XBlast

McGill University

ECSE 321 – Winter 2014

XBlast

McGill University

ECSE 321 – Winter 2014

Authors:

David Liu

Yike Liu

Eddy Lu

Christopher Reny

Xuzhi Shu

YaHan Yang

February 9th, 2014

Table of Contents

Introduction

Purpose and Scope

The purpose of this document is to provide the team with a high level guide for developing the Bomberman game, XBlast. The goal is to minimize communication time between team members while meeting expected requirements while having clear objectives set for each stage.

During the design of the class diagram, developers will aim to realize the requirements listed in Section 3 with the help of the use cases. This will allow for an abstract visualization of the project. Class diagrams will outline the input, output, and function. During the implementation phase, programmers will create modules independently and refer to the class diagrams to avoid mismatch.

Definition

Bomberman Game:

A game composed of multiple stages with one or two player-controlled characters. The player's goal for each stage is to kill all enemies. The player's main method of attack involves placing bombs on the ground that explode after a set time. Enemies have a variety of attacks including the placement of bombs, collisions, and even projectiles.

References

No external sources were utilized in the creation of this document.

Overview

Refer to section 2 for use cases which allow abstract visualization. Refer to section 3 for detailed requirements which will be used as the guide to game design.

2.1 Product Perspective

Bomberman is a fun java-based game for users of all ages. The goal of the game is to blow up obstacles and kill enemies by dropping bombs.

2.1.1 System Interfaces

The system incorporates a user interface in which the input key functions are used to navigate the menus and gameplay.

2.1.2 User Interfaces

The game menu allows the user to select the number of players, start a game, receive help, look at statistics, pause the game, and end the game. Once the game starts, the users can move and drop bombs by pressing different keys.

2.1.3 Hardware Interfaces

Computer hardware criteria to run the gaming system:

- requires a mouse and keyboard
- needs enough memory storage

2.1.4 Software Interfaces

The gaming system can be run on any operating system. The system calls to the file management system of the operating system to store and retrieve game states.

2.1.5 Communications Interfaces

None

2.1.6 Memory Constraints

The game requires 500kb of Ram memory.

2.2 Product functions and use cases

- Allows the user to register with a name
- Allows two users to team up
- Record the user’s game scores in decreasing order
- Play sound while there is an explosion
- End game
- Close game

Overall Description

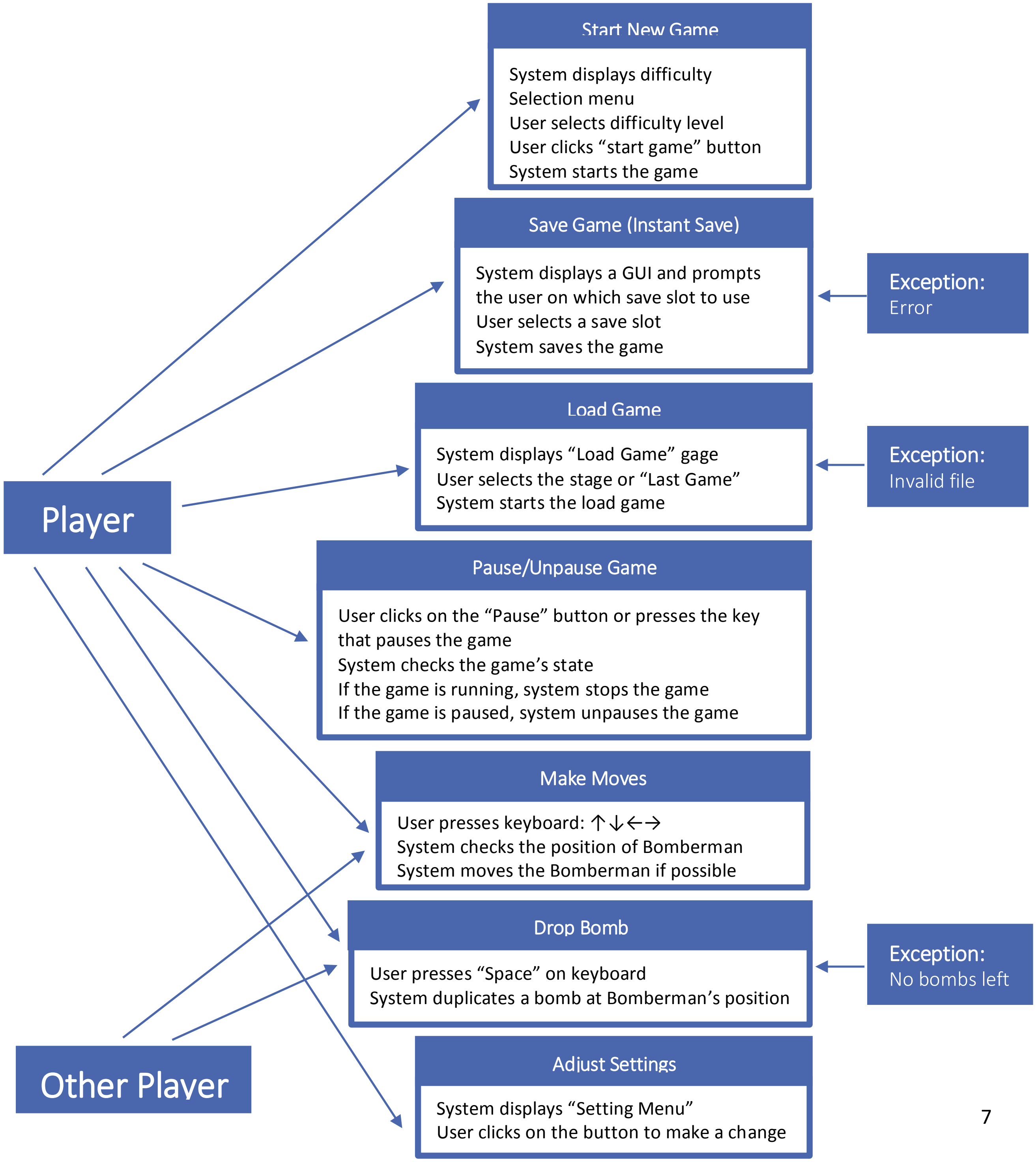
Product Perspectives

Although this product is based on the classic Bomberman game, all contents of the product are solely created by the developers with complete independence from other forms of this game.

As stated in the Overview, this Bomberman game is a java-based game in which users attempt to become victorious by blowing up obstacles and killing enemies by dropping bombs. This game runs on a user interface that can be run with little RAM on computers that run java. Users can utilize a mouse and keyboard to input controls for the game.

Product Functions and Use Cases

In order to begin the game, the user has a choice of selecting different options, such as the difficulty, in the game's main menu. During the game, the user can simply save the game with a simple GUI that displays the file location as well as a verification of a successful save. The user also has the capability of pausing the game while it is running. Different moves such as navigating and dropping bombs can be executed by pressing on different keys on the keyboard. This game also contains the feature where the user is able to manipulate the game's settings. In addition, the user has the ability to end the game at will.



Player

Other Player

Start New Game

System displays difficulty
Selection menu
User selects difficulty level
User clicks "start game" button
System starts the game

Save Game (Instant Save)

System displays a GUI and prompts
the user on which save slot to use
User selects a save slot
System saves the game

Exception:
Error

Load Game

System displays "Load Game" gage
User selects the stage or "Last Game"
System starts the load game

Exception:
Invalid file

Pause/Unpause Game

User clicks on the "Pause" button or presses the key
that pauses the game
System checks the game's state
If the game is running, system stops the game
If the game is paused, system unpauses the game

Make Moves

User presses keyboard: ↑↓←→
System checks the position of Bomberman
System moves the Bomberman if possible

Drop Bomb

User presses "Space" on keyboard
System duplicates a bomb at Bomberman's position

Exception:
No bombs left

Adjust Settings

System displays "Setting Menu"
User clicks on the button to make a change

Start New Game

Participating actors: user, system
Preconditions: user clicks on the "NEW GAME" button on the main menu.

Flow of events:

1. System displays difficulty selection menu, which has 4 different buttons: "easy", "normal", "hard" and "start game". "Normal" is selected by system at first.
2. User selects difficulty level and clicks "start game" button.

Post conditions: System starts the new game with selected difficulty level.

Save Game (Instant Save)

Participating actors: user, system
Preconditions: user selects "save game" from menu.

Flow of events:

1. System displays a GUI and prompts the user on which save slot to use.
2. User selects a save slot.
3. System writes current data to file and relates the file to the save slot.
4. System validates that the file is successfully made.
5. System display "save successful".

Post conditions: system display "save successful" or user exits the save game GUI.
Exceptions: If the file is not successfully created, the system should display an error message and then exit the save game GUI.

Load Game

Participation actors: user, system
Preconditions: user clicks on the "LOAD GAME" button on the main menu.

Flow of events:

1. System displays "load game" page and shows unlocked stages on the page and "last game" button.
2. User clicks on the unlock stage that he\she wants to play or clicks on the "last game" button to continue his last saved game.

Post conditions: If users selected an unlocked stage, then System starts the stage that users selected. If users selected "last game", then System starts the last saved game.
Exceptions: If there is no last saved game recorded, System makes the "last game" button invisible and functionless.

Pause/Unpause Game

Participating actors: user, system
Preconditions: user clicks on the "pause" button or presses the key that pauses the game.

Flow of events:

1. System checks the game's state right before the user tries to pause/unpause.
2. If the game is running, the system pauses it. If the game is paused, the system unpauses it.

Post conditions: the system unpause or pause the game.

Make Moves

Participating actors: user, system
Preconditions: A game is in progress and the user wants to move their game character.

Flow of Events:

1. The use case starts when the game starts
2. The user chooses his/her action using keyboard keys: go left, right, up, down.
3. The system checks if there are obstacles that block the user’s action. If this action is not allowed, the game character remains where it was.
4. If the action is allowed, the system let the game character perform the move.

Post conditions: the system performs the move and wait for user’s next action.

Drop Bomb

Participating actors: user, system
Preconditions: A game is in progress and the user wants to drop a bomb.

Flow of Events:

1. The use case starts when the game starts
2. The user presses the keyboard key to drop a bomb.
3. The system let the screen to display a bomb where the game character is.

Post conditions: the system places a bomb and wait for user’s next action.
Exceptions: the player cannot drop a bomb if there are no remaining bombs left

Settings

Participating actors: user, system

Preconditions: user selects the “SETTINGS” button on the main menu (or side menu in game).

Flow of events:

1. System displays a GUI which shows the control bar of volume, types of the character to use, the control keys of the character, “Save Changes” and “Quit Settings” button at the bottom.
2. User adjusts the volume (or not).
3. User changes the character type (or not).
4. User changes the control keys (or not).
5. System applies the changes made and displays “Save Settings Successful” if the user clicks on “Save Changes”.
6. System displays “Would you like to save the changes?” if user clicks on “Quit Settings”. System saves the settings if user chooses “Yes” and recover the settings if user chooses “No”.
7. System returns to main menu (or on-going game).

End Game

Participating actors: user, system
Precondition: user clears the current stage

Flow of events:

1. System save the stage progress.
2. System displays a GUI that shows top ten scores and corresponding players' names.
3. If user's score is in the top ten, system will ask the user to input his name and records his score and name.
4. If user's score is not in the top ten, system will delete his score.
5. System restarts current stage if user chooses "Play Again" button.
6. System goes to next stage if user chooses "Next Stage" button.
7. System returns to main menu if user chooses "Main Menu" button.

User Characteristics

The Bomberman game will be designed with all gamer types in mind. The user can range from those with little computer experience to experts. The Bomberman game will also have a two player mode to allow to player's to play against each other.

Constraints

Like the original Bomberman games, this Bomberman game will be very simple and easy to learn. The game interface will be simple as well, with the only heads up display being the items currently collected and the number of power-ups gathered. In addition, the user's controls will also be very basic. The movement controls will be the basic W, A, S, and D to move player 1 and I, J, K, and L to move player 2 (If in two player mode). To drop a bomb, player 1 will use SPACE and player 2 will use M.

Assumptions and Dependencies

Assumptions:

- Two Player vs. Mode
- Two Player Co-op
- Second Player Controls
- Upgrades
- Bosses
- Saving and Loading
- Game Difficulty Progression
- Leveling System
- Game Ending

Dependencies:

- HUD
- Game Content Images and Sprites
- Level Layout
- Level Background
- Character Controls
- Item Drops
- Main Menu
- Bombing System
- Collision Detection

Specific Requirements

Functional Requirements

Desirability is ranked from least to most:	Optional, Desirable, Essential
Difficulty is ranked from least to most:	Easy, Medium, Hard

Inputs and Outputs

1. The inputs for the Bomberman game will be the movement and attack keys. For single player, the movement keys will consist of up (up), down (down), left (left), and right (right).
   - Desirability: Desirable
   - Difficulty to achieve: Medium

2. For two players, the movement keys for player one consist of W (up), A (left), S (down), and D (right). These keys will also be used to navigate the menu screens. Player one will also use the SPACE BAR to place bombs during gameplay or select the highlighted menu option in the menu screen.
   - Desirability: Desirable
   - Difficulty to achieve: Medium

3. The movement keys for player two are I (up), J (left), K (down), and L (right). The attack key for player two will be M.
   - Desirability: Desirable
   - Difficulty to achieve: Medium

4. There is a sound system which plays different music depending on the current state of the game.
   - Desirability: Essential
   - Difficulty to achieve: Medium

5. There is simple AI that controls enemies. The simple AI is capable of moving enemies around randomly and changing direction when a collision occurs.
    - Desirability: Essential
    - Difficulty to achieve: Medium

6. Players and enemies will have movement animations. At least 3 frames of images are used to animate each movement direction (up, down, left, right).
    - Desirability: Essential
    - Difficulty to achieve: Easy

Stored Data

1. When the “Save Game” button is clicked, the game will store save data in a file named with a timestamp indicating the time of creation and the level the user was on. This will allow for the user to more easily load the proper file. The contents of the file will include the current level as well as their power-ups and bonuses.
    - Desirability: Desirable
    - Difficulty to achieve: Hard

2. There is a “load game” button in the menu for the player to load a saved game.
    - Desirability: Desirable
    - Difficulty to achieve: Hard

Error Handling

Error handling in the Bomberman game will revolve mainly around loading a game. While loading a previously saved game, if the contents of the save file are corrupted or invalid, the Bomberman game will output an error message saying “The Chosen Save File is Corrupt or Invalid” and the game will bring the user back to the initial menu screen.

Initialization/Shutdown

1. Upon initialization, the game will bring up a menu and await a user input. The user will be able to choose between “New Game”, “Load Game”, “Quit”, and “Restart Game”.
    - Desirability: Desirable
    - Difficulty to achieve: Medium

2. There is a “Restart Game” button on the “game end” page to restart the game at stage 1.
    - Desirability: Essential
    - Difficulty to achieve: Medium

3. The player could choose the difficulty when he starts a new game (easy, medium, hard)
    - Desirability: Desirable
    - Difficulty to achieve: Hard

4. The user can shut down the game if they select “Quit” from the initial menu or from the in game menu.
    - Desirability: Desirable
    - Difficulty to achieve: Medium

Other Functional Requirements:

1. The game screen will be divided into grids each having dimensions 32x32 pixels.
    - Desirability: Essential
    - Difficulty to achieve: Easy

2. The game will have several stages in numerical order. The player has to beat the previous stage in order to move onto the next stage.
    - Desirability: Essential
    - Difficulty to achieve: Medium

3. For each stage, game objects will be generated at set spots.
    - Desirability: Desirable
    - Difficulty to achieve: Medium

4. The player beats a stage when all enemies in the stage are destroyed.
    - Desirability: Essential
    - Difficulty to achieve: Easy

5. Each game object will be mapped to a grid.
    - Desirability: Essential
    - Difficulty to achieve: Easy

6. Movable game objects can move between grids.
    - Desirability: Essential
    - Difficulty to achieve: Medium

7. Movable game objects can only move within the border of the screen
    - Desirability: Essential
    - Difficulty to achieve: Easy

8. Movable game objects cannot move through obstacles game object such as blocks.
    - Desirability: Essential
    - Difficulty to achieve: Medium

9. All game objects have a positive integer assigned as **health** when it's first created. A game object will be destroyed when its **health** is less or equal to 0.
    - Desirability: Essential
    - Difficulty to achieve: Easy

10. The player can only place bombs at his current position.
    - Desirability: Essential
    - Difficulty to achieve: Easy

11. Bombs placed will be stationary and detonate in **x** seconds depending on the current upgrade the bomber man has.
    - Desirability: Essential
    - Difficulty to achieve: Medium

12. The bomber man can only place a maximum of **m** bombs on the screen simultaneously; **m** depends on his current upgrades.
    - Desirability: Essential
    - Difficulty to achieve: Medium

13. After a bomb detonates, fire of length n will be created in 4 directions (up, down, left, right), with the detonated bomb as the center.
    - Desirability: Essential
    - Difficulty to achieve: Hard

14. The fire will be stopped if it touches an obstacle game object such as blocks.
    - Desirability: Essential
    - Difficulty to achieve: Hard
15. All game objects touched by fire will have its **health** decreased by **x**, which depends on the strength of the bomb.
    - Desirability: Essential
    - Difficulty to achieve: Hard

16. Detonated bombs will be removed from the game.
    - Desirability: Essential
    - Difficulty to achieve: Easy

17. Random items could appear after a game object is destroyed.
    - Desirability: Desirable
    - Difficulty to achieve: Medium

18. Only the player could pick up the random items.
    - Desirability: Desirable
    - Difficulty to achieve: Medium

19. The random items can provide various benefits such as: increase length of fire, increase maximum bombs, increase movement speed, increase bomb damage, and increase health.
    - Desirability: Desirable
    - Difficulty to achieve: Medium

20. The game ends if the player is destroyed, or the player beats the last stage.
    - Desirability: Essential
    - Difficulty to achieve: Easy

21. There is a high score system. The top 10 highest score will be stored for each stage and displayed after each stage is finished.
    - Desirability: Essential
    - Difficulty to achieve: Medium

Quality requirements

Performance

1. The delay between key press and on-screen action should be <50ms.
    - Desirability: Essential
    - Difficulty to achieve: Easy

2. The size of the save files should be miniscule, no more than 3Kbs.
    - Desirability: Essential
    - Difficulty to achieve: Easy

3. The game should have 50 frames per second.
    - Desirability: Essential
    - Difficulty to achieve: Easy

4. The load time for each stage should be less than 2 seconds.
    - Desirability: Desirable
    - Difficulty to achieve: Medium

5. Saving the game should take no more than 2 seconds.
    - Desirability: Desirable
    - Difficulty to achieve: Medium

6. Loading a saved file should take no more than 2 seconds.
    - Desirability: Desirable
    - Difficulty to achieve: Medium

Usability

Due to the simplicity of the controls, any user should be able to navigate the initial menu with little difficulty, it should take no more than 30 seconds for any computer-familiar person to start navigating the initial menu. The game shall also come with a manual.txt file explaining all the controls in detail.

1. 85 out of 100 users should be able to figure out how to play the game in under 30 seconds without a guide.
    - Desirability: Desirable
    - Difficulty to achieve: Easy

Reliability

1. The game should crash no more than 2 times every 10 hours of game play.
    - Desirability: Desirable
    - Difficulty to achieve: Medium

2. When the game detects a crash, it will save the current game automatically.
    - Desirability: Optional
    - Difficulty to achieve: Hard

Precision

1. The high score is stored as an integer.
    - Desirability: Essential
    - Difficulty to achieve: Easy

2. The position (x, y) of movable game objects is stored as double.
    - Desirability: Essential
    - Difficulty to achieve: Easy

Design Constraints

Use of only Java Programming
Some algorithm may seem to be more complicated in java than in C or C++

Designers have no experience in arcade game design
None of our group members have experience in designing arcade games

Use of the Waterfall Process Model:

Pros:

- Simple and easy to use
- Practiced for many years
- Easy to manage
- Facilitates allocation of resources
- Works well for smaller projects where requirements are very well understood

Cons:

- Requirements must be known up front
- No feedback of system by stakeholders until after the testing phase
- Major problems aren’t discovered until late in the process
- Lack of parallelism
- Inefficient use of resources

Other Requirements

1. Advanced AI: the advanced AI is capable of chasing after the player and dodging bombs.
    - Desirability: Optional
    - Difficulty to achieve: Hard

2. Various enemy types: There will be different types of enemies capable many different types of attacks. There will also be bosses every few levels.
    - Desirability: Optional
    - Difficulty to achieve: Medium

3. Two player mode: Two players can either co-op to beat the game or fight each other.
    - Desirability: Optional
    - Difficulty to achieve: Hard

4. Instantaneous save: the player can save his progress anytime in game. The game will also periodically auto save.
    - Desirability: Optional
    - Difficulty to achieve: Hard

5. Story mode: there is an interactive story line which will be presented with images, sound and text.
    - Desirability: Optional
    - Difficulty to achieve: Medium

6. Multiple endings: Story mode could lead to multiple different endings.
    - Desirability: Optional
    - Difficulty to achieve: Medium

7. Multiple playable characters: the player can choose between several characters to play as.
    - Desirability: Optional
    - Difficulty to achieve: Easy

Glossary

Bomberman Game
A game composed of multiple stages with one or two player-controlled characters. The player's goal for each stage is to kill all enemies. The player's main method of attack involves placing bombs on the ground that explode after a set time. Enemies have a variety of attacks including the placement of bombs, collisions, and even projectiles.

Destroy
When a game object is destroyed, it will no longer be rendered onto the screen. It will also be removed from the data structure (currently a LinkedList) which stores it.

Game object
A graphical component of the game that's seen by the user. The player, enemies, bombs, and wall are examples of game objects. Different kinds of game objects interact with each other. For example, when a "fire" object touches an "enemy", the enemy takes damage. A game object is usually 32x32 pixels. Some game objects, such as bosses, may have different dimensions.