Software Engineering

Software Requirements Specification

For

Ant-World Game

Version: Draft

Monday 5th March 2012

08

**Fall**

Group 11

Requirements Specification

Version Draft

6th March 2012

Approved by: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Robert Johnson Date

Approved by: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Mark Merriman Date

Approved by: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Oliver McCarthy Date

Approved by: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Andreas Nicolaou Date

Approved by: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Mark Purser Date

Approved by: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Joanne Robert Date

Approved by: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Eleanor Shakeshaft Date

Approved by: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ben Watt Date

**Revision Notice.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Revision | Date | Section | Description of Changes | Author |
| Version: Draft | 02/03/2012 | All | Set up template | Eleanor Shakeshaft |
|  | 03/03/2012 | Introduction | Wrote sections | Eleanor Shakeshaft |
|  | 03/03/2012 | Systems Requirements | Outlined requirements | Eleanor Shakeshaft |
|  | 05/03/2012 | Design & Implementation Constraints | Written. | Eleanor Shakeshaft |
|  | 07/03/2012 | Design & Implementation Constraints | Updated | Eleanor Shakeshaft |
|  | 07/03/2012 | User Classes and Characteristics | Written. | Eleanor Shakeshaft |
|  | 07/03/2012 | System Requirements | Use Case Diagram & finalised Reqs 1-7. | Eleanor Shakeshaft |
|  | 07/03/2012 |  |  |  |
|  | 07/03/2012 |  |  |  |
| Version: 1.0 |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
| Version: 2.0 (Final) |  |  |  |  |
|  |  |  |  |  |

**Document Revision History**

Table of Contents.

1. Introduction. 5

1.1 Purpose. 5

1.2 Document Overview. 5

1.3 Intended Audience. 5

1.4 Project Scope. 5

*1.4.1* *Out of Scope* 6

1.5 References. 6

2. Overall Description. 6

2.1 Product Flowcharts. 6

2.2 User Classes. 8

2.3 Operating Environment. 9

2.4 Design and Implementation Constraints. 9

2.5 User Documentation. 9

3. System Requirements 10

3.1 Req 1 – User uploads ant brain. 10

3.2 Req 2 – Software checks ant-brain syntax. 10

3.3 Req 3 – Matches 2 users to play 10

3.4 Req 4 – Create ant-world. 10

3.5 Req 5 – Software checks ant-world conditions. 11

3.6 Req 6 – Run simulation. 11

3.7 Req 7 – Ant characteristics. 11

3.8 Req 8 – Log game statistics. 11

3.9 Req 9 – Calculate winner. 11

3.10 Req 10 – Send copy of simulation to user servers. 11

3.11 Req 11 – User to have control of viewing simulation. 12

3.12 Req 12 – Tournament. 12

3.13 Req 13 – Produce Highest Scores for website. 12

3.14 Req 14 – Upload Highest Scores to website. 12

3.15 Req 15 – TBD. 12

3.16 Req 16 – TBD. 12

4. External Interface Requirements. 13

4.1 User Interfaces. 13

4.2 Hardware Interfaces. 13

4.3 Software & Communications Interfaces. 13

5. Other Non functional Requirements. 13

5.1 Performance Requirements 13

5.2 Security Requirements 13

5.3 Software Quality Attributes 13

6. Analysis Models 14

6.1 Case Modelling 14

6.2 Sequence Diagram 14

6.3 Class Diagram 14

6.4 State Diagram 14

1. Introduction.
   1. Purpose.

This Software Requirements Specification (SRS) indentifies the requirements for the Ant-World Game (AWG) for the Software Engineering Module.

* 1. Document Overview.

Section 2 is an overview of the descriptions of requirements for the AWG, and the requirements are categorised and defined in section 3. Sections 4 and 5 contain general information to aid in the understanding of this specification.

* 1. Intended Audience.

This document is primarily intended for the design developers, but also as a reference for the project managers, programmers, QA testers, and documentation writers. Read over the overview sections and proceed through the sections that are most pertinent to your requirements.

* 1. Project Scope.

Our customer is a games designer who has an idea for a new computer game, and wants us to provide a solid implementation of it. The game is a competitive two-player strategy game where the players (“users”) design and upload an ant-brain to pit against each other. The ants are placed into two ant colonies within a randomly-generated world simulation which also contains two anthills, some food sources, and several obstacles. The ant world will then simulate the behaviour of both kinds of ants; they must explore the world, find food and bring it back to their anthill. The winner is the anthill which has the most food at the end of 300,000 rounds. The highest scores and players will be put on a website along with statistics. At the basic level, all the users require is the outcome of the game and statistics, however we will implement an attractive graphical interface to make the game more engaging.

* + 1. *Out of Scope*

The software will not allow user profiles and it will not store ant-brain codes after the game(s) have completed; only the simulations, statistics and user aliases will be stored.

* 1. References.

Customer Requirements: <http://www.sussex.ac.uk/Users/mfb21/se/project/require.html>

Quality Manual: <http://www.sussex.ac.uk/Users/mfb21/se/project/quality.html>

Configuration Management: <http://www.sussex.ac.uk/Users/mfb21/se/project/config.html>

Project Website: <http://dl.dropbox.com/u/12957105/Website/home.html>

Project Plan: <http://dl.dropbox.com/u/12957105/Website/Project_Plan.pdf>

Google AI Challenge: <http://aichallenge.org/>

1. Overall Description.
   1. User Classes and Characteristics.

Players (“Users”) High frequency and primary (only) user, will have varing technical expertise but will not be required to have any technical skills apart from ability to code ant-brains. Target audience is expected to be but not limited to males aged between 16-40.

Users only involved in managing the AWG:

Client Manager Low frequency user, does not need any technical skills but will be expected to understand the licensing agreement. Has high privilege levels but will not be a user of the software or databases.

Client System Administrator High frequency user, will have technical expertise with servers and databases (but not necessarily gaming), will be able to install and administer the software using only the Installation Guide. They will need to understand basic web administration to update the game website if needed, however this can be outsourced by the company.

* 1. Operating Environment.

The software will be designed to operate on PCs only and be will be cross-platform. Users will not rbe required to hold any software on their PCs as the game will be played within a browser, Chrome will be the recommended browser to use. The highest scores will be displayed on a website.

The Client will need a database & server.

* 1. Design and Implementation Constraints.

Deadlines:

Acceptaince Criteria:

Deliverables Review (Draft): Monday 5th March

Deliverables Review (Final): Friday 9th March

Final Version: Monday 12th March

High-level Design:

Deliverables Review (Draft): Monday 5th March

Deliverables Review (Final): Friday 9th March

Final Version: Monday 12th March

Detailed Design Document:

Deliverables Review (Draft): Monday 1st th March

Deliverables Review (Final): Friday 27th April

Final Version: Tuesday 1st May

Deliver Software & Documentation to client:

Deliverables Review (Draft): Monday 13th May

Deliverables Review (Final): Friday 1st June

Final Version: Monday 11th June

Presentation: Wednesday 13th June

Programming language:

Java.

Website:

HTML, CSS & PHP – designed primarily for Chrome.

Graphical User Interface:

2D only required, must be designed with the target audience in mind.

User must be able to adjust the speed of the simulation, as well as pause it.

Statistics must be displayed alongside simulation.

All information will be in English (UK) only.

Version Control: GitHub

To be used on every update, and log all changes.

Programming Standards:

Client’s organisation will be responsible for maintaining the delivered software and helping user’s with issues.

* 1. User Documentation.

The client will be given a document, along with the software, which will contain two sections:

Installation guide: This is written for the system administrator (who can be assumed to have reasonable computing knowledge, but knows nothing about gaming in general this software in particular) installing the program. It must be possible to install the software and get it to run from the installation guide only.

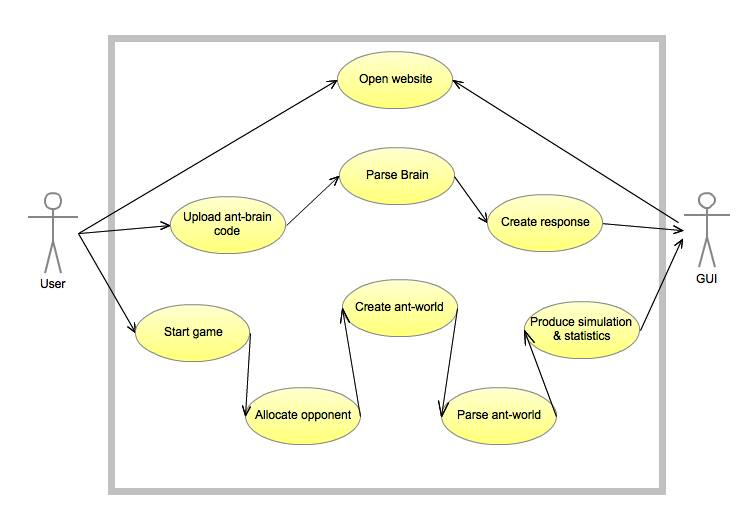
User Manual:  This documentation is to be read by users of the program (who cannot be assumed to have any particular computing knowledge), and shall describe the user interface and functionality of the system, in terms that the user understands.

It will also detail the licensing agreement for the software.

The client will also be given a website which will hold the highest scores and statistics.

1. System Requirements

A high level use case diagram of the functions of the AWG with regards to the Users’ interactions with the GUI provided in *Figure 1* consisting of: 1) User opens the website where the GUI is displayed, 2) User uploads the ant-brain code – this is parsed and the results fed to the GUI, 3) If the ant-brain code is verfied then the User can start a game – the software matches them with another player, creates an ant-world, parses the ant-world, runs the simulation then sends the data to the GUI to be displayed to the User.

****

*Figure 1 – High Level Use Case Diagram for the Ant-World Game, User & GUI.*

* 1. Open website.
  2. Authenticate alias and ant-brain code file-type.

***Description and Priority.***

User(Player) fills in an alias for high-score table and uploads their ant brain code.

Priority = high.

***Stimulus/Response Sequences***

Stimulus: User fills in alias name and selects file to upload

Response: System validates if alias conforms, and file is correct type. If correct allows user to submit code.

Stimulus: User submits code.

Response: Systems responds that code is being verified.

Step 1: User opens software / website?

Step 2: User starts new session & fills in alias.

Step 2: User loads ant-brain code document and submits.

***Functional Requirements***

Authenticate.Alias The system shall ensure maximum length of 8 characters of only letters or numbers.

Authenticate.FileType The system must only accept plain text files, and will respond with error message it the file is not.

* 1. Parse ant-brain code.

***Description***

Once ant-brain code is uploaded the software must check if it is syntactically correct.

Priority = high.

***Stimulus/Response Sequences***

Stimulus: User submits code.

Response: System parses the code to check if it is syntactically correct.

Stimulus: Code is verfied.

Response: Save code and send confirmation to User(Player) via GUI.

Stimulus: Code is not verfied.

Response: Delete code and send confirmation of error to User(Player) via GUI.

***Functional Requirements***

ParseAntBrain.

ParseAntBrain.

* 1. Start Game

Add User to queue.

* 1. Allocate opponent

***Description***

The software must allocate 2 verified users to battle.

Priority = high.

***Stimulus/Response Sequences***

Stimulus: More than one user has opted to start a game and is waiting in a queue..

Response: System pops off User(Player) from top of queue to play as Red, then pops off next User(Player) to play as black.

***Functional Requirements***

AllocateOppenent. Check queue has more than 2 Users(Players)

AllocateOppenent.

* 1. Create ant-world.

***Description***

The software must generate a random ant-world for the game to take place.

Priority = high.

***Stimulus/Response Sequences***

Stimulus: 2 Users(Players) have been allocated to play and given colours

Response: Randomly create ant-world and populate with ant-colonies.10

***Functional Requirements***

CreateWorld.Dimension 150 × 150 cells

CreateWorld.Perimeter The cells on the perimeter are always rocky.

CreateWorld.Anthills 2 anthills (hexagonal of side length 7)

CreateWorld.Rocks 14 rocks

CreateWorld.Food 11 blobs of food (5-by-5 rectangle, with each cell containing 5 food particles)

CreateWorld.Positions Chose randomly.

There must always be at least one empty cell between non-food elements.

No elements must overlap.

CreateWorld.AntPositions All ants are initially facing in direction 0.

* 1. Parse ant-world.

***Description***

Once ant-world is created the software must check if it is correct as per the specifications.

Priority = high.

***Stimulus/Response Sequences***

Stimulus: An ant-world has been created and populated.

Response: System parses the world layout to check if it is correct as per the specifications.

Stimulus: Ant-world is verified.

Response: Send information to GUI to display to Users(Players) that a battle is commensing.

Stimulus: Ant-world is not verfied.

Response: Repeat 3.6.

***Functional Requirements***

ParseAntWorld.

* 1. Run simulation.

***Description***

The game will commense and the entire simualtion will run through.

***Functional Requirements***

Step 1: World has been verified.

Step 2: Run simulation until end.

Each ant will make 300,000 moves then the simulation will end.

* 1. Req 7 – Ant characteristics.

***Description***

How the ant’s should behave to other ants and surroundings.

***Functional Requirements***

Can communicate or leave trails by means of chemical markers in order to find their way back to the anthill and store their food.

Can sense but not modify, the markers of the other species.

Different ant species can attack each other by surrounding them; If an ant ever finds itself adjacent to 5 (or 6) ants of the other species, it dies

When the ant dies as a result of an attack it becomes 3 food particles.

* 1. Req 8 – Log game statistics.

***Description***

<<<

***Functional Requirements***

<<<

* 1. Req 9 – Calculate winner.

***Description***

<<<

***Functional Requirements***

Only the number of food particles currently in the anthill cells of each colour are counted, food being carried by an ant does not count, even if it is standing on its own anthill

* 1. Req 10 – Send copy of simulation to user servers.

***Description***

<<<

***Functional Requirements***

<<<

* 1. Req 11 – User to have control of viewing simulation.

***Description***

<<<

***Functional Requirements***

<<<

* 1. Req 12 – Tournament.

***Description***

<<<

***Functional Requirements***

Be able to play tournaments, where an arbitrary number of players can upload ant-brains, who are all paired up to play against each other. They will fight twice on each contest world, each being red and black once. The winner gains 2 point, if it is a draw both players get 1 point. The overall tournament winner is the ant brain that wins the most individual games or has the biggest amount of food in its anthill.

* 1. Req 13 – Produce Highest Scores for website.

***Description***

<<<

***Functional Requirements***

<<<

* 1. Req 14 – Upload Highest Scores to website.

***Description***

<<<

***Functional Requirements***

<<<

1. External Interface Requirements & Other Non-functional Requirements
   1. User Interfaces.

<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>

* 1. Hardware Interfaces.

<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>

* 1. Software & Communications Interfaces.

The software must communicate with the database which will store the statistics and scores, the database will in turn communicate with the website so that the high scores can be updated. The software must also communicate with the user servers so that the simulations can be sent.

* 1. Performance Requirements

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>

* 1. Security Requirements

Users will not have log-ins so no data is stored or kept about users. Only an alias will be input and stored for the high scores & statistics.

1. Analysis Models
   1. CRC Cards
   2. Object & Class Diagram
   3. Sequence Diagram.
   4. State Diagram

Analysis Model.

This section describes the desired behaviour of the system.

The main abstractions required by the problem domain, and the classes and objects which will be used to represent these abstractions, should be identified and described here, together with their roles, responsibilities and collaborations.

The analysis model is given as a collection of scenarios illustrating each function point of the system or subsystem under consideration.

Scenarios may be described using CRC cards, object and class diagrams in UML notation, and/or finite state machines as appropriate. Scenarios should be divided into primary - those that pertain to the key behaviour of the system - and secondary - behaviour pertaining to exceptional circumstances.