Team 4 – Project 2

Sprint 1 Retrospective, New Product Backlog, and Sprint 1 Burndown Chart

Members: Patrick Casey, Matt Hacker, and Megan Kerins

CSCE 315-501

When we initially set up the list of things to be done for the first sprint, we really did not realize what we were getting ourselves into. Megan was the only one with Java experience, however, she had never done anything with graphics. So we all had difficulty understanding how to implement the Interface. With Spring Break occurring during the length of the first sprint and all of us in different locations, those of us without the experience had difficulty understanding how Java methods work and how funtions are called. Due to this, we had much difficulty creating the board interface and adding various components to it. Whether to use paint(Graphics g) or panels to create the background, JFrame or JPanels, how to attach buttons, frames, and how to make the buttons once "clicked" go to what window we wanted, proved more challenging than originally calculated.

This is late to be turned in because the team leader, Megan, thought it was due at Midnight and takes complete responsibility.

<u>**NEW** PRODUCT BACKLOG:</u>

- User launches program a window opens with a menu bar at top with buttons and a grey screen with the Title Fanorona. The buttons are:
- "New Game"- starts a new game
- "About"- displays the team members and team number
- "Instructions"- displays the rules of the game. (How to Play)
- "Exit"- displays an "Are you sure?" to which if you click yes, the game exits
- The game will not be started until the user selects New Game, to which they are prompted that all current progress will be lost should they choose to start a new Game against the CPU
- Before the game can be started the AI intelligence must be chosen (possibly before interface is displayed)
- Also, when New Game is selected you will have options:
- An option to play locally against an AI will be displayed
- An option to play locally against another person will be available
- An option to play against a remote opponent (regardless of AI/person) will be available
- During the game there will be buttons on the top of the board reading "New Game", "Board State", and "Undo"
- New Game will simply reset the pieces to how they were when the game was started. No Titles or creator names will display
- Board State will open a separate window of the *current* board state
- If the board state is displayed over the screen, a button will be displayed that says "Back" which takes you back to the game of play
- The initial board state is displayed if the Board State button is pushed before game play
- "Undo" undoes the previous move, this is only possible for the Human player

- Three Buttons will be given as choices when New Game is selected and you play against a CPU
- Create a utility function
- Generate a minimax tree
- Evaluate the tree with alpha-beta pruning one level at a time
- The AI will use the minimax tree, the difficulty level will determine how far ahead it looks
- There might be a “Tree” button for the player to see the minimax tree as well that is available to be clicked during game play
- Easy will make the AI look ahead 1 move
- Medium will allow the AI to look ahead 2 moves
- Hard will allow the AI to look ahead 3 moves
- Our board will have a graphical interface, not command line
- P1 will move by a click-and-move process
- User gets response of invalid if proposing an invalid move
- User also gets a response if they still have valid moves to initiate
- Whether the move is valid or invalid, either a green/red box or simply text (saying P1 or invalid) will display at bottom of screen alerting the player
- After the turn is over, the box will turn red or the text will switch over to AI
- The user sees a display of the current board state, again to exit the button “Back” must be clicked
- Board state is checked to determine if either player has won, and is checked after every move
- Game is checked to determine if maximum number of moves has been exceeded. This too is done after every move.
- Possibly a counter will be displayed counting down the moves until they are exceeded
- In a computer move, it first identifies all valid moves it can make
- AI then checks the board state (it does not appear across the screen this is done internally)
- Computer looks ahead in tree (again, done internally. How far ahead depends on its skill level)
- After every move, Human or Computer, the Minimax tree in the Computer’s class is updated for its best possible move
- Computer can ONLY select valid moves
- Like the user, after turn is over the board state is updated and encouraged to be clicked
- In the Tree’s window (if clicked), the “Back” button must be selected to return to game
- The game is again checked to determine if maximum number of moves has been exceeded.
- Iterative deepening implemented
- Board evaluation function works for determining winner/loser
- At the end of the game the victor’s name will be displayed
- The board will have no pieces on it and the option to play again will be offered

<u>BURNDOWN CHART FROM SPRINT 1:</u>

All team members were to work on the project during the Spring Break.
Therefore, all the hours spent were separate; except the 2 lab meetings before and after the Break.

<u>Researching the Game:</u> 2+3+3=8 (Rather than 4-5. If we had all been working together on this part however, I'm sure we would have done it in about 4 hours).

<u>Researching How to Implement the Interface Using Java</u>: 6+8+10 = 24 hours (This was not in the initial burn down chart for sprint 1, it included looking up examples of other games and reading into how to implement GUI in Java, like whether to use JFrames or JPanels, how to attach JButtons, how methods are called, etc.).

<u>Graphic Board Setup</u>: Rather than the Board set-up, we only got the initial startup scenario implemented. This took up about 20 hours from combined team members. Rather than the estimated 3 hours. We are going to fix this by meeting more and attending office hours. These hours somewhat merge with the Researching category above.

The last two Items in our sprint "Placing game pieces" and "Functionality" were not reached. We had too high of expectations, but for the next sprint we are hoping to have the entirety of our old Sprint 1 completed.

<u>BURNDOWN CHART UP TO THIS POINT:</u>

Main priority was to actually have an interface for the game with all of the pieces on it in the correct positions and correct colors ready to execute when the program is run. This sadly, did not happen. Would we have met over Spring Break it could have been a possibility.

We wanted all the buttons should at least be there, whether they work or not is not an issue. For the first sprint we want the game to have all its pieces ready to implement.

This was not the case, a lot of time was spent on actually learning Java GUI implementation which set us back a couple hours. The time for the Startup Scenario and New Game Options are sort of merged, same with the Fixing Code; because we are all novices when it comes to graphical Java.

Time Estimation:
Total initial graphic Interface: 28 Hours **In Progress..**
Startup Scenario-18 Hours **COMPLETED in individual hours (20+8+10= 38 hours)**

New Game Options- 5 Hours **IN PROGRESS, Patrick worked 10 hours on this and the above (Startup Scenario).**
Board States- 5 Hours **X**
Game Implementation: 35 Hours **X**
Minimax Tree- 8 hours
Piece Movement- 5 hours **X**
Board State updating-7 hours **X**
Game State checking- 10 hours **X**
AI difficulty level- 5 hours **X**
Client Server Interface: 20 hours **X**
Client end- 10 hours **X**
Server end- 10 hours **X**
General debugging and fixes: 32 hours **LOTS OF TIME ON THIS SO FAR, HARD TO SAY. IT WOULD HAVE TO BE indiv (4+2=6 hours). This is only from the first Sprint, and is somewhat combined with the other categories.**

Total Estimation: 125 hours **-- Time Spent So Far: ~44 hours**