Method Selection and Planning - Runtime Errors - Team 25

Names:

Charles Stubbs

Alex Shore

Annabelle Partis

Kieran Ashton

Yu Li

George Tassou

Part A

Throughout the software engineering process, as a team we have used a wide variety of tools in order to complete the project to the highest possible quality. These tools range from development tools we have used for the creation of Auber as well as collaboration tools to increase the efficiency of group working and communication.

To aid communication throughout the process, we have decided to use Zoom for all of our meetings, Zoom is a VoIP software allowing for groups of people to communicate using webcams and microphones on their computers/smartphones. Zoom enables us to schedule meeting sessions in advance, ensuring as many group members as possible can make it to the meeting. By having group meetings with as many people present as possible, the communication within the group should be more efficient, increasing productivity across the group and therefore allowing for as many tasks as possible to be completed.

Another tool used for group communication is Slack. Slack is a free communication platform used as a group chat tool. Within our Slack 'HQ' our communication is split into different chat channels, this helps us to be organised as a group as certain communication is restricted to specific channels, for example, the "code-help" channel is specifically for when any group member may need help for coding. By using Slack, we can be more organised in our communication, leading to the group being more efficient in our work and therefore helping with productivity as team members will know where to look for specific information. Discord was a viable option for group communication that we also considered when initially beginning the project, Discord works very similarly to Slack in terms of using channels to communicate, however it is clearly targeted toward the gamer demographic as supposed to collaborative working, for this reason we chose Slack as small features, like being able to make a poll for questions, will aid us when it comes to making group decisions.

Furthermore we have used Google Drive to organise all of our documents within the project. Google drive allows for shared drives to be used where we can place all documents for the project and they can be edited by anyone with access to the document. This allows for group feedback to be made as anyone can read the documents as well as edit them. Google docs also allow for collaborative working as multiple group members can work on a specific document at one time, this would therefore increase efficiency for working through documents.

To create Auber, we have used IntelliJ combined with the libGDX application framework, libGDX uses Java as the coding language. IntelliJ and libGDX allow for the production of code in Java, this allows for software development of Auber. Using IntelliJ and the libGDX engine will help in terms of speed and efficiency when it comes to making Auber, the use of a premade engine cuts development time as we do not have to develop our own engine for the game to run on, therefore allowing for our deadlines to be met in terms of when everything should be completed. A possible alternative we considered for IntelliJ was the Eclipse coding environment, also for Java. Eclipse was a viable option however we had trouble connecting it to GitHub as well as using a LibGDX with Eclipse, this meant we would've spent too long trying to setup the initial configuration when it came to coding, and would not have been the best use of our time.

In addition, we have used GitHub as a collaborative tool to upload and share our code between the group. GitHub allows for group repositories to be created and used by any member who is invited to it, when a group member is coding, they can ‘pull’ the most recent data for Auber from the GitHub repository and edit it accordingly, the user can then ‘fetch’ their most recent updated version to GitHub for the team members to use. This allows for constant alterations of the code to be made, allowing for a greater level of collaboration and teamwork while developing the software and therefore making a higher quality game.

All of the collaboration and development tools chosen were chosen to help aid the software engineering and collaborative working process. We have chosen an agile software engineering methodology, this involves creating the game in small increments, working on one element of the game at each time. Constant communication between the team members is critical for Agile software development to work, this is why we have chosen software like Slack and Google Drive as well as GitHub since they allow for sharing of work in a number of different ways. Furthermore, the ability to push and pull code from GitHub and immediately open it up to test in IntelliJ is incredibly helpful when it comes to testing code that has been written by other teammates.

Overall the choices we have made regarding software and collaborative tools should heavily aid us in the development of our game. The use of collaborative tools for communication will make sure our agile structure is viable and the software tools will allow for implementation of Auber in an efficient manner.

Part B

As a group, we chose to adopt the Scrum approach to team organisation and structure. Having a Scrum approach involves three major roles for the composition of the team, these are the Product Owner, Scrum Master and Development Team. The product owner is considered the leader of the group, organising the product backlog (a series of requirements that must be met for the product to be of sufficient quality) and ensuring that the development team completes these in a timely manner. The Scrum master is crucial when it comes to making sure the values and rules, that have been established for the team, are kept intact and followed reliably to

A scrum team structure is reliant on all members being capable of cross-functionality, making sure each group member can take part in all aspects of the project, in this case all team members must be able to take part in both coding and documentation aspects of the project. This allows for the Scrum Master to assign team members to any task that may need to be completed, by doing this the team will be able to adapt to different situations and have someone working on a piece of the project at all times. For example, if there is a significant amount of implementation needing to be done, the scrum master will allocate implementation tasks to a large amount of the team, allowing for these tasks to be completed in line with whatever deadline there may be. This is applicable to our group project as the tasks within the brief can vary from implementation to documentation, meaning that everyone in the group should be capable of completing both types of tasks, the members of the group should still do what they are best at however this approach to organisation allows for work to be done where it is needed most.

Furthermore, within the team we have a variety of skills, some of us are better at documentation whereas others are better at implementation, using this Scrum team organisation allows us to get the best out of the skill set within the group, as the team members are still working at what they feel most comfortable with, this will help team morale as the tasks being presented won’t be intimidating, allowing for motivation to be higher and therefore helping the productivity of the team.

Part C

Index	What are we doing?	Priority (H/M/L)	Start Date	End Date	Dependencies
1	Create a Slack HQ, Google Drive and GitHub repository.	H	15/10/2020	16/10/2020	No dependencies as this is the first task.
2	Read brief and outline key points for project.	H	22/10/2020	23/10/2020	Dependent on index 1. Outlines must be uploaded to shared drive.
3	Choosing an engine for Auber and coding environment/programming language.	H	29/10/2020	31/10/2020	Dependent on index 2. Need to be aware of what the engine needs to be capable of and what programming language we will use.
4	Create launch window and main menu.	H	5/11/2020	9/11/2020	Dependent on index 3. Coding environment needed to begin implementation.
5	Create a draft of the map for Auber.	H	5/11/2020	10/11/2020	Dependent on index 2. Need to know the requirements of the map, e.g. how many rooms.
6	Create and implement a Game and Control Screens.	H	5/11/2020	10/11/2020	Dependent on index 3. Coding environment needed to begin implementation.
7	Create Input Processor for the game.	H	5/11/2020	11/11/2020	Dependent on index 3. Coding environment needed to begin implementation.
8	Implement a temporary sprite for	M	5/11/2020	12/11/2020	Dependent on index 7.

	the player to test movement mechanics.				
9	Set background as map.	H	9/11/2020	11/11/2020	No dependencies.
10	Full implementation of the System.java class.	H	10/11/2020	12/11/2020	Dependent on index 9, map must be implemented for systems to appear on it.
11	Retexture of map, player, systems and menus.	L	12/11/2020	12/11/2020	Dependent on index 10, 9, 6.
12	Full implementation of TeleportPad.java class.	H	12/11/2020	12/11/2020	Dependent on index 9, map required for pads to be present.
13	Create enemy class in Java.	H	12/11/2020	15/11/2020	No dependencies.
14	Room by room illumination implementation.	H	14/11/2020	16/11/2020	Dependent on index 9, map required for rooms to be present for illumination
15	Implement player movement in demo mode.	M	15/11/2020	19/11/2020	Dependent on index 9, map required for player to be moved around.
16	Implement enemy special abilities and sabotage systems.	H	19/11/2020	20/11/2020	Dependent on index 14, enemy class must be created for implementation.
17	Implement health system.	H	19/11/2020	21/11/2020	No dependencies.
18	Create Win/Loss Screen	M	12/11/2020	13/11/2020	Dependent on index 13.