CSCE 431 - Project Software Requirements Specification

Team BMORPG (Barely Multiplayer Online Role-Playing Game)

Brandon Adame

James Freyman

Josue Martinez

Stephen Simmons

Luke Yeager

February 8, 2012

Jaakko Järvi, Instructor

Tolga Cifti, T.A.

1. Intro

1.1 Purpose

Our purpose is to create a game in which multiple users can interact in a competitive and social environment. The players will connect to a server that pairs clients together in a secure and reliable fashion. The server also calculates all in game interactions to facilitate fair game play.

The game is intended for players of age ten or more who are interested in a game with a strategic, competitive environment.

1.2 Scope

The game implements two primary software products: the Client and the Server.

The Client serves primarily as a GUI for interaction with the user. The client takes input from the user and transmits it to the server. It is not expected to store any character information, ensuring validity of information.

The Server connects all clients together and keeps track of multiple games simultaneously. Once paired, players rely on the server for in-game calculations, and when the game is over, the Sever notifies the Client that they may update their character statistics.

The game is primarily intended to provide place where multiple game players can battle each other and eventually increase their character's skill.

1.3 Definitions, acronyms, abbreviations

Client – Program running on the player's computer. The Client sends the Player's commands to the Server.

Server – Program running on the server. The Server is responsible for connecting Clients together to create a game, and for monitoring the game while in progress.

RPG (Role Playing Game) - A game in which players assume the role of their characters in a fictional setting. Players can increase their player attributes through integrations and battle in the virtual world.

MMORPG (Massively Multi-player Online RPG) - An RPG based in a virtual environment that allows a huge number of players to take part in the game.

2 Overall description

2.1 Product perspective

- System Interfaces - This product is self-contained and operates independent of any larger systems.

- User Interfaces - The client side application will be the main interface for users. It needs to include a page to customize characters, and the option to be matched to another player to battle. Once in a battle, users need to pick a move from a set list, or change their weapons according to what they have in their inventory. There may also be a feature that allows two battling users to chat, in which case a chat window will be needed to display messages and a field to take user input.
- Hardware Interfaces - The server will be constantly listening over a certain port waiting for clients to log in (exact port numbers will be determined at a later date). Once users are paired to battle, they will communicate with the server on a certain port. Each game thread will use a different port.
- Software Interfaces - This application is designed to run on a Windows operating system with .NET Framework 4 installed. We will use a SQL database to store player and game data.
- Operations - In the case that a user gets disconnected during a battle, he will forfeit the game. It is possible that a feature will be added in the future to allow a user to reconnect within a certain amount of time.

2.2 Product functions

- Network Connection - Throughout the game system, secure sockets will be used. We aim to design a system that cannot be easily hacked, but which offers real-world security. This is especially important when sending login passwords over the network.
- Client connection - When the Client program starts up, players have the choice to either create a new account or log-in to an existing account. When logging into an existing account, the Client simply sends their username and password to verify their identity, then the Server either accepts or denies the request. When creating a new account, the Client sends all the information necessary to specify a new account, and the Server creates the account, then notifies the Client.
- Pairing players - The Server will pair players together by looking at their levels and trying to create the fairest game possible from the list of currently connected Clients.
- Player characteristics - After each game, the Player has the option to upgrade their player characteristics. The amount of points they have to spend on new weapons and abilities depends on their level, the level of their opponent, and the outcome of the match.

2.3 User characteristics

This game is intended for all players of at least ten years of age with a computer and access to the Internet. There is no preexisting technical skills or knowledge required to play the game.

2.4 Constraints

This project is intended to be an academic exercise, demonstrating that we can design a system of programs with secure reliable network communication. We do not intend to develop a system which can handle a large number of concurrently connected clients. Therefore, we will be using a personal computer for the server rather than spending money on a production-level server.

- Regulatory policies - At most 130,000 players can simultaneously be participating in battles.
- Hardware limitation - The number of available ports limits the total number of players that can connect to the server.
- Interfaces to other applications - In addition to the server and client interfacing, the server portion of the code will interact with a SQL database.
- Parallel operations - Since each game is independent from the rest, the Server will allocate a new thread for each game. In this manner, each game will run concurrently and will not affect the run-time performance of any other game.
- Signal handshake protocols - We will be using TCP sockets to ensure reliability of data transfer. The handshake procedure will be clearly defined in documentation, with all network communication being sent as JSON objects.
- Reliability requirements - The client program needs to be able to handle the loss of connectivity to the server (server goes down/upgrading components). The server needs to handle the loss of connectivity to a client without crashing the game for those two players. The loss of a single game should not cause the entire server to fail.
- Safety and security considerations - The security of the server needs to be hardened and verified so that an attacker cannot deny service to clients, obtain control of the server, or gain access to the database of users.

2.5 Assumptions and dependencies

- The project assumes all code will be executed on a Windows platform with .NET Framework 4 installed as the client will be written in C# using Windows Presentation Foundation.
- Software also assumes access through any given firewall to allow for outgoing socket connections to the server.
- Client software requires there is an active server to connect to. Client has no standalone functionality.