Defence Science and Technology Group (DSTG) and Swordfish Computing Project

Distributed Decision-Making

THE UNIVERSITY of ADELAIDE

Sprint Retrospective 4 – a1734056

Group COMPLEX 8

a1734056	Hayden Lee
a1734069	Vinh Nguyen
a1743599	Nathan Van der Hoek
a1744852	Harrison Bagley
a1746088	Daniel O'Connor
a1746146	Patrick Capaldo
a1748751	Sarah Damin
a1749935	Sam Davies
a1773841	Hayley Richardson

I attended the sprint review/planning meeting on 17/10 with the tutor

What went well in the sprint?

The team worked smoothly and efficiently throughout the fourth sprint. Progress within subteams was well underway, and so the development team members were motivated to meet more frequently and worked with more organization. In doing so, good groundwork was being laid out for each team. This also helped the division of subtasks so that everyone was being productive and able to make progress. With a good foundation to work from, it was much easier to sensibly distribute the tasks more evenly. This was all evidenced on the project GitHub. The project board was more detailed and active with issues being opened and closed frequently, and the code repository had a lot more commits and pull requests. When discussing and consolidating our progress, we were also engaging with James, the product owner, a lot more which was great to help us better understand what tasks needed to be done and to ensure that we all shared the same vision for the upcoming sprint.

What could be improved?

The team observed that there was a lack of communication of progress throughout the sprint so this was identified as a key area that could be improved. This could be remediated by improving the documentation for work so that individuals from other subteams could read and catch up with everyone else's work. Another potential area for improvement would be to spread out working on SEP throughout the week instead of crunching it all in at the end of the week, as this causes difficulties if unforeseen external circumstances arise on the weekend. This could be improved by better time management/allocation. Coordination with other team members could also be improved, which would also facilitate ease of integration of different components between subteam members. This also includes finding common time to set together and work on things, as well as proper git merging processes.

What will the group commit to improve in the next sprint?

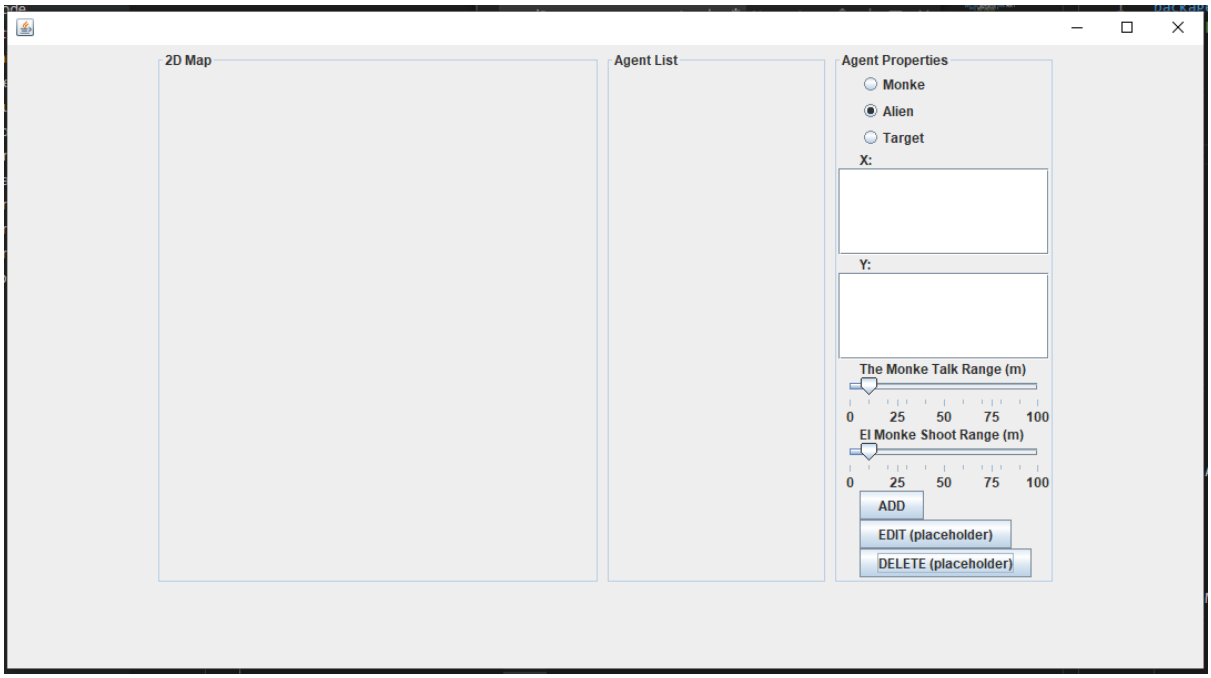
To be explicit about when group members have time to work on SEP, the group will commit to posting deadlines for upcoming assignments. This will keep everyone informed of the development team's priorities as well as the times when they will be able to work on SEP. The group will also commit to having an early deadline for an mvp, which has been currently set to Monday 24/10. This will allow for ample time to fine tune any further issues throughout the proceeding week. To help keep other subteams informed of progress, each subteam will commit to posting minutes/progress updates in the team's main slack channel for the entire team to see. Ideally these will be posted throughout the week so that everyone can stay up to date, rather than having to read everyone's 'last minute' summary.

Comment on your progress this sprint

My tasks:

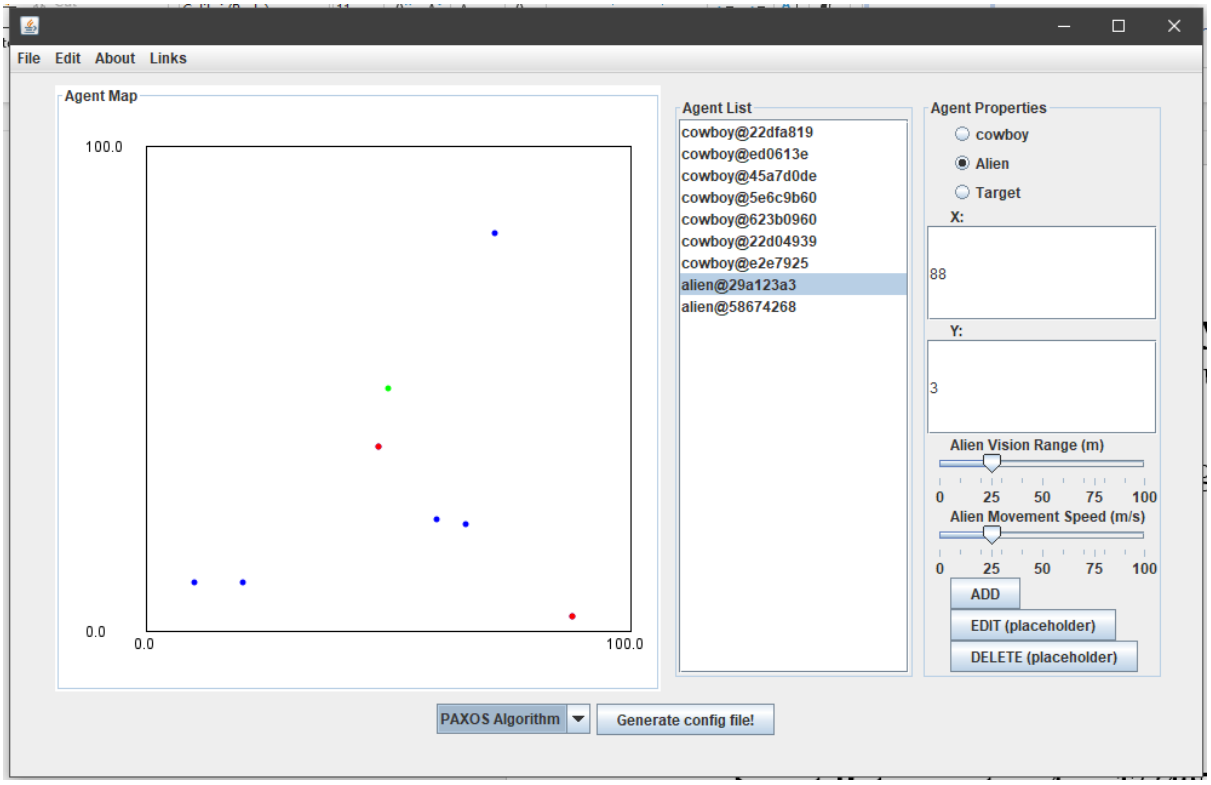
- Configuration Profiles #3
- Frontend GUI
- Easy Configuration #7

A thorough refactoring of the code was accomplished. Instead of cramming all the code into a single spaghetti-mashed java file, the code setting up the JComponents of each of the three panels was distributed into three java files. All the functionality that connected each panel was controlled by a main file, and the alien/cowboy classes also had their own file. This made software development significantly easier, as it made coding much smoother and easier to navigate. At the end of the first week of the sprint, this was what the frontend GUI looked like:



After the code refactor was complete and we reacquired the baseline functionality that we had from the previous sprint, I pushed these changes onto the GitHub so that the other subteam members could immediately start working on it and incrementally build on it. During this time, I also did some research into a popular design pattern called the observer pattern, which is analogous to a design methodology called model/view. This would definitely be the proper approach to make our program extensible/scalable, but the learning curve was too great to pursue. Instead, it was easier to implement the program so that it was entirely event-driven with button/mouse clicks instead, although model/view may be an avenue for exploration in the future.

I also developed a method for the user to keep track of the agents as they are created in the form of JList. This visualises each agent, and allows for each one to be selected which then updates the agent properties panel with the appropriate information. Here is a screenshot of the frontend GUI layout:



The complexity of the tasks through this sprint was relatively low. Most of the refactoring changes were straightforward, as it mostly involved capturing functionality that had previously already been developed. Afterwards, creating new features for the GUI simply required some trial and error. These new features were also double checked to be adequately connected and synchronised with all previously existing elements of the system.