Blaze Brigade

- Software Requirements Specification -

SFWR ENG 3XA3 - Section L02
007 (Group 7)

Jeremy Klotz - klotzjj
Asad Mansoor - mansoa2
Thien Trandinh - trandit
Susan Yuen - yuens2

October 11, 2016

Contents

List of Tables

List of Figures

Table 1: **Revision History**

Date	**Author**	**Notes**
October 10, 2016	All	Revision 0 of Requirements Document
October 21, 2016	Thien Trandinh	Added changes to functional and Non-functional requirements
December 3, 2016	Thien Trandinh	Added section 5 symbolic parameters and revised document for current project

1 Project Drivers

1.1 The Purpose of the Project

Blaze Brigade serves as a tribute and adaptation of a popular genre: tactical simulation role-playing games. The game aims to provide a form of entertainment to the general public, presenting a fun strategical challenge to its players. The purpose of the project is to recreate an already existing tactical role-playing game, Tactics Heroes, and to ultimately further improve on the overall user experience by implementing additional features not already included in Tactics Heroes.

1.2 The Stakeholders

1.2.1 The Client

The client of the project is Dr. Spencer Smith in the Computing and Software Department at McMaster University.

1.2.2 The Customers

The customers of the game include members of the general public who are interested in the tactical role-playing game genre and are willing to play the game.

1.2.3 Other Stakeholders

Other stakeholders of the project include:

- The Blaze Brigade team

  The project will serve as a means of entertainment and project development experience to its developers.

- Tactics Heroes developers

  The game may serve as a guideline to the original developers of Tactics Heroes to further improve their game by incorporating our projects additional add-on features, as well as modern software development tools and concepts.

1.3 Mandated Constraints

1.3.1 Solution Constraints

Description: The game shall support Windows OS.
Rationale: The client will access the game using a device running Windows.
Fit Criterion: The game shall be developed to support these three operating systems, and testing will ensure that the game runs smoothly on all three operating systems.

1.3.2 Off-the-Shelf Software

The product will be utilizing XNA Game Studio for game development, Adobe Photoshop for graphic editing, and Audacity and FL Studio for music and sound editing. A device that is able to run Visual Studio with XNA Game Studio is also required for project development. Customers must be in possession of an operating system to be able to play the game, and a pointing device to be able to select items on the screen.

1.3.3 Schedule Constraints

Full development of the game does not have any applicable schedule constraints. However, the basic functions of the game must be fully completed by December 2016.

1.3.4 Budget Constraints

The budget for the project is \$0 as all software required for development is provided for free. In addition, all hardware required is already owned by the project's respective developers.

1.4 Naming Conventions and Terminology

Term	Definition
RPG	Role-playing game, in which the player plays the game as a character.
AI	Artificial Intelligence.
Unit	A single character in the game.
Enemy	Units that are hostile to the player's units and may attack the player's units when in range.
Critical hit	Causes the user to deal more damage than expected from a normal hit.
HP	Health points; the amount of damage a unit may take before being defeated.
STR	Strength; Used in calculating physical damage dealt by the unit.
INT	Intelligence; Used in calculating magical damage dealt by the unit.
SKILL	Used to calculate unit's hit rate and critical hit rate.
SPEED	Used to calculate a unit's number of hits on the enemy.
DEFENSE	Used to calculate the melee damage dealt to the unit.
RESISTANCE	Used to calculate the magic damage dealt to the unit.

1.5 Relevant Facts and Assumptions

1.5.1 Relevant Facts

The game that Blaze Brigade will re-implement, Tactics Heroes, was initially named Fire Emblem Factory. The game is heavily influenced from a Japanese game series called Fire Emblem, which was released in 1990 by Intelligent Systems and Nintendo. Fire Emblem revolutionized the genre by incorporating RPG elements, and as such, the success of the genre in modern day can largely be attributed to this series. In addition, Tactics Heroes is still in alpha stage, and only basic gameplay have been fully implemented. Future content for additional features in Tactics Heroes is still under works by its developers.

1.5.2 Assumptions

It is assumed that upon project launch, the host site for the game's download will be free. It is also assumed that members of the general public will not use any aspect of the project or its source code for malicious intent or for resale for monetary value.

2 Functional Requirements

2.1 The Scope of the Work and the Product

Video games are a popular source of entertainment in our society, and tactical RPGs can be traced back to the classics in 1980s. Therefore Blaze Brigade will be a new entertaining game with a throwback on nostalgia. The objective of Blaze Brigade is to move and order your units to defeat all of the enemy player's controllable units. The game will incorporate common tactical RPG features such as path finding, RNG, animations, sound effects, and unique weapons and unit types.

2.1.1 The Context of the Work

The game will be available to play on all desktop and laptop capable of running Windows (or Windows VM), and will be packaged and deployed through Visual Studio. The developers will first develop a working game, and will then allow others to use and evaluate the prototype. Afterwards, the developers will use the feedback to refine the code further and repeat the evaluation process. These evaluation stages are crucial for developing software that meets the customers' preferences, as the game should be fun and entertaining for those who play it.

2.1.2 Work Partitioning

Event #	Event Name	Input	Output
1	Mapping System	Developer Code	Executable File
2	Mechanics	Developer Code	Executable File
3	Units	Developer Code	Executable File
4	Terrain Types	Developer Code	Executable File
5	Sprites	Developer Code/Graphics	Executable File
6	Map Creation	Developer Code	Executable File
7	Audio	Microphone	Audio Output Device
8	Deployment	Developer Code	Executable File

Event #	Summary of BUC
1	A grid system that simulates the playing board.
2	Built-in combat system.
3	A class that all playable units are modelled by.
4	Properties of terrain types.
5	A collection of sprites used for terrain types, and playable units.
6	A playing board with all sprites added.
7	Sound effects and background music.
8	To make Blaze Brigade playable on all operating systems.

2.1.3 Individual Product Use Cases

Product Use Case	Description of Use Case
Playing the Game	The user plays the game and has a sense of enjoyment.
Stimulating the Brain	The user thinks about the game at a deeper level. This causes the player to exercise and train their brain.

2.2 Functional Requirements

The following contains the functional requirements of the project:

2.2.1 GUI

1. The GUI will be controlled with mouse input
2. The game will contain a main menu on screen allowing users to play game, read how to play, or exit upon launch
3. You will be able to select New Game from the main menu
4. You will be able to select How-To-Play from the main menu
5. You will be able to select exit Game Game from the main menu
6. You will be able to select playable units, as well as navigate the unit drop down menu and confirmation buttons to play the game.

2.2.2 Game Structure

1. The game shall be turn-based
2. The game will consist of 2 players alternating turns
3. Each player will receive the same amount of units, and be unable to gain anymore until the next match.
4. During a unit's turn, clicking a unit will give a drop down menu with available actions
5. A unit will be able to perform each of these 4 actions available from drop down menu once per turn:
    (a) Move A unit will be able to move
    (b) Attack A unit will be able to perform an attack on an enemy unit
    (c) Item A unit will be able to equip a (different) weapon from the item drop down sub-menu
    (d) Wait A unit's turn will end and not allow further control of the unit until next turn
6. A unit shall only be able to move and attack once per turn
7. One side will be victorious when the other side has no playable units left

2.2.3 Unit Movement

*Refer to 5.1 for details on 'move'

8. Units will be able to select move as an available option after clicking on a unit that has yet to perform its action
9. Units will only be able to move within their move range as indicated by blue highlighted tiles
10. Clicking on a movable tile will result in a unit moving from their present tile to clicked tile
11. Units shall not be able to pass through obstacles

2.2.4 Unit Attacking

*Refer to 5.1 for details on 'attack'

12. Units will be able select attack as an available option after clicking on a unit that has not attacked this turn, regardless of having moved or not
13. A unit will only be able to attack an enemy unit within its attack range (these tiles will be highlighted in red)

14. Attack information displaying damage, hit rate, and crit rate will be displayed to screen upon clicking on an enemy unit within range after selecting attack
Refer to 5.3 for details on how damage calculation is dealt

15. An attack confirmation button will appear after attack information is displayed. Selecting attack again will result in the attack action be executed

16. Units will counterattack upon receiving an attack as long as they survive and is within attack range

17. Units will lose HP according to damage calculation.

18. Units that fall below 1 HP will be deceased and no longer be able to be used in the current battle.

19. Player shall be able to select which weapon each unit uses to perform an attack.

20. Units will be unable to move after attacking.

2.2.5 Unit Structure

21. All units shall have a corresponding Unit type class.

22. Classes shall include:

    (a) Warriors, which will be physical oriented units with a focus on Strength and Defense.
    (b) Archers, which will be a ranged physical oriented unit with a focus on Strength, Skill, and Speed.
    (c) Mages, which will be a magical oriented unit with a focus on Magic and Resistance.

    *Refer to 5.2 for details on each class

23. The stats of the game shall include:

    (a) Strength stat will be used to calculate physical damage.
    (b) Intelligence stat will be used to calculate magical damage.
    (c) Defense stat will be used to calculate physical defense.
    (d) Resistance stat will be used to calculate magical defense.
    (e) Skill stat will be used to calculate hit, miss, and critical rates.
    (f) Speed stat will be used to calculate if a unit can attack twice.
    (g) Health stat will be used to measure how much damage a unit can take before dying.

24. All units will have a unique corresponding HP bar.

3 Non-functional Requirements

3.1 Look and Feel Requirements

3.1.1 Appearance Requirements

The main menu page will consist of a “nature-like” background offering a soothing and melodic feel. The in-game map will be a wilderness theme with rocks, trees and rocks as obstacles.. The characters to be controlled will be drawn and coloured in such a way that they do not blend into the background. The resolution of the game will be preset and unchangeable from a 960x640 resolution.

3.1.2 Style Requirements

The style of the game will be a pixelated tactical RPG, similar to classics like Fire Emblem, and will have a nostalgic appeal to those whom are familiar with this genre of old school RPGs.

3.2 Usability and Humanity Requirements

3.2.1 Ease of Use Requirements

The game shall be playable by persons who are capable of thought and are able to operate a pointing device (ie. mouse or touchpad).

3.2.2 Learning Requirements

The user shall be able to operate a computer or pointing device.

3.2.3 Understandability and Politeness Requirements

The user shall be able to operate a computer or pointing device.

3.2.4 Accessibility

The game shall be playable on Computers running Windows, with a screen size greater than 960x640.

3.3 Performance Requirements

3.3.1 Speed and Latency Requirements

The response between the game and user input should be near instant.

3.3.2 Safety-Critical Requirements

The game will not contain content that causes harm to the user, including traumatizing experiences. The game will also not allow player input to modify or change the code in any way or form.

3.3.3 Precision Critical Requirements

The gameplay combat statistics shall accurately reflect what is displayed on the screen. Changes in these statistics shall also be accurately reflected upon proper user input.

3.3.4 Reliability and Availability Requirements

The game shall be available for as long as the user has access to a Windows machine.

3.3.5 Robustness or Fault-Tolerance Requirements

The game shall run for as long as the machine is on, or until the user exits the game.

3.3.6 Capacity Requirements

The game shall be able to support gameplay exactly 2 players.

3.3.7 Longevity Requirements

The game shall be playable for as long as Windows OS supports running executable.

3.4 Operational and Environmental Requirements

3.4.1 Expected Technological Environment

The user shall be able to access and navigate the game from a computer running Windows.

3.4.2 Productization Requirements

Before the product is playable, it must be deployed through Visual Studio 2015 and XNA Game Studio.

3.4.3 Installability Requirement

The game will run without requiring any installations upon clicking the game executable (.exe).

3.5 Maintainability and Support Requirements

3.5.1 Maintainability

The game should be ready for launch without need for further maintenance.

3.6 Security Requirements

3.6.1 Access Requirements

The game shall be made available to the general public through both gitlab and github. The public shall be able to access the game given that they have previously acquired Internet access as well as access to a machine capable of running the game.

3.6.2 Integrity Requirements

The game shall not accept invalid user input, nor have the source code modified by any persons or apparatus who are not permitted to do so.

3.7 Cultural Requirements

The game shall not contain any form of content that may be found offensive to any religious or ethnic group. The game shall state that any resemblance in game to real people or events is purely coincidental.

3.8 Legal Requirements

The game shall comply with the law.

3.9 Health and Safety Requirement

The patterns of lights from the game screen can result in epilepsy seizures and, or blackouts to a very small percentage of certain individuals. This may occur in individuals with no past history of seizures of blackout.

4 Project Issues

4.1 Open Issues

The developers of Blaze Brigade must become familiarized with the C++ programming language and investigate how to efficiently adapt to the specific coding standard and testing methodologies. With the use of new technologies, such as a XNA Game Studio, additional time would need to be allocated to learn this tool in depth. In addition, a feasibility study is yet to be completed to indicate whether a mouse-only control would be a feasible for all the possible scenarios present throughout the game play.

4.2 Off-the-Shelf Solutions

Blaze Brigade makes use of Tactics Heroes, an open source program that motivates the creation of a tactical simulation role-playing game. The engine streamlines the process of implementing the framework, and the developers of Blaze Brigade would simply need to implement maps, units, characters and the storyline. Rather than re-inventing the framework, we can use the pre-existing model of Tactics Heroes, and simply polish it in order to fulfil the enhancement requirement of the project. In addition, the open-source program presents a software license agreement and encourages developers to contribute to the program by getting involved within the forums.

4.3 New Problems

No new problems are tracked at this stage of the project.

4.4 Tasks

4.4.1 Project Planning

The game shall be completed using the prospective timeline:

Task	Resource	Start Date
Problem Statement	Developers	Sept 20
Development Plan	Developers	Sept 23
Requirements Document - Revision 0	Developers	Oct 3
Test Plan - Revision 0	Developers	Oct 21
Design Document - Revision 0	Developers	Sept 23
Development of Blaze Brigade	Developers	Oct 31
Testing of Game	Developers	Dec 1
Debugging of Game	Developers	Dec 1
Survey Round 1	Client	Dec 12
Implement changes from survey feedback	Developers	Dec 13
Survey Round 2	Client	Dec 20
Implement final changes from survey feedback	Developers	Dec 21

4.4.2 Planning of the Development Phases

The tasks listed in the table above will serve as the main guide for the developers to ensure that deliverables are completed in correct succession. In that regard, the output will serve the purpose to ensure that all of the requirements have been met before starting the next iteration of development phase. In addition to the development phases, documents such as Problem Statement, Development Plan, Software Requirements Specification, Design Document, and Test Plan would need to be created and updated concurrently for the stakeholders as the project proceeds.

4.5 Migration to the New Product

Not applicable at this stage of the project.

4.6 Risks

The risk associated with the project comes with the nature of developing a game. For instance, accommodating testing into the game play would be a time-consuming and trivial task. Hence, the investigation on how to proceed with a specific testing standard and whether to introduce an automated testing framework is still underway at this stage of the project. In relation to the design aspect, a feasibility study is yet to be completed to indicate whether a mouse-only control would be feasible throughout all the scene of the game play. Another risk is that while the code structure for each module has been designed to add future additions seamlessly, there is a slight chance newer additions might conflict with existing implementations. In addition, extra time would need to be allocated in the process of learning and adapting to new technologies to ensure all of the objectives are met in the constrained time frame.

4.7 Costs

There is no cost associated in the development of the product. The developers of Blaze Brigade will make best use of their personal computers for the main development environment, with the addition of already-owned tools such as XNA Game Studio and Adobe Photoshop.

4.8 User Documentation and Training

4.8.1 Training Requirements

Readable instructions will be available to the player at the beginning of the game, and will carefully instruct the user on how to play Blaze Brigade.

4.9 Waiting Room

There are a few extra features that would increase the entertainment value of the game but have yet to be implemented. One major addition would be an AI that a player can face, instead of the game always requiring 2 players. Another would be another game mode called "Capture the Flag", where you would attempt to capture objective flags on the map, and bring it back to your base without that unit dying and losing the flag. Unit level increasing is also another feature that will be implemented in the future, as well as a larger variety of unit types. An additional feature to be implemented will be more weapons types as well as weapon animation during attacking. A further feature would be adding different tile weights, such as costing 2 movement to walk through a water tile. The last extra feature would be a playable tutorial upon starting the game replacing the screen of instructions on how to play.

4.10 Ideas for Solutions

The path finding algorithm used for unit moving and attacking currently is BFS. However this would only work since we do not yet account for weighted tiles (costing more then 1 movement to go through a tile). This would turn it into a weighted graph, and we would need to instead switch to Dijkstra's algorithm for path finding.

5 Symbolic Parameters

5.1 Action Terminology

5.1.1 Moving

Moving refers to selecting move from a selected unit, then clicking to one of the blue highlighted nodes to move the unit from its previous location to the location newly clicked.

5.1.2 Equip weapon

Equip weapon refers to selecting a different weapon from the Item drop down menu, which in turn changes your unit's stats for damage calculations.

5.1.3 Attack

Attack refers to selecting attack from the character drop down menu, and clicking on an enemy unit standing on a red highlighted node. The unit will then deal damage to the enemy unit based on damage calculation then be counterattacked by that same unit should it survive.

5.1.4 Killing a unit

Killing a unit refers to a unit getting attacked and having its health being reduced to under 0. That unit is then "killed" and no longer playable or seen in the current game.

5.2 Unit Terminology

For unit structure, refer to 2.2.5 in functional requirements.

5.2.1 Warrior

Warriors are a frontline physical bruiser and tank. They can equip swords, move quickly, and attack units within adjacent range. Their high attack and defense allows them to kill archers and mage easily once in range, but can fall prey to mages due to their lower resistance.

5.2.2 Archer

Archers are a backline ranged physical attacker. They can equip bows giving them an attack range of 2 or 3. Their long range is compensated by the fact that they cannot counterattack against adjacent units. Their high attack and skill allows them to accurately land high physical attacks with a potent critical rate. This makes them the ideal unit for killing off mages, although they struggle against warriors.

5.2.3 Mage

Mages are a backline ranged magical attacker. They can equip different spell tomes giving them an attack range of 1-2. Although their mobility is limited, their extremely dangerous magical attacks allows them to easily kill off warriors, despite struggling slightly against archers.

5.3 Combat Terminology

For attacking with units, refer to 2.2.4 in functional requirements.

For unit x = attacking unit, and unit y = defending unit where:

- unit.skill gets the unit's skill
- unit.str gets the unit's strength
- unit.int gets the unit's int
- unit.def gets the unit's defense
- unit.res gets the unit's resistance
- unit.spd gets the unit's speed

5.3.1 Hit Rate

The % that a unit will land an attack.
hitRate calculation= [(x.skill/10 - y.skill/10)+1]*0.8 *100,
where hitRate is 80% if both players have the same skill, and adds or subtracts 8% to hitRate in favour of the unit with the higher skill stat.

5.3.2 Critical Rate

The % that a unit will perform a critical attack (attack that deals x2 damage).
critRate=[(x.skill/3 - y.skill/3)+1]*0.81*100,
where critRate is 10 if both players have the same skill, and adds or subtracts 3.33% to critRate in favour of the unit with the higher skill stat.

5.3.3 Physical Damage

The amount of damage a physical unit deals should an attack hit before modifiers.
physicalDamage = x.strength - y.defense,
where physical damage is the damage calculated from a physical attack.

5.3.4 Magical Damage

The amount of damage a magical unit deals should an attack hit before modifiers.
magicalDamage = x.int - y.res,
where physical damage is the damage calculated from a magical attack.

5.3.5 Number of Hits

The number of attacks a unit will perform when attacking.
numHits = 2 if x.spd is greater than y.spd+4, else numHits = 1.

5.3.6 Damage Dealt

The amount of damage actually inflicted on an enemy unit when a unit performs an attack.
damageDealt = (physicalDamage or magicalDamage) * numHits * (hitRate, which will be 1 if the attack hits or 0 if the attack misses) * (1.5 if it is a critical hit or 1 if it is not a critical hit).