Software Requirements Specification
For
Arcade game “Alpha”
CS-243 Team 22
February 19, 2016

Prepared by :
1)140101035 Longkiri Bey
2)140101057 Rajan Garg
3)140101058 Rishabh Sharma
4)140101019 Dasari Bindu Bharadwaj

Table of Contents

1. Introduction

1.1 Purpose

This is the Software Requirements Specification (SRS) for the “Alpha” project. The purpose of the document is to describe the purpose and functionality of the game in CS-243 project under the instructor of course Saurabh Joshi. The SRS will include the details

of the project's requirements, interface, design issues, and components.

1.2 Intended Audience and Reading Suggestions

The audience of this project will be people who like to play PC games. Specially this game would be popular among children as it requires less intelligence and more fun to be played.

1.3 Product Scope

This product is specially for children. Mostly children can play simple arcade games rather than difficult games. So it would be popular among children as well as teenager people who pass their time playing PC games.

1.4 References

Tools used: Unity 5.3 (Unity is the ultimate game development platform. It is used to build high-quality 3D and 2D games and deploy them across mobile, desktop etc.)

Platform used: C# (needs no explanation).

References: http://unity3d.com/learn/tutorials (Online tutorials)
http://docs.unity3d.com/Manual/index.html (reference manual)
http://unity3d.com/learn/live-training (training sessions)
http://unity3d.com/community (Discussion forums)

1.5 Definitions, Acronyms, and Abbreviations

TERM	DEFINITION
CS-243	Computer Science, course code 243
SRS	Software Required Specifications (this document)
UML	Unified Modelling Language
2D	Two dimensional
Side scrolling game	GamePlay action is viewed from a side-view camera angle, and the onscreen characters do move from the left side of the screen to the right or opposite to meet an objective.
Use Case	It is a list of actions or event steps, typically defining the interactions between a role and a system, to achieve a goal. It is a collection of related success and failure scenarios, describing

	actors using the system to support a goal.

2. Overall Description

2.1 Product Perspective

Alpha is a 2D arcade game viewed from a side-view camera angle. The player controls Alpha, an adventurer who seeks points in various levels of the game. The aim of the game is to collect as much points as possible in the levels and then finishing the level within the stipulated time to reach to the next level. Stages will have enemies and other difficulties that can kill the player. Each level must be completed within a time limit or else the player loses the chance to enter next level within the same Alpha life.

2.2 Product Functions

This is the game so its most important function is that it is a source of entertainment. Also has interactive and fancy animations and UI.

2.4 User Documentation

User manual is available on:
http://docs.unity3d.com/Manual/index.html

2.5 Assumptions and Dependencies

1. The event and animation logger component will be fully functional.
2. Game remains stable and compatible with Windows 7 and greater.
3. The game will be functional with visible activities and animations

3. External Interface Requirements

3.1 User Interface

The interface for this program will be relatively simple. As the target users are generally children, this product will be as graphically oriented and appealing as possible. Portion of the interface will require the keyboard; all input will be accomplished via mouse clicks or keyboard arrow keys/power keys.

3.2 Hardware Interfaces

There are interfaces for us to interact with keyboard and mouse. All of these are handled by the Unity classes and will mostly be simple for us to interact with. The program requires rather modest computer hardware.

Hardware requirements for graphic's features

Data reference: http://docs.unity3d.com/Manual/RenderTech-HardwareRequirements.html

3.3 Software Interfaces

It depends very much on the platform but in general if the platform has decent 2D/3D support and up to date drivers, you should not need anything else.
Unity uses PhysX for its physics. PhysX has a software fall back but you will get better performance if your system has a GPU that PhysX knows how to use for acceletation. Look PhysX for more details.

Generally content developed with Unity game can run pretty much everywhere. How well it runs is dependent on the complexity of your project. More detailed requirements:
Desktop:
OS: Windows XP SP2+, Mac OS X 10.8+, Ubuntu 12.04+, SteamOS+
Graphics card: DX9 (shader model 2.0) capabilities; generally everything made since 2004 should work.
CPU: SSE2 instruction set support.
Web Player (deprecated): Requires a browser that supports plugins, like IE, Safari and some versions of Firefox

iOS: requires iOS 6.0 or later.

Android: OS 2.3.1 or later; ARMv7 (Cortex) CPU with NEON support or Atom CPU; OpenGL ES 2.0 or later.
WebGL: Any recent desktop version of Firefox, Chrome, Edge or Safari
Windows Phone: 8.1 or later
Windows Store Apps: 8.1 or later

3.4 Communication Interfaces
There aren't any communication interfaces as currently the game is single player and does not send out the data it collects unless run in the lab.

---

4. Software Features

Events Logged

#Stimulus/Response Sequences

Events from the system and user will be logged constantly. Events include: jumping of player, change of scene, coins and enemy counter etc.

Multiple Levels

#Description and Priority

The game will consist of multiple levels. At the completion of a level, another level will be generated. Each new level generated will be slightly more challenging by adding new attributes, time, enemies.

#Stimulus/Response Sequences

Each level will be randomly generated or selected by the user when the new level is started.

#Functional Requirements

1: At the completion of a level, a new level will be generated.

2: At the completion of the all levels, the game will end.
3: More points will be given for more difficult levels.

- There will be a Alpha named sprite which will move in according to directions given by us.
- This is a side-scrolling game that is GamePlay action is viewed from a side - view camera angle, and the onscreen characters do move from the left side of the screen to the right to meet an objective.
- Game make use of scrolling computer display technology.
- The objective of the game is to clear the level(s) by dodging enemies and earning points.
- Alpha can earn points and there will be a score board which will show the number of points scored.
- Alpha will have to survive against the enemies who will arrive in the scenario as the game progresses.
- When level will be completed next level would come.
- There would be 3-4 levels of the game which will have different type of arcade game set-up.
- To make game more excited we shall make 4 lives of Alpha.
- Score, levels and lives will be visible on the screen.
- There will be all of the features of an arcade game.
- There will be buttons on screen to control some actions of game.
- There will be background sound of Alpha Game.
- There will be a Score board which will be shown when game will end.

4. Use Case

<interface> item is:
* toString(): String
* used():boolean
* getXposition():int
* getYposition():int
* isCaught():boolean

<stereotype> block is:
* xPosition:int

* yPosition:int
further regular block, coins block and item block are:

1>Regular Block:
* xPosition:int
* yPosition:int
* image:image
* getTouched():void

2>Item Block:
* xPosition:int
* yPosition:int
* image:image
* item:IItem
* getTouched():Item

3>Coins Block:
* xPosition:int
* yPosition:int
* image:image
* item:IItem
* getTouched():Coins

stereotype <player> is:
* xPosition:int
* yPosition:int
* HP:int
* Coins:int
* item:List<IItem>

* getXPositon:int
* getYPositon:int
* getHP:int
* getCoins:int
* getItem:List<IItem>
* jump():void
* moveForward():void
* moveBackward():void
* hit(IObstacle):void
* touch(Iobstacle/Iitem):void

The player has the following sets of use cases:

Use case: Regular Player

Brief Description

The player normal stance for interaction other than enemies and coins/items touching

Before this use case can be initiated, the player has already accessed one of the level of

the game
The follwing descirption defines the normal interaction
* image:image

1. useItem():void
2. jump():void
Jump non-interacting with enemies/coins/items
3. moveForward():void
Moving forward and non-interacting with enemies/coins/items
4. moveBackward():void
Moving backward and non-interacting with enemies/coins/items
5. touch(IObstacle/Iitem):void
Interaction with enemies/coins/items

Use case: Player + Items Block

Brief Description:
The player's interaction with an item block

Before this use case can be initiated, the player has already accessed one of the level of the game and there is presence of an item block on the game screen.
The follwing descirption defines the interaction of the player with item block
*image:image
*item:Iitem

1. xPosition:int
The x-coordinates of the image of item
2. yPosition:int
The y-coordinates of the image of item
3. getTouched():Item
Interaction when the player touch the block. It depends on the orientation of the block with respect to the player as well.Player hits the block from the following orientation:
a) Left -no effect
b) Right -no effect
c) Below -effect activated corresponding to the function of the block
d) Standing on the item- no effect

Use case: Player + Coins Block

Brief Description:
The player's interaction with an coin block

Before this use case can be initiated, the player has already accessed one of the level of the game and there is presence of a coin block on the game screen.
The follwing descirption defines the interaction of the player with item block
* image:image
* item:Iitem

1. xPositon:int
The x-coordinates of the image of coin

2. yPositon:int
The y-coordinates of the image of coin
3. getTouched():Coins
Interaction when the player touch the coin. It depends on the orientation of the coin block with respect to the player as well.Player hits the coin block from the following orientation:
a) Left -no effect
b) Right -no effect
c) Below -effect activated and the score of the layer increases and the effect is shown on high scoreboard as well
d) Standing on the coin block- no effect

ADDITIONAL FEATURES TO BE ADDED IF TIME ALLOWS:

- Some levels of the game are inhabited by creatures that must be avoided or vanquished.
- Some levels contain hidden passages that can only be discovered by finding the appropriate lever or switch hidden in that particular level.
- Magical gems can be found in some levels and these can extend the time allowed to reach the next level.
- In some levels the player can pick up a magic torch which illuminates a larger portion of the area.
- Stars can be found in some levels and these can make Alpha unbeatable against enemies for a short span of 8 seconds.
- Flowers can also be found in some levels and these gave Alpha the special power to fire balls and devastate its enemies.

6. Requirements

6.1 Functional Requirements

- This game required a player who control the Alpha.
- A computer, on which, this application is run.
- There is a keyboard which is used to control Alpha pressing arrows keys and power keys.
- The game will be controlled by mouse and keyboard.
- The game will be displayed on a visual display and will be fit to the screen(adjustable) in size.
- The game will feature music and sound effects; however these will not be

crucial for gameplay. Speakers or headphones are an optional user requirement.

- The completed game file must be capable of being embedded in a web page.
- The game must have a title screen with buttons that allow navigation to the game screen, instructions screen and credits screen.
- The game must have an instructions screen. The user can navigate from the instructions screen back to the title screen..
- The game will feature ‘next level’ screens that appear between the levels of the game. These screens show the current score. A button allows navigation to the next level.
- The player character can move up, down, left and right, using the arrow keys. >The game will have more than one level.
- Each level must be completed within a time limit or the game will end.
- Each level has a time limit.
- The player character must reach the level end within the time limit to proceed to the next level.
- The player can pick up points.
- The game ends when the player character completes the last level.
- Scoring in the game is based on how much points has been collected in each level.
- The current score and level are displayed on the game screen.
- The game will feature appropriate sound effects.
- A sound will play when the player picks up points.
- A ‘victory’ sound will play when the player successfully completes a level.
- Appropriate music will play throughout the game.

6.2 Performance Requirements

The game must be able to run at 30-60 frames per second on Windows platform OS. At this frame rate, the game will remain constant and playable. It will not be technically demanding and able to run on lower end computers.

Only one person can use a single instance of the product. However, the product will reside on the Internet so more than one user can access the product and download its content

for use on their computer.
The system should respond to each user input instantly. There are no other performance requirements.

6.3 Design constraints

The main design constraint is that the game must be playable by people suffering with autism, therefore the proper considerations must be made for the users. There are certain safety requirements that we must follow.

The game will be written in C# using the Unity libraries and functions. Therefore, the game will comply to the programming practices of C#. The delivered software will be open-source.

6.4 Legal Requirements

The only legal requirements in the game is the legal requirements of the platform used for building this game which is Unity 5.3 (game development software). The terms and conditions ("Terms") and our Privacy Policy, which may be found at http://unity3d.com/legal/privacy-policy and which is incorporated here in by reference into these Terms. These Terms govern access to and use of the Site and the Services and, except as otherwise provided in these Terms, all text, graphics, images, music, audio, video, information or other materials available through the Site and Services ("Content") and constitute a binding legal agreement between you and Unity.

The Unity Asset Store has its own terms and conditions, which can be found at http://unity3d.com/legal/. Additionally, certain areas of the Site (and your access to or use of certain Services, Content or User Content (defined below)) may have different terms and conditions posted or may require you to agree with and accept additional terms and conditions. If there is a conflict between these Terms and terms and conditions posted for a specific area of the Site, Services, Content or User Content, the latter terms and conditions will take precedence with respect to your use of or access to that area of the Site, Services, Content or User Content.

6.5 Quality Attributes

The two main quality attributes that the game will be focusing on is correctness and usability. The system needs to be able to track time correctly. Since the game will need to record events and timestamp them with extremely high accuracy, it is important that the timing is accurate and consistent. If one of these events were to be off by a small amount then the research data provided could be incorrect.

Usability is also a priority because this is a game. If the user were to be confused or annoyed by the User interface, then the game would be less enjoyable. There are also special considerations for autistic children to include when developing the UI. These are more detailed on the autisim collab wiki.

Reliability will be ensured by extensive testing by the team members and mentors, if available .Maintainability is a primary goal for this project.

6.6 Other Requirements

There are no additional requirements at this time.