Computer Science 113 / Informatics 125
Computer Game Development

Design Document Structure

Introduction

The Design Document is a critical factor in determining the success of your efforts to create a computer game. With commercial games, it is often created over a period of many months, and is a "living" document, updated as the game is developed. For this course, your Design Document will probably describe a larger and more complex game than you will be able to implement by finals week. That's perfectly OK, but you should indicate clearly what parts you plan to accomplish.

Your Design Document will be a web document. You should use HTML, PDF, or another universally readable format.

Your Design Document should have four major components: the Overview, the Game Specification, the Technical Specification, and the Schedule. In addition, an essential component of your document is careful proofreading. Look for misspelled wrods, awkward grammer, needless repetitions, unstated assumptions, vague assertions, needless repetitions, and omissions important features. Every member of the team should have carefully read the document before it is considered final.

Overview (Executive Summary)

The goal of this section is to give the reader, in no more than two pages, a clear understanding of what your game is all about. This means not only a brief overview of the background story, but also a description of game play, important rules, art design, and the technical platform. A person who reads just this section should be able to form an accurate mental picture of the game.

Game Specification

The Game Spec section serves as a "rule book" for the game. It describes the types of interactions the player can have with the game. It should be fairly easy for a writer to turn the Game Specs into a manual that could be included with the game. The nature of the Game Specs is somewhat dependent on the type of game being developed. I'll give examples and descriptions for four genres of games, top-down shooters (TDS), driving sims (DS), first-person shooters (FPS), and role-playing games (RPG).

- **Rules and mechanics.** This is often an extensive section, and may need to be divided into subsections. For any type of game, explain how the game ends or is finished. **TDS:** How ships/tanks/units move, the types of weapons they have, how damage or hit points are calculated. **DS:** The physics of the world, how the car / vehicle is controlled, race rules, information about brakes, tracks, surfaces, parts, etc. **FPS:** How damage or hit points are calculated, how the protagonist moves, types of weapons and their capabilities. **RPG:** Rules for creating characters, their stats, skills, special powers. How inventory, buying/selling works.
- **Artwork and user interface.** For all types of games, show screen shots, sketches, prototypes, samples. Describe the overall "look and feel" of the game. **FPS:** Often tries to "shoot" for the sense of having no user interface. But there always is one, right?
- **Gameplay and balance.** What can the player(s) do? See? Hear? How does he or she interact

with the game world? How are weapons or other tools acquired, used, lost? What are the puzzles that have to be solved? Does left click mean something different than right click? What kind of physics model, if any, is being used? Describe how the game is designed so that whatever options the user chooses, the game is still interesting and playable.

- **Music and sound.** Even the simplest music and most rudimentary sound effects add immensely to the pleasure and "immersiveness" of playing a game. It may be hard to describe the music and sound you plan to have, but you should describe where the music and sound will come from. List important sound effects.
- **Background story. DS:** Often sketchy or non-existent. **TDS, FPS:** Typically fairly simple. **RPG:** Usually quite complex, with many side-stories. Should integrate closely with the artwork, the characters, and the levels.
- **Characters.** This section might be integrated with examples and sketches of artwork. Discuss both the characters the player can play, if any, and the NPCs the player will encounter. **TDS:** Spaceships, asteroids. **DS:** Other racers, pace car, police. **FPS:** Usually a variety of monsters, aliens, and bad guys. **RPG:** Every NPC should be described, his/her/its background, skills, powers, possessions. useful knowledge. For important characters, this should be a "character bible."
- **Levels.** These are the phases of the games. They have a variety of names: **TDS:** Levels. **DS:** Locations, tracks. **FPS:** Missions. **RPG:** Quests. Describe the story, goals, interactions, etc. of several levels. If your levels have puzzles or secrets, describe those. For some games, a storyboard is an extremely useful way of communicating the action in a level. Briefly describe several levels, if appropriate, and give a large amount of detail about a single one you plan to implement this quarter.
- **Scripts.** There are several kinds of scripts: lines for voice actors to speak, text for text-based interfaces, and programming scripts for controlling the action and behavior of some objects or characters. (If you have all three of these, you may want to break them out into separate sections.) **TDS:** Typically little. **DS:** Some, e.g. bridge and traffic light controls. Before and after race sequences. **FPS:** Interactions with monsters, guards, minor non-adversarial characters. **RPG:** Extensive. Ideally, the exact text of dialogs will be presented in the Design Document.
- **Cut scenes.** If you plan to have non-interactive sections of the game, at the beginning, the middle, or the end, describe these in detail.
- **Artificial Intelligence.** How do NPC's, enemies, and followers behave?

Technical Specs

In "real life" the Technical Specs are hundreds of pages long. They take all the rules from the Game Specs and make them specific, unambiguous, and implementable. When writing the Technical Specs, think through implications, exceptions, exact limits. For instance, if the the Game Specs say "The player will be able to make the car move faster by pressing the up arrow key," the Technical Specs should elaborate: "Each press of the up arrow key causes the car's speed to be multiplied by 1.01, up to a maximum of 500 and down to a minimum of -100 (reverse)." For this example, you'd need to explain also exactly how the user shifts between forward and reverse.

The Technical Specs should also discuss or include:

- Programming languages, including compiler and emulator.
- Libraries that will be used.
- Any code or engines to be used. If you plan to use a game engine, describe it and explain what its interface (to the rest of the system) is.

- Target hardware and Operating System.
- Data Structures, or better, interfaces of ADTs and classes. A list of all (C++/Java/C#) classes and their interfaces is appropriate in this section.
- Exact algorithms. If an NPC "knows" how to walk back and forth between to positions, avoiding a large rock in between, discuss the AI algorithm (or script) that will be used.
- Back-up and version control plans. You don't want to lose 40 hours of work in tenth week!

Schedule and Personnel

It's important that each member of the team have a clear set of responsibilities, and a clear time-line for various tasks. You should include the following:

- Deliverables as of the end of seventh week, Friday, November 11.
- Deliverables as of the middle of tenth week, Wednesday, Nov. 20.
- Deliverables as of the end, period, Wednesday, December 7.

As far as possible, state the deliverables by team member. Structure the sequence of your deliverables so that core art and gameplay comes first, and extra features, levels, characters, puzzles, AI, etc., are scheduled later. You should plan on having a playable mini-version of your game by November 18.

For each team member, state the role(s) he or she will play in the project, and what contributions he or she will make to the final game. Also include a statement from each team member about what he or she needs to learn in the remainder of the course.

If you are working with people who are not enrolled in this course, you should mention them and explain their roles. In particular, if you are working with LCAD artists and include any of their artwork, make sure to give them credit.