CMPT370 – Oct 21 /16 @ 12:15pm – 12:45pm

Meeting 14:

Agenda:

- Figure out the changes in the existing architecture, finalize class descriptions
- Decide how to split up the work
    - Arianne: editing, plus changes made between requirements and design
        - Point form please, hard info
- Make template for structuring the UML
- Download/test Draw.io for UML diagram

Meeting Notes:

- Going over meeting with Dr. Dutchyn
    - The only thing that's downloaded from JSON is the AI function for the robot
    - Interpret what the function does with the Forth Interpreter
    - At the start of the semester, Dutchyn said that whether or not we use messaging is irrelevant, but the functionality needs to be there
        - For messaging: receive message (receives string), send message (sends string)
        - Inbox should be protected so that other robots can access and add to the stack
    - JSON downloaded from robot librarian, is private, when a robot is instantiated, store it in a team (team is stored in a team list)
        - From the JSON file, load max health etc. and store inside robot
            - Health is stored and is private, only robot itself can know that
            - Has GetHealth() and IsAlive() function
            - Damage inflicted will be stored in JSON, which is an instant look up dictionary
        - Action methods:
            - Move()
            - MovesRemaining()
            - Turn()
                - The int is the amount of times to turn
            - Shoot()
            - Play()
        - The team itself and the robot is stored in JSON
- Structure for UML:
    - Intro
    - Purpose/functionality
    - Variables
    - Functions
- 1 paragraph per class (7 – 9)

- Certain functions that aren't self explanatory need descriptions
- Diagrams:
    - MVC diagram – high level components
    - UML diagram – class specific
- Each class gets uploaded to git in this format:
    - Class_Classname.txt