# Remote Raiders

# Requirements Specification

| | |
|---|---|
| **Group Members** | Jeremy Bennett, Vincent Engard,<br>Zach Howell, Michael Irizarry,<br>Tomas Rodriguez, Matt Walker |
| **Faculty Advisor** | Dr. Filippos Vokolos, Ph. D. |
| **Project Stakeholder** | Dr. Michael Wagner, Ph. D. |

# Revision History

| Name | Date | Reason for Change | Revision |
|---|---|---|---|
| Jeremy Bennett, Zach Howell, Tom Rodriguez | 11/7/2015 | First Draft - Sections outlined | 0.9 |
| Jeremy Bennett, Vincent Engard, Zach Howell, Michael Irizarry, Tomas Rodriguez, Matt Walker | 11/21/2015 | Second Draft - Content added to all sections | 1.0 |
| Jeremy Bennett, Vincent Engard, Zach Howell | 12/5/2015 | Third Draft - Revisions from other students' feedback. Removed R4.2.5. | 1.1 |

# Table of Contents

# 1. Introduction

## 1.1 Purpose of Document

This document will provide all of the requirements for the game Remote Raiders. It will serve as a reference for the developers and the customers for developing the final version of the game.

## 1.2 Scope of Document

This document will contain enough information such that a developer will be able to easily translate the requirements to code without ambiguity.

## 1.3 Overview of Document

This document will contain functional and nonfunctional requirements for the game Remote Raiders, as well as use cases, diagrams, and UI mock-ups. The game contains a component that will run on a PC and a component that will run on mobile phones. Henceforth the PC component will be called **server** and the mobile component will be called **mobile**.The requirements will be provided for both the PC component and the phone version of the document. This document will also contain mock-ups for the graphical user interface and use cases.

# 2. Description

## 2.1 Product Perspective

Remote Raiders is a party game designed to be played in one room with a group of friends. When each player joins the game, they will be placed on a central map with the other players and given a hidden objective. To complete their hidden objective, the player will need to explore the map and choose to help or hinder fellow players who likewise seek to complete objectives.

**Figure 1:** *A player controlling his avatar with his mobile phone.*

The game is intended to be run on one computer, the server**,** and displayed on a projector or large screen such that it's visible to all players. In order to accommodate any size party, each player will use their smartphone device as a controller (mobile). The controller will be provided as another application that each player will need to install on their phone.  The smartphone serves a dual purpose in that it can display the user's objective and other hidden information solely to that player.


**Figure 2:** *Multiplayer video game Legend of Zelda: Four Swords of Adventure. See Reference 1.*

The game itself will be 2-D, and the camera will show a top-down view of the map as displayed in Figure 2. The map will contain many rooms, and many diverse items and traps to encourage interaction among the players. The game will have a theme of archeology (think *Indiana Jones*) mixed in with other, related themes involving the ancient world. The players will use items found in the level to accomplish an objective and prevent others from doing the same.

### 2.1.1 Server PC Interface

The server is intended to be projected on the playing screen that can be viewed by all players. It will contain a menu for starting a game so that players with the mobile controller can connect and begin playing. The server will display the time left in round, the map, and all connected players' **avatar** on that map.

### 2.1.2 Mobile Interface

The mobile interface will act as an individual player's controller. The player will be able to connect to the server after a menu, and then the controller will be displayed on their phone. The top half of the controller displays context sensitive information. The bottom half is static and mainly

controls their avatar. The bottom half will have a joystick for movement, buttons to use items, and a button to attack. The player will also be able to access a menu to change various options and quit.

## 2.2 Product Functions

### 2.2.1 Server Functionality

The Server will have the following functionality:

- Ability to host a game session over the Wi-Fi network, and manage multiple phones and players
- Ability to run the game and restart after it's over
- Ability to generate content by placing items, environment objects, and objectives

### 2.2.2 Mobile Controller Functionality

The Mobile Controller will have the following functionality:

- Ability to detect hosts and connect to hosts
- Ability to send messages to affect the player's avatar
- Respond to events on the server by updating UI elements

## 2.3 User Description

The ideal users for Remote Raiders would be a small group of friends (4-10) or a group of people in a party setting.

## 2.4 Assumptions and Dependencies

### 2.4.1 Unity

Unity is a professional game development engine that allows users to create video games. This project is completely dependent on Unity functioning and working properly. If Unity crashes or ceases support, the team will be unable to continue working, and the project itself fails. There are other engines the team could work on, but that would require starting over entirely and remaking most, if not all, of the assets. The team is assuming Unity will work as necessary and without error.

### 2.4.2 Unity Networking

Networking is required for this concept to function properly because phones are required to wirelessly connect to computers. If this functionality breaks, so does the game. This networking relies entirely on how well Unity's networking feature functions. If it does not work, the game will be buggy, glitchy, and players will end up disconnecting at random points of the game. Currently there is an issue where players disconnect during play. The team is working to fix this issue, so it will not happen in the future.

### 2.4.3 Assets from Art Team

The client is providing the programmers with art assets. These assets are needed to properly schedule work. These assets are not necessary to make the desired game, but because it is the client's wish to include these assets, they are required. However if the team does not get the assets from the client, it is not the team's responsibility for creating those assets. Deadlines have been exchanged between teams for check ups on the coding side and the necessary art assets from the art team. As long as these deadlines are followed, the project will be delivered on time.

### 2.4.4 Android Support

Remote Raiders is only being programmed to work on Android (5.0+) devices. Unity supports this functionality, so there are no concerns over whether the game can be run on Android devices. The team is assuming that the Android devices will not crash at any point of running the game. Any crashes that happen are assumed to be a programming malfunction that the team is working to avoid and prevent.

## 2.5 Requirements Apportioning

| Priority Level | Description |
|---|---|
| **1** | **Priority 1** requirements are essential to the product and must be in the final build. These requirements must be tested and verified to ensure proper functionality. |
| **2** | **Priority 2** requirements are not required for the final build, but will be provided if there is sufficient time. The system will be designed such that it is extendable to easily incorporate these requirements at a later time. |
| **3** | **Priority 3** requirements are not required, and will not be considered in the design of the system. If sufficient time remains the requirement will be incorporated. |

# 3 Functional Requirements

## 3.1. Server

### 3.1.1 Map Layout

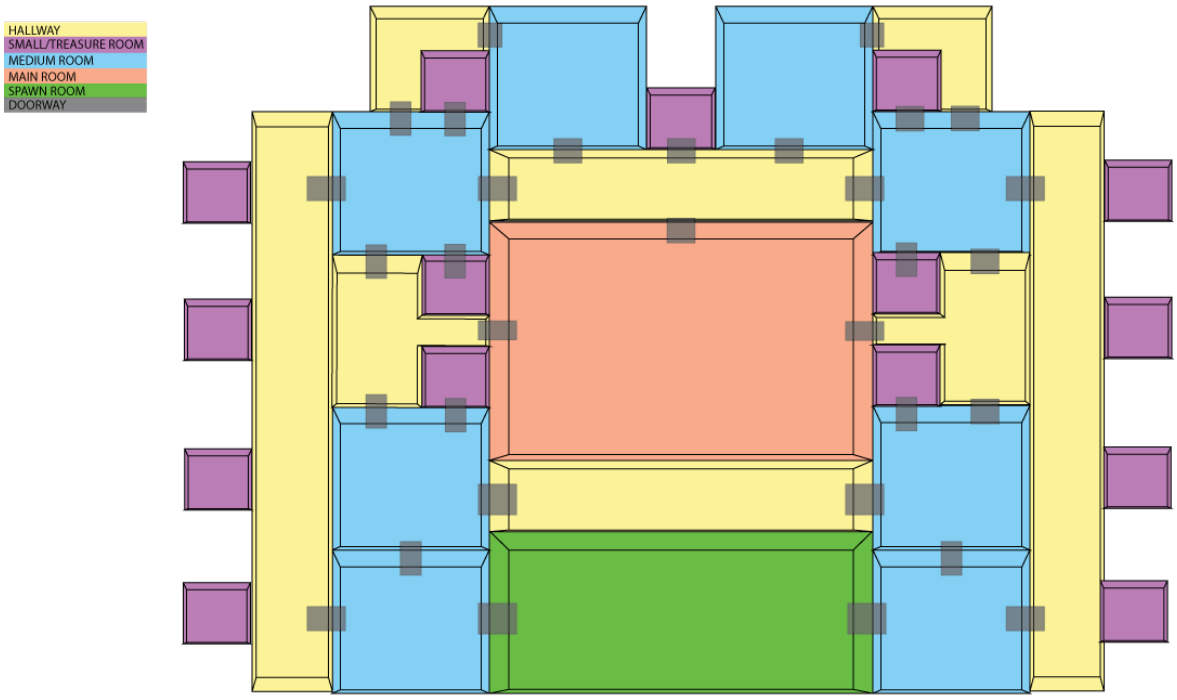


**Figure 3:** *The map layout. Treasure Rooms will spawn items, and player avatars will spawn in the Spawn Room. The gray door areas should let avatars walk into another room. See R2.1 and R2.2.*

**R1.1 Map**

R1.1.1 Players will control avatars that move around a virtual map. That virtual map will look similar to Figure 2. **Priority 1**

R1.1.2 Avatars will not be able to walk through each other or walls. Avatars may only walk through the designated door areas to move to another room. See top down games like “The

Legend of Zelda: Triforce Heroes” for basic movement. **Priority 1**

R1.1.3 Avatars will be able to walk through above sections marked doorway. **Priority 1**

R1.1.4 If multiple maps have been provided by the server, then Players will be able to select any of the available maps before a round starts. **Priority 3**

### 3.1.2 Content Generation

**R2.1 Player Spawn**

R2.1.1 When a player connects to the server, an avatar should automatically be spawned in the starting arena. The maximum number of unique players that can join will be 10. **Priority 1**

R2.1.2 The player should be assigned a Role. The appearance of the player’s avatar will reflect their role. (See R2.3) **Priority 2**

R2.1.3 The player should be assigned an Objective that is displayable only on their controller. **Priority 1**

R2.1.4 The player should be assigned a unique color. The player’s avatar will have this color, that color will be displayed on the mobile phone Main Screen, and as an identifier on the Round End Screen. **Priority 1**

R2.1.5 The player should be assigned 5 health points upon spawn. If a player respawns after being Knocked Out, he will have his health points reset back to 5. The maximum health a player can have is 5. (See 3.1.3 for how a player can lose health). **Priority 1**

**R2.2 Item Generation**

R2.2.1 When the game starts, 4 items should be spawned randomly by the server. An item is spawned randomly by randomly selecting a room and a random item, then placing the item in the center of the room. **Priority 2**

R2.2.2 Items on the map are all represented by the same asset such that all item boxes are indistinguishable. **Priority 2**

R2.2.3 When an item is dropped by a player (on the map) or displayed in their inventory (on mobile), the item's unique image will be used. **Priority 1**

R2.2.4 No more than 1 item can be spawned in a room at any time. **Priority 2**

R2.2.5 Every 1 minute 3 new items are spawned randomly in rooms that do not contain items. If all rooms already contain items, no more items can be spawned. Multiples of the same item can be spawned on the map at once (See Golden Orb and Gem exceptions in R2.2.7). **Priority 2**

R2.2.6 The items Golden Orb and Gem are spawned randomly at the start of the game. Each is spawned in the center of different random rooms that do not already have an item already present. **Priority 1**

R2.2.7 The Golden Orb and Gem cannot be spawned again randomly. If a player who has the Golden Orb or Gem disconnects from the server then the item will immediately be dropped on the board near where the player disconnected. **Priority 1**

**R2.3 Player Role**

| Requirement | Role | Description | Priority |
|---|---|---|---|
| R2.3.1 | Villain | More likely to get "Kill | 2 |

| | | | |
|---|---|---|---|
| | | the Hero” Objective (See R4.2.7). | |
| R2.3.2 | Scientist | More likely to get “Use Item on Player” Objective (See R4.2.6). | 2 |
| R2.3.4 | Gangster | More likely to get “KO Other Player” Objective (See R4.2.2). | 2 |
| R2.3.5 | Merchant | More likely to get “Pick up Many Items” Objective (See R4.2.3). | 2 |
| R2.3.6 | Archaeologist | More likely to get “Bring Item to Location” Objective (See R4.2.4). | 2 |
| R2.3.7 | Hero | More likely to get “Kill the Villain” Objective (See R4.2.8). | 2 |

**Table 1:** *Avatar Roles and their descriptions*

Upon entering the map, each player should be assigned a role. The role will determine which hidden objective the player will need to complete. See *Table 1* for the list of different roles and responding objective requirement.

### 3.1.3 Map Interactions

The avatars on the map (See R1.1.1) may interact with each other through combat. A player will be able to attack another player with an item

and damage that player. When a player is on the map, they can be in various states depending on whether they are attacking, are attacked, or recovering from an attack.

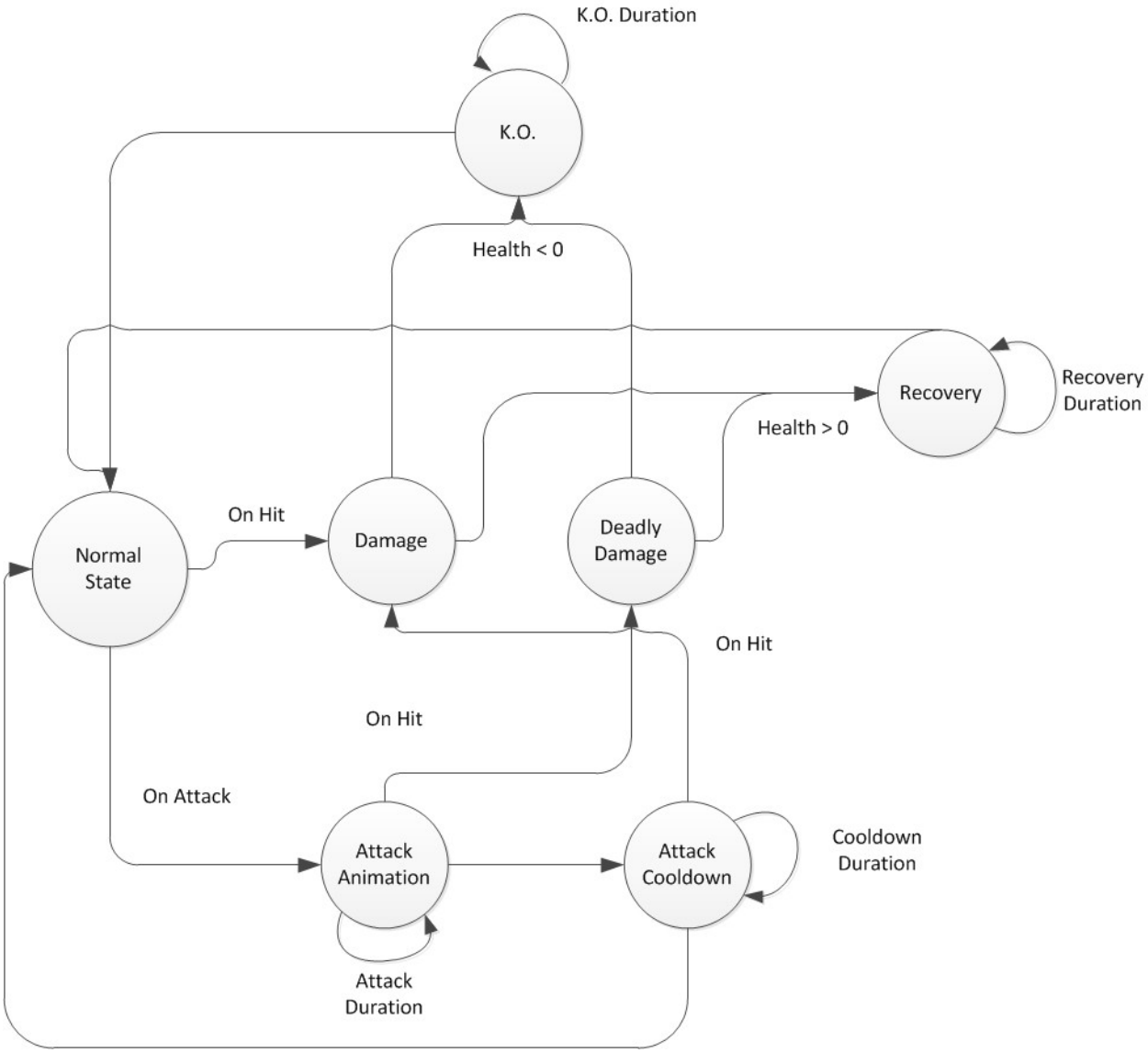


**Figure 4:** *State diagram and transitions for player combat*

As shown in *Figure 4*, all players start in the “Normal State.” Depending on whether a player is hit or is attacking they may transition to another state. The K.O. state (for “Knock Out”) is when a player loses all their health. The state descriptions are outlined in the *Table 2* below.

**R3.1 Combat State Requirements**

| Requirement | Player State | Description | Priority |
|---|---|---|---|
| R3.1.1 | Normal | Avatars should start in the Normal state. The player can move their avatar, attack, pick up and use items in this state. | 1 |
| R3.1.2 | Attack Animation | The player cannot move their avatar or use the attack button. Avatar goes into the attack animation in this state.<br>If players are hit while in the Attack Animation state, they move to Deadly Damage state. | 1 |
| R3.1.4 | Attack Cooldown | The player can move their avatar, but cannot use the attack button. Avatars can pick up items in this state. | 1 |
| R3.1.5 | Damage | Avatar loses 1 health and is not able to attack. Avatar goes into hurt animation. | 1 |

| | | | |
|---|---|---|---|
| R3.1.6 | Deadly Damage | Avatar loses 2 health and is not able to attack. Avatar goes into hurt animation. | 2 |
| R3.1.7 | K.O. | Avatar cannot move, drops all items, and is respawned at the spawn point. Avatar goes into K.O. animation. | 1 |
| R3.1.8 | Recovery | Player can move avatar, but cannot attack or pickup items. Avatars cannot take damage in this state. Avatars will flash while in this state. | 1 |
| R3.1.9 | Item Cooldown | Player can not use items after using an item, for a short time. | 3 |

**Table 2:** *Description of the different avatar states*

### 3.1.4 Player Objectives

**R4.1 Player Objectives Requirements**

R4.1.1 Players should be assigned objectives at the beginning of the game. **Priority 1**

R4.1.2 Players should be able to complete objectives by performing various actions. **Priority 1**

R4.1.3 Players should receive new objectives when older objectives are completed. **Priority 2**

R4.1.4 Players will receive points when older objectives are completed. **Priority 2**

R4.1.5 Objectives will give a set amount of points to the player, based on a score value specific to each Objective. **Priority 2**

**R4.2 Player Objectives to be Completed**

| Requirement Number | How does a player complete the objective? | Score | Priority |
|---|---|---|---|
| R4.2.1 | The player must hold the Golden Orb in their inventory for as long as possible. The player who holds the Golden Orb longest at end of round scores this objective. | 5 | 1 |
| R4.2.2 | The player must K.O. another specific player by combat use or item use. | 2 | 1 |
| R4.2.3 | The player must have picked up the most unique items in their inventory. | 3 | 2 |
| R4.2.4 | The player must get X item, drop it off at Y room. | 1 | 2 |
| R4.2.6 | The player must use a certain item on another player. | 2 | 2 |
| R4.2.7 | The player must Knock Out the player whose role is Hero. | 3 | 2 |
| R4.2.8 | The player must Knock Out the player whose role is Villain. | 3 | 2 |

**Table 3:** *The Hidden Objectives. These objectives will be assigned to player and will be displayed only to them on their mobile controller*

Each of the above objectives in *Table 3* will have a name and description. This is generic and will be provided by the client. Listed in the table is how a player will be able to complete the objective - detecting how an objective is completed will require unique code.

### 3.1.5 Items

**R5.1 Items Requirements**

R5.1.1 Players should have space for 4 items in their inventory. If another item of the same type is picked up by the player, it will use another slot in their inventory. **Priority 1**

R5.1.2 Players should be able to pick up items when their inventory is not full and players may have instances of the same item. **Priority 1**

R5.1.3 Players should be able to use items by pressing the corresponding item on their mobile controller. **Priority 1**

R5.1.4 Players should not be able to use items while in the Recovering state (See R3.1). **Priority 2**

R5.1.5 Once used, most items are consumed, removing them from the player's inventory. Whether items are single use or not is displayed in *Table R5.2*. **Priority 2**

**R5.2 Items and Effects**

| Requirement Number | Item and Description | Uses | Priority |
|---|---|---|---|
| R5.2.1 | Golden Orb - The golden orb should make the player glow when it is in their inventory. | None | 1 |
| R5.2.2 | Dynamite - Upon activating dynamite, it should appear on the map in the direction the player is facing. After 2 seconds, the | One | 1 |

| | | | |
|---|---|---|---|
| | dynamite should explode with an expanding radius for 2 seconds. A player that touches any point of this radius will lose 5 health. | | |
| R5.2.3 | Whip - When the player presses his attack button, the Attack Animation should take place in the direction they are facing. If another player is hit by the whip, they shall lose 1 health, be placed in stun status, and drop an item at random. The whip is not part of the inventory and can never be dropped or picked up. | Infinite | 1 |
| R5.2.4 | Gem - No use, only referenced by objectives. | None | 2 |
| R5.2.5 | Basket of Snakes - Upon using the item, it should spawn a snake avatar on the board in the direction the player is facing. | One | 3 |
| R5.2.6 | Teleporter - Upon using the item, the player should instantly transport forward in the direction he is facing. Given the distance between the player and the next wall, the player should transport randomly between 50-100% of the distance. | One | 3 |
| R5.2.7 | Monkey - Upon using the item, a monkey sprite is spawned on the map next to the player. The monkey sprite then moves towards the closest other player at twice the speed of normal player movement. An item at random is taken from the player and then added to the inventory of the player using the monkey item. | One | 2 |

| | | | |
|---|---|---|---|
| R5.2.8 | Harmful Teleport - Upon using the item, a large circle the size of a player will be placed at the user's location. The teleport will remain on screen for 10 seconds. When any player comes in contact with the teleport, they will be placed in the center of a random room on the map excluding the current room. | One | 3 |
| R5.2.9 | Trap Placement - Upon using the item, a item box will appear in the current player's location. This trap will have the same appearance of an item box on the map. When a player comes in contact with the item box, the player will lose 1 health and enter the stunned state. | One | 2 |

**Table 4:** *The possible items a player may pick up on the map and their corresponding effects.*

### 3.1.6 Environment Objects

**R5.3 Environment Requirements**

R5.3.1 Players should be able to trigger environment objects through various actions. **Priority 2**

R5.3.2 Players should be affected by the environment in various ways - typically when their avatar is near the object. **Priority 2**

R5.3.3 Environment objects have a set location in the maze. The set of environment objects may be distinct and locations may vary in different maps. **Priority 3**

**R5.4 Interactable Items in the Environment**

| Requirement | Object | Priority |
|---|---|---|

| Number | | |
|---|---|---|
| R5.4.1 | Environment Switch - A switch may be moved from “on” to “off” state by hitting with a whip. Upon triggering, the switch may trigger either a Closing Wall trap (See R5.4.6), release of a snake (See R5.4.2), Boulder rolling (See R5.4.4), or a release of a mummy (See R5.4.3). The switch will remain in the off state for 30 seconds after it has been triggered. | 2 |
| R5.4.2 | Snake - A snake should advance in a single direction until it reaches a wall, and then turn in the opposite direction and continue. If a player comes in contact with a snake, they should lose 1 health and enter the stun state. After coming in contact with a player the snake will disappear. If a snake is spawned by an Environment Switch then it will face a random compass direction (North, South, East, West). | 2 |
| R5.4.3 | Mummy - A mummy should advance in a single direction which it is facing. Every 3 seconds the mummy should randomly change direction (compass direction). If a player comes in contact with a mummy, they should lose 1 health and enter the stun state. After coming in contact with a player the mummy will disappear. | 2 |
| R5.4.4 | Boulder - Upon triggering of an Environment Switch, a boulder will spawn in the room of the switch. The boulder will start moving after 1 | 2 |

| | | |
|---|---|---|
| | second in the direction of the switch. The boulder will accelerate until it comes in contact with another wall. Any player that comes in contact with the boulder will lose 2 health. | |
| R5.4.5 | Pit Trap - Upon triggering, a section of floor on the map will visually “fall out”. Avatars standing on this section of floor will lose 2 health and enter the stunned state. | 2 |
| R5.4.6 | Closing Wall - Upon triggering by a switch, all four walls of a room will both move towards the center. Upon touching in the center of the room, the walls will retract and move back to their original positions. Any avatar that comes in contact with two opposing walls in the center will lose 5 health. A player will not be able to move while both walls are touching them. | 3 |
| R5.4.7 | Choice Switch - The switch initially will be in the “middle position”. When hit with a whip, the switch will move in either the left or right direction depending on the angle of the hit. Hitting in one direction will give the player an item in his inventory, while hitting in the other direction will take away 1 health. A sign in a different room will indicate which direction will reward them with an item. | 2 |
| R5.4.8 | Sign - Upon an avatar coming into contact with a sign, a message will be displayed in their Information Panel on their controller. This message will only be displayed while the | 2 |

| | | |
|---|---|---|
| | player is in contact with the sign. The sign will contain information on what action a certain switch corresponds to. | |

**Table 5:** *The list of interactable objects in the environment.*

### 3.1.7 Gameplay

**R6.1 Round**

R6.1.1 A Round starts upon entering the Main Screen on the server. A Round is a single game session and should last for 5 minutes. All state is reset at the beginning of each round. **Priority 1**

R.6.1.2. At the end of a round, gameplay stops, all mobile controllers are moved to the Round End Screen, and the server moves to the Round End Screen. **Priority 2**

# 4. Non-Functional Requirements

## 4.1 Network Performance

**R7.1 Lag Management**

There should be little to no lag between frames, devices, or inputs within the game. Once the programming is running properly with the recommended number of players simultaneously, the team is assuming the lag will be small and will not negatively affect the game. There are currently issues with high amounts of lag between individual frames. Some issues have been solved, and it is a constant priority to reduce the lag between the phone and the computer screen. In order to ensure game quality, the game will be

playtested (See 4.5) before it's released and we'll survey the playtesters for acceptable quality. **Priority 1**

## 4.2 Host Operating System Requirements

**R8.1 Server Support**

The game is expected to support Windows 7,8, and 10, along with Mac OS 10.X for the computer build to support the game itself. Other Operating Systems will not be tested, so they will not work with Remote Raiders. **Priority 1**

## 4.3 Mobile Operating System Requirements

**R9.1 Android Support**

Mobile build of the game will be tested on Android 5.0 and above. Other phones will not be tested and are not expected to work with Remote Raiders. **Priority 1**

**R9.2 iOS Support**

Mobile build of the game will be tested on iPhone5 and above. Other phones will not be tested and are not expected to work with Remote Raiders. **Priority 3**

## 4.4 Accessibility

**R10.1 Server Download**

The Remote Raiders server application will be available to download to players. **Priority 1**

**R10.2 Client Download Instructions**

The Remote Raiders server application will include instructions on how to download the client application for phones. **Priority 2**

**R10.3 Client Download**

The Remote Raiders client application will be downloadable from the GooglePlay Store. **Priority 1**

**R10.4 Small File Size**

The file size of both applications will be kept small to decrease download times. **Priority 2**

## 4.5 Playtesting

At the end of each build for a prototype, the game will be playtested with approximately 10 college-aged players unfamiliar with the game. The playtesting will be used to ensure the players face little difficulty with the controls and understanding of the game. The playtesting will also let us ensure the quality of the game is adequate (See R7.1). The game will be playtested towards the each of each Drexel Quarter (Fall, Winter, Spring). After each playtesting session the group will have each playtester fill out a form, and these forms will be reviewed by the team and the stakeholders.

# 5. User Interface

## 5.1 Server

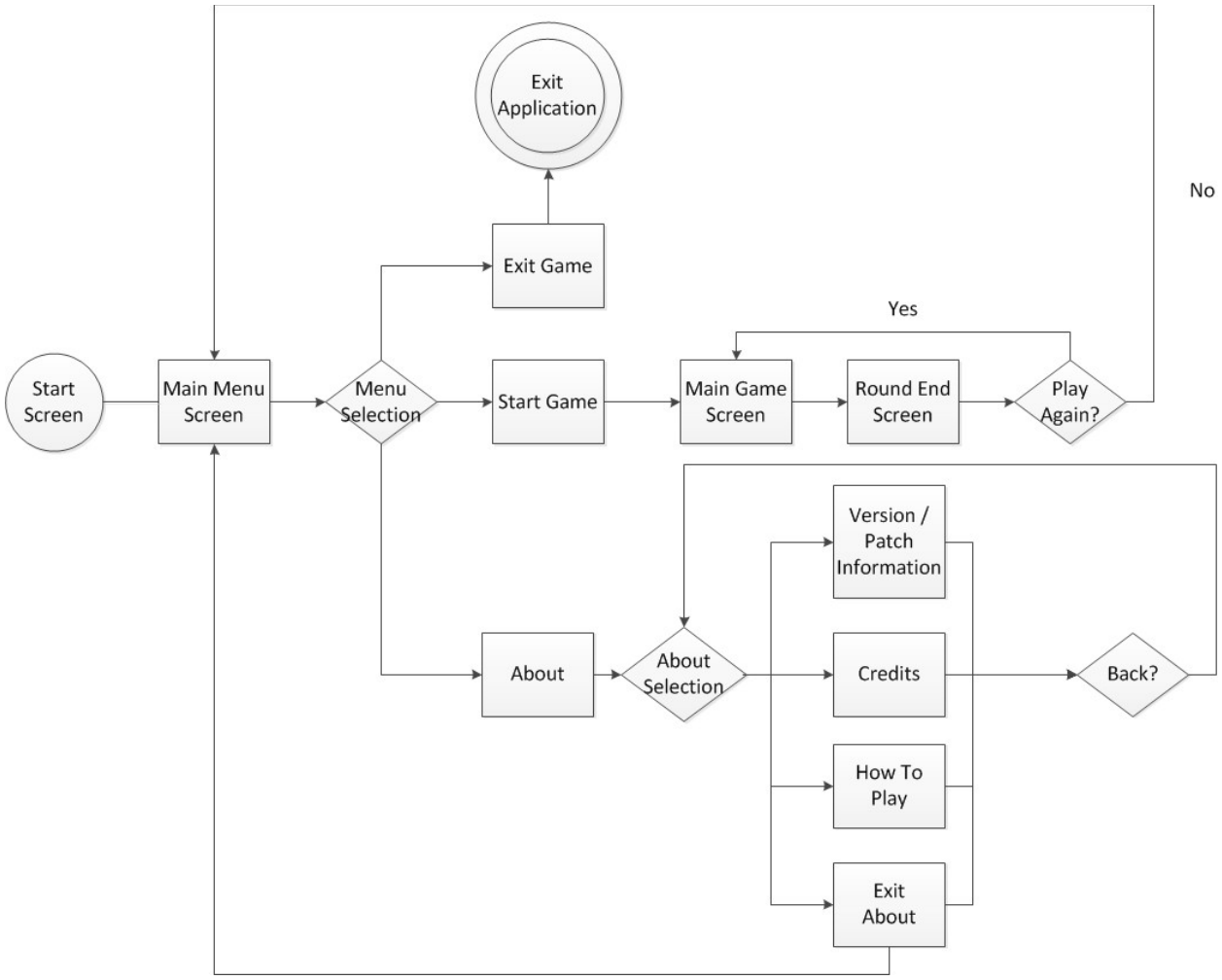


**Figure 5:** *Navigating menu screens on the server.*

The server will contain a main menu screen that will let the players start the game, exit the application, or enter the About section to learn more about the game. *Figure 5* shows the flow in navigating screens for the server. Below the requirements are outlined for each screen.

### 5.1.1 Start Screen

R11.1 Splash screen with game title and press enter prompt. Pressing enter will display Main Menu Screen **Priority 3**

### 5.1.2 Main Menu Screen

The Title Screen will feature three buttons: Start Game, Exit Game, and About.

R12.1 Start Game button will start a game instance and display the Main Game Screen. **Priority 1**

R12.2 Exit Game button will exit the game application. **Priority 1**

R12.3 About button will display About Screen. **Priority 2**

### 5.1.3 Main Game Screen

The majority of the screen is taken up by the maze - a virtual 2D space where the avatars can move around. See Map Layout section (See R1.1).

R13.1 IP Address of server game timer that counts down to the end of the game round are displayed in a header bar at the top of the screen. **Priority 1**

R13.2 Once the timer of 5 minutes ends the Round End Screen will be displayed. **Priority 1**

### 5.1.4 About Screen

About screen has four buttons: How To Play, Version and Patch Notes, Credits, and Return to Main Menu.

R14.1 How To Play button displays instructions for playing game **Priority 2**

R14.2 Version and Patch Notes button displays game version number and latest patch changelog **Priority 3**

R14.3 Credits button displays game credits **Priority 2**

R14.4 Return to Main Menu button displays the Main Menu Screen **Priority 2**

### 5.1.5 Round End Screen

R15.1 Display how many objectives each player has completed. Relating to other games, this is essentially a score and indicates who “won”. **Priority 1**

R15.2 Show each player that is ready to play again. **Priority 1**

R15.3 Display timer that counts down to the start of the next game round. **Priority 1**

R15.4 Display IP Address to allow non-connected players to connect to server. **Priority 1**

R15.5 Quit Game button ends game instance and displays Title Screen **Priority 1**

### 5.1.6 Credits Screen

R25.1 Display all the members of the team and their roles on a one screen list. **Priority 1**

R25.2 There will be a back button to return to the main screen. **Priority 1**

## 5.2 Mobile Controller Screen

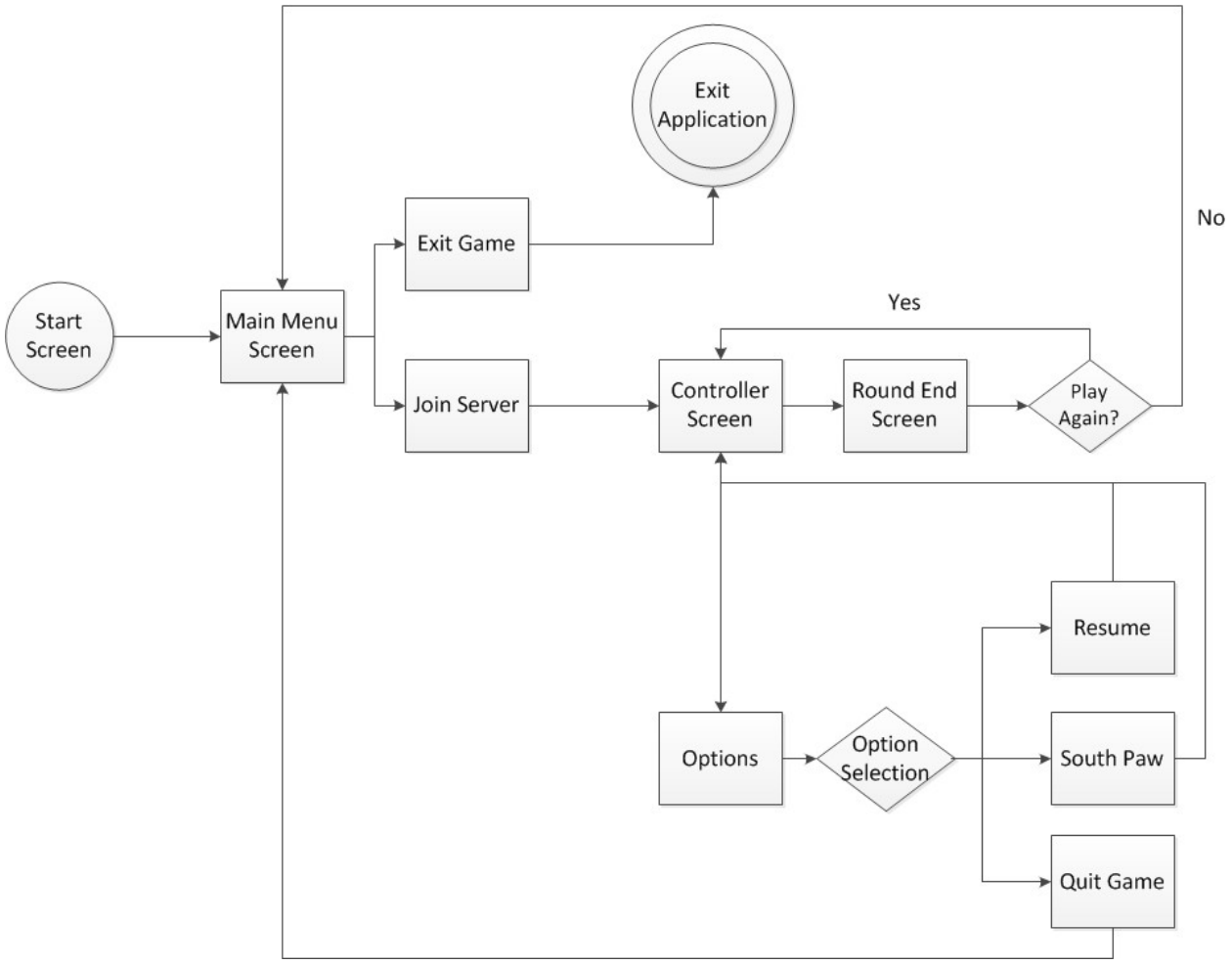


**Figure 6:** *The flowchart for navigating menus on the client.*

The mobile controller will contain a Main Menu screen. From this menu the player may either exit the game or join a server to begin playing the game. While in game, the user will have an Options screen to configure settings. *Figure 6* shows the flow in navigating screens for the mobile controller. Below the requirements are outlined for each screen.

### 5.2.1 Main Screen


**Figure 7:** *The joystick controller layout for the mobile controller.*

The main screen on the mobile controller is what the player will use to control his avatar. The screen should contain a joystick (See *Figure 7*), the player's avatar, the player's inventory, an attack button, and the player's description of their objective. There should also be an settings button in the upper right-hand corner.

### 5.2.2 Joystick

R16.1 The Joystick will be able to move in any direction by moving their finger from the center of the joystick in any direction. **Priority 1**

R16.2 The Joystick will let players move their avatar in the same direction as the Joystick. **Priority 1**

### 5.2.3 Inventory

R17.1 The player will be able to press buttons to use items **Priority 1**

### 5.2.4 Display Box

R18.1 The player will have a text panel that will display their objective. This text panel will be collapsible. **Priority 1**

### 5.2.5 Drop Item Button

R19.1 To drop an item, the player will drag the item to the drop button **Priority 2**

### 5.2.6 Help Button

R20.1 To view information about an item, the player will drag the item to the help square. There would be a visible square on the phone screen with a question mark. Upon an item being dragged and  touching this icon it will display information about the item in the Display Box. The information will only be displayed in the Display Box for as long as they are holding it there. **Priority 3**

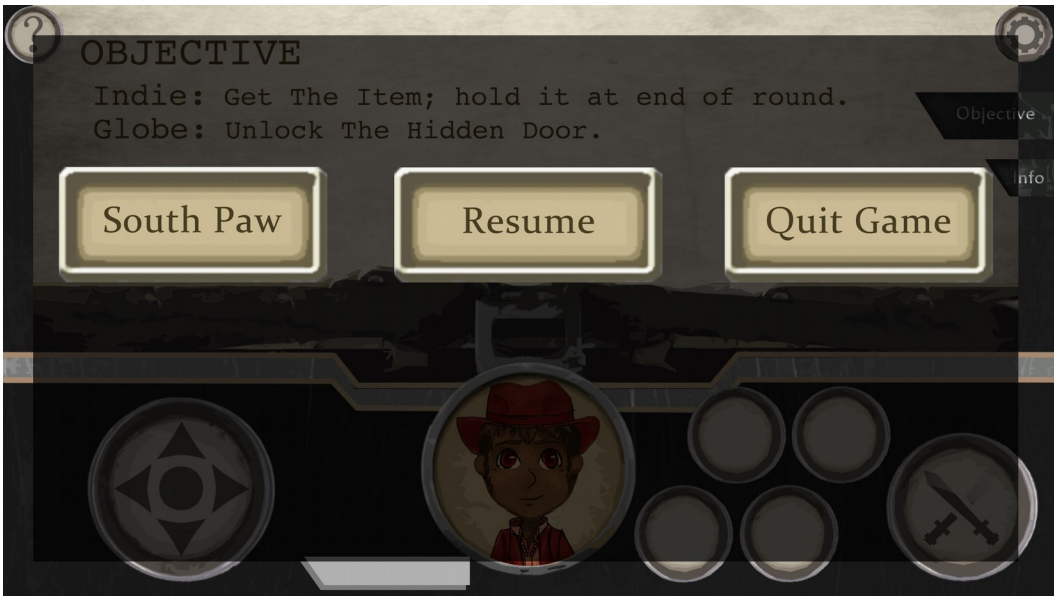


**Figure 8:** *The options menu for the mobile controller.*

### 5.2.7 Options Menu

R21.1 Clicking the Gear icon from the main screen will bring up the Options Menu (See Figure 8). **Priority 1**

R21.2 Tapping Quit Game disconnects the player from the game. A disconnected player's avatar will enter the KO state and wait to be reconnected. The mobile menu will switch to the Title Screen.

**Priority 2**

R21.3 Tapping South Paw will allow the joystick to be readjusted (switched to the right side of the screen) for left handed players.

**Priority 3**

### 5.2.8 Title Screen

R22.1 Players will need to enter the IP Address of a game they wish to join. After typing the correct IP Address and pressing the

“Join Game” button they will get an avatar in the game and their phone will move to the Main Screen. **Priority 1**
R22.2 Instead of entering an IP address, a list of servers on the same network will be displayed. Then simply tapping a local server will allow a player to join that game. **Priority 3**

#### 5.2.9 Round End Screen

R23.1 At the end of a round, the Round End Screen will be displayed. **Priority 2**
R23.2 At the end of the timer on the server Round End Screen the player will enter the Main Game Screen with a new Avatar. **Priority 2**
R23.3 On tapping the “Quit Game” they will enter the Main Menu Screen and disconnect from the server. **Priority 2**

#### 5.2.10 Health Display

R24.1 A Player’s health will be presented in the form of red hearts on the phone screen. **Priority 1**
R24.2 As a player gets damaged, their heart count will decrease.  Damage can occur from multiple sources (See R3.1.5, R3.1.6, and R3.1.7**)**. **Priority 1**

# 6. Use Cases

## 6.1 Use Case Flow

### 6.1.1 Picking up an item

Walk over an item, pick it up.

**Precondition:** An item is on the map, and the player's inventory has atleast 1 vacant spot.
**Action:** Player walks over item
**Postcondition:** Item is no longer on the map and is in the player's inventory.

### 6.1.2 Using an item

Tap an item button to use it. The item will typically appear in the direction the player is facing. Some items will have no effect when touched and remain in inventory (See Table 4 for which items can be used).
**Precondition:** A useable item is in the player's inventory
**Action:** Player taps the button with the item to use
**Postcondition:** The item is no longer in the player's inventory. The item's use effect resolves on the map.

### 6.1.3 Taking damage

Various effects will cause a player to take damage. Examples include being hit by traps or other players' items.
**Precondition:** The player is in a normal or cooldown state, with at least one health remaining.
**Action:** The player suffers damage due to a trap or attack
**Postcondition:** Players will lose some amount of health; this will update the health UI on their phone. If the resulting damage would bring the player's health above zero, the player's avatar will enter the Recovery state (See R3.1.5). Otherwise the player is KOed, as seen in the following use case.

### 6.1.4 Getting KOed

**Precondition**: A player has 1 health, and is attacked or is hit by a trap

**Action:** A player just took damage and their health has reached zero.

**Postcondition**: The player will drop all their items and return to the spawn point. Their health is restored to their starting health (See R3.1.7).

### 6.1.5 Completing Objectives

**Precondition:** A player was assigned an objective

**Action:** A player has met the completion requirements for a specific objective.

**Postcondition:** A new objective will be assigned to the player to replace the completed objective (See R4.1.3).

### 6.1.6 Starting Game (Server)

**Precondition**: The user is on Title Screen

**Action:** The user clicks start server

**Postcondition**: The server will allow mobile clients to connect to the game, and a new game timer will appear on screen.

### 6.1.7 Starting Game (Mobile)

The client starts the application and is given an input to enter the IP address of the server.

**Precondition**: The client enters the IP address of the server in the input provided.

**Action:** The Client clicks the join game button

**Postcondition**: The client will have a direct connection to the server, and the server will have a visual representation of the client on its screen.

### 6.1.8 Ending and restarting game (from Server)

The game ends after 5 minutes have passed from entering the Main Game Screen on Server. Then the game enters the Round End Screen on the screen. After 1 minutes have passed from entering the Round End Screen the game will start a new instance of the Main Game Screen. All players still connected to the server will be placed in new game instance with new avatars.

**Precondition**: Game timer ends

**Action:** Players wait a few moments

**Postcondition:** The new instance of the Main Game Screen will have representations of all connected players and newly spawned items.

### 6.1.9 Restarting game (from Mobile)

The game ends after 5 minutes have passed from the beginning of the round, according to the Sever. The Client enters Round End Screen, with two buttons. One button to play again, and one button to exit game.

**Precondition**: The timer on the server ends

**Action:** Player chooses option to play again

**Postcondition** : The player is connected to the next game and the game starts anew.

### 6.1.10 Exiting Game(from Mobile)

**Precondition**: The timer on the server ends

**Action:** Player chooses option to exit game

**Postcondition:** The player exits from the current instance and goes to the Title Screen.

## 6.2 Activity Diagram

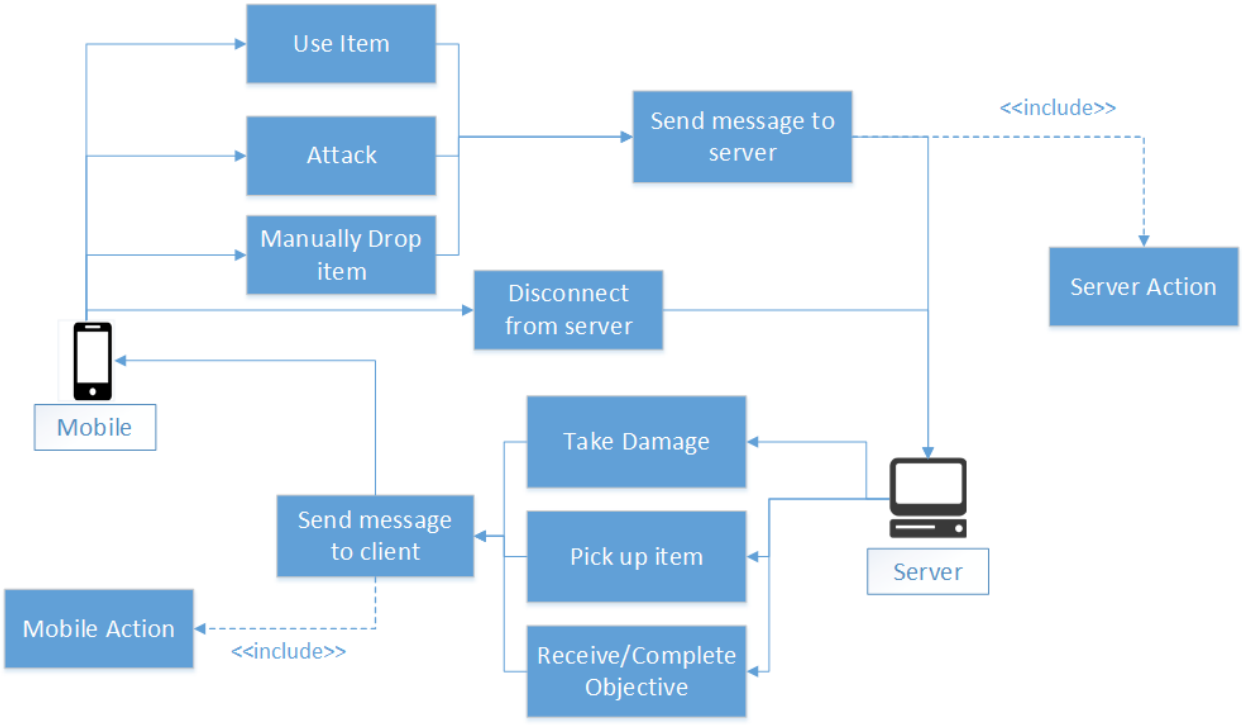


**Figure 9:** *The activity diagram for the use cases.*

# 7. Glossary

**Avatar** - The sprite that the player is controlling on-screen.

**Items** - Objects that can be collected by an onscreen avatar and used by the
player.

**K.O.** - The state when a player loses all of his health.

**Mobile** - Mobile controller of the game.

**Networking** - Tasks to enable communication between mobile clients and server.

**Objective** - Tasks randomly assigned to the Players to complete in-game.

**Remote Raiders** - Name of the game that is being made.

**Server** - Global map of all players connected to the game instance.

**South Paw** - Controller layout meant for a left-handed player.

**Unity** - Game development platform which will be used for development.

# 8. References

[1] http://ocremix.org/game/589/legend-of-zelda-four-swords-adventures-gcn