CMPT 370 D1

CONSTRUCTION DOCUMENT

Nov 26, 2016

Contents

1 INTRODUCTION

This Construction document has been created to communicate to the client an overview of the activities that took place during software construction of the Robot Game system. It includes a code review, individual reports on paired programming, and changes made to the system since the design phase.

1.1 TEAM MEMBERS

Resource Name	Role
Dimaano, Nico	Developer / Tester
Neijmeijer, Niklaas	Developer/ Tester
Seidenthal, Kyle	Developer / Tester
Sterma, Brendon	Developer / Tester
Zang, Jiawei	Developer / Tester

2 CODE REVIEW

For our code review, we met to review the game class in our system. This is the largest piece of the system, and acts as the controller. We felt this was a good choice for review because it is the most complicated class, and it interfaces with almost all other classes in some way. Much of the game logic and communication takes place in the game class. Each feature interacts with it in a specific way, and understanding the layout and design of all the functions within is crucial for further development.

We booked a room in the spinks building with a projector so we could display the code for everyone to see. We had the group members who wrote the functions in the class describe an overview of what the function does, and then delve into more details. We asked a lot of questions, and made sure everyone was on the same page with each piece. We also discussed future improvements and plans for the class.

We mainly discussed the effectiveness and functionality of the code itself. We discussed proper commenting and documenting procedures as well. We had discussions about how the game will communicate with the server, RobotLibrarian, and the view. The turn ordering for robots and the system used to keep track of it was also discussed.

We benefited from this by having an all hands meeting where we had the opportunity to focus on the controller part of the game. We got the chance to see the current state of the game, and what needed to be fixed and added. We feel that this time was used well to improve the code for the game class. We also found and fixed many small bugs that will save headaches later.

3 PAIRED PROGRAMMING

The team used paired programming for parts of the construction phase. Each member took part in two sessions, each with a different member of the group. One member was to navigate while the other sat at the keyboard, then they would switch at some point in the session. Below are the reports of each member.

3.1 NICO DIMAANO

Paired with Kyle

Me and Kyle were going to work on the Networking system for our project. Before the session, me and Kyle talked about the design of our system and what goes where. First Kyle was the driver and I was the navigator. We first typed up the basic structure, functions, and variables we wanted in and the two main cases (Client and Server). After a while I was the driver and Kyle was the navigator. We came across a problem that functions were not working together and some threads were not coordinated and we fixed our design a little bit and worked on it again. We then finished our coding. What's left is debugging and have it work what it was intended to do.

Paired with Vigor

Vigor and I decided to fix the view of the game. First, we did the basic's grabbed some pictures for tanks and read and tried to understand the code we will be fixing. Me and Vigor didn't have a consistent driver or navigator it was more who had an idea and tested while the other person commented on it. We worked around the code of the view in changing the colours making the game look pretty and some selected default functionalities. We then noticed our board is small so we had to look into changing multiple number in the file to test if it changes the board to look like the size we were going for. We then fixed some padding errors and misplaced numbers.

3.2 NIKLAAS NEIJMEIJER

For my first paired programming session, I was partnered with Brendon. In this session, we started, tested, and finished the Board and Tile classes of our program. There were two things I didn't consider at the time that I feel hindered the effectiveness of the session. The first was that, unlike my last time paired programming, I am not working with someone I have coded with for the past seven months, resulting in me making assumptions that delayed our progress. The second thing was that since I do not have eclipse set up on the lab computers, we were using my laptop to code. This meant that when I became co-pilot Brendon had to use a machine he was unfamiliar with, causing more delays. The benefits of the session were that we had someone else to double check any math that was used, and when the pilot encountered a bug, the co-pilot was the one to find the solution. The two major reasons I feel like we accomplished more together than we would have individually were that we could run any changes that relate

to other people's by the partner, rather than unilaterally changing it, and I, personally, am less distractible when working with someone else than I am when working alone.

For my second paired programming session, I was partnered with Kyle. We began working on the controller, though the amount of coding we were able to do was minimal, the primary benefit was communication. Since we had respectively worked on the view and the networking portions of the project, and the controller was what connected the two of them, we were able to clear up potential misunderstandings, so that after the session was over, we were able to accomplish more than we would have without the session.

3.3 KYLE SEIDENTHAL

For my first paired programming session, I worked with Nico Dimaano. We decided to focus our session on the networking piece of the software. I started out as the driver. We quickly found some of the flaws with our design for the server class, and decided to break it up into two separate classes: the client and the server. We got quite a bit of work done. As we worked through how the system would work, he helped me navigate around many errors that may have caused headaches later. After about an hour, we switched places. I helped navigate through the code we had already written, and pointed out flaws in some of the logic we had come up with. Overall, we got a tremendous amount of work done. In the two hours we allocated for this session, we had completed most of the fundamental functionality of the server and client features. I would definitely do this again.

For my second session, I worked with Niklaas Neijmeijer. We decided to tackle to controller with our session. We made a lot of progress with the functionality. Because I had worked on the networking side, and he had done many of the view elements, we had a lot of good arguments about how to make the controller work well with both elements. This was very productive, and together we sidestepped a lot of headaches that would have been caused by different understandings of how some of the functions should be implemented. By combining our knowledge, I think we came up with the most effective solutions for the communications between the view, controller, and server. This was again a very successful session, and I would do it again in the future.

3.4 BRENDON STERMA

Brendon And Vigor

We met together to confront and solve an issue we were having with reading json files. While we were working, we were able to explain ideas to each other much easier than we would be able to over text. We were able to find out what problems we were facing and how to come up with a solution for our next work period. We plan on having another session with each other once we have found a solution in the library documentation.

Brendon and Niklaas

We met to put together the code for our board class, and various other classes that were involved in it. We were able to communicate our ideas more clearly than we could before, and our understanding of the program increased as a result. We were able to effectively troubleshoot many problems we encountered with Eclipse and were able to devise a solution in only a few hours. We plan to have more such sessions to continue building the system. We accomplished more together than we may have working separately because we would not have the second opinions of the other person to drive us along**.**

3.5 JIAWEI ZANG

I was working in collaboration with Nico Diamaano to improve the view performance by adding CSS files. During pair programming, we tried a lot of methods to access the CSS file, and we did a lot of research about it. Then we found out by accessing files by File objects, which will also work on showing the image and reading JSON file. Then we added few classes applied into button, background, and font style. And we also fix some issues in the view originally in the code.

I also worked with Brandon on JSON and some extra work about view, firstly, I did some research and I found that Google have JSON library which called "Gson" in java, and I taught to Brandon how to code, such as read JSON file and write JSON file by using Gson. However, something is wrong about it, the method I used to access CSS file is not working for accessing JSON file, therefore, we tried to do external project which just read and write JSON, and we finally succeeded. Then we create additional package which will holding all JSON reader and writer which applied into robot to wait for server and controller finished.

4 CHANGES FROM DESIGN

One major change we have made since our design phase was to the networking side of the system. We realized that what we had just would not work. We have decided to break the server class up into a server and a client. This allows us to run one server thread on a machine to host all clients connected to the match. Many new functions were created in the game class to support this. We have also decided to make the client and server part of the controller, because they communicate directly with it to maintain the state of the game.

Another thing we changed was the addition of a player and observer list to the board class, as well as the two variables we use to keep track of who's turn it currently is. This was to allow us to simply send a copy of the board over the network for most of the changes that will happen during the game. This also worked with the updateGame method that was added to the gameView class, which takes in the board and determines what to display based on what is contained there, rather than needing several attributes. Lastly several functions were cut from the user class and its children because they were simply things the controller did, and didn't need anything from the users.