Introduction to Game Design in the Large Classroom

Jim Whitehead
Univ. of California, Santa Cruz
Dept. of Computer Science
1156 High Street, SOE3
+1.831.459.1227
ejw@cs.ucsc.edu

ABSTRACT
We present an experience report on teaching a large enrollment course at the University of California, Santa Cruz providing an introduction to game design. Created to attract non-Computer Science majors seeking to satisfy their campus general education requirements, the course has several interesting qualities. These include a requirement to create a working game (as opposed to just a paper design), use of a specialized form of blogging called gamelogs to reflect on game design, and integration of course assignments with a university library collection of computer games. Experience with using a non-computer-based game design project as a final exam is described. Course contents and student demographic information are also presented.

Categories and Subject Descriptors
K.3.2 [**Computers and Education**]: Computer and Information Science Education—*Computer science education, curriculum;*

General Terms
Design, Human Factors, Languages

Keywords
Computer game design, game education, game curriculum, course design, general education.

1. INTRODUCTION
In the Winter quarter of 2006, the Department of Computer Science at the University of California, Santa Cruz (UCSC) introduced the course Foundations of Interactive Game Design (CS 80K). Explicitly designed to be a large enrollment course, in its first offering 172 students enrolled, while a second offering in Winter 2007 had 212 students. This is perhaps the largest introductory game design course currently being taught in the United States.

Computer games are a relatively new and powerful form of computational media that has become increasingly mainstream in the past few years. As compared to other computational media such as the Web, email, or instant messaging, games are more graphically rich, interactive, and immersive. As a result, it is not surprising to find that computer games are an appealing medium for many freshmen and sophomore students. Students who have played many computer games naturally think about additions they might like to make to games they have played, or new games they might create from scratch. The broad appeal of the level design tools that are available for many games speaks to the interest students have in learning how to harness the expressive capacity of computer games.

Due to this interest in computer games, when the Computer Science department sought to develop a new undergraduate focused computer science course, computer game design emerged as a promising topic. One motivation for offering this new course was increasing overall enrollments in computer science in the face of rapidly declining numbers of computer science majors from 2002-2005, a national trend. The same motivation drove the creation of a similar course, GAM 224 (Introduction to Game Design) at DePaul University [2]. UCSC's Computer Science department has also developed new courses on technology applied to social problems (CS 80J, Technology Targeted at Social Issues, and CS 80S, From Software Innovation to Social Entrepreneurship) that have attracted 91 (80J, Spring'07) and 38 (80S, Fall'07) students in their most recent offerings.

Like many universities, UCSC requires its students to take a general set of undergraduate courses in order to produce students with a broad base of knowledge in addition to their degree specialization. One category of these General Education requirements is Natural Sciences, which is the closest match for computer science. There were two objectives for making Foundations of Interactive Game Design satisfy this general education requirement. We anticipated creating demand for the course among students who needed to satisfy their general education requirements. We also wanted to attract students from many different majors into the course, as well as students who have not yet declared a major. There was some hope that students who took the introductory game design course would find computer science attractive, and hence the course would act as a gateway into the major, especially UCSC's new BS Computer Science: Computer Game Design degree. Each year the course is offered, there are appx. 1-3 students who declare the computer game design degree as a direct consequence of taking the course.

In this paper we present our experience from teaching this new course. Since many computer science departments are struggling with reduced enrollments, we feel Foundations of Interactive Game Design provides a useful model for others considering similar course offerings at this scale. In the remainder of this paper, we begin with an overview of course contents and student demographics. This is followed by a description of critical analysis assignments, including our use of a novel form of blogging called gamelogs, and its integration with a library collection of computer games. Non-computer and computer-based game projects complete the paper.

2. STRUCTURE OF THE COURSE

The course is primarily a lecture-based course, with 1 hour and 10 minute long lectures delivered 3 times a week, on Mondays, Wednesdays, and Fridays, over a 10 week long quarter. There is only one section of the course; all students receive the same lectures. Student grades are determined based on their performance of the following activities (2007 offering):

- Two midterm examinations (34%)
- Critical analysis of games (gamelogs and a multi-game analysis essay, 20%)
- Functioning computer game project and associated project documents (30%)
- Final exam (16%)

On non-exam days, each class begins with an in-class demonstration of a computer game related to that day's topic. The first few classes use games selected by the instructor, then shifts to games presented by students. Students are asked to volunteer to demonstrate a game of their choice, and then the instructor matches the volunteered games to class sessions. There have been more student volunteers than demonstration slots in both course offerings. Students enjoy the opportunity to demonstrate their chosen game, as they are typically fans of the game, and can also show their mastery of the game to the class.

From an instructor perspective, the game demonstrations serve several goals. They provide a visually interesting start to each class, and provide motivation for attending the lectures since the game demonstration is not captured in the podcast for each class. Students arriving late to class are not very disruptive to the demonstration. The game demonstrations also provide increased awareness of a range of games to students in the class. While most students have a broad range of experience playing computer games, the huge number of existing games means that the typical student will not have seen all of the games demonstrated in class. Games demonstrated in class can be used as examples in the lectures, since students can now be expected to know these games. Students also frequently have access to games that the instructor does not, creating a two way learning situation.

There are two primary threads of information delivered in the class: game design principles, and use of the Game Maker tool that most students use to create their computer games.

An outline of the weekly sequence of instruction for the Winter 2008 offering of the course is given below (additional detail at: www.soe.ucsc.edu/classes/cmps080k/Winter07/syllabus.html)

- Course overview, key issues in computer games
- Game rules
- Overview of creating games with Game Maker
- Game elements ontology
- Genre overviews: Shmups, Platformers, RPGs
- Genres and Game Maker, Challenge and Conflict
- Play, Level Design
- Game Maker Advanced Techniques, History of Computer Games
- Narrative, Games and Culture
- Massively Multiplayer Online Games
- Project Demonstrations

In the first offering of the course there were two teaching assistants, far too small a number for the amount of correcting and game project assistance required (TAs are expected to work up to 20hrs/week). The second year had three teaching assistants (one was half time, 10hrs/week, shared with another class) and four undergraduate reader/tutors (up to 10hrs/week). This was much better, since there were effectively 7 people to share the load. However, even with 7 people correcting fell behind towards the end of the quarter, indicating that adding one or two more people would be ideal.

3. STUDENT DEMOGRAPHICS

A breakdown of the students taking the course by sex and class status is presented below.

Table 1 - Student Demographics

	2006			2007		
	Male	**Female**	**TOT**	**Male**	**Female**	**TOT**
Freshman	40 74.1%	14 25.9%	54	72 80.0%	18 20.0%	90
Soph.	50 94.3%	3 5.7%	53	54 85.7%	9 14.3%	63
Junior	29 85.3%	5 14.7%	34	34 87.2%	5 12.8%	39
Senior	26 83.9%	5 16.1%	31	15 75.0%	5 25.0%	20
	146 84.4%	27 15.6%	172	175 82.5%	37 17.5%	212

Several trends are clear from this table. Despite being a general education course, the class still maintains the typical engineering ratio of 5 men to every 1 woman in the class, indicating the course content is much more appealing to men than women. As expected for a general education course, the class is skewed towards freshmen and sophomore students (62% in 2006, 72% in 2007).

Since the UCSC student records system does not track a student's major at the time they took a course, it is not possible to precisely analyze the distribution of student majors at this time. An analysis of student majors of the students at the time of writing of this paper indicates that 22% of 2006 students and 32% of 2007 students are in a School of Engineering major (BA/BS Computer Science, Computer Science: Computer Game Design, Computer Engineering, Bioinformatics, Applied Math and Statistics). DePaul's GAM 224 course has ranged from 40%-85% Computer Science, Telecommunications, and Information Systems (CTI) students, a number that is skewed by a large number of Computer Games Development majors in the class. Still in recent Winter and Spring offerings, GAM 224 has had 50%-60% non-CTI students. The UCSC and DePaul experiences indicate that general education game design courses are effective at attracting non-engineering students.

4. CRITICAL ANALYSIS OF GAMES

One pedagogic goal for the course is the development of student capacity to perform critical analysis of computer game design. Many students entering the course have strong opinions on the quality of games they have played, but often lack a rich vocabulary and intellectual framework for expressing these opinions. The primary approach used by the class to develop these critical analysis skills is multiple assignments that involve writing about games. These come in two forms, writing blog entries about gameplay sessions for individual games (gamelogging), and writing an analysis that performs a compare/contrast of multiple games.

4.1 Gamelogging

Gamelogs are a form of blog (web log) specialized for writing about computer game play sessions[1]. Created by Jose Zagal and Amy Bruckman at Georgia Institute of Technology, they were designed as a technique for making students more reflective about game design [2]. Zagal and Bruckman report gamelogs as a positive learning experience, noting an improved relationship with videogames as a medium, broadening and deepening students' understanding of computer games, and acting as a vehicle for expression, communication, and collaboration. Based on these beneficial qualities, gamelogs were adopted for use in the Winter 2007 offering of CS 80K.

Students were required to perform five gamelog assignments over the quarter. For each assignment, they were required to play a computer game for at least 45 minutes, then write their first gamelog entry. Students then need to play for another 45 minutes, then write a second entry about the same game. The rationale for the second gamelog entry is to encourage deeper exploration of gameplay, since students tend to be more reflective about their second game play session after having written about the first. Players often try different things in their second gameplay session after reflecting on their initial gameplay session. People often confirm hypotheses they have made about their gameplay, or verify observations they have made, or decide on things to try out to have something additional to write about. Additionally, if the first gamelog entry covers superficial qualities of the game, the student clearly needs to go deeper for the second entry.

Our overall impression is the gamelog assignments did help students become more articulate about describing their gameplay sessions. It is difficult to write about a nonlinear interactive experience, and these assignments forced students to engage writing about this kind of experience. There were mixed results in achieving the goal of being more reflective about game design. Perhaps due to the construction of the assignment as writing about a gameplay session, many students just wrote about what happened when they played the game, and did not go further to engage issues of game design in their gamelog posts. Grading feedback on gamelog entries focused on improving design reflection in gamelog entries, with mixed results. Many students did improve the design reflection in their entries based on this feedback, while other students did not. We plan on exploring a combination of rewriting the assignment description along with earlier and more focused feedback to students to address this issue in the next course offering.

One unexpected benefit of the use of gamelogs was an increased sense of community in the class. Since the gamelog website provides a list of the most recent weblogs when you go to the site, it is a kind of activity indicator for the class. Instead of sitting alone in a dorm room performing the class assignments, students get a better sense of being part of a community of students completing the same assignment. This ready access to other student gamelog entries encourages students to read each other's gamelogs. A small number of students also took advantage of the site's ability to leave comments on other student gamelogs.

We were fortunate that Jose Zagal created a specialized grading interface for the gamelogs website, permitting subdivision of grading work among multiple TAs and reader/tutors. This site also made it easy to see whether a student had completed their assignments. Since the site was web-based, correcting could be done online, which was perceived to be more convenient since it could be done at home.

4.2 Library game collection

One issue that arises when assigning students to play computer games for a class assignment is ensuring that students have (legal) access to several games. We also wanted to have all students play a small number of games for some gamelog assignments so they could benefit from reading other student responses to the same game they had played.

To support access to computer games and create a small pool of games used in assignments, we worked with the UCSC Science and Engineering Library to create a small collection of computer games and consoles. Initially, this collection contained three lending consoles (Nintendo Entertainment System (NES), Nintendo 64, and Playstation 2 (PS2)) and fifteen games (called the "classics list")[2], five for each console. Each console and game had five lending copies. Consoles and games were placed in the library's reserve lending category, meaning students could check out games and consoles for three days. Three of the five gamelog assignments required students to play games on the classics list.

There were several challenges in creating the library collection. The Science and Engineering Library was an enthusiastic partner in creating this small collection, but wanted to ensure it was tightly coupled with instructional use. The librarians involved in the project were non-gamers, and so an important aspect of the project was educating them on the myriad combinations of systems, controllers, game media (cartridges, CDs, DVDs, etc.), and game genres. Other issues included storage for games (behind the checkout counter), bags for consoles (we bought many backpacks, one for each lending console), funding (combination of library start up funds from multiple faculty and faculty gift funds), cataloging, and lending periods.

Overall the experience of making games available for checkout was a positive one. Students did check out games and consoles during the quarter, and no consoles were damaged. Students appreciated having access to the games and consoles, especially the Playstation 2 games. One lesson learned was the three day lending window was too small. Several students wanted the ability to check out games and consoles for longer, so they could more fully experience a game in a student's fragmented schedule.

4.3 Multi-game analysis essay

As another approach for students to perform critical analysis of computer games, we assigned a 2-4 page essay in which students were asked to compare/contrast three computer games. The three games were required to either all come from the same genre, or be three members of a series of games, such as the Legend of Zelda

[1] www.gamelog.cl

[2] www.soe.ucsc.edu/classes/cmps080k/Winter07/classics.html

series. Students were asked to reflect on the game design in their chosen games, including issues such as challenge and conflict, how fictional background contributes to gameplay, how game rules contribute to gameplay, strengths and weaknesses in level design, reward structures, and others.

This assignment was successful at causing students to reflect on the game design elements of their chosen games. The essays mostly did engage one or more game design issues, occasionally in quite insightful ways. There were no incidents of plagiarism in this assignment, perhaps due in part to its structure, which does not lend itself to a web-based cut-and-paste approach. Students also generally displayed a strong interest in the games they wrote about, and hence were motivated to perform the assignment. The depth of analysis displayed in many essays may be a reflection of the deep background many students brought to class from having played games for many years.

5. NON-COMPUTER GAME PROJECTS

Building a working computer game typically takes far more time than creating non-computer games, such as board games and card games. As a result, an assignment to create a non-computer game can be a useful part of a course on computer game design, since it permits rapid exploration of the construction of rule sets. CS 80K tried two approaches for including a non-computer game project into the course.

In Winter 2006, student teams were required to turn in a completed non-computer game in the third week of instruction. No restrictions were placed on the kind of game that could be created, so long as the game was substantially original, the rules fit one two typewritten pages, and game play did not involve breaking laws or violating campus regulations (the "don't get your professor in trouble rule"). Students were required to submit their game, as well as an essay describing their experience playing the game. This assignment produced a broad array of games, and did cause students to think about the construction of unambiguous rule sets.

There were a few flaws in the assignment. One was explicitly permitting students to create drinking games. This could be construed as encouraging student drinking, which is more commonly discouraged in the United States. Drinking games tend to have simple rules, and hence they appear to be easy to make. Unfortunately, it is very hard to make a good drinking game, since the game needs to maintain a fine balance between having players play the game, and consuming a beverage. Too much of either makes the game unbalanced. Due to these two factors, there were an unusually high fraction of drinking games submitted for this assignment, and most were clearly bad games. We note that most students opted to not drink alcoholic beverages during their game testing since the assignment was due mid-week. We recommend explicitly disallowing drinking games in non-computer game creation assignments.

One student team complained about the length of the rules permitted for the assignment. Two pages were not long enough to describe the rules for their strategy game, such as rules for movement, combat, and piece characteristics. Many teams created new card games, which were generally of low quality. Some of the card games created were very similar to existing card game variants, and it was unclear whether the students had plagiarized the game or had simply reinvented an obscure variant. Since plagiarism was not generally an issue in class, we believe that inadvertent reinvention is the most likely explanation. However, due to this we also recommend against allowing card game variants.

In Winter 2007, we again had a non-computer game assignment. Somewhat unusually, we used this as a take-home final examination for the course. Since the course requires the construction of a working computer game, a substantial project for many students, we wanted to make the final examination less time consuming to allow students to focus their energy on other courses during finals week. We required each student to create a new game, and bring it to the final exam. During the exam, students were required to play the game of one other student (they were able to choose which game they played), and ensure their game was played. They then wrote up their experience playing the game in the form of a brief essay, which was turned in, along with their game rules.

This worked very well. Though some students complained about having a new creative project assigned to them so late in the quarter, most students were able to create a new game without taking a long time to do so. The final exam was utter chaos (in a good way), as students played each others' games, spreading out into every spare corner of the classroom to do so. One game even involved ricocheting Frisbees off of the outside of the classroom building. Since the exam involved a subjective assessment of how well a game was played, it was possible to ignore most typical exam conventions, and permit talking, collaboration, and open books. It was fun.

While using the non-computer game project as a final exam was overwhelmingly a positive experience, there were some drawbacks. We were unable to develop reasonable assessment criteria, and hence graded students on completion of the assignment only. While this is reasonable given the open-ended nature of the assignment and speeded grading of the finals, having some assessment criteria would likely improve the exam as a learning experience. Additionally, some students felt this assignment would have been more useful early in the quarter.

6. COMPUTER GAME PROJECTS

The capstone project for the course is the development of a working computer game. Students in the class are strongly encouraged to form teams of two students, which most do. The rationale for this is to encourage students to engage in pair programming, and introduce students to the issues of working collaboratively on a piece of software. Working in teams also helped smooth over the differences in technical and artistic abilities among students.

Some students resist forming teams of two students, and strongly want to work alone. In the first offering of the course, students were pushed into teams, even when they wished to work alone. This required substantial administrative oversight by the TAs, and in the second offering students were not required to form groups of two. Groups of three were disallowed unless a compelling rationale was presented, since there was too great a risk of the "free rider" student in such cases, and no effective way to monitor this at large scale.

The primary game development technology taught in class is Game Maker[3], created by Mark Overmars [1]. Game Maker can be viewed as a visual programming language for game

[3] www.yoyogames.com/Game Maker/

construction. Game Maker distinguishes between sprites, which provide the visual details of game elements, and objects, which are typically connected with a sprite, and provide the behavioral aspects of game elements. Behavior is programmed in a event-driven manner, with the game creator specifying what action should occur after, say, a collision event. A step event provides a place for main game loop logic.

Game Maker has multiple pedagogic benefits. In a single session, a beginning user can typically make a game element appear on the screen, move, and make simple responses to user input. The primary interface is windows-based, and so syntax does not interfere with the initial learning of procedural thinking. The object-oriented, event-based programming seems natural to students in this domain, and has not been a substantial stumbling block. For more advanced students, Game Maker provides a built-in scripting language that allows more complex behaviors to be specified. Many third-party extensions have been developed as DLLs, allowing advanced students access to sophisticated physics engines, and lighting engines. As a result, Game Maker provides a nice progression from beginning visual programming environment to more traditional text-based programming experience. Game Maker also exposes sufficient low-level detail such that it is possible to teach about collision detection and how to achieve in-game effects like the jumping mechanic in platformers. Game Maker is also inexpensive, free in the Lite version, and US $20 for the Pro version, cheaper than most textbooks.

Game Maker does have several disadvantages. While it is straightforward to make an initial platform game where most elements work, it is exceedingly hard to make a well-polished platform game. Game Maker is not well suited for the creation of role playing games, since the effort required to create leveling systems, weapon systems, dialog systems, etc. is quite high in the tool. Game Maker also has poor scalability characteristics. Large sound files especially cause initial game load times and memory usage to grow quickly. Game Maker version 6 (the version used in the two class offerings, but now one version behind the current version 7) would occasionally create corrupt save files as games grew large, leading to multiple incidents where students lost their work. Game Maker is also not well suited for collaborative work, as the facilities for merging the work of two developers are awkward, forcing all of the second developers' work into a sub-hierarchy of objects. Game Maker is also Windows-only. Due to the above issues, some students chose to work with other game creation tools. Of these, the most popular was RPG Maker, a tool designed for the creation of computer role playing games.

Project work was organized into a series of deliverables. These were: (1) team name and members; (2) game concept document; (3) work breakdown and schedule; (4) partially operational prototype and progress report; and (5) final project submission. From the first offering of the class, we learned that a portion of the class only really worked on their games in the few days prior to a due date. To reduce the end-of-the-quarter crunch, we added the requirement to submit a working prototype. Exercises developing a schedule and assessing progress towards this schedule encouraged students to think about time management.

To provide assistance for students creating their games, drop-in help sessions with Game Maker were held in the week prior to deadlines. This structure made it possible for the course to avoid a regularly scheduled lab, and be more responsive to when students were ready to receive help.

In Winter 2007, all but two students in the class were able to complete their game project (the two students had stopped doing *any* work in the course). We believe our experience indicates that lack of programming experience should not be a deterrent for assigning students to create working computer games as class projects. As a learning experience, the creation of a working game is much stronger than creating just a paper game design document. Game creation projects engage both experiential and inquiry-based modes of learning, as students form models of their game, enact them, and then iteratively refine their properties based on experience.

All students have a chance to publicly demonstrate their project. In Winter 06, we spent two class sessions with four simultaneous projection screens showing demos. This was too chaotic, as it was hard to properly queue students to projectors, and it was hard for observers to see what was going on. In Winter 07, we split the class into multiple classrooms. Students then demonstrated their game in their assigned classroom, over a two class period. In Winter 07, the six best class games were selected to demonstrate to a panel of external judges from the computer games industry. The top four winning teams received donated prizes. We noticed that the prizes provided substantially increased motivation to produce polished games, as the best games in Winter 07 were substantially better than the best games in Winter 06. Two fairness issues that remain unresolved is the prize bias towards computer science students, and the bias towards Game Maker games and away from RPG Maker games (which do not demo well in a short time span).

7. CONCLUSIONS

Foundations of Interactive Game Design demonstrates that an introduction to game design course can be effectively delivered in a large classroom setting, aimed at a freshman/sophomore audience. Indeed, 81% (2006, 90 of 172 students answered question) and 79% (2007, 75 of 212) of students rated the "course overall as a learning experience" as very good or excellent in student evaluations. Such courses are effective at attracting non-engineering students, but at typical engineering gender ratios. Even with large numbers of students with little or no programming background, it is possible to have a final project in which students create a working computer game. We believe this course can serve as a useful model for other institutions that may wish to create similar courses to reach out to a broad on-campus student audience.

8. ACKNOWLEDGMENTS

We would like to thank Jose Zagal for his invaluable assistance in using GameLogs in CS 80K in Winter 2007.

9. REFERENCES

[1] M. Overmars, "Learning Object-Oriented Design by Creating Games," *IEEE Potentials,* 23(5), 2004, pp. 11-13.

[2] A. Settle, R. Burke, L. Dettori, "Game Design as a Writing Course in the Liberal Arts," In *Proc. FECS 2007: The Int'l Conference on Frontiers in Education: Computer Science and Computer Engineering,* Las Vegas, NV, June, 2007.

[3] J. Zagal, A. Bruckman, "GameLog: Fostering Reflective Gameplay for Learning," *Proc. 2007 ACM SIGGRAPH Symposium on Videogames,* San Diego, CA, pp. 31-38.