Change Report

C3 Group 6
Team WHNI

Aaratrika Das
Arkoday Roychowdhury
El Foster
Hanna Pieniazek
Lewis Keat
Matthew Baghomian
Matthew Ford
Subham Magar

2. Change Report [23 marks]:

a) Briefly summarise the processes, tools and conventions your team used to plan, make, keep track of, and review changes to the Assessment 1 deliverables, documentation and code you inherited from the other team. (3 marks, ≤ 1 page)

b) For each of the following items, include a brief explanation and justification of any changes made to the other team's Assessment 1 deliverables (pointing to specific sections of the deliverables that have been added/updated). Include the precise URLs of the original (i.e. the other team's) and the updated versions of the deliverables. The updated deliverables do not need to adhere to the page limits for Assessment 1. If there is no change to report, please state and justify why no change was necessary.
(Maximum 5 marks per item, to a maximum total of 20 marks; ≤ 8 pages in total with ≤ 3 pages per item)

a)
Plan:
    - Group meetings + google docs comments
Keep track of:
    - Implementation -> github commits
    - Others -> Google docs comments and history
Review:
    - Google docs version history, github commits

In order to plan changes made to the Assessment 1 deliverables from team 7, our team held group meetings to discuss the potential changes to make for the deliverables and using google docs comment feature to suggest potential amendments to changes made. This was an efficient way as it allowed team members to make suggestions from home if they couldn't make the group meeting and allowed us to make suggestions in our own time.

To keep track of changes to the implementation, the team used different tools for the code and the documentations. For the code, we used GitHub and made frequent GitHub commits to have a history of all changes made to the code. For other deliverables, google docs comments and update history were used in the same way as github commits. This allowed us to keep track of changes made easily as we could make changes and allowed other team members to be notified of the changes without having group meetings.

To review changes, the team used Google Docs version history and GitHub commit logs. Before finalising the deliverables, we checked through the history to make sure the changes matched what we planned, we also had regular group meetings discussing what we had changed.

i. Requirements

Original URL

https://docs.google.com/document/d/1ngYajAJN6rmtBGfGmcLhWwxpPHrspHE9K0e9XBzgVcl/edit?tab=t.0

Updated URL

https://docs.google.com/document/d/1yNOgKQyDuBPPPKUZYJAjrtk5uo2KJk7RQPjG0feBR1Q/edit?tab=t.0

The requirements gathering process has been presented as a table as the previous diagram was hard to understand as it was not clear what answers linked to each user requirement and some of the text was hard to read because of how small the text was. Constraint requirements have also been identified in consideration to time constraints and catering to a wide range of desktop devices.

Changed UR_MAP from several maps for the user to choose from to a single map that the user can play as there is only one map that the user can play. Reworded UR_PROGRAMMING_LANGUAGE to add clarification that libraries can be used but they have to be in java. Added UR_RUNTIME, UR_CLEAR_DESIGN and UR_GAME_DESIGN in order to better and more precisely assign functional and non-functional requirements to user requirements. Added UR_LEADERBOARD and UR_ACHIEVEMENTS as the product brief was updated and a leaderboard and achievement system needed to be in the game.

Changed NFR_RUNTIME to FR_RUNTIME, NFR_SYSTEM_RESTRICTIONS to FR_SYSTEM_RESTRICTIONS, and NFR_LANGUAGE to FR_PROGRAMMING_LANGUAGE as they are functional requirements and not non-functional requirements as they are important in the functionality of the game.

Added NFR_TIMER_ACCURACY and NFR_COLOUR_BLINDNESS in order to make sure that the game is more fair especially for people who have visual impairments such as colour blindness. Removed NFR_GAME_TIME as the requirement was already covered by other requirements and so was unnecessary.

We also changed FR_EASY_DIFFICULTY to FR_DIFFICULTY, as the original requirement implied that the game's default difficulty was fixed to easy and that the user would not be able to change it throughout the game. By revising this requirement, we made it broader, allowing the game to either give the user a choice of difficulty modes (e.g. easy, medium, hard) or to integrate elements of varying difficulty within the gameplay itself.

Within FR_NEGATIVE_EVENTS we also changed the amount of negative events to five from two as that is what was asked for within the brief.
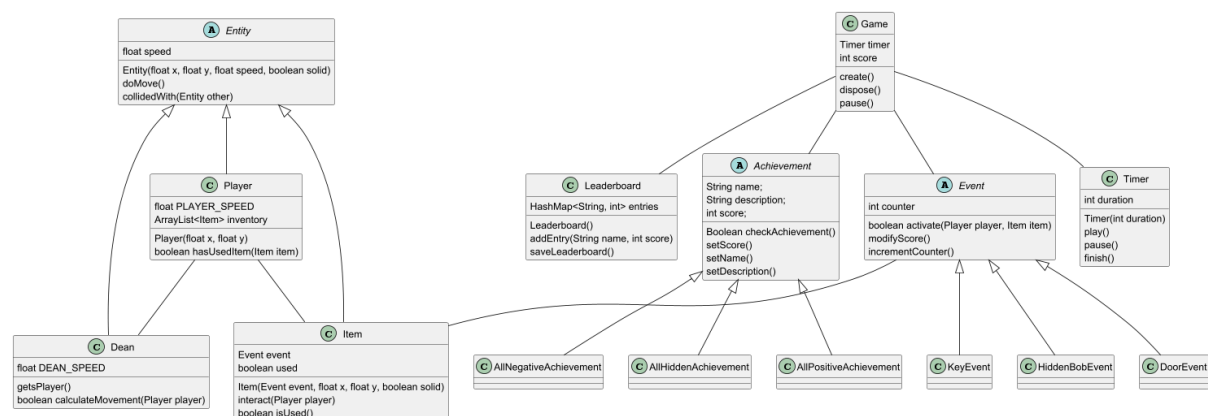
ii. Architecture

Original URL

https://docs.google.com/document/d/1FPW2JIYeeZot3kfPM1KW8omuhrwMYhmw0hsKkVBPnA4/edit?tab=t.0

Updated URL

https://docs.google.com/document/d/1tZj-xju2rxV_Ka-Xd3AfEI7jmZZW18TLvkw0v1DEn24/edit?tab=t.0
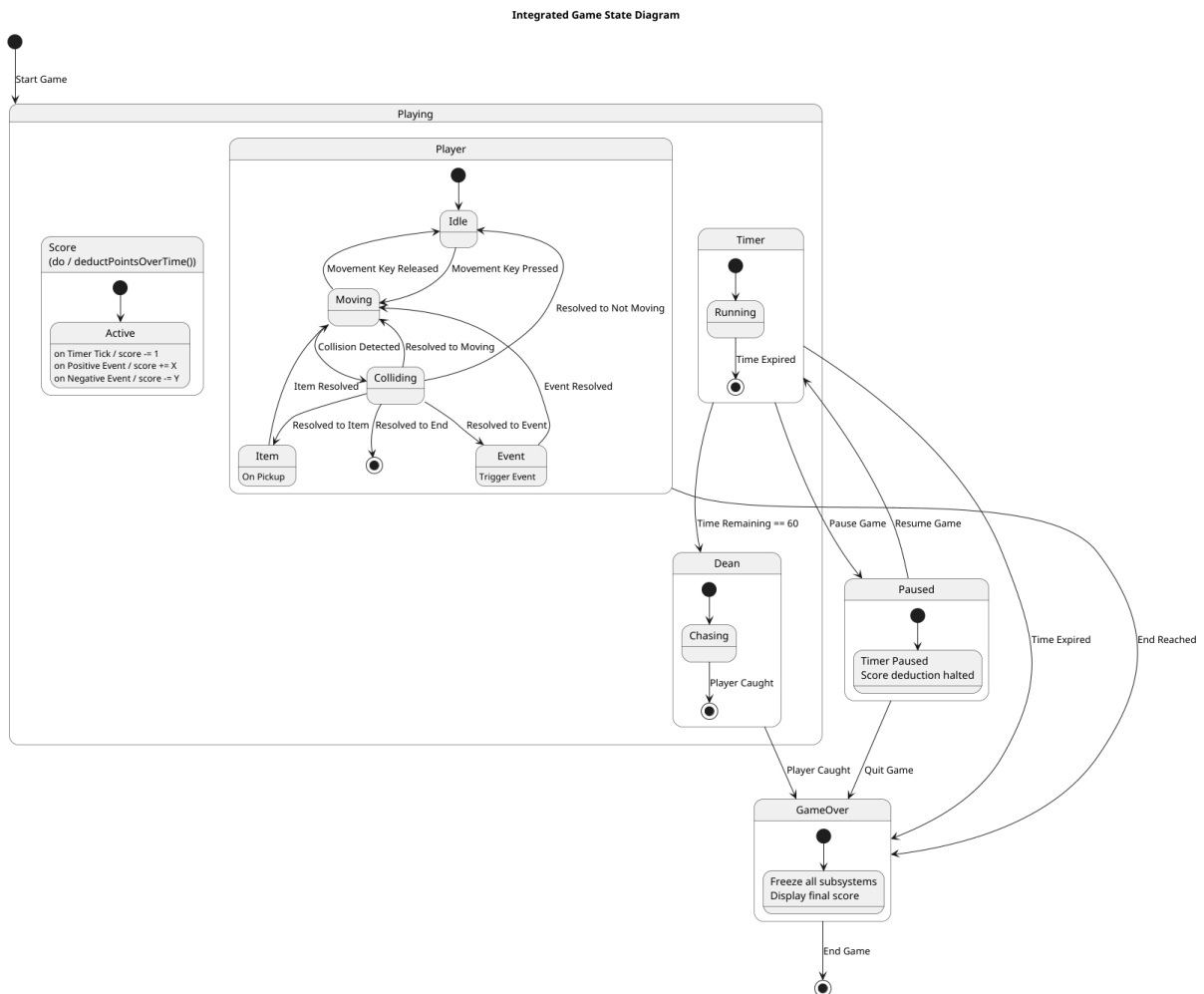

The architecture design of the classes has been substantially altered because the original diagram was too abstract to be useful and more details were needed in order for it to be useful for future development.

The **Entity** class was made abstract so that the **Dean, Player and Item** can inherit the **Entity** class as they share functionally as they are all entities that share functionality such as collision.

The **Event** class was made abstract so that all of the events could inherit from the class as they are related as they all modify the score and increment the event counter. Additionally inheritance makes sure that all of the required functions are implemented.

The **Leaderboard** class was added to the architecture design because of the new functional requirement (Insert leaderboard requirement) meaning that the leaderboard had to have the ability to have scores added to it.

The **Achievement** class was added to the architecture design because of the functional requirement (Insert achievement requirement) meaning that achievement had to be gained when the player did certain actions. The **Achievement** class was made abstract as all the achievements would share similar functionally with each having a name, description and score associated with it. This reduces code duplication and forces these attributes to be present for each achievement.

Integrated Game State Diagram

The game state diagram has been altered substantially to make it more clear how the game is intended to function. This was done by renaming some of the transitions for increased clarity, simplifying the **Timer** substate, altering the **Dean** substate, and making the **Player** substate more accurate and complete.

The **Player** substate was altered by adding an end point for if the player reaches the end of the game. Additionally the **Item** state contained both event behaviour and item behaviour and so the event behaviour got separated into the **Event** state. The "startEvent()" transition was removed as its purpose was ambiguous. The item collision transition was moved to originate from the colliding state as item collision and collision detected are not mutually exclusive.

The **Timer** substate had the expired state removed as it was unnecessary and made the substate more complicated than needed.

The **Dean** substate had the "reachGoal()" transition removed as its purpose was ambiguous and was replaced by an isolated starting point. The starting point leading out to the player substate was also removed as its purpose was ambiguous.

The transitions between substates were improved as there was no behaviour shown for winning the game so new transitions were added from the **Player** substate. Transitions were moved so that they pointed directly to the relevant substates contained within the **Playing** state for additional clarity.

iii. Method selection and planning

Original URL

https://docs.google.com/document/d/16SH8_Rr24bEfgM3DNTOfwcqot6sYGhwGdArPDcUHDCA/edit?tab=t.0

Updated URL

https://docs.google.com/document/d/1Z0Jcvfufvq3DDoq_0qQQEymT1WagHX3yLTOEGjdIDfE/edit?tab=t.0

Firstly, within Development tools there was the inclusion of PlantUML to help with architectural modelling. While the original architecture deliverables by team 7 utilised PlantUML, the original method selection and planning did not include it. This has now been updated in the updated method selection and planning to ensure consistency across the project documentation.

Another major change that was made to this document was the Systematic plan for Assessment 2. We followed the same structure that the group before us had already implemented, but changed it to follow the brief from Assessment 2 and what we wanted to achieve by the end of this. This plan breaks the work into iterations, tasks with dates, dependencies, and priorities to ensure all assessment requirements are met.

We also added a section outlining the software methodology that we used within our project. By defining this section it helped explain what kind of structure was used within the project. This also helped us to improve collaboration and communication and was a crucial part within our planning as it kept the group organised and up to date.This framework also helped us to deliver documents and software more efficiently.

The document as a whole has also been changed by removing unnecessary details and repetitive points that do not need to be mentioned multiple times. Multiple grammar mistakes have been corrected.

iv. Risk assessment and mitigation

Original URL

https://docs.google.com/document/d/1bjvXn3E_nOmSWOLkwJRjOViVbE1ufPBUpZU4MZObV0s/edit?tab=t.0

Updated URL
https://docs.google.com/document/d/1tux0f9n2CSBk49_rwR9vWetEkbWTcCkVkYtS9VCUILY/edit?tab=t.0

General mitigation and avoidancy strategies were implemented and discussed, something that was omitted from the original risk assessment report. Such as methods to reduce the bus factor. More detailed risk management process discussion, delving into further details. The introduction to the risk assessment has been expanded on to give further details and procedures our team used during the project.

The colour coding scheme from the original risk assessment has been kept because it provided an immediate recognisable indication of priority of risk and which one will be most likely to occur. The standard Low/Medium/High categorisation of likelihood of risks and impacts associated with them was kept due to it being a standard format of risk management. There was no discernable reason to change it to a numerical system. As this is a small, simple project with generally lower consequences to risks, a simplified rating system was sufficient.

A risk type section has been added to each risk in the updated risk assessment. The original risk assessment listed the risk but did not categorise them, which made patterns and recurring issues harder to identify. By introducing categories for the risk, the team is able to recognise which types of risk occur most frequently and to understand more clearly where the project is most vulnerable. This helps the team focus on the underlying causes of risks rather than treating them individually and makes it easier to see which areas of the project are most likely to cause future issues. Additionally more risks were identified and monitored. This resulted in more project/business risks identified which allowed us to keep track of risks in a more manageable way.

The ownership section was changed to fit our team and restricted to at most 2 individuals, usually 1 individual compared to team 7. This change was made so the responsibility of risk monitoring was clear and easier to manage as there are less team members involved so there is less confusion on who is responsible for managing the risk.

Formatting issues were fixed as the risk assessment had inconsistent size in their table. The text in the risk assessment consisted of text size 8 instead of >10, this was fixed to be in lines with the assessment criteria. Also, the "Risk(ID)" section was changed and split into two sections for the risk table, this being the "ID" and "Description". This change was made to make the table easier to understand and to improve the structure.