

Project: Jarl
Requirements document
Group Number: 18

Magnus Andermo
Mattias Frånberg
Carl Johan Gustavsson
Pontus Stenetorp

Contents

1	Preface	3
1.1	Readership	3
1.2	Version history	3
2	Introduction	4
2.1	Who are the users and what problem does the system solve for them?	4
2.2	The main uses of the system.	4
2.3	The context/enviroment in which the system is to be used.	5
2.4	The scope of the system.	5
2.5	The main factors that need to be taken in to account when designing and building the system.	5
2.6	Technologies and Risks.	6
3	Glossary	8
4	User requirements definition	9
4.1	Functional requirements	9
4.1.1	General	9
4.1.2	Lobby	9
4.1.3	Gameplay	11
4.1.4	Hero actions and attributes	14
4.2	Non-functional requirements	15
4.3	Use cases	16
4.3.1	LOBBY SYSTEM: CREATE A GAME SESSION	16
4.3.2	LOBBY SYSTEM: JOIN A GAME SESSION	16
4.3.3	DESTROY A CASTLE	17
4.3.4	MOVE THE HERO	17
4.3.5	USE A HERO'S POWER	18
4.3.6	CONSTRUCT A BUILDING	19
5	System architecture	20
6	System requirements specifications	22
6.1	Functional requirements	22
6.1.1	General	22
6.1.2	Lobby	23
6.1.3	Gameplay	25
6.2	Use cases	32
6.2.1	STARTING THE GAME.	32
6.2.2	CREATE A GAME SESSION	32
6.2.3	JOIN A GAME SESSION	33
6.2.4	SEARCH FOR A GAME SESSION	34
6.2.5	REGISTRATION	34
6.2.6	CONSTRUCT A BUILDING	35
6.2.7	ATTACKING WITH THE HERO	36
6.2.8	USING A HERO POWER	36

7	System evolution	38
7.1	The fundamental assumptions of the system	38
7.2	Planned changes	38
7.3	Anticipated changes	38
8	Appendices	39
8.1	User database	39

1 Preface

1.1 Readership

This document is intended for stake holders, project managers, system architects and system developers.

1.2 Version history

04-01-2008 First complete requirement document version to be handed in and available to the public.

2 Introduction

2.1 Who are the users and what problem does the system solve for them?

The users are primarily male and 10 to 25 years of age, players of this age and gender are usually drawn towards more intense games. We do not exclude female players but choose to focus on male players. The users have previously played action-oriented strategy games and play games at least one hour a day. Their problem is that there is no modern mix between action, roleplaying elements and strategy. Games such as these are too old (six years old or more) and no new games which support the same kind of online multiplayer gameplay have been released. The users problem is to find a modern game within the genre. Since there is a gap in the genre which we intend to fill and thereby provide value to our users.

2.2 The main uses of the system.

“Bob a 17 year old boy, is bored, he needs something to do to cheer himself up, but he is not interested in playing an advanced online game which takes many hours per day (Bob usually plays for one hour a day) and have a monthly fee. Instead he is looking for some quick action and starts up Jarl, which he previously downloaded from the web. He starts the game and presses “Quick game”, he is the automatically set up with other players also looking for a quick game and is roughly at the same skill level. Bob has played the game before and knows the controls. He plays a tactic which involves carefully balancing defensive actions against the player behind him and attacking the player in front of him. He spends an extensive amount of resources on defensive towers and defensive units so that his hero is free to charge along with the attacking units. He succesfully defeats all other players and when the game ends his statistics and his current ranking is displayed. Bob is pleased with his ranking and having bashed some other players. He shuts down the game and resumes his school work.”

“Jim, who is 24 years old, is bored with his current set of games and is looking for a new one. He wonders if there is a free game out there that might be fun and stumbles upon Jarl, some sort of strategy game with a mix of game elements that looks promising. He sees a set of screenshots but he isn’t sure if he would like to download it. Then he notices a nice “Play now! No download required.”. He clicks the button and since he has Java installed an animated splash-screen loads. The game starts and he is asked if he wants to have hints shown during game play. Jim answers yes, since it’s his first time. He then chooses quick game. He finds the game enjoyable since it’s easy to learn and has a fast-paced gameplay. After the game he is shown his ranking and asked if he would like to register in order to save the ranking. He enters his email address and a username and is registered and then shuts down the game. Jim has not played a online multiplayer game with this mix of action and strategy gameplay since his late teens.”

“Alex 21 years old and Ethan also 21 years old, talk to each other on an instant messenger and Alex invites Ethan to a game of Jarl. They have both downloaded and installed the game and starts up Jarl using it’s icon on the

desktop and Alex uses the “Start game” function to create the game, she also sets a password in order to keep the game private. Ethan then simply uses the “Search game” function to find any game Alex has created. He joins the game after entering their password. After playing they both shut down the game and resumes their conversation on the instant messenger.”

2.3 The context/enviroment in which the system is to be used.

The system will be used at home, at school and at internet cafes. An internet connection is required since the game is played online. Since we will be using Java the game will be portable between operating systemms and run on most personal computers that supports Java, but we will limit ourselves to Windows XP, Mac OS X Leopard and Ubuntu running Java version 1.5.

2.4 The scope of the system.

Topic	In	Out
“Quick games”, under 30 minutes	X	
Ingame music.		X
Sound effects		X
Support for Mac OS X Leopard, Windows XP and Ubuntu	X	
Support for up to 8 players per game	X	
Support to play on cell phones		X
Player ranking system	X	
Game lobby, where players can find games.	X	
Units controlled by artificial intelligence	X	
Optional install, can be run using Webstart	X	
“Get started”, tutorial	X	
Allowing a private local area network server		X

Table 1: Table of the scope of the system

2.5 The main factors that need to be taken in to account when designing and building the system.

- We don’t have more than 15 seconds to hook a potential player, this must be considered. Java Webstart is a must, users are not willing to download some game they see for the first time. We need to get them hooked without a download and then, get them to download it after a game or two since the downloaded version will load much quicker which will be pointed out in the Webstart splash screen.
- We need keep players hooked, in order to make a succesful game you need people to play it over and over again. People are competitive by nature, and love to rank themselves among others. By presenting their current ranking and their ranking relative to others we can encourage them to

play more and play a few times a day. This is essential for a long term success.

- Keeping our limited resources in mind, we need to use an extensive amount of API;s written by others. The more we can borrow, the more we will be able to focus on what we really care about, the success of the project. There exist 3D engines, off-the-rack game servers and so on. We need to glue them together.
- Using a new way of presenting graphics, we may reduce the amount of artwork, the amount of game logic and give the game a unique look. We will present the graphic using 2D layers in a full 3D enviroment using a Z-axis that only allows you to move between a few levels of depth. While giving the user a sensation of 3D it allows us to use 2D textures for animation which greatly eases the burden on the artist and also simplifies the game logic.

2.6 Technologies and Risks.

- **Java 1.5**, we intend to use Java as our programming language. This is because we all have previous experience with Java and it will lead to few new surprises. Also, there are third party libraries to suit all our needs, they are mentioned below.
- **Webstart**, Java Webstart enables you to start a Java application from the web without limiting the application. Pontus has previous experience with Java Webstart.
- **JInput**, Java has a platform independent input system, unfortunately, it's not made for games. JInput solves this but is limited to a lower number of platforms. JInput has support for Windows XP, Mac OS X Leopard and Ubuntu and will therefore be an excellent choice to solve the problem with input. Pontus has previous experience with JInput, but the rest of the group has no previous experience.
- **Eclipse**, is a cross-plattform integrated development enviroment. We all have previous experience working with Eclipse.
- **OpenGL**, is an open graphics library. OpenGL is cross-plattform and is a part of several Java graphics suites, such as JMonkey, Java3D and LWJGL. Pontus has previous experience with OpenGL and Magnus has previous experience with 3D programming in general.
- **Concurrent Versions System**, we intend to use CVS to manage our code. We all have previous experience using CVS.
- **Painter 9**, in order to produce graphics for the game we have chosen Painter. Painter is easy to use and we have previous experience with it.

Risk	Probability	Effects
Third party libraries can't deliver the performance expected	Medium	Rework of design and the need to spend additional time on work arounds.
Third party libraries prove to be more complex and take more time to learn	High	Additional time need to be spent learning the API;s
Java has performance issues	Low	Java performance is too low to handle the game logic and additional time need to be spent reworking the code in order to remove bottlenecks.
Webstart is too slow	Medium	Might result in Webstart not being used at all or additional time spent on improving performance and looking into Webstart workarounds.

Table 2: Table of risks and effects

3 Glossary

Artificial Intelligence, a system that makes decisions based on experience and is able to acquire new knowledge. In this case, a computer controlled unit.

CPU, Central Processing Unit.

Game session, starts the moment the game world is shown and ends when the user returns to the lobby. When a game session is *started* the no more players can join the game session and the players can interact with the game world.

Game world, the 3D world in which the game session takes place.

Hero, a game element, whose movement is controlled by the user and can interact with other game elements.

Hero powers, a set of skills the hero can initiation to damage nearby units.

Java, a programming language developed by Sun Microsystems, Inc.
See <http://java.sun.com>.

Java Runtime Environment, the program required to run programs written in Java.

Java Webstart, a technology to enable users to start Java programs from a web browser.

JRE, see Java Runtime Environment.

Mac OS X Leopard, an operating system developed by Apple Inc. For more information see <http://www.apple.com>.

Mana, an energy that the hero needs to perform hero powers.

Micro-management, a term to describe very detailed management of units and resources.

OS, an abbreviation for Operating System.

Pentium III, a CPU produced by the Intel Corporation.

Ubuntu 7.10, an operating system developed by Canonical Ltd. For more information see <http://www.ubuntu.com>.

Unit, a game element whose movement is not controlled by the user.

Unit distribution, the ratio of attacking and defending units.

Windows XP, an operating system developed by the Microsoft Corporation.
For more information see <http://www.microsoft.com>.

4 User requirements definition

4.1 Functional requirements

4.1.1 General

1. **MULTIPLAYER**

The system shall use the Internet for multiplayer connections.

Rationale: Most users are at home or at work and wants to play with friends in other locations.

2. **NUMBER OF PLAYERS**

The system shall have at least 2 players per game session.

Rationale: Two users is the minimal amount of users that is required for the system to be a multiplayer game.

3. **KEY BINDINGS**

The user of the system shall be able to change all key bindings. A key binding is an input event bound to a key on the keyboard or the mouse.

Rationale: Users wants customize the key bindings to suit what they are used to in other systems, or they may not like the default key bindings.

4. **RESOLUTION**

The user of the system shall be able to change the screen resolution. The resolution can only be changed when the user is not involved in a game session.

Rationale: The resolution of the system is a major factor on the amount of system resources the system uses. Users also have different monitors that supports different resolutions and they most likely want to run the system in a resolution that suits their monitor.

5. **JAVA WEBSTART**

The system shall be startable via Java Webstart.

Rationale: The user wants to be able to start the system via a web browser.

6. **FAILING NETWORK CONNECTION**

In case of the network connection failing the system shall notify the user. The system shall ask if the user wants to reconnect to the server or terminate the session.

Rationale: The user wants to be informed if the network connection fails.

4.1.2 Lobby

7. **LOBBY**

The system shall provide a lobby in which the user can find game sessions that he wants to participate in.

Rationale: A lobby is needed to make it possible for players to find each other.

8. **CREATE GAME SESSION**

A user shall be able to create a game session.

Rationale: The users wants to be able to set-up a game session.

9. **SET THE NUMBER OF PLAYERS**

A user who creates a new game session shall be able to set the number of players needed for that session.

Rationale: Two players may want to play against each other without any other players interfering.

10. **GAME PASSWORD**

A user who creates a new game session shall be able to set a password that must be entered by other players in order to join that session.

Rationale: The user who creates a game session may only want to play against players that he gives permission to.

11. **JOIN GAME SESSION**

A user shall be able to join a game session.

Rationale: Because the system is a multiplayer game it is an essential feature for the use of the system.

12. **SEARCH GAME SESSIONS**

The user shall be able to search for game sessions in the lobby

Rationale: The user wants a way to look for specific game sessions.

13. **QUICK GAME**

The user shall be able to join a game session that the server chooses. The choice will be based upon the user's statistics and the players currently available for a quick game.

Rationale: The user wants to be able to join a game session without having to explicitly find one.

14. **CHAT**

The user shall be able to exchange text messages with other players in the lobby.

Rationale: The users wants to be able to talk with other users before joining a game session or just to find other players to form a game with.

15. **REGISTRATION**

The user shall be able to register a user account for the system.
An account enables the system to keep track of the user over time.

Rationale: A user account is needed to store information about a user between uses of the system.

16. **USER STATISTICS**

The system shall keep statistics of the number of won and lost game sessions for users with an account.

Rationale: Many users are interested in statistics and comparing themselves against other users.

4.1.3 **Gameplay**

17. **TERRITORY**

The players shall be placed at equal distance from each other
When a player kills another player, he gets his territory, i.e. he can attack the player that the killed player was able to attack.

Rationale: This is a vital part of the game's balance and gameplay.

18. **PLAY ORDER**

Each player shall only be able to attack one player. Each player can only be attacked by one player.

Rationale: This is for the game's balance and gameplay.

19. **CASTLE**

The user shall have a castle at the beginning of the game session. If the castle is destroyed the user has lost the game session.

Rationale: This is the key element for winning or losing a game session.

20. **PRODUCTION BUILDINGS**

The user shall be able to construct at least two types of buildings in his territory to enable him to create that building's specific type of unit.

Rationale: Production buildings are needed to create units. Different units are good at different things and this allows users to use different tactics.

21. **DEFENSIVE BUILDINGS**

The user shall be able to build at least one defensive building that attacks the other players' units and heroes.

Rationale: The user wants to be able to defend himself from the other players to increase his chances of winning.

22. **CHAT**

The user shall be able to communicate with other users via text messages during a game session. The text messages will be seen by all users in the current game session.

Rationale: Communicating during the game session will enable a user to discuss tactics and alliances with other users.

23. **HERO**

Each user shall have a hero that the user can control. The control shall include the movement of the hero. The user shall select a hero to use during the game session at the start of a game session.

Rationale: The user wants to be able to interact with the game world using the hero.

24. **ARTIFICIAL INTELLIGENCE CONTROLLED UNITS**

All units produced in a production building shall be controlled by AI. The user shall not be able to move these units explicitly, he will only be able to select the distribution of his attacking and defending units.

Rationale: The user wants to focus on strategy instead of micro-management.

25. **UPGRADING**

The user shall be able to upgrade his buildings. The upgrade of a building shall improve the units produced by that building. The effect of the upgrade will only apply to units produced after the upgrade has been completed.

Rationale: The user wants to be able to upgrade his units which will increase the probability of the user to win.

26. **MINIMAP**

The minimap shall show the user an overview of the game world.

Rationale: The user wants to gain information about what happens outside his current game view.

27. **WINNING**

The user shall win when the castles of all of the other players is destroyed.

Rationale: There must be a winner to end a game session.

28. **LOSING**

The user shall loose when his castle is destroyed. All of the losing player's territory shall be added to the player who destroyed the castle. Loosing a game session means that the affected user no longer is able to interact with the game session, other than viewing the current game session.

Rationale: In order for someone to win the game session some other player must loose.

29. **LEAVE GAME SESSION**

The user shall be able to leave the game session at anytime

during a game session. Leaving the game session results in losing the ability to interact with the game session and the leaving user is moved to the game lobby. If the game session has not been started, the number of players in the game session will be lowered by one.

Rationale: The user wants to be able to leave the game session if he feels the need to do so, for example, to answer a phone call. The user also wants the number of players to be lowered when a player leaves before the game starts, in order for another player to take the leaving players spot in the game.

30. **MOVE CAMERA**

The user shall be able to move the game camera which allows him to see other parts of the game world.

Rationale: The user wants to be able see parts of the game world other than his initial game view in order to make tactical observations.

31. **DISTRIBUTE ATTACKING AND DEFENDING UNITS**

The user shall be able to distribute the amount of attacking and defending units.

Rationale: The user wants to be able to control the amount of attacking and defending units in order to manage his tactics.

32. **HEALTH POINTS**

Each building, hero and AI character shall have an assigned amount of health points.

Rationale: The user wants to be able to deal damage to buildings, heroes and characters.

33. **ATTACKING**

Each defensive building, hero and AI character shall be able to attack other buildings, heroes and AI characters.

Rationale: The user wants defensive buildings, heroes and AI characters to be able to attack, in order to reduce the hit points of other players buildings, heroes and AI characters.

34. **TAKE DAMAGE**

Each building, hero and AI character shall be able to take damage.

Rationale: The user wants buildings, heroes and AI characters to take damage in order for them to have their hit points reduced.

35. **DYING**

Each unit and hero shall die when their hit points are reduced to zero.

Rationale: The user wants units and heroes to die after taking a certain amount of damage.

36. **DESTROY BUILDINGS**

Each building shall be destroyed when their hit points are reduced to zero.

Rationale: The user wants to be able to destroy other players' buildings.

4.1.4 **Hero actions and attributes**

37. **MOVE HERO**

The user shall be able to move his hero.

Rationale: The user wants to be able to move his hero.

38. **HERO POWER SETS**

Each hero shall have a set of three powers, the sets of powers are presented in Table 8, 9 and 10.

Rationale: The users wants their hero to be more powerful than AI characters.

39. **HERO RESURRECTION**

When a hero has been defeated it shall be resurrected within 1 minute. When a hero is defeated it is no longer available in the game world, i.e. no interactions with the hero can be made until it is resurrected.

Rationale: The users wants other users to be crippled for a short duration of time when their heroes are killed in battle.

40. **USING A HERO POWER**

The user shall be able to use a hero power if the hero has the given amount of mana required by the hero power.

Rationale: The users wants to be able to use hero powers in order to reduce the amount of hitpoints of buildings, heroes and AI characters.

41. **MANA**

Each hero shall have a mana level. Each hero shall have a maximum amount of mana that the hero can hold.

Rationale: The users wants the hero to have a mana level in order to determine if a hero power can be used. The users don't want other heroes to have more mana than a given maximum at any given point.

42. **MANA REGENERATION**

Each hero shall receive an increase in mana level at a given time interval.

Rationale: The users want heroes to regain mana over time in order to use hero powers again once their mana level is below the mana requirement for that power.

4.2 Non-functional requirements

43. OPERATING SYSTEMS

The system shall run on Ubuntu 7.10, Windows XP SP2 and Mac OS X Leopard with a Java Runtime Environment installed.

Rationale: These are three major OS:es and they covers most users. Also, if it works on Ubuntu it probably will work on any Linux distribution.

44. JAVA

The system shall be written in Java and shall run on Java Runtime Environment version 1.5 and 1.6.

Rationale: Java has good portability and there are Java libraries for game development that supports the operating systems that the system shall support.

45. SYSTEM REQUIREMENTS

The system is required to run on standard PCs with moderate performance. The system shall run on a computer with at least a 1 GHz Pentium III CPU and 512 MB primary memory.

Rationale: The user wants to be able to use the system although he do not have access to a high-end computer.

4.3 Use cases

4.3.1 LOBBY SYSTEM: CREATE A GAME SESSION

Primary actor: The user

Scope: Lobby

Level: Summary

Stakeholders and interests:

- The user - To successfully create a new game.
- Additional players - To join the new game.

Precond: The user is in the lobby.

Minimal guarantees: None

Success guarantees: The user starts playing a game session.

Trigger: The user wants to create a game

Main success scenario:

1. The user creates a new game session.
2. The user sets the user limit for the game session.
3. The user sets the password for the game session.
4. The user starts the game session.
5. The game session starts when the user limit is reached.

Extensions:

3a **No password**

3a.1. The user choose not to set a password.

4.3.2 LOBBY SYSTEM: JOIN A GAME SESSION

Primary actor: The user

Scope: Lobby

Level: Summary

Stakeholders and interests:

- The user - To join a game session.

Precond: The user is in the lobby and there is atleast one game session.

Minimal guarantees: None

Success guarantees: The user starts playing in a game session.

Trigger: The user wants to join a game.

Main success scenario:

1. The user searches for an existing game session.
2. The system presents the user with available game sessions.
3. The user joins a game session.

4. The game session starts.

Extensions:

3a The game is password protected

3a.1. The system requests the password.

3a.2. The user enters the password.

3a2a The user enters the wrong password

3a2a.1. The user is returned to the lobby.

3a2a.2. The use case ends.

4a All slots is not occupied in the game session

4a.1. The game session starts when all slots are occupied.

4.3.3 DESTROY A CASTLE

Primary actor: The user

Scope: Game session

Level: Summary

Stakeholders and interests:

- The user
- The opponent whos castle is destroyed
- Other players

Precond: The user is playing in a game session.

Minimal guarantees: The user defeats the opponent.

Success guarantees: The user defeats the opponent.

Trigger: The user destroys an opponent's castle.

Main success scenario:

1. The opponent's territory is added to the user's territory.
2. The game continues.

Extensions:

1a The user is the last player with a castle

1a.1. The user wins the game and the statistics is updated.

1a.2. The use case ends.

4.3.4 MOVE THE HERO

Primary actor: The user

Scope: Game session

Level: Summary

Stakeholders and interests:

- The user

Precond: The user is involved in a game session and his hero is not dead.

Minimal guarantees: None

Success guarantees: The user moved the hero.

Trigger: The user wants to move the hero

Main success scenario:

1. The user requests the hero to move to a new location.
2. The system validates the new location.
3. The system moves the hero to the new location.

Extensions:

2a **The location is not valid**

- 2a.1. The system informs the user that the location is not valid.
- 2a.2. The use case ends.

4.3.5 USE A HERO'S POWER

Primary actor: The user

Scope: Game session

Level: Summary

Stakeholders and interests:

- The user

Precond: The user is involved in a game session and his hero is not dead.

Minimal guarantees: None

Success guarantees: The hero uses the power that the user selected.

Trigger: The user wants to use one of the hero's powers.

Main success scenario:

1. The user requests the hero to use a power.
2. The system validates that the hero can use the power at this moment.
3. The hero's mana is decreased by the amount corresponding to the selected power.
4. The power is applied.

Extensions:

2a **The hero's mana is not sufficient for the selected power**

- 2a.1. The use case ends.

4.3.6 CONSTRUCT A BUILDING

Primary actor: The user

Scope: Game session

Level: Summary

Stakeholders and interests:

- The user

Precond: The user is playing a game session.

Minimal guarantees: None

Success guarantees: The selected building is constructed at the specified location.

Trigger: The user wants to build a building.

Main success scenario:

1. The user selects a building type.
2. The system validates the building type.
3. The system requests a location for the building.
4. The user selects a location where the building should be built.
5. The system validates building location.
6. The building is constructed.

Extensions:

2a The user has not enough resources for the selected building type

- 2a.1. The system informs the user that the resources are not enough.
- 2a.2. The use case ends.

5a The location is not valid

- 5a.1. The system informs the user that the building location is not valid.
- 5a.2. The use case ends.

5 System architecture

The network architecture is a client-server type architecture, the server is thin and the client thick. A thin server, means that a minimal amount of game logic takes place at the server. A thick client, means that the client manages a majority of the game logic takes place at the client. Each client has an identical simulation of the game and all clients gets the same input from the network and each random generator has the same seed on all clients, therefore the AI will come up with the same decisions on all clients and no synchronisation is required. The physics engine will also come up with the same output on all clients due to the same reason.

Each client take input from the user and transforms the inputs into high level commands that are sent to the server. The server runs iterations where the number of iterations per second is the same as the frame rate of the game. During an iteration the server takes incoming commands and sends them to the clients. To mark the end of an iteration the server sends an “iteration end command” to all clients. The commands sent during an iteration should be executed in the same frame on all the clients. There is a one-to-one mapping between the index of the iteration that sent a command and the frame the command should be executed at. Some kind of heuristic on the client shall be utilised to decide when the next frame should be simulated and shown.

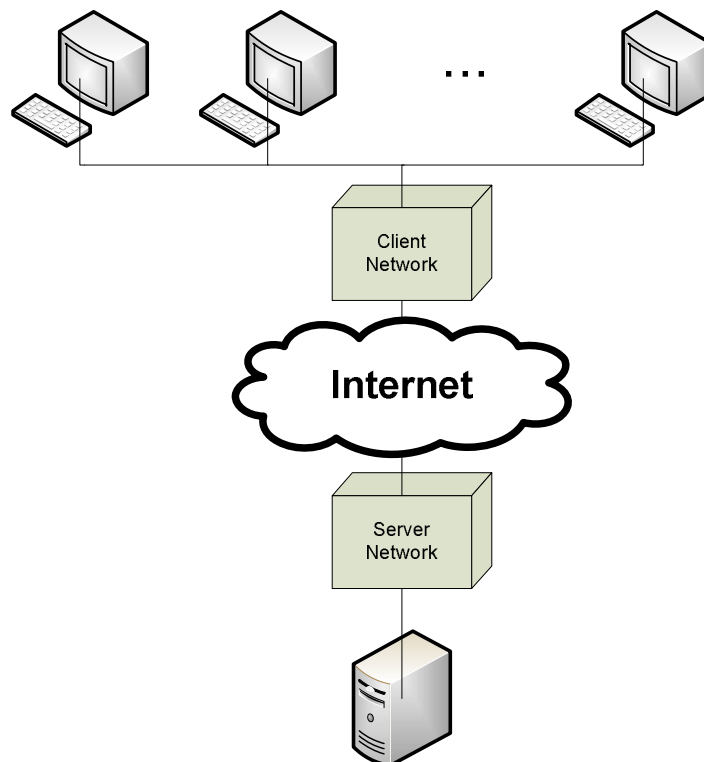


Figure 1: A figure representing the network architecture.

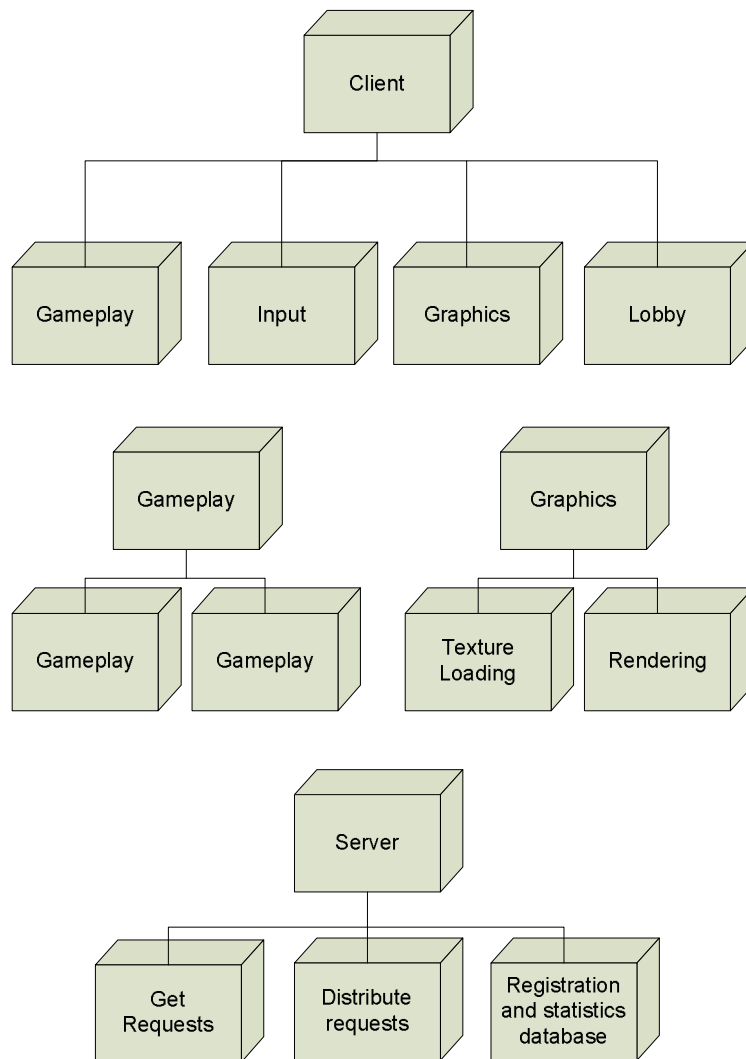


Figure 2: The figure displays how the different systems is composed of sub systems.

6 System requirements specifications

6.1 Functional requirements

6.1.1 General

1. FAILING NETWORK CONNECTION

Description: The users network connection fails and the system shall handle the failure.

Function: See Table 3

Requires: The user has at some time been connected to the server.

Client		
State	Error	Consequence
Lobby	The client's network connection is failing.	The system shall inform the user that there is no connection with the network. And that the user is no longer able to join a server until the network connection is restored.
In game	The client's network connection is failing.	The system shall return the user to the lobby, the system shall then inform the user that there is no connection with the network. And that he cannot join, create or search games until the network connection is restored.

Server		
State	Error	Consequence
In game	One of the clients connections is failing.	The action taken by the system shall be equivalent to that of leaving the game (See system requirement 8).

Table 3: Describes the system responses to a failing client connection.

2. SCREEN RESOLUTION

Description: Determines the resolution presented in a started game session.

Function: The resolutions available are those supported by the users monitor. All bitmap graphics shown in the game will be adapted to one resolution, and will therefore have to be stretched to match higher resolutions or compressed for lower resolutions.

Requires: The screen must support at least one resolution.

3. REGISTRATION

Description: A user shall be able to register an account in the game in order to let the game server keep statistics for the user.

Function: When a user registers the system shall create an entry for that user in the user database. The user must supply the fields that are listed in Table 4 and comply to the restrictions in order to register successfully.

Requires: A connection to the server.

Field	Restriction
Username	3-20 Alphanumeric characters.
Password	3-20 Alphanumeric characters.

Table 4: Table of registration restrictions

4. USER STATISTICS

Description: How the system shall handle user statistics.

Function: For each user the system shall have a database row containing wins and losses. The system shall update the database row when a user wins or loses a game.

Requires: That the user has registred an account.

6.1.2 Lobby

5. GAME SESSIONS

Description: The general functionality of a game session.

Function: A game session is on game instance set up for players to join, when it is started the players can interact with the game world. Every game session must have the information in Table 5 specified on the main server. A game session starts automatically when the number of *joined* players is equal to the *set* number of players.

Requires: None.

Data name	Description	Limits
Name	A name of the game session that can be used when searching.	3-30 alphanumeric characters or spaces.
Password	A passphrase needed to access the game session.	3-30 alphanumeric characters.
Player limit	The number of players needed to start the game.	Integer greater or equal to 2.

Table 5: Table of the information of a game session.

6. CREATE GAME SESSION

Description: How the system handles the creation of a game session.

Function: When a client requests a game session to be created and inputs a valid password and a maximum amount of players, the system shall create a new game session and assign the requesting player to the game session. A valid password contains only alpha and numerical characters and a valid maximum amount of players is greater or equal to two (See user requirement 2).

Requires: None.

7. JOIN GAME SESSION

Description: How the system shall handle a client joining a game session.

Function: When a client requests to join a game session the system shall assign the client to the requested session. If the game session is not able to assign the client to the requested game session due to the maximum player limit being reached, the system shall inform the user that the player limit has been reached and return the client to the lobby.

Requires: The client is in the lobby state and is connected to the game server.

8. LEAVE GAME SESSION

Description: How the system shall handle a client leaving a session.

Function: Leaving a game session shall result in losing (See system requirement 19) the current game session if the game session has started or freeing the player slot if the game session has not yet started and returning the leaving client to the lobby.

Requires: The client has joined a game session.

9. SEARCH GAME SESSION

Description: How the system handles client game session searches.

Function: Upon receiving a search query from a client, the system shall return all game sessions matching the query. A search query shall consist of the contents of Table 6, supplied in same order as they are mentioned in the table and separated with spaces.

Requires: The client is connected to the server and is in the lobby.

10. QUICK GAME

Description: How the system handles quick game requests.

Function: Upon receiving a user request for a quick game, the system shall check all game sessions for game sessions that has not been started and is one user short of reaching their maximum player limit and assign the user a game session. If there are multiple game sessions that is one user short from reaching their maximum player limit, the system shall choose the game session which players results in a score closest to zero according to the given formula

$$|r.wins/r.losses - \sum p.wins / \sum p.losses / \sum players|$$

Query option	Data supplied by client	Effect
User in session	Username or blank	If not blank, limits the results to only game sessions containing the given user.
Only non-full sessions	Yes or no	If yes, limits the results to game sessions that have not yet reached their maximum amount of players.
Only non-started sessions	Yes or no	If yes, limits the results to game sessions that are not in started state.

Table 6: Table of search query options

where r is the requesting user and p is a user involved in the game session currently being compared. If no game session one user short of reaching it's maximum player limit is found, the system shall make one check every two seconds, if no suitable game session is found after ten seconds, the system shall inform the user to try again later or try to join a game manually and return the user to the lobby.

Requires: The user is in the lobby state.

6.1.3 Gameplay


11. PLAY ORDER

Description: Defines boundaries of the gameplay.

Function: Each user shall only be able to attack one user. Upon defeating another user the defeating user will change the user that he is to attack to the user that the user he defeated was attacking (See figure 3 for a detailed example).

Requires: None.

A = Attacking
D = Defending

 = User

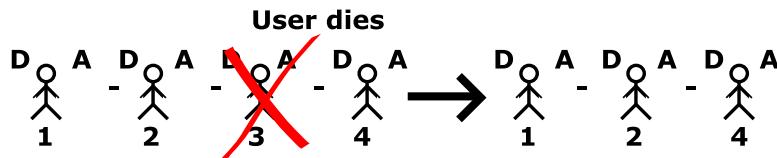


Figure 3: An example with 4 player showing what happens when user 3 dies.

12. DISTRIBUTE ATTACKING AND DEFENDING UNITS

Description: How the system handles the distribution of attacking and defending units.

Function: The system shall allow the user do change the distribution of attacking and defending units. The distribution shall be defined as follows $x:y$, where x means x attacking units per y defending units. Both x and y shall be positive values. Before changed, the ratio shall be set to 1:1.

Requires: None.

13. SPAWNING UNITS

Description: Describes how the system spawns new units.

Function: At a fixed time intervall the system shall spawn a set of units. The set of units shall consist of units from Table 7. For each player, the spawn set shall consist of units equivalent to the amount of production buildings (See user requirement 20) he controls. When spawning the units, the system shall distribute each type of unit according to the current distribution of attacking and defending units (See system requirement 12) for that player. The system shall spawn all units for all players simultaneously.

Requires: The player is involved is a game and his castle has not been destroyed.

Close-combat unit	Has a range which is zero.
Ranged-combat unit	Has a range which is range which is logarithmicly dependent on the accuracy attribute.

Table 7: Table of Unit types

14. HERO

Description: Describes the user's hero.

Function: Each user shall have a hero and each hero shall have one of the hero power sets as seen in Table 8, 9 and 10 assigned by the user, the system shall handle a hero in the same way that it handles units, only that the hero is contrrollable by the user that owns the hero.

Requires: None.

15. HERO RESURRECTION

Description: How the system shall handle the resurrection of a hero.

Function: When a hero is killed (See user requirement 35) that hero shall be returned to the owning users control with full hit points but only half of his maximum mana points after a time delay, starting when the hero is killed.

Requires: That the user has not lost the game session.

Rain of arrows	The hero fires a large amount arrows that rain down over the nearest hostile unit.
Power shot	The hero fires an arrow that deals more damage than an ordinary arrow on the nearest hostile unit.
Slice 'n dice	The hero deals a series of rapid blows to the nearest hostile unit.

Table 8: Table of Hero power set 1

Razor leaves	The hero sends a set of razor leaves that deals damage the nearest hostile unit.
Fire ball	The hero sends a fire ball that deals damage to the nearest hostile unit.
Tidal wave	The hero sends a tidal wave that deals damage to the the nearest hostile unit.

Table 9: Table of Hero power set 2

16. UNITS AND BUILDINGS ATTACKING UNITS, BUILDINGS OR HEROES

Description: How the system handles automated attacks, not controlled by the user.

Function: The system shall for each unit and building that has another hostile unit, building or hero within it's range and is not currently engaged in an attack, the system shall then change the state of the unit or building to attacking. The system shall then deal damage to the hostile unit, building or hero which is closest to the attacking unit, if a hostile unit is in range the system shall choose the closest unit over any building in range, if two or more hostile units are equally close, the system shall choose one of them arbitrarily.

Requires: None.

17. DEAL DAMAGE

Description: How the system deals damage to units, buildings and heroes.

Function: The system shall first check if the attack is succesful, an attack will deal damage if the following is true for the attacking unit $rand(0, 100) < accuracy$. If unsuccessful, the attack shall deal no damage. If successful, the attack shall result in deducting the following amount of hit points from the attacked unit $rand(attack, attack + 5) - defence$. If the damage dealt results in the amount of hitpoints for a unit, building or hero being zero or lower, actions should be taken according to system requirement 23.

Requires: None.

18. HOSTILE UNIT, BUILDING AND HERO

Description: What is a hostile unit, building or hero.

Power slide	The hero slides to the nearest hostile unit and deals damage to that unit.
Power strike	The hero deals a powerful strike that deals extra damage to the nearest damage.
Shock wave	The hero deals damage to all hostile units within a specified range.

Table 10: Table of Hero power set 3

Function: The system shall consider a unit, building or hero hostile towards another unit, building or hero if they belong to different players.

Requires: None.

19. LOOSING

Description: Conditions for loosing and actions to be taken when a loosing.

Function: A user shall loose a game when his castle is destroyed. Loosing a game shall result in updating the users statistics (See system requirement 4) adding one loss to his statistics. The loosing users territory shall be added to territory of the previous player. The loosing player shall then be moved to the lobby or be able to observe the rest of the game without interacting with the game.

Requires: None.

20. INGAME CHAT

Description: How the system handles the ingame chat.

Function: Upon receiving a chat message from a user in a game session, the system shall present the text message to all users in the same game session as the sending player.

Requires: None.

21. MOVE CAMERA

Description: How the system shall handle user requests for change of camera view.

Function: Upon the user requesting the camera to be moved in a horizontal direction, left or right, the system shall pan the users camera view in the given direction.

Requires: None.

22. MINIMAP

Description: The minimap shows a miniature view of the game world.

Function: The minimap shall show a simplified and miniature version of the entire gameworld. It shall also indicate which area the main view currently is viewing.

Requires: None.

23. DYING AND DESTRUCTION

Description: The actions taken when a unit, building or hero's hit points reaches zero.

Function: The system shall, for each unit, building or hero which hit points reaches zero or lower. Remove any representation of the unit, building or hero from the game and prevent it from interacting with the game world. If the actor removed was a hero, the steps in system requirement 15 shall be taken.

Requires: None.

24. CASTLE

Description: The castle allows a player to continue playing.

Function: At the beginning of a started game session each player shall have a castle. The castle's function is to allow users to keep playing. When a user's castle is destroyed the user instantly loses (See system requirement 19).

Requires: A started game session.

25. MOVE HERO

Description: How the system shall handle the movement of heroes.

Function: When a user requests it's hero to be moved horizontally, left or right, the system shall increment the hero's position in the given direction. If another unit, building or hero occupies the position that the user's hero was intended to move to, the hero should remain in it's last position.

Requires: The hero of the user who requests to move his hero is not dead.

26. CONSTRUCTING A BUILDING

Description: How a request for constructing a building shall be handled.

Function: When a user sends a request for a building to be built, if the requirements listed below are met, the system shall place a building at a vacant location in the requesting players territory and subtract the appropriate amount of resources from the requesting player. If the requirements are not met, the system shall notify the user which one of the requirements he has failed to meet, if multiple requirements are not met, the system shall notify the user of one of the failing requirements chosen arbitrarily.

- There is a buildable location suitable for the requested building type, controlled by the requesting player and that is not occupied by another building.
- The user has enough resources to construct the building.
- The building which shall be constructed is listed in Table 11.

Name	Type	Function
Barracks	Production	produces close-combat type units.
Archery range	Production	produces ranged-combat type units.
Defensive tower	Defensive	attacks hostile units within it's range.

Table 11: Table of buildings

Requires: None.

27. BUILDABLE LOCATIONS

Description: How buildable locations shall affect the users ability to build buildings.

Function: There shall be two types of buildable locations, production building locations and defensive building locations. At the start of a game session, each user shall have an equivalent amount of building locations of each type. Depending on the building location type, different buildings may occupy the buildable location. Production building locations may only contain production buildings and defensive building locations may only contain defensive buildings (See Table 11 for available buildings and their types). If a building is destroyed it shall no longer occupy the buildable location it previously occupied and the buildable location shall be freed so that the owning user can construct new buildings on it.

Requires: None.

28. ATTRIBUTES

Description: Description of attributes and how they shall influence game play.

Function: At the beginning of a game session, all attributes for all users involed in the game session, shall be set to the first level. An attribute shall affect all units a user controls. There shall be no upper limit on the level for any attribute. An attribute shall be upgradeable as described in system requirement 29. All attributes and their effects are listed in Table 12.

Requires: None.

Name	Description
Attack	a factor in the calculation of the amount of damage dealt by landing a succesful hit.
Defense	a factor in the calculation of the amount of damage taken by being hit.
Accuracy	determines the chance to succesfully hit an opponent.
Health	determines the amount of hit points.

Table 12: Table of attributes

29. UPGRADING ATTRIBUTES

Description: How the system shall handle upgrading of attributes.

Function: When a user requests an attribute to be upgraded, the system shall increment that attribute for the requesting player and deduct an amount of resources from the requesting players resources according to a formula where the cost is exponential to the level, if the amount to be deducted is greater than the current amount of resources of the requesting player, the system shall notify the user that he does not have enough resources.

Requires: None.

6.2 Use cases

6.2.1 STARTING THE GAME.

Primary actor: The user

Scope: General

Level: System

Stakeholders and interests:

- The client - Wants to connect to the main server.
- The user - Wants to connect to the main server.
- Main server - None.

Precond: The user has a working network connection.

Minimal guarantees: None

Success guarantees: The client connects successfully to the main server.

Trigger: The user requests a connection to the main server.

Main success scenario:

1. The client sends a connection request to the main server.
2. The main server adds the client to a connected client list.
3. The main server informs the client that it is connected.
4. The client informs the user that it is connected.

Extensions:

1a The main server does not respond

- 1a.1. The client informs the user that the it could not connect to the main server.
- 1a.2. Use case ends.

2a The maximum amount of clients is met

- 2a.1. The main server informs the client that main server is full.
- 2a.2. The client forwards the information to the user.
- 2a.3. Use case ends.

6.2.2 CREATE A GAME SESSION

Primary actor: The user

Scope: Lobby

Level: System

Stakeholders and interests:

- The user - To set up a game session that is accessible for other players.

Precond: The user has a connection to the server.

Minimal guarantees: None

Success guarantees: A game session is set up.

Trigger: The user requests that a game session is set up.

Main success scenario:

1. The system requests information about the game session (see Table 5).
2. The user provides the system with required information.
3. The system validates the provided information.
4. The system registers the new game in the list of all games.

Extensions:

3a **Invalid information from the user**

- 3a.1. The user is informed that some part of the provided information is invalid (see Table 13). The use case resumes at the start of step 2.

Field	Format
Game session name	3-20 alphanumeric characters or spaces.
Password	3-20 alphanumeric characters.
Number of players	An integer larger than or equal to 2.

Table 13: Table describing invalid game session information.

6.2.3 JOIN A GAME SESSION

Primary actor: The user

Scope: Lobby

Level: System

Stakeholders and interests:

- The user - To join a set up game session.

Precond: The user has a connection to the server, and there is at least one set up game session.

Minimal guarantees: None

Success guarantees: The user joins the game session.

Trigger: The user requests to join a game session.

Main success scenario:

1. The system requests information about the game session.
2. The user requests provides information about the game session.
3. The system validates that the user is able to join the game session.
4. The system updates the game session about the new player.
5. The sytem provides the user with information about the specified game session.

Extensions:

3a **Invalid information about the game session.**

3a.1. The user is informed that some part of the provided information is invalid (see Table 13).

3b **Invalid game session id.**

3b.1. Use case ends.

6.2.4 SEARCH FOR A GAME SESSION

Primary actor: The user

Scope: Lobby

Level: System

Stakeholders and interests:

- The user - Search games matching a text string.

Precond: The user has a connection to the server.

Minimal guarantees: None

Success guarantees: The user is provided a list of matching game sessions.

Trigger: The user wants to search a game session.

Main success scenario:

1. The system requests a search string.
2. The user provides a search string.
3. The system requests filtering options (see Table 6)
4. The user provides filtering options
5. The system matches the list of set up game sessions and adds matching (within boundaries of the filter) game sessions to the result list.
6. The system provides the user with the result list.

6.2.5 REGISTRATION

Primary actor: The user

Scope: General

Level: System

Stakeholders and interests:

- The user - Wants register his username on the server.

Precond: The user is connected to the server.

Minimal guarantees: None

Success guarantees: The user registers his username on the server.

Trigger: The user requests to register his username.

Main success scenario:

1. The system requests username and password.
2. The system validates the username and password (see table 4).
3. The system inserts a new row in the database with the user name and password.

Extensions:

2a Invalid username or password

- 2a.1. The system informs the user.
- 2a.2. Restart at step 1.

3a The system can communicate with the database

- 3a.1. The system informs the user.
- 3a.2. Use case ends.

6.2.6 CONSTRUCT A BUILDING

Primary actor: The user

Scope: Ingame

Level: System

Stakeholders and interests:

- The user - Wants to construct a building at selected location.

Precond: The user is connected to a started game session.

Minimal guarantees: None

Success guarantees: The user joins the game session.

Trigger: The user requests to construct a building.

Main success scenario:

1. The system validates that user's resources are sufficient to the corresponding building.
2. The system requests a location for the building.
3. The user provides the location details.
4. The system validates the location.
5. The system subtracts the buildings cost from the user's resources.
6. The system constructs the building at the specified location.

Extensions:

1a The resources are not sufficient

- 1a.1. The system informs the user that his resources are not sufficient.
- 1a.2. Use case ends.

4a The location is invalid

- 4a.1. Restart at step 2.

6.2.7 ATTACKING WITH THE HERO

Primary actor: The user

Scope: Ingame

Level: System

Stakeholders and interests:

- The user - Wants to attack a unit within range.

Precond: The user is connected to a started game session.

Minimal guarantees: None

Success guarantees: The specified unit is attacked and loses hit points accordingly.

Trigger: The user requests to attack a unit.

Main success scenario:

1. The system validates that the user's is within range of the unit.
2. The system checks that the hero hits unit.
3. The system calculates the amount of lost hit points (derived from the formula specified in system requirement 23) and subtracts them from the attacked unit.

Extensions:

1a **The hero is not within range**

1a.1. Use case ends.

2a **The user misses the unit**

2a.1. Use case ends.

6.2.8 USING A HERO POWER

Primary actor: The user

Scope: Ingame

Level: System

Stakeholders and interests:

- The user - Wants to use a hero power

Precond: The user is connected to a started game session.

Minimal guarantees: None

Success guarantees: The hero power is used and attacks the units within range.

Trigger: The user requests to use a hero power.

Main success scenario:

1. The system validates that the user has the hero power.

2. The system validates that the user has enough mana.
3. The system system subtracts the mana corresponding to the selected hero power.
4. The system finds the units within range of the hero and subtracts the number of hit points corresponding to the hero power from their health.

Extensions:

1a **The user does not have the power.**

1a.1. Use case ends.

2a **The hero does not have enough mana.**

2a.1. Use case ends.

4a **There are no units within range.**

4a.1. Use case ends.

7 System evolution

7.1 The fundamental assumptions of the system

The system assumes that Java 1.5 or a Java environment compatible with Java 1.5 will be available for the operating systems Mac OS X, Ubuntu and Windows XP.

7.2 Planned changes

- *A forum for players and developers to interact.* In order to maintain contact with the user base a forum could be developed. On the forum the developers can detect features that the players would like to add or features that the players dislike. It also gives the players an opportunity to report bugs and interact with each other to develop tactics and boast about their accomplishments.

7.3 Anticipated changes

- *Changes in user needs.* The users are likely to demand more advanced graphics. This, how ever, is not a problem since the expected life-time of a game is short. Instead of spending time on improving the system, resources will instead be spent on a new game, perhaps one which incorporates technologies from earlier games.
- *Future Java versions won't support Java 1.5.* The code is likely to require rework. In case of the external third-party libraries also failing to comply with future Java versions we risk reworking all interfaces towards the third-party libraries. However, such radical changes in the Java API should not take place within the systems expected life span.

8 Appendices

8.1 User database

The user database shall store at least the data in Table 14.

Name	Type	Restriction
Username	String	3-20 alphanumeric characters
Password	String	3-20 alphanumeric characters
Won games	Integer	Greater than or equal to zero
Lost games	Integer	Greater than or equal to zero

Table 14: Table of data in the user database

List of Tables

1	Table of the scope of the system	5
2	Table of risks and effects	7
3	Describes the system responses to a failing client connection. . .	22
4	Table of registration restrictions	23
5	Table of the information of a game session.	23
6	Table of search query options	25
7	Table of Unit types	26
8	Table of Hero power set 1	27
9	Table of Hero power set 2	27
10	Table of Hero power set 3	28
11	Table of buildings	30
12	Table of attributes	30
13	Table describing invalid game session information.	33
14	Table of data in the user database	39

List of Figures

1	A figure representing the network architecture.	20
2	The figure displays how the different systems is composed of sub systems.	21
3	An example with 4 player showing what happens when user 3 dies.	25

Index

- AI, 12
- attack, 13, 27, 36
- attribute, 30
 - upgrading, 31
- building, 35
 - construct, 19, 29
 - defensive buildings, 11
 - destroy, 14
 - hostile, 27
 - production buildings, 11
 - upgrading, 12
- camera
 - move, 13, 28
- castle, 11, 29
 - destroy, 17
- chat, 10, 11, 28
- damage, 13, 27
- destroy, 17, 29
- die, 13, 29
- distribution, 13, 26
- game session
 - create, 10, 16, 23
 - general, 23
 - join, 10, 16, 24, 33
 - leave, 12, 24
 - search, 10, 24
- health points, 13
- hero, 12
 - hostile, 27
 - movement, 14, 17
 - power, 14, 18
 - power set, 14
 - resurrection, 14, 26
- Java, 15
- Java Webstart, 9
- key bindings, 9
- lobby, 9
- losing, 12, 28
- mana, 14
 - regeneration, 14
- minimap, 12, 28
- multiplayer, 9
- network connection
 - failing network, 9, 22
- number of players, 10
- operating system, 15
- password, 10
- quick game, 10, 24
- ratio, 13
- registration, 10, 22, 34
- resolution, 9, 22
- search, 34
- spawning, 26
- starting, 32
- statistics, 11, 23
- system requirements, 15
- take damage, 13
- unit
 - hostile, 27
 - spawning, 26
- units, 12
- upgrading, 12
- winning, 12