Change-Report

Team 6
Sanjel Sadikaj
Adam Hagan
Tudor Ciobanu
Na Tang
Armintas Zadeika
Liu Zhang

Change management

Change management is a process of analyzing the advantages and costs of changing aspects of a project. A formal process of change management includes the processes of checking, costing, and approving. We took this formal approach as it would allow us to efficiently select changes to make to the project, but also justify our decisions in doing so due to the linear nature of the approach. Below you can find what occurred at each process of adding a new change to the project along with justification of why the change occurred.

Checking:

Checking is the process of identifying any changes that need to be made in the project. Changes were identified by viewing the previous project files and comparing them to the tasks we have set out to perform in the new project.

For each change identified we first located where the file the change would occur is located. We then proceeded to view the whole document to see how the change would affect the document as a whole. Finally, we then proceeded to update the document and remove or add any information that was required for the change. After this process was completed we noted down the change (shown in tables below) and the location of where the change was.

Costing:

1. Every change identified in checking is then analyzed regarding it's benefits and negatives. The cost of time, the priority of the change as a whole, and the benefit of the change in relation to its costs all are discussed regarding a change.

2. The influence the change will have to the whole project is also considered. For each deliverable, the changes should optimize the deliverable to the requirements of the project's criteria. If the change does not optimize the deliverables then it is an overall negative impact and will not aid the project.

For documentation and code, changes to code must be documented to the architecture. Before committing any changes to the code it must be thoroughly tested to ensure that the rest of the code is not affected.

Approving:

After completing the cost analysis of a change, it will then undergo the approving process to determine whether it will be approved as a change. This comes in the form of two scenarios:

1. When approving changes, all documents must be modified accordingly to the change to ensure consistency (information about where the change is should be included to allow retractability).

2. If the change was seen to show failures in the cost analysis, the change is not approved and removed as a change. Therefore, the change will not occur.

Management of previous project files:

Small changes of deliverables(such as minor errors in documents) should be implemented immediately and not recorded due to the insignificance of minor details. Previous version(s) of the documents and code should be saved, and can be viewed as a progressive comparison.

Requirement

Change	Explanation	Justification
Removed FR_MINIMAP 2.0.8	The previous team had listed a requirement that there would be a minimap on the screen during the race - this requirement was not completed, and we removed it.	This requirement was removed from the table as we found no evidence that the customer had requested this feature.
Removed UR_CHANGE_RESOLUTION 1.2.2	The previous team had listed a requirement that it should be possible to change the resolution of the game window at any time - this requirement was not completed, and we removed it.	This requirement was removed as the customer explicitly stated that this was not a requirement.
Added requirement of being able to save the game. User Requirement 1.1.2. Functional requirement: 2.0.11	As part of updating the project 3 more requirements need to be added. One of which we decided was the ability to save the game and also to be able to load the game.	As saving is a new feature it also will require to be a user requirement and a functional requirement of the game. Saving allows the user to play the game and stop the game at any time they wish.
Added requirement of the game needing power ups. User requirement: 1.0.3 and functional requirement: 2.0.12	As part of updating the project 3 more requirements need to be added. One of which we decided was allowing the user to pick-up power-ups which float along the river.	The feature of being able to pick up and use power-ups will require their own set of requirements in the requirements document, hence we added their respective user requirement and functional requirement.
Added requirement of being able to change the difficulty of the game. User Requirement: 1.3.2, Functional Requirement: 2.0.13	Another one of the 3 additional requirements is the feature of being able to change the difficulty of the game. The difficulties of the 3 options are easy, medium and hard.	The feature of changing the difficulty requires it to be a requirement in the requirement documents and therefore a user requirement and functional requirement was added to relate to the feature.
Removed constraint requirement description	Removed the description of how the requirements need to be constrained to the stakeholders and customer.	The description was removed as it wasn't necessary as each requirement implicitly linked to the constraints. Removing the description also allowed us to conform to the page limit of the documents.
Modified FR_OBSTACLES 2.0.3 testing criteria.	Changed the description success criteria for the requirement as it doesn't match the requirement.	This is an important change as it can cause a clash of requirements and lead to inconsistency. Inconsistency is required to be avoided as it may lead to implementing incorrect features or testing the wrong aspects of requirements.

No other changes were added to the requirement, other than the changes seen above, as all requirements were implemented in the project and successfully met. Therefore changes were rare to occur as the requirements were met. Any additional requirements that were added were recorded in the table above. Changes made to requirements naturally caused changes to the architecture which is seen below.

Link to changes:
https://mrpoketes.github.io/mrpoketes.github.io-ENG1-Team6-Website2/docs-new/req2.html

Architecture

Change	Explanation	Justification
Modified the abstract architecture diagram	Extension of old content	As we have gotten new requirements to implement the game architecture also changes, thus it had to be modified.
Added a class diagram for the abstract architecture	Extension of old content	The state diagram was not enough to explain the abstract architecture, it was abstracting the main logic of the game.
Modified the concrete architecture diagram	Extension of old content	As we have gotten new requirements to implement the game architecture also changes, thus it had to be modified.
Added information about InfoUI, ChooseDifficultyUI, OptionsUI.	New UI elements.	As the previous team never implemented the InfoUI and OptionsUI, which were needed according to (UR_CHANGE_SETTINGS, UR_DISPLAY INSTRUCTIONS), they had to be implemented. Same goes for ChooseDifficultyUI (UR_CHANGE_DIFFICULTY).
Added information about MenuUI showcasing a Continue button	New requirement	A new requirement (UR_SAVE_GAME) had to be implemented and as it is an important piece of the architecture, information had to be added.
Added information about pressing ESC to pause the game	New requirement.	As this is not a requirement that was explicitly said, it ties with (FR_SAVE_&_LOAD) requirement.
Modified information about Map class	Refactoring of code	We decided that the Game class does more process then it should, thus we have moved a lot of the logic to from Game class to Map class
Added information about power ups	New requirement	A new requirement (FR_POWER_UP_ITEMS) had to be implemented and as it is an important piece of the architecture, information had to be added.
Added a justification as to why we added a new class diagram to the abstract architecture.	Improving the architecture document	The architecture abstracted away the important logic of the game.
Modified information about the Game class	New requirements and refactoring of the game	As said, a lot of logic has been moved from Game class to Map class. Besides that, a save and load feature was implemented (FR_SAVE_&_LOAD).

Link to changes:
https://mrpoketes.github.io/mrpoketes.github.io-ENG1-Team6-Website2/docs-new/arch2.html

Methods and plans: software development methods and tools; team management approaches; plan for Assessment 2 (you must include the precise URL of the updated plan)

Change	Explanation	Justification
Added updated Gantt chart of assessment 2 and assign task priorities and critical path of project	Extension of old content as this is a new team previous to the old team we need to update	Important information missing and continuation of the gantt chart was required in order to show allocation of tasks and time management.
Maintained the majority of the tools used for the project by the previous group but added an additional tool as well.	The tools used to work on the project were not changed.The use of GitHub,Git,LibGDX and Github pages has been adopted. However, one new tool was added which was Intellij.	This was done in order to ensure a smooth transition and take-over from the project. Given the time constraint it is most effective to continue with the existing work structure rather than adapting the code to a new environment. IntelliJ was added however as it helped in the process of pushing and pulling code from our repositories.
Changed the software development method approach from SCRUM to Feature-Driven Development	The previous group used the SCRUM approach to software development which we weren't familiar with. Consequently, our Feature-Driven Development was adapted to the project in order to complete the new project goals successfully by the allocated date.	We have a small team that focused more on independent work especially as a consequence of the Covid-19 epidemic.This was the best approach for us to be able to focus on each required feature without overcomplicating the adjustment to the previous groups' SCRUM approach.
Team structure and dynamic was retained from our own project as opposed to adapting the previous group.	A diplomatic approach to decision making has worked best for our group.The biggest drawback is the lack of a leader and thus someone in charge of accountability. However, meetings were used as progress checks on tasks with the customer's intervention used if needed.	By the end of the first assessment our team achieved an efficient workflow with our decision-making style.To continue this for the second assessment this was maintained as it had already been proven somewhat successful in the past.
Communication tools were maintained and discussed in the report as to why they are maintained.	Discord was used as the main communication avenue and Google Drive storage space for deliverables.	Use of these tools was further expanded upon from the first assessment, with the higher efficacy this was the clear choice.

Link to changes:
https://mrpoketes.github.io/mrpoketes.github.io-ENG1-Team6-Website2/docs-new/plan2.html

Risk assessment and mitigation: approach, presentation, risks, mitigations

Change	Explanation	Justification
Removed risk; “player controls don't work” Id 10	The risk of having the player controls not working has been removed as they are fully functioning features in the current project. The likelihood of the controls being broken is highly unlikely as controls do not need to be modified.	The risk was removed as it’s no longer required for the project and therefore does not exist as a risk we could encounter in our current project.
Added major risks which were not in the previous group files, the changes have the following ID’s: Id 5,6,7,10	There were a few major risks missing in the risk assessment report therefore it needed updating to add the risks.	Major risk analysis can help team members to react with serious risk. Without displaying or showing awareness of risks severe issues could arise in our project, therefore the risks were added.
Integrated some risks together **[Id new(list of old Id that are included)]** Id5 ← (5,8) Id7 ← (6,7,10) Id6 ← (12,14,15,16) Id10 ← (13)	Some risks were closely related and therefore have been merged together to improve readability and organisation.	Every bug or misbehavior due to poor coding or poor specification is not a risk, therefore some previous risks were merged together for it to classify as a full risk to the project. This also helped our current team figure out risks we may encounter.
Changed the description of mitigation for the following risks: Id 1 Id 2	The previous mitigation did not explain how to reduce the risk. It only referred to how to reduce the probability of the risk happening.	mitigation is about reducing the severity of the effects, not to try to prevent occurrence of risks completely. Therefore changes were made to id 1 and id 2 as their mitigation description did not match the correct form of mitigation.
Extend the content of risk description and mitigation detail Id 1	Added more detail to the description of the risk to improve the understandability of the risk and its mitigation.	This can help people to understand what kind of situation the risk could possibly occur in and how to mitigate it.

Link to changes:
https://mrpoketes.github.io/mrpoketes.github.io-ENG1-Team6-Website2/docs-new/risk2.html