RMIT ROBOCUP TEAM INITIATION

Programming Project 1 – Flexible Summer Semester 2018

Ryan Couper, Ran Lu, Louis Stekhoven-Smith
s3575400, s3583185, s3530225 respectively

Table of Contents

Project Description 2

Major Deliverables 2

Project Road Map 4

Minutes of weekly planning sessions 4

Overview 4

Early Alternatives 5

A Set Path 6

The Achieved Pathway 8

The Burndown 9

Trello boards screenshots 10

The Screenshots 10

Issue and Risk Registers 15

System Design 16

Overview 16

Software Architecture Diagram 16

User Manuals/Project closure report (if needed) 18

Learning Outcomes 19

Enabling Knowledge 19

Identification of New Knowledge and Skills 19

Project Management 20

Critical Analysis 20

Problem Solving 21

Communication 21

Appendix A: Bibliography 23

Project Description

RoboCup is an annual international robotics competition including both standard platform league and simulation leagues. The aim of this project is to setup RMIT and its future students with the tools and information needed to begin contributions to the development of autonomous humanoid robot soccer AI, acting as an initial stepping stone towards winning a RoboCup World Cup.

As a consequence, the objectives of the project is to identify existing software and tools that can be used to rapidly create a codebase for Nao robot soccer and as such act as a launching pad for further development. This project's primary focuses are to: create and document a development environment , create documentation and user guides on the development pipeline and give recommendations on what future teams can work on. The primary goal being to smooth out bugs and development operation issues to allow rapid on boarding of future development teams.

Major Deliverables

- A VirtualBox image with the development environment pre-setup. Includes pre-installed simulation server, monitor and the complied source code.
- A Github repository with the source code ready to be developed. This can be found on our Github here: https://github.com/RMIT-RoboCup-Standard-League/PP1-Nao-Soccer
- An installation manual explaining how to set up a development environment from scratch or how to use the existing VirtualBox image. Also includes system specification requirements. This is in a filed called "up_and_running.pdf".
- A developers user guide. Includes high level diagrams, code examples and traps to avoid. This can also be found in the file "up_and_running.pdf"
- Recommendations on improvements to be made both in operations and software development. This is in a file called "Developer_recommendations.pdf"

Note: Deliverables can be found in the folder named "Deliverables" that accompanies this report except the VDI, which is 12GB. It can be found here:
https://drive.google.com/open?id=1NkRU0_Pc2tCodKehz9uFYxsa8qFy5oPh

Library files for the build are currently hosted on a student's RMIT Google Drive here:
https://drive.google.com/drive/u/1/folders/1tgJ3LX_x4D3dDWfczn3vq-W-8sGQMeyE

These files should be permanently relocated to an RMIT server space to prevent access issues in the future.

Project Road Map

Minutes of weekly planning sessions

All meeting minutes can be found in the folder named "MeetingMinutes" that accompanies this report.

Overview

Ongoing weekly communications with the team mentors assisted in directing this project towards a research/documentation dominated scope rather than that of software development.

The first two weeks were focused on research into the problem space and defining the project at hand. Works included a broad overview of available pathways plus a focused research section on desirable pathways. It was during this time the project team identified the University of New South Wales open source code rUNSWift. This code base has been developed explicitly for Nao robot soccer. Sprint one began in week three and revolved around getting the rUNSWift and Simspark (a virtual environment and server used running robots) up and running as well as attempting to activate some skills on the simulated robots. Due to several unexpected roadblocks getting rUNSWift running took significantly more time then expected and as such the latter of these tasks were pushed into the second sprint in week four.

Sprint two's listed tasks proved unachievable due to continued issues with the rUNSWift codebase. Several bugs where causing progress to be slow, it was not until the start of sprint three were we able to get the code working. Using a completely unknown codebase makes it difficult to determine time frames and it was discovered that we had have massively underestimated the effort required to get the codebase working. As such much of the development of new behaviours was descoped and the project was refocused on delivering a solid base for future teams to development on. Week six, was allocated to finalising deliverables and the project report ready to be shipped.

Early Alternatives

In the beginning, the project roadmap was simple. Preliminary research and talks with people already working with the Nao robot and the RoboCup Soccer League indicated getting the existing UNSW codebase for rUNSWift up and running would require minimal research and effort. At the end of the first week on this six week project, our roadmap looked similar to Figure 1 - The Original Roadmap.

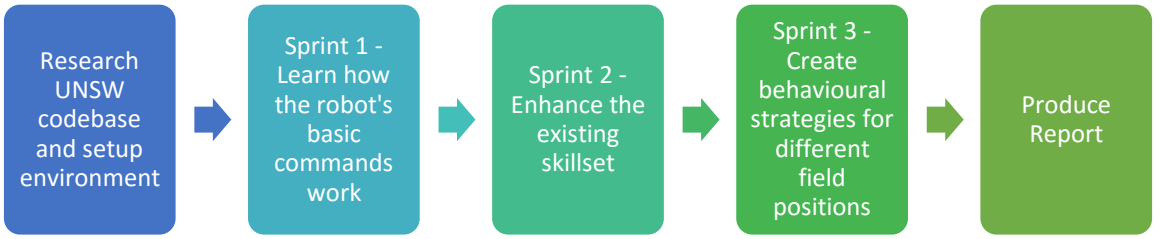


Figure 1 - The Original Roadmap

Meeting minutes at the end of week two indicate significant roadblocks impeding the environment setup. Further research into the rUNSWift revealed that the 2017 release was particularly buggy and was released unfinished and largely untested by the UNSW team, indicating it would require considerably more effort then anticipated to get it up and running. More information on the issues encountered can be found on RMIT's Nao RoboCup Soccer League's Github Wiki under troubleshooting (here: https://github.com/RMIT-RoboCup-Standard-League/PP1-Nao-Soccer/wiki/Troubleshooting).

At this point, after discussions with the project mentors, it was decided that if the environment could not be setup by a specified date, the rUNSWift code would be sidelined in favour of developing RMIT's own RoboCup Soccer League codebase from scratch. This alternate pathway would have altered the roadmap to look akin to the activity diagram in Figure 2 - Alternate Decision Pathway.

Early in the first sprint (week 3 of the project), the team managed to get the simulation environment built and major parts of it working. This was enough to solidify the decision to continue with rUNSWift

as a launching platform for RMIT to its own development.

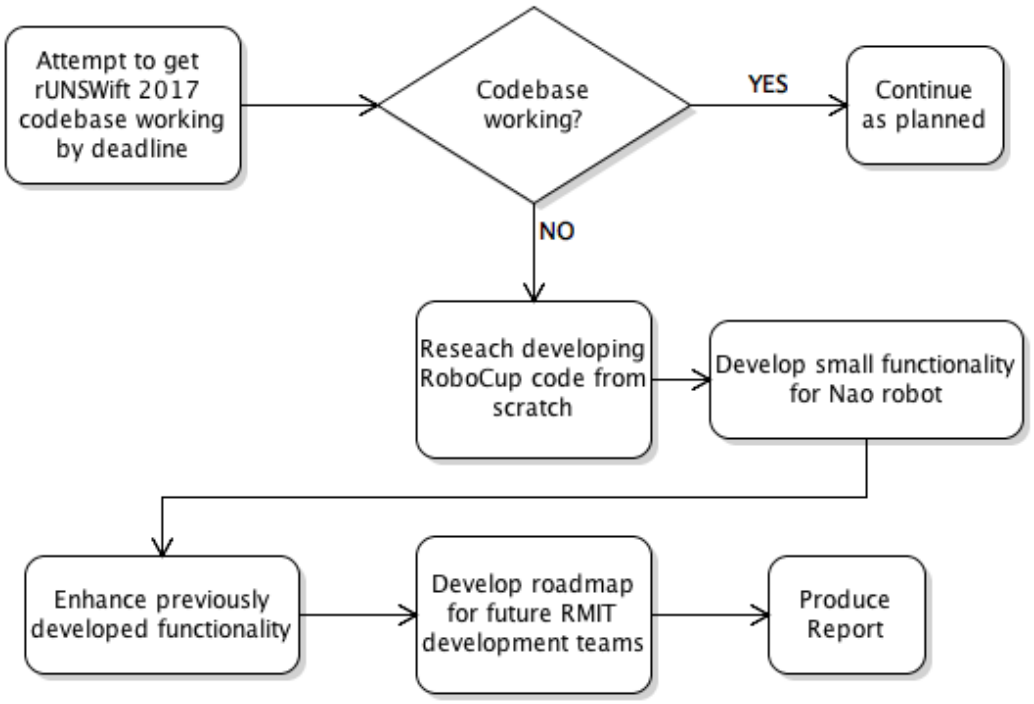


Figure 2 - Alternate Decision Pathway

A Set Path

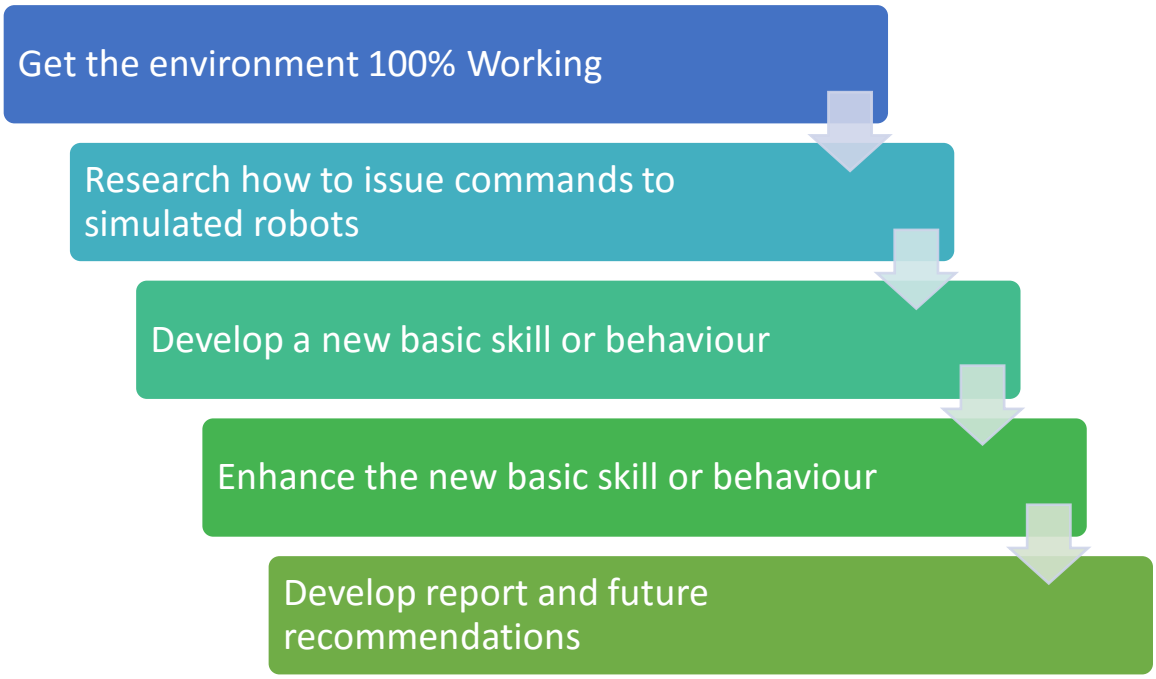


Figure 3 - The New Roadmap

Knowing from which codebase the project will be launched, the team was able to clearly define what the project would set out to achieve. Figure 3 - The New Roadmap shows the plan from this point.

The sprint retrospective meeting minutes from the first sprint indicates the teams failure to accurately estimate the workload. The sprint proved to lack significant progress while the team continued to work independently in an attempt to get various aspects of the environment operational. The sprint's end goal of getting the simulated robot to move was not achieved in Sprint One or in Sprint Two, despite the team's best efforts. The rUNSWift codebase and the tools used had considerable path issues and missing dependencies which required the team to scour the code to find the relevant issues. To further accentuate the issue, documentation on how to use the code was scarce, and unified modelling language diagrams on the existing code for reference were non-existent. More on these issues can be found in the bugs listed in the Github issue tracker (here: https://github.com/RMIT-RoboCup-Standard-League/PP1-Nao-Soccer/issues?q=is%3Aissue+is%3Aclosed).

Due to these roadblocks development of new behaviours were descoped with a more focused effort now being made in getting the rUNSWift code base stable for future development. In our sprint retrospective meetings it was determined that the way we were approaching the problems as a team needed improvement as the dynamics of how the team worked were changed. The team agreed that for spike related tasks each member would simultaneously conduct their own research and then compile our findings as a team. Further major bugs would be tackled through pair programming as this had shown better results in solving such issues. These decisions helped the team reach the milestones of a working simulated robot by the start of the third sprint and established a clear pipeline for activating and developing behaviours.

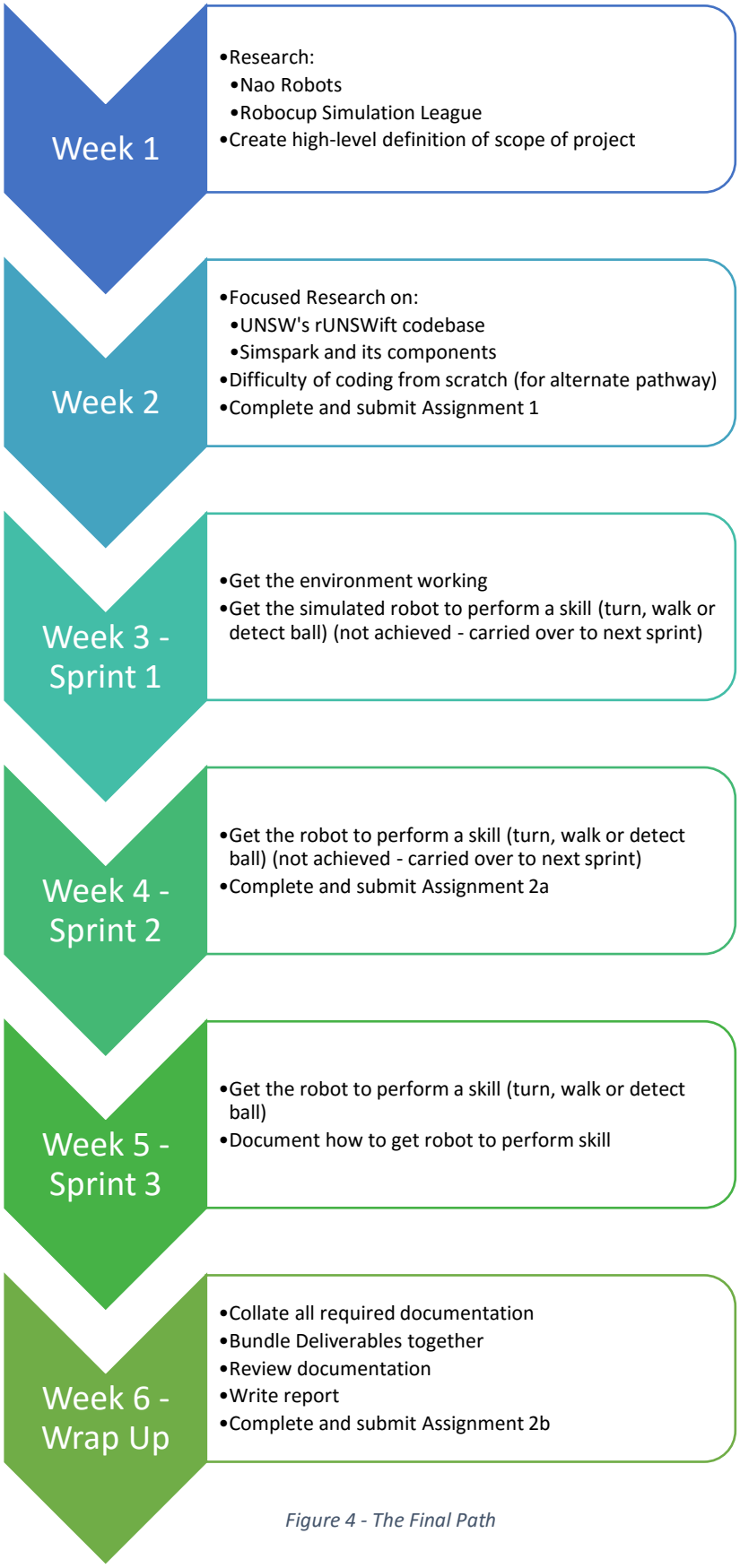


Figure 4 - The Final Path

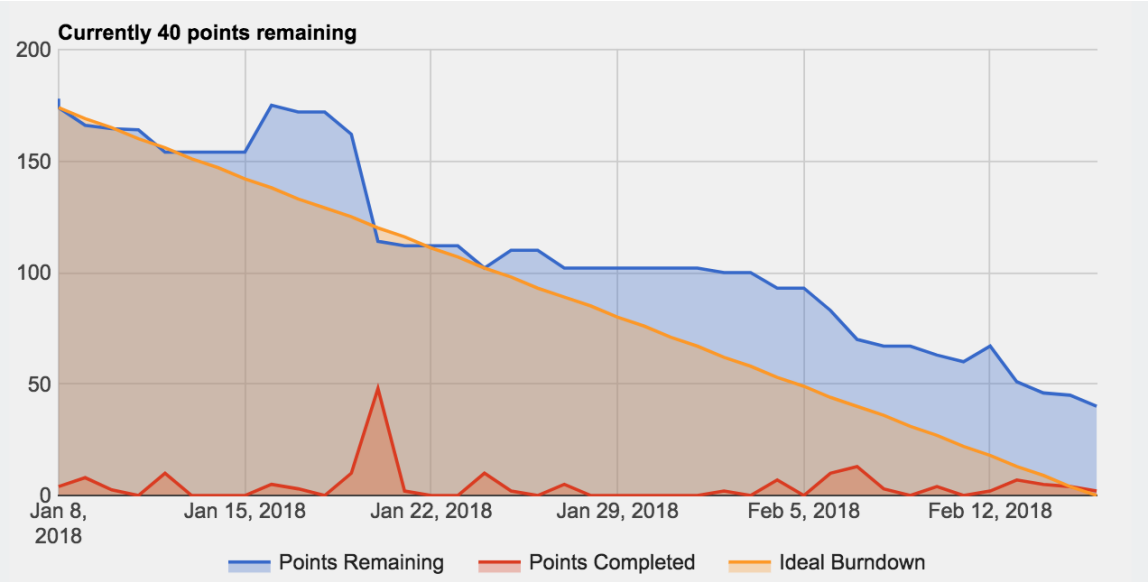
The Achieved Pathway

Sprint three represented the last week of actively working on rUNSWift . Followed up by the project wrap up where deliverables and documents were refined and finalised.

The team worked hard in a final push to develop a workable piece of code that can be demonstrated to newcomers and may give future project-potentials an idea of what can be done if they were to join the RMIT RoboCup Standard League development team.

Many pathing issues, bugs and dependencies were fixed in rUNSWift and as such if the developed user-guides are followed properly, a new developer can be setup and issuing commands to a simulated robot within a day. Figure 4 - The Final Path displays the actual path taken by the team throughout this project.

The Burndown



The Burndown chart is the closest representation to the actual work required. It shows in week two the considerable increase in work required suddenly as the team was made aware of more issues with the codebase they were working with. The drop in the remaining points in week three represents some milestone breakthroughs and discovery's in the codebase that cut the required workload down considerably. The following three weeks the remaining points did not seem to have much movement as the team struggled with getting the simulation working. Around the fifth of February the team started making a noticeable difference as simulations were in full working order and codebases were running as expected. The continued decline from that point on represents a burn through work with both understanding the codebase and developing documentation and preparing deliverables for the end of the project.

Trello boards screenshots

Screen shoots can be found in the folder named "TrelloScreenShots" that accompanies this report.

Here is a direct link to the Trello board for ease of reading: https://trello.com/b/liM9WSWS/capstone-project-robot-soccer

The Screenshots

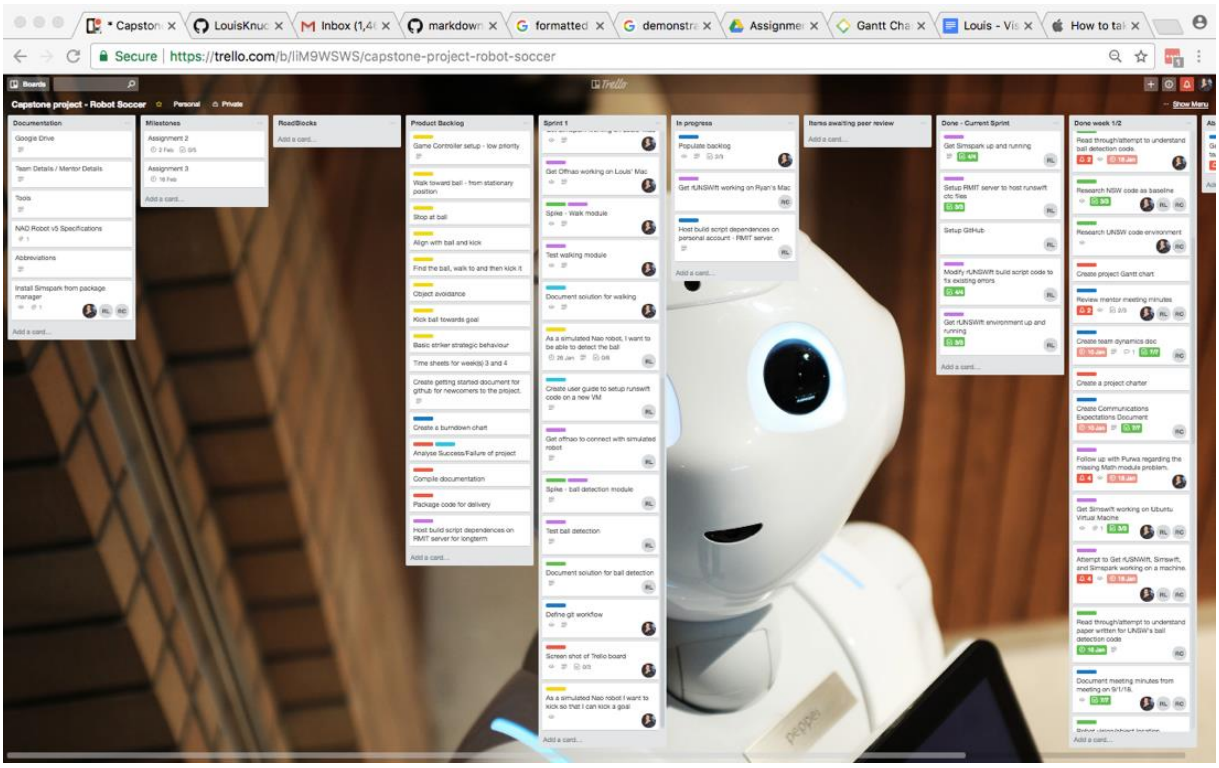


Figure 5 - Trello Screenshot for weeks one and two

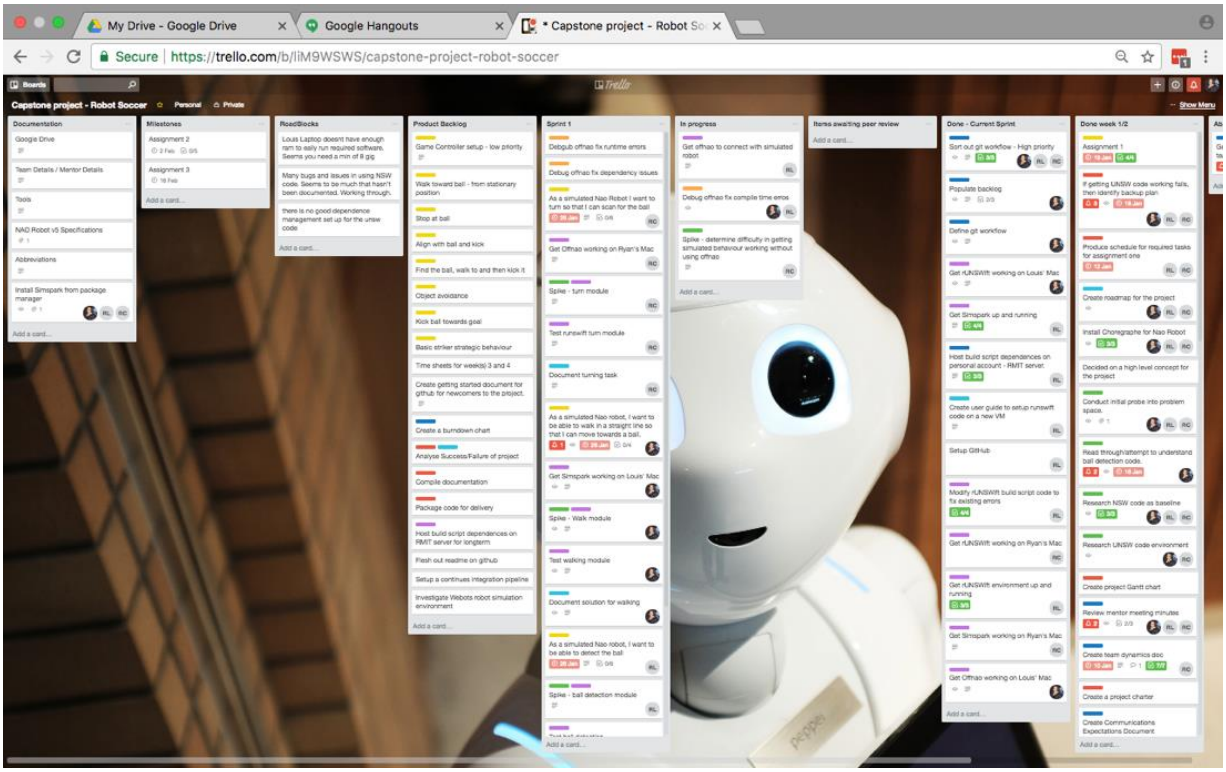


Figure 6 - Trello Screenshot for Sprint One, Week Three

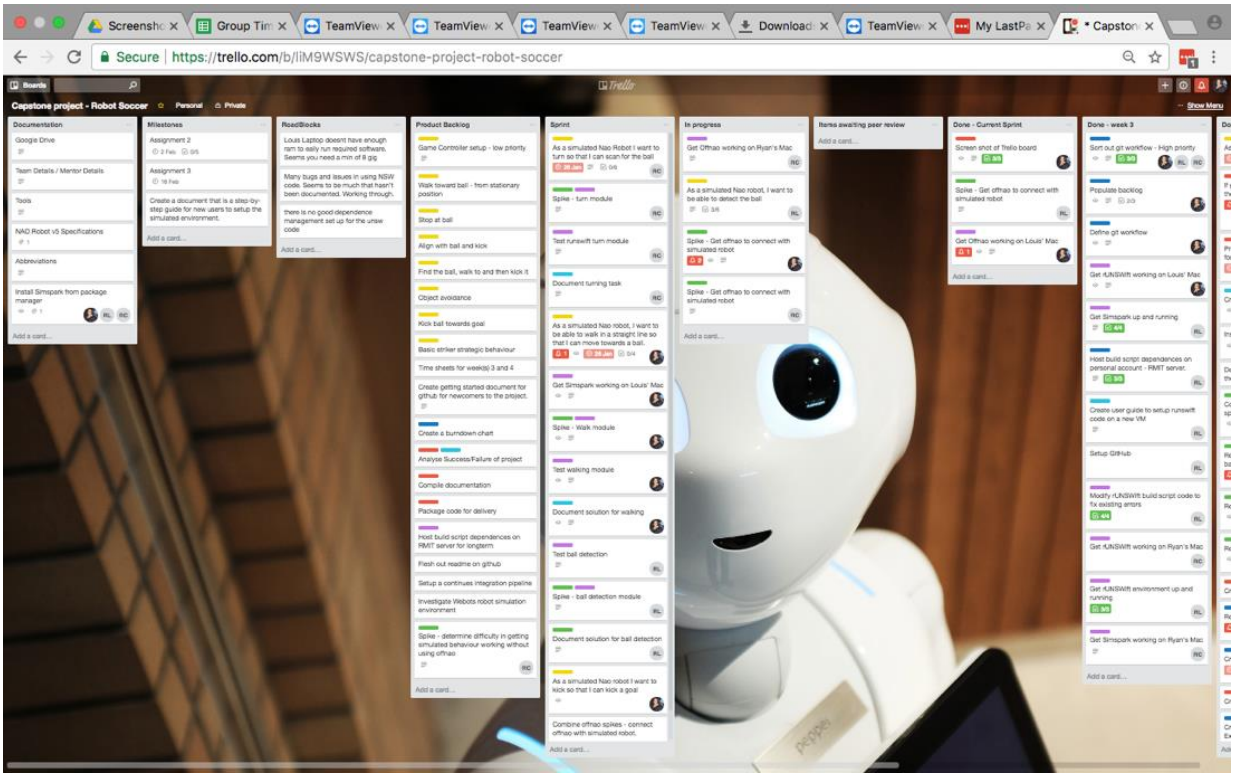


Figure 7 - Trello Screenshot for Sprint Two, Week Four - 1

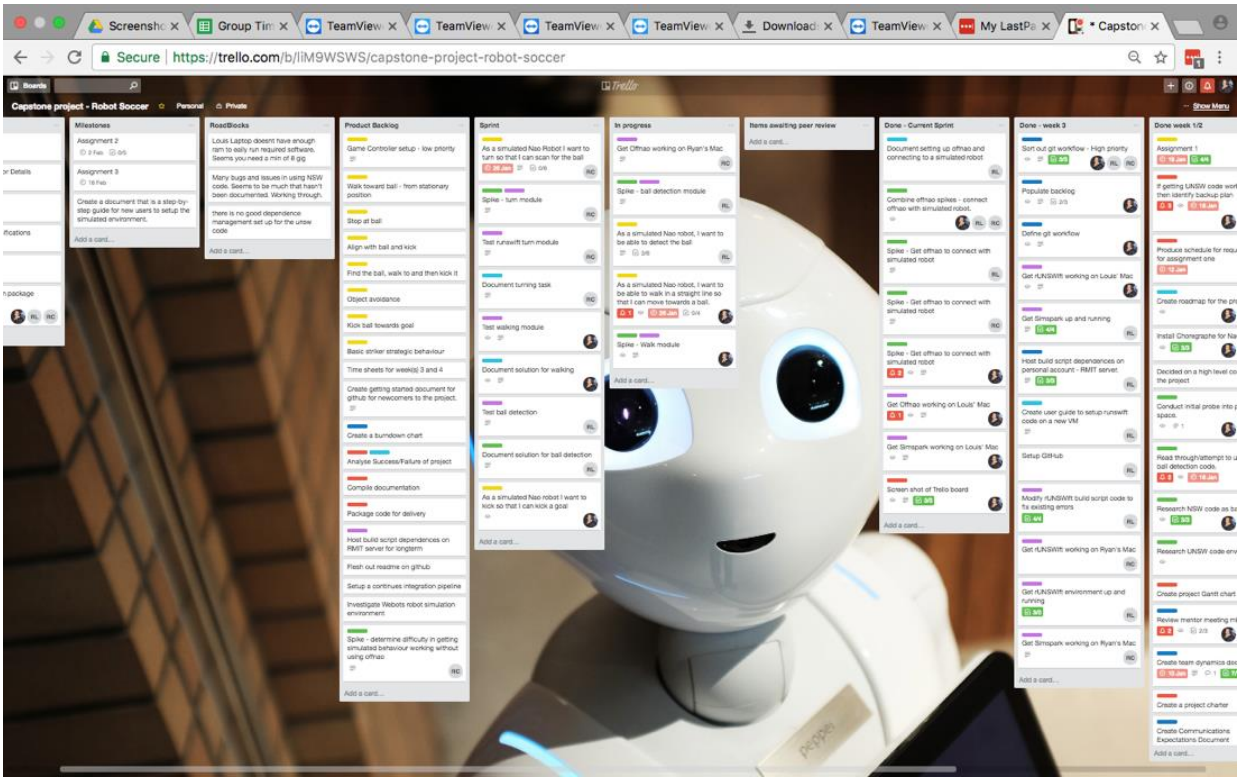


Figure 8 - Trello Screenshot for Sprint Two, Week Four - 2

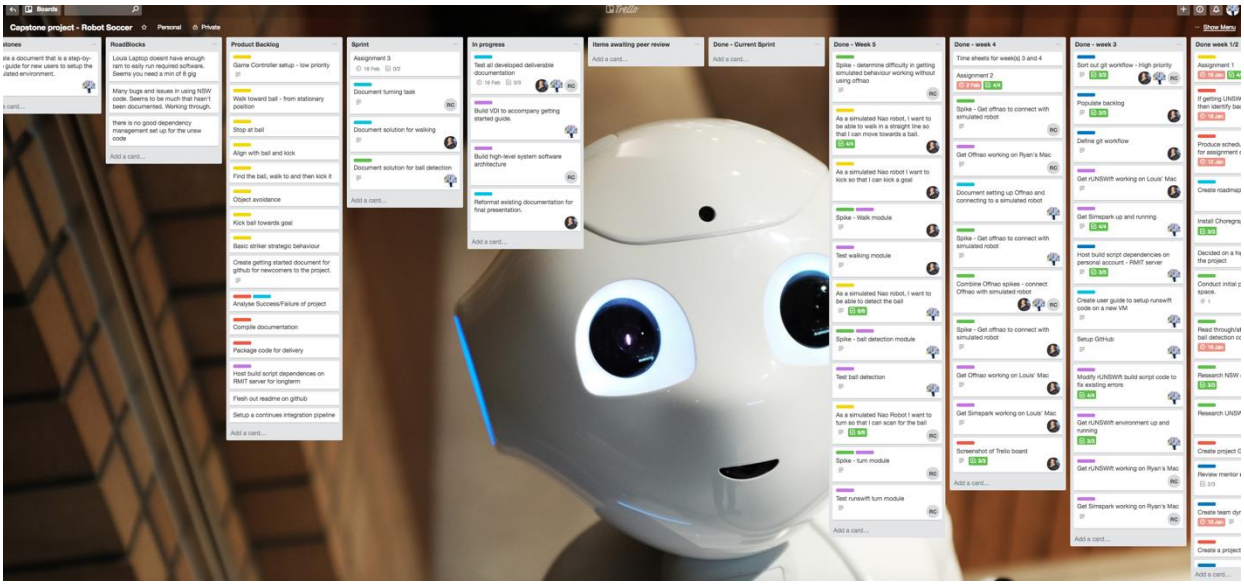


Figure 9 - Trello Screenshot for Sprint Three, Week Five - 1

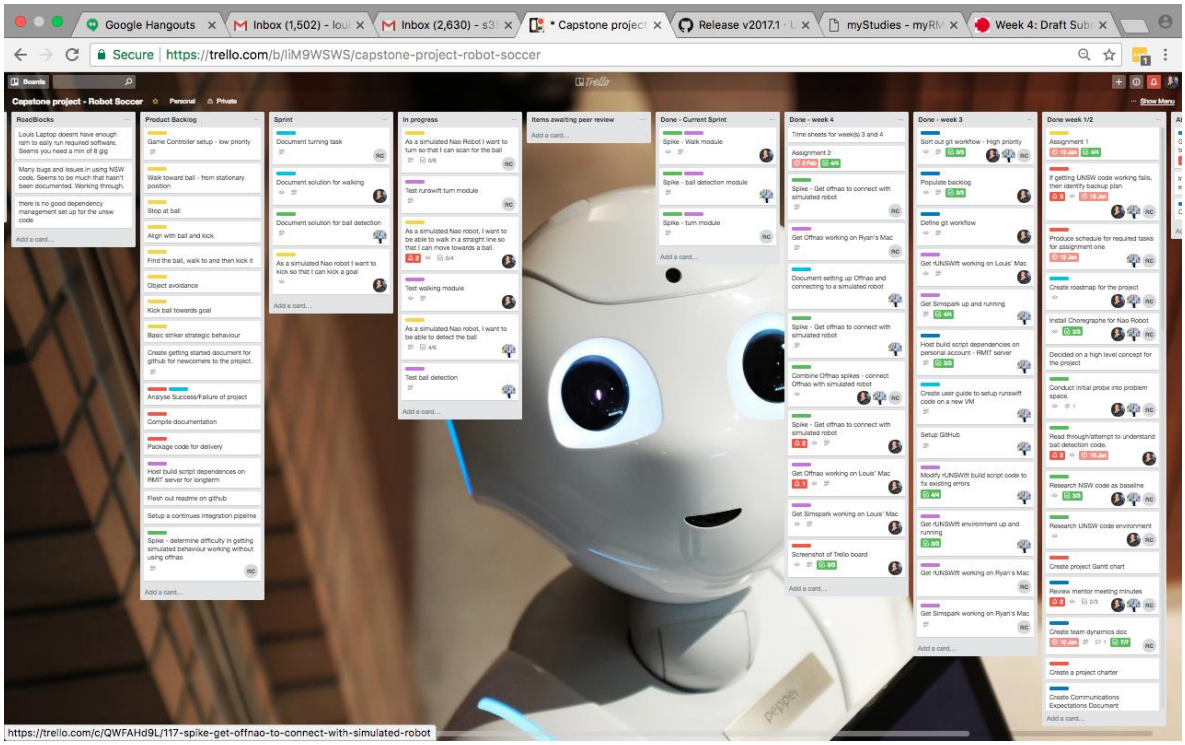


Figure 10 - Trello Screenshot for Sprint Three, Week Five - 2

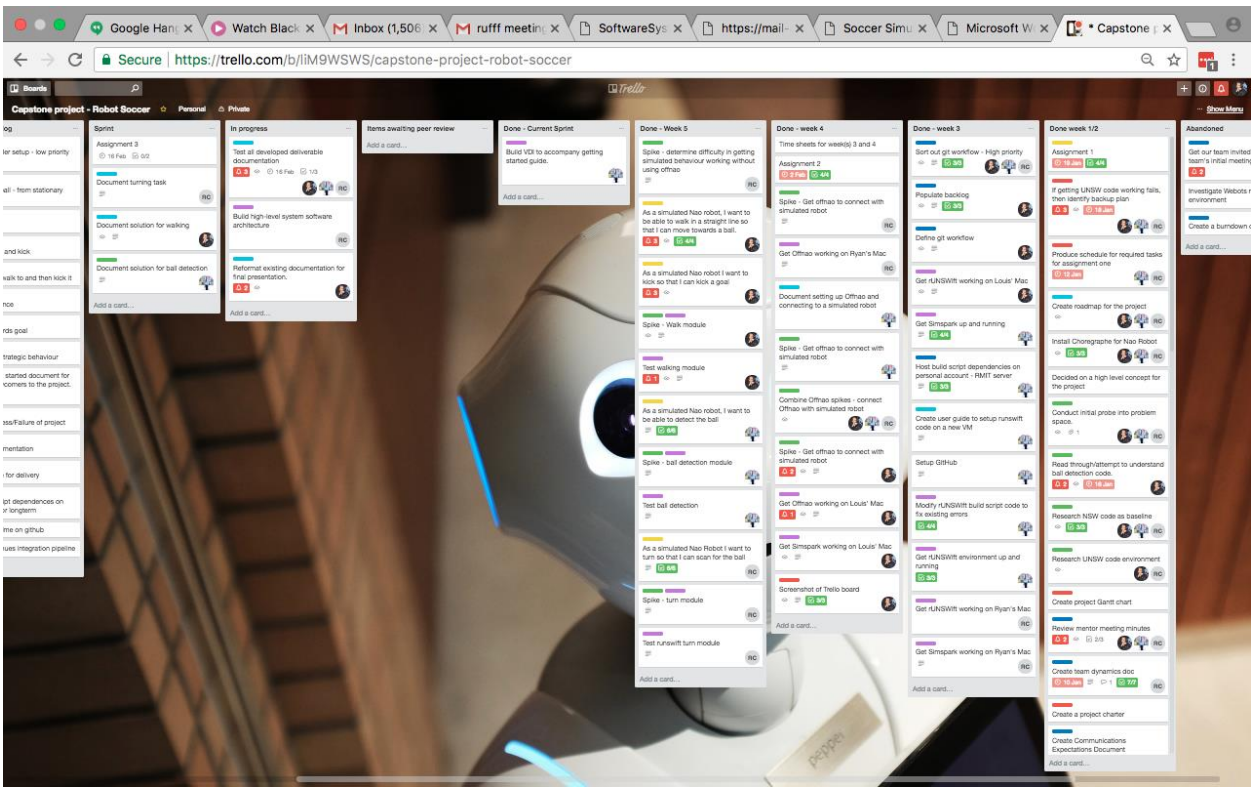


Figure 11 - Trello Screenshot for Project Wrap Up, Week 6 - 1

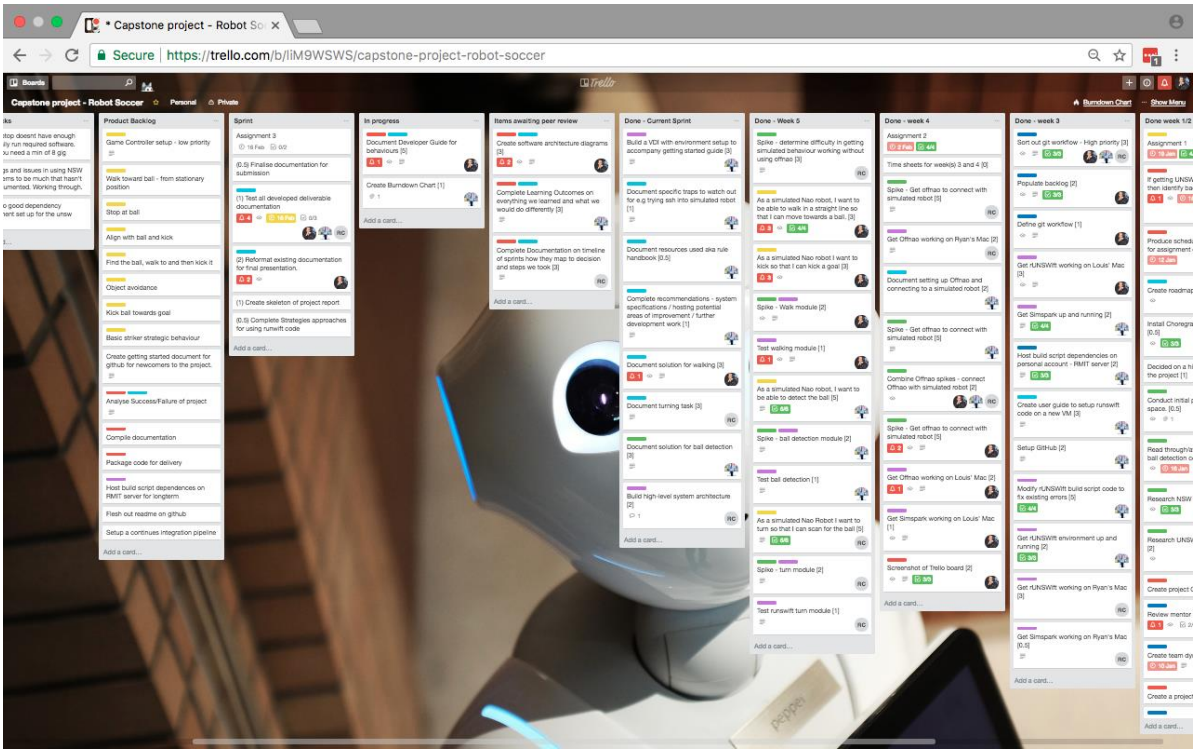


Figure 12 - Trello Screenshot for Project Wrap Up, Week 6 - 2

Issue and Risk Registers

The risk register created in the beginning of this project.

ID No#	Description	Proximity Date	Weighting	Mitigation Strategy
1	Unable to get the UNSW codebase working	16/1/18	12	Disregard the UNSW's code and begin extending the Nao's base code from scratch.
2	Resources that UNSW have not released that we need.	16/1/18	16	Source alternative places to get the required tools and dependencies
3	UNSW's codebase is spaghetti	22/1/18	9	Discuss with mentors the option to redefine project scope to refactor sections of the UNSW code base
4	Collective group knowledge base for C++ and python is small	26/1/18	12	Limit the mount of new development required and focus more on getting the codebase working
5	Will run out of time to complete all of the project deliverables	02/01/18	12	Discuss with mentors the barriers or roadblocks faced that slowed the project, and request a reduction of the required deliverables.

	Very Likely	5	5	10	15	20	25
	Likely	4	4	8	12	16	20
Likelyhood	May Occur	3	3	6	9	12	15
	Unlikely	2	2	4	6	8	10
	Rare	1	1	2	3	4	5
			1	2	3	4	5
			Negligable	Low	Moderate	Significant	Catastrophic
					Severity		

STOP	Stop works until fixed
UNACCEPTABLE	Urgent action to be taken to rectify issue
UNDESIRABLE	Action Required
Acceptable	Monitor
Desired	No Action

A list of issues can be found in the GitHub Issue Register.

System Design

Overview

Figure 13 - Software Interaction Overview Diagram is a visual representation of how the different programs interact.

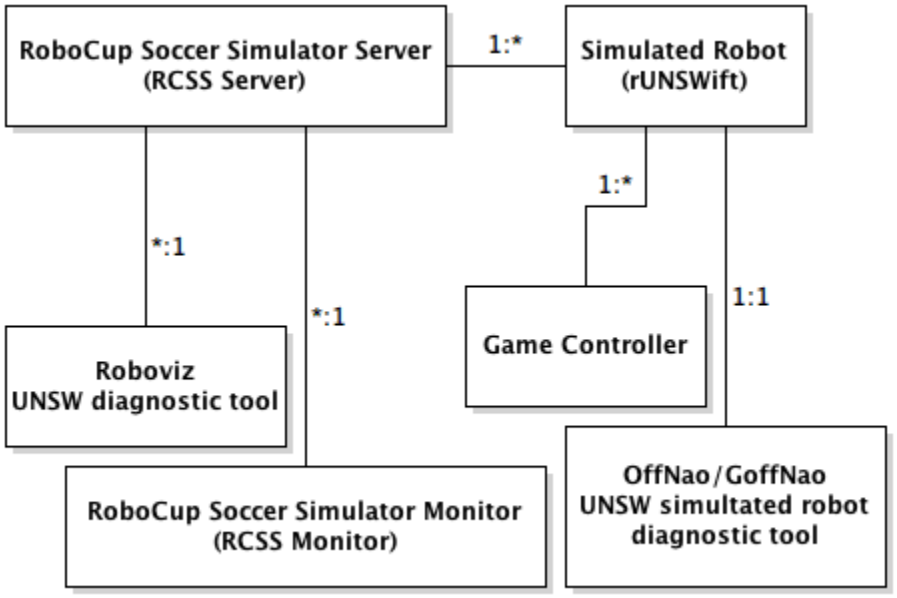


Figure 13 - Software Interaction Overview Diagram

Software Architecture Diagram

This is a high-level diagram illustrating the overall software architecture with a focus on the behaviour code. The architecture has been simplified for the purpose of making this diagram a more useful visual aid and as such does not perfectly represent the system. In particular the complexities of the threads and the adapters beside the PerceptionThread and the BehaviourAdapter have been omitted.

NOTE: For a more readable version of the image on the next page, please see the file "software_architecture.png" that accompanies this report.

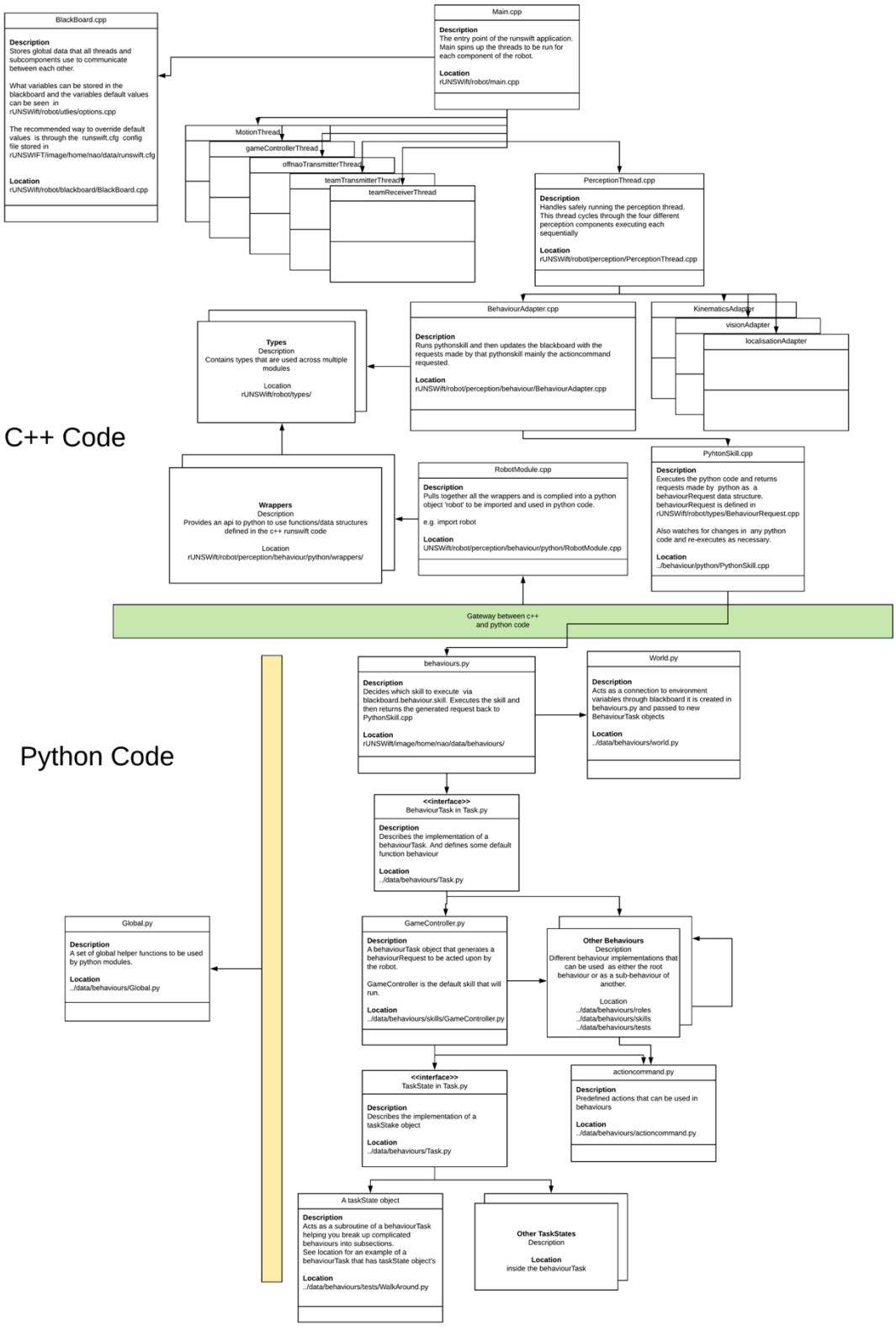


Figure 14 - Software Architecture Diagram

User Manuals/Project closure report (if needed)

The manuals and user guides developed as part of this project are in the folder "Deliverables" that accompanies this report. The file is called “up_and_running_with_runswift.pdf”.

The PDF contains:

- Minimum and recommended system specifications
- Installation guide for rUNSWift and the developer environment
- A getting started guide for Offnao
- Developer guide for developing robot behaviours
- Common traps to watch out for when developing

Learning Outcomes

Enabling Knowledge

Throughout the project we were able to employ a large list of skills learned during our degree at RMIT. A comprehensive list would be too long for this report, so only the most prominent items are listed here:

The skills that proved to be most useful throughout this project were:

- Knowledge of how agile methodologies work. In particular, Scrum was used as the primary development method, with week-long sprints and sprint planning and retrospective meetings.
- Knowledge of how to utilise git for version control, branching/merging played a prominent role in making development across a team manageable.
- Project management skills were a major asset. Planning the project and maintaining continued communication and team reviews, allowed us the confidence to persevere when faced with the many roadblocks. Pre-planning many of the smaller things also allowed the team to quickly and efficiently organise adequate documentation when required (such as meeting minutes and Scrum board tasks) allowing more time to get on with the major tasks.
- Knowledge of procedural and object oriented coding practices were significant advantages with this project. Even though no one in the team were particularly experienced in C++ or Python, all team members knew enough about the concepts and practices of procedural and object oriented languages that they were able to read through the code and understand it.
- Knowledge of the benefits of extreme programming, specifically pair programming was put to use to jointly tackle the major issues maximising the team's ability to achieve the projects goals.

Identification of New Knowledge and Skills

This project has given all members of the team the opportunity to work in an industry-style environment incorporating common business practices and process for developing code. It has provided the team with experience in working with an unknown, complex codebase.

Heavy emphasis was put on learning to use dependencies and linking to them in make files as well as learning about different types of Paths within the code (such as R-Paths). This has given the team greater insight into the workings C++ and the power of CMake and the available tools that reduce build

and run times. C++ is also a large area of knowledge improvement as none of the team has worked with C++ before.

Project Management

The team had massively overestimated what was possible in six weeks with only three weeks of dev time. This project proved as an invaluable lesson in better estimating time frames. And served as a good example of the types of hidden issue's one can run into when dealing with a pre-existing code base.

Critical Analysis

Project methodology

Agile scrum worked well for this project in helping us rapidly rescope the project in light of setbacks. Allowing the project to deliver the most value possible to the product owners with little to no scope creep. The scrum methodology was lacking when it came to team work we found and as such we added extreme programming aspects to our management style to help fill in these gaps. Overall the sprint retrospectives, sprint planning meetings and consistent communication through slack and Trello where highly successful.

Code quality

The design of rUNSWift is actually quite decent however much of the code lacks meaningful comments and the code structure and naming conventions can be improved. This increased the difficulty in analysing and understand the code base and as such should ideally be improved on.

Meeting the requirements

rUNSWift meets the project requirements well, being built specifically for our purposes. Our core requirements of delivering a starting point for future RMIT teams have been met and we have effectively utilised seven years of prior development for RMIT to catch-up with other schools.

Problem Solving

Initially, we encountered compiler issues in UNSW's rUNSWift code. For example, a math module was missing in the python script and the build process terminated abnormally. The solution was to search online resources for similar issues and figure out where that module was. This task was particularly perplexing because the Math module is typically built into python, or when it is imported, it gets imported from the system's Python path. In this case, a custom python module had been created and the build's Paths pointed to the wrong directory. This case was time consuming for two reasons: first because of the abnormal nature of the issue there were limited online resources representing similar issues, and second because the team was unfamiliar with the code.

Another problem is unfamiliarity with existing utilities such as Offnao, which is a debug tool to test robots' vision, behaviours, ball positions and colour calibration. The team was not able to connect to the simulated robot using Offnao in the beginning due to dependencies that were incompatible with the chosen OS .

Another significant roadblock was overcoming issues in the software architecture involving activating behaviour modules in a simulated robot. This issue was mainly cause by a pathing issue when trying to execute the python code from the C++ code. After several days of work it was found the fix was to rename a folder from test/ to tests/.

Communication

We set up Google Hangouts as a virtual tool and Slack as a live messaging tool for group communication. We arrange schedules for mentor meeting each week on Tuesday 11:00am. Generally team face to face group meetings were held twice each week for sprint planning and reviewing. All team members responded to messages quickly on Slack and attended every group meeting. The advantage of the face to face group were evident as these tended to allow quicker expression of ideas and problems through visual aids.

The roles were as follows:

Louis Stekhoven-Smith - Scrum Master / Team Leader / Acting Product Owner
Ran Lu - Documentation Control / Team Member
Ryan Couper - Meetings Organiser / Team Member

Since we are responsible for different tasks and levels of expertise varies, the team member who completed his task would explain the process to other team members during face to face meetings or virtual meetings. If there were issues encountered during the process, the person who is responsible would raise an issue on Github and all team members were able to review it.

Responsibilities

The rUNSWift codebase is redistributable and modifiable under the terms of the Gnu Not Unix (GNU) General Public License (GPL) as published by the Free Software Foundation (FSF); either version 2 of the License, or (at your option) any later version as modified below.

Further to the GNU license, as the original developers and to ensure a significant contribution to the Artificial Intelligence (AI) community (which is the main focus of RoboCup), UNSW has added some clauses of their own. For full details of their added clauses, see their LICENSE file here.

Appendix A: Bibliography

Collette J 2017, *Using 3D Simulation to Develop Robot Code*, Github, viewed 19 January 2018, <https://github.com/UNSWComputing/rUNSWift-2017-release/blob/master/docs/Collette-Using_3D_Simulation_to_Develop_Robot_Code/Collette-Using_3D_Simulation_to_Develop_Robot_Code.pdf>.

Couper, R, Lu, R & Stekhoven-Smith, L, *MIT-RoboCup-Standard-League/PP1-Nao-Soccer Issues, GitHub, viewed 14 February 2018, <*https://github.com/RMIT-RoboCup-Standard-League/PP1-Nao-Soccer/issues?q=is%3Aissue+is%3Aclosed>.

Lu, R 2018, *RMIT-RoboCup-Standard-League/PP1-Nao-Soccer Troubleshooting,* GitHub*, viewed 15 February 2018, <* https://github.com/RMIT-RoboCup-Standard-League/PP1-Nao-Soccer/wiki/Troubleshooting>.

N.d. 2016, RoboCup Objectives, RoboCup, viewed 16 February 2018, <http://www.robocup.org/objective/>.

N.d. 2017, *Simspark Wiki Main Page*, blog, viewed 19 January 2018, <http://simspark.sourceforge.net/wiki/index.php/Main_Page>.

RMIT University 2017, *STAFF PROFILE Professor John Thangarajah*, RMIT University, viewed 19 January 2018, <https://www.rmit.edu.au/contact/staff-contacts/academic-staff/t/thangarajah-professor-john>.

RoboCup Technical Committee 2016, *RoboCup Standard Platform League (NAO) Rule Book*, blog post, viewed 18 January 2018, <http://www.robocup.org/system/sub_leagues/rules_papers/000/000/005/original/Rules2016.pdf?1473627747>.

Schmidt, P 2017, *UNSWComputing/rUNSWift-2017-release/LICENSE*, Github, viewed 19 January 2018, <https://github.com/UNSWComputing/rUNSWift-2017-release/blob/master/LICENSE>.

Schmidt, P 2017, *UNSWComputing/rUNSWift-2017-release Wiki*, Github, viewed 19 January 2018, <https://github.com/UNSWComputing/rUNSWift-2017-release/wiki>.