Change Report2

Cohort 1, Team 1:

Connor Blenkinsop
David Luncan
Emily Webb
Muhidin Muhidin(MO)
Sam Laljee
Sooyeon Lee

Change Management Summary of Formal Approaches

To deal with many changes occurring within the span of a software development life cycle (SDLC), we felt a strong need to find a robust change management structure to handle them properly. Following **Lewin's change management model** of 3 stages: Unfreeze, Change, and Refreeze, our team decided to implement these formal approaches for the change management. Lewin's model was chosen due to its simplicity and intuitive concept, but we also took account of other theories such as Kotter's change management theory to finalise our change management process.

Below are the steps of the **change management process** we agreed to follow.

1. Create a change request (CR).
2. Review and discuss the CR with available members of the team relevant to the task.
3. If the CR is approved, plan how to change the components of the project.
4. Implement necessary changes and review its performance, whether the change has been effective or not.
5. If the change was unnecessary and ineffective, revoke it to the state before.
6. If the change was necessary and effective, close the CR.

This change management process was used to manage changes of all deliverables, documentations, and code. Whenever the member of the team felt the need to 'Unfreeze' a certain aspect, the individual would start from creating a change request and notifying the team. We made sure that thorough communication has been done in this step by having a group meeting and messaging others constantly. Especially with the codes, when someone was attempting to make a major change to the code itself or classes, all attempts were notified to the team and were able to be revoked once the team decided the change was unnecessary (referring to step 5).

For the change management of deliverables and documents, the process often differed depending on the urgency of change and the members responsible for the task. For example, as only one person was mainly in charge of the website, if there was a need for change regarding that deliverable we did not undergo group discussions for minor changes (step 2). The person in charge of the website had the authority to take on actions wherever it felt necessary. Also, when the change had to be done urgently, the person could override step 2 and 3 and directly implement necessary changes.

Now, we will proceed to introduce a brief explanation and justification of any changes made to Assessment 1 deliverables from the previous team in the next section. We included URL links to the previous team's deliverables and their revised versions. We also agreed to follow the same format when possible, by explaining our changes into 3 categories: **Removed, Changed, and Added.**

I. Requirements

Team 6 original deliverable: https://tomnicho.github.io/eng1-team-6/Req1.pdf
Team 6 original requirements: https://tomnicho.github.io/eng1-team-6/requirements.html
New Full Requirements:
https://eng1team1.github.io/assessment1/team1-deliverables/Req2-Full.pdf
New Deliverable Requirements:
https://eng1team1.github.io/assessment1/team1-deliverables/Revised-Req1.pdf

While the explanation and theory behind Team 6's requirements largely matched our own, their choice of layout, specifically in regards to which requirements are displayed in the deliverable and which are omitted (in order to be displayed in a larger document on their website) was very different. Due to the large number of requirements and risks that would be created when Assessment 2 requirements are added, we chose to mostly keep the layout, but change what requirements are displayed/risks are discussed. We were surprised that the team chose to include some requirements not needed in Assessment1, that we would have otherwise added. For this reason, URs such as UR_GAME_CURRENCY will not be included. Changes made to the deliverable can be split into Removed, Changed and Added. Here is a summary for each area:

Removed:

1. UR_NPC_PIRATES and FR_NPC as the assessment brief now asks for "combat with other ships" we do not want it to be required that NPCs are all stationary
2. FR_RESTART_CLEAR more a NFR, this may also not always be the case and may be in direct contradiction with NFR_SAVING

Changed:

1. SSON: team 6 used their own paragraph write-up for it. Though we appreciated it's conciseness, we felt there was no better SSON than the assessment brief that was released on the VLE. It is longer and more detailed but contains an accurate vision of the application for all stakeholders involved.
2. Paragraph on the criteria by which requirements to be displayed in the deliverable: this was different than our team's approach to this task - while we broady kept the layout set by Team 6, we wanted to include more requirements in this and also change the basis by which a requirement is displayed and/or discussed in the deliverable; this new paragraph reflects this.
3. UR_PLATFORMS and UR_SCALABLE were merged and are now called UR_USABILITY as per our Asessment1: they serve the same purpose and UR_USABILITY is a more succinct explanation of this, exactly as given to us by our client
4. Updated the priority of several URs as these are now Shall requirements under assessment 2 (e.g. UR_NPC_PIRATES, UR_AI_PIRATE
5. Fit Criteria for several NFRs: some were not specific enough and so we chose a more quantifiable metric, e.g. NFR_HARDWARE_REQUIREMENTS previously required "Runs well without issues on a department laptop" this has now been changed to "Any response should be within 0.5 of an input"
6. What risks, URs and NFRs appear in the deliverable itself

Added:

1. A number of NFRs:
    a. NFR_TIMING, in response to UR_TIMING (previously UR_GAME_LENGTH)
    b. NFR_FILE_SIZE as no restrictions on this had been layed out previuosuly
2. FR_SHOPPING on top of the UR_GAME_CURRENCY as the assessment brief, our SSON calls for this

3. New FR and URs required to complete the brief for assessment 2, that were not previously needed:
    a. UR_SHIP_COMBAT -> FR_NPC_COMBAT
    b. FR_SHOPPING
    c. UR_NEW_OBSTACLES -> FR_POINT_EARNING_WEATHER and FR_BAD_WEATHER.
4. New FRs, URs and NFRs for the additional criteria to complete assessment 2 that were released closer to the starting date:
    a. UR_POWERUP -> FR_POWERUP
    b. UR_SAVE_STATE -> FR_SAVING and NFR_SAVING (this counted both as a FR and as a NFR due to it's practical implications on the user, as well as indirect applications on the system state behind the scenes)
    c. FR_GAME_DIFFICULTY, added to the already existing UR_FIRST_TIME_PLAYER
5. Risk added for FR_GAME_DIFFICULTY, as there is potential for compromise on URs such as UR_FIRST_TIME_PLAYER
6. FRs to the deliverable sheet itself

II. Abstract and Concrete Architecture

Team 6 original architecture deliverable: https://tomnicho.github.io/eng1-team-6/Arch1.pdf
New full architecture deliverable:
https://eng1team1.github.io/assessment1/team1-deliverables/Arch2.pdf

Removed:

1. Removed paragraph Explaining that they separated the entities and world class as we are no longer using their abstract and it was not necessary to explain if we used the entities and world class separately from the beginning.
2.Removed paragraph explaining that they used an xml file as this is now included in the abstract architecture so included in that explanation.
3. Removed final paragraph explaining about adding calculations class as this would now be included in the abstract architecture.

Changed:

1. Changed the abstract architecture to be the structure of the original concrete architecture but including the proposed new classes, also not displaying all the variables and methods to allow it to easier see the structure and the relationships between objects. Also used two diagram to make it less confusing.
2. Changed the paragraph about the abstract architecture to represent the description of the new architecture including explaining that we will use a xml file to load entities
3.Changed the concrete diagram to edit and implement that are required for the second brief namely Powerups, Obstacles, GoldShop, Pause and Difficulty. Also adding an ai factor into the world class allowing the Npc boats to follow the user and attack them

Added:

1. Added explanation to why the ai class was scrapped in the concrete
2. Added explanation to how the pause menu and Goldshop menu was called in game as it was called when the game was still running
3. Added explanation for adding class Difficulty
4. Added a table that goes through some requirements and what method implements it

III. Methods and Plans

Team 6 original deliverable: https://tomnicho.github.io/eng1-team-6/Plan1.pdf
Team 1 new deliverable:
https://eng1team1.github.io/assessment1/team1-deliverables/Plan2.pdf

---

We chose to use an agile methodology for this project as agile software development is adaptable and flexible, providing frequent deliveries of software. This did not require any change as the previous team has also used an agile method.

Removed:

The largest change to team organisation was the removal of a team leader. We found this to be unnecessary as our team functioned well with shared leadership, and responsibility could be evenly distributed amongst us by appointing section leaders, responsible for a deliverable. Alternatively we decided to appoint a team member who was not too closely tied to any one part of the project so that they could be flexible and provide extra manpower to any part of the team that was struggling to stay on track.

Changed:

1. We decided not to follow a waterfall approach as initially proposed, as we thought it would be inefficient to attempt to complete each stage one after the other. Changing this allowed us to have small groups of people working on different sections at the same, that could each be reviewed and improved.

2. We changed the **communication tool** to Facebook Messenger. This worked well for our team as we all had accounts and had previously been using a group chat to communicate between meetings. We also used Zoom when we couldn't attend a meeting in person as it allowed any missing members to still take part in the meeting, using an app they had previous experience navigating.

3. We added several other tools. The first being **LibGDX**, a framework within which we developed our Java code, that catered best for our purpose. In order to create diagrams, we added the tools Draw.io and Microsoft Excel, for architecture and planning respectively. We chose to continue to use Google Drive and GitHub so there was nothing to change here.

Added:

1. A new Gantt chart was created to set out the expected timeline of Assessment 2 with the new deliverables. We chose to use a simpler Gantt chart to ensure the information was clear and readable, condensing from a daily plan to a week-by-week basis. However, we were able to update our Gantt chart each week with the percentage completion of each task and track expected duration against the actual time spent on each section.

2. We increased the number of meetings held each week as we thought this necessary to keep us on track and to ensure we were properly communicating any problems we were facing. In alignment with the previous plan, we planned to work over the break but left free weeks at the end of the project in case we fell behind schedule because of this.

IV. Risk Assessment and Mitigation

Team 6 Old Deliverable Risk Assessment and Mitigation:
https://tomnicho.github.io/eng1-team-6/Risk1.pdf
Team 6 Old Full Deliverables: https://tomnicho.github.io/eng1-team-6/deliverables.html
Revised Risk Assessment and Mitigation:
https://eng1team1.github.io/assessment1/team1-deliverables/Risk2.pdf

---

From the risk register created by Team 6, we figured that it required some changes. First, additional types of risks could be introduced to specify the categorization. Second, there were miscellaneous, outdated, or irrelevant risks that should be removed or rectified. For example, R5 is very unlikely to happen whilst R11 and R12 can be combined into one risk item. Third, we should redefine the owners of each risk item to the current team's members. Fourth, risk descriptions and mitigation tactics need clarification and in some cases, appropriate changes. Last, the table itself should be refined to look better and help understanding.

On the other hand, some features of the previous risk register shall continue to remain. We will keep the 3 levels of likelihood and severity; Low/ Moderate/ High. Risk ID will maintain the same format as well.

Removed:

1. We removed some of the previous risk registers' risk items: R3, R5, and R10. These risk items were irrelevant to our project.
2. R10- 'Member's discord account gets hacked' won't be our team's concern as we don't use discord for communication.
3. R3 and R5: These were risk items related to team members' availability during the holidays and their intention to keep studying at the university. We agreed that the probability of anyone taking a leave of absence would be extremely low, so we decided to remove it.

Changed:

1. We reordered the risk items and redefined most of the risk descriptions. Since we use Facebook and Google drive for communication, we changed the technical tools specified in the previous risk register into the ones we are utilising.
2. Risk likelihood and severity columns have been reassessed, and became

colour-coded to visually aid understanding.

3. Owners of risks have been reassigned according to delegated tasks for each member of the team.
4. In some cases, duplicate risks were combined into one risk item. What used to be previously R11 and R12 is now represented as R5. This measure was taken to increase the clarity of the risk register and avoid any inconvenience.

Added:

1. New risk type, **Technology**, has been introduced. Technology type risks are related to either the software or hardware side of software engineering. Adding a new risk type led to adding related risk items such as R5, R6, R7, R8, and R9.
2. We also added a new risk item, R4. Based on our own experience working on the 1st assessment, we learned that adequate scheduling and task delegation is as important as product development.
3. Finally, R10 became a new addition to the risk register as there is always a customer related risk throughout the software development process.