

Method selection and planning

Group 6 - M6

Members:

- Mir Baydemir
- Adam Fraulo
- Azib Hj Abu Akmar
- Agata Pittarello
- Esther Scanlon
- Jazz Stubbins
- Sonia Szetela

Part A:

As a team we have been influenced by the Agile Manifesto, in particular the principle of building projects around motivated individuals and trusting them to get the job done. The agile manifesto also states that "Face-to-Face conversation is the most efficient and effective way to convey information" Although as a team we discussed some topics online using whatsapp, we also made sure to meet in person at least once a week to review our progress as well as discussing ways on how to become more effective as a team. Agile Method is suitable for this project because the size of our team is small (7 people) and there is a flexible project environment. This method allows us to adapt quickly to new ideas or challenges and improve the project over and over again. We have particularly influenced the Scrum approach, we have set ourselves weekly goals and each person has done their task before the weekly meetings where progress was reviewed and new goals were planned. This approach has helped us to keep the time aligned and each member busy as well as encouraging us to keep a detailed record of our work and easily adapt to new ideas and challenges.

We created Whatsapp group chat for instant communication between team members. All important announcements like meeting times, progress updates etc... are made by Whatsapp. When team members were not able to attend a meeting or expected to be late, they informed the rest of the group by Whatsapp. This ensured that everyone was aware of each other's situation and tasks could be redistributed accordingly. This allowed the team to stay organised and respectful of other member's time. We have considered using social media platforms such as Snapchat or Instagram for our communication but then we decided that whatsapp is a more professional and reliable option. That is because Whatsapp provides a more focused environment with less distractions and it has features like file sharing that could be useful for the project. Furthermore, using Whatsapp has greatly helped us to make use of the Scrum approach. Because while using the Whatsapp group chat we were able to assess progress everyday rather than waiting until the weekly meeting. Therefore, Whatsapp gave us an opportunity to get continuous feedback, stay informed about each other's progress and make small adjustments daily. In addition, Whatsapp also helped us to self-organise as a team, which reflects another principle of the Agile Manifesto, which states that "The best architecture, requirements, and designs emerge from self organising teams" . By maintaining frequent and transparent communication we were able to coordinate tasks effectively and make very fast decisions.

We used Google Docs instead of Microsoft Word because Google Docs is more transparent and it allows multiple collaborators to work simultaneously on the same document. This made it easier for team members to review and edit content in real time. In addition, not every group member had a Microsoft account but everyone had a Google account so Google Docs was a better option.

For the architecture design we have used draw.io and lucidchart.com to create system and UML diagrams. They allowed us to visualise data flow, object interactions and game structure. Both of them support collaborative editing which allows multiple team members to work on the diagrams at the same time. By using this approach we made sure that our designs were transparent and they were aligned with Responsibility Driven Design (RDD) principles by defining clear class responsibilities. We could have used StarUML as an alternative for creating UML diagrams. It provides more advanced features like design documentation and reverse engineering. However we had to make a payment to use it and it was much more complex to use compared to draw.io and lucidchart.com. As our project is

small in terms of size, draw.io and lucidchart.com were more than enough. They are very easy to use, free and contain all the tools to create an UML diagram. In addition, two of our members have already used draw.io and lucidchart.com before the project so they already had some familiarity with them.

For the implementation part, we coded mainly in Java 17, using Visual Studio Code (VS Code) as our main IDE. We could have used the IntelliJ IDEA as an alternative IDE because it has far more superior support for Java and it has advanced debugging support. However, we chose VS Code because it is much simpler to use and all team members have familiarity with the VS Code. The project was managed from GitHub which provided full transparency. Every Team member had access to the shared repository, allowing everyone to track progress and review code. The built-in LibGDX libraries were used to support game development features like object interactions and graphics rendering. Furthermore, [Paint.net](https://paint.net) was used to trim graphical assets which helps us to optimise visuals for game's design and performance. We could have used GIMP instead of [Paint.net](https://paint.net) which offers more advanced tools. However, [Paint.net](https://paint.net) is much easier to use and offers faster workflow hence that is why [Paint.net](https://paint.net) is chosen instead.

Our approach to implementation follows one of the principles of the Agile Manifesto: "Working software is the primary measure of progress" We frequently reviewed the game throughout development rather than waiting until the end. By doing that we made sure that the software was always in a usable state and improvements were done by the feedback from team members. Another principle of the Agile Manifesto is that "The team should regularly reflect on how to become more effective" After each task was completed we talked about what went well and what could have been better, helping us to become more effective.

Part B:

The first thing we did as a team was to create an excel sheet called 'Work Split' in which we assigned tasks to each team member according to their strengths and interests. We made sure to distribute the workload evenly so that each member contributes roughly equal amounts of time and effort to maintain fairness. Each deliverable (Requirements, Architecture, Planning, Risk, Implementation, Website) was given one main responsible person and several supporting team members with moderate involvement. For deliverables that require a greater effort and have a more workload (e.g Architecture and Implementation) two main members share responsibility. This approach is consistent with the principle of avoiding a low bus factor as we made sure that no critical task depends on a single person. This approach also gave us an opportunity to share out the leadership role as we had different leaders for different tasks which prevents one person from carrying excessive responsibility and instead spreads it across the team.

As a team we knew the importance of constructive criticism and we made sure that feedback was always given respectfully. All the team understood that honest and realistic feedback was essential for improving the quality of our work and avoiding mistakes. To support this, all members were given access to both Github and Google Drive, allowing every member of the team to review each other's progress and leave suggestions. This approach reflects all three elements of the HeaRT principles (Humility, Respect and Trust). In terms of humility, all members of the team accepted the views of others and made changes in their work when necessary. Respect was shown by recognising team member's effort and accomplishments and through kind communication. Finally, trust was shown by having confidence that teammates are competent and will do the right thing.

In addition to the HeaRT principles we are also aware of the fact that frequent peer reviewing also increases the bus factor which reduces the amount of mistakes that we make but also strengthens collaboration and team resilience.

While preparing our schedule, we ensured that our assumptions and expectations were pessimistic rather than optimistic. As implied by Murphy's Law "Anything that can go wrong will go wrong" so we ensured that our plan accounted for potential delays and unexpected challenges. As a result, we left sufficient time for each task. We also were a flexible team that was able to replan whenever necessary. Just like how the Swiss Army manual advises "If the map and terrain disagree, trust the terrain" we applied this principle throughout our project by frequently reviewing our progress and changing our schedule when actual conditions differed from our plan. This attitude helped us to maintain a realistic and well-coordinated workflow.

Part C:

Introduction:

Our plan had an agile structure followed by weekly meetings (although when it was not necessary to do a meeting we discussed the topics to be discussed on Whatsapp and did not do a meeting, by doing that we saved time that could have been effectively used for the project).

The Planning process was pessimistic rather than optimistic. As implied by Murphy's Law "Anything that can go wrong will go wrong". Therefore we left enough time between important tasks for unexpected challenges. This approach has reduced the pressure on team members and reduced the number of potential delays.

KEY TASKS	Starting Date	Finishing Date	Priorities	Dependencies
Risk1	17/10/2025	24/10/2025	High	Plan-B
Plan1-A	14/10/2025	24/10/2025	High	None
User Requirements	3/10/2025	14/10/2025	High	None
System Requirements	3/10/2025	14/10/2025	High	User requirements
Plan1-B	17/10/2025	24/10/2025	Low	Plan1-A
Creating the Website	3/10/2025	7/10/2025	High	System requirements Arch1 Impl1 Plan1
Arch1 UML Class Diagram)	14/10/2025	24/10/2025	High	Impl1
Arch1(General plan relationship Diagram)	14/10/2025	3/11/2025	Medium	None
Arch1(UML component diagram)	3/11/2025	7/11/2025	High	Arch1(General relationship Diagram)
Impl1	14/10/2025	7/11/2025	High	Req1

Plan 1-C	3/10/2025	9/11/2025	Medium	Plan1-A Plan1-B Req1 Risk1 Arch1 Impl1
----------	-----------	-----------	--------	---

How did the plan evolve?

The plan has changed a few times because some of the tasks were more challenging and more time consuming than we expected. For example, our plan was to finish requirements at 10/10/2025 but there was a slight delay so we finished it at 14/10/2025. This delay did not cause any big disruptions because we had a pessimistic planning approach meaning that we were always prepared for the worst case scenario. Therefore as a team we have quickly adapted for this situation and created some free time for this task to be finished as soon as possible.

Similarly, for Arch1 the planned internal deadline for creating UML class diagrams was 21/10/2025 however the maze game system is being written in Java using LibGDX and there was a problem with the java code so it took longer than expected. As a result there was a 3 day delay for making the UML class diagram which was not a problem at all because as a team we quickly reorganized and that date was already much before the submission deadline.

Lastly, for Plan1A the planned internal deadline for finishing it was again 21/10/2025 but we finished it at 24/10/2025. Which again did not cause any major disruptions and instead taught us that the tasks could take more time and be potentially harder than it seems so it is very important to be prepared.

After each task was completed, we reviewed what went well and what could have been improved for the next phase. By doing that we became more realistic with the timing and improved the accuracy for later tasks especially in terms of time.

References:

- 1) Murphy, E. (1949) Murphy's Law. Quoted in Project Planning and Risk Management. [Lecture Slides]. ENG1: Engineering 1, Available via VLE: <https://vle.york.ac.uk>
- 2) Beck, K, et al. (2001) Manifesto for Agile Software Development. Available at: <https://agilemanifesto.org/> (Accessed at 22/10/2025)