Method Selection and Planning

Cohort 3 Team 1: Pixels of Promise

Movitz Aaron, Jake Adams, Oliver Belam, Anna Burt, Emily Foran, Sky Haines-bass, Job Young

University of York

ENG1: Software & Systems Engineering

November 11, 2024

4 Method Selection and Planning

4.1 Software Engineering Method Approach

***a)** Give an outline and justification of the team's software engineering methods, and identify any development or collaboration tools that the team has used to support the project or the team working. Justify the fitness of the selected tools with the team's software engineering methods and discuss alternatives considered.*

4.1.1 Agile Methodology & Scrum

Our team made use of the Agile Software engineering methodology throughout our project. One example of how this methodology was applied was regularly employing short iteration cycles, known within the Agile framework as 'Sprints', for each of the project's deliverables. To use the framework appropriately to fit our project's constraints best, we deviated slightly from the standard Agile practice of Sprints lasting 2 - 4 weeks *(Sommerville Ian, 2015)*, and often had 1 week-long sprints to create a near-finalised draft of each deliverable for the Scrum group to discuss. This was most suitable for the constraints of both our project's deadline and group availability.

As a result of these constraints, another deviation from typical Agile practices was hosting only two Scrum meetings a week, rather than daily; our group members were typically too busy with other university responsibilities to meet more often than twice a week, meaning that deliverables usually weren't able to be finalised until the consolidation of group work at the beginning of the week.

Additionally, due to the smaller team size, we had significantly fewer than standard people assigned to work on each deliverable, but ensured that one person in each group took responsibility for the deliverable being completed; this meant that during Scrum meetings, the ScrumMaster was able to consult at least one member assigned to each deliverable who would be familiar with both the amount of work that had been done, and if unfinished, what needed to be done. Despite the smaller team size, we ensured that every piece of work had more than one person assigned to it at any given time, as per the colloquially recognised 'Bus Factor' within Team Management, which highlights the risk involved in disproportionate responsibility within a project.

4.1.2 Flexibility & Management

Agile was our chosen Software engineering methodology due to its flexibility and focus on iterative development. Flexibility was crucial for our team as for several team members, it was the first time they had developed a game, and no team members had experience with the LibGDX game engine. Anticipating the risk of running into an unforeseen issue that could halt our plan, we decided to follow a flexible methodology that would allow us to re-adjust our priorities throughout the project.

This also allowed regular communication with the customer to shape our project's development; while the brief we received from the customer was thorough, this flexible approach removed the risk of miscommunication between the customer and team members from permanently affecting the project, as clarifications in design can later be incorporated into the solution.

4.1.3 Jira for Project Management

One tool our team used to aid with collaboration was Jira, a project management tool that allowed the team to decompose a given project into tasks, and then visualise the timeline of when these tasks need to be completed.

An alternative project management tool we considered was Trello, both owned by the same company and designed for similar purposes. Jira, however, provided more in-depth workflow reporting by allowing team members to further break down each task into components, and report issues faced during development for the team to address. A disadvantage however of Jira is that the tool is fairly difficult to use effectively without prior experience, which, at the time, none of the team members had. While Jira allowed the project to be planned in a greater amount of depth when compared to Trello, it may have been more advantageous to use a simpler project management tool to aid with communication and understanding, additionally, a considerable amount of time was spent learning Jira before we could begin planning the project. The aim of this however was to gain valuable experience with a significantly powerful planning tool, which could be capitalised upon within the remainder of the engineering project, and beyond that, future projects each team member may work on.

Jira worked effectively with our Agile approach, as we were able to assign the responsibility of our tasks to the members of each scrum team on the platform. The tool also provided the flexibility to readjust the priority of tasks after each iteration.

4.1.4 Communication Tool Selection & Justification

Our team found WhatsApp the most effective tool for communication. One reason for this was that it was an app already regularly used by most team members, and was particularly accessible on phones, meaning messages were more likely to be seen faster by team members and less likely to be missed. WhatsApp additionally allows for separate communication channels to be created and categorised through the use of 'Groups', by maintaining separate channels it is similarly less likely for messages to be missed by team members and faster to find previous messages, allowing for more effective team communication. Our team created a Group for each deliverable, allowing members working on the deliverable to discuss progress and issues. This additionally allowed for a new group member later assigned to that deliverable to review previous discussions and be caught up on what work needed to be accomplished.

Alternative communication tools considered were Discord and Slack, similar platforms that allow teams to organise discussion via separate communication channels, and provide helpful features for restricting who has access to specific information within a Group or Server. We decided against using either of these tools as they were less accessible to use on a phone than WhatsApp, and due to our smaller team size, features for restricting information were unnecessary.

Bibliography:

Sommerville, Ian. *Software Engineering, Global Edition*, Pearson Education, Limited, 2015. *ProQuest Ebook Central*, https://ebookcentral.proquest.com/lib/york-ebooks/detail.action?docID=5185655.

4.2 Team Organisation

***b)** Outline the team's approach to team organisation, and explain why the chosen approach is appropriate for both the team and the project.*

Our team's adoption of agile methodologies to develop this project directly influenced our organisational process. Agile methods emphasise flexibility, collaboration, and iterative development, which was cultivated through defined roles and delegated responsibilities within teams. We divided into smaller groups responsible for managing their own tasks and workflows, based on the marked sections in the assessment document. Separate group chats were established within a WhatsApp community, allowing members of the small teams to brainstorm, plan future and current tasks, and make small decisions. Larger group meetings that involved all seven of us were used to review progress, update the larger workflow chart on Jira, and finalise major decisions that would affect the group.

The network-like structure of our group eliminated rigid hierarchies, allowing team members to share the leadership role when guiding conversations during the meetings. This promoted shared accountability among all members, rather than relying on a singular appointed project manager. Another reason why we were against solely one person taking on a leadership role was to avoid a single point of failure in decision-making. Instead, decisions were reached through consensus, and when differing opinions emerged, compromises were made, leading adjusted outcomes.

To maximise learning, we adopted a cross-functional approach during the first assessment; allowing each member to get exposure to various aspects of the project, especially during the implementation stages. This broadened and sharpened everyone's skills for the next assessment where we could decide on either a specialised roles approach based on past performance, or continue with the current organisation. The collaborative foundation efforts ensured that every member of the team developed a well-rounded understanding of the various project components, which further facilitated informed and effective decision-making during group meetings.

The chosen approach was suitable overall for not only the team, but also the project due to its ability to mitigate risks and ensure each component would be completed in a timely manner. By adopting agile methods with smaller teams, we were able to break down the project into manageable sections that corresponded with the complexity and duration of the tasks. The use of Jira to monitor progress helped maintain transparency, ensuring crucial milestones were met, and with the promotion of iterative development, we were continually making improvements and adjustments to the calendar. As mentioned previously, shared leadership and cooperative responsibility for decision making ensured that we met the fast-approaching deadline and did so inclusively, obtaining approval from all our team members.

4.3 Plan Development

c) *Give a systematic plan for the project. Your plan should lay out the key tasks, their starting and finishing dates, as well as task priorities and dependencies. Discuss how the plan evolved throughout the duration of the project.*

4.3.1 Project Overview

The goal of this project was to develop a single-player university-building simulation game suitable for all age groups, mainly targeting and engaging open day attendees. The game's objectives centred on designing a functional campus and maximising student satisfaction by reacting to planned or unplanned events. Requirements were mainly based on the number of specific types of buildings placed, ensuring the game's playability in a five minute window, as outlined in the product brief. Before creating a plan of action for all aspects of the project, we created a list of questions and met with the stakeholder to clarify our understandings on the game's substance, as their interpretation was essential to our approach. With this understanding following the meeting to elicit requirements we moved forward with planning the phases of development.

4.3.2 Systematic Plan

Our project required the completion of **6 deliverables**, and with our phased plan they were broken down into key tasks with team members allocated to them. The "public face" of this project is the website, where all of the PDF versions of our deliverables were linked as well of the game. Since the website needed to document the team's weekly plan, it was the first delegated task. Work on this deliverable was planned to begin 07/10 and completed by 11/11, with Mo assigned to this task.

Following our interview and analysis of recording with the customer we continued into the next phase of the plan, where we solidified the user requirements and identified risk factors for the project development. Identifying the user requirements involved developing a table of user, functional and non-functional requirements. This also required designing and upholding a requirements referencing system to highlight the relationships between requirements and make the table easier to interpret. The task of creating the components of the tables were divided among Sky, Jake, and Emily, with Sky also drafting a concise introduction documenting how the requirements were gathered. This process began on 07/10 with the plan for it to be completed by 14/10. Risk management and mitigation was broken down into two tasks: explaining the risk management process the team followed used to identify, analyse, mitigate, and monitor risk, and produce a systematic tabular presentation of the identified risks. Though this deliverable was planned to begin on 07/10 and had a target completion date of 14/10, Oliver and Anna, the group members assigned to the task did discuss an extension at the beginning since further risks may arise as the project went on.

The next phase of the project included documenting our plan and methodology choices, and group meeting discussions. This included recording meeting minutes and continually updating our Jira board weekly, which happened alongside phases one and three. While our methodology and team organisation approach was determined through group discussions following learning about the different types, specific members were assigned to the section to back up our decision with extensive research and justification into which software methodology and tools would be most suitable for our project. This ongoing documentation and planning process began on 7/10 and was scheduled for completion by 14/10, with Job, Jake, and Emily initially assigned to this task, but we anticipated extending the deadline as the documentation would continually evolve throughout the project.

The final phases of the project focused on planning the architecture of our solution, followed by and eventually overlapping with its implementation (decision made on 20/10). Architecture planning began 14/10 with a target completion date of 20/10 allowing it to be completed before the introduction of implementation. Mo, Oliver, Job, Anna, and Sky were initially assigned to this task. Key tasks included weekly visualisations of the project plan in a UML diagram, which were documented on our website along with justifications for these decisions. Additionally, CRC cards were created to outline system processes and represent relationships between functions with references to the elicited user requirements to justify the architecture approach. Our original plan was created during the design process of the game, so the key tasks to implementation were yet to be developed, but we knew they would be dependent on our architecture approach. Continuing to maximise the team's cross functional learning opportunities, all team members were assigned to work on this task, starting 21/10 to be completed by 11/11.

4.3.3 Project Evolution

Our project met several unforeseen issues which required discussion to overcome. One such issue was the customer's plan for the game proving incongruent with aspects of our initial design; in particular that it should be possible for the player to lose, with the game terminating early. After discussion with our customer about the game's expected audience, we elicited that those playing the game would be prospective university students and their family. In light of this, we anticipated scenarios that the game would have to accommodate for, such as; individual users would most likely play for only several minutes or less; users could begin playing at any state of the game's progress; the game would be left unattended for minutes at a time. These factors were not conducive to the possibility of a 'losing state' as it would be disheartening for a player, beginning near the end of the game's timeline, to suddenly lose due to either the game previously being left unattended or, due to a prior player's actions. To answer these possible scenarios, we altered our plan to not implement a losing state. This communication with our customer also highlighted the importance of providing an accessible experience, so that the game could be understood, and therefore enjoyed, regardless of video game experience, which going forward became a high priority when planning our game.

Other unforeseen circumstances included rescheduling deadlines for deliverables, due to either external factors, or progress on work highlighting the need to adjust priorities. For example, we had initially planned to begin work on the implementation of our game after finishing the architecture deliverable, however, due to the lack of experience group members had with the LibGDX game engine, those assigned to the architecture deliverable were unaware which methods to include in each class, and so were unable to plan the structure of the solution. Due to this, members working on architecture began work on implementation early, and the deadline for architecture was extended to match the implementation deliverable, to be continually updated as work progressed; similarly the deadline for the method selection and planning deliverable was extended until the project's completion, as it required the continual documentation of the project's plan.