Acceptance Criteria
For
Ant-World Game

Version: Final
Monday 12th March 2012

Group 11

Acceptance Criteria

Version 2.0

12th 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.

Document Revision History

Revision	Date	Section	Description of Changes	Author
Version: Draft	02/03/2012	All	Initial version	Mark Purser, Ben Watt
Version: 1.0	12/03/2012	Usability tests, all	Added usability tests, fixed things across the document	Ben Watt
Version: 2.0	12/03/2012	All	Final submission	Mark Purser, Ben Watt

Contents

1. Introduction. 5
2. Test Environment. 5
3. Acceptance Criteria. 6
   3.1 Functionality Tests 6
   3.2 Usability Tests 8
   3.3 Efficiency Tests 11
   3.4 Reliability Tests 11

1. Introduction.

Testing will undergo two phases while the program is being developed – white box testing and black box testing. The first phase, white box testing, will test the innards of the program, checking each method and class for any inconsistencies. The second phase of testing will be black box testing, after the program is mostly completed. Here, the testing will focus on the inputs and outputs of the program. This will allow us to check if it conforms to the requirements that have been specified, and if it does not, give us an opportunity to amend the code.
In general, the testing methodology used for the project will not include test driven development.
The acceptance criteria presented in this document comprise a suite of black-box tests. For acceptance of the software, the customer must run through these tests and affirm that they match the expected results. The tests are designed to maximise the coverage of all requirements and testable components of the system.

2. Test Environment.

The testing environment will be a standard desktop or laptop computer, meeting the following specification:

- Windows 7 operating system*
- 64-bit processor dual-core or equivalent
- 2Gb memory

and with the following software installed

- Java runtime environment
- JUnit testing framework
- JMeter or Grinder for network load testing
- Quickcheck for Java
- Git
- Google Chrome or Mozilla Firefox browser
- Custom test scripts and programs

* note that the OS is a restriction of the test environment only; the game will run on other platforms that support the JRE

Load conditions

Unless stated otherwise, the tests require that the operating system is running with the following conditions:

- no unnecessary desktop applications open that use excessive CPU time or memory
- no Windows updates or virus scans running in the background

Data files used for testing

A third-party high level scripting language is used to create scripts that automate the tests. The automated tests are run sequentially at regular intervals to alert the team of any points of failure.
Other data files are:

- A variety of finished and syntactically correct ant brains
- A variety of finished and syntactically correct ant worlds

3. Acceptance Criteria.

3.1 Functionality Tests

3.1.1 Trace correlation test

Analysis model being tested
Section 8.3 SCI-2

Prerequisites
Ant server
Ant client
Customer files tiny.world, sample.ant, uncompressed 'dump.all' trace
A third-party text diff program

Test
1. Start the ant server and client
2. Upload the tiny.world world file and the sample.ant ant file in to the simulation
3. Set the randomisation seed to s0 = 12345
4. Run the simulation
5. Using a diff program, compare the output trace to the customer's 'dump.all' trace

Expected result
The simulation output trace is identical to that output by the customer simulation

3.1.2 Client visualisation test

Analysis model being tested
Section 3.1 Website.Compatibiliy Website.Updates

Prerequisites
Client visualisation test script
Test network injector program
Latest Internet browser- Chrome or Mozilla
Ant client

Test
1. Run the client visualisation test script
2. The script automatically starts the test network injector program and client
3. Observe the pre-set diagnostic that is displayed on the client

Expected result
The client displays the diagnostic correctly and reports no errors

3.1.3 Ant-brain syntactic checker program

Analysis model being tested
Section 3.5 ParseAntBrain.CheckSyntax

Prerequisites
Ant-brain syntax checker test script
Ant-brain syntax checker program
Valid and invalid ant-brain files

Test
1. Run the ant-brain syntax checker test script
2. The script automatically starts the ant-brain syntax checker program
3. View the output results

Expected result
A success is reported on the syntactically correct files and an error is reported on the others

3.1.4 Ant world syntactic checker program

Analysis model being tested
Section 3.6 ParseAntWorld.CheckLayout

Prerequisites
Ant world syntax checker test script
Ant world syntax checker program
Valid and invalid ant world files

Test
1. Run the ant world syntax checker test script
2. The script automatically starts the ant world syntax checker program
3. View the output results

Expected result
A success is reported on the syntactically correct files and an error is reported on the others

3.1.5 Upload of high scores to web site

Analysis model being tested
Section 3.10 GameStatistics.WebScores

Prerequisites
High score test script
Ant server
Ant client

Test
1. Run the high score test script
2. The script automatically starts the server and client
3. The client starts a game
4. After the simulation is over, the client's score is uploaded to the server
5. Observe the score on the website

Expected result
The score is successfully uploaded to the website.

3.1.6 Allocate Opponent

Analysis model being tested
Section 3.4 AllocateOpponent.Allocate AllocateOpponent.Check

Prerequisites
Opponent allocation test script
Ant server
Ant clients
Ant brain files

Test
1. Start the opponent allocation test script
2. This starts the server and two clients
3. Both clients upload an ant brain and elect to start a game
4. Observe that the players are matched with each other

Expected result
The server reports that the two players have been matched.

3.2 Usability Tests

3.2.1 Enter user alias

Analysis model being tested
Section 3.2 SingleGame.FirstPlayer

Prerequisites
Ant server
Ant client
Ant brain and world files file

Test
1. Start the ant server and client
2. Enter username in client
3. Upload ant brain to the simulation
4. Run the simulation
5. Observe username of the player in game
6. Observer the username of the player in high scores

Expected result
The username of the player appears correctly in the simulation and on the scoreboard.

3.2.2 Upload ant-brain

Analysis model being tested
Section 3.4 Authenticate.FileType

Prerequisites
Ant server
Ant client
Ant brain file

Test
1. Start ant-brain uploading script
2. This starts the client and server, and automatically tries to upload the ant-brain

Expected result
The ant brain is uploaded onto the server.

3.2.3 Upload ant world

Analysis model being tested
Section 3.6 Single game Ant-world

Prerequisites
Ant server
Ant client
Ant world file

Test
1. Start ant-world uploading script
2. This starts the client and server, and automatically tries to upload the ant-world

Expected result
The and world is uploaded to the server.

3.2.4 Start simulation

Analysis model being tested
Section 3.9 Run Simulation

Prerequisites
Ant server
Ant client
Ant world and brain files

Test
1. Start simulation testing script
2. This starts the server and two clients
3. Both clients set username and upload an ant brain
4. The server then starts the simulation

Expected result
The simulation is started without problems.

3.2.5 Playback

Analysis model being tested
Section 3.11 ViewSimulation.View

Prerequisites
Ant server
Ant client
Ant world and brain files

Test
1. Start simulation testing script
2. This starts the server and a client
3. The client sets a username and uploads and ant brain
4. The server then starts the simulation
5. From the GUI menu, select debug statistics to be on
6. The client is observed during playback
7. Debug statistics are observed to check for irregularities

Expected result
The client displays the results of the simulation correctly. Graphics display ants and the world correctly.

3.2.6 Fast-forward and rewind

Analysis model being tested
Section 3.11 ViewSimulation.Navigation

Prerequisites
Ant server
Ant client
Ant world and brain files

Test
1. Start simulation testing script
2. This starts the server and two clients
3. Both clients set username and upload an ant brain
4. The server then starts the simulation
5. From the GUI menu, select debug statistics to be on
6. In the client, fast-forwarding and rewinding the simulation are tried
7. The performance of the client and and current state are observed

Expected result
Fast-forwarding and rewinding do not cause any performance issues, and the state shown being displayed in the client changes at the appropriate rate for the speed.

3.2.7 Start tournament

Analysis model being tested
Section 3.7 Tournament Ant-world

Prerequisites
Ant server
Ant client
Ant world and brain files

Test
1. Start simulation testing script
2. This starts the server and *n+1* clients, where *n* is the size of the tournament
3. The first user sets their name and uploads an ant brain, and *n* is chosen as the tournament size
4. They elect to enter a tournament
5. The rest of the clients do the same
6. The clients after the first will be observed joining the tournament
7. The tournament will then start
8. Player *n+1* will try to join the tournament

Expected Result
The first *n* users join the tournament successfully. The tournament starts using each of their ant brains and user names. Player *n+1* is unable to join the tournament, and instead joins the queue for starting a new one.

3.3 Efficiency Tests

3.3.1 Client screen refresh test

Analysis model being tested
Section 8.4 PE-4

Prerequisites
Ant server
Ant client
Response test world and ant-brain files

Test
7. Start the ant server and client
8. Upload the response test world and ant-brain files in to the simulation
9. From the GUI menu, select debug statistics to be on
10. Run the simulation
11. Observe the frame update rate on the debug stat display

Expected result
The frame update rate doesn't drop below 10 frames/sec throughout the simulation.

3.4 Reliability Tests

3.4.1 Simulation reliability test

Analysis model being tested
Section 8.4 PE-1 PE-2 PE-3

Prerequisites
Reliability test script

Test
1. Run the reliability test script
    a. *The script automatically starts the ant server*
    b. *Two client bots open and connect to the server and automatically start a simulation run*
2. *The client bots start another run of the simulation* when the previous one has finished, and so on ad infinitum.*
3. Leave the client bots running for a long period of time

* The parameters of the simulation change on each run. A randomisation seed is selected which is used to generate a world, an ant-brain and to seed the simulation. The randomisation seed is output to a trace file so that the conditions can be recreated if the system crashes.

Expected result
The system runs for more than 24 hours without crashing

3.4.2 Server stress test

Analysis model being tested
Section 8.4 PE-1 PE-2 PE-3

Prerequisites
Ant server
Server stress test script
Server test world and ant files

Test
1. Run the server stress test script
    a. *The script automatically starts the ant server*
    b. *12 client windows now open, connect to the server, automatically upload ant brains and start a tournament simulation*
2. Observe that the tournament runs correctly and that review of the games are available from all clients

Expected result
The visualisation can be viewed with a reasonable frame update rate and response time on all clients

3.4.3 Maximum connected clients (Server)

Analysis model being tested
Section 8.4 PE-1 PE-2 PE-3
This test is for determining the highest feasible number of connected clients to a single server. This test will be performed in tandem with second runs of the maximum ant load and maximum simulation speed tests.

Prerequisites
This requires a range of proper map files and a range of functional ant brains.
Ant server
Server stress test script
Server test world and ant files

Test
1. Run the server stress test script
    a. *The script automatically starts the ant server*
    b. *12 client windows now open, connect to the server, automatically upload ant brains and start a tournament simulation*
2. Observe that the tournament runs correctly and that review of the games are available from all clients

Expected result
The visualisation can be viewed with a reasonable frame update rate and response time on all clients

3.4.4 Maximum theoretical ant load (Server)

Analysis model being tested
Section 8.4 PE-1 PE-2 PE-3
This tests the simulation to see how it performs when the maximum possible number of ants are simulated. This will be done to confirm whether or not the performance of the simulation is up to the standards specified.
To perform this test, multiple simulations will be run. Each time, the number of ants will be increased. This will be done until the performance of the program falls below some defined threshold.

Prerequisites
A range of proper map files and a range of functional ant brains.
Ant server
Server stress test script
Server test world and ant files

Expected result
The visualisation can be viewed with a reasonable frame update rate and response time on all clients

3.4.5 Maximum theoretical ant load (Client)

Analysis model being tested
Section 8.4 PE-1 PE-2 PE-3
This tests the client to see how well it performs when the maximum possible number of ants are displayed.

To perform this test, multiple simulations will be run. Each time, the number of ants will be increased, increasing the number of ants that need to be displayed in the client. This will be done until performance falls below some defined threshold.

Prerequisites
A range of proper map files and functional ant brains, and a fully working ant world simulator.
Ant server
Server stress test script
Server test world and ant files

Expected result
The visualisation can be viewed with a reasonable frame update rate and response time on all clients

3.4.6 Maximum simulation speed(Server)

Analysis model being tested
Section 8.4 PE-1 PE-2 PE-3
This will test how the performance of the server holds up when simulation is set to be done at it's maximum speed.

Prerequisites
A range of proper map files and functional ant brains, and a fully working ant world simulator.
Ant server
Server stress test script
Server test world and ant files

Expected result
The visualisation can be viewed with a reasonable frame update rate and response time on all clients

3.4.7 Visual effects load test (Client)

Analysis model being tested
Section 8.4 PE-1 PE-2 PE-3
This test will measure the performance of the program as the graphics options (if they exist) are set to the maximum level. This load test will be performed in tandem with a second tests of fast-forwarding and ant load.

Prerequisites
This requires a range of proper map files and functional ant brains, and a fully working ant world simulator.
Ant server
Server stress test script
Server test world and ant files

Expected result
The visualisation can be viewed with a reasonable frame update rate and response time on all clients