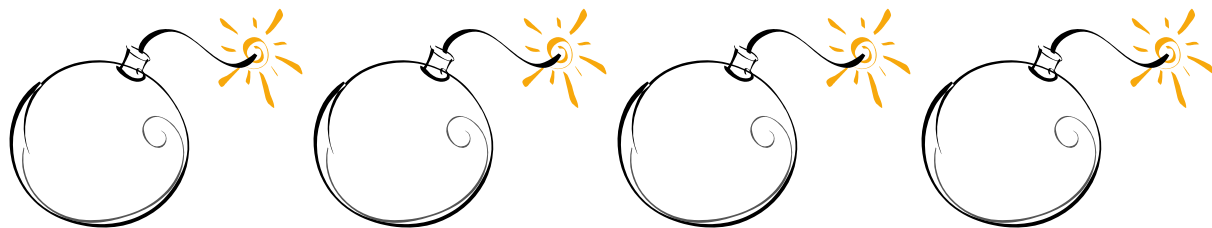# SOFTWARE REQUIREMENTS SPECIFICATION

Project Bomberman: Reloaded

FEBRUARY 9, 2014

TEAM RETRO REVOLUTION

Written by:

Hang Kang, Simon Krafft, Qianyu Liu, Alistair Russell, Sean Ryan, Wayne Tam

Table of Contents

1. Introduction

1.1 Purpose

The objective of this Software Requirement Specification (referred to as SRS thereafter) document is to help guide the complete design and development of a gaming system (referred to as the system thereafter). The SRS will be based on the client's written description of the desired product. Overall, this document will enumerate the features, core components and constraints to create an action, maze-based game similar to those of the original *Bomberman* franchise developed by Hudson Soft first published in 1985.

1.2 Intended audience

This document is intended for all members the software development team as a reference to the system's requirements and specifications. This document is also created for client to have a clearer perception of the gaming system and its functions. The Specific Requirements section constitutes the vital components that will be implemented in the Bomberman game.

1.3 Project Scope

The focus of this project is to develop a maze-based game similar to the original *Bomberman* video game franchise. The game will user to control a character and unlock levels. The game is designed to increase difficulty over time and will allow user to save and load game at each level. The game should support single player mode as well as two-player mode with alternating turns. The main objectives are to design a game software that functions and runs smoothly while implementing all the fundamental features mentioned on the document. The global scope of the project is to learn to collaborate on a major project with different people.

1.4 Definitions, Acronyms & Abbreviations

Client: refers to the person who requests the initiation of the project and decide the key necessary requirements of the software gaming system.

Game: refers to *Bomberman: Reloaded*, the current game in development. The game is also sometimes referred as the application.

User: refers to the person(s) who uses the software system and provide input when needed. The user will also sometimes be referred as the player in the context of the actual game.

Bomberman: refers to the main character which is controlled by the player

A.I.:   refers to the enemy or enemies in the game of which the player must destroy in order to pass the game.

Bomb: refers to the weapon the character plants on the screen which as the ability to destroy the Bomberman itself, A.I.s, blocks and other objects.

Power-up(s): refers to obtainable item(s) in the game for the player to provide additional gameplay advantages.

Square: refers to the unit of measurement for length in this game. For example, 1x1 square is the size of Bomberman character.

BGM: background music, also referred as music

SFX: special sounds effects

GUI: Graphical User Interface, refers to the user interface that allow the user to interact with the system.

JFC: Java Foundation Class, refers to the graphical framework for building portable Java-based graphical user interfaces

API: Application Programming Interface, specifies how software components should interact with each other

1.5 References

The link below and its associated sites are studied to have a better understanding of the rules of the game, properties of power-ups and the behavior of the Bomberman and his enemies.

"Bomberman." Internet: http://strategywiki.org/wiki/Bomberman, Nov. 17, 2013 [Feb. 2, 2014]

1.6 Overview

The rest of this SRS document is consisted of two sections: overall description and specific requirements. The overall description (Section 2.x) will give general product perspective, functions and use cases, user characteristics, constraints and assumptions of the project. This section is primarily written for the client and other non-developers to have a basic knowledge of the development phase. The overall description is also used as a reference for the specific requirement.

On the other hand, the specific requirements (Section 3.x) is mainly written for the development team to guide through the completion of the design and implementation phase. This section contains details of functional and non-functional requirements, organized by categories. The document will also include any technical information about the graphics, external interface requirements, and other necessary requirements. The contents of this section is based on the client's description of the desired final product. All specific requirements will be numbered to facilitate further references.

2. Overall Description

2.1 Product Perspective

The project will be a remake of the *Bomberman* game from the original series of the same name. The design and appearance of the game will have added personal touches from the development team. However, the overall emphasis is to design a Bomberman game with all the core functions and features notable in the original Bomberman games.

2.2 Product Features and Functions

2.2.1 Menus

The game will have a main menu and an in-game menu. The main menu is an interface first presented when the user open the game where the user can start the game. The main menu contains features including but not limited to: start new game, load game, select levels and options. The in-game menu is an interface presented only when the user press pause during the game. The in-game menu allows user to change some of the options, such as volume of background music and SFX. The in-game menu will also display features such as score, time and lives remaining.

2.2.2 Game Play

Game play refers to the overall functions and features of the game which will allow the user to control the Bomberman's movement, go through obstacles and A.I.s, and to proceed to the next level. The player will be able to place bomb, destroy blocks and defeat A.I.s to gain points. The player will also have the opportunity to pick up power-ups that gives player additional gameplay advantages. The user's score, time, and lives remaining will be displayed on the screen as well as the in-game menu when the game is paused.

2.2.3 Options

The user will be allowed to change settings through the main menu or the in-game menu. The option function will allow the user to adjust features such as background music volume and SFX volume. Controls and properties of all power-ups will also be described in the option menu as a guide, but not available for the user to change.

2.2.4 Exit

This function will exit the game.

2.3 Use Cases

The below use cases define the interaction between the user (player in the context of the game) and the Bomberman game system. All use cases are essential to the development unless otherwise indicated.

2.3.1 Use Case: Main Menu Navigation

The main menu is presented when the user launches the application or when the user selects "Return" in any of the submenus or "Quit" the in-game menu. All use cases under main menu navigation is initiated by the user, while the submenu differs corresponding to the option the user selects. The user can choose from six options in the main menu: "New Game", "View High Score", "Load Game", "Select Levels", "Options" and "Credits".
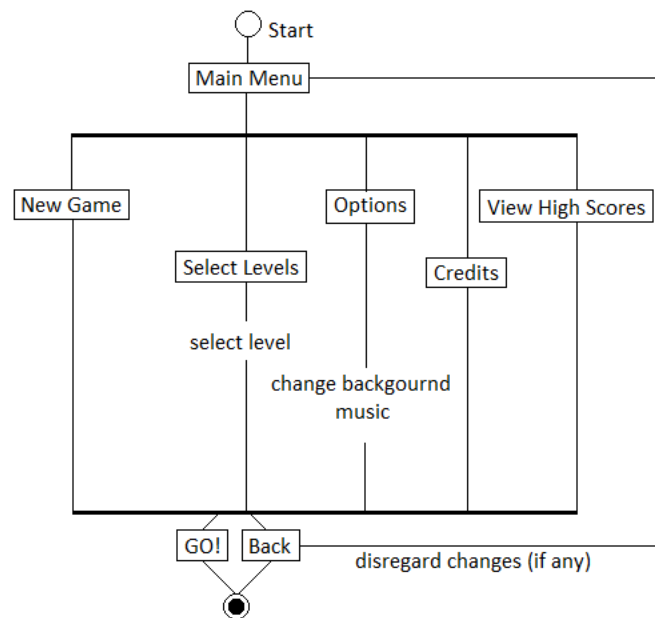
Figure 1: Use Case Main Menu

2.3.1.1 New Game

Use Case Name:	Navigating New Game
Participating Actor(s):	User, System
Entry Conditions:	User select "New Game" from the main menu

Flow of Events:	1. If user selects "Go!"
	  1.1. Application runs level from level 1
	  1.2. Bomberman character is displayed on the top left corner of screen
	2. Else if user selects "Back"
	  2.1. Application disregards all input
	  2.2. Application returns to the main menu

Exit Condition	1. User selects "Go!" (see 2.3.3 Play Game), or 2. User selects "Back"

2.3.1.2 View High Score (desired)

Use Case Name:	Navigating View High Score
Participating Actor(s):	User, System
Entry Conditions:	User select "View High Score" from the main menu
Flow of Events:	1. System displays the "View High Score" menu 1.1. Application displays a top 10 list of player names and their

	corresponding scores in descending order from top to bottom
	2. User view high score
	3. User selects "Back" and returns to the main menu
Exit Condition	User returns to the main menu
Quality Requirements:	The system should automatically update and sort the list of scores after every game.
Exception:	The "View High Score" menu should be disabled when there is no score stored in the game system.

2.3.1.3 Select Levels

Use Case Name:	Navigating Select Levels
Participating Actor(s):	User, System
Entry Conditions:	User select "Select Level" from the main menu
Flow of Events:	1. System displays the "Select Level" menu with options to choose levels to continue the game
	1.1. Application prompts user to select a level
	1.2. User selects level
	2. If user selects "Go!"
	2.1. application runs selected level
	3. Else if user selects "Back"
	3.1. Application returns to the main menu
Exit Condition	1. User selects "Go!" (see 2.3.3 Play Game), or
	2. User selects "Back"
Feature Requirements:	Application should have 10 levels with increasing difficulty
	Each "level" should be enable only when that level has already been unlocked (by any player)

1.3.1.4 Options (Music)

Use Case Name:	Navigating Options
Participating Actor(s):	User, System
Entry Conditions:	User select "Options" from the main menu or the in-game menu
Flow of Events:	1. System displays the "Option" menu with options to change background music volume. The controls and explanation of all power-ups will also be shown here.
	1.1. User adjust background music volume,
	1.2. User view controls and power-ups
	2. User selects "Back"

	2.1. Application save changes (if any)
	2.2. If user accesses "Options" in main menu
	    2.2.1. User returns to the main menu
	2.3. Else if user accesses "Options" in in-game menu
	    2.3.1. User returns to the in-game menu

Exit Condition:	User returns to the main menu or the in-game menu (depending on the access point)
Feature Requirements:	System should save the setting changes for the remainder of the application session. Option interface should be clean and easy to navigate: user should be able to complete setting change within 30 seconds 9 out of 10 tries.

2.3.1.5 Credits (desired implementation)

Use Case Name:	**Navigating Credits**
Participating Actor(s):	User, System
Entry Conditions:	User select "Credits" from the main menu or when player finishes the entire game (unlock all the levels)
Flow of Events:	1. System displays list of names of participants and their role in the development process including, but limited to, the client and the members of the development team. 2. Credits ends and automatically returns to the main menu, or 3. User selects "Back" and returns to the main menu
Exit Condition:	User returns to the main menu

2.3.2 Use Case: In-Game Menu Navigation

The in-game menu is presented when the user pauses the game. When paused, all features (time, A.I.s) will stop moving, and the user will not be able to control the Bomberman. All use cases under in-game menu navigation is initiated by the user, while the submenu differs corresponding to the option the user selects. The user can choose from three options in the in-game menu: "Save Game", "Resume" and "Options".
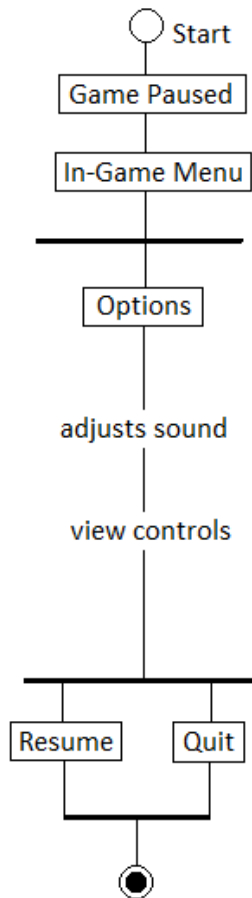
Figure 2: Use Case In-Game Menu

1.3.2.1 Resume

Use Case Name:	Selecting Resume
Participating Actor(s):	User, System
Entry Conditions:	User select "Resume" from the main menu
Flow of Events:	1. Application exits in-game menu and returns to the game interface 2. Application display a count-down from 3 (during this time the game should still be paused) 3. When count-down finishes, game continues
Exit Condition	User exits in-game menu and plays game (see 2.3.3 Play Game)

2.3.2.4 Options (Music and SFX)

See 2.3.1.4 (Navigating Options)

2.3.3 Use Case: Play Game

This section describe the flow of events when the user plays the game. All use cases under play name is initiated by the user and based on user's input.

The system will also automatically saves user's high score when the game finishes updates corresponding score in the "View High Score" submenu of the main menu (if the score is within the top 10. See 2.3.1.2 View High Score).
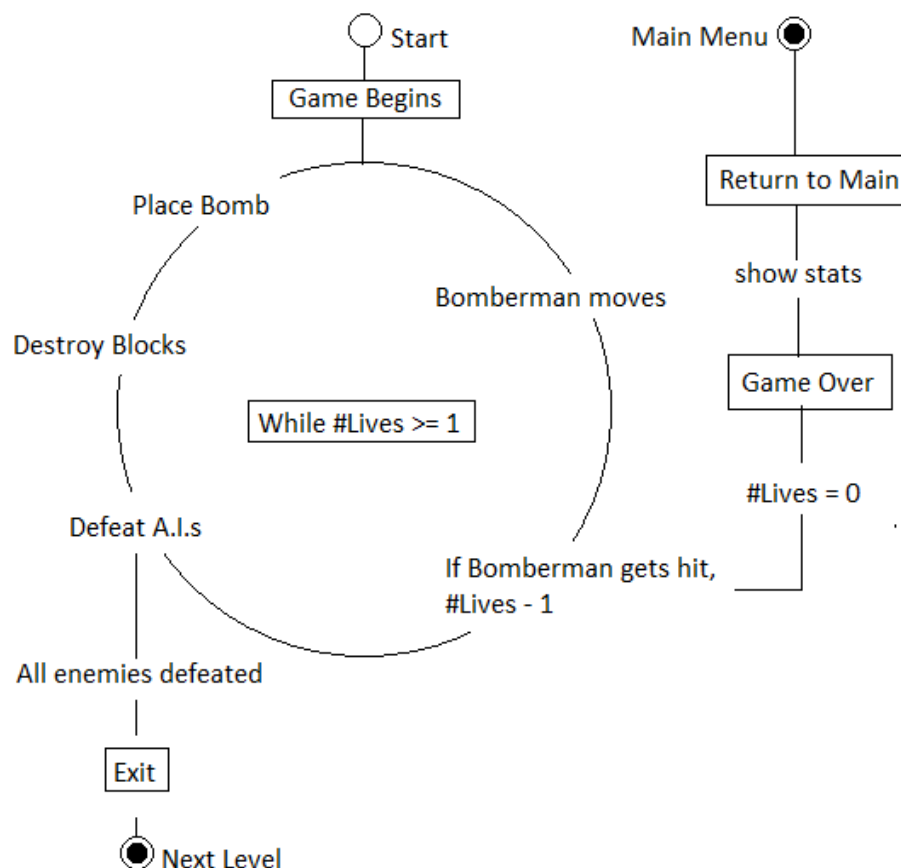

Figure 3: Use Case Play Game (Single Player)

2.3.3.1 Playing Mode

Use Case Name:	Play Game – Single Player
Participating Actor(s):	User, System
Entry Conditions:	1. User selects "Go!" (from "New Game" or "Select Levels"), or 2. User selects "Resume" in in-game menu

Flow of Events:	1. System starts (or continue in case of loading and resuming)
	    1.1. User input is enabled. User can use command keys to control the Bomberman character
	    1.2. All A.I. starts moving
	2. While player's number of lives is greater than zero, user control Bomberman character and player plays game by destroying blocks and defeating A.I.s
	    2.1. If A.I.s or blocks get hit by bomb
	        2.1.1.1. The A.I. or block disappear from the screen
	        2.1.1.2. Points are added to the score
	    2.2. If player gets hit by bomb or A.I.s
	        2.2.1.1. Number of lives deducts 1
	        2.2.1.2. Player is teleported back to the top left corner
	        2.2.1.3. Player is free from damage and placing bomb for 2 seconds.
	3. If user completes level (after defeating all A.I.s and exits level)
	    3.1. If user completes the last level, or
	        3.1.1. System displays "You Won!" message
	        3.1.2. System displays player's score
	        3.1.3. System saves score to memory
	        3.1.4. User selects "Back to Main" and returns to the main menu
	    3.2. Player proceeds to the next level
	        3.2.1. Points are added to the score
	        3.2.2. New level unlocks (user can now load from this level)
	        3.2.3. New level begins (loop back to 1)
	4. If number of lives equals zero
	    4.1. System displays "Game Over" message
	    4.2. System displays player's score
	    4.3. System saves score to memory where it will be updated in the "View High Score" menu (if the score is in the top 10)
	    4.4. User selects "Back to Main" to return to the main menu (see 2.3.1 main menu)

	5. Else if user pauses the game 　　5.1. User exits game play interface 　　5.2. User enters in-game menu interface
Exit Condition:	1. User selects "Back to Main" after losing to completing game (see 2.3.1 Main Menu), or 2. User pauses the game and enters in-game menu interface (see 2.3.2 in-game menu)

2.4 User Characteristics:

The game should be playable by any users at any technical skill level, regardless of age and experience. Instructions, controls and properties of power-ups will be described in the game. However, the user is expected to know the fundamental rules of the game beforehand. The game should not require any previous acquired knowledge about the game functions to be able to play the game. Other than simple controls for moving, placing bombs and pause game, the user should not expect to learn any other sets of commands.

2.5 Constraints

2.5.1 Memory Constraints

Since this game will be of small scope compared to most games on the market today, memory usage should not be a critical constraint. The current estimate is that the game should require less than 100 MB hard drive memory. Memory allocation and optimization will be used on a needed basis and implemented more if there is extra time after the game has passed its testing phase successfully.

2.5.2 Design and Implementation Constraints

This software is designed to be playable by users of all ages and skill levels. However, the user is expected to know some of the fundamental rules of the game. Users should not have access to the source code of the product. The bulk of the code will be written in java with the main IDE being Eclipse. The main GUI will be implemented through the Java Swing library import. Java Swing will be implemented through the JFC which will be the API for providing the GUI to the developers. The McGill web server will be utilized to publish the game to facilitate the availability to users. All corrections and addition of code must be posted to GitHub for transparency to all team members.

2.5.3 Time Constraints

The *Bomberman: Reloaded* is expected to be completed by late March. The game is expected to be released in its entirety to the client by the early April. The official release of the game will be depend on the decision of the client.

2.6 Assumptions and Dependencies:

- This product shall require an operating system that can recognize and run java code.

- This product shall require the utilization of java libraries imported through Eclipse.

- The graphics quality will be dependent on the quality and implementation of code.

1. Specific Requirements

3.1 Detailed Functional Requirements by Category

3.1. CT Controls

3.1. CT.1 WASD (essential; not yet implemented; medium)

3.1. CT.1.1 Bomberman character's movements are only controlled by WASD.

3.1. CT.1.2 Bomberman character moves in the direction of the key pressed.

3.1. CT.1.3 When the player presses and holds an arrow key, the character will move in the corresponding direction until the key is released.

1.1. CT.2 Space Key (essential; not yet implemented; medium)

3.1. CT.2.1 When space key is pressed or held, the character will place one bomb.

3.1. CT.3 Pause Key (essential; not yet implemented; medium)

3.1. CT.3.1 When a designated key is pressed, the game will pause. See 3.1. GP.3 Pause for specifications for pausing the game.

3.1. MN Menus

Menu requirements contain any functional requirement that is related to the main menu and the in-game menu. Both main menu and in-game menu have submenus. Menu provides a platform for the user to interact with the game. The user will have to navigate through menus and submenus in order perform functions such as play game and change settings.

3.1. MN.1 Menu Graphics (essential; not yet implemented, medium)

3.1. MN.1.1 Menus (main and in-game menus) should be an interface with hyperlinks that connects to various submenus.

3.1. MN.1.2 Submenus should extend menus in term of graphics.

3.1. MN.2 Main Menu (essential; not yet implemented, medium)

3.1. MN.2.1 The main menu appears after the introductory clip or whenever the user selects options that leads to the main menu.

3.1. MN.2.2 The main menu contains the options that link to submenus. See 3.1. MN.4 – 3.1. MN.9 for the specifications of main menu submenus.

3.1. MN.3 In-game or Pause Menu (essential; not yet implemented, medium)

3.1. MN.3.1 The in-game menu appears when the user pauses the game.

3.1. MN.3.2 The in-game menu contains the options that link to submenus. See 3.1. MN.10 – 3.1. MN.11 for the specifications of in-game menu submenus.

3.1. MN.3.3 To exit in-game menu, user must selects "Resume". See 3.1. GP.3 for specification of pausing the game.

3.1. MN.4 View High Score (optional, not yet implemented, difficult)

3.1. MN.5.1 View High Score displays the top 10 player names and their corresponding scores

3.1. MN.5.2 High Score should automatically updates its content after every game in single player mode

3.1. MN.5 Select Levels (essential, not yet implemented, difficult)

3.1. MN.7.1 The submenu allows the player to select a level

3.1. MN.7.2 Each "level" should be accessible only when that level has already been unlocked (by any player)

3.1. MN.6 Options (essential, not yet implemented, medium)

3.1. MN.8.1 The submenu that allows the player to adjust the volume of background music and SFX. See 3.1. AV for specifications of sounds and audio.

3.1. MN.8.2 System must save any changes made and implement the changes for the duration of the game play

3.1. MN.8.3 The submenu must display controls and descriptions of each power-ups in image format

3.1. MN.7 Credits (optional; not yet implemented, easy)

3.1. MN.9.1 The submenu displays the list of names of participant in this project.

3.1. MN.9.2 The format of presentation is scrolling texts

3.1. MN.8 In-Game Options (desirable, not yet implemented, medium)

3.1. MN.11.1 See 3.1. MN.8 Options for specifications of options

3.1. MN.9 Back (essential, not yet implemented, medium)

3.1. MN.12.1 All submenus of main menu must have the option to return to the main menu

3.1. MN.12.2 All submenus of in-game menu must have the option to return to the in-game menu

3.1. MN.12.3 In-game menu must have the option to return to the main menu

3.1. MN.12.4 In-game menu must have the option to return to the game. See 3.1. GP.4 for specification for resuming game

3.1. MN.10 Play (essential, not yet implemented, medium)

3.1. MN.13.1 The main menu must have the option to start new game.

3.1. MN.13.2 "Select Level" of main menu must have the option to start game at desired level, which should only be enabled when a level is selected.

3.1. MN.11 Sound (essential, not implemented, difficult)

3.1. MN.14.1 See 3.1. AV for specifications of background music and SFX.

3.1. BM Bomberman (Main Character)

3.1. BM.1 Graphics (essential; not yet implemented, easy)

3.1. BM.1.1 Bomberman image is based on the original *Bomberman* game.

3.1. BM.1.2 The size of image is a 1x1 square

3.1. BM.2 Actions (essential, not yet implemented, medium)

3.1. BM.2.1 Bomberman's movements are controlled by the user based on user's input. See 3.1. CT Controls for specification of command keys.

3.1. BM.2.2 WASD keys will control Bomberman's movements

3.1. BM.2.3 Bomberman cannot move through blocks.

3.1. BM.2.4 Bomberman cannot move out of the map

3.1. BM.2.5 Space key will allow Bomberman to place a bomb

3.1. BM.3 Life (essential; not yet implemented, medium)

3.1. BM.3.1 Bomberman will have 3 lives initially (when user starts a new level)

3.1. BM.3.2 Bomberman's number of lives is displayed on the screen during the game play.

3.1. BM.3.3 Number of lives decreases when Bomberman is hit. See 3.1. GP.5 Hit Detection − Bomberman for specification of when Bomberman is hit.

3.1. BM.3.4 At the start of every level, the number of lives is reset to 3.

3.1. BM.4 Bomb (essential; not yet implemented; difficult)

3.1. BM.4.1 There is a limit to how many bombs the Bomberman can place.

3.1. BM.4.2 Bomberman can place the a bomb immediately after the first bomb explodes.

3.1. BM.4.3 Bomb produces a cross-shaped explosion. See 3.1 AV.4 for specification of bomb explosion visual effect

3.1. BM.4.4 The default range of an explosion is 1 block-size

3.1. BM.4.5 Bomb explodes after certain amount of time (will be determined during implementation)

3.1. BM.4.6 Bomb disappears after explosion

3.1. BM.4.7 Bombs have position

3.1. AI Artificial Intelligence (Enemy)

3.1. AI.1 Graphics (essential, not yet implemented, easy)

3.1. AI.1.1 Character image is based on the original Bomberman game.

3.1. AI.1.2 The size of image is a 1x1 square

3.1. AI.1.3 A.I.s images vary for different types of A.I.s

3.1. AI.2 Movement (essential, not yet implemented, difficult)

3.1. AI.2.1 A.I.s moves by itself without user input

3.1. AI.2.2 A.I.s cannot move through blocks.

3.1. AI.2.3 A.I.s cannot move out of the map

3.1. AI.3 Implementation (essential, not yet implemented, difficult)

3.1. AI.3.1 A.I.s cannot be initially placed at the same place as a block or wall

3.1. AI.4 Intelligence (desirable, not yet implemented, difficult)

3.1. AI.4.1 A smart A.I. has the ability to detect a bomb, and avoids it

3.1. AI.4.2 A smart A.I. has the ability to detect the player, moves towards the player.

3.1. BL Blocks

3.1. BL.1 Graphics (essential, not yet implemented, easy)

3.1. BL.1.1 Block image is based on the original Bomberman game.

3.1. BL.1.2 The size of image is a 1x1 square

3.1. BL.2 Properties (essential, not yet implemented, medium)

3.1. BL.2.1 Block does not move for the duration of the game unless it is destroyed.

3.1. BL.2.2 The player cannot pass though blocks

3.1. BL.2.3 Blocks have positions

3.1. BL.4 Implementation (essential, not yet implemented, difficult)

3.1. BL.4.1 Blocks will be distributed in patterns by design.

3.1. BL.5 Types (essential, not yet implemented, medium)

3.1. BL.5.1 Destructible blocks can be destroyed by bomb. See 3.1. GP.7 Hit Detection – Blocks for specification of destroying blocks.

3.1. BL.5.2 When a block is destroyed, points in added to player's score.

3.1. BL.5.3 Indestructible blocks cannot be destroyed by bomb.

3.1. LV Levels and Maps

3.1. LV.1 Graphics (essential, not yet implemented, easy)

3.1. LV.1.1 Each level must have a different map

3.1. LV.1.2 All levels contain A.I.s, blocks and power-ups. See 3.1 AI.3 and 3.1 BL.4, and

3.1. LV.2 Difficulty (essential, not yet implemented, medium)

3.1. LV.3.1 There should be a total of 10 levels.

3.1. LV.3.2 Each level is comparatively more difficult than the previous. Primary method of measuring difficulty is by the type and number of A.I.s in each level.

3.1. LV.3 Exit (essential, not yet implemented, medium)

3.1. LV.3.1 Each level has an exit, where the player must pass through to proceed to the next level.

3.1. LV.4 Score and Time (essential, not yet implemented, medium)

3.1. LV.4.1 The game must display current time and score on screen when playing game.

3.1. LV.4.2 Score gained from one level is carried and added to the next level.

3.1. LV.4.3 System should organize all stored scores in a sorted manner (descending order) with their corresponding player names.

3.1. LV.4.4 User must be able to see the list of top scores. See 3.1. MN.5 View High Score for specification of viewing high score.

3.1. GP Game Play

3.1. GP.1 Game Begins (essential, not yet implemented, medium)

3.1. GP.1.1 Bomberman's initial position is fixed (not placed randomly)

3.1. GP.1.2 The enemy starts to move as the game starts

3.1. GP.2 Pause (essential, not yet implemented, medium)

3.1. GP.2.1 When game is paused, all objects (including the player and A.I.s) must stop moving

3.1. GP.2.2 When game is paused, application stops timing

3.1. GP.3 Resume (essential, not yet implemented, medium)

3.1. GP.3.1 When game is resumed, all objects starts moving

3.1. GP.4.2 When game is resumed, application starts timing

3.1. GP.4 Hit Detection – Bomberman (essential, not yet implemented, difficult)

3.1. GP.4.1 If the Bomberman is touched by an A.I. or when it is within the range of bomb explosion, the Bomberman is considered "hit".

3.1. GP.4.2 When the Bomberman is hit, its number of lives is deducted by 1

3.1. GP.4.3 Bomberman is free from damage for 3 seconds after it is hit.

3.1. GP.5 Hit Detection – A.I. (essential, not yet implemented, difficult)

3.1. GP.5.1 If an A.I. is within the range of bomb explosion, it is considered "defeated" by the Bomberman.

3.1. GP.5.2 When an A.I. is defeated, it is removed from the screen.

3.1. GP.5.3 Points are added to the score.

3.1. GP.6 Hit Detection – Blocks (essential, not yet implemented, difficult)

3.1. GP.6.1 If a block is within the range of bomb explosion, it is considered "hit".

3.1. GP.6.2 When a block is hit, it is removed from the screen

3.1. GP.6.3 Points are added to the score

3.1. GP.7 Consume Power-ups (essential, not yet implemented, difficult)

3.1. GP.7.1 If the Bomberman is at the same position of the power-up, it is considered that the player "consumed" the power-up

3.1. GP.7.2 When the Bomberman consumes a power-up, its corresponding properties changes. See 3.1. PU.3 Types for specification of different power-ups

3.1. PU Power-Ups

3.1. PU.1 Image (essential, not yet implemented, easy)

3.1. PU.1.1 All power-ups images will be taken from the original Bomberman game.

3.1. PU.1.2 All power-up images will be 1x1 square.

3.1. PU.2 Implementation (essential, not yet implemented, difficult)

3.1. PU.2.1 Power-ups will be hidden under where destructible blocks are positioned, randomly displayed when that block is destroyed.

3.1. PU.3 Types (essential, not yet implemented, difficult)

3.1. PU.3.1 Bomb Range: when consumed, range of bomb increases by 1 square.

3.1. PU.3.2 Health: when consumed, player's health increases by 1

3.1. PU.3.3 Speed: when consumed, the speed of Bomberman's movement increases.

3.1. PU.3.4 Double: when consumed, player's current score doubles.

3.1. AV Audio & Visual Effect

Background music will be present throughout the game. The main menu and the gameplay will feature retro style Bomberman themed music.

3.1. AV.1 Background Music (essential; not yet implemented, medium)

3.1. AV.1.1 Background music will be taken from the original Bomberman game with reference.

3.1. AV.1.2 System plays a continuous looping retro Bomberman game theme music when user is in main menu.

3.1. AV.1.3 System plays a continuous looping retro Bomberman game theme music when user is game play and in-game menu.

3.1. AV.2 Visual Effects (desired, not yet implemented, difficult)

3.1. AV.2.1 Visual effects will be taken from the original *Bomberman* game.

3.1. AV.2.2 Power-ups: the system plays a 1-second animation indicating that the Bomberman has consumed a power-up.

3.1. AV.2.3 Bomb Explosion: the system plays a 1-second animation portraying the explosion of bomb

3.1. SL Storyline

Storyline requirements contains any functional requirement that is related to the function of the plot for *Bomberman: Reloaded*. Storyline includes, but limited to, introductory clips, cut scenes, and ending clips.

3.1. SL.1 Graphics (desirable, not yet implemented, easy)

3.1. SL.1.1 Clips will either be video clips taken from the original game

3.2. SL.2 Start Screen (desirable, not yet implemented, easy)

3.1. SL.2.1 An introductory start screen appears every time user runs the game.

3.1. SL.2.2 The start screen links with the main menu

3.2. SL.3 End Screen (desirable, not yet implemented, easy)

3.1. SL.3.1 An end screen appears when the player either loses or wins the game

3.1. SL.2.2 The end screen links with the main menu

3.4 External Interface Requirements:

3.4.1 User Interfaces:

The game software requires the use of keyboard to play the game. Mouse is not required but recommend to navigate and interact with the menus. The standard controls will allow the user to load a game, save a game and play the game. The user also can bring up a menu for additional options such as adjusting the volume of music and SFX. The game will be organized through a main story line that will progress by different playable levels, each of different variety and increasing difficulty.

3.4.2 Hardware Interfaces:

There will be no specific hardware interface for this game as it should be able to run on nearly any operating system, the main ones being Windows, Mac, and Linux. If a certain operating system proves more effective at running and organizing the functions of the game it should be noted and evaluated for further improvement if necessary.

3.4.3 Software Interfaces:

The Java designed game will be capable of being run on any version of Windows, Mac or UNIX. More information will be added to this section if necessary when further details are revealed and more knowledge is gained by the developers.

3.5.4 Communications and Error-Handling Interfaces:

Main communication will be through pop-up windows displayed in the event of errors or other important messages. This game will not require internet connection.

3.6 Design Constraints

This software is designed to be enjoyable for all users.

See more in 2.5.1 Memory Constraints and 2.5.2 Design and Implementation Constraints

3.7 Other Requirements

3.7.1 Reliability

The system is only allowed to crash at most 5% of the time.

3.7.2 Availability

The game will be free and be able to run on any computer with Java installed.

3.7.3 Security and Privacy

The game should not request, collect nor exchange any user information from the user or from the computer on which it is installed.

3.7.4 Maintainability

As Bomberman game system is only a semester-long project, this software will not be designed with maintainability in mind.

3.7.6 Other requirements

The software will be package with proper documentation in text file. The documentation includes, but limited to: terms and conditions, instruction on installation, basic rules of the game and a link to which the user can find more information.