Requirements and Analysis Document for Project Blast (RAD)

Table of Contents

Version: 1.0

Date 2014-05-24

Authors: Mattias Nilsen, Anton Freudenthaler, Alex Tao & Axel Savén Östebo

This version overrides all previous versions.

1 Introduction

This section gives a brief overview of the application.

1.1 Purpose of application

The purpose of this project is to make a game based on the classic bomberman game where the players are pitted against each other on a maze-like map and the objective is to kill the other players. The game should offer additional choices to the players and be more focused on the competitive aspect by introducing new objectives for the players.

1.2 General characteristics of application

The application will be a desktop, stand-alone (non-networked), multiplayer application with a graphical user interface for the Windows, Mac and Linux platforms.

The game will be a real-time action game. Two or more players are playing simultaneously against each other in opposing teams. The team that gets a specified amount of score first wins. To get points the players can either defeat the other players from the other team or capture towers that will also give them different benefits.

1.3 Scope of application
The game will not support multiplayer over network or internet.
The game lacks AI and can only be played against other players.
The game only supports input via the keyboard.

The application will not support suspending games.

1.4 Objectives and success criteria of the project

- Four heroes: Mage , Brute , Enforcer and Bomber.
- At least one Map with interactable objects made for two players.
- Working title menu and match settings.
- Tower capturing and tug-of-war-style score system.
- Winning and losing.
- Complete graphics set
- Sound and music
- Multiplayer on same computer
- All hero abilities
- Themed stages
- Stationary hazards (Fire etc.)

1.5 Definitions, acronyms and abbreviations

Bomberman - Bomberman is the name of the game this application is based on. Bomberman is based on a grid system where the players can navigate inside a maze of blocks. This maze, including all other game elements are seen from above in a top-down view. The ultimate goal of this game is to win by being the last survivor. This is done by controlling a character which can place bombs to blow up the other player(s).

Mage - The Mage is a magic wielding Hero who can hurl fireballs at his enemies and fire a beam of light that incapacitates the enemy.

Hero - A Hero is a player-controlled character with two distinct abilities, a primary- and a secondary ability. The primary ability has the function to destroy other players, allies and enemies alike.
The secondary ability is an ability that either assists the user of that ability, or harms other players without destroying them.
These abilities are different on different hero types.

Bomber - The Bomber is a Hero which places bombs and detonates them remotely. If these bombs are left alone, they will detonate on their own after a few seconds.

Brute - The Brute is a Hero which can create instant explosions in front of him without firing projectiles or placing bombs. He can also push blocks and bombs.

Enforcer - The Enforcer is a sucker for technology, this Hero can place drones which will seek out enemy players and explode when it is close enough to harm the other player. Force fields can also be deployed by this maniac which will decrease the speed of other heroes

Tower - The Tower is a structure a player can overtake to gain points and other benefits which will enhance their speed, power or ammo, depending of the tower type. It can be destroyed by blowing it up a few times, a hero can then capture it by walking over it. Towers will fire explosions against heroes that belong to a different team from itself. This object is impassable until it is destroyed and vulnerable for other heroes to overtake.

Map - This is the entire stage where the game takes place. It's consists of a maze of blocks, towers and two or more heroes.

Block - A Block can not be influenced by hero abilities or other explosions from other sources. Its only purpose is to limit player movement in the map. This object is impassable.

Box - A Box is a crate that can be influenced by different hero abilities. This object is impassable.

Stationary hazards - A space with an dangerous object that will affect the player negatively if stepped upon.

Power-ups - Power-ups are effects that will enhance the hero's abilities. These power-ups are divided into three categories: **Speed, Power, Ammo.**

Speed - Speed is the velocity the hero moves around the map.

Power - Power is the power of the explosion created by the hero. The more power the hero has, the bigger will his explosions be.

Ammo - Ammo defines the amount of explosives the hero can have active on the map at once.

Explosives - Explosives are an object a hero can place on the map. These explosives can create explosions, and will trigger in different ways depending on the explosive type.

2 Requirements

2.1 Functional requirements

The players should be able to.
1. Choose between different heroes to play
2. Start a new game.
3. Play out the game using their different abilities.
    a) Use their primary action to kill the other player
    b) Use their secondary action to complement their primary action
    c) Capture towers to gain power ups and points.
    d) Win the game by reaching a certain number of points.

2.2 Non-functional requirements

2.2.1 Usability

The GUI should give the user a clear picture of what to do and how to do it.
Everything should be simple and straightforward. There should be no need to consult an instruction manual or check a help section to figure out how to play it.
The goal of the game is to entertain the player.

2.2.2 Reliability

The game should not crash without reason.

2.2.3 Performance

The FPS should stay at a constant 60 FPS during normal gameplay on a modern computer (2012 model and later).

2.2.4 Supportability

This application should support Windows, Mac and Linux. There should be automated test verifying all use cases. Code related to the GUI should be tested manually.

2.2.5 Implementation

The game will be written entirely in the Java programming language, all hosts must have the JRE installed and configured to be able to run the game. The game needs to be downloaded and ran from the computer.

2.2.6 Packaging and installation

The application will be aimed to be delivered with a single .exe file containing all resources to run the game.

2.2.7 Legal

There are possibly some legal issues regarding rights to the bomberman game, they are not discussed here.

2.3 Application models

2.3.1 Use case model

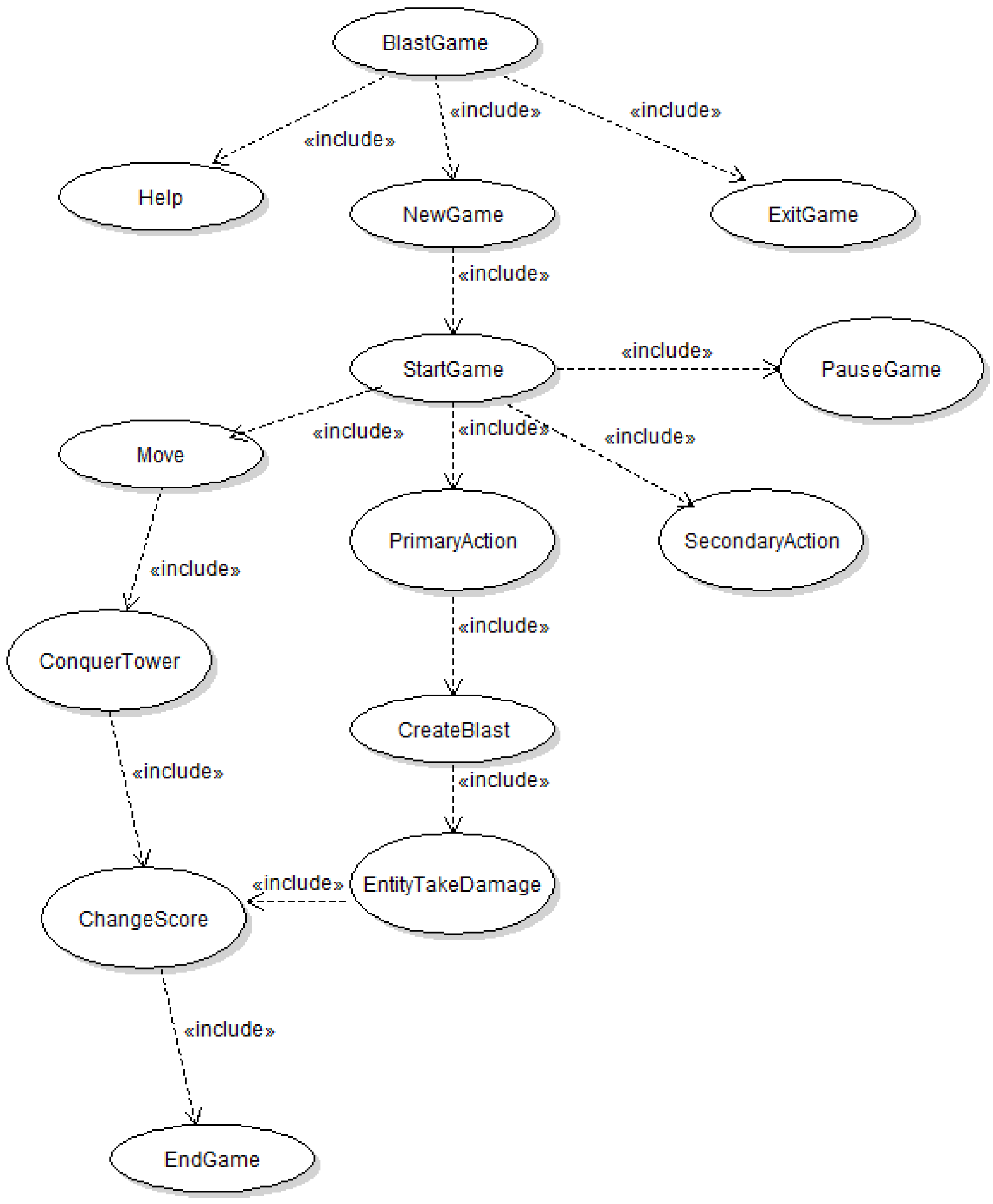

Figure 1. Use Case Diagram

2.3.2 Use cases priority

1. PlayerMove
2. Primary Action
3. Create Blast
5. Secondary Action
6. PlayerDeath
7. New Game
8. Exit Game
9. Player Respawn
10. Conquer Tower.

2.3.3 Domain model

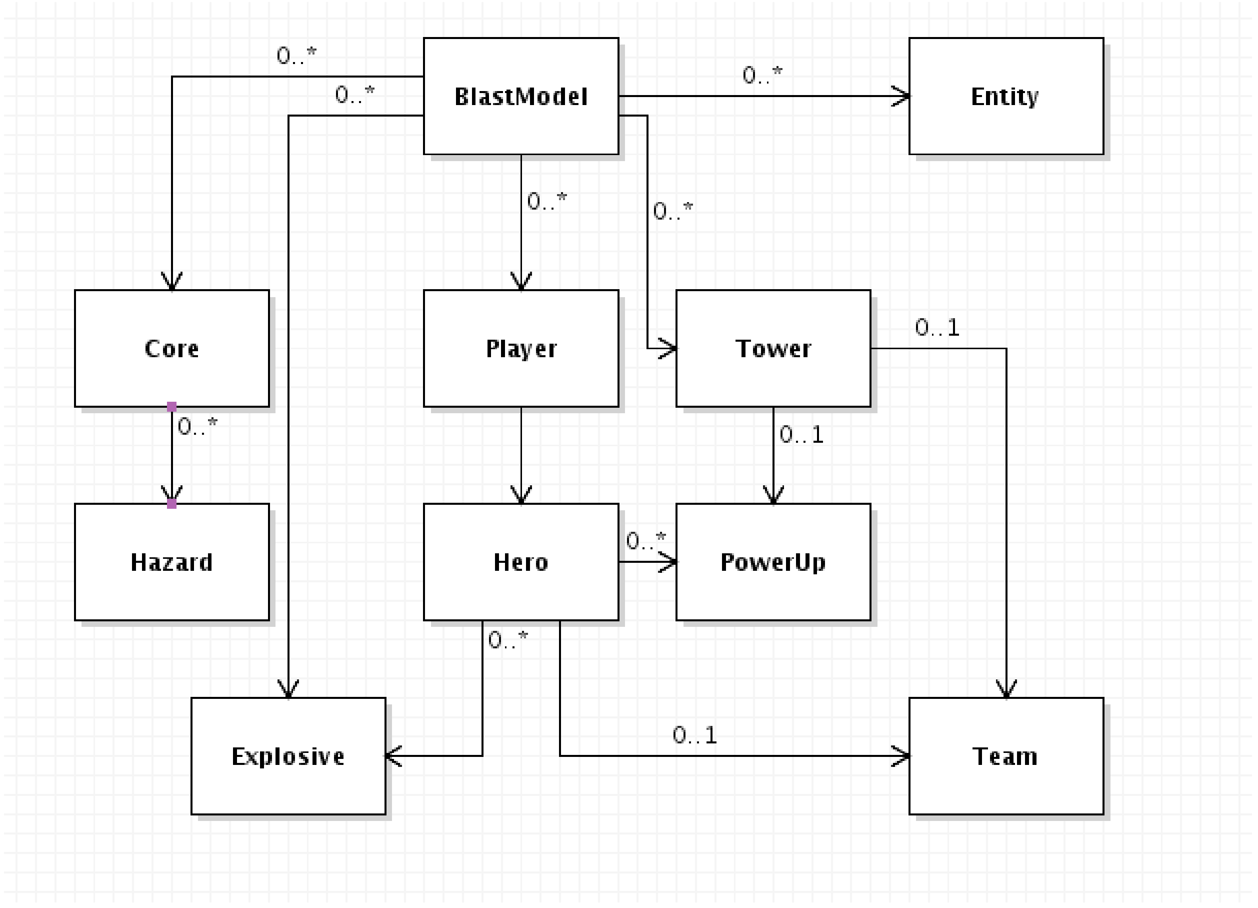

Figure 2. Domain Model

2.3.4 User interface

The application will use a fixed interface that can not be customized by the users. No options to change the graphics will be included and the screen size will not be accomodated.

2.4 References

Bomberman: http://en.wikipedia.org/wiki/Bomberman

Appendix:

Use Case: New Game

Summary: A new game is initialized.

Priority: High

Extends:
Includes:

Participators: User

Normal flow of events

	Actor	System
1	User presses “New Game”	
2		Brings out game settings
3	User enter settings and press “Start”	
4		Game starts (StartGame)

Use Case: Help

Summary: Open up the help file for the user to read.

Priority: Normal

Extends:
Includes:

Participators: User

<u>Normal flow of events</u>

The help file is opened

	Actor	System
1	User selects Help from the title screen	
2		Opens up help file

Use Case: Exit Game

Summary: Shuts down the entire application.

Priority: High

Extends:
Includes:

Participators: User

<u>Normal flow of events</u>

The game shuts down.

	Actor	System
1	User selects Exit from the title screen.	
2		Shuts down the game.

Use Case: Start Game

Summary: When a user presses “start game” button, a new game session will be created.

Priority: High

Extends:
Includes:

Participators: User

Normal flow of events

Starts an completely new game session.

	Actor	System
1	User clicks on the start game button	
		Creates a game board with the chosen settings and places the players on the field.

Use Case: Pause Game

Summary: When a user presses an input key the game should pause in its current state

Priority: Low

Extends:
Includes: ReadHelp

Participators: User

Normal flow of events

A simple pause of the game in action.

	Actor	System
1	Pause key is pressed	
2		The game is paused in its current state. An option screen becomes visible with options: resume, restart, help and exit. No update to the game should be done in this state.
3	Resume button is pressed	
4		The option screen is removed. The game is resumed starting with its paused state. Everything becomes available instantaneously.

Alternate flows

Flow 3.1: User want to restart the game

	Actor	System
3	Restart button is pressed	
4		The option screen is removed.

		The game is restarted with the same user options as the game currently in progress.

Flow 3.2: Player wants to exit game

	Actor	System
3	Exit button is pressed	
4		The option screen and game screen is removed. The main menu is made visible.

Flow 3.3: Player wants help

	Actor	System
3	Help button is pressed	
4		The tutorial slideshow is made visible.
5	Close help button is pressed	
6		The tutorial slideshow is removed.

Use Case: Move

Summary: When a player presses an input key the character should move in the corresponding direction

Priority: High

Extends:
Includes: Player respawn

Participators: User

__**Normal flow of events**__

A simple move with no consequences.

	Actor	System
1	Move key is pressed	
2		Player character is moved in the corresponding direction (up, down, left or right).

__**Alternate flows**__

Flow 2.1: Player can't move because an object is in the way

	Actor	System
1		Player character remains in its square.

Flow 3.1: Player character is moved into danger

	Actor	System
2		Player character is killed
3		Increase the points of the other team. If points are at win condition for other team, end game.

4		Player character is respawned.

Use Case: Primary Action

Summary: The primary action for the player, which is different depending on which hero is used.

Priority: High

Extends:
Includes: CreateBlast

Participators: User

Normal flow of events
Places a bomb

	Actor	System
1	Player presses his main action key	
2		Places an explosive on the position the character is at.
3	The bomb is detonated.	
4		Detonate the bomb (CreateBlast)

Alternate flows

Flow 2.1: Fires an explosive blast

	Actor	System
1	Player presses his main action key	
2		Fires an explosive projectile from the character position. The player loses one actionpower.
3	The projectile hits an object	
4		Detonate the projectile (CreateBlast)

Flow 2.1.1: Character is facing an object

	Actor	System
2		Nothing happens

Flow 2.2: Smashes the ground

	Actor	System
1	Player presses his main action key	
		Smashes the tile in front of him, creating explosive shockwaves in all directions from that tile except toward the player himself. (CreateBlast)

Flow 2.2.1: Player is facing an indestructible block

	Actor	System
		Nothing happens.

Flow 2.3: Deploys tracking drone

	Actor	System
1	Player presses his main action key	
2		Places an explosive drone.
3		The drone tracks other players.
4		When the timer is out. Destroys the drone and creates an explosion (CreateBlast)

Flow 2.3.1: Current space is occupied by something else than the current player

	Actor	System
2		The explosive can’t be placed

Flow 2.3.2: The drone can’t find a nearby enemy

	Actor	System
3		The drone tracks nearby destructible block and starts to move against it.
4		The drone creates a blast when timer is out.

Flow 2.3.3: A fireball is thrown

	Actor	System
3		The fireball starts moving in the direction the player is facing
4		The fireball explodes(CreateBlast) when it hits another object.

Flow 2.4: Character has no ammunition

	Actor	System
2		Nothing happens

Use Case: Secondary Action

Summary: The secondary action for the player, which is different depending on which hero is used.

Priority: High

Extends:
Includes:

Participators: User

Normal flow of events
Detonates all bombs

	Actor	System
1	Player presses his main action key	
2		Detonates all bombs the player has placed. If none, nothing happens.

Alternate flows

Flow 2.1: Fires a paralyzing beam

	Actor	System
1	Player presses his main action key	
2		Fires a straight beam that paralyzes everyone touching it.

Flow 2.2: Pushes destructible block or bomb in front

	Actor	System

1	Player presses his main action key	
2		Pushes block or bomb in front forward, if any.

Flow 2.3: Activates a slowing force field

	Actor	System
1	Player presses his main action key	
2		Activate an aura that follows the player and slows down everyone in its radius except the player that cast it.

Use Case: Create Blast

Summary: A bomb explodes, creating a blast force in all four directions.

Priority: High

Extends:
Includes:

Participators:

<u>Normal flow of events</u>

Something creates an explosion

	Actor	System
1		Explosion tiles spreads in four directions (in a plus shape).

<u>Alternate flows</u>

Flow 2.1: The explosion was caused by a ground pound.

	Actor	System
1		The explosion spreads in all but one direction.

Flow 2.2: Explosion tiles meet indestructible block

	Actor	System
1		The explosion stop spreading in that direction

Flow 2.2: Explosion tiles meet destructible block

	Actor	System
1		The block takes damage. The explosion stop spreading in that direction(EntityTakeDamage)

Flow 2.3: Explosion tiles meet tower

	Actor	System
1		The tower takes damage. The explosion stop spreading in that direction (EntityTakeDamage)

Flow 2.4: Explosion tiles meet player

	Actor	System
1		The player takes damage.(EntityTakeDamage)

Use Case: Conquer Tower

Summary: Player conquers a tower

Priority: High

Extends:
Includes:

Participators: User

<u>Normal flow of events</u>

A player walks into a Tower

	Actor	System
1	Player walks into and takes over a Tower	
2		The tower changes color to the color of the player. The gains points for each tower controller (ChangeScore). The tower doesn’t shoot at that player anymore.

<u>Alternate flows</u>

Flow 2.1: The tower is not destroyed.

	Actor	System
1		Player is blocked and can’t move to this square. Nothing else happens.

Use Case: EntityTakeDamage

Summary: When a hazard hits an object it takes damage.

Priority: High

Extends:
Includes:

Participators:

<u>Normal flow of events</u>

An explosion hits an object.

	Actor	System
1	An explosion hits an object.	
2		That object is destroyed and removed from the game.

<u>Alternate flows</u>

Flow 2.1: An explosion hits a player

	Actor	System
1	An explosion hits a player	
		The player is killed.

Flow 2.1: An explosion hits a tower

	Actor	System
1	An explosion hits a tower	
		The tower takes damage and loses 1 health.

Use Case: ChangeScore

Summary: When certain things happen the score is changed.

Priority: High

Extends:
Includes:

Participators: User

Normal flow of events

A player controls one or more towers.

	Actor	System
1	1 or more towers are controlled by a player	
2		That player gets a number of points that are related to the number of towers that player has at regular intervals

Alternate flows

Flow 2.1: A player is killed.

	Actor	System
1	A player is killed by an explosive	
		The other player gets points.

Use Case: EndGame

Summary: When a condition is met the game ends

Priority: High

Extends:
Includes:

Participators: User

<u>Normal flow of events</u>

One player has the correct number of points

	Actor	System
1	One of the players has the correct number of points.	
2		That player wins the game.