Creating a Computer Game Design Course.

Simon McCallum*
University of Otago
P.O. Box 56
Dunedin, New Zealand
simon@cs.otago.ac.nz

Jayson Mackie
University of Otago
P.O. Box 56
Dunedin, New Zealand
jayson@cs.otago.ac.nz

Lennart Nacke
Otto-von-Guericke-University
Magdeburg, Germany
nacke@indyvisuals.com

ABSTRACT
This paper discusses the creation of a computer game design course at the University of Otago. The course was run over a six week intensive period, which exposed students to the stress and working conditions experienced in real industry. We present a postmortem of the course in the style of a game development postmortem.

Categories and Subject Descriptors
K.3.2 [**Computing Milieux**]: COMPUTERS AND EDUCATION—*Curriculum , Computer science education*; K.8.0 [**PERSONAL COMPUTING**]: General—*Games*

General Terms
Computer Game Design, Computer Science Education

Keywords
Computer Game Design, Computer Science Education, Learning by Example

1. BACKGROUND FOR THE COURSE
For many year computer games have been considered a waste of time, and not worthy of academic investigation. Games were a "silly" pastime that took time away from more important areas of research. This sort of intellectual snobbery has delayed the introduction of courses focused on computer game design into traditional universities. There has however, been a seismic shift in the attitude of academics toward computer games.

This shift can be linked to several significant events:

- The value of the industry surpassing that of the Hollywood boxoffice[3]
- Academic summits held in conjunction with the Game Developers Conference(GDC)
- The development of a curriculum framework by the academic committee of the International Game Developers Association (IGDA).[1]
- Establishing of Digital Interactive Games Research Association(DIGRA)
- and the children of the 1970's coming of age and establishing themselves as university academics.

In New Zealand there have been additional events that have also accelerated the acceptance of computer games as an academic discipline:

- The creation of the New Zealand Game Developers Association(NZGDA)
- Sidhe Interactive releasing Stacey Jones Rugby League
- The expertise and innovation shown in the development of "The Lord of the Rings"
- Interest from the government in developing the computer game design industry.

This course, COSC360, was developed in this climate of development and change.

1.1 Initiation
The initial suggestion for a course on computer game design at Otago came from discussions with a student, Tim Nixon, who was the recipient of a "student scholarship" to the GDC, in March 2002. The idea for a games course needed a champion. Someone to drive the idea through the swamp of university regulations and committee meetings. As with the development of a game, a feasibility study had to be conducted and a break even point established. This point was calculated with reference to the number of students, staffing levels and equipment costs. All these numbers had to be predicted for three years into the future, so that long term commitments could be assessed. Three years is a long time in the games industry and required a fair amount of "casting of bones". The break even point for the course was 12 students. Given the interest from students this course was obviously not going to run at a loss.

*Creator of COSC360 Computer Game Design course

NZGDC '04, June 26-29, 2004, Dunedin, New Zealand.

The second, and most important, part of the justification for the course is in terms of academic content. Before the course could be taught the content had to be shown to follow the stated educational objectives of the Department and the University. The IDGA's curriculum framework document provided a solid base, from which we created the proposed course curriculum. Given that the course was part of a computer science course there is an emphasis on the technical aspects of game development. Demonstrating that a game design course would be academically rigorous was relatively easy given the size and complexity of the game development process. Every part of a computer science education could be found in a game project, from networking, through software engineering, to Turing machines. This coverage and the motivation of student to enrol was enough to convince the University that this course would benefit students.

1.2 learning objectives

From the beginning there were several objectives that we felt were essential for a game design course:

- Learning to appreciate the diverse range of skill required for game development.
- familiarity with model game development practice
- understanding group development under pressure.
- recognising the business side of game development.
- gaining transferable skills related to interactive software.

These objective were be accomplished by providing an environment were there was:

- input from other departments, particularly Design.
- the use of industry standard tools (C++, DirectX).
- an industry like environment for student projects.
- a requirement to produce a full game as part of a group.
- pressure in the last week similar to industry "Crunch".
- exposure to the businessmen working in the games industry.
- and introduction to algorithms and design principles needed for interactive software.

1.2.1 interdisciplinary development

Perhaps the most important aspect to modern computer game design is the interdisciplinary nature of game development. Games requires input from:[1]

- Design
- Visual Art
- Computer Graphics, Networking, AI
- Software engineering
- Music, Audio
- Management
- Marketing

In addition to these there are a number of related fields: Physics, English, Psychology, Sociology, Film and Media Studies, Mathematics, Zoology, History and even Physiotherapy. This list may need to be extended depending on the nature of the game being developed. Games companies either have to develop these skills in house or acquire them from outside by contracting specialists or outsourcing part of the development.

For students to understand the industry, they need to be exposed to this range of topics. Communication was identified as one of the major problems when working with people from these diverse disciplines. By exposing students to lecturers and content from outside their "comfort zone" we hoped to improve their ability to discuss ideas with experts from other areas of game development.

Technical arrogance is a problem in many computer science graduates. Many seem to feel that unless you understand their jargon, then you are obviously stupid and your opinion is not important. This type of arrogance is also common in the art and music fields. As part of the development of this course we wanted to encourage students to identify their weaknesses and recognise the difficulty of other areas of game development. Having this introduction to the language used, and the approach taken by other disciplines, hopefully decreases the barriers to forming effective creative and technical teams.

1.2.2 Company environment

To create a company like environment required several conditions. These included: one machine per student with 24 hours access per day, customisable working environment, group members being seated together, and a supervisor in the lab during "working hours". These conditions provided groups with identity, and allowed student to improve there working environment.

Computer games companies identified working in groups as one of the most important skills for modern game developers. Most employers are looking for people who will fit into existing teams. To be employable students need experience working in groups with source control tools, time management tools, group management tools, and communication techniques. By requiring project managers for the teams, and having weekly managerial meeting we were able to provide a great deal realism for the students.

2. POSTMORTEM

A Postmortem has become a standard tool for game developers to analyse the game development process. We now apply the same tool to the development of this course.

[1] these areas are all identified as different parts of game design in the IGDA curriculum framework[1]

Good

In this section we discuss the five best parts of the course.

2.1 Group workload

This course ran as an intensive six week course, which required a group project that was so large than no one student could complete the project by themselves. It also provided enough work that even a group of students would need to work hard to finish a game. This overcame one of the major problems associated with group work at university. Often one student in the group will do all the work. This is a result several factors:

1. Students working toward different marks. The students looking for an A grade tend to pull a heavier workload,
2. Size of projects. Projects have to be small enough so they can be completed by a group of average students in the time allocated. Problems often have to be toy problems and so do not provide realistic challenges. These projects can be completed by a single able student willing to spend some extra time.
3. Organisation vs generation time. The time required to organise people into group meetings is often a significant portion of the time required to complete the assignment.
4. Multiple projects. Each student in a group will have assignments from other courses. There is occasionally a desire to trade, "I will do assignment 2 of X if you do assignment 3 of Y". This is against the rules but it is hard to trace and even harder to stop.
5. Egocentric programming. Some student feel that if they don't code it then it won't work properly. This attitude is still present in the games industry in those companies that feel that if it wasn't built here then it won't be good enough.

For this course students were motivated to create the games rather than by the final grade. The games they created were too large to be created by one programmer. Thus it was essential to identify the abilities of each group member and work together to get the job done. This is a much closer facsimile of real game programming than could be achieved during normal semesters, given normal assignment workloads.

2.2 Organising Groups

As the major assessment for this course was a group assignment, the creation of functional groups was a vital to the success of the course. Random assignment of students was not going to be good enough. We decided that each student should apply for a position in a group. This required students to write a CV to be handed in by the end of the first week of the course. This was extremely effective and resulted in much better groups that would have been possible otherwise.

As part of the CV we requested the name of one other person that the student would like to work with. This allowed us to create teams of four students, from groups of two, who knew each other and were comfortable working together. This gave an excellent balance between self selection and assigned groups. The information from the CVs also allowed us to spread skills among the various groups so that each group had a logical project manager and lead programmer.

We would recommend this process to any academic who needed to create effective groups for software development.

2.3 Commitment and leadership

The students demonstrated enormous commitment to the course. This commitment needed focus and direction. The three authors, the main educators for the course, moved our offices to the front of the lab. Thus we were always on hand to discuss problems and to assist the groups. Not only did we act as educators, we also took the role of producers for the games. This encouraged the groups to work together and created a business like atmosphere. Our continual presence allowed us to assess each individuals work within the groups, and to deal with problems while they were still minor issues.

The student responded to our commitment to the course. We did not expect the students to work harder than we did. In the first week of the course we spend between 60 and 72 hours working. This was designed to provide the sort of motivation leadership that is common among lead designers and game company CEO's. The feedback from the students indicated that this commitment was appreciated and made them want to work harder.

2.4 Interdisciplinary Content

Computer game design is not a subset of computer programming, nor is it wholly part of Design. To create a game a team has to master a wide range of disciplines. One of the successes of this course was the involvement of other departments from around the University. All of the academics that we approached, were interested in how their research interests overlapped with parts of computer game development. The department who gave seminars were:

- Design - teaching one quarter of the course
- English - gave a seminar on narrative
- Film and Media Studies - gave a seminar on meaningful play, and what games are
- Theatre Studies - gave a workshop on understanding motivation and its effect on human motion. This gave insight into how to emote with the body, rather then just the face.
- Marketing - gave a lecture on the importance of marketing in developing a game. The principle that marketing is not the last stage of game development.

One of the major advantages of being part of a traditional university is that we had access to experts in all of these areas who were willing to participate in this new and exciting area. The involvement of these experts was also intended to show the students the range of expertise that could be

applied to game design. Recognising when you need professional help is one of the most difficult parts of game design.

We also included lectures from three business experts, Jon Labrie, Mario Wynands, and Peter Ashford. Mr Wynands is the head of the NZGDA and CEO of Sidhe interactive. Mr Labrie is the the ex CTO of Weta Digital, and has established a company for developing games for mobile phones. Mr Ashford has worked in the Telecommunications industry and for Virtual Spectator, he has great experience with project management for both large and small projects. These guests provided a important link to the real world of game development. Without this link there would have been the danger that what we taught was inconsistent with current best practice.

2.5 Curriculum

The Curriculum framework developed by the IGDA was essential to the development of this course. The framework allowed us to focus on the technical aspects of the curriculum and ground the knowledge in the appropriate context. Every student received a copy of the framework which allowed them to discuss the topics that they wanted to cover.

Selecting a textbook for a course is much like selecting middleware for a game project. The textbook provides content that can be rearranged and built on for the final course. A good textbook supports the curriculum and allows students another perspective on the material. We could not find a text that adequately covered both design and programming. Thus we resorted to two required textbooks, "Andrew Rollings and Ernest Adams on Game design" and "Core Techniques and Algorithms in Game Programming" by Daniel Sanchez-Crespo. The combination of these two books allowed the development of a much stronger course than would have been possible with a single book.

BAD

As is customary for game development postmortems, we now discuss five aspects of the course that caused problems. This type of analysis has become less common in academic institutions as competition for students increases the desire to hide imperfections. This lack of open discussion makes it harder for people outside the institution to assess courses in a fair and objective manner. Hopefully this sort of critical analysis will become more common in courses related to game design.

2.6 Lack of development time

One of the main problems for the course was the lack of development time. We went form the suggestion of a course in March 2003, to teaching in January 2004. This did not give us enough time to learn the skills required in all the areas and develop the curriculum. Our ambitious curriculum required expertise in an enormously diverse range of topics. The course would have been easier to develop if we had limited ourselves to just 2D graphics and removed some of the additional material, such as motion capture.

This lack of time resulted in a couple of major problems.

- Lack of familiarity with tools. We choose to use 3D Game Studio, but did not have time to become familiar enough with this product to provide sufficient support.
- Inconsistencies in material. With late changes to assignment weightings there was an inconsistency with specifications. For example in one handout an assignment was worth 5% and in another it was 10%. This was resolved using a plussage system. The assignment was worth 10% if it was higher than the rest of the student's marks. No-one was disadvantaged by this mistake, but it was a problem that could have been avoided given more time.

This problem could have been alleviated by employing experts in game design to develop the course. This unfortunately was not an option for the Department. Instead we spent time turning experienced educators into game designers, rather than game designers into teachers.

2.7 Middleware

Selecting appropriate middleware is one of the toughest decisions for a game development company. This course was no different. Mr Nacke spend six weeks reviewing different game engines, and given his review we selected 3D Game Studio. This is a good tool, but unfortunately it did not promote the type of learning that we had intended for the course. Each group was given the choice of working in 2D using either DirectX or SDL, or using 3D Game Studio to develop 3D games. Three groups choose 3D games and four choose 2D.

3D Game Studio is a full game engine with an integrated modeller and three levels of interface: graphical, scripting, and C++ dll's. Students spent too much time wrestling with the scripting language and not enough time learning transferable skills. The groups who choose to work in 2D using C++ and Direct X, gained more from the course than the groups who decided to attempt to develop 3D games.

There are two solutions to this problem, remove the middleware tool, or provide a tool that is faster to learn and easier to use. The later might be accomplished by building tools specifically designed for educational purposes rather than straight development. This would however be accompanied with significant cost, and would provide an unrealistic environment.

2.8 Prerequisite

It was decided early in the development of this course that student would need to be competent programmers before they started the course. There was not time to teach both game design and programming. The enter requirement was set at the completion of stage two computer science. This limited the number of design students who could enrol in the course. Given that the major assignment was a group project it would have been possible to have a design student in a group of programmers.

Many of the students in the first year of this course where much more capable than the minimum requirement. The course had two PhD students, a teaching fellow, and several honours graduates. Over half the students enrolled where

graduates, making the ability of the average student in the course much higher than we first expected. This created a dilemma, should raise the level of the course, or leave students bored by lack of content. We erred on the side of increasing the difficulty of the content. This made the course very difficult for students who had only completed the minimum prerequisite.

We expect that the ability of student enrolling in the course will stabilise in the future, and the content of the course will adjust to match.

2.9 Workload

You will have noticed that workload was included as one of the good aspects of the course. It also needs to be included as one of the problems with the course. The speed of the course meant that there was little time for the students to reflect on what they were learning. The students were focused on the output rather than the process. The pressure to complete the games also had a detrimental affect on the creativity within the groups.

It was made clear to all students before the course started that the workload was going to be heavy. This had the effect of decreasing the number of students who enrolled. The workload was heavier than any other course that the students had taken and as some commented was greater than courses worth twice as many points[2].

The only real solution to the workload issue would be to decrease the amount of content and restrict the size of the final project. Both of these solutions would run counter to the theme of creating an environment similar to real business. Following suggestions from students we will have additional tutorials aimed at discussing the process of game creation rather than just focused on output. We will also review the content and find way to decrease the excessive workload.

2.10 Gender Balance

There were no women in our course. This is a major problem. There were two female students enrolled, but both received job offers for the summer period and so decided to withdraw from the course. Given the ratios of game programmers[3] in industry of approximately 3%[2] it is not shocking that none of the 30 students were females. Given that ratio you might expect on average one female programmer. This however is not good enough. We need to discuss with female students why game development might be of interest to them, and how we can address their concerns about the course. The number one concern expressed by women who choose not to do the course was the workload. They all had other commitments and felt that the course would require more work than they could commit.

We have already committed to reviewing content with the intention to bring the workload into line with other course. This will be a delicate balance between reasonable workload and managing expectations.

[2]points represent credits toward a degree. A BSc requires 120 points and this course was worth 6 points. Each point should required 4 hours work per week in summer school

[3]the course was technically focus so game programming is the most appropriate industry figure to use

2.11 Other notable events

There were a couple of additional events in the course that warrant special mention. In the second week of the course we held a group meeting with the whole class. This meeting discussed the features that we had covered and those that had to be dropped during those two weeks. The meeting then decided the direction for the next four weeks. This meeting was run along the lines of a project meeting with input from each department (in our case each group). The intention of this meeting was to demonstrate how a meeting discussing the progress on a game would work. This also helped the students feel part of the course development.

The use of the motion capture lab was a highlight for the course. Each group was allowed 30 minutes in the lab to capture as much data as they could. This data was incorporated into one of the games[4], and another used it as inspiration for their hand crafted animation[5]. Each group had to storyboard the actions they wanted to capture and instruct the body actor on the actions on the day before capture. We

Figure 1: Lennart Nacke in the Physiotherapy Motion capture laboratory

used this exercise to discuss the role of a tools programmer, the use of middleware, and interaction with an artist[6].

3. THE FUTURE

The future of game design at the University of Otago looks promising. There are a large number of academics interested in research related to computer games. There is a strong link between the Design department and the Computer Science department which will form the basis for further developments. Discussions have begun on the possibility for a Bachelor of Applied Science in Computer Game Design. This degree would involve core papers from Design, Computer Science, Marketing and Management. If established

[4]stickmen

[5]commando monkeys

[6]a classically trained actor

this would provide students with enough programming expertise to work as a standard software developer as well as a game developer. Students would also learn enough design to be able to present innovative design solutions for any digital medium.

We have started the process of establishing a games lab with the necessary infrastructure to support postgraduate study as well as internships with business and content creation contracts. This lab will provide the research base to support lecturing at a university level.

In the immediate future COSC360 will be taught in 2005, and will be followed by an advanced course including topics on pixel and vertex shaders, group AI, and networking.

4. CONCLUSIONS

This course was described by students as one of the best courses they had every taken. Mr Wynands was surprised by the quality of games produced, but even more surprised by the reality of the lessons being learnt. The intensity of the course and the commitment of all involved made this one of the hardest and most enjoyable courses we have been involved in.

The process of generating a postmortem has been extremely useful in the identification of problem areas for the course. The process of writing this paper will hopefully improve the quality of the course in the future, and might provide some lessons, both good and bad, for other academics. We also hope that developers in the industry will gain an insight into the process behind developing an academic course and perhaps come forward with ideas to improve courses at there local institutions.

5. ACKNOWLEDGEMENTS

The authors would like to thank the students of COSC360 for making the course an enjoyable experience. Their commitment to the course and to learning was contagious and helped us get through the long hours and late nights.

6. REFERENCES

[1] I. E. Committee. Igda curriculum framework - the study of games and game development. Version 2.3 beta, February 2003.

[2] J. Olsen. 2003 game development salary survey. *Game Developer*, February 2004.

[3] Science and Technology Report. Consol wars. *The Economist*, June 2002.

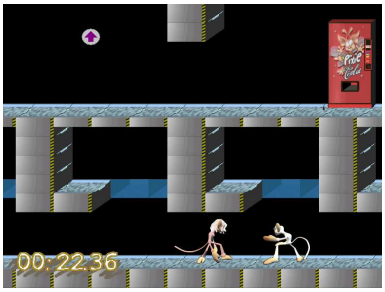
APPENDIX

A. GAMES CREATED

There were seven groups that created games for the course. Every group initially generated a Game Design Document of between 20-40 pages which formed that blueprint for their games. Following is a list of the games that were created, with an associated screenshot.

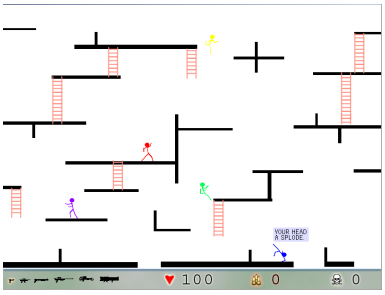
A.1 Commando Monkeys

type	Platform, puzzle game
players	Minimum 2 player
network	Networked TCP/IP
features	Co-operative
	Very complete
	Integrated level builder
	Maya generated graphics
	Complete game



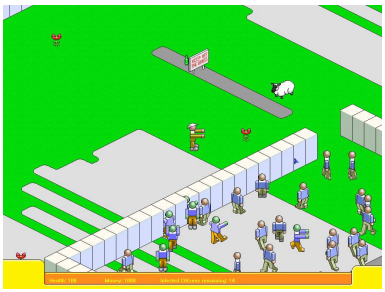
A.2 Stickmen

type	Platform action game
players	Multiplayer max 4
network	Networked TCP/IP
features	Skill and action
	Cartoon death
	Very intuitive
	Easily understood
	Instantly fun
	Motion captured stickmen not actually overkill as the motion helps with engagement.



A.3 Pharmacon

type	tile based Isometric 3D action game
players	single
network	none
features	Heal the sick
	Action game
	Pixelated graphics
	Game with large scope
	Too ambitious
	Some issues with engine need to be tidied up
	Level editor not complete
	Fun minigame



A.4 Waldrick

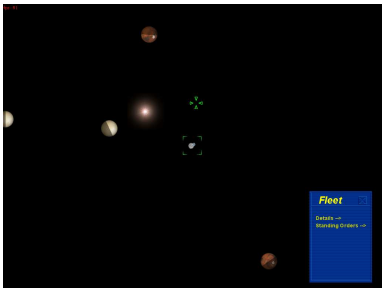
type	3D adventure RPG
players	one
network	none
features	You are a fish
	Gain experience
	Turn based fighting
	Well developed story of a lost fish
	Cute graphics
	Interesting combo fighting mechanism

A.5 Haggard Wars

type	Third person 3D Turn Based Strategy
players	two
network	none
features	Alien vs human Unit selection and upgrading with fixed cost Many unit types Story based game 4 levels Guns and cartoon blood splatter, units disappear when dead. Plays like a pen and paper war game

A.6 Revolutions

type	2D space Real Time Stragety (RTS)
players	two
network	TCP/IP
features	Playing field constantly changing Resource management Timing of travel essential Whole fleet combat model Networking not finished Only achieved a tech demo but easily extended to full game



A.7 Trash Wars

type	FPS and TBS
players	one
network	none
features	Controlling garbage collection areas Backstory developed Game still a bit raw Atmosphere well developed Two stage gameplay, global turn based, combined with missions that are 3D action.