**Software Design Document (SDD) Template**

Software design is a process by which the software requirements are translated into a representation of software components, interfaces, and data necessary for the implementation phase. The SDD shows how the software system will be structured to satisfy the requirements. It is the primary reference for code development and, therefore, it must contain all the information required by a programmer to write code. The SDD is performed in two stages. The first is a preliminary design in which the overall system architecture and data architecture is defined. In the second stage, i.e. the detailed design stage, more detailed data structures are defined and algorithms are developed for the defined architecture.

This template is an annotated outline for a software design document adapted from the IEEE Recommended Practice for Software Design Descriptions. The IEEE Recommended Practice for Software Design Descriptions have been reduced in order to simplify this assignment while still retaining the main components and providing a general idea of a project definition report. For your own information, please refer to IEEE Std 10161998 1 for the full IEEE Recommended Practice for Software Design Descriptions.

(Team Name) (Project Title) Software Design Document

Name (s): Lab Section: Workstation:

Date: (mm/dd/yyyy)

Índice

[1. INTRODUCTION 5](#_Toc123568660)

[1.1 Purpose 5](#_Toc123568661)

[1.2 Scope 5](#_Toc123568662)

[1.3 Overview 5](#_Toc123568663)

[1.4 Definitions and Acronyms 6](#_Toc123568664)

INTRODUCTION

1.1 Purpose 2

1.2 Scope 2

1.3 Overview 2

1.4 Reference Material 2

1.5 Definitions and Acronyms 2

2. SYSTEM OVERVIEW 2

3. SYSTEM ARCHITECTURE 2

3.1 Architectural Design 2

3.2 Decomposition Description 3

3.3 Design Rationale 3

4. DATA DESIGN 3

4.1 Data Description 3

4.2 Data Dictionary 3

5. COMPONENT DESIGN 3

6. HUMAN INTERFACE DESIGN 4

6.1 Overview of User Interface 4

6.2 Screen Images 4

6.3 Screen Objects and Actions 4

7. REQUIREMENTS MATRIX 4

8. APPENDICES 4

# INTRODUCTION

## Purpose

This software design document describes the architecture and system design of a game called Cute Crush Epidemic.

* 1. **Scope**

Provide a description and scope of the software and explain the goals, objectives and benefits of your project. This will provide the basis for the brief description of your product.

Action multiplayer game based in a epidemic, the game can be played by 2 to 4 players. Each player has its own color, as they walk around the squared board, each square they step on will gather to color of that player and earn him score. This game is meant for children so it’s rated has a PEGI +6, but adults can also play it. The same reason we avoided using images of syringes to avoid shock. This is meant to be a fun familiar game to play next to each other now that it is possible to do it again.

* 1. **Overview**

This document will be structured beginning with a full description of the mechanic’s, assets and features. After we will provide a full disclosure on the architecture of the game, how it functions in the background using Data Flow Diagrams for a better understanding.

* 1. **Definitions and Acronyms**

Use cases:

Priority’s:

P1-Essential;

P2-Moderate;

P3- Non-Essential;

Diagrams:

Ch1- Character1;

Ch2-Character2;

Assets:

Freeze- In game asset that players can catch to freeze the opponents moves;

Quick Shoes- In game asset that makes the player who catch’s it move faster in the arena;

Diamond- The player who catches this asset will have all of the points available added to is score counter;

2. SYSTEM OVERVIEW

Give a general description of the functionality, context and design of your project. Provide any background information if necessary.

Action multiplayer game based in king’s storytail. It can be played by 2 to 4 players. Each player can choose its own color and character, as they walk around the squared board, each square they step on will gather to color of that player and earn him score. These squares can change colors if another player steps on them. Meanwhile there will be assets falling from the sky, such as Diamonds, Quick Shoes., or Freeze. Diamonds will give the player who catches it the ability to get all the squares into is score, As for Quick Shoes it will allow for the character to walk on the board faster. Lastly, Freeze will “freeze” the opponent of the character who catches it, not allowing the character to move. The winner is the player with most points when the timer reaches 0. At the end of the game, there will be a score board with the top 10 highest scores.

3. SYSTEM ARCHITECTURE

3.1 Architectural Design

Develop a modular program structure and explain the relationships between the modules to achieve the complete functionality of the system. This is a high level overview of how responsibilities of the system were partitioned and then assigned to subsystems. Identify each high level subsystem and the roles or responsibilities assigned to it. Describe how these subsystems collaborate with each other in order to achieve the desired functionality. Don’t go into too much detail about the individual subsystems. The main purpose is to gain a general understanding of how and why the system was decomposed, and how the individual parts work together. Provide a diagram showing the major subsystems and data repositories and their interconnections. Describe the diagram if required.

3.2 Decomposition Description

Provide a decomposition of the subsystems in the architectural design. Supplement with text as needed. You may choose to give a functional description or an objectoriented description. For a functional description, put toplevel data flow diagram (DFD) and structural decomposition diagrams. For an OO description, put subsystem model, object diagrams, generalization hierarchy diagram(s) (if any), aggregation hierarchy diagram(s) (if any), interface specifications, and sequence diagrams here.

3.3 Design Rationale

Discuss the rationale for selecting the architecture described in 3.1 including critical issues and trade/offs that were considered. You may discuss other architectures that were considered, provided that you explain why you didn’t choose them.

4. DATA DESIGN

4.1 Data Description

Explain how the information domain of your system is transformed into data structures. Describe how the major data or system entities are stored, processed and organized. List any databases or data storage items.

Data storage is performed locally. While the game is open, the ranks and names of the player character’s will be updated on to the leader board.

After the game is closed, the rank table suffers a reset, since there’s no database being used to save data. The leader board will be updated on the next gaming sessions.

4.2 Data Dictionary

Alphabetically list the system entities or major data along with their types and descriptions. If you provided a functional description in Section 3.2, list all the functions and function parameters. If you provided an OO description, list the objects and its attributes, methods and method parameters.

5. COMPONENT DESIGN

In this section, we take a closer look at what each component does in a more systematic way. If you gave a functional description in section 3.2, provide a summary of your algorithm for each function listed in 3.2 in procedural description language (PDL) or pseudocode. If you gave an OO description, summarize each object member function for all the objects listed in 3.2 in PDL or pseudocode. Describe any local data when necessary.

Box Triger > See Scripts in Unity

Player Movement > See Scripts in Unity

6. HUMAN INTERFACE DESIGN

1.

|  |  |
| --- | --- |
| Use Case Name: | Connection to the server |
| Actor: | Client, Server |
| Priority: | P1 |
| Action: | User accesses Webpage |
| Pre requisite: | User is connected to the internet and on the game page |
| Basic Path: | 1. Client forms TCP connection with the server. 2. User opens game webpage |
| Alternate Path: | 1. Connection Failed:    1. Inform user that connection was not possible    2. Try again on automated page reload    3. Connection not possible due to internet issues or server is down |
| Postcondition: | User client establishes connection and can send/receive data from the server. |
| Exception Path: |  |

2.

|  |  |
| --- | --- |
| Use Case Name: | Character Selection |
| Actor: | Client |
| Priority: | P1 |
| Action: | Player 1 chooses the character |
| Pre requisite: | User scrolls to each character and chooses one. To play with. |
| Basic Path: | 2.1-Make connection with server  2.2-Read mouse Input |
| Alternate Path: |  |
| Postcondition: | Player 1 chooses character; Player 2 performs the same actions as Player1. |
| Exception Path: |  |

3.

|  |  |
| --- | --- |
| Use Case Name: | Start Game |
| Actor: | Client |
| Priority: | P1 |
| Action: | Player’s click Start Game button |
| Pre requisite: | User scrolls to each character and chooses one to play with. |
| Basic Path: | 3.1-Make connection with server  3.2-Read mouse Input |
| Alternate Path: |  |
| Postcondition: | Game Starts |
| Exception Path: |  |

4.

|  |  |
| --- | --- |
| Use Case Name: | In Game moves |
| Actor: | Client, Server |
| Priority: | P1 |
| Action: | Player’s use specified keys to move in the arena. |
| Pre requisite: | Start Game |
| Basic Path: | 4.1-Player 1 uses “A”,” W”,” S”,” D” to move in the arena  4.2-Player 2 uses Directional Arrows to move in the arena  4.3- Read mouse Input |
| Alternate Path: |  |
| Postcondition: | Client sends the move options from booth players to the server. |
| Exception Path: |  |

5.

|  |  |
| --- | --- |
| Use Case Name: | Game state Check |
| Actor: | Client, Server |
| Priority: | P1 |
| Action: | Server receives moves from booth players |
| Pre requisite: | In Game moves |
| Basic Path: | 5.1-Server receives moves from users  5.2-Check if the move is valid  5.3-Sends updated game state to booth players |
| Alternate Path: |  |
| Postcondition: | Update screen for booth players |
| Exception Path: |  |

6.

|  |  |
| --- | --- |
| Use Case Name: | User’s catch in game assets |
| Actor: | Client, Server |
| Priority: | P1 |
| Action: | Connection is reachable |
| Pre requisite: | In Game moves |
| Basic Path: | 6.1-User’s catch assets  6.2-Information sent to server  6.3-Server sends updated score to booth players  6.4-Sends updated restrictions depending on asset caught up |
| Alternate Path: |  |
| Postcondition: | Update screen for booth players |
| Exception Path: |  |

7.

|  |  |
| --- | --- |
| Use Case Name: | End game |
| Actor: | Client, Server |
| Priority: | P1 |
| Action: | Timer reaches end |
| Pre requisite: | Start Game |
| Basic Path: | 6.1-Maximum timer set for 2min  6.2-Game ends  6.3-Client sends update scores to server  6.4-Server sends update scores to client |
| Alternate Path: |  |
| Postcondition: | Update screen for booth players, Restart button |
| Exception Path: |  |

|  |  |
| --- | --- |
| Use Case Name: | Restart?? |
| Actor: | Client, Server |
| Priority: | P1 |
| Action: | Timer reaches end |
| Pre requisite: | Start Game |
| Basic Path: | 6.1-Maximum timer set for 2min  6.2-Game ends  6.3-Client sends update scores to server  6.4-Server sends update scores to client |
| Alternate Path: |  |
| Postcondition: | Update screen for booth players, Restart button |
| Exception Path: |  |

6.1 Overview of User Interface

Describe the functionality of the system from the user’s perspective. Explain how the user will be able to use your system to complete all the expected features and the feedback information that will be displayed for the user.

6.2 Screen Images

Display screenshots showing the interface from the user’s perspective. These can be handdrawn or you can use an automated drawing tool. Just make them as accurate as possible. (Graph paper works well.)

6.3 Screen Objects and Actions

A discussion of screen objects and actions associated with those objects.

Assets

Diamonds will give the player who catches it the ability to get all the squares into is score.

As for Quick Shoes it will allow for the character to walk on the board faster.

Lastly, Freeze will “freeze” the opponent of the character who catches it, not allowing the character to move.

Timer- The timer will start at 2 minutes and goes down until reaches 0, where the game is over.

Scores- Booth users will have associated scores to keep track on how many points each player has.

Top10 Scoreboard- At the end of the game, a scoreboard will be displayed with the hights 10 scores.

7. REQUIREMENTS M ATRIX

Provide a crossreference that traces components and data structures to the requirements in your SRS document.

Use a tabular format to show which system components satisfy each of the functional requirements from the SRS. Refer to the functional requirements by the numbers/codes that you gave them in the SRS.

8. APPENDICES

This section is optional.

Appendices may be included, either directly or by reference, to provide supporting details that could aid in the understanding of the Software Design Document.