

# SE 3XA3: Development Plan Mario

Team 9, Ninetendo  
David Hobson - hobsondd  
Jose Miguel Ballesteros - ballesjm  
Jeff Pineda - pinedaj

December 7, 2016

**Contents**

**List of Tables**

**List of Figures**

This document describes the requirements for Mari0. The template for the Software Requirements Specification (SRS) is a subset of the Volere template (?).

# **1 Project Drivers**

## **1.1 The Purpose of the Project**

The purpose of this project is to recreate the game, Mari0, to allow the players to entertain themselves and alleviate their boredom. Mari0 is a combination of Super Mario Bros. and Portal, challenging a player's platforming abilities and their puzzle solving skills.

## **1.2 The Stakeholders**

### **1.2.1 The Client**

The client for the Mari0 project is the game publisher.

### **1.2.2 The Customers**

The customers for this project are people interested in platforming and/or puzzle games.

### **1.2.3 Other Stakeholders**

The other stakeholders for Mari0 are the game's developers and designers. Along with the client and the customers that are interested in playing games and puzzles.

## **1.3 Mandated Constraints**

The constraints as mandated by our client are as follows:

- Have each deliverable finished by the deadlines given in the course outline
- The game's physics will be handled by the Unity Game engine
- The product should be runnable on all operate systems

## 1.4 Naming Conventions and Terminology

## 1.5 Relevant Facts and Assumptions

The users of this game are the players, and despite Mario0 being derived from the games Super Mario Bros. and Portal, the mechanics and game knowledge that come from playing these games is not assumed.

It is also assumed that the user has a computer, a keyboard, and a mouse to play the game.

# 2 Functional Requirements

## 2.1 The Scope of the Work and the Product

### 2.1.1 The Context of the Work

The game design and interface will be simple and effective. The user will be able to provide input through their computers and will be able to receive the expected response, as shown in Figure ???. For example, if the user presses the up button and expects to jump, the character on screen will provide a jumping animation for feedback.

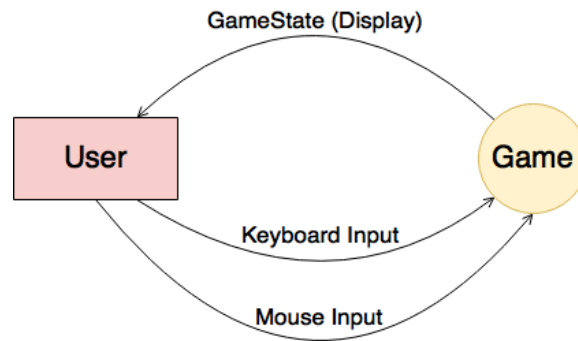


Figure 1: Context Diagram of the gaming system

Table 1: **Revision History**

Date	Version		Notes
October 6, 2016	1.0		Created Document, rough draft of section 1
October 11, 2016	1.1		Added Problem Issues
October 11, 2016	1.2		Finished Project Drivers and Non-Functional Requirements
November 28, 2016	1.3		Updated Functional Requirements to be more aligned with project
December 5, 2016	1.4		Removed some terminology and definitions

Table 2: Terminology and Definition Table

Term	Definition
Mario	The character that the player portrays
Portals	Two connected portals that allow characters and projectiles to enter one and exit through the other, whilst maintaining physics properties such as velocity and acceleration
Goomba	Enemy character that is defeated after the player stomps on the top of its head
A.I.	Artificial Intelligence
Lives	The amount of times the player can die before game over
Question Block	Blocks found that when hit give the player coins or power ups

### 2.1.2 Work Partitioning

Table 3: Work Partitioning

Event Name	Input/Output	Summary
Menu Change	Keyboard(in)/Mouse(in) GameState (Out)	When the user selects a menu that they'd like to visit, the menu will display it
Movement	Keyboard(in) GameState (Out)	When the user presses a key to move the character, the character will display its movement animations
Jumping	Keyboard(in) GameState (Out)	When the user presses a key to jump the character, the character will display its jumping animations
Portal Shooting	Mouse(in) GameState (Out)	When the user clicks where and which portal they'd like to shoot, the game will display them accordingly
Enemy Hits Player	GameState(Out)	When an enemy hits the character the game will display the defeat animation and the game will reset with one less life
Player Hits Enemy	GameState(Out)	When the character hits an enemy on the right spot, the enemy will disappear
Scoring	GameState(Out)	When the character collects points, the game's score display will increase
Pause	Keyboard(in) GameState(Out)	When the user presses the pause key, the game will display the pause menu

### 2.1.3 Individual Product Use Cases

Use Case #: 1

Name: Play Game

Trigger: The user selects to play a new game

Precondition: The game is in its main menu

Post Condition: The game begins

Use Case #: 2

Name: Help

Trigger: The user selects to view help options

Precondition: The game is in its main menu

Post Condition: The help menu displays

Use Case #: 3

Name: Character Moves

Trigger: The user presses a key to move their character in a certain direction

Precondition: The game is playing

Post Condition: The character will move in the corresponding direction

Use Case #: 4

Name: Portal gets fired

Trigger: The user places the mouse where they'd want a portal and clicks left or right for which portal they want

Precondition: The game is playing and the portal is in a plausible place

Post Condition: The portal will get formed

Use Case #: 5

Name: Pause

Trigger: The user presses the pause key

Precondition: The game is playing

Post Condition: The pause menu is shown

## 2.2 Functional Requirements

1. **The game will have a main menu with multiple options for the user.**

*Rationale:* The game must feature a place to have all gameplay options organized before a new game begins

*Fit criterion:* Main menu is displayed upon running.

2. **The game will feature a help menu.**

*Rationale:* The game must have some sort of way of telling the user how to play the game.

*Fit criterion:* Help menu is displayed when selected

3. **The user will be able to pause the game.**  
*Rationale:* The game must allow the user to pause gameplay so that they can take a break when needed  
*Fit criterion:* Gameplay is paused and the game stops.
4. **The game will display the character's current state.**  
*Rationale:* The user needs a method to track lives and score  
*Fit criterion:* The game displays the character's current score
5. **The game will feature portal system.**  
*Rationale:* The gameplay must feature a method to help the user escape difficult situations  
*Fit criterion:* The user is able to select which portal they want and where they'd place it
6. **The user will be able to move their character.**  
*Rationale:* The user must have a way to move their character around the screen  
*Fit criterion:* The user is able to move the character to their desired location
7. **The user will be able to make their character jump.**  
*Rationale:* The user must have a method to make their character escape obstacles  
*Fit criterion:* The user is able to see their character jump as wanted
8. **The game will feature powers ups.**  
*Rationale:* The game must allow for ways to help the user's character  
*Fit criterion:* The user gets benefit from the power up
9. **The character will be able to collect coin.**  
*Rationale:* The user needs ways to increase their character's score  
*Fit criterion:* The game will increase score total when the character



collects a coin

10. **Enemies will defeat the character**

Rationale: The game uses this technique to add difficulty to the game

Fit criterion: The character will display the defeat animation when hit by an enemy

11. **Characters will be able to defeat enemies.**

Rationale: The game needs a method to defeat enemies

Fit criterion: Enemies will be defeated when hit by the character from the top

12. **The character may fall off the platforms.**

Rationale: The user's ability to control their character will add difficulty to the game

Fit criterion: Character will be beaten after a fall off screen

13. **The character will be able to increase their score.**

Rationale: The game needs a way to increase score to have different gameplay scenarios

Fit criterion: The character can increase score after beating an enemy or collecting a coin

### 3 Non-functional Requirements

The numbers to the left of each listed requirement indicates requirement number/label.

#### 3.1 Look and Feel Requirements

16. The game will load the two dimensional level the player is currently on. The playable foreground should be composed of Mario, enemies, and a variety of bricks with different attributes that can serve as platforms to

jump to and from. The background of the level will change depending on the where the level is located. If the level is outside, the background will consist of a blue sky with a few clouds, small hills, and small bushes. If the level is underground or in a castle, the background will be pure black with no additional aesthetics added.

17. The interface displaying the player's number of lives remaining, the current level, coins earned, and the time remaining in a non-obtrusive manner.

### **3.2 Usability and Humanity Requirements**

18. The game should be easy to learn for all player's, regardless of their experience with the Super Mario Bros., Portal, or other similar games.
19. The game should be fun and entertaining.

### **3.3 Performance Requirements**

20. The game's visuals should respond to all commands inputted by the user without a noticeable delay. This will ensure that the player feels in full control at all times.

### **3.4 Operational and Environmental Requirements**

21. The game should be able to run on any laptop or desktop computer.

### **3.5 Maintainability and Support Requirements**

22. The game should be able to execute on Windows, OSX, and Linux based machines.

### **3.6 Security Requirements**

23. The game should not be able access or alter any information on the user's computer besides updating the player's saved game files.

### **3.7 Cultural Requirements**

24. The game's written components, such as messages and menu options should be written in English.
25. The game should not have any messages, images, or depictions of religious, political, or ethnic significance.

### **3.8 Legal Requirements**

26. The game should be licensed under the Creative Commons by Non-Commercial Share-Alike 3.0 license.

### **3.9 Health and Safety Requirements**

27. The game should not have any triggers for epileptic seizures.

## **4 Project Issues**

### **4.1 Open Issues**

Currently, there are no known major issues for the project, however as the implementation continues there may need to be changes and problems may start to occur. Here are the list of some current issues with the game that may need to be improved:

- Some old operating systems such as Windows XP and Windows 7 have difficulty running the game properly.
- Lack of modding support.
- Players run into problems accessing the save folders.
- Players have trouble using the Love framework in order to run the game

### **4.2 Off-the-Shelf Solutions**

A lot of the issues will be solved easily with the use of Unity, since it is professionally made for game developers, and it is accessible to many operating systems. Also players will not have to deal with the Love framework that

is apart of the current Mario game, since the entire project will be created in Unity. Furthermore, if there are any huge problems, the code that has created the current game can be slightly modified to fit with Unity and will be essentially used as a prototype. In addition, Unity has it's own physics engine that will be a huge help in creating the environment for the game, instead of creating the entire game environment from scratch. It may also help to look into other successful Unity games to see how things have been implemented, some platforming games may give us insight into how to make sure the physics works best for our final product.

### **4.3 New Problems**

New problems have yet to arise in the implementation. However, using Unity may add more problems in how things will be ran on certain systems if we would like to port the game to a mobile device or a console system. Overall, no new problems are expected to rise as a result of this project.

### **4.4 Tasks**

Tasks are listed and numbered below.

1. Structures - Create class hierarchies and main game objects.
2. Overall Mechanics - Getting the character moving between two portals and interacting with environment
3. Level Design - Creating levels that the user can play and the character can be placed into.
4. Interfaces - Main programming interfaces such as Menu, Game Over, and Pause Screens.
5. Graphics and Sound - Main graphics, music and animation for the game.
6. Improvements - Adding different aspects of the game such as newer levels or different mechanics.

## **4.5 Migration to the New Product**

Since we are creating the same product, there is no migration to a newer product at this time.

## **4.6 Risks**

Overall, risks are few and far between when it comes to recreating Mario, but there are some risks that we would like to minimize. Flashing colours on the screen may trigger epileptic seizures for some users. Also, if the game is not optimized well, overheating of the system may damage users systems or cause minor burns. Although the chances of these problems arising are extremely low, they will be kept in mind when creating the final product.

## **4.7 Costs**

Currently there are no costs associated with this project.

## **4.8 User Documentation and Training**

User Documentation will be created as per the SFWR 3XA3 guidelines. Training/Tutorial will be implemented into the game through screen shots or a small in game user manual.

## **4.9 Waiting Room**

There are currently no requirements or problems that have not been met or solved. This section will be updated as needed.

## **4.10 Ideas for Solutions**

There are currently no ideas for solutions and no overall plan for these solutions. This section will be updated as needed.

## 5 Appendix

This section has been added to the Volere template. This is where you can place additional information.

### 5.1 Symbolic Parameters

The definition of the requirements will likely call for `SYMBOLIC_CONSTANTS`. Their values are defined in this section for easy maintenance.