|  |
| --- |
|  |
| Project Deliverable 4 |
|  |
| Verification and Validation |

**NWTTT**

November 26, 2016

**Team Members:** Vargha Hokmran, Nick Hanley, Christopher Feener

Project Deliverable 4

Verification and Validation

# Test Plan

# Static V&V

## Tested Items

## Checklist

## Inspection Results

## Rework and Follow-up

# Dynamic V&V

## Unit Testing

## Interface Testing

## System Testing

## Non-functional Requirements Testing

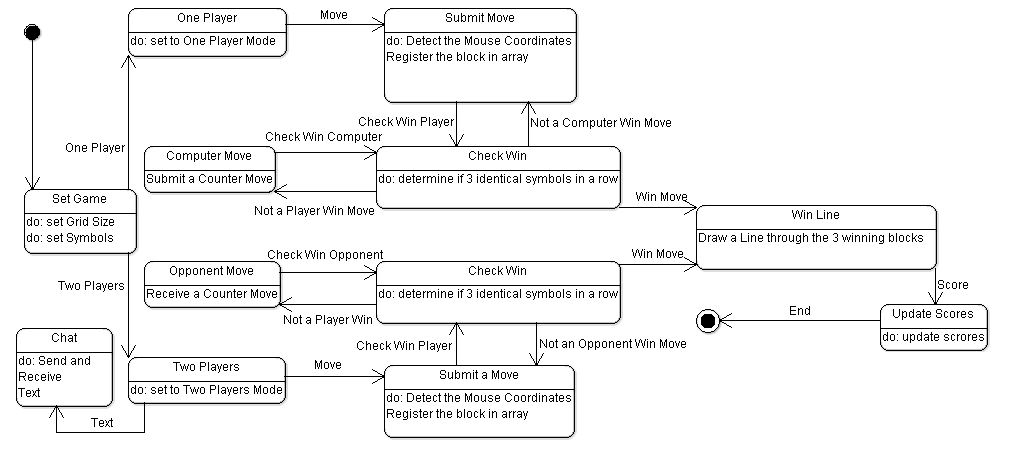
# Test Plan

**The testing process:**

Testing process for this software includes two sub-tests: testing the game against the computer, and testing the game against another player, over the network. The major tests for this software would be:

* Single-player mode test
* Two- player mode test

For each of these major tests, it is necessary to test the game with all right-side setting, and follow the game to the end, until one side wins the game or the game end up to be tie.



**Requirements Traceability:**

Testing game against computer:

* Open the game
* Choose opponent to be “CPU”
* Pick symbols for player and computer, from the right settings
* Pick who starts first, from the right settings
* Start the game by hitting the “Let’s Start a New Game” button.
* Change the difficulty level, and test a completer game in each level
  + Left (predictable): Computer simply chooses blocks from top left to the right
  + Middle (random moves): Computer chooses random blocks
  + Right (smart): Computer blocks players vertical and horizontal moves
* As soon as player/Computer puts three symbols vertically, horizontally, or diagonally next to each other the game shall draw a line and show the winner blocks
  + If all blocks get taken, and none of the sides win, or after one side’s win, the game shall stop, and renew the results on the right side
  + If everything goes as expectations of these steps, the single-player mode is fine.

Testing game against another player:

* Open the game, on two computers over a network.
  + Make sure there is no firewall on the network or other computers
* Choose opponent to be “Human”
  + A new window shall open for network settings
  + Choose one of the computers to be “Server”. The local IP address shall be shown on the text field.
  + On the other computer, select “Client”, and insert Server’s IP address, in the text field.
  + Clicking “Cancel” button shall remove all two player functionalities from the main window of the game.
  + Clicking on “OK”, shall set the game for two player mode:
    - Game window shall remove:
      * Difficulty Level, from the bottom
    - Instead, the game window shall add:
      * Chat Box
      * Send Button
      * Chat Log
* By typing in the white chat box and pressing “Enter” key from the keyboard, or by clicking on “Enter” button from the game window, the message shall appear on both player’s chat log text area, on top of the chat box
* Pick symbols for yourself and opponent, from the right settings. Note: Each player can set his/her favorite symbols for his/her own game, not the other player’s game
* Pick who starts first, from the right settings
  + Changing this setting shall affect the other players screen
* Start the game by hitting the “Let’s Start a New Game” button.
* Change the difficulty level, and test a completer game in each level
  + Left (predictable): Computer simply chooses blocks from top left to the right
  + Middle (random moves): Computer chooses random blocks
  + Right (smart): Computer blocks players vertical and horizontal moves
* As soon as player/Computer puts three symbols vertically, horizontally, or diagonally next to each other the game shall draw a line and show the winner blocks
  + If all blocks get taken, and none of the sides win, or after one side’s win, the game shall stop, and renew the results on the right side
  + If everything goes as expectations of these steps, the two-player mode is fine.

# Static V&V

## Tested Items

The products of the software process that are to be tested

* Different symbols on the game panel
* If the game work on different difficulty levels, if playing against the computer
* Detecting the winner, if any of players win the game
* Drawing win line on winner symbols
* Detecting tied games if all blocks are taken, nobody wins the game
* Playing as the first and second player
* Sending and receiving messages in two-player mode
* Sending and receiving moves in both sides in two-player mode
* Updating chat-log in both sides in two-player mode

**Testing schedule:**

**Overview meeting**

The software has been programmed modular, and after programming each module by one of the programmers, the other programmer always tested the module, and the team finalized the improvements.

There is also an overview meeting scheduled with another group, to test the software with individual programmers. This meeting is scheduled to be on Monday 11/28/2016.

**Inspection meeting**

Inspectors agreed to return results by Tuesday 11/29/2016, so we would be able to work on the last debugs and improvements of the software.

## Checklist

**Test recording procedures:**

The items need to be specifically tested by inspectors are listed in the table below. Inspectors shall fill the table with final results of tests, and also add their comments and/or recommendations for possible debugs or improvements.

|  |  |  |  |
| --- | --- | --- | --- |
| Test | Passed | Failed | Comment / Recommendation |
| Game panel graphics |  |  |  |
| Do the player and opponent symbols, grids, and line lines function correctly, in different screen sizes? | | | |
| First Move |  |  |  |
| Does the game function fine in network and single-player mode with either side starting first? | | | |
| Tied games |  |  |  |
| The game is supposed to stop and renew the results after all blocks taken and no win | | | |
| Game results |  |  |  |
| Does the game result get renewed in both game modes for win-ended and tied games? | | | |
| Difficulty Levels |  |  |  |
| Does all three difficulty levels act as described in the requirements traceability section? | | | |
| Settings Window |  |  |  |
| Does the settings apply changes correctly, and does the game work fine in different grid sizes? | | | |
| Change Background |  |  |  |
| Does all backgrounds fit, and stretch fine? | | | |
| Server IP detection |  |  |  |
| IP detection section should show the local IP address. Does it function fine? | | | |
| Network connection |  |  |  |
| * Do the computers connect fine over the network? * After closing the game the port shall be closed, and be ready for other connections. To test this, close the game and open it again, and try using the port again by one more game connection. | | | |
| Chat log |  |  |  |
| Does the chat log area show your sent and received text messages fine? | | | |

**Hardware and software requirements:**

This game at minimum requires:

* Software
  + Windows 7 or higher
  + Mac OS X or higher
  + Java
* Hardware
  + Normal home computer (Laptop/Desktop)
    - To test the network functionality, it is recommended to test client, and Server on two different computers
  + Intel Pentium IV, AMD A6 or higher processor
  + Network

**Constraints**

The network needs to has no firewall, or to has open access on port 1201, to test the network functionality

## Inspection Results

Inspectors tested the program and filled the provided test table with the results as below:

|  |  |  |  |
| --- | --- | --- | --- |
| Test | Passed | Failed | Comment / Recommendation |
| Game panel graphics | **** |  | Functional in default and resided screen sizes |
| Do the player and opponent symbols, grids, and line lines function correctly, in different screen sizes? | | | | |
| First Move | **** |  | Functional in network and single mode |
| Does the game function fine in network and single-player mode with either side starting first? | | | | |
| Tied games |  | **** | If the winner move be the last one, instead of adding to win, result for tie gets added by one |
| The game is supposed to stop and renew the results after all blocks taken and no win | | | | |
| Game results |  | **** | Does not renew if win on the last move |
| Does the game result get renewed in both game modes for win-ended and tied games? | | | | |
| Difficulty Levels | **** |  | Functional in all three levels |
| Does all three difficulty levels act as described in the requirements traceability section? | | | | |
| Settings Window |  | **** | Not implemented yet |
| Does the settings apply changes correctly, and does the game work fine in different grid sizes? | | | | |
| Change Background | **** |  | Functional |
| Does all backgrounds fit, and stretch fine? | | | | |
| Server IP detection | **** |  | Detects local ip correctly |
| IP detection section should show the local IP address. Does it function fine? | | | | |
| Network connection | **** |  | - Connected over the network, and game plays fine  - Connection Port is safe |
| * Do the computers connect fine over the network? * After closing the game the port shall be closed, and be ready for other connections. To test this, close the game and open it again, and try using the port again by one more game connection. | | | | |
| Chat log | **** |  | All chat functionalities are working fine |
| Does the chat log area show your sent and received text messages fine? | | | | |

## Rework and Follow-up

|  |  |
| --- | --- |
| Work | Result |
| Inspectors reported three problems, and missing parts in the software. Two of them was related to a bug in software related to not detecting the win on last moves. | The issue got fixed. |
| Settings Window was not implemented | Setting window got implemented, and its functionality got tested in single and network based games |

# Dynamic V&V

## Unit Testing

Since this software is a Swing based game, it is not really easy to write test unit codes without GUI. Some of the Junit tests for this software are listed below:

// Test if the game can detect the first available block

**public** **void** checkWin() {

GamePanel gp = **new** GamePanel();

gp.*board*[0][0] = gp.*board*[0][1] = gp.*board*[0][2] = 'P';

*assertEquals*(**true**, gp.*checkVert*(0,1));

}

// Test if the game blocks nonvalid moves

@Test

**public** **void** val() {

GamePanel gp = **new** GamePanel();

gp.*board*[0][0] = 'P';

*assertEquals*(**false**, gp.isVal(0,0));

}

// Test if the game makes all blocks available at the beginning of the game

@Test

**public** **void** startBlocks() {

GamePanel gp = **new** GamePanel();

gp.setGame();

**int** myFlag = 1;

**int** i, j;

**for** (i=0; i<5; i++)

{

**for** (j=0; j<5; j++)

{

**if** (gp.*board*[i][j] != 'A')

myFlag = 0;

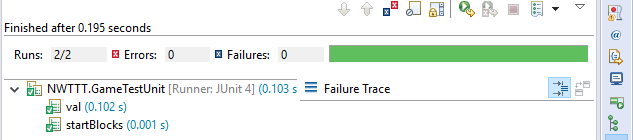
}

}

*assertEquals*(1, myFlag);

}

**Test Results:**

****

## Interface Testing

Interface Testing is performed to evaluate whether systems or components pass data and control correctly to one another. It is to verify if all the interactions between these modules are working properly and errors are handled properly. Here are some test cases that check the connection between different modules:

// Test if the difficulty level affects compMove() method - Easy

@Test

**public** **void** difficultyTest() {

GamePanel gp = **new** GamePanel();

gp.setGame();

gp.*diff* = 1; // set to easy

gp.compMove();

*assertEquals*(**false**, gp.isVal(0,0));

}

// Test if the difficulty level affects compMove() method - Hard (Smart)

@Test

**public** **void** difficultyTestTwo() {

GamePanel gp = **new** GamePanel();

gp.setGame();

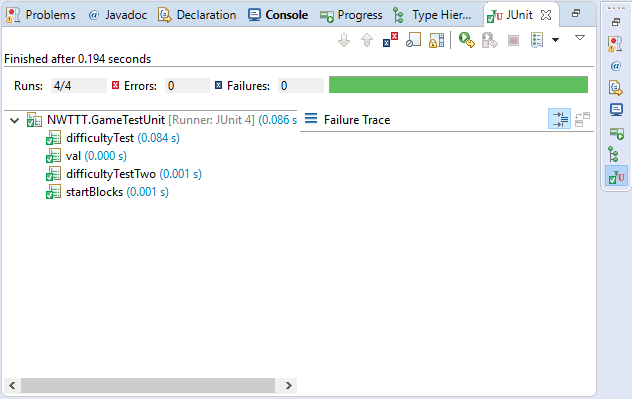
gp.*diff* = 100; // set to hard

gp.*board*[0][0] = gp.*board*[0][2] = 'P';

gp.compMove();

*assertEquals*(**false**, gp.isVal(0,1)); // computer should block the middle block

}



## System Testing

At the testing day, software had lots of bugs, and errors.

* The network part was not written:

This part included network setup, dealing with threads, passing chat messages, moves, and commands over the network, and a lot minor details. It is written now, and debugs are done in the group level.

* It was freezing after random number of moves in computer mode:

Freezing error was related to random moves, which were moving out of array. The issue got resolved.

* Computer was not able to any smart move:

Smart move functionality got added to system, and now the system is able to block opponents winning possibilities

## Non-functional Requirements Testing

## According to project SRS, a number of non-functional requirements are required for this software. Blow is the list of these requirements. Writing a test suit for these requirements are complicated that the project itself, and for many of them, it is not even possible to test them in JUnit. These tests are to be done be programming team members, and inspectors.

## Performance Requirements

* Setting network shall not be more than seeing IP-Address on Server side, and entering the same IP address on Client side. All other settings, including which port is being used, and what are the network settings shall be set in codes
* Submitting the move on GUI shall not take more than a second
* Product shall show a diagonal, vertical, or horizontal winning line, to make it easier for user to see how the winning happens
* Each symbol shall have a different color and/or shape

## Safety and Security Requirements

* Product shall close the network port, at the time of exiting program
* Product can only receive a single connection through the port
* Product shall not make any edit on other software on the system.
* Product shall not provide any data for other software on the system, unless required for product’s main functionality
* Product shall not send any user data to any system other than the client side of the game
* Product shall not send any user data other than product’s functionality commands, and chat comments

## Software Quality Attributes

* Product shall have a re-sizable user interface, so that the symbols and the board re-size do not deform
* There should be buttons and/or shortcut keys, visible for player to start or quit the game with only one click or push of a button (or combination of at most two buttons), to start or quit the game