CS 1530 – SPRINT 4 DELIVERABLE

Software Engineering Cowards

Brian Ashworth – RedSoxFan

John Johnson – jdj20

Edwin Mellett – WynWinz

Aaron Tamenne – atamenne

Name: Brian Ashworth

https://github.com/RedSoxFan/2174\_CS1530\_SoftwareEngineeringCowards

(15 MAR 2017)

SPRINT 4 ACCOMPLISHMENTS

This spring the team communicated via a group text. This is the same way that we have been communicating since the beginning and appears to be working fine for us. We have not have any disagreements this sprint and we have also not changed much since the first sprint. Nothing was changed this sprint in terms of our process, but we will probably be focusing more on refactoring as we go since there seemed to be a good amount that was in drastic need of refactoring. Most of this was due to integration and working on things in parallel. Other changes were simply larger methods that should be split up into smaller easier to manage ones.

This sprint, I worked on refactoring the graphic user interface code. I wrote it at the very beginning, but it was awful to look at. This seemed like a good time to clean it up. The biggest problem with refactoring the code was figuring out the best way to do so. I decided to split each section into a separate method. I also extracted commonly used code into helper methods instead of repeating them. This will allow us to easily change what is displayed and will allow for images and more polished graphics in a later state.

We also noticed after refactoring that we could no longer move the king to any special square. After debugging, we found that we were missing tests for these cases and some bugs slipped through code reviews. I created a fix for this and also created the missing test cases. In the process of writing the tests, I also realized that moving a king to a corner no longer resulted in a game over state. I was quickly able to find the issue, resolve it, and create the missing tests for that as well. From this, my team and I have learned that we need to be more careful when looking at test coverage, when refactoring, and when performing code reviews to make sure that we do not have similar issues in the future.