# Project Management

For this project, I decided to adopt an agile methodology for software development. I decided upon this as I wanted to be able to develop a quality product that was able to adapt quickly to feedback I received from the project stakeholders. Being able to schedule frequent supervision meetings with Dave Pearce allowed me to develop and deploy working prototypes and receive feedback on my design and implementation much quicker and more frequently than planning and developing a single product would have allowed for.

## Agile Methodology

As I adopted agile project management techniques for this project I followed the agile manifesto:

“**Individuals and interactions***over processes and tools*

**Working software***over comprehensive documentation*

**Customer collaboration***over contract negotiation*

**Responding to change***over following a plan”* [1]

If I instead developed the project using a more traditional waterfall methodology I would be unable to quickly adjust the plan and re-develop components based on feedback, I found this feedback crucially important to develop a quality final product.

I quickly developed and iterated on prototypes having frequent supervision meetings with Dave Pearce where we discussed these prototypes and what additional features or improvements could be made to the software.

## Prototypes

During the project, I developed three prototype visualisations. Each of the prototypes introduced a core component that was necessary to realise the final baseband visualisation:

* Drawing and updating user-modifiable waves
* Performing a DFT on a user-configurable waveform
* Encoding a user-modifiable data stream using a user-selected line coding scheme.

Developing these prototypes in an agile manner allowed me to quickly develop a working product and receive feedback to make improvements to each component without lengthy development cycles.

### Prototype 1 – Automatically Updating Sine Waves

The first prototype I developed for the project, allowed me to learn how to draw automatically updating waves on an HTML 5 canvas element. The prototype gave the user 4 sliders allowing them to modify the: amplitude, phase, frequency, and velocity for each wave. I also included an Argand diagram [2] displaying the amplitude and phase of each wave.

The user can click on the Argand diagram as an additional method of setting the amplitude and phase of each wave.

A screenshot of the prototype can be found below. An interactable version along with the code can be found in the Prototypes section of the appendices.

A screenshot of a computer screen

Description automatically generated with low confidence

Figure 1 - Screenshot of the first prototype.

### Prototype 2 – Discrete Fourier Transform

The second prototype I developed for this project involved performing a DFT over a summation of four user-defined sine waves, plotting the frequency content of the wave and then attempting to identify each of the four frequencies that constituted the wave.

This prototype allowed me to ensure I had properly implemented the DFT algorithm. I initially was only plotting the real component of the frequency content which resulted in the frequency content being displayed incorrectly. As this prototype was quick to develop, I was able to quickly schedule a supervision meeting with Dave Pearce, where he quickly pointed out the issue, thus I was able to fix my implementation of the DFT algorithm for this prototype and future implementations.

A screenshot of the prototype can be found below. An interactable version along with the code can be found in the Prototypes section of the appendices.

A screenshot of a computer screen

Description automatically generated with low confidence

Figure 2 - Screenshot of the second prototype.

### Prototype 3 – Line Coding

The final prototype I developed before beginning development on the baseband communication visualisation was a line coding prototype. This prototype includes the Binary Signal Generator which I used for the final visualisation, the user can interact with the data stream in the same ways they can interact with the final visualisation. The encoding schemes I introduced in this prototype are also included in the final visualisation.

When I first spoke with Dave Pearce regarding this prototype he gave feedback that it would be nice to include MLT-3 Coding. Due to the agile nature of the project, I was able to quickly develop and include this in the prototype and quickly receive feedback regarding my implementation.

A screenshot of the prototype can be found below. An interactable version along with the code can be found in the Prototypes section of the appendices.

A screenshot of a computer

Description automatically generated with medium confidence

Figure 3 - Screenshot of the third prototype.

## Reflection

Whilst I believe the project overall was a success, having been able to meet the goals of developing a web-based baseband communication visualisation. There were improvements I could have made to my project management techniques that I believe would have helped me to better utilise my time and therefore have more time to further develop the project and make improvements.

### Agile methodology

I believe I made the correct decision to develop the project using an agile methodology. However, I do believe I would have been able to better manage my time by utilising scrum techniques [3] such as sprints, sprint planning and sprint retrospectives. I naturally utilised some of these techniques during the project, such as sprints, but I did not think of them as such and thus some vital components of these techniques were lost.

For example, on some weeks I spent Friday afternoon, Saturday and Sunday developing a prototype or set of features for the project. However, I did not designate this time as a sprint- rather adopting the mantra of “It’s done when it’s done”. Unfortunately, this meant that by some Sunday evenings, I was not close to having a version ready for deployment. If I instead conducted a sprint planning session before the start of the sprint this would have helped in setting appropriate goals for the sprint and helped ensure I was ready to deploy a version by the end of the sprint. Additionally, I did not have sprint retrospective sessions, these sessions evaluate how effective the last sprint was and how future sprints can be improved. Conducting these sessions would have helped me to set appropriate goals and objectives for subsequent sprints.

### Software Testing

Whilst I am happy with the testing performed on the project I do believe this testing was performed too late in the project. If I instead adopted Test Driven Development [4] techniques from the start of the project and performed unit tests on core functions when they were created this could have helped avoid time loss due to debugging later in the process.

For example, my complex division function was incorrectly implemented during earlier stages of the project. As this function did not need to be utilised until implementing the IDFT I did not spot the incorrect implementation, this resulted in time being wasted debugging the IDFT function and plotting methods as I was unsure where the error was being introduced. If I performed unit testing on the complex division function when it was first implemented then I would not have lost this time.

## Risk Assessment

Below is a risk assessment table I created earlier in the project. I will evaluate how successful it has been throughout the project in mitigating risks and if any additional risks should have been considered.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Risk | Severity | Probability | Seriousness | Mitigation |
| Losing project code due to laptop crashing. | 10 | 3 | 30 | Back-up code on GitHub, keep an offline backup. |
| Report documents may be lost or become corrupted | 10 | 3 | 30 | Create regular backups of all report documents. Store these backups in multiple places including a cloud backup. |
| Scope creep, leading to being unable to finish the core aims of the project by the deadline. | 10 | 4 | 40 | Stick to the must-have requirements until the core aims are met. Create regular working prototypes to ensure aims are always being addressed. Only implement nice-to-have requirements after the core aim is met |
| The project is not progressing according to the schedule | 8 | 3 | 24 | Follow the timetable closely. Schedule regular supervision sessions and ask for help when it is needed. |
| Falling ill and being unable to make the required progress on the project | 8 | 2 | 16 | Ensure must-have requirements are met far ahead of the deadline. Communicate potential issues early and draft sections of the final report as soon as possible. |

The first risk ‘Losing project code due to laptop crashing’, did occur. In February of this year, my laptop’s drive became corrupted. This would have been disastrous for the project if the code was not backed-up on GitHub. The offline backup was created, however, was fortunately not required as GitHub contained the most recent version of the software.

The other risks were fortunately not encountered during the project. However, I wish I had included the risk of “Core components failing to work as expected”, this risk could be mitigated by ensuring that each component was fully tested before work should be started on additional features.

## Time Management

Throughout the project, I have been utilising a Gantt chart, which is included in the Gantt chart section of the appendices, to plan my progression towards both achieving the project’s goals and writing the initial and final reports.

The Gantt chart shows my progression towards both achieving the project’s specifications and writing the initial and final reports. The objectives are planned each week so that the plan can be tracked with enough detail to ensure deadlines are always hit. But, without so much fine detail so the timetable remains realistic.

The Gantt chart includes the time allocated for researching the web technologies which I utilised for creating the visualisation as well as the techniques required to implement baseband modulation, including the DFT, filtering techniques and line coding methods. Additionally, the Gantt chart includes the time spent on the three prototype simulations described above.

I believe the Gantt chart has been essential for keeping the project on schedule by providing me with visibility into what tasks have been completed and how many tasks still need to be completed to meet the deadline.

# References

[1] K. Beck, et al. (2001). *Manifesto for Agile Software Development*. AgileManifeso.org. [Online]. Available: <http://agilemanifesto.org/> [Accessed 6 May 2023].

[2] Wolfram MathWorld. *Argand Diagram.* Mathworld.Wolfram.com. [Online]. Available: <https://mathworld.wolfram.com/ArgandDiagram.html> [Accessed 7 May 2023].

[3] C. Drumond. *What is scrum and how to get started?* Atlassian.com. [Online]. Available: <https://www.atlassian.com/agile/scrum> [Accessed 8 May 2023].

[4] G. Steinfeld. *5 steps of test-driven development.* IBM. [Online]. Available: <https://developer.ibm.com/articles/5-steps-of-test-driven-development/> [Accessed 8 May 2023].