Team 06 - DPM Fall 2017 Final Report

Alex Hale (260672475), Xianyi Zhan (260672960), Justin Tremblay (260743650), Joshua Inscoe (260659992), Xu Hai (260661832), Frederic Cyr (260674545)

ECSE 211 – Design Principles and Methods

December 1, 2017

Introduction

The goal of the ECSE 211 Design Project is to expose students to a simulation of the real-world Engineering design process. As explained in the course syllabus, the Engineering design process is a "systematic procedure that begins with the formulation of a precise specification of the project at hand and ends with the specification of a procedure and/or mechanism that meets the requirements outlined in the project specification". In this course, the Engineering design process is approached from a systems perspective, where a problem is abstracted as a system comprised of inputs, outputs, and a process model that implements the desired input-output behaviour. The Engineering process consists of the following components:

- **Design Specifications**: the client submits the requirements in the form of a PDF document.
- **Research and Development**: the team drafts potential solutions to the problem through the weekly laboratories.
- **Optimization**: incremental improvements are made to at least one of the drafted solutions
- **Implementation**: one final design is chosen. The hardware is constructed and the software is written in full.
- **Testing and Refinement**: tests are conducted on the design to verify that the specifications are met. If not, refinements are made. This is a repeating process conducted throughout the project on all individual and combined components.
- **Project Management**: one team member is designated to create the team schedule, ensure the team is on a path to success, and control the team's budget and GANTT chart.

At the end of the project, each team member should have a better understanding of what it is like to work in the Engineering profession, as well as an understanding of why a variety of academic backgrounds are needed in a successful team.

Team Organization

Once the team was formed, roles were assigned based on experience and interest. It was determined that software development would require more working hours than hardware development. However, only three of the team members expressed confidence in working on the project software, and two of those members were more interested in the documentation and project

management roles. Once each member had a role, tasks were allocated from the GANTT chart to the appropriate team member.

The initial GANTT chart was based heavily on the template GANTT chart provided by the Professors. Each of the tasks on the template was broken down into smaller tasks of a maximum length of three days each. Finally, each of the tasks was assigned to one of the team members, creating the resources chart.

Having never completed a similar project, the initial task breakdown was performed using experience from the weekly laboratories, advice from TAs, Professors, and friends who had taken the course previously, and a healthy dose of wild guesses.

When creating the first version of the GANTT chart, an attempt was made to balance the total hours worked for each team member. In addition, the goal was to ensure that each team member would not have too much or too little work at any one time.

Issues Encountered

Since fewer team members were assigned to Software than was eventually required, there were some Software tasks on the GANTT chart where the dependencies broke down. For example, from November 2 – November 5, there was a group of software tasks assigned to be completed concurrently, but only the Software Manager was available to work on those tasks, necessitating they be completed subsequently. This caused cascading delays down the remaining tasks in the GANTT chart, which caused the searching algorithm to be created at the last minute at the end of the project.

The critical path on the GANTT chart can be viewed by clicking the “Show Critical Path” button.

In addition to the GANTT chart structure, there were other issues that impacted the project. The most significant issue was teamwork and communication related to software development. The software development was completed nearly entirely by one person, which meant the rest of the team hardly even looked at the code. At the end of the project, five of the six team members were not able to help complete the searching algorithm, no matter how much they wanted to. The next issue arose because most of the team’s communication occurred online. Online communication is fantastic for efficiency, but requires extra effort toward clear communication – this effort was not put forth by all team members. The lack of clear online communication meant that the team members weren’t the best of friends, which made people even less likely to call in-person meetings, so even more communication was done online, and the cycle continued. Finally, the GANTT chart wasn’t followed as closely as it should have been, meaning team members were not entirely clear about which task they should be working on. This created confusion and slowed project progress.

Somewhat to the team’s surprise, the project ran relatively close to the original plan! Since the GANTT chart task structure followed the logical sequence of events that would be undertaken in the absence of a GANTT chart, the team didn’t pay the price for failing to follow the GANTT chart exactly. The weekly client meetings and documentation preparation served as a checkup to verify the progression of the project, so everything stayed on-track.

Budget

The team was under-budget for the entirety of the project, so the budget placed minimal constraints on the team. The influence of the budget was to force team members to think about what they were working on and how many hours they were spending, since it would be documented.

In the opening weeks of the project, resources were allocated to all the project tasks, allowing for an estimation of the overall budget. This meant that the budget expenditure could be monitored as the project progressed, and resources could be allocated to or removed from different areas as required. This step was also crucial in assuring that the team would not unknowingly run out of budget at some point during the project.

Since the team finished the project under budget, a larger budget of hours would not have made much of a difference to the team. However, if no budget restrictions had been placed at all, budget usage would likely have been lower, for two reasons. Firstly, when a team member works fewer hours than their budget quota (which was usually the case during this project), they are likely to slightly inflate their hours to be closer to the mark that is "expected" of them. Secondly, with no budget restrictions, a team member is more likely to quickly finish their work as fast as possible since the measurement of their performance is solely the finished product, rather than a document of their hours worked. For this team, the budget served as a guideline of how much time to spend on a week-to-week basis, rather than an overall budget of time to use for the entire project.

The team was weak in resources when it came to developing software, and, by extension, performing software testing. One available software developer was already spending their full budget most weeks on documentation. Another available software engineer worked nearly exclusively on the GANTT chart and budget for the first four weeks of the project, then spent a lot of time on software right at the end. This was not ideal, because it didn't give the testing team the chance to conduct any software tests on that last-minute code. Fewer budgetary constraints would have allowed the documentation manager and project manager to contribute more consistently on software. However, this would have created a large imbalance in terms of hours worked between the three software-developing team members and the other three team members. The resource that the team really needed was additional software developers, especially in the first four weeks of the project.

The Design Process

The Engineering design process was instrumental in our team achieving the goals of the project. The process gave team members a structure to follow, rather than working on whatever they felt was necessary at the time.

The most difficult part of the process to follow was having team members consult the tasks outlined in the GANTT chart. The GANTT chart was often updated to reflect completed and changed tasks, so members would need to consult it frequently to keep up to date. However, the GANTT Project software does not allow for easy collaboration, so this was unlikely to happen. Instead, the Project Manager should have posted daily updates in the Slack workspace about what each team member should be accomplishing that day. Another difficulty in following the process arose because most team members worked a variable amount of time from day to day. A team member that puts in lots of time

for a few days and not much the next few days will naturally cause conflicts with the schedule in the GANTT chart, since tasks have a specified period in which they need to be completed.

Approximately 21% of the overall budget used was put toward testing. Most of this time was put toward testing individual components in the early weeks of the project. The testing in the first four and a half weeks of the project was sufficient. There were plenty of tests conducted, and the results from each test were given back to the software and hardware teams to improve the robot. However, the testing in the last week and a half of the project was not adequate. This problem arose because the components that required testing were not completed early enough. The searching algorithm needed to be tested individually, but it was not complete until the day before the competition. Full-game testing needed to occur, but since the searching algorithm was completed so late, there was only time to run one short test. While waiting for the searching algorithm, the testing team should have run full-game simulations while ignoring the searching phase entirely. This test would consist of playing the game normally, but navigating to the searching zone and navigating away to the next step without searching at all. However, due to a lack of communication between the software team and the testing team about the current state of the robot and readiness for testing, this test did not happen.

The preliminary tests about the individual components of the robot produced sufficient useful information for the use of the hardware and software teams. However, the tests about the final competition performance of the robot were not sufficient. The Professors warned all semester that integration testing was the most important phase of the project. The team had this in mind, but since the searching algorithm was completed so late and the necessary adjustments to the test weren't made due to the lack of communication mentioned above, the required amount of integration testing was not completed.

It was estimated that a total of 10 days of resources would be required for full integration testing. Based on the amount of time that the earlier phases of testing took to complete, and the amount of time it took to complete the small amount of integration testing that the team completed, it would likely have been enough time.

A few strategies could be used to make the design process more effective. Firstly, the test report-writing responsibilities could be better distributed. The tester could take notes, and somebody else could write the report. This is especially applicable for those who struggle with writing – the task could be distributed to someone who is a better writer. Additionally, only one tester is required to perform each test. Having multiple testers perform a test at the same time is a waste of resources. There are some exceptions to this: for example, when the testing engineer does not know how to program the testing script, so a software developer must be there. However, even in this case, resources could be saved by having the software developer perform the test instead.

The beta demo gave the team a deadline to meet that was well in advance of the final demo, which was a great way to keep the team on track. The beta demo also caused some panic within the team: the robot completely failed the demo because of the two random coordinates that were supplied for the end of the zip line in the red zone. The beta demo served as a swift kick to get the project into gear. Once the in-house beta demo was completed successfully, calm was restored.

Design Success

Compared to the performance of the other teams during the competition, the robot performed quite well. The searching algorithm was too slow to be effective, so it was skipped – only 20 seconds was allotted for searching, and most of that time was spent navigating to the start of the searching zone from the end of the zip line or bridge. Another issue was the number of bugs present at the start of competition, which should have been ironed out beforehand. The two most significant bugs were a typo in the code (which ran the robot into a wall in the first demo) and an infinite localization loop (which plagued the robot after the zipline dismount in the second and third demo). Finally, the navigation accuracy of the robot was a shade too low. In the final run, the robot got stuck when mounting the zip line because it was slightly off its intended path. In the only run where the robot dismounted the zipline and was heading to the home corner, the navigation accuracy caused the robot to finish one tile off.

The performance of the robot entirely exceeded the team's expectations. Only one team member had touched the code or tested the robot as they worked on the searching algorithm in the final few days leading up to the competition, so only they knew how it was going to perform. That team member informed the rest of the team that they had "very low expectations" without giving any details, so the rest of the team was very skeptical about the robot's performance. However, when combining all the runs, the robot performed all but two of the aspects of the competition successfully. This is a success!

The aspects of the robot that failed were unsurprising. The searching aspect was failed because the team made the decision not to use the rotating sensor motor, in favour of simplifying the robot's operation (see Software Design Document, section 3.7). This achieved the goal of simplicity, but made the searching algorithm very slow, as the robot needed to periodically turn and look into the zone with the ultrasonic sensor. In the interest of using the allotted time to perform as many components as possible, the searching procedure was not given the amount of time it needed, leading to its failure. The other component to fail was the robot's return home after dismounting the zip line. This component failed because the navigation procedure was not quite accurate enough. The inaccuracy in navigation was due to the lack of odometry correction while the robot moved (see Software Design Document, section 3.5). This inaccuracy was tolerable in all the other aspects of the robot's performance, because localization was performed before and after each piece of navigation, removing accumulated error. In the navigation between the end of the zip line and the home corner, localization was performed only after the zip line dismount, not near the home corner, meaning the robot failed to enter the corner tile (see Software Design Document, section 3.6).

Conclusions

Many things were learned from this course. Most obviously, all the team members improved their software development, hardware development, scheduling, time management, and teamwork skills. In addition, the team learned that a strong team manager is required to motivate the team to get work done in an efficient manner, even when team members don't have a personal investment in the team or in the course. The team also learned that following the outlined procedure (in this case, a GANTT chart) is critical to success in a group project. Finally, the team learned that when changes to the

procedure (GANTT chart) are required, the team needs to discuss options, and the team manager needs to make the final call.

The Engineering design process is necessary when working with a team because, at any given time, each team member needs to know their assigned role and how it contributes to the larger project. Since the components fit together in a specified order, the tasks need to be completed in the specified order. The design process also helps the team avoid disagreements, particularly about what a team member should be spending their time on.

This project taught skills that can be applied to nearly any other course. Specifically, the programming skills taught can be applied to ECSE 223 or ECSE 224 and the time management skills can be applied in any course where a group project is involved. More importantly, the skills taught by this project can be applied to a project at a real Engineering firm!

If the team were to undertake this project again, some approaches would be changed:

- Keep the final competition in mind when performing research and development labs. Remind oneself that the labs are R&D for the project; they're not just for their own marks.
- Allocate team members to a wider variety of tasks. This team needed more people on software, but 2/3 people available were taken by Documentation and Project Management and could not help as much as anticipated. If Hardware and Testing members had been able to help more with Documentation and Project Management, the Documentation and Project Managers could have helped more with software.
- Hold more in-person meetings. Make them happen. Even if they're short, they produce more good things than expected. Discussing over Slack only works if everybody is descriptive and clear, which is not always the case.
- Make GANTT chart modifications official. Instead of just discussing a change, make the change in the chart right away, so everybody is perfectly clear on their new task.
- Pay closer attention to the GANTT chart. Constant (daily) reminders from the Project Manager are not annoying – they are helpful.
- If one is unsure about how a task is going, speak up. If somebody gives an unclear answer to a question, push them for a better answer.
- Ask for more clarifications from Professors and TAs about required tasks and competition specifications.

The team may have encountered some struggles along the way, but in the end, everybody was united around the demo floor, watching the performance of our beloved robot: *Dumb Piece of Metal.*

The undersigned members of Team 06 agree that the contents of both this report and the information handed in on CD, DVD or memory key, provide an accurate representation of the work done on this course and the contributions of each team member.

_____ _____

_____ _____

_____ _____