SOFTWARE DEVELOPMENT PROJECT

TEAM A

GROUP 2

Process Report

Team:
Stefan IVANOV
Oana RADU
Miro TRIFONOV
Asta BOGDELYTE
Jianmeng YU
Justin ALISAUSKAS
Rosen CHAKAROV

Supervisor:
Angus PEARSON

THE UNIVERSITY *of* EDINBURGH

February 14, 2017

1 Scope and purpose

The following process report is made to detail our interactions within the group and team based on a comprehensive strategy of project development in order to achieve our target of improving both the hardware and software of the Fred robot.

2 Communication

Our first project management decision was regarding the manner of communicating among all the individuals involved in the project. As we wanted from the beginning to adhere to the agile methodology, this step was paramount for our further interactions.

2.1 Group communication

We decided to use Slack as our main communication option. We set up our own Slack group, as we needed the admin privileges to set up GitHub plugin for tracking of commits. We also added separate channels for hardware, software, report and work scheduling.

An important aspect of our communication were the regular meetings we held. Each week we have an informal meeting with all the group members and a formal meeting with our mentor to discuss progress, identify the key areas which we have problems with and set the agenda for moving forward. In addition to this, we have shorter (10-15 min) technical meetings every day, during which we give updates on what each of us changed since the last time and set our daily goals. Also, we would share this information on Slack to keep everyone updated at all times and we would consult each other on any changes that we have made to the code or the robot itself.

2.2 Team communication

An additional Slack group was created for discussions between Group 1 and Group 2, as we thought that it will be beneficial to keep each other updated on the progress that our groups make. However, we have not yet worked together apart from giving each other general advice, as we decided to wait for the team phase to begin so that we can solely concentrate on individual performance until then.

3 Task allocation

3.1 Initial stage

From the beginning we divided the group members into hardware and software teams- each subteam participated in the according workshops(Justin, Jianmeng and Rosen at the hardware workshop and Stefan, Asta, Oana at the software workshop). From the information gathered from these workshops, we knew that our main first task needed to be sorting the robot design itself. As we did not understand the code well enough, we split everyone into software and hardware teams, without allocating the software team to smaller subteams (Vision, Strategy and Communication) as we deemed it to be a lot more sensible to split the tasks when people are familiar with the code and can state which part they would prefer to work on. For the first couple of days the software team

(Stefan, Miro, Asta, Oana) had a task of analysing several Git repositories of robots code from previous year, so that we could decide which code will be used as a base for our own implementation. The hardware team (Jianmeng and Justin) had a task of building a basic robot for an initial testing of the system.

3.2 Further development

At the start of the second week our software team had already familiarised themselves with the code and the hardware team managed to develop several prototypes; one of the prototypes was a copy of Fred which was used as a testbed for arduino code and another prototype was a first iteration of design for a robot with three holonomic wheels.
robots photoooooos with text
Afterwards, we decided to make improvements to our group structure. The software team was split into subteams: Vision - Asta, Strategy - Stefan, Miro, Rosen, Oana, Communication - Miro. The hardware team remained the same - Jianmeng and Justin, with Jianmeng becoming the main person responsible for hardware and Justin taking the group manager role. We do most of tasks allocations during the daily meetings, as then we know the amount of time that each person can spend on SDP during that day. Also, even though we got assigned specific roles, we were still putting a lot of effort to keep up to date on each others progress, mainly because this allowed us to allocate more people to help a subteam which gets a more demanding task.

4 Progress tracking

4.1 Trello

As we needed to be sure that progress was achieved in a timely manner regardless of any milestones encountered, we decided to use a professional tool to help us manage progress tracking. Thus, we chose Trello as once we had to precisely split tasks, Trello seemed to provide a convenient and easy way of doing that. We set up a board where we had a list with the tasks we gave ourselves for that particular week. This decision buttresses our perspective of handling an agile process with one week long iterations.

Every card created on Trello underlines the due date each task is supposed to be finished by. We tried to give ourselves realistic deadlines, even though sometimes they needed readjusting. Trello also has integration for automatically creating Gantt charts and we utilized this to better know how long each task is taking us to complete.This feature is essential as it showed us which areas needed more of our attention.

Another way Trello proved useful is how it enables the creation of to-do lists for each task. Thus, splitting tasks further into smaller subtasks eased the entire development process as we always knew the next mini-goal we had and that managed to keep us on track. In addition to this, we used task comments section to note any problems we were encountering.

GRAPH HERE
GRAPH HERE

Also, we have allocated team members to each task on Trello in order to easily track who is working on what. Multiple people working on the code at the same time necessitated the use of a version control system too. Therefore, we had to integrate Git into the project. As some members of the teams were not used to working with it, we even had a quick tutorial about it that was held by Justin.

4.2 Git

4.2.1 Using forks

At first every one of us forked the main repository and made changes to code in our own forks, as updating the main repository by making pull requests was the most secure approach.

4.2.2 Using separate branches

After using the forks approach for some time, we noticed that this approach is better suited for tasks that are less interconnected and it better suits less experienced Git users. We solved this issue by making a switch to simpler system, using only one repository and multiple branches on it. We set up a branch for every contributor and kept the master branch up.

5 Plans and Milestones

5.1 Milestones

The main milestones that we encountered and will encounter in the future, are underlined in the following list:

1. January 22nd - Software basis sorted, robot moves
2. January 24th - 3 holonomic wheel robot assembled, basic movement works
3. January 28th - Robot is capable of grabbing and kicking
4. February 1st - Friendly match 1
5. February 15th - Friendly match 2
6. February 17th - Process report submission
7. March 1st - Friendly match 3
8. March 22nd - Friendly match 4
9. March 29th - Rehearsing of presentations
10. April 5th - User guide submission
11. April 7th - Final match day
12. April 19th - Technical report submission

However, even though we strictly followed our schedule, there were several extra road-blocks that we had to deal with:

- Adding sensor for ball detection in the grabber is a more difficult task than expected. The sonar which was given by electronic technician performed not as expected - it could detect only the objects which are further away from it, and thus is not suitable for our application. The laser sensor would be a better option though, but due to our technician being ill and delivery times taking more than a week, we cannot expect the sensor to arrive before second match. Thus we are trying colour sensor as a temporary solution.
- Failure of making the robot face the ball - we found out that the delay of data transfer, processing and sending Arduino commands is one of the main reasons why we struggle to make robot face the ball and face the gate. To solve this problem, we allocated more time for code optimisation.

5.2 Work plan

Our work plan is split into weeks, because of our adherence to the Agile management process. Each major task is split into subtasks, which then are assigned to one or more members of the team, depending on difficulty and the qualification of the people working on it. *Achieved - until the second friendly match* During the first week our team was working to get first two goals done. We focused on setting up a basis on which both hardware and software groups could work further.

- The first goal for a hardware team was to create a four holonomic wheels robot which could be used for testing the code belonging to teams from previous years. In order to achieve this task, the team had to be:
    1. Going to the hardware workshop to get introduction on structurally rigid robot building (20 January)
    2. Building a copy of Fred (a 2016 Holonomic Drive Example System) (20-21 January)
    3. Setting up RF stick for connection between Arduino in the bot and DICE machine. (20-21 January)
    4. Loading Arduino C++ code to the robot, which enables the controls of motors and propeller. (21 January)
- The software team was working on basic implementation of vision, strategy and communication which entailed the following subtasks:
    1. Going to the software workshop to get introduction on all vital parts of the software system. (20 January)
    2. Researching on both holonomic drive example system (Java) and 2 wheel drive example system (Python). (20-21 January)
    3. Selecting the preferable system (holonomic drive example) based on overall performance, quality of vision system and quite easy to understand strategy.
    4. Implementing and testing the Fred system on our first iteration of Fred robot. (22-24 January)

As we finished these two goals during the first half of the week - earlier than we expected, we started thinking more about our performance at the first match. We identified the main area which required major improvement - the propeller, which was a really bad implementation of a kicker, and set new tasks for software and hardware teams.

- Hardware team had to come up with robot design which could accommodate a kicker. It meant that the whole robot had to go through major changes and we went through several iterations of design.

    1. Trying to implement kicker into a robot which uses four wheels (23 January)
    2. Implementing a three holonomic wheels system due to lack of space for kicker and questionable rigidity of the design shown above. (Software development depended greatly on decisions made by hardware the team, thus in one day a basic three wheeled robot without kicker was built for movement strategy testing.) (23 January)
    3. Designing rigid and space saving robot implementation with a kicker. (23-24 January)

- Software team had to adjust the project code to work with the changes made by the hardware team.

    1. Write and test code made for the kicker on the test-robot which has basic kicker and four wheels. (23-25 January)
    2. Edit and test movement code to work with the three wheeled robot design. (24-27 January)
    3. Implement the changes to the code which would allow the three wheeled robot go towards the ball and kick it with new kicker. (27 January)

The second part of our plan was done on 27th of January, at around the middle of the second week. There was still room for improvement for our first match,thus we established a plan for the time period until the first match:

- Hardware team had to:

    1. Add a grabber, so that the robot would be able to catch the ball and turn towards the gate. (28 January)
    2. Make the kicker more precise, as the rules require our robot to kick straight. (28 January)
    3. Add a rigid holders for batteries and boards on the robot. - 28 January Fix the green plate on top of the robot without blocking the access to arduino and other boards connectors. (28 January)

- Software team had these tasks:

    1. Add support for the grabbing action. (28-29 January)
    2. Find and implement the way for the robot to detect that the ball is in the grabber. (28-29 January)
    3. Make the robot face the goal when it catches the ball and then kick it. (29-31 January)

After winning the match during first friendly tournament we decided to concentrate our work on fixing the flaws which we noticed during the fights. The plan spanning from February 2nd all the way to 15th of February covered:

- Hardware team:
  1. Prevent robot from flipping over when it drives with one wheel onto the incline near the side of the wall.
  2. Add sensor which could detect the ball in the grabber.
  3. Switch the wiring between motors and arduino, so that the software team could get readings on rotation of motors.
- Software team:
  1. Increase the accuracy of the robot when it is trying to face the goal.
  2. Fix the case when the robot is driving with ball in grabber.
  3. Fix the vision system, which loses robot a lot more often in Room 1, mainly because of higher amount of glare and more uneven lighting conditions.

Desired - after the second friendly match As we are following the Agile methodology, our work plan becomes more comprehensive on a weekly basis. However, we know the key points that have to exist in our work plan for the following weeks:

- Fixing the problems which we were unable to solve before the second match
- Starting a lot more thorough conversation with Group 1
  1. Organising a team meeting, with both mentors involved.
  2. Discussing the options on joining strategy and/or vision.
  3. Deciding on roles of the robots (Defender/Attacker).
- We also need to do such tasks as:
  1. Finalising the process report
  2. Schedule the work for the sprint week

6 Risk Assessment and Contingency Planning

Inadvertently, we encountered several unexpected issues in the development process. As we were aware of these occurrences, we decided that a mandatory part of each meeting we have is discussing current issues and approaches to handle them. Some of these problems would often recur and, as a good practice, we decided to always have prepared solutions for them. Thus, we made the following risk assessment table:

Problem	Solution
Room conditions change constantly, vision breaks	Remember to save previous vision calibrations for different conditions, create new ones
Rooms are sometimes full, cannot test	Reevaluate due dates, do not leave things for last moment before matches
Code does not run when switching branches	Set up a guide on how to run it properly i.e. add the required libraries etc.
Certain task takes longer than expected	Reevaluate due dates, allocate more people to it, look at new approaches
A new feature breaks the code, makes the robot worse overall	Pull last working version from GitHub
Team members are unavailable	Reevaluate due dates, contact mentor
Robot batteries are dead	Make sure a pair is always charged, ask other teams if required
Robot keeps breaking in action	Reinforce robot structure
Cannot understand a team members' code	Everyone should properly add comments to their own pieces of code
Members with keys from lockers are absent	Leave a key at Forest Hill

7 Conclusion

Our group had and will continue to have a well organized structural organisation that can be noticed by our easy and continuous communication between group members, a flexible and comprehensive work plan, a profound awareness for the existence of manifold milestones, and an astute preparation when exposed to risks.