FIT3140 Advanced Programming
Assignment 5, Iteration 2, Final Report
Zhong Kein James Lee

Topic 1: Agile software development practices
Throughout the FIT3140 course this semester my partner, Daniel, and I have applied multiple Agile software development practices to our software development life cycle, below is a list of them in order of the largest impact with respect to the assignment. Each item will be followed by a brief description and reflection on the practice's impact on our project:

Sprints:
Our team heavily relied on the concept of sprints throughout this assignment. At the start of each assignment cycle, we would sit down together and plan our sprints. With trello, we compiled a backlog of features that needed to be implemented and tasks that needed to be done. We would usually break down tasks into the smallest possible deliverable, and then divide the work amongst ourselves. Each picking whichever task they were comfortable and/or confident with. Our sprints were roughly 3 days long, however this was seldom the case as usually had other obligations and commitments to attend to, we worked on the assignment whenever it was possible. Even so, we did manage to complete our assigned tasks on time.

Every fortnight is when we reach a sort of “check mark” period. This is when we usually would do our sprint reviews and retrospectives. In our sprint reviews, we show each other what are the features we've built and teach each other the general flow of the feature. The sprint reviews are a way for the both of us to update each other about parts of the overall program that we are not familiar with. In the sprint retrospectives, we reflect on the past fortnight's sprints. We look at what went wrong, what went right, areas that we can improve on and areas that we need to improve on. Even though it's not a professional setting, we do our best to be so to encourage that sort of mentality when it comes to dealing with the work that we have given each other. This further pushes the both of us to do good work.

Program Flow:
Through the use of activity diagrams, sequence diagrams and use case diagrams, we were able to share with each other our ideas on how the program's flow should be like. As the assignment got more and more complex, the code needed proper structure. More often than not, segments of the entire code structure were built by the both of us, and we had to make sure that our program remained integral as a whole, lest we waste a lot of time going through unfamiliar segments of code to figure out what some errors are.

Continuous Integration:
With GitHub, Daniel and I pushed and merged our code as soon as a new deliverable has been completed. Not only does this allow us to make have a working prototype at any moment, this allows us to share the workload of some particularly large modules. I find that this practice also encourages us to be familiar with segments of the program that we ourselves did not write.

Code Refactoring & Test-Driven Development:
These are 2 agile practices that we decided against for this assignment. We decided to not use code refactoring because each iteration of the assignment had us using different platforms and technology, in very small parts. So code refactoring had no real value here. For a similar reason, we decided that the time cost of using test-driven development does not outweigh the value gained in quality when implemented.

Topic 2: Working in teams
Our team consists of 2 members, Daniel and myself. We coordinate communicate with each other through an app called Whatsapp since we both have access to it. Since we were not physically together for most of the time the work was done, we relied on Trello boards heavily to keep ourselves up to date on the work that needs to be done, what’s currently in progress and what’s already been completed.

As much as possible we try to keep a professional sense of communication when we talk about work. Whenever we have an issue that needs to be discussed we do so in an honest and open manner. This works because I believe that the both of us truly want to do well in this class and for this assignment, so it really helps because we are both able to take criticism objectively and constructively.

Tasks were allocated based on who is more proficient with the skills required to complete the task, who has more time available and of course, who currently has the Arduino board in their possession. Although it was a little bumpy at the start, this turned out to be a rather good arrangement as we both had strengths in areas where the other did not. As a result I believe we complemented the groups work as a whole.

In a small programming company, I believe this practices would hold. One key point to note is that both Daniel and I earnestly wanted to score well for this assignment, which led to a very objective mentality that we both had when we tackled our work. This assignment has led me to believe that a unified sense of wanting to excel and achieve more would inevitably cause a group or team to progress as a whole.

Topic 3: Design

For our morse code app, I believe it would be fairly easy to support additional variations to the app. Our code structure features low coupling and good encapsulation. Therefore adding additional functions will be a simple act of adding a few lines of code in key locations.

Fixing bugs however is a different story. More often than not the slow response time of the PIR motion sensor is the major drawback when hunting for bugs in the program. As a result, the error fixing process consumes a lot of time. Aside from that, it helps that our code structure was designed in a very understandable and encapsulated format.

With Material Design Lite, our app features a very simple and clean design, with a text box that acts as a console. This features provides a feedback mechanism to users which gives them a very engaged and hands-on feeling when using our app. Whenever a you hit a switch, feedback is given, indicating that something has changed in the app. This gives a very responsive feeling to our program. Besides that, the clean and simple design makes using the app very intuitive and hassle free. There are no distractions and users can immediately focus on what's important in the morse code app.

Similar to adding additional functionality in the code architecture, adding new controls is simple process too. The code is designed in such a way that adding new features (be it new controls or back-end functions) are relatively simple. Coupled with the fact that most functions can be reused to do small tasks, such as updating the console messages, adding new UI-based features will not be difficult. As for the design aspect of the app, we did not focus much on it as it was not the focus of this assignment. We do not have a separate CSS file for the design of the app. Even though we do use the MDL template, which has most of the necessary design specifications already, adding major changes to the look and style of the app would be a sizeable task. However, minor changes will not be an issue.

Conclusion:

In the end, I've learnt a lot about working as part of a team and at the same time striving to do well in an assignment throughout this class and assignment. Even though having to plan out my work, communicate things and reflect can be easily perceived as having to do more work that doesn't necessarily contribute to the final app. I have realised it's long term advantages now that I've stuck with these practices throughout the course of this assignment.