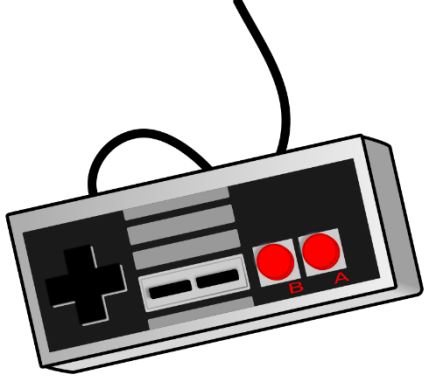


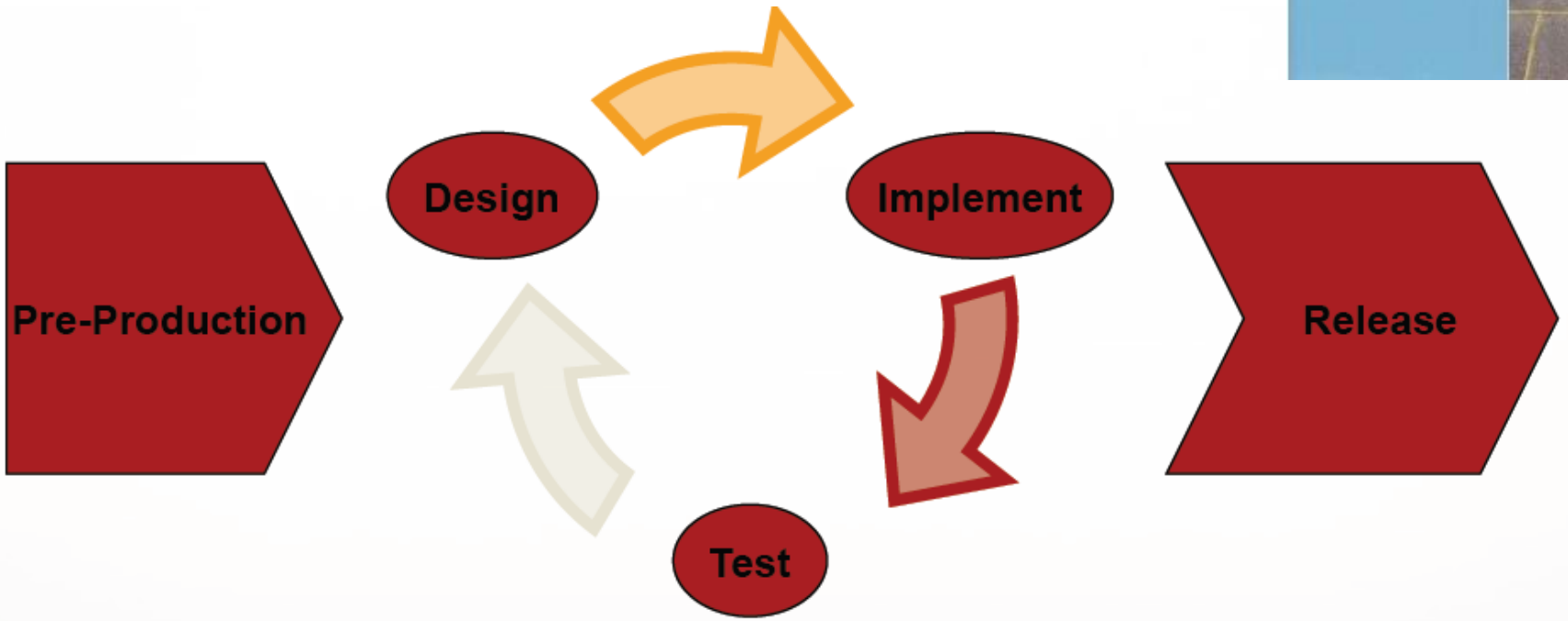
Game Design Document

420-141-VA - GAME PROGRAMMING 1 - VANIER COLLEGE

Objectives

- Game Design Document
- Game Design Document Examples
- Game Physics
- Programming Topics

Early Playable Game



- *Rules of Play*: should be playing no later than 20% into development!

The Game Design Document

It describes the team's shared vision for all aspects of the game development process.

It is the team's main working document, is **continually revised** based on feedback and experience/playtesting.

Writing starts before game development, but it evolves and finishes only with the complete game.

In the industry this is over 100s of pages. But for your team project, about 25-40 pages is OK (short, so augment with regular team meetings – keep minutes).

A working GDD is to be submitted in Week 9

Design Document Layout

1. Title page
2. Executive summary
3. Overview
4. Related games
5. Player composites
6. World
7. Characters
8. Progression graph
9. Art direction
10. UI storyboards
11. Tags and dialogue
12. Technology plan
13. Software architecture
14. Controls
15. Level design
16. Mechanics analysis
17. Schedule
18. Budget
19. Change log

Title Page

- Like the frontispiece for a book:
  - A picture (screenshot or box art)
  - Game title and tagline (optional)
  - Developer name(s)
  - Date and revision number of this design document.

Executive Summary

- Describes the entire game in three sentences:
  - Basic setting
  - What makes the game interesting
  - Some selling point to convince the publisher to fund the game development.

Overview (2-3 pages)

- Follows the proposal worksheet
- It must pitch the core concept, describe major qualities of this game, relate to other games and overall, provide the reader with a mental image of the game.

Related Games

- Reviews related games to be able to draw upon their design, unique selling points, success, failure, etc. Check metacritic.com for objective reviews from the community.
- More games than listed in overview and about one page per game.

Player Composites

- Marketing model of the players
- Helps create intuition for design decisions and to recruit play testers.
- Most games have primary and secondary markets. You will need the demographics for both.
- Create a profile like:

"John Brooke, 27, accountant. Single. Graduate of Loyola College. Plays games alone about once a week, and with male friends on a next-generation console plugged into an HDTV in his living room on weekend afternoons. Focuses on competitive, action games like Gears of War and Madden. Watches football one day a week. Favorite TV shows are Lost and Sopranos reruns. Drives an Audi A4. Drinks imported beer."

Player Composites

Answers the following questions about this target player.

- When and where does this person play games?
- Who buys the games this person uses?
- What platforms does this player use?
- How much time does this person spend in each session, and how frequent are gaming sessions?
- Who does he or she play with?
- What does the player like about games?
- What (non-game) brand images appeal to this player?
- How much disposable income does this player have?
- What licensed content would appeal to this player?
- What competes with gaming time for this player?

Game World

Setting and narrative are significant aspects for most games – fiction for the players mental model.

- Consistency of the virtual world is important
- Give the background - may be history of an entire civilization or just the conditions surrounding the main character, information about areas and events to keep the decisions made by various team members consistent.
- Not all world information revealed to players, but their consistency will be felt. Sequels and expansion packs are a great opportunity for revealing aspects that were kept secret in the initial game

Game Characters

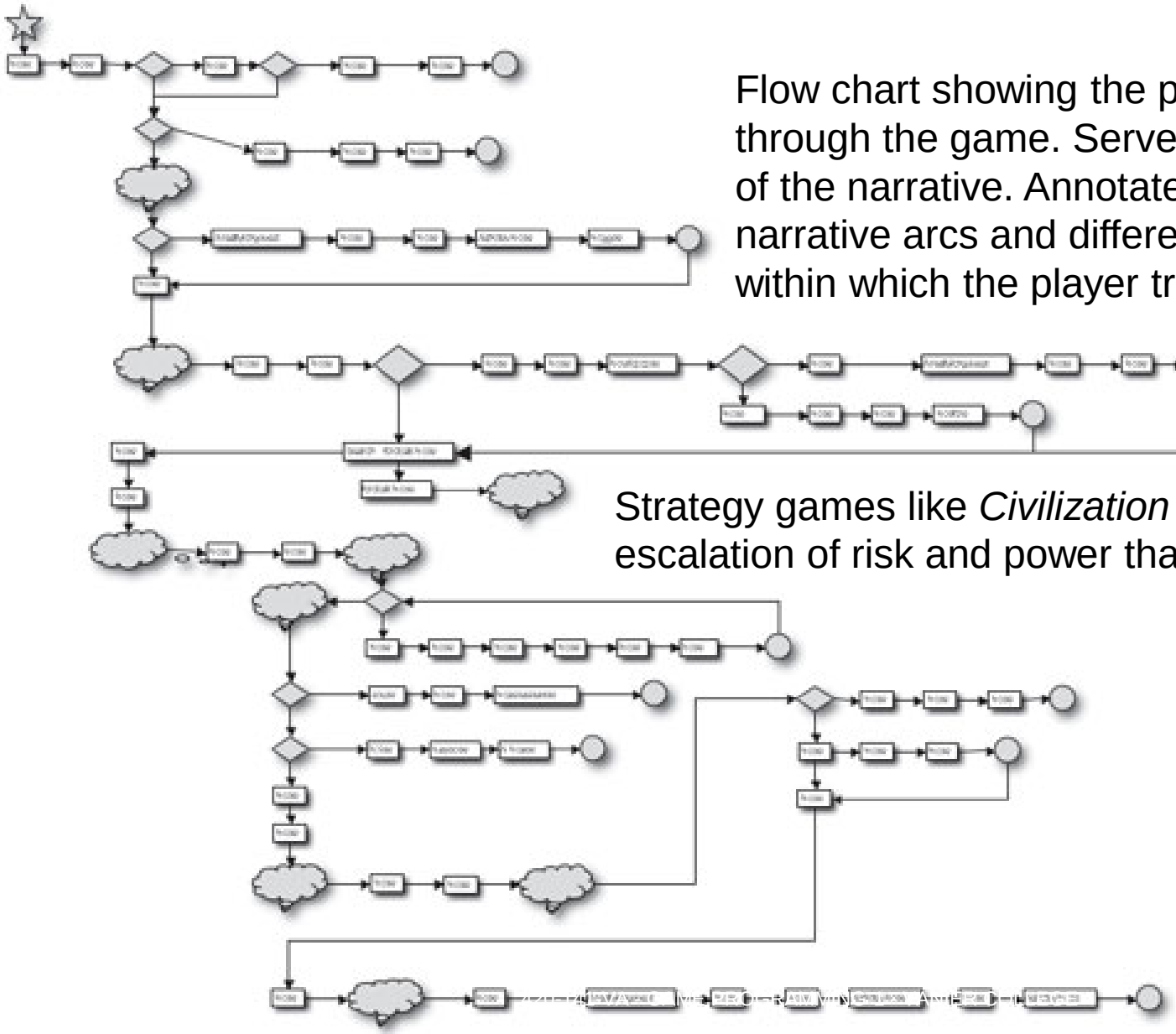
Describe each character's background and motivation, especially important for the main character.

List aspects including:

- motivation
- physical description
- likes and dislikes
- family
- friends and enemies
- vital statistics
- education
- occupation
- transportation
- tools/weapons
- clothing

For nonhuman characters, make sure their origins and race are well developed.

Game Progression - Plot Graphs



Flow chart showing the player's progression through the game. Serves as the nonlinear map of the narrative. Annotate the graph to show narrative arcs and different physical regions within which the player travels.

Strategy games like *Civilization* typically see an escalation of risk and power that can be shown here.

Beginning

Solve murder in Trinsic

(In other words, get some idea
about who did it.)
You'll get a password.

Find Lord British

Hmm, so, there's this mysteriously benevolent new cult
and ol' LB thinks I should join RIGHT up?!
Sure, old buddy, I get the hint!

Join Fellowship

What ze hell kinds of questions
were these???

Talk to Batlin

Deliver package to Minoc

...and report to

Talk to Margareta

Good time to perform this step here,
while you're around Minoc!

Visit Moonglow Observatory

Hear all about the Alignment

We need to talk
to the Time Lord.
Wisps know how!

Find the Wisps

Art Direction

[Assassin's Creed Concept Art]

This section is a combination of descriptive text and images that convey the graphic style of the game. It typically includes concept art as shown above, reference art from other sources to inspire the style, instructions on the use of lighting in a 3D game, font samples, and constraints specified by the underlying technology.

User Interface Storyboards

- For a computer game, the user interface is the installer, the title screen, menus, dialogs, in-game heads-up display and information displays, options screens, …
- Each of these must be storyboarded for both content and graphic style, although content is the primary purpose here.
- Storyboards can be simple sketches that show the GUI elements, with callout boxes and labels indicating the function of each element.
- State machines needed for each major GUI interaction showing the game modes and how the player's actions change modes.

Tags and Dialogue

Basically an indexing scheme.

A database that maps descriptive labels (*tags*) used by the programmers and designers to actual text/audio that the user will see and hear in game. For example, button: "Launch Attack!" in the game; tag: ATTACK. Later designers decide that button is: "Commence Firing."

Tags also used for dialogue lines. Tag: HelloXX could map to "Salutations, weary traveler. Come in and rest yourself for a while."

Tags are critical for internationalization.

Technology Plan

Enumerates the technology needed for producing the game, whether it will be bought or built in-house.

- That built in-house: include rationale for why and specifications.

A video game requires several major pieces of software.

Some are part of the game itself: core engine (rendering, physics, and networking) and the gameplay code (user interface, artificial intelligence, and game logic).

- Most difficult are core engine decisions: open source or mod of an engine.

Technology Plan, cont.

Other tools: maintaining the design document (e.g., Microsoft Word, wiki), the artist tools (e.g., 3DS Max, Photoshop), programmer tools (e.g., XNA, Visual Studio, VTune), management tools (e.g., Project, Excel, Trac), asset management (e.g., AlienBrain, Subversion), level editors, art exporters, and so on.

Hardware: development consoles, workstations, servers, and often special-purpose technology like motion capture and 3D scanners

Reusing existing technology allows you to focus more on design and innovation and less on reinventing the wheel!

Software Architecture

Same as for any complex software system.

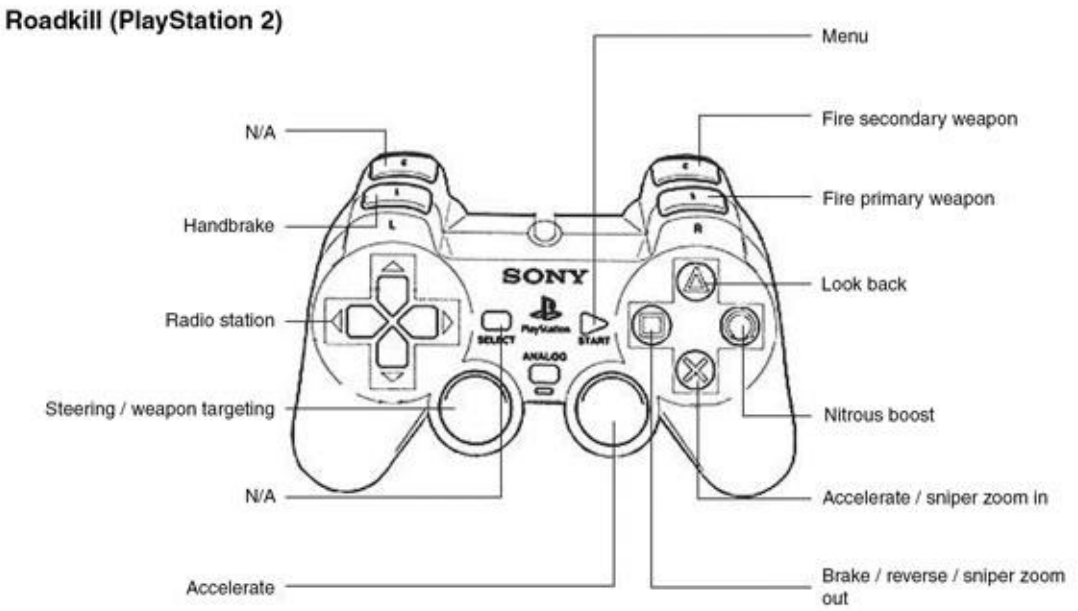
Major packages, components, modules, APIs (including algorithms to implement them), and the data and control flow.

Overview of Game Engine, Art Pipelines, Middlewares

All the details are generally in the TDD

(Technical Design Document produced by Engineers)

Controls



Video games depend on their control schemes. An otherwise great game can fail because the input mechanic is too awkward or is targeted at the wrong audience. The controls section details both the mapping of buttons to in-game functions and the algorithms mapping analog inputs to actions.

Level design

Game *levels* are a simple mechanism to “segment” the game world (a scenario or map) and gameplay (a discrete change in difficulty or style of play).

Progression to different levels shown in plot graphs.

For games with location-based levels,

- create maps that show the layout and the connectivity of the level.
- Indicate major encounters, key item locations, and goals.

Level design, cont.

Provide analysis of the level space, diagramming the flow of characters through the area and identifying *choke points* and *focus nodes*.

A choke point is a small area that controls transition between levels. *Counter-Strike* has numerous choke points, for example, the double doors in the *Dust2* map

A focus node is a location of a shared resource, increasing player interaction (e.g., a gold mine)

Counter-Strike

Mechanics Analysis

This section is the heart of the gameplay design. Economics textbooks give good examples of how to analyze, model, and balance mechanics.

Game Mechanics are small groups of rules that outline a strategic conflict, inspire a particular emotion in the player, or move the game forward. They also define the genre of the game.

Two Examples:

- Racing is a classic mechanic. - puts time pressure on the player, creating stress and motivation; need not be as explicit as cars on a track.
- Shooting is a popular mechanic. It involves reflex actions, strategy in setting up a shot before the enemy appears, and leading the target when projectiles are slow.

Mechanics Analysis

Indicate what other games use similar mechanics and describe the differences. Motivate the gameplay reason for each mechanic - prevent an undesirable strategy, simulate some aspect of the game's setting,… Explain alternative mechanics that could have been employed.

Give guidelines for making it balanced and useful for gameplay. Support with text, graphs and diagrams, and probability and other mathematical tools (e.g., outcome matrices).

For each measurable property of the game (e.g., points, gold, health), give a target graph of how it should ideally vary over time.

John Aljets discusses the GDD

https://www.youtube.com/watch?v=V-uIpXemFiA

Schedule & Related Elements

Schedule: broken down into prototype releases and milestones, and those are divided into individual tasks assigned to developers. Task dependency is put in a Gantt chart to graphically indicate when a task has been inappropriately scheduled before another task it must build on.

Staffing Plan: describes how developers will be hired over time to meet the demands of the schedule.

Key Developers: Briefly describes qualifications of team leads, producers, and managers.

Status: How is the project going?

Task / Issue Tracking

More sophisticated version of maintaining a "to-do" list. Issues are known *bugs* (also: errors or drawbacks) of the game, customer support requests, and features that have been requested but not approved or scheduled. Issues are what you aim to eventually change into lines on the change log.

Issues are commonly owned by one developer, but they are stored in a format that is visible to management and the rest of the company. A video game has so many issues that they are usually stored in a database and not in the design document itself.

Request ID	Summary	Open Date	Priority	Status	Assigned To	Submitted By
1599482	Renderbuffer name clash	2006-11-20 00:20	9	Open	nobody	scratch
1594744	iCompile recursive library dependencies	2006-11-11 11:46	9	Open	morgan3d	morgan3d
1569044	Patch to fix demos not compiling in current cvs	2006-10-02 00:11	9	Open	morgan3d	takhisis
1599034	Capsule error report	2006-11-18 19:14	7	Open	morgan3d	morgan3d
1594798	OS X apps crash on exit	2006-11-11 15:02	7	Open	ckodonnell	morgan3d
1598867	jpeglib uses tmpfile, which can break	2006-11-18 10:08	6	Open	nobody	morgan3d
1602621	Odd documentation rendering bug when viewing files locally	2006-11-25 02:17	5	Open	radioimp	takhisis

[1053332] 7.00: MeshAlg::weld must not merge 2-sided vertices

Un-monitor (?)

Submitted By:
Morgan McGuire - morgan3d

Last Updated By:
morgan3d - Settings changed

Number of Comments:
11

Data Type: (?)
Bugs

Category: (?)
None (admin)

Assigned To: (?)
None (admin)

Status: (?)
Open

Summary: (?)
7.00: MeshAlg::weld must not merge 2-sided vertices

I verified that MeshAlg::computeNormals functions correctly, however inside IFSBuilder the vertex normals are always zero.

Date Submitted:
2004-10-24 15:00

Date Last Updated:
2006-08-16 22:42

Number of Attachments:
0

Submit Changes

Group: (?)
None (admin)

Priority: (?)
3

Resolution: (?)
None

Private: (?)

Task Breakdown Exercise

Commercial Game Documents

Game Proposal:

- Diablo
- Bioshock

GDD:

- Fallout
- Leasure Suit Larry

For your Team Project

Game Proposal Document: Due October 18 (or sooner)

Simplified GDD: Due November 8 (or sooner)

- Executive summary (1 page max): Describe what the game is, description of the world, characters, enemies, main game mechanics, what needs to be done to beat the game.
- User Interface Mock-up(s): Provide a visual representation of the game, levels, user interface and flow between screens. This can be hand-drawn, or collage from online images.
- Controls: Describe how the game mechanics map to controller inputs.
- Tasks and Schedule breakdown: What are the tasks required to implement the game, assign team member responsible for each task and establish a deadlines.