Software Requirements Specification

For

Hero

Prepared by Nathan Harvey
Nathan Paul
Jacob Pearse
Kyle Spahn
Chris Steinberg

January 26, 2006

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Kyle Spahn	1-29-06	Compiling everyone's sections	1.0
Jacob Pearse	3-28-06	Updating for finished game implementation.	2.0

1. Introduction

1.1 Purpose

Hero is a turn based strategy game where the objective is to defeat an opponent by killing the other player's hero and destroying all the structures the player controls. This game is a mix between Heroes of Might, Magic and the popular board game Risk. The game is two-player, network based, and the board will consist of hexagonal tiles the player can select and click on. Game play consists of: attacking enemy units, upgrading/building cities or outposts, taking control of tiles, casting spells on enemies with the hero, troop movements, building walls, gathering resources, etc. The game play as well as the details and specifications of the functionality of this game are defined in sections 3 and 4. An overview of the game is given in section 2, and a list of requirements is given in section 5. This documentation will be for the final version of the game.

1.2 Document Conventions

This document follows the IEEE standard formatting for software development. The standard defines a regular formatting this document follows including writing to be done in third-person, passive voice as well as readable and grammatically correct text.

1.3 Intended Audience and Reading Suggestions

This document is not intended for the end user because it provides a detailed specification of how the software is to be implemented. Since a user needs information on how to play the game, instead of how to make the game, this document is more geared towards testers and mostly the developers of the game. The document starts off with an overview of the functions and specifications for this game in section 2, then moves on to describe the requirements for interfacing with external hardware and software in section 3. Section 4 describes the game functions in great detail and section 5 lists various requirements the game must adhere to after completion. It is suggested that all audiences of this document start with section 2 first to get a general idea of the software requirements. Testers should next read sections 5.1 through 5.4 (performance, safety, security requirements and software quality attributes). This is to get an idea of how the game will affect them and the system they are running it on, as well as the aspirations for quality. Next a tester should read section 3.1 (user interfaces) followed by all of section 4 (system features). Reading the document in this order will give the tester an idea of what to expect in the interface at first glance, and then they may test all the individual functions to make sure they adhere to the specifications.

After reading section 2, game developers should read the remaining sections in order because this document was designed specifically for the purpose of developing the game. The developer needs to get an overall idea in section 2 of the game. Then, how it needs to interface with everything else in section 3 (so they have an idea of what tools to use and possibly how they should use them). Section 4 is the most important to a developer because it describes all the functions of the game in great detail and it will help with making decisions in writing actual code for the game. Section 5 is considered least important but the developer should still read it to make sure their game has adhered to the given ideals.

1.4 Product Scope

Hero is a turn based strategy game developed on a Java platform. It has a game board, supports network play, and is designed for one on one competition. The objective of the game is to destroy the opponent by killing the opponent's squads, hero and buildings. Each person has control of a hero, army, and any buildings they own. They can manipulate these objects to destroy their enemy, fortify their own resources, and execute strategic maneuvers to gain control over the board.

The game is designed to be as easy to use as it is fun for the player. It is intended to run on any computer that can use Java and be easily runnable by a computer with average processing power. The on-line stats browser is intended to promote competition and thus increase the excitement of the game.

1.5 References

N/A

2. Overall Description

2.1 Product Perspective

Hero is a turn based strategy game. It is an implementation of a basic 2D graphics engine in the Java runtime environment. It is an original mixture of some popular games that have entered both the video game and board game markets. These examples can include:

Hero's of Might and Magic: A turn based strategy game made for the PC. Set in mythical mid-evil times, you take on a role as an empire's ruler, with valiant hero's who lead squads of units from your home territory into enemy lands in some quest to defeat your enemy and to gain territory for your empire. Much of the game consist of moving your hero's about the map, upgrading your cities, and building units using resources collected from captured resource points. Also Hero's of Might and Magic has implemented a battle engine, where your squads units move about a battle field and take turns attacking each other in an initiative based movement system. Victory is generally achieved by capturing all of your enemy's cities or completing some level objective set forth in a story mode.

Risk: A turn based strategy board game of world domination. Set in the 1700's, Risk places you as a ruler of a semi-modernized world. You have control of units that are placed on countries that you control, the amount of units you receive is based on the amount and location of the countries/territories that you control. Battle occurs when one player declares an attack on an opposing player whose territory is adjacent to the attacking players. Battle is resolved through a series of dice rolls that determine a battles victor, and the defeated player looses units based on the degree of victory. The game is won by whoever gains control of the entire game board.

Hero is going to be based heavily upon the concept of Hero's of Might and Magic (HMM), as it will be based in a mythical mid-evil setting. You will control one hero, a feature that differs from the HMM franchise. This hero will be your primary unit in the game. He is able to form squads of units with him to fight enemies, and he is the only unit in which you can capture enemy or neutral territory tiles. He is also able to cast spells that can improve the quality of your units or diminish the quality and strength of enemy units. You also control cities and outposts, similar to HMM, these units can produce units to fight for you and act as strong holds on your territory. They protect tiles, and units may be fortified in these cities and outposts to increase the defensive ability of the buildings. Battles will be determined automatically, as it will be too difficult to implement a battle engine with moving units and attack animations. This is also a big difference to the HMM franchise. To be victorious in the game, you must destroy the opponent's hero and buildings. Hero is a game of board domination and conquering your opponent.

Hero will be implemented as a Java Application. The organization of the game system will stem from a base GameCore class that is responsible for handling input, through the use of an InputManager class, and it is also responsible for managing the display. This is performed using a ScreenManager class. All game play elements are extensions of a basic Sprite class that holds Animations and GameActions that can be customized for each unit. To optimize network connectivity, all of the game components will be local on both the host and the client.

2.2 Product Functions

Major functions that Hero must perform for its end user is as follows:

- Network communication – Hero must be able to both host a game, listening for connections from remote clients, and join a game hosted on a remote system.
- Innovative game play – Hero should be innovative, mixing popular computer based game systems, with popular board based game systems.
- Unit selection – Hero players must be able to select units in the game to view information as well as perform functions based on their input.

2.3 User Classes and Characteristics

Hero should be designed for ease of use, thus any user class that accepts gaming as a past time should be able to sit down and play. Interfaces and options should be simple, descriptive and easy to navigate. Simplicity of Hero's feature set will allow for any generic user to play the game, but should provide a significantly challenging game system for advanced players.

2.4 Operating Environment

Since Hero is being developed in the Java runtime environment, it is very cross platform compatible. Users on a PC, Mac, or some UNIX operating system shouldn't have a problem running the software. It is being developed in Java 1.5, and thus the user is required to have a compatible version of the Java Runtime Environment installed on their system, which can be found free, at java.sun.com.

2.5 Design and Implementation Constraints

Since Hero is being designed and implemented in a single semester as a project for Michigan Technological University, Team Software Project, it is possible that time is the most limiting factor in this development cycle. This time constraint may require this team to scale back the project from its currently proposed scope. There are no other limiting factors though, as there is really no cost to development of such a product. Another constraint is that Hero is supposed to be a network game, but nobody in our group has taken any computer network courses. So our network coding may take longer than expected, since our project is much larger than most projects developed in this class.

2.6 User Documentation

The Hero website will contain all of the user documentation, as well as a digital user manual in the Jar file that is released at the completion of the development cycle. This will contain the game rules as well as a tutorial for running Java applications on a variety of platforms.

2.7 Assumptions and Dependencies

We will be depending upon some previously written code for a basic game engine to run as a back end for the game system. We are also assuming that Java will be able to handle the slow updates and minimal network communication that we are requiring for our multi-user game. It is felt that Java will be able to support this but further testing must be performed to determine whether or not we will be capable of producing the complexity that we are proposing in the Java environment.

3. External Interface Requirements

3.1 User Interfaces

The user interface in Hero will appear as a board game composed of hexagonal tiles. The tiles for the

board will be hexagonal so that there will be six adjacent tiles next to every tile that is not on the outside of the board. The user will be able to see certain things that are on these tiles. If the user clicks on them, he will be able to view more specific information about the objects on the tile. Certain tiles will allow further interaction because they have objects on them. Units, cities, buildings, walls, and outposts will all allow further interaction. If a user clicks on one of these objects on the tiles, a menu will be presented in the unit's options panel on the screen that will allow the user to take a certain action with the tile. For instance, if a user clicks on his units during his turn, he can move them to a different tile or attack nearby enemy units. The options menus will have a different set of options present depending on what is clicked on. There is also a chat menu that can be used anytime during the game. The chat will be displayed in the bottom-right of the screen and will be displayed in a terminal window. There will be certain commands available by typing them into the terminal chat box. For example, "/q" will quit the current game.

3.2 Hardware Interfaces

Hardware Interfaces will include a mouse, the keyboard, and the display monitor. There is not much heavy hardware needed to run the game other than a simple computer with Java 1.5 and a monitor. The mouse left click will allow the user to interact with certain objects. The mouse right click, on the gameboard, will allow the user to move the view of the gameboard around by holding the button down and then moving the mouse to scroll the view on the gameboard. The keyboard is used for mainly chat and simple terminal commands.

3.3 Software Interfaces

In order to run the game, the user will need to have version 1.5 of Java Runtime Environment (JRE) installed on his computer. This software is needed because the program is written in java. If the user does not have this software he can go to http://java.sun.com/j2se/1.5.0/download.jsp and download it from there.

3.4 Communications Interfaces

In order for a user to start a game he will need to have an internet connection established. He can then select join game on the main menu and select a game to join from the list of games that are waiting to be started. If the game has a password, the user will need to know the password to get into the game.

The user can also start a game by selecting start game as long as he has an internet connection. Then, the user must fill in a user name, game name, game password (optional), port number and select a map to play on.

4. System Features

Our system features are covered in-depth in separate appendices for scenarios and a class diagram.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The game must be able to run on a minimum of Window's 2000 and Linux. It is recommended to have at least 256 Mb RAM, although 512 Mb is preferred. The graphics card should have at least 64Mb of on-board memory.

5.2 Safety Requirements

As with any computer game, there is a risk of epilepsy. If you have been known to have epilepsy, talk to your doctor before playing this game.

5.3 Security Requirements

This game will not gather any private information from people through the client-server connection. We have to make sure that the connection is as secure as possible to reduce the risk of attacks. The only information we gather from the user is the IP address.

5.4 Software Quality Attributes

This software must be robust and as bug-free as possible to ensure the players have a positive experience. The game should be easy for a beginner to pick up and get started, with a minimal learning curve. The game should be flexible enough to allow the easy creation of additional content, while preserving the ease of use to the consumer. Finally, and least importantly, the game should be portable to the systems specified in section 5.1.

5.5 Business Rules

It is the policy of the development team to follow all codes of conduct established by the University.

6. Other Requirements

Appendix A: Glossary

N/A

Appendix B: Analysis Models

A class diagram is available online.

Appendix C: Scenarios

Reference scenarios for game play specifics.