Requirements Specification and Management Plan

S1 Requirements specification Group D (WumboSoft Inc.)

Due: February 8th 2016

Jeff Werenka (100116489)

Cameron Ryan (100116969)

Sulian Shore (100114094)

Table Of Contents

[1. Introduction](#h.gjdgxs)

[1.1. Purpose](#h.30j0zll)

[1.2. Scope](#h.1fob9te)

[1.3. Definitions, Acronyms, and Abbreviations](#h.3znysh7)

[1.4. Project References](#h.2et92p0)

[1.5. Overview](#h.tyjcwt)

[2. Overall Description](#h.3dy6vkm)

[2.1. Product Perspective](#h.1t3h5sf)

[2.1.1. System Interfaces](#h.4d34og8)

[2.1.1.1 Concept of Operations](#h.2s8eyo1)

[2.1.2. User Interfaces](#h.17dp8vu)

[2.1.2.1 Main Menu](#h.3rdcrjn)

[2.1.2.2 Character Select Screen](#h.lnxbz9)

[2.1.2.3 Character Name Screen](#h.35nkun2)

[2.1.2.4. Main Game Screen Concept](#h.1ksv4uv)

[2.1.2.5. Pause Menu](#h.44sinio)

[2.1.2.6 Kittypedia](#h.2jxsxqh)

[2.1.2.7 Inventory Screen](#h.z337ya)

[2.1.2.8 Player Info Screen](#h.3j2qqm3)

[2.1.2.9 Save Screen](#h.1y810tw)

[2.1.2.10 Battle Screen](#h.4i7ojhp)

[2.1.3. Hardware Interfaces](#h.2xcytpi)

[2.1.4. Software Interfaces](#h.1ci93xb)

[2.1.5. Communications Interfaces](#h.3whwml4)

[2.1.6. Memory Constraints](#h.2bn6wsx)

[2.1.7. Operations](#h.qsh70q)

[2.1.8. Site Adaptation Requirements](#h.3as4poj)

[2.2. Product Functions](#h.1pxezwc)

[2.2.1 “Creating a Character” Use Case](#h.49x2ik5)

[2.2.2 “Moving to next area” Use Case](#h.147n2zr)

[2.2.3 “Obtaining Item” Use Case](#h.3o7alnk)

[2.2.4 “Using Item” Use Case](#h.23ckvvd)

[2.2.5 “Initiate Battle” Use Case](#h.ihv636)

[2.2.6 “Attacking” Use Case](#h.32hioqz)

[2.2.8 “Save” Use Case](#h.1hmsyys)

[2.2.9 “Load” Use Case](#h.41mghml)

[2.3. User Characteristics](#h.2grqrue)

[2.4. Constraints](#h.vx1227)

[2.5. Assumptions and Dependencies](#h.3fwokq0)

[2.6. Apportioning of Requirements](#h.1v1yuxt)

[2.6.1 Basic Functional Requirements](#h.4f1mdlm)

[2.6.1.1 Plot Requirements](#h.2u6wntf)

[2.6.1.2 Character Requirements:](#h.19c6y18)

[2.6.1.3 Item Requirements](#h.3tbugp1)

[2.6.2 Human Interface Requirements](#h.28h4qwu)

[2.6.3 Undesirable Characteristics](#h.nmf14n)

[2.6.4 Basic Non-functional Requirements](#h.37m2jsg)

[3. Specific Requirements (Sulian)](#h.1mrcu09)

[3.1 External interface requirements](#h.46r0co2)

[3.1.1 User interfaces](#h.2lwamvv)

[3.1.1.1 Title Screen](#h.111kx3o)

[3.1.1.2 Character Creation](#h.3l18frh)

[3.1.1.3 Movement Window](#h.206ipza)

[3.1.1.4 Inventory/Stats Window](#h.4k668n3)

[3.1.1.5 Battle Window](#h.2zbgiuw)

[3.1.2 Hardware Interfaces](#h.1egqt2p)

[3.1.3 Software Interfaces](#h.3ygebqi)

[3.1.4 Communication Interfaces](#h.2dlolyb)

[3.2 Classes/Objects](#h.sqyw64)

[3.2.1 Character](#h.3cqmetx)

[3.2.2 Kitty](#h.1rvwp1q)

[3.2.2.1 Status Points](#h.4bvk7pj)

[3.2.2.2 Attributes](#h.2r0uhxc)

[3.2.2.3 Experience](#h.1664s55)

[3.2.2.4 Level](#h.3q5sasy)

[3.2.3 Area](#h.25b2l0r)

[3.2.3.1 Home](#h.kgcv8k)

[3.2.3.2 Playground](#h.34g0dwd)

[3.2.3.3 School](#h.1jlao46)

[3.2.3.3.1 Classrooms](#h.43ky6rz)

[3.2.3.3.2 Elevators](#h.2iq8gzs)

[3.2.3.3.3 Roof](#h.xvir7l)

[3.2.3.3.4 Hallway](#h.3hv69ve)

[3.2.4 Encounter](#h.1x0gk37)

[3.2.4.1 Enemy](#h.4h042r0)

[3.2.4.2 Boss](#h.2w5ecyt)

[3.2.5 Item](#h.1baon6m)

[3.2.5.1 Warm Milk](#h.3vac5uf)

[3.2.5.2 Cat Nip](#h.2afmg28)

[3.2.5.3 Tuna](#h.pkwqa1)

[3.3 Performance Requirements](#h.39kk8xu)

[3.4 Delay Constraints](#h.1opuj5n)

[3.5 Software System Attributes](#h.48pi1tg)

[3.5.1 Reliability](#h.2nusc19)

[3.5.2 Security](#h.1302m92)

[3.5.3 Maintainability](#h.3mzq4wv)

[4. Supporting Information](#h.2250f4o)

[4.1. Management Plan](#h.haapch)

[4.1.1. Introduction](#h.319y80a)

[4.1.1.1. Project Overview](#h.1gf8i83)

[4.1.1.2 Project Deliverables](#h.40ew0vw)

[4.1.1.3 Evolution of the Project Management Plan](#h.2fk6b3p)

[4.1.1.4 Reference Materials](#h.upglbi)

[4.1.2 Project Organization](#h.3ep43zb)

[4.1.2.1 Process Model](#h.1tuee74)

[4.1.2.2 Organizational Structure](#h.4du1wux)

[4.1.2.3. Organizational Boundaries and Interfaces](#h.2szc72q)

[4.1.2.4 Project Responsibilities](#h.184mhaj)

[4.1.3 Managerial Process](#h.3s49zyc)

[4.1.3.1 Managerial Objectives and Priorities](#h.279ka65)

[4.1.3.2 Assumptions, Dependencies, and Constraints](#h.meukdy)

[4.1.3.3 Risk Management](#h.36ei31r)

[4.1.3.4 Monitoring and Controlling Mechanisms](#h.1ljsd9k)

[4.1.3.5 Staffing Plan](#h.45jfvxd)

[4.1.4 Technical Process](#h.2koq656)

[4.1.4.1 Methods, Tools, and Techniques](#h.zu0gcz)

[4.1.4.2 Software Documentation](#h.3jtnz0s)

[4.1.4.3 Project Support Functions](#h.1yyy98l)

[4.1.5 Work Packages, Schedule, and Budget](#h.4iylrwe)

[4.1.5.1 Work Packages](#h.2y3w247)

[4.1.5.2 Dependencies](#h.1d96cc0)

[4.1.5.3 Resource Requirements](#h.3x8tuzt)

[4.1.5.4 Budget and Resource Allocation](#h.2ce457m)

[4.1.5.5 Schedule](#h.rjefff)

[Appendix A- Sequence Diagrams for Use Cases](#h.3bj1y38)

*This document provides the requirements specification (S1 deliverable) from group D.*

# 1. Introduction

## 1.1. Purpose

The purpose of this game is to design a product that can be marketed to school children and parents. The funds raised from the sales of this game will go towards sending the HKES grade two co-ed kickball team to the nationals competition in Iqaluit. The school also plans to have an arcade style implementation of the game to be played by the children at recess and lunch break.

## 1.2. Scope

The scope of this project includes the development of a game application that will simulate a turn based combat system in RPG style. The game will be designed and implemented by our company Wumbo Corp and progress on the development can be found at our website ( <https://sites.google.com/site/wumbounivesity/home> )

The front end of this system will be a Graphical User Interface where the user will be able to point and click options presented to them in a menu based system, along with the keyboard ‘WASD’ movement. In addition, for compatibility with Arcade Machine systems, support for gamepads/joysticks will also be included.

The back end of this system will be implemented in either Python or Java and will be a stand alone application that can be distributed and installed on the machines in HKES.

## 1.3. Definitions, Acronyms, and Abbreviations

HKES - “Hello Kitty© Elementary School”

RPG - “Role Playing Game”

S1, S2 etc. - “Supplier deliverable 1, Supplier deliverable 2”

C1, C2 etc. - “Customer deliverable 1, Customer deliverable 2”

WASD Movement - Refers to the use of the W, A, S, and D keys on a keyboard to control a character on screen. Each key would provide specific directional movement, in their respective directions.

Kittypedia - This is a device in the game that will allow the player to view all of the kitties they’ve seen before

## 1.4. Project References

We will be using information given to us by Hello Kitty© Elementary School as reference for the construction of this project. All information provided to us will be available at <http://falcon.acadiau.ca/~110854m/COMP3663/> (~~WARNING: Until further notice, it is advised that if you suffer from epilepsy, do not visit this site, as it contains many flashing colours.~~). In addition to this, the requirements provided to us by HKES gives several other works as reference for this project. As a result, we will gathering inspiration, and influence from several other games/sources, including but not limited to the Pokémon RPG game series.

NOTE: For references used to create the project and documentation that is not limited to the game material itself, please see chapter 4.1.1.4

## 1.5. Overview

# 2. Overall Description

## 2.1. Product Perspective

### 2.1.1. System Interfaces

### 2.1.1.1 Concept of Operations

*Hello Kitty RPG*  can be in any of the following states:

* Initialize: The user creates a player and starts the game
* Momma’s House: Player can save, quit from this location, and move to the school
* School: Player must proceed through the school to win the game
* Waiting:
  + Player may save or quit
  + The Player can move to the next room or floor
* Encounter: Begins a battle, if the Player wins they continue, if not start over from the beginning

### 2.1.2. User Interfaces

#### 2.1.2.1 Main Menu

This is the main menu of Hello Kitty RPG. The player has the option to continue their game, start a new game, and quit the game. The Continue box is larger than the others, because it will display information regarding the player's progress through the game.



#### 

#### 2.1.2.2 Character Select Screen

When starting a new game, the play can select a character.



#### 2.1.2.3 Character Name Screen

After choosing their character, the play may enter their name.



#### 2.1.2.4. Main Game Screen Concept

This is the main screen the player will see as they play the game. Their character will be centered on the screen, and ~~the~~ they may move throughout the world and the school’s playground. At the bottom of the screen, text may show up, depending on whether the character is interacting with something in the world, or another character. Most times, this box at the bottom will not be visible.



#### 2.1.2.5. Pause Menu

When the player pauses the game, the pause menu will appear on the right side of the screen. This will give options to view their Kittypedia, Bag will show a screen containing information about the player's inventory items. The third category will be the player's name, and will give information regarding their progress through the game. Save will save the game, return will close the menu, and quit will allow the player to quit after reminding them to save the game.



#### 2.1.2.6 Kittypedia

The Kittypedia shows a list of all kitties, and displays whether or not the character has seen one, or owns one. If the character has not seen one, it will not show up in the list.



#### 2.1.2.7 Inventory Screen

This shows a list of items with a short description of what each item does.



#### 2.1.2.8 Player Info Screen

This shows information including Player Name, how many battles they’ve won (versus how many they’ve lost), how many kitties they’ve seen, their total playtime, when they created their file, and how many bosses they’ve beaten



#### 2.1.2.9 Save Screen

When the player saves the game, a menu pops up over the screen displaying information about the save.



#### 2.1.2.10 Battle Screen

When the player gets into a fight with another student or teacher, they fight using their kitties. This screen will give the player the option to attack, use their items, or run away from the battle. The interface will also show information regarding the kitties such as their health and their name.



### 2.1.3. Hardware Interfaces

A mouse and keyboard will be used, with gamepads as a ~~stretch goal~~ desired requirement.

### 2.1.4. Software Interfaces

This is used to save information regarding the game.

This includes areas as well as information regarding the Player’s character.

### 2.1.5. Communications Interfaces

None. This game is single-player offline only.

### 2.1.6. Memory Constraints

Memory constraints are predicted to not prove any issue in the course of this development. As both Java and Python (both are candidates for what we will be using to program this game) have automatic garbage collection, we will not use an amount of memory that is large enough to necessitate this constraint. In addition, this game will be able to run on systems with less than ~~four GB~~ 512 MB of RAM.

\*\*\* The game is required to run on the Acadia Advantage laptops, so no more than 512MB of RAM will be used. The game content will use no more than 50 MB of hard drive space.

### 2.1.7. Operations

The user will be able to save and load their game.

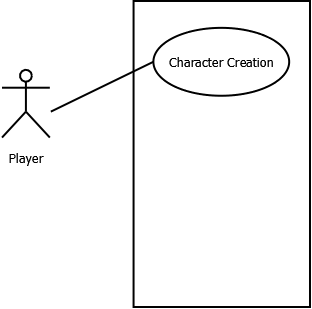
### ~~2.1.8. Site Adaptation Requirements~~

~~Hello Kitty Elementary School will have to adapt their site for the installation of arcade machines for this game.~~

## 2.2. Product Functions

This section gives a brief explanation of the game's functions.

### 2.2.1 “Creating a Character” Use Case

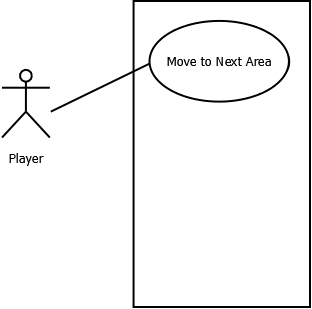


1. System displays three characters to choose from by clicking.
2. Player creates a name for the character by typing in the text field and accepting

.

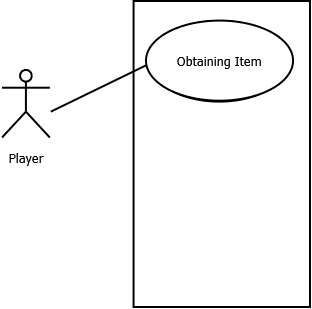
### 

### 2.2.2 “Moving to next area” Use Case



1. Player pushes an arrow key to move to the corresponding relational position.
2. System determines if the desired movement is legal, moves the player if so or refuses action otherwise
3. System initiates a battle when the Player walks into a room with a valid opponent
4. System moves the character to the corresponding room if the character’s position is the entry point to that room.
5. System updates the display to represent the new situation.

### 2.2.3 “Obtaining Item” Use Case

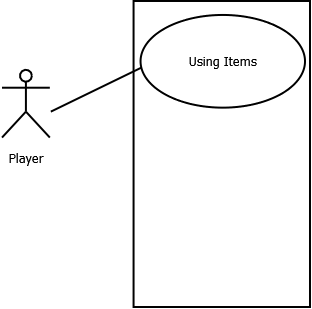


1. Items can be obtained either by talking to Mommy or winning an encounter
2. a. By talking to Mommy you can obtain a Warm Milk

b. After winning an encounter there is a chance you will obtain a random item.

3. Items are put into the player’s inventory

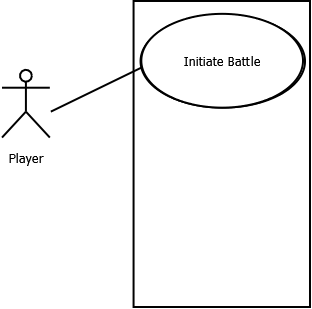
### 2.2.4 “Using Item” Use Case



1. Player chooses the bag option from either the pause menu, or the battle menu
2. System displays the player’s inventory menu
3. Player selects an item from their inventory
4. System displays information about the item as well as giving the player the option to use the item (Depending on whether the player is in/out of combat, the system may or may not display certain items)
5. Player chooses to use the item
6. The item effect is applied

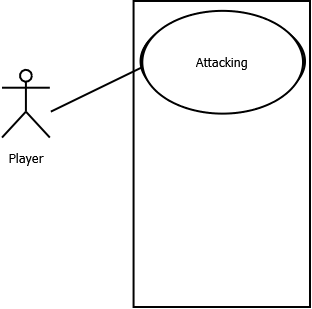
### 

### 2.2.5 “Initiate Battle” Use Case



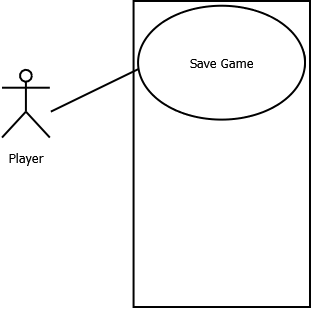
1. Player has entered a room
2. System switches to battle graphical representation mode.
3. Player chooses from available actions until battle completion.
4. System displays the outcome (exp or items gained)
5. System switches back to movement representation mode.

### 2.2.6 “Attacking” Use Case



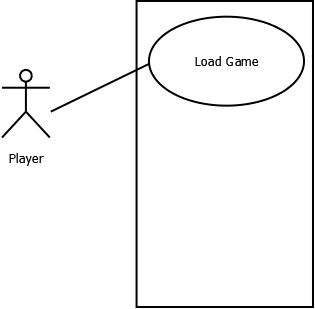
1. Player chooses an Attack to use
2. System calculates damage based on Attack, Strength, enemy Defense
3. System determines if the attack is successful.
4. System informs the Player of the result.
5. Enemy attacks the user, using the same system.

### 2.2.8 “Save” Use Case



1. Player chooses the ‘Save’ option from the pause menu
2. System verifies that the player wishes to save the game
3. Player chooses ‘Yes’
4. System displays information about the save file, and communicates that the game has been successfully saved.

### 2.2.9 “Load” Use Case



1. Player chooses the ‘Load Game’ option from the main menu.
2. System displays the current existing save files
3. Player chooses the save file they wish to use
4. System returns to main menu
5. System displays the currently loaded save file in the ‘Continue’ section of the main menu

## 2.3. User Characteristics

This game is aimed toward young school children (Grades K - 6) and their parents.

## 2.4. Constraints

Deliverables must be received on time. No deliverable can be accepted before a previous deliverable has been received and approved.

The entire project must cost no more than 35 man-days (at 8 hours/day).

Must run on ***any*** Acadia Advantage laptop ~~under~~ newer than Windows XP.

Implemented in Java or Python.

Can be downloaded and installed from a web homepage as an executable.

## 2.5. Assumptions and Dependencies

* System assumes save files are valid and have not been modified.

## 2.6. Apportioning of Requirements

### 2.6.1 Basic Functional Requirements

F1- Each play of the game has a limited time frame. The player can set this.

F2- Provides a school with interconnected areas.

F3- There is one area from which the game begins (The Playground)

F4- Each area must have a name, a unique graphical image and its own set of no less than 3 environmental variables such as the number of monsters in the room, and type of monsters, the items in the room, etc

F5- Each area has connections to one or more other areas.

F6- Areas contain people and items that can be interacted with.

F7- Must allow the creation and maintenance of various game characters. Each character has a unique name and image and a set of qualities such as life points, strength, defense (optional), attack rating, and attack power that can be assigned points by the player outside of battle. The system must allow the player to create these characters, define the qualities and select a character image.

F8- No more than two characters can be at the same location in the room at one time

F9- When two characters meet at a position in the room they battle.

F10- Once battle is complete the main character, if s/he still exists and has the remaining qualities to do so, can move to a new location based on the player’s choice of direction. Moves deplete resources – so characters cannot simply just move for ever without having their qualities replenished in one form or another

F11- Must allow users to query for (and potentially print) a report of current character and area status by showing all quality and environmental variable values

F12- Must provide a method of saving and restoring the characters as a set.

F13- Should have a method of saving and restoring the game state so that a game can be stopped and started at a later time.

##### 2.6.1.1 Plot Requirements

F14- *Hello Kitty RPG* is a role playing game centered around battling Kitties.

F15- The Player character must train and battle with their Kitty to defeat classmates and teachers

F16- The Player character will have to proceed through many rooms and floors in the school

F17- In each room there will be a ~~classmate~~ teacher (or other enemy) with a Kitty to battle

F18- While moving through the school it is possible to find items that can be used later.

F19- The goal is to battle through the entire school to prove you are the best Kitty trainer

##### 2.6.1.2 Character Requirements:

F20- Character receives a Kitty at the beginning of the game

F21- Each Kitty will have stats such as:

* Health
* Strength
* Defense
* Endurance
* Dexterity
* Experience
* Level

##### 2.6.1.3 Item Requirements

F22- Catnip - Permanently increases Strength

F23- Warm Milk - Increases Health to max

F24- Can of Tuna - Permanently increases Defense

### 2.6.2 Human Interface Requirements

* Must be GUI based with efficient use of mouse point and click and menu features
* Must be user friendly by providing:
  + Intelligent display and sequencing of queries and menus
  + User oriented error and warning messages
  + Online and offline help information

### 2.6.3 Undesirable Characteristics

* Lengthy user manual - focus on great on-line help and reference documentation as well as innovative "getting started" tutorial. Ask yourself, how would I like to learn how to use the system.
* Need for external components such as backup utility or DBMS that might have to be purchased by your customer (these issues can be discussed with your customer and TA).

### 2.6.4 Basic Non-functional Requirements

* The game takes place in an elementary school and the player battles their kitty against other the kitties of teachers

# 3. Specific Requirements ~~(Sulian)~~

## 3.1 External interface requirements

### 3.1.1 User interfaces

The user interface will be represented by several graphical windows.

These windows include:

* Title screen
* Character selection
* Movement Window
* Inventory/Stats Window
* Battle Window

### 3.1.1.1 Title Screen

This window shows the options to create a new character, load a previously saved character, load the last played character or quit the game. Creation of a new character loads the character selection window. Loading a previously saved character loads the movement window. Pressing the quit button quits the game.

### 3.1.1.2 Character Creation

This window shows the selection of characters to choose from. Once a character is selected another window opens where you can enter your character’s name. Clicking the finished button will load the movement window and initialize the game.

### 3.1.1.3 Movement Window

This window will show the the layout of the area the character is in. This is where the character can move around. Upon selection of the menu button the Inventory/Stats window opens. Upon an encounter, the battle window is loaded.

### 3.1.1.4 Inventory/Stats Window

This window will show the stats of your kitties as well as any items you are carrying. From here you can also use items.

### 3.1.1.5 Battle Window

This window will show character and enemy graphics, as well as the status of both, and any actions that the character may take. A message will display the results of any actions and (optional) animations will also play. Upon victory, the movement window is loaded. On defeat, the title screen is loaded.

### 3.1.2 Hardware Interfaces

This game will support the use of a mouse and keyboard as input devices.

### 3.1.3 Software Interfaces

This game will require the Java 1.7 runtime environment ~~(Possible)~~

### 3.1.4 Communication Interfaces

Network connections will not be required for this game, as it is a single player offline game.

## 3.2 Classes/Objects

### 3.2.1 Character

This is who the player will be controlling throughout the game. They will get to choose of a small set at the beginning of the game.

### 3.2.2 Kitty

These are the the characters that the player and npc’s will use to battle.

#### 3.2.2.1 Status Points

Each kitty will start with a certain amount of status points that depend on its’ type.

#### 3.2.2.2 Attributes

Attributes are permanent status points which each kitty possesses. These depend on the type and level of the Kitty. These can be modified by leveling up or using certain items.

#### 3.2.2.3 Experience

If the player wins an encounter, their Kitty will gain a ~~set~~ number of experience points.

#### 3.2.2.4 Level

The level of the Kitty depends on the amount of experience points earned

### 3.2.3 Area

#### 3.2.3.1 Home

This is where the player can return in order to heal their Kitty or talk to Mommy.

#### 3.2.3.2 Playground

This is where the game starts and the player can have their first battles.

#### 3.2.3.3 School

##### 3.2.3.3.1 Classrooms

These are areas where the player will battle other Kitty owners.

##### 3.2.3.3.2 Elevators

These are areas that can transport the player between floors, including the roof, in the school.

##### 3.2.3.3.3 Roof

This is where the player will fight the final boss

##### 3.2.3.3.4 Hallway

The hallway is a connecter between the classrooms and the elevator.

### 3.2.4 Encounter

#### 3.2.4.1 Enemy

This is who the character will be fighting against in the school. They become more difficult as the player ascends floors.

#### 3.2.4.2 Boss

The final character the player will fight. Also the most difficult enemy. This will be a tasteful depiction of Danny Silver.

### 3.2.5 Item

#### 3.2.5.1 Warm Milk

Giving this to your Kitty will fully heal them.

#### 3.2.5.2 Cat Nip

Giving this to your Kitty will increase their strength.

#### 3.2.5.3 Tuna

Giving this to your Kitty will increase their defense.

## 3.3 Performance Requirements

Upon initialization the game should load and be ready to commence the game within 30 seconds. No action in the game should take longer than 10 seconds to process.

## 3.4 Delay Constraints

Hello Kitty RPG must be written to allow code reuse and extendability in the case the further updates are made.

## 3.5 Software System Attributes

### 3.5.1 Reliability

Hello Kitty RPG should not fail more than once in 100 complete plays of the game. There should be a friendly error message and restart on failure.

### 3.5.2 Security

There will be no password protected save files in this version of Hello Kitty RPG

### 3.5.3 Maintainability

All classes should be reusable for future updates to Hello Kitty RPG

# 4. Supporting Information

## 4.1. Management Plan

### 4.1.1. Introduction

#### 4.1.1.1. Project Overview

*Hello Kitty RPG* is a simple Roleplaying Game that has been commissioned by Hello Kitty Elementary School, and is being developed by WumboSoft Entertainment. The goal of this project is to develop a fun and enjoyable game for the school children who attend HKES, and their parents. The children will play characters who attend HKES and must battle their way through the school to become HKES’s best Kitty Trainer.

#### 4.1.1.2 Project Deliverables

Below is a table describing the deliverable portions of this project, and their respective due dates.

|  |  |  |
| --- | --- | --- |
| ***Deliverable*** | ***Description*** | ***Date*** |
| S1 | Requirements, Specifications, and Management Plan | February 8, 2016 |
| S2 | System Overview Presentation | March 2, 2016 |
| S3 | Detailed Design and Test Plan | TBD |
| S4 | User Manual | TBD |
| S5 | Project  Demo, Game and Documentation | TBD |
| S6 | Project Evaluation | TBD |

#### 4.1.1.3 Evolution of the Project Management Plan

This document will be maintained on a weekly basis by the Project Manager, Jeff Werenka. A direct link to this document will be provided, where changes may be viewed in real time via Google Docs. It is the responsibility of the project leader, and of WumboSoft to keep this document up to date. This document adheres to the format of IEEE 1058.1-1987.

#### 4.1.1.4 Reference Materials

For this project, we will be using reference materials as given by the instructor of COMP 3663, Danny Silver. This may include, but is not completely limited to the following:

* Braude, Eric J. SOFTWARE ENGINEERING: An Object Oriented Approach

NOTE: It is here that we would like to reinforce that this document may continue to be updated. If more material is referenced, we will be updating this section.

### 4.1.2 Project Organization

#### 4.1.2.1 Process Model

The development of this project will follow an iterative cycle. During the creation of each of the project deliverables, we will be revisiting the previous one(s) to mark improvements, and/or make any necessary changes. This will continue throughout the development process.

#### 4.1.2.2 Organizational Structure

The project roles are as follows:

Project Manager, Communications Representative, Lead Developer, Lead Designer/UI Specialist, Lead Requirements, Lead Documenter, Webmaster, Lead Quality Assurance.

These roles have been distributed as seen in the table below.

|  |  |  |
| --- | --- | --- |
| Cameron Ryan | Jeff Werenka | Sulian Shore |
| Lead Developer  Lead Designer/UI Specialist  Webmaster | Project Manager  Communications Representative | Lead Requirements  Lead Documenter  Lead Quality Assurance |

#### 4.1.2.3. Organizational Boundaries and Interfaces

The project team will interact with the course professor, Danny Silver, as well as the course’s TA, Stephen Isiuwe. Additionally, communication between WumboSoft and HKES will take place between each teams Communications Representative: Jeff Werenka, and Jimmy Flemming respectively.

#### 4.1.2.4 Project Responsibilities

Each team member will be responsible for ensuring the quality of each deliverable, and will participate in each deliverables completion. However, it will be the responsibility of the teams Webmaster, Jeff Werenka, to ensure that each deliverable is uploaded to the correct location, on time for each due date.

### 4.1.3 Managerial Process

#### 4.1.3.1 Managerial Objectives and Priorities

The main objective for this project is to ensure that the expectations of HKES, and any additional requirements given to us by them throughout the development process are met.

Additionally, it is the second priority of this project to ensure that each deliverable is submitted on time, and to ensure that each deliverable meets the standards of both WumboSoft, and HKES.

If all other objectives and priorities are met, it is an additional objective that any additional/optional features may be implemented as well.

#### 4.1.3.2 Assumptions, Dependencies, and Constraints

Java has been chosen as the primary programming language for *Hello Kitty RPG*, and as such the project will be dependent on and constrained by Java.

#### 4.1.3.3 Risk Management

There are several risks that have to be managed. These include:

Time and Project management

Meetings interrupted due to weather conditions

Other risks that have not yet been identified

#### 4.1.3.4 Monitoring and Controlling Mechanisms

Weekly team meets have been scheduled both prior to COMP 3663, from 5:45 pm - 6:45 pm on Mondays, as well as 3:00 pm on Fridays.

#### 4.1.3.5 Staffing Plan

More information on Staffing plans is detailed in section 4.1.2.2. Information given in this section does not limit the roles/responsibilities of each team member, as all members are expected to participate in the completion of each deliverable.

### 4.1.4 Technical Process

#### 4.1.4.1 Methods, Tools, and Techniques

For this project we will be using the most up-to-date version of Java for our main programming language. We will be following coding style as provided by Oracle. In addition to this, our code will be documented using javadoc.

#### 4.1.4.2 Software Documentation

All supporting documentation will be provided including the generated Javadoc, a user manual, and copies of the requirements analysis and design documents.

#### 4.1.4.3 Project Support Functions

None

### 4.1.5 Work Packages, Schedule, and Budget

#### 4.1.5.1 Work Packages

Work will be divided among our team members by the project manager with assistance from the leader for each deliverable. All members will participate in each deliverable and perform the tasks assigned to them.

#### 4.1.5.2 Dependencies

The progress of the project will depend on WumboSoft completing each deliverable on schedule as well as HKES reviewing the submitted deliverables in a timely manner.

#### 4.1.5.3 Resource Requirements

According to the provided outline by HKES we will require a team of three for this project. Each team member will require a laptop computer with specifications equal to or greater than that of an Acadia Advantage laptop.

#### 4.1.5.4 Budget and Resource Allocation

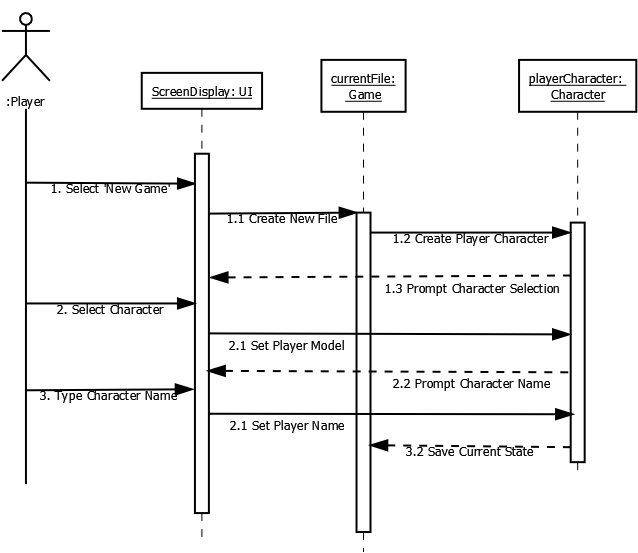
The budget for this project allows for 35 man days at 8 hours per day. This time will be spread amongst all team members equally.

#### 4.1.5.5 Schedule

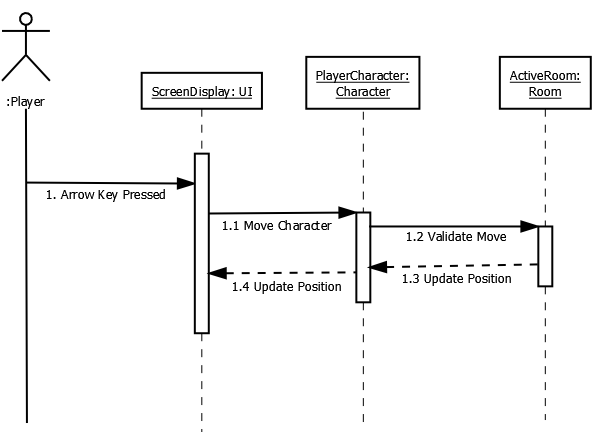
Please refer to section 4.1.1.2 for information on deliverables and their due dates.

# Appendix A- Sequence Diagrams for Use Cases

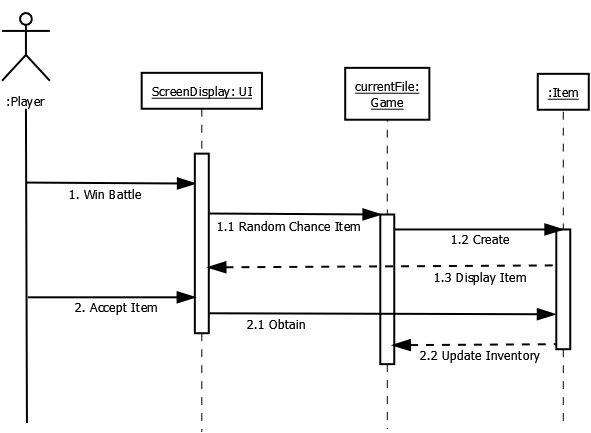
Character Creation



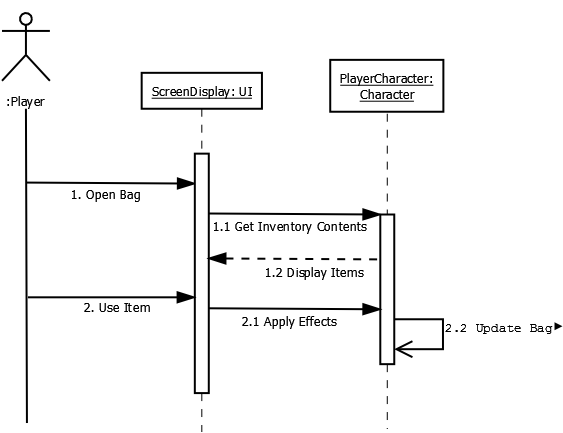
Move Character



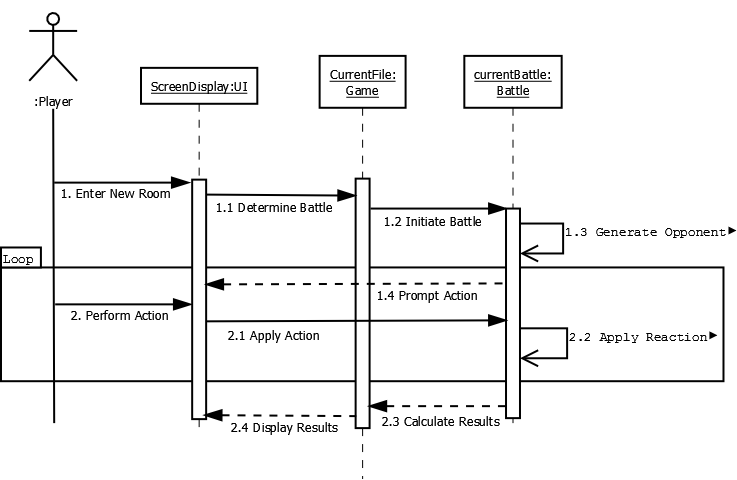
Obtain Item



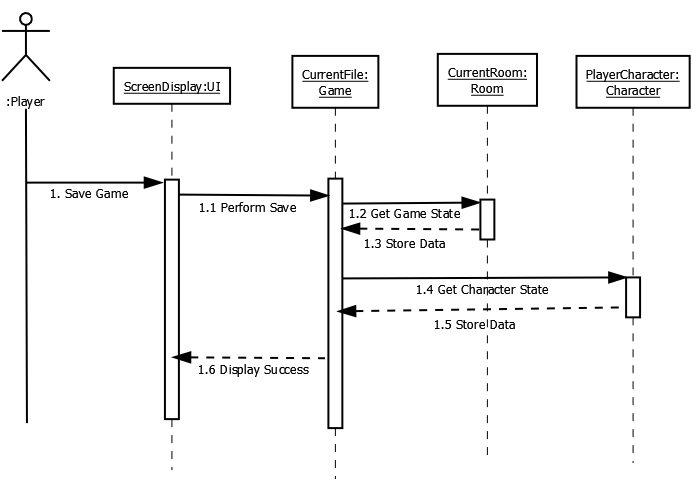
Use Item



Initiate Battle



Save Game



Load Game

