Press, Los Alamitos, CA, 2003, 2-21.
5. *International Game Developers Association Curriculum Framework*. Retrieved September 6, 2005 from http://www.igda.org/academia/curriculum_framework.php.
6. Jones, R. Design and implementation of a computer games: a capstone course for undergraduate computer science education. In *Proceedings of 31st SIGCSE Technical Symposium* (Austin, TX, March, 2000), ACM Press, New York, NY, 2000, 260-264.
7. Maxim, B. R. Game design: games for and the World Wide Web. In *The Internet Encyclopedia*, Wiley, Hoboken, NJ, 2004.
8. Maxim, B. R., *Software Requirements Analysis and Design*, NIIT, Atlanta, GA 2004.
9. Overmars, M. Teaching computer science through game design. *Computer* 37 5 (April 2004), 81-83.
10. Parberry, I., Roden, T., and Kazenzadeh, M. Experience with an industry-driven capstone course on game programming, an extended abstract. In *Proceedings of 36th SIGCSE Technical Symposium* (St. Louis, MO, February, 2005), ACM Press, New York, NY, 2005, 91-96.
11. Pleva, G. Game programming and the myth of child's play. *Journal of Computing Sciences in Colleges* 20 2 (December 2004), 125-136.
12. Rabin, S. *Introduction to Game Development.* Charles Rivers Media, Hingham, MA, 2005.
13. Rollings, A. and Morris, D. *Game Architecture and Design.* New Riders, Indianapolis, IN, 2004.
14. Rouse, R. *Game Design: Theory and Practice.* Wordware, Plano, TX, 2001.
15. *Software Engineering Curriuclum 2004*. Retrieved September 30, 2005 from http://sites.computer.org/ccse/SE2004Volume.pdf
16. Sweedyk, E. and Keller, R. Software engineering design: fun and games: a new software engineering course. In *Proceedings of 10th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education* (Capprica, Portugal, June, 2005), ACM Press, New York, NY, 2005, 138-142.
17. <http://www.engin.umd.umich.edu/~bmaxim/> (10 January 2006).
18. <http://www.engin.umd.umich.edu/CIS/course.des/cis487.html> (10 January 2006)
19. <http://www.engin.umd.umich.edu/CIS/course.des/cis488.html> (10 January 2006)