Method and Planning

Mozzarella Bytes | Team 18

Assessment N°1

Daniel Benison

Elizabeth Hodges

Kathryn Dale

Ravinder Dosanjh

Callum Marsden

Emilien Bevierre

Method and Planning

Different Methodologies

The Waterfall method is a rigid way of working where a project is split into stages, with each stage needing to be completed sequentially [1]. This method is best suited to large teams working to a strict project brief [2] and so is unsuitable for us as we are a small team working on a bespoke game where the requirements are likely to change. However, as our project has been split into four assessments that build on each other we are forced to incorporate elements of the Waterfall methodology to keep to assessment deadlines.

The ideas behind an Agile methodology underline many modern methods of working. It is an iterative system [1] designed to be flexible and allow changes to be made when needed, making it very suitable for us to us as we need to be able to go back and revise work that was completed earlier to ensure that it's all in line with the requirements. Communication with the client is one of the four core elements of the Agile approach [2], frequent communication between the development team and the customer is important to ensure the software meets the clients needs. Overall an Agile approach is well suited to our project as we are creating a bespoke game for a specific client and so all features we develop should be purely client driven.

SCRUM is a specific type of Agile method. It divides time into fixed length 'sprints' which all have clearly defined goals and provides constant iterations of a working piece of up-to-date software [2]. At the start of each sprint, a set of goals is defined based on the brief given by the client and any feedback from previous sprints. During sprints there are daily meetings (SCRUMs) between team members to keep everyone updated as to the stage of all pieces of work [2]. Reviewing the project's progress regularly and communicating frequently as a team would be an effective way to keep the team on track.

We have decided to adopt a Scrum method as it has been proven to be an effective way to manage a small team project with constantly changing requirements [3] and it will help us to keep the project on track. Due to our other commitments we will conduct a virtual Scrm every two days instead of the recommended daily 15 minute Scrum [3]. These meetings can help us prevent any individual team member from falling behind and will be conducted online to make it easier for all team members to be present. Furthermore, Agile's strong focus on regularly producing small working pieces of code will provide us with something to show the client that will help us to catch any problems that the client has early on, giving us the ability to rectify issues quickly.

Collaboration Tools

- **Git**
  - We decided to use Git for our Version Control System. Git allows us to share our work among the team, work on the code collaboratively and restore previous versions. Git's decentralised version control allows team members to create a branch off of the main code where they can work on, and test, parts of the code before merging it back with the main code. This feature eliminates the risk of a team member breaking previously working code. In addition, as each member of the team will have a local version of our project they can work on it offline.

- **GitHub**
  - Our Git repository and website is hosted on GitHub. This is because its intuitive interface easily allows the code to be viewed, tracked and commented on.

- **Facebook Messenger**
  - We can organise meetings and conduct our virtual SCRUM via Facebook Messenger as well as using it to create a group chat where we can communicate as a team. As it is a mobile app, team members will be notified of new messages making them likely to see and respond to messages quickly. Due to messenger being designed as a communication tool it's interface makes it easy to respond to, and keep track of messages.

- **Google Suite (Drive, Docs etc.)**
  - The Google Suite (primarily Google Drive and Google Docs) enables us to create a shared drive for the project where team members can upload and view relevant documents. Work saved on the Drive is automatically backed up minimising the risk of losing documents or progress. Furthermore, people can add comments and work on documents collaboratively.

Planning Tools

- **GitHub Projects**
  - Allows tasks to be allocated and managed. It also creates a virtual "Kanban-style" board where the project's progress can be visualized and tracked.
- **StarUML**
  - StarUML was used to design models of our architecture. We chose this tool because it creates easy to read diagrams and has a simple to use drag-and-drop style interface.

Team Organisation

Roles

The following roles evolved from group discussions and research into the SCRUM methodology. We modified the main SCRUM roles [3] to suit our team and added additional roles that we felt were important for creating the product and completing the assignment. To allocate roles we created an online survey (can be viewed on our website [4]) to discover how each person works best, their strengths and which first year modules they felt most/least confident with to give us a better understanding of everyone's abilities. This, and people displaying an affinity towards particular roles in group discussions, were used as the basis for allocating roles.

Project Owner: Responsible for producing the final product. This role includes identifying key tasks, keeping the project on track and ensuring the final product meets the stakeholders' requirements [3]. Kathryn's organizational, team building and leadership skills make her a good fit for this role.

SCRUM Master: Responsible for helping the team implement the SCRUM methodology [3] and organizing the teams virtual SCRUM every two days. They also identify, mitigate and manage the project's risks as the SCRUM master should remove obstacles that will slow the development team's progress [3]. Ravi is a good fit for this role due to his planning skills and ability to keep calm under pressure.

Development Lead: Typically the development team is self-organized [3], however as not everyone is confident at coding, we created a head of development to oversee this aspect of the project. They are in charge of allocating tasks to team members, creating tests and making sure the code is consistent to reduce the likelihood of code clashing. Emilien suits this role due to his coding skills, experience with game design and interpersonal skills.

Client Collaborator: Maintains communication between the team and the stakeholder as client collaboration is one of the four main concepts of agile development [6]. They also schedule meetings with the client to identify their needs before relaying them to the team and to update the client on the project. Callum's interpersonal and communication skills make him suited to this role.

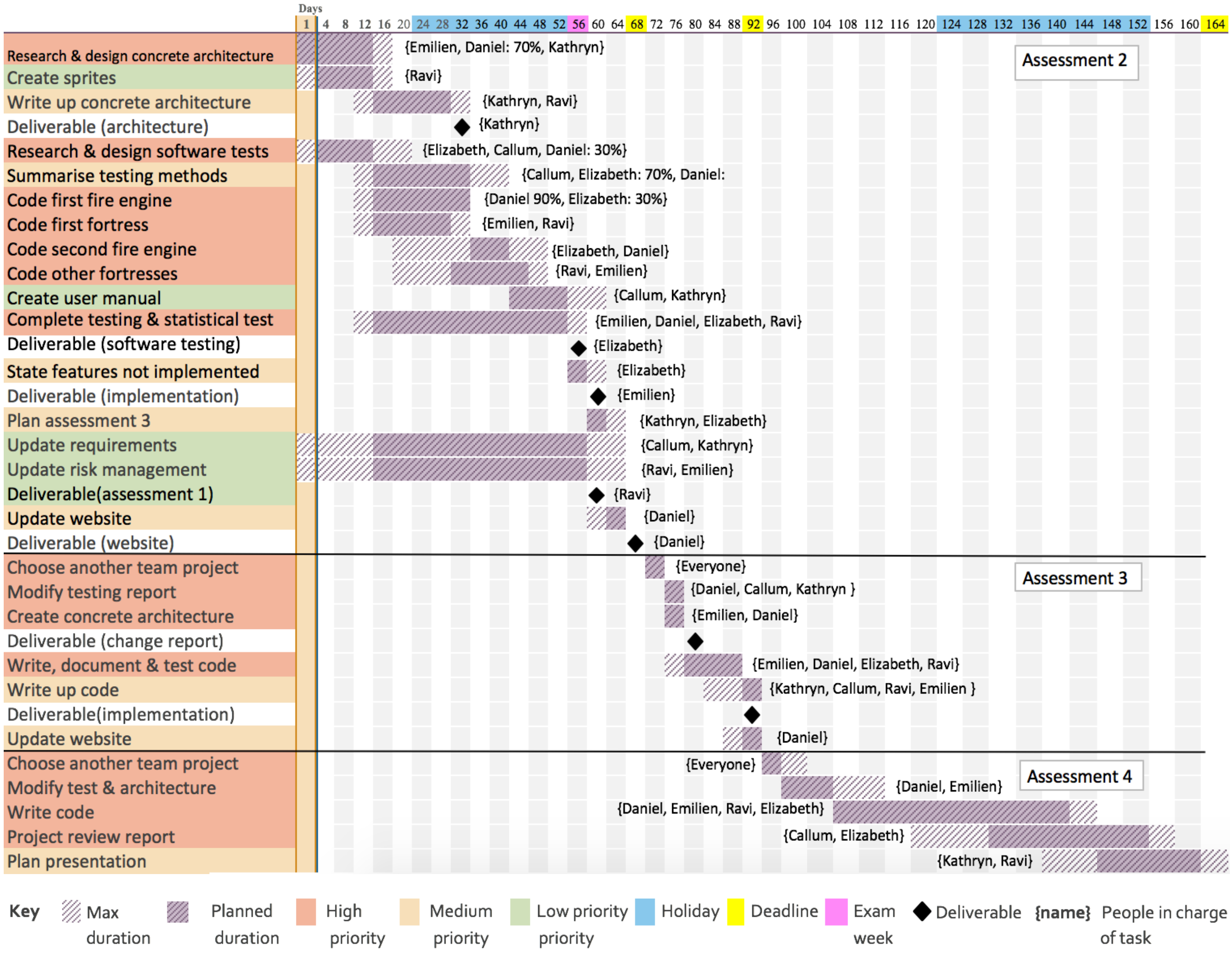
Report Editor: Edits and submits the deliverables, as well as checking that documents are internally consistent, readable, within the page limits and that the content meets the assessment criteria. Elizabeth's writing, organisational skills and keen eye for detail means she is suited to this role.

Website Coordinator: Ensures the website where the team shares important information about the project with the client is up to date and easy to navigate. This person is responsible for building, maintaining and uploading deliverables to the website. Daniel suits this role due to his experience with website development.

Meetings

As SCRUM master Ravi will host our virtual SCRUM every two days where we will discuss the team's progress and deal with any issues that have arisen. This is as well as at least two face-to-face meetings a week where we can discuss the project in more depth.

Planning



Explanation of key:

- *Task dependencies:* Tasks are dependent on previous tasks if their planned start date is later than the previous tasks end date
- *Critical path delay:* Composed of the high priority tasks in the order shown
- *High priority:* Critical tasks
- *Medium priority:* Tasks that are important but not critical
- *Low priority:* Tasks that need to be completed but no real deadline
- *Task allocation:* Dependent on personal strengths/weaknesses, other tasks allocated, tasks performed in assessment 1
- *Planned duration:* Dependent on how priority, available time, perceived amount of time needed
- *Maximum duration:* The furthest we can feasibly fall behind schedule
- *Key tasks:* Identified using a work breakdown structure based upon deliverables from assessment brief

Change management plan: The chart template allows the user to input the % of each task completed. Every four days a task owner will input the tasks % completion. If a task is behind schedule the task owner will update the change management log (allowing everyone to keep up with changes) [7] with a proposed solution. Once the project owner authorises the decision it will be implemented and added to the chart otherwise a new solution will be discussed.