Project 7 - Stratego
OOAD - Final Project
Vernon Walker and Victoria Bockman

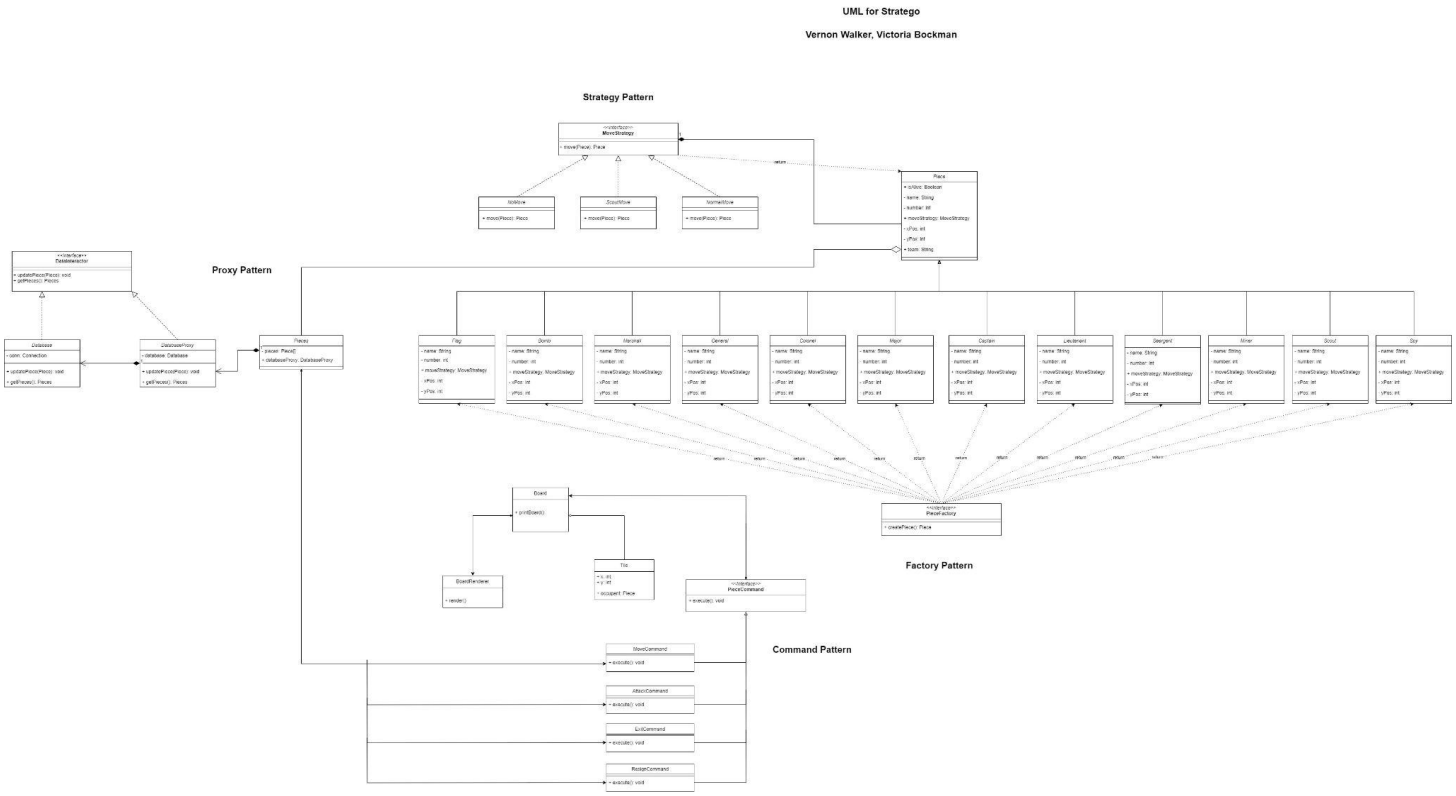
Final Project Report

Final State of System Statement:

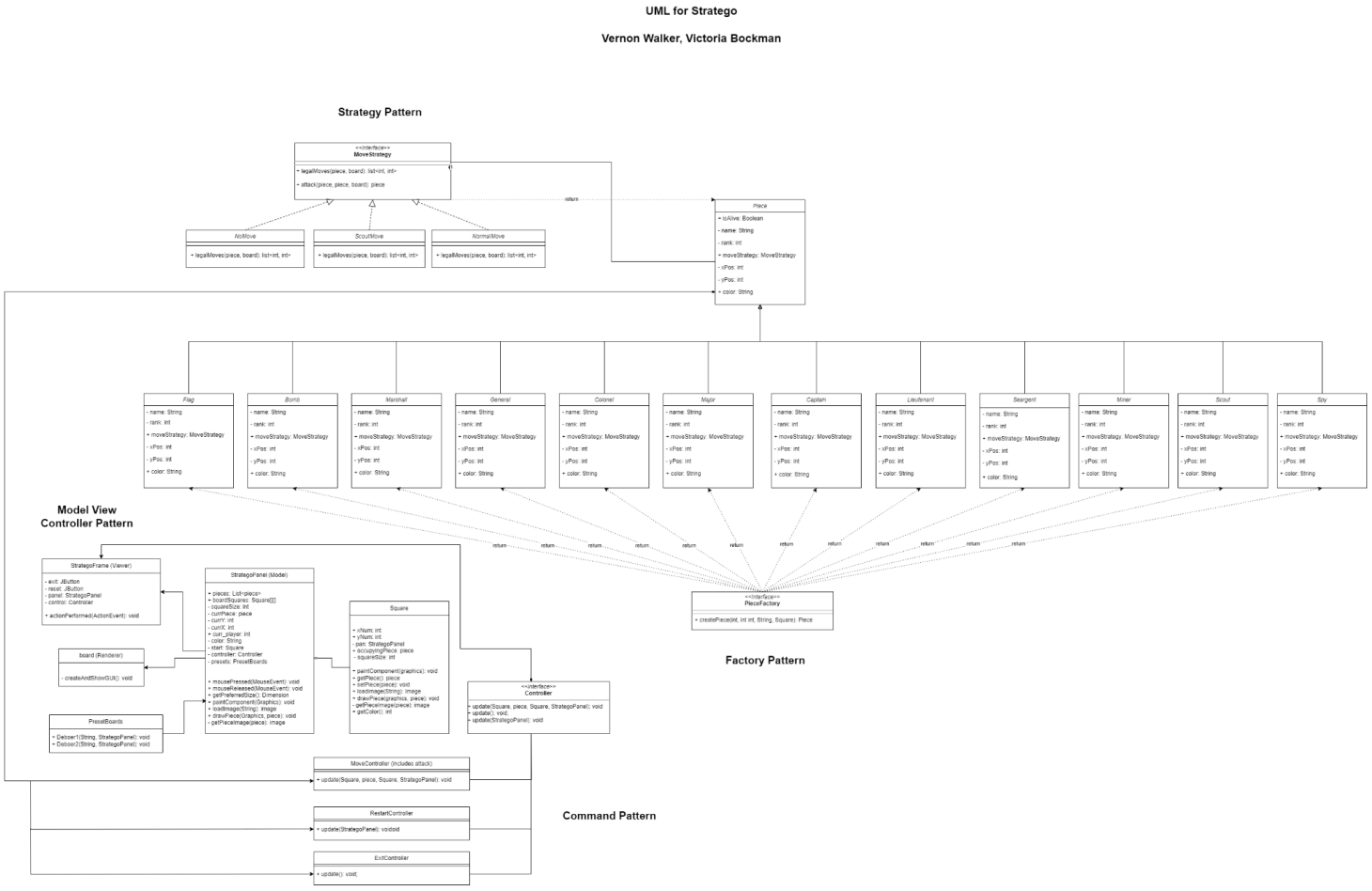
The final state of our project is a little different then what we originally wanted to do. First off, we decided to not implement the Proxy pattern and instead opt for a Model View Controller. Before, we were hoping to implement playing the game on a single computer while using two monitors. Towards the end of the project, we decided this might be a little above and beyond our goals and reduced the gameplay to a single screen - even though it might make it a little harder to play and reveals the enemy pieces. Without the need for the Proxy, we also didn't store our data in a MySQL database. Instead, we have all of our data kept in memory throughout the running program. There was one other rule for an official Stratego game that we didn't implement, and this was restricting movements so you can't move back and forth from the same squares over three times in a row. Other than that, all of our plans and capabilities stayed the same.

Final Class Diagram and Comparison Statement:

Original UML diagram (also in Github):



Updated UML diagram (also in Github):



Key changes:

- No Proxy Pattern
- Added Model View Controller Pattern
    - Subsequent interface and other classes
- Some of our original classes grew in size (the Tile/Square class, and the Board/Stratego Panel class) as we added functionality

Third-Party code vs. Original code Statement:

When we started our project, we used some beginner code to help us from Stack Overflow: https://stackoverflow.com/questions/53593173/gui-manipulations-for-board-game. As we started working with this code we ended up changing many of the elements to help with our specific requirements. We also looked at various other examples besides this one for help understanding how all the different pieces work but they didn't make an appearance in our code.

We continued to use Java AWT and Java Swing for our graphic UI as we started development. Much of our actual board representation was made using these libraries and frameworks. Something that also helped us create the Mouse Listener and rendering the objects correctly was looking at example code for a chess game created with java: https://github.com/jlundstedt/chess-java

Statement on the OOAD process for your overall Semester Project:

When working throughout this final project, we used several OOD principles in our design process. The Single Responsibility Principle not only helps make your classes less complex and easier to manage, but we also noticed that this helped us organize our code better. We made very clear choices for what each class does and created new classes or modules to deal with separate components. Another design principle we used was programming to interfaces, not implementations. When we did this, we noticed that much of our coding was handled by the interface directly, which meant less repetition (DRY principle) and less work for us. We also attempted to encapsulate what varies with the use of private and protected variables, however, with such a large amount of files that are interacting with each other, it was sometimes easier to just make the variables public. This isn't a hard principle to abide by, but it does require a bit more work than we had time for.

Code Submission

- ☑ ~~The code should be well structured and documented with appropriate comments.~~
- ☑ ~~Uses of OO Patterns or other design principles should be noted in the code, and any third-party elements should also be noted (with URLs or other citations).~~
- ☑ ~~Include a basic README Markdown file with the names of team members, the language version, and any special instructions to run the code (graders may request assistance from you during review)~~

Demonstration in Github

- ☑ ~~The recorded video should be brief, 10 to 15 minutes; all team members should participate. Zoom is an effective way of sharing a screen for your application and allowing the team to comment on the work while recording to an MP4 file. Include the recording in your repo or provide an external link for viewing.~~