Change Report

Team 15

Joe Wrieden

Benji Garment

Marcin Mleczko

Kingsley Edore

Abir Rizwanullah

Sal Ahmed

Change Report v0.6

Preface

This document is intended to be a log of all the changes made to the original documentation of Team 14, the team whose project we have taken on and are continuing.

- v0.1 Creation of document, team's formal approach to change management summary started
- v0.2 Changes made and updated for Requirements
- v0.3 Team's change management approach completed, but needs reviewing
- v0.4 Changes justified for Requirements, and changes added for Architecture
- v0.5 Changes made and updated for Method Selection and Planning, and Risk Assessment and Mitigation
- v0.6 Remaining changes justified. These could be revised. Still need to add hyperlinks to documentations on the website

Introduction - Part A

The team's formal approach to managing any changes to the requirements was aided mostly by the Scrum methodology used in Assessment 1, which incorporated the providing of weekly deliverables (the current version of the artefact of the week) and regular meetings with the customer and the two subteams (Documentation-oriented and Implementation-oriented). This allowed the customer to be included in the development process should they wish to have access and updated them on the progress thus far, keeping them in the loop. This also kept the two subteams up to speed with one another's progresses and they provided each other with information that they had researched on the different ends which would be relevant to the other subteam. Scrum also allowed the team members to keep the whole team informed about potential risks that could occur, as detailed in Risk Mitigation and Management here.

Upon analysis of the deliverable, the stakeholder could subsequently provide feedback at any point that a concern of theirs may be raised, in accordance with any changes be they due to changes in environment, user expectations (gathered through feedback in pilot operations) or even technological advancements. This would lead to any future necessary adjustments to the project to be found, logged into the regularly checked project management tool (we chose Team Gantt), and subsequently made as swiftly as possible.

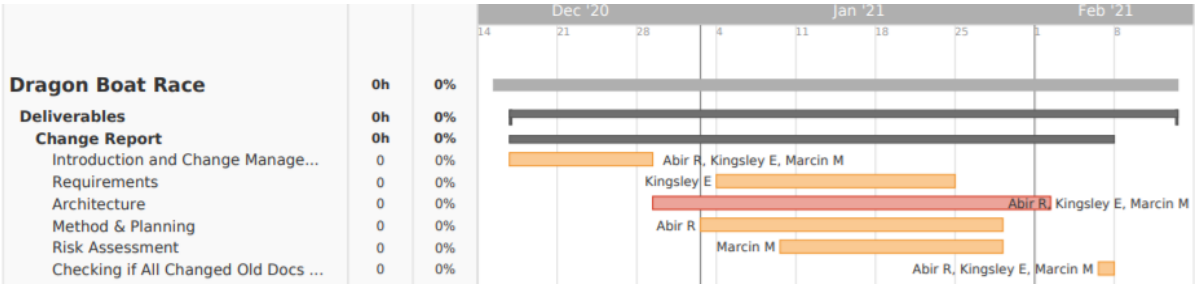
The splitting of the team into subteams allows not only for the workload to be distributed, but also for individuals to work closely with the people similarly specialising in their niches, so the changes could be carried out simultaneously on different areas of the project, and so greater quality of deliverables are produced quicker. However, interchangeability between the teams has remained possible, in order to maintain flexibility due to the distribution of the workload not always being even.

Each individual was initially allocated a change task. Within the Implementation-oriented team, Sal chose to take on implementing the new requirement UR_Save_Game, while Joe took on UR_Difficulty_ Level and Benji took on UR_Power_Ups. However, when one individual finished a task or felt they were making enough headway with it, they stepped into helping out another sub teammate who was struggling to progress by themselves, as Joe and Benji did for Sal, due to UR_Save_Game having the greatest amount of workload.

A timeline of each Deliverable change can be found below in the form of a Gantt chart. All changes to the original Team 14 were highlighted in dark blue in the Changed Assessment 1 Documentations, while the unchanged things are left in black.

Changes Made to the Assessment 1 Deliverables - Part B

Team Customer Meeting - 21/01/2021
A timeline of each Deliverable change can be found below in the form of a Gantt chart:



i. Requirements

Changed Requirement Document.
Requirements that were detailed outside of the requirements table were removed in order to avoid redundancy and keep conciseness, as well as discussions on implementation (which is irrelevant to the Requirements Engineering process).

Since the first part of the deliverable failed to provide any evidence of what approach Team 14 took, we added the specifics of the Requirements Engineering process to the original document produced by Team 14. We also provided the justifications for using the approaches used when outlining requirements, in terms of having better product quality in the long run. Since there was a lack of evidence of looking into literature and research on the Requirements Engineering process. We used specific books and articles to research what approach we planned to take as well as finding reasons to justify why we took that approach and how well it worked.

The User and System Requirements were cohesive, so we didn't feel the need to add many more, excepting the three new requirements: UR_Power_Ups, UR_Save_Game and UR_Difficulty_ Level. The System requirements were subsequently added.

ii. Architecture

Changed Architecture document.

We removed the entire draft for the UI as this only really focused on the describing aspects of the project and were of little relevance to the production of architecture.

The UML diagram was both unreadable and didn't employ strictly correct UML notation, and was also unlabelled so caused confusion as to which architecture it referred to. Therefore, we decided to create the UML class diagram such that it would be accessible for any future readers of the documentation, and so that they could analyse the trajectory of the various iterations of our project of their own accord. This new UML diagram was labelled as the class diagram which is one of the components of the Abstract representation of the architecture, for readability purposes. This diagram is also supported by the state machine diagram that we chose to embody the behaviours of the system and its components in response to actions that occurred. This provides an alternate behavioural view of our implementation plan, supporting the structural view that the class diagram presents.

Further on in the Software Development lifecycle, we developed a concrete representation of the game's Software Architecture plan, using the UML Class Diagram tool provided by the IntelliJ IDE, so that a bridge between the abstract architecture and the code was formed. This representation consists of a structural diagram representing the static features of the system. To clarify that the consistency in satisfying the requirements has been kept between the various versions of the product, justifications for how the concrete architecture builds from the abstract architecture have been added. The components of the concrete architectural diagrams' relation to system requirements (in turn derived from user requirements, so we are making sure that we are still following through with the requirements we elicited) are justified under "Justification" in the Architecture document. The tools used as well have been specified in order to give credit due to the tool we made use of, as well as the justification of our use of the selected modelling notation (UML).

We also added key decisions such as the object-oriented approach we planned to use for the implementation, as these would influence our architecture greatly in that we would appropriately lay out our architecture in a format suitable to that approach. We also added the references to the articles that influenced our design decisions. We also added the systematic justifications for every component of our architectures, in order to show how the architectures directly relate back to the requirements.

Both of the architectural diagrams we produced shows the states (with correct UML notation and self explanatory names) that the game will consist of, and we checked that every component of both models relates back to the user requirements, for the concrete diagram through the abstract architecture. This traceability is all to ensure that we were not diverging from the requirements set out by the stakeholders.

We added justifications on our abstract and concrete architectures to show that investing time and effort in producing them improved the quality of our end code, as opposed to having to make more costly decisions upon further thought and coming across conflicts, had

we delved into a less higher level implementation first. The abstract diagram allows all parties involved to be brought together and understand what is going on in the project. On the other hand, the concrete architecture is much more closer to the implementation of the code. This would make sure that any reader of the document's understanding, be it coder or stakeholder with varying levels of technical experience, would be able to understand the plan. Thus any qualms could be raised by any member affiliated with the production before we moved onto actual implementation, when changing things would be more difficult and costly.

iii. Method Selection and Planning

Changed Method Selection and Planning.

Our method of approaching Assessment 2 is detailed in the 3.4 Updated Plan for Assessment 2 section of the changed Team 14's Method Selection and Planning deliverable. The updated plan for Assessment 2 can be found at the bottom of the Weekly Snapshots section on the Assessment 2 page here, or alternatively the image can directly be accessed here.

We want to amend the Gantt chart embodying the project plan, as it only plans for Assessment 1 as opposed to the whole project. Having a Gantt chart which embodies a plan for the whole project would make it easier for any teams that inherit our project in the future, as well as guide us through our project in Assessment 2, especially in relatively short time constraints which places an emphasis on our being organised. This Gantt chart also left little room for the iterative nature of the agile Scrum method, so we felt this rather linear planning structure needed to be amended in order to support our way of carrying out the project. The tasks labelled as present were quite high level, and we felt these could be broken down into more modular and low level tasks. However due to our time constraints we may not be able to modify the Gantt chart we have produced for Assessment 2 to include the Assessment 1 project timeline as well.

We provided more research into the advantages of the method chosen. Additionally, we added the tools that were used to generate the project plan Gantt chart, create the architectures, implement the program (IDE and frameworks) and manage the project as a whole. We also justified our using the tools/method we did, after considering and dismissing other alternatives, to make it certain that they were suitable for our means and would maximise our productivity. The tools were also of importance as they allowed everyone to adhere to their own responsibilities.

We were more specific about team organisation roles, such as allocating the scrum master, product owners, the development team and the team coordinator. This was so that we had people with clearly defined responsibilities in regards to keeping the sprints on track, as well as taking care of the product's gradual iterations towards the final product respectively. Additionally, we set the role of the team coordinator, which needed to be clarified as there was a need for a bridge between the two subteams to exchange information that would be of vital importance to the other.

We added a project plan that was built upon the one provided by Team 14 (and resized the former team's to make space), and took into account regular communication between team members, as well as providing flexibility to the team, in the way we planned and managed our project. This is justified in the project evaluation which was also added to evidence how well our approach had worked in unforeseen circumstances

iv. Risk Assessment and Mitigation

Changed Risk Assessment Document

We added a Preface and version numbers to the document to keep track of what had been done and what hadn't been done in the documentation, on top of the use of Team Gantt. The Severity and Probability columns in the main risk table have been changed to "Low - Medium - High" system. This coupled with the colour coding, makes the table easier to read, saves space and is easier to understand. Thus, the assessment of the risk register in the weekly reports can be quick. Sentences lacking in clear wordings and grammar were also reworded to make them more clear to the reader.

The "Situation" and "Influence" columns have been merged into one column named "Description". This is because the two columns were very similar and merging them saves space and improves readability. Tables and columns were also resized to make the tabulation more readable. Vague wordings were changed to be more concise and specific, in order to make it to-the-point and meaningful.

An "Owner" column has been added. This will ensure that every risk will be monitored by at least one member of the team, and that, should one's likelihood increase, the team will be notified of it as soon as possible by the member in charge of that risk. More risks were also identified and the amount of risks in the risk register were doubled. This saves us time and effort in the future from being wasted, if we possibly encounter a problem that was not considered and tabulated before.

Risks were grouped into different categories depending on their similarities, to improve readability and organise the risk table in a more appropriate format. This was through the "Type" header being added. This allows us to group risks, meaning that certain areas are easier to maintain. Also, assigning an owner is more effortless as people can pick a risk from the "Type" they have the most experience with.

There was no justification of the risk format. They should have explained how the tabulated risks and the appropriate headlines actually help them deal with risks. This would have allowed any reader of the document to understand the reason we have spent time identifying potential risks and coming up with plans to avoid or mitigate them. To account for this, we added an explanation and justification about the headlines of the main table. Even if this may have taken up greater portions of our time, this system of careful risk management allowed for us to work well together as a team and provide deliverables of good quality on time as we had prepared for any risks that may have come beforehand.