



Interactive Media and Game Development

Frontiers 2009

Mark Claypool



What Do You Think Goes Into Developing Games?

- Choose a game you're familiar with
- Assume you are inspired (or forced or paid) to re-engineer the game
- Take 1-2 minutes to write a list of the tasks required
 - Chronological or hierarchical, as you wish
- Trade write-ups with another student
- What do we have?



Outline

- Background (next)
- Tutorial 1
- What is a Game?
- Genres
- Tutorial 2
- The Game Industry
- Game Timeline
- Team Sizes



Professor Background (Who am I?)

- Dr. Mark Claypool (professor, "Mark")
 - Computer Science
 - Interactive Media and Game Development
- Research interests
 - Networks
 - Audio and Video over Internet
 - Network games
- I'm playing
 - *Fable II*, *deBlob*, various Itouch games, *FIFA'09*
 - Looking forward to the next *Uncharted!*
 - *Uncharted 2: Among Thieves*





Teaching Assistant Background

- Dan Tennant
- WPI Alumnus ('09), IMGD Major
- Artistic concentration, Writer
- What else...?



Student Background (Who Are You?)

- Year?
 - Junior, Senior, ...
- Interest?
 - Art or Programming or ...
- Computer Programming?
 - (what's a program?) 1 to 5 (hacker!)
- Gamer?
 - (casual) 1 to 5 (hard-core!)
- Built any games?
- What are you playing right now?
 - What type of game is it? Why is it fun?
- Other ...





Course Materials

- Facebook group:
 - <http://tinyurl.com/imgd-frontiers-09>
- Will have:
 - Slides (ppt and pdf)
 - Links to tutorials, etc.
 - Discussion groups
- Need permission to join
 - Dan and I are both admins



Overall Course Structure

- 8:30-10:30
 - Technical/Design aspects of IMGD
 - Several 3D games,
- 10:30-12:30
 - Communication Workshops
- 1:30-3:30
 - Artistic/Design aspects of IMGD
 - 3D game, Unreal Tournament Mod
- 3:30-4:30
 - Lab - work on either morning or afternoon stuff on your own





Technical Course Structure (1 of 2)

- Start around 8:30
 - Me: lecture + discussion for 15-30 minutes
 - You: work for 30-60 minutes
 - Repeat
- Probably more of me talking the first few days, more of you working last few
- During work, Dan & I circulate around for help



Technical Course Structure (2 of 2)

- Topics
 - Game Programming
 - No previous programming experience required!
 - Game Design
 - What is a game, what makes it fun, how to design
 - Game Art
 - Not make your own, but principles
- Use game development environment... *Kodu*
 - Make games using only your Xbox controller!
 - (More shortly)





Rough Timeline

- Days 1-4
 - Aspects of game development, several small games
- End of day 4
 - Idea for your own, final game
- Day 5-7
 - Work on game (or games!), including play testing
- Day 8
 - Game fest!
- Day 9
 - Field trip
- Day 10
 - Closing ceremonies



Outline

- Background (done)
- Tutorial 1 (next)
- What is a Game?
- Tutorial 2
- Genres
- Tutorial 3
- The Game Industry
- Game Timeline
- Team Sizes





Tutorial 1 - Kodu Basics

- The Main screen
- Within a Level
 - Tool palette
 - Selecting objects
 - Programming and Tweaking
 - Terrain
 - ...



Assignment 1 - Combat

- Two-player combat
- Make basic world
 - From scratch or modify tutorial level
- Add one object
 - Move with stick, fire with buttons
 - Blips and Missiles
- Repeat for player 2
- Play!
- Even more:
 - Can you make powerups? Shields? Limited ammo?
 - What are the differences in the objects?
 - What can you do to make it have more long-term appeal?





Outline

- Background (done)
- Tutorial 1 (done)
- What is a Game? (next)
- Genres
- Tutorial 2
- The Game Industry
- Game Timeline
- Team Sizes



What is a Game? (1 of 3)

- Movie? (ask: why not?)
 - no *interaction*, outcome fixed
- Toy? (has interaction ... ask: why not?)
 - no *goal*, but still fun (players can develop own goals)
- Puzzle? (has goal + interaction ... ask: why not?)
 - strategy and outcome is the *same* each time

"A computer game is a software program in which one or more players make decisions through the control of game objects and resources, in pursuit of a goal."



What is a Game (2 of 3)

- A Computer Game is a *Software Program*
 - Not a board game or sports
 - Consider: Chess vs. Soccer vs. Warcraft
 - Ask: What do you lose? What do you gain?
 - Lose: 1) *physical pieces*, 2) *social interaction*
 - Gain: 1) *real-time*, 2) *more immersive*, 3) *more complexity*
- A Computer Game involves *Players*
 - "No, Duh". But stress because *think* about audience. The game is not for *you* but for *them*.
 - Don't just think about your story or the graphics or the interface, but consider the *players*.
 - Ex: complicated flight simulator (say, you are a flying geek) but audience is beginner



What is a Game (3 of 3)

- Playing a Game is About *Making Decisions*
 - Ex: what weapon to use, what resource to build
 - Can be frustrating if decision does not matter
 - Want good *gameplay* (next major topic)
- Playing a Game is About *Control*
 - Player wants to impact outcome
 - Uncontrolled sequences can still happen, but should be sparing and make logical
 - Ex: *Riven* uses train system between worlds
- A Game Needs a *Goal*
 - Ex: Defeat Ganandorf in Zelda
 - Long games may have sub-goals
 - Ex: recover Triforce first, then Sword of Power
 - Without game goals, a player develops his/her own (a toy)





What a Game is Not (1 of 2)

- *A bunch of cool features*
 - Necessary, but not sufficient
 - May even detract, if not careful, by concentrating on features not game
- *A lot of fancy graphics*
 - Games need graphics just as hit movie needs special effects ... but neither will save weak idea
 - Again, may detract
 - Game must work without fancy graphics
 - Suggestion: should be fun with simple objects

"When a designer is asked how his game is going to make a difference, I hope [s/]he ... talks about gameplay, fun and creativity - as opposed to an answer that simply focuses on how good it looks"

- Sid Meier (*Civilizations, Railroad Tycoon, Pirates*)



What a Game is Not (2 of 2)

- *A series of puzzles*
 - Most games have them
 - But not gameplay in themselves
 - Puzzles are specific, game systems spawn more generic problems
- *An intriguing story*
 - Good story encourages immersion
 - But will mean nothing without gameplay
 - Example: Baldur's Gate, linear story. Going wrong way gets you killed. But not interactive. Interaction in world all leads to same end.





Games are Not Everything

- Most important ... *is it fun, compelling, engaging?*
 - And these come from a superset of games
- Computers are good at interactivity
 - Allow for interactive fun
 - *Interactive Media and Game Development* ☺



Discussion

- What are some examples of interactivity that is fun but not a game?
 - *SimCity* - very compelling, but mostly no goals. More of toy than a game, but still fun.
- What are some examples of fun graphics and video that are not interactive?
 - *Uncharted* - good visuals, story, etc. But need to play the game to proceed. Could have skipped to just watch story. Would still have been *fun* without the gameplay.





Outline

- Background (done)
- Tutorial 1 (done)
- What is a Game? (done)
- Tutorial 2 (next)
- Genres
- Tutorial 3
- The Game Industry
- Game Timeline
- Team Sizes



Tutorial 2 - Levels in Kodu

- Tutorials - games that teach you something about Kodu programming
- Techniques - simple worlds to illustrate one behavior (tricks, methods to do something)
- Sample Games - mini games, see what Kodu can do. All fully editable!





Assignment 2 - Breadth of Kodu

- Try out all the sample games!
 - Play for fun
 - Keep mind open for exposure of ideas
 - Consider genres (types of games)
- Keep in mind the tutorial and techniques for when you need them



Outline

- Background (done)
- Tutorial 1 (done)
- What is a Game? (done)
- Tutorial 2 (done)
- Genres (next)
- Tutorial 3
- The Game Industry
- Game Timeline
- Team Sizes





Game Types/Genres

- What are some types of games?
 - Provide examples
- What separates them from others?



Arcade Games

- Reaction speed are the most important aspect of the game
 - Examples: scrolling shooters, maze games like *Pacman*, paddle games like *Breakout*, *Pong*
- Relatively easy to make
- Normally 2-d graphics
- Good starting point for first game





Puzzle Games

- Clever thinking is the most important aspect
- Ex: Many maze games are actually more based on puzzle solving rather than on reaction speed
- Other examples include board games and sliding puzzles
- Normally 2-dimensional
- Relatively easy to create
 - Except when played against a computer opponent
 - Artificial Intelligence can be harder
 - Ex: How to program the computer to play chess?



Role Playing Games

- Steer a character through a difficult world
 - Examples are *Diablo* and *Baldur's Gate*
- Development of character to learn new skills, becoming more powerful, and finding additional and better weapons
- Opponents become more powerful as well
- Can create 2-d or 3-d
- Generally harder to make because must create the mechanism of character development
- Also normally need large world
- Good level design is crucial





Strategy Games

- Real-time (RTS) or turn-based
- Player only indirectly controls the character
 - Tactics less important than Strategy
- Examples include *Age of Empires*, *Warcraft III*...
 - Also, usually "God Games", such as *B&W*
- Generally take a lot of time to create
 - Require many different game objects, each with animated images and specific behavior



Adventure Games

- Game is about adventure and exploration
 - Story line is rather crucial
- Can be 2-d or 3-d
- Actions easy (just move)
- Difficulty is in making exploration/adventure interesting
 - Interesting, funny, and surprising story line
 - Corresponding artwork
- Artists role crucial





First-Person Shooters

- 3-d version of many arcade-style games (move and shoot)
- Emphasis is on fast-paced action and reaction speed, not on cleverness and puzzle solving
- Many examples: *Doom*, *Quake*, ...
- Need to be 3-d
- Relatively difficult to create because of models



Third-Person Action

- Player directly controls a game character (avatar) through a hostile world
 - Ex: *Tomb Raider*
- Not much emphasis on character development
- Fast action and discovering the game world
- Some have story line, other adventure game aspects
- Can be 2-d or 3-d
- Can sometimes be created easily





Sports Games

- Real-life sport, made virtual
- Ideas, rules in place
- Making realistic, challenging, fun like sport can be difficult



Racing Games

- Drive a vehicle, as fast as possible or sometimes for exploration or combat
- Special type of sport game
- Either realistic (ex: *Formula 1*) or focused on fun aspects (*Midtown Madness*)
- Both 2-d or 3-d





Party Games

- Variety of types
 - Ex: Mario Party, DDR, Karaoke
- Social aspects important with participants in the same space
- Allow for rapid change of turns
- Allow for disparate abilities (beginners and experts, both have fun)



Simulators

- Try for realistic representation
 - Ex: flight simulators
- Other simulations include world simulation
 - Ex: *simCity* or *simEarth*
- Relatively difficult to create since getting details right a challenge





Educational Games

- Games are great at teaching ... how to play the game!
- Educational games are designed to teach player knowledge or skill that is valuable outside the game
 - Ex: math, reading, problem solving



Kodu and Frontiers

- Can make many genres, but best for:
 - Arcade, Puzzle, Racing ...
 - Small games (but cool!)
- More difficult:
 - Large world exploration
 - Leveling up, inventory,
 - Lots of AI
- Given time:
 - Think small, but expand if time
 - Few levels (show core idea)
 - Have *playable* game early
- Discuss/share ideas on Kodu Forum!



Outline

- Background (done)
- Tutorial 1 (done)
- What is a *Game*? (done)
- Tutorial 2 (done)
- Genres (done)
- Tutorial 3 (next)
- The Game Industry
- Game Timeline
- Team Sizes



Tutorial 3

- TBD





The Game Industry

- 60% of all Americans play video games
 - 35% of Americans rated playing computer and video games as *the most fun entertainment activity* for the third consecutive year
 - Over 50% for young people
- Computer/video game industry on par with box office sales of the movie industry
 - \$7-8 Billion/year for U.S. Sales
- Development
 - Costs \$3M to \$10M to develop average game
 - Takes 12-24 months



Game Studios - Vertical Structure

- Developers
- Publishers
- (Distributors)
- Retailers

- Much like a mini-Hollywood





Developers

- What are some game development studios you know?



Developers

- *Design and implement games*
 - Including: programming, art, sound effects, and music
 - Historically, small groups
 - Analogous to book authors
- Structure varies
 - May exist as part of a Publisher
 - May be "full-service" developers or may outsource some
 - Motion Capture (to replicate realistic movement)
 - Art and Animation (can be done by art house/studio)
- Many started on PC games (console development harder to break into)
- Typically work for royalties & funded by advances
 - Do not have the capital, distribution channels, or marketing resources to publish their games
 - May seem that developers don't get fair share of profits
 - Can be unstable



Publishers

- What are some game publishing companies you know?



Publishers

- *Fund development of games*
 - Including: manufacturing, marketing/PR, distribution, and customer support
- Publishers assume most of the risk, but they also take most of the profits
- Relationship to developers
 - Star Developers can often "bully" Publishers, because publishers are desperate for content
 - Most Developers are at the mercy of the almighty Publisher
- Originally grew out of developers
- Massive consolidation in recent years
- Most also develop games in-house



Retailers

- What are some game retailers you know?



Retailers

- *Sell software*
- Started with mail-order and computer specialty stores
- Shift in 80's to game specialty stores, especially chains (today about 25%)
 - *EB Games, GameStop*
- Shift in 90's to mass market retailers (Today 70%) (ask)
 - *Target, WalMart, Best Buy*
- Retailers generally earn 30% margin on a \$50 game
- Electronic download of games via Internet still in infancy
 - Big but not huge (today about 5%)





Outline

- Background (done)
- What is a Game? (done)
- Genres (done)
- The Game Industry (done)
- Game Timeline (next)
- Team Sizes



Game Development Timeline (1 of 5)

- Inspiration
 - getting the global idea of the game
 - duration: 1 month (for a professional game)
 - result: treatment document, decision to continue
- Conceptualization
 - preparing the "complete" design of the game
 - duration: 3 months
 - result: complete design document
 - (continued next slide)



Concept

Define Game Concept

Define Core Game Features

Find/Assign Developer

Estimate Budget & Due Date



Based on notes from Neal Robison, ATI



Concept: Van Helsing (1 of 4)

Gameplay:
Still firing after being hit



Based on notes from Neal Robison, ATI



Concept: Van Helsing (2 of 4)



Based on notes from Neal Robison, ATI



Concept: Van Helsing (3 of 4)

(Van Helsing
Pre-Production)

Based on notes from Neal Robison, ATI





Concept: Van Helsing (4 of 4)

(Van Helsing
Finished
Concept)

Based on notes from Neal Robison, ATI



Game Development Timeline (2 of 5)

- Prototypes
 - Build prototypes as proof of concept
 - Can take 2-3 months (or more)
 - Typically done a few months in
 - In particular to test game play
 - Throw them away afterwards
 - Pitch to Publisher
- (Continued next slide)



Prototype or 1st Playable

GDD & TDD = "The Bibles"

Production Budget & Detailed Schedule

Submit Concept to Sony, etc.

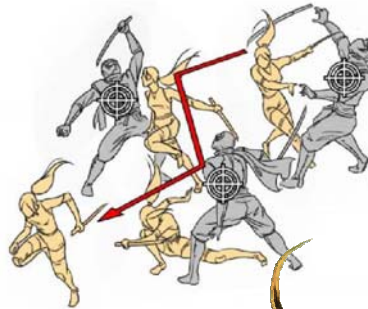
Working Prototype, with Game Mechanics

Focus Test



Based on notes from Neal Robison, ATI

Prototype: Red Ninja (1 of 3)



Based on notes from Neal Robison, ATI



Prototype: Red Ninja (2 of 3)

(Red Ninja
Pre-
Production)

Based on notes from Neal Robison, ATI



Prototype: Red Ninja (3 of 3)

(Red Ninja
Final
Production)

Based on notes from Neal Robison, ATI





Game Development Timeline (3 of 5)

- Blueprint
 - separate the project into different tiers
 - duration: 2 months
 - result: several mini-specification
- Architecture
 - creating a technical design that specifies tools and technology used
 - duration: 2 months
 - result: full technical specification



Game Development Timeline (4 of 5)

- Tool building
 - create a number of (preferably reusable) tools, like 3D graphics engine, level builder, or unit builder
 - duration: 4 months
 - result: set of functionally tools (maybe not yet feature complete)
- Assembly
 - create the game based on the design document using the tools; update design document and tools as required (consulting the lead designer)
 - duration: 12 months
 - result: the complete game software and toolset



Other Development Milestones: Alpha Definition

- At Alpha stage, a game should:
 - Have all of the required features of the design implemented, but not necessarily working correctly
 - Be tested thoroughly by QA to eliminate any critical gameplay flaws
 - Still likely contain a certain amount of placeholder assets
 - (Continued next slide)



Alpha Definition

- Feature Complete
- "Localization" Begins
- Focus Test
- Play Testing
- Marketing Continues



Based on notes from Neal Robison, ATI

Alpha: Crash Bandicoot (1 of 2)



Based on notes from Neal Robison, ATI



Alpha: Crash Bandicoot (2 of 2)

(Crash
Bandicoot)





Game Development Timeline (5 of 5)

- Level design
 - create the levels for the game
 - duration: 4 months
 - result: finished game with all levels, in-game tutorials, manuals
- Review
 - testing the code, the gameplay, and the levels
 - duration: 3 months (partially overlapping level design)
 - result: the gold master



Other Development Milestones: Beta Definition

- At Beta stage, a game should:
 - Have all content complete
 - Be tested thoroughly for bugs and gameplay tweaks
 - Be shown to press for preview features
 - (Continued next slide)





Stages of Development: Beta

- Polish, Polish, Polish
- Game Balancing
- Localization Continues
- Demo Versions



For Display Only



Based on notes from Neal Robison, ATI



Other Development Milestones: Gold Master Definition

- At Gold Master stage, a game should:
 - Be sent to the platform holder/s (where applicable) for TRC testing
 - Be sent to press for review
 - Be sent to duplication for production
 - Be backed up and stored
 - (Continued next slide)



Final/GMC/Gold

- [The Game is "Done"
- [Testing, Testing, Testing
- [Intense Pressure
- [Submit to Console developers
- [Manufacturing Timing



Based on notes from Neal Robison, ATI



Post-Mortem

- [Analysis of PR, Marketing
- [Analysis of Production, Source Code
- [Archive All Assets
- [What went **right**, what went **wrong**
- [Kick-off the Sequel!



Based on notes from Neal Robison, ATI



Outline

- Background
- What is a Game?
- Genres
- The Game Industry
- Game Timeline
- Team Sizes (next)



Development Team Size

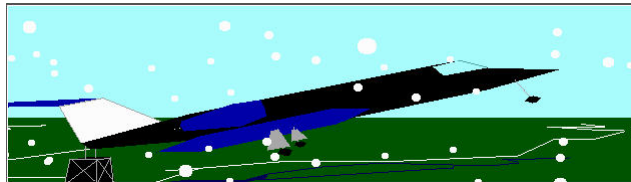
- As late as the mid-80's teams as small as one person.
- Today, teams today ranging from 10-60 people.
- Programming now a proportionally smaller part of any project
- Artistic content creation proportionally larger
- See *Gamasutra*, (www.gamasutra.com)
 - Search for "post mortem"
 - Game data at bottom includes team size and composition



Development Team 1988

- Sublogic's *JET* (early flight sim)
 - Sublogic later made scenery files for Microsoft flight simulator
- 3 Programmers
- 1 Part-Time Artist
- 1 Tester

Total: 5



Laird and Jamin, EECS 494, Umich, Fall 2003



Development Team 1995

- Interplay's *Descent*
 - Used 3d polygon engine, not 2d sprites
- 6 Programmers
- 1 Artist
- 2 Level Designers
- 1 Sound Designer
- Off-site Musicians

Total: 11



Laird and Jamin, EECS 494, Umich, Fall 2003



Development Team 2002

- THQ's *AlterEcho*
- 1 Executive Producer
- 1 Producer
- 4 Programmers
- 2 Game Designers
- 1 Writer
- 3 Level Designers
- 3 Character Modelers and Animators
- 1 2d and Texture Artist
- 1 Audio Designer
- 1 Cinematic Animator
- 1 QA Lead and Testers

Total: 19+



Laird and Jamin, EECS 494, Umich, Fall 2003

Development Teams for Online Games

- Star Wars online (2003?)
- Development team: 44 people
 - 50% Artists
 - 25% Designers
 - 25% Programmers
- 3 Producers
- "Live" Team (starting at Beta, 6 months before done)
 - 8 Developers
 - 50-60 Customer support (for 200K users)
 - 1000 Volunteer staff (for 200K users)

Laird and Jamin, EECS 494, Umich, Fall 2003





A (Larger) Developer Company Today

- Designing and creating computer games is serious business
 - Large budgets (\$1 million+)
 - Large number of people involved
 - Large risk
- Wisdom
 - Use modern software development techniques
 - Keep creativity where it belongs
 - In the design
 - Not during the programming

