Changes to Design

In the original design document, the team planned to have two high level entities: the client and the server. The client would handle the GUI and send the player's moves to the server, while the server would handle game mechanics and the use of AI. These entities and their roles remained essentially the same throughout the design process, as the description of what each entity does is very general. As with the first project, as the abstraction level was lowered, the adherence to the original design was changed as well, due to the fact that we needed different design of classes and functions for the final project. For example, in the final project we did not need the "Player" class, even though we created the class when we originally planned out the project. There were several other minor changes made throughout the project, although the creation and later removal of the Player class was the most significant. The next most significant was the fact that the AI class was rewritten almost from scratch several times because of the way that the server sent and received information.

The low level entities carried over to the final project relatively similarly to the original design document. This is because the team decided to go with a slightly higher level of abstraction for this design than for the design of the first project. This resulted in the team having a weaker base to start with, but it also meant that the design document accurately represents most of what is it in the final submission.

Issues Encountered

There were many issues encountered in this project, so it was significantly more difficult to finalize than the first project. One of the largest issues encountered was dealing with the communication between the client and the server. Only one of the team members had ever had any experience with servers before this class, so she was put in charge of setting up the server. Unfortunately, her knowledge of servers was not enough to prevent the myriad of bugs the team encountered when trying to send information between the client and server. These were mostly small bugs, but the resolution of them consumed almost all of her time, resulting in the three other members of the team completing the rest of the project. Although her work was incredibly valuable and necessary, it was unfortunate that she had to spend so much time fixing the server purely because she had very limited experience with servers.

Another major issue for the team with this project was the timing of the submissions. Because the main logic of the project (the AI and game mechanics) were assigned during a two week period when two team members had major projects and tests and one of the members attended an academic conference, the AI and game mechanics implementations were not completed on time for the submission deadlines. As a result of this, the team spent the majority of the last week attempting to fix what should have been completed in the first place, leaving us precious little time to fix bugs and work on the GUI.

Another major issue with the project was the fact that it was assigned immediately after the first project. Because the first project was submitted late, the team was not only exhausted at the beginning of the second project, but was also already behind in regards to how much time they had to implement the pieces of the second project. In future projects, it would be nice to have at least a few days of rest before the second assignment was assigned in order for the team members to have time to regain their energy and start thinking about the implementation of the next project. As it was, the team did not have time to really sit down and think about how the project should be implemented because there was always a deadline looming around the next corner.

Lessons Learned

Our team learned a lot about working on a project where the requirements consisted of elements that none of the members had encountered before as well as working on a project where sacrifices of functionality would have to be made in order to preserve the integrity of the final submission. These are summarized as follows:

1. It is important for team members to communicate their schedules well in advance in order to prevent one member from saying they can't finish a piece and then handing it to the rest of the team.
2. If a team member does not have the knowledge to complete a part of the project,they need to inform the other team members immediately so that the other team members can help them or find them help as soon as possible.
3. Sometimes functions have to be cut in the final submission in order to keep the final project intact and make sure it completes at least most of the requirements.

4. Sometimes it is easier to rewrite a section of code rather than spend a long time attempting to build on top of it. That said, it is also important not to waste time re-implementing something that worked perfectly fine in the first place.
5. It is important for every member to have a specific task to complete. This prevents team members from wasting time with a general objective and not knowing exactly what they need to work on.
6. It is important to set a deadline for code to be completed to several days before the final submission is due because there are always bugs that need to be fixed and these bugs generally take longer than expected to fix.

Work Load Distribution

Howard Cheng

Portion of project completed: 24%

Worked on: GUI, testing, design document

David Cross

Portion of project completed: 24%

Worked on: Game Mechanics, Java Client, AI, testing, design document, post production notes

Noemie Nakamura

Portion of project completed: 28%

Worked on: C++ client and Server, design document, Game mechanics, GUI, Java Client, AI, testing

John Pickering

Portion of project completed: 24%

Worked on: AI, design document, testing