

Interweaving Game Design into Core CS Curriculum

Yolanda Rankin
Northwestern University
2133 Sheridan Road
Evanston, IL 60208
847-467-5635

yrankin@northwestern.edu

Bruce Gooch
Northwestern University
2133 Sheridan Road
Evanston, IL 60208
847-491-3500

bgooch@cs.northwestern.edu

Amy Gooch
University of Victoria
CS Department & ECS Bldg 504
PO Box 3055, STN CSC
Victoria, BC Canada V8W 3P6

Amy.a.gooch@gmail.com

Abstract

Computer Science departments across the country have embraced computer gaming classes as part of the core curriculum. However, instructors need to define guidelines that accommodate students' proficiency in game development and consequently address the growing needs of industry. We develop and evaluate a Game Authoring Class in an attempt to begin to close the gap between industry and academia. Learning objectives include students applying game development concepts to implementation of 2D and 3D games for multiple platforms. Subsequently, we evaluate the course based on two factors: formal assessment of students' understanding of game design principles and students' perceived learning. The results of our evaluation serve as the basis for establishing effective pedagogical strategies for game development curriculum.

Categories and Subject Descriptors

K.3.2 [Computer and Information Science Education]: computer science education, computer graphics.

Keywords

Computer science curriculum, game development, evaluation, educational assessment, pedagogy.

1. Introduction

Currently, enrollment in Computer Science programs across the nation has dropped 60% and the trend shows a continued decrease over the next few years [16][17]. While industry faces the impact of fewer computer science majors, academia has employed several different approaches to address this critical situation. In an attempt to combat the declining interest in computer science, computer science departments are leveraging the appeal of video games to entice the next generation of computer science majors. Thus, game development courses are quickly becoming a part of the Computer Science curriculum at universities and colleges across the country.

However, changes in the curriculum do not necessarily equate to students developing the necessary software skills and expertise appreciated by the game industry. Oftentimes discrepancies exist between the learning goals presented in a classroom setting and the skills and

experience that are specific to the needs of the gaming industry [1][2][3][7]. It is not sufficient to simply write code that creates a game if students have failed to grasp design concepts that are crucial for game development.

In response to this dilemma, we design a Game Authoring Class and teach the course at Northwestern University during Spring Quarter 2006. We propose modeling game development courses as closely to the software development process followed in game industry. The course emphasizes research in the field of computer games and hands-on experience developing games. Students review current trends in computer game programming and build their own 2D and 3D games on top of available game engines[7][10][15]. Additionally, this class encompasses building games for multiple devices, such as cell phones, PDAs, Pocket PCs, desktops and laptops. Adopting industry game design principles, we require students to implement these design principles in three project deliverables: 1) game design document and play-test criteria, 2) 2D game for a mobile device, and 3) 3D game module that runs on an existing game engine. Finally, we review the Game Authoring class and establish best practices for teaching any game development class that prepares students for careers in the gaming industry.

2. Apprenticeship Game Design

Research shows that video games are an often under-utilized learning environment that can be extended to various domains, including mathematics, physics, history, and language learning [6][8][9][14][15][16]. We delve into yet another domain, that of computer science, maximizing the attraction of games to revamp traditional computer science curriculum and train the next generation of software developers. The goal is to close the widening gap between industry and academia by modeling industry practices; before we can accomplish this goal, we must incorporate pedagogical strategies into the design of game development courses.

Cognitive scientists understand that learning occurs in the context of meaningful tasks [4][5][8][11]. In actuality, these tasks represent regular practices of a particular community or group of people. Rather than introducing theoretical concepts separate from real-world applications,

effective teachers guide students through situated activities that demand the application of information or a specific skill [4][5][11]. As the student participates in activities under the guidance of the teacher, the student develops the skill set required to complete the task. Cognitive scientists refer to this as the apprenticeship model [5].

We extend the model of cognitive apprenticeship to the design and instruction of the Game Authoring Course. We carefully select assignments (e.g. writing a game design document, conducting play-tests of game prototype, etc.) that reflect authentic game design practices in the gaming industry. As students acquire knowledge necessary to complete the assignments, they develop proficiency using the tools of the trade. Under the careful guidance of the instructor, class discussions create a safe environment in which learners feel that they contribute to a community of learners while gaining expertise in a particular activity or task. Game play demonstrations give students the opportunity to assess their peers' work and to refine their ideas about what it means to be a game designer.

Interaction with students enrolled in the course provides secondary means for support and self-reflection as students develop proficient skills as game designers. Additionally, the assignments instantiate tasks that game developers do on a regular basis. Hence, the project deliverables represent opportunities for situated learning as students play the role of game developers throughout the Game Authoring Course. We now turn our attention to the learning objectives for each project and assess students' mastery of game design principles.

3. Game Authoring Course

We taught the Game Authoring class during the Spring Quarter 2006. Fourteen students, thirteen male and one female, were enrolled in the class. Fifty percent of the students were computer science majors. The class was scheduled for 10 consecutive weeks with the class meeting twice a week for lectures during the first seven weeks and ending with three weeks of laboratory meetings. The course required students to develop at a minimum level of basic proficiency for standard software development tools such as Microsoft Visual Studio, Microsoft DirectX, C++, OpenGL, and Python. One graduate and three undergraduate students were designated as teacher assistants (TAs) for the course. The TAs created a class wiki (<http://cs.northwestern.edu/~gaming/>), which facilitated communication between students and the instructor/TAs and served as a digital notebook of students' progress on assignments. Students were responsible for three projects: 1) creating a web-based game design document and defining play-test criteria for evaluating games, 2) developing a 2D game for a mobile device, and 3) developing a 3D game module that runs on top of the Half-Life2 game engine.

Project 1: Game Design Document

The first assignment set the stage for students to think deeply about the structure of games (i.e. players, objectives, rules, procedures, conflict and possible outcomes) and the dramatic elements (story, challenge, sense of fun, etc.) that create memorable experiences for gamers [7][12][16]. To assist students with writing a game design document, students first wrote a review comparing and contrasting two games of their choice. Students evaluated the games according to the following criteria:

- What is the appropriate audience for this game?
- What is fun about the game and why?
- What is bad/not fun about the game and why?
- How does it compare to similar games in the genre?
- Why is it better or worse than similar games?
- What are the highlights/low points of game play?

Class participants posted game reviews on the wiki. One student added additional criteria, asking the following questions: "Does the game's artwork portray a realistic world or does it use a more abstract artistic representation? Is the artistic style reminiscent of another game's artwork or even art style in a different medium (such as a style of painting or architecture)? Does the game's artwork fit the mood of the game's story, environment, and game play?"

These questions indicate an understanding of how every detail adds to the immersive environment of games; one poorly conceived design detail could destroy this carefully crafted virtual world. As a result, each student identified play-test criteria used to evaluate the 2D mobile games developed by their peers. Thus, the assignment functioned as the initial point for students' forming a foundation that enables them to incorporate game design principles without sacrificing the element of fun.

Project 2: Developing a 2D Mobile Game

The second assignment emphasized the iterative process of game design and the importance of involving users in the early stages of game development [7]. Initially, students were required to use the Microsoft DirectX software development kit to design the 2D mobile game. However, the three undergraduate TAs and only one student enrolled in the class were able to get the DirectX SDK downloaded, installed and running. The remaining 11 students in the course experienced difficulty installing and executing the DirectX SDK and requested to use other software development tools to complete the assignment. Adequate support for use of software development tools is necessary if we expect academia to prepare students for employment in the game industry. Therefore, continuous communication between industry and academia helps to ensure that the computer science curriculum reflects the needs of industry. The remaining students relied on

additional software development tools (PyGames, Flash 5.0, JavaScript, etc.) to implement working prototypes. While this was an unexpected outcome, it actually gave students experience developing 2D games for various platforms, a transferable skill to industry.

Students followed the iterative design process, generating storyboards, formalizing formal and dramatic elements, implementing a working prototype and conducting play testing [7][16]. Play-testing occurred on multiple levels, including self-test testing, testing with friends, and testing with strangers [7]. Self-testing ensured that the game demonstrated limited functionality (e.g. ability for players to navigate game controls) before release of the working prototype. In contrast, peer testers used the RITE method to provide immediate evaluation and recommended changes that would enhance the game play experience [13]. Students made changes accordingly and demonstrated the final version of the game during class time.



Figure 1. Parachute 2D Mobile Game.

Project 3: 3D Game Module

The third project gave students the opportunity to gain hands-on experience designing a 3D game module on top of the Half-Life2 game engine. The three undergraduate TA's had developed a game module entitled "Project Echo" in a previous undergraduate course. Project Echo is a first-person shooter that features one game level, two weapons (machine gun and leech launcher), sound effects for weapons and limited AI for non-playing characters (NPCs). Due to time constraints, students were tasked with software modifications that would increase game functionality and improve the game play experience. Before any software changes were made, students play-tested Project Echo and identified the strengths and weakness of the game. Students were encouraged to work in groups to foster teamwork typical of the game industry which involves a team of developers, including graphics artists, animation artists, user interface programmers, sound engineers, etc. Students pitched the high-level concepts of their game modifications, outlining a project schedule for software

development. All files and executables, documentation, and proposals were posted to the class wiki.

Game modifications covered a wide range of additions: students created new models for weapons; rigged and skinned a model of a monster in Maya; designed an additional game level using Half-Life2's Hammer tool; added customized sound effects that were scripted to promote emotional involvement during game play; implemented a more sophisticated AI to allow NPCs to harvest local resources; and revised the game interface to give players current status information.

4. Best Practices

Each project afforded us the opportunity to identify what worked and what failed as it related to our learning objectives. In addition, 64% of the class completed Northwestern University Course and Teacher Evaluation Councils (CTECs), which gave us insight into students' expectations and opinions of the class. Based on student evaluations and lessons learned assessment, we identified effective practices for game development curriculum.

4.1 Pipeline between industry and academia

If there were one thing that we could do differently, it would be to engage companies in dialogue with our students as part of the course. One participant suggested that the course include guest speakers; this would have enabled students to establish connections with industry. Additionally, students expressed frustration with the inability to get DirectX up and running for the 2D mobile game project. One student reported, "We were supposed to use certain software to program a game but apparently the TA's never properly prepared the computers for this so we resorted to a different program." Unfortunately, this gave the student the impression that 2D mobile project was not well prepared. Oftentimes instructors as well as students experience a learning curve before becoming proficient in the use of industry tools. This feedback led us to consider direct access to Microsoft DirectX personnel as we modify the syllabus for future class objectives and assignments. By establishing a direct pipeline between industry and academia, we can attain real-time customer support, shorten the learning curve and assist faculty and students with developing proficiency of software development tools. This becomes a crucial factor for hiring qualified employees that can immediately contribute to company revenue.

4.2 Incorporating Industry practices

Industry has criticized academia for failing to embrace the latest technological practices and advancements supported by industry. In an attempt to address this issue, we

designed a game development course that closely followed industry practices and methodology for game development. It is common practice for game designers to first create a paper prototype of a game as a means for testing their ideas [7][17]. To emulate this practice, students presented a paper prototype of their 2D game to their classmates before any software development. As a result, classmates helped students refine game mechanics, suggesting ways to increase the level of difficulty and identifying potential problems that might impede enjoyment of the game.

The Game Authoring class required students to embrace the iterative game design paradigm for both the 2D mobile game project and the 3D game module. Students created storyboards, identified the structure of the game, and implemented a working prototype [7]. As students communicate their ideas to their classmates, these same classmates would comment on the ideas, helping the author to conceptualize the formal and dramatic elements that support a memorable game play experience. Play-testing on multiple levels ensures early feedback and prevents costly mistakes that produce poorly designed games.

4.3 Teamwork amongst students

The 3D game module included a written assignment regarding the “lessons learned” over the course of the last project, forcing students to compare their initial design proposal with the actual work completed. Students frequently admitted that they did not accomplish all of the goals. This failure was attributed to two factors. First, students underestimated the amount of work required to develop a game module. For example, before one can make any software changes, one must first sift through several lines of code to determine the logical flow. Once the student traces the logical flow, the student must master the software development tools (e.g. XSI for creating models of weapons) to implement code changes. Secondly, 71% of the class chose to work solo on the 3D game module project. Experienced game developers know that it takes a team approximately 2 – 3 years to design a game that may or may not reach the consumer market. This team of people is comprised of graphic artists, voice actors/actresses, sound engineers, music editors, user interface programmers, level designers, etc. who meet constantly to review the development schedule and work together to produce one final product [7]. Thus, our students failed to appreciate the benefits of teamwork. As a result, the amount of time students spent working on the game module did not correlate to high quality product created by students. One student suggested that we coordinate teams of students to design the game module. By allowing the majority of students to work alone, we missed a great opportunity to teach students that teamwork is a requisite for the gaming industry and that teamwork results in greater productivity. To better address this issue,

we suggest that the assignment should be subdivided into individual assignments that form the whole product. Thus, no two students will have the same assignment and yet each assignment produces a component that is crucial to the construction of the final product.



Figure 2. Game level in for 3D Game Mod.

5. Course Assessment

64% of students completed a single blind survey that assessed students’ perception of the class. In summary, students ranked the overall instruction 4.67 on a scale of 1 to 5 with 5 being the highest and 1 being the lowest. Scrutiny revealed that 30% of the survey participants thought the overall course was excellent while approximately 44% indicated that the class was a good or satisfactory course. We contribute the students’ high rank of instruction to the instructor’s ability to communicate his enthusiasm for game design and ability to establish rapport with the students. This led us to question whether students were learning anything in the class. More than 40% of survey participants expressed that they had learned a substantial amount about the game development process; 22.22% believed they had attained some knowledge. Only, 11.11% of the students indicated they had minimally increased their knowledge because of taking the course.

The fact that less than half of the class felt that they had learned a substantial amount raised a red flag, perhaps some students were not learning due to the open-ended structure of laboratory exercises. During the last three weeks of class, students attended programming laboratory to attain assistance with debugging code and develop proficiency with specific software packages such as XSI Modeling toolkit in the absence of course lectures. Students commented that reviewing the code for the Half-Life2 game engine was a lengthy process that proved to be frustrating at times. For those students who lacked patience and persistence to do work outside of the designated course

time and laboratory meetings, these same students did not believe that they had accomplished the learning objectives of the class. Furthermore, students rated the intellectual stimulation provided by the class to be an average of 4.89 on a scale of 1 to 5, suggesting that the course challenged students to think about new concepts in a creative manner. 78% of the class spent a minimum of four additional hours outside of class and lab time. We posit that the number of hours spent outside of class and laboratory meetings played a critical role in assisting students with understanding game design and ultimately writing code that produced a working 3D game module. The time spent on time also suggests the need for in-depth tutorials that assist students to shortening the learning curve.

Table 1. Student Rankings of the Game Course

Course Criteria	Class Average: Rank on Scale 1 (lowest) to 5 (highest)
Provide an overall rating of the instruction.	4.67
Rate the effectiveness of the instructor in stimulating your interest in the subject.	5.22
Estimate how much you learned in the course.	4.67
Rate the effectiveness of the course in challenging you intellectually.	4.89
Provide an overall rating of the course.	5.11

In summary, students highly ranked the course to be 5.11 on a scale of 1 to 5 with five being the highest and 1 being the lowest. This is extremely high in comparison to traditional computer science courses taught at Northwestern University, which average a rating of 3.

6. Conclusion and Acknowledgments

We offered the Game Authoring course as part of the computer science curriculum at Northwestern University. We purposefully identified effective practices that support the implementation of industry practices in the classroom setting. Furthermore, we establish these practices based student evaluations of the mistakes we made and the things that worked well. These observations serve as the starting point for revising computer science curriculum as we prepare the next generation of computer programmers. We would like to thank Microsoft Research for providing the necessary funding and equipment for making this research possible. Additionally, we express our appreciation to the National Science Foundation for their support of this research.

References

- [1] Adams, E. Bad Game Designer, No Twinkie! *Gamasutra*. (March 13, 1998).
- [2] Adams, E. How to Get Started in the Game Industry Part 1. *Gamasutra* (December 11, 1998).
- [3] Adams, E. How to Get Started in the Game Industry Part 2. *Gamasutra* (December 18, 1998).
- [4] Brown, J.S., Collins, A., and Duguid, P. Situated Cognition and the Culture of Learning. *Educational Researcher*, (18)1, 1989, 32-41.
- [5] Collins, A., Brown, J.S., and Newman, S.E. Cognitive Apprenticeship: Teaching the Crafts of Reading, Writing and Mathematics. In L.B. Resnick (Ed.), *Knowing, learning, and instruction: Essays in honor of Robert Glaser*, Erlbaum, Hillsdale, NJ 1990, 453-494.
- [6] Elliot, J., Adams, L., and Bruckman, A. No Magic Bullet: 3D Video Games in Education. In *Proceedings of the International Conference of Learning Sciences (ICLS 2002)* Seattle, Washington, 2002.
- [7] Fullerton, F., Swain, C., Hoffman, S. *Game Design Workshop: Designing, Prototyping, and Playtesting Games*. CMB Books, San Francisco, CA, 2004.
- [8] Gee, J. *Situated Language and Learning: A Critique of Traditional Schooling*. Routledge, 2004.
- [9] Gee, J. *What Video Games Have to Teach Us about Learning and Literacy*, Palgrave Macmillan, New York, NY, 2003.
- [10] Koster, R. *A Theory of Fun for Game Design*. Paraglyph Press, Scottsdale, AZ, 2005.
- [11] Lave, J. and Wenger, E. *Situated Learning: Legitimate peripheral participation*. Cambridge University Press, Boston, MA, 1991.
- [12] Malone, T. W. What Makes Things Fun to Learn? Heuristics for Designing Instructional Computer Games. ACM (1980).
- [13] Medlock, M. C., Wixon D., Terrano, M., Romero R., Fulton B. (2002). *Using the RITE Method to improve products: a definition and a case study*. Usability Professionals Association, Orlando FL July 2002
- [14] Prensky, M. *Digital Game-Based Learning*. Donnelley and Sons Company, Chicago, IL, 2001.
- [15] Rankin, Y., Gold, R., and Gooch, B. 3D Role-playing Games as Language Learning Tools. In *Conference Proceedings of EUROGRAPHICS Education Program 2006*. Vol. 25. Vienna, Austria, 2006.
- [16] Rankin, Y., Gold, R., and Gooch, B. Evaluating Interactive Gaming as a Language Learning Tool. In *Conference Proceedings SIGGRAPH Educators Program*, Boston, MA, 2006.
- [17] Salen, K. and Zimmerman, E. *Rules of Play: Fundamentals of Game Design*. MIT Press, Boston, MA, 2003.
- [18] Snyder, N. *Universities See a Sharp Drop in Computer Science Majors*, Tennessee.com September 2006.
- [19] Vesgo, J. Interest in CS as a Major Drops Among Incoming Freshman. Computing Research News, May 2005 Vol. 17/No. 3. <http://www.cra.org/CRN/articles/may05/vesgo>