UNIVERSITY OF YORK
DEPARTMENT OF COMPUTER SCIENCE

ENG 1
GROUP 9 - Kitchen Tossups

Change Report

Team Members:

Adam Penny
Ben Brown
Cameron Fox
Eoin O’Connor
Jack Cameron
Liam Burnand

PART A

When planning the change report, each of the received assessment one deliverables were duly reviewed and analysed, with any errors identified and changed. To do this, we downloaded the deliverables off of team 8's GitHub repository page and read through them, making note or highlighting things we believed relevant. For the documentation, Google Docs was used. We believed this to be the appropriate choice because it would allow for simultaneous editing and we can keep track of previous versions of documents. For the implementation of PiazzaPanic, IntelliJ IDE was used for editing the code and GitHub was used as the version control. 'GitHub actions' was used to implement our continuous integration method.

To keep track of the changes we made, a "change table" was created to list and index all changes made to reference them in Part B of the report.

When altering the requirements, we compared both our old requirements from the Greenfield phase with the received requirements and added any we felt necessary to add to the list. We also added multiple other requirements based upon those mentioned in the second brief we received and from the requirements meeting we held with our customer. These changes were kept track of on the document history tab in Google Docs, whilst the meeting notes were simply jotted down.

For the risk assessment, the document team proofread team 8's file and then noted down changes we thought necessary. Then we spoke to the team whose project we chose regarding the feedback they received for their change report. We also took time to consider any points from our risk assessment that we considered valid. Any ideas that we thought were relevant were added. These changes were kept track of via Google Docs/Drive.

For the architecture, LucidChart and PlantUML were used to create the diagrams we needed. We felt that team 8 had not done a good job making their diagrams for assessment 1. We therefore decided to use LucidChart to create CRC cards as we felt this was the more sensible option for our code. We also didn't like how they had made their class diagrams. We looked at their original code and used PlantUML to create new class diagrams that better represented their code. Google Docs/ Drive was again used to track these changes and perform the changing and extension of the justification of the architecture.

Finally, the method selection and planning report was also changed using Google Docs. However, due to it being trivial to change a plan of something already done, we tried to keep changes to this document to a minimum. We didn't like their Gantt Chart format. The tasks weren't grouped together well, and their chart seemed to imply it only took a day per task to be completed. We reformatted their plan into a PlantUML Gantt Chart that better reflects our preferred style. We think this style is more appropriate because it is easier to follow. This is due to the period of time it took to complete each task being better represented and a more aesthetically pleasing colour scheme to differentiate between tasks.

PART B

1. Requirements

Original deliverable: https://kitchentossups.github.io/ass2oldDocs/Req1%20(1).pdf
Updated deliverable: https://kitchentossups.github.io/ass2newDocs/Req2.pdf

Upon choosing the project we wanted to continue the development of, we first read through the documentation received for any missing requirements that we felt could've been added. We also checked for any misplaced requirements that we felt belonged in different sections or FRs and NFRs that we felt had different related URs. After this, we held a second requirements meeting to gather more information about the project we needed to deliver for assessment 2. New requirements that were discussed in the meeting or were mentioned in the product brief were added to the requirements table, culminating in a completed table that contains the requirements for both the Greenfield and Brownfield development phases combined. Some of these changes were: the additions of UR_ENDLESS, UR_REP_POINTS and UR_POWER_UPS amongst others (ID of changes in change table: REQ_ENDLESS, REQ_POWER_UPS, REQ_DIFFICULTY, REQ_REP).

The requirements statement was extended to include a description of the Brownfield requirement elicitation and what that brought to the requirements section of the project. It also included a brief description of how these requirements were elicited (ID of changes in change table: REQ_STATEMENT). There were also changes to the relations for some FRs and NFRs, as we felt that some existing requirements had better links to others than the links we received (ID of changes in change table: REQ_RELATIONS). Also, the priority of some of the requirements was also changed to better represent the needs of our project. (ID of changes in change table: REQ_SHOULD/SHALL).

Another issue that we recognised was the fact some requirements seemed too vague, and so these were fleshed out to be more specific and relate more to the linked requirements (in the case of FRs and NFRs) to create a more cohesive requirements table (ID of changes in change table: REQ_FLESHING).

2. Architecture

Original deliverable: https://kitchentossups.github.io/ass2oldDocs/Arch1%20(2).pdf
Updated deliverable: https://kitchentossups.github.io/ass2newDocs/Arch2.pdf

The first, major change we made was to the UML diagrams we received. When first opened, we noticed that the diagrams were blurry, too complicated, too large and contained methods and relationships that were either incorrect or unnecessary. This led to a complete remake of team 8's final class diagram to be more readable and to contain the correct information. We then removed their blurry final diagram and replaced it with our own. Once the final code was completed, the class diagram was extended to represent our finished product. (ID of changes in change table: ARC_UML_SIZE, ARC_UML_CONTENTS, ARC_UML_CONTENTS).

The received use-case diagram was also hard to follow, and so a replacement was created, similarly to the class diagrams. This was just a resized replica and was the exact same as the original that was provided by team 8 (ID of changes in change table: ARC_USE_CASE). Then, the ECS diagram used to plan the architecture didn't correlate to the actual code architecture as it didn't contain any entities or components, rather libGDX actors. This meant that we had to provide a planning method for the architecture that corresponded to how it was going to be built, and so CRC cards were produced to perform this function (ID of changes in change table: ARC_CRC).

After that, an extension to the justification was required due to the continuation of the architecture and development. The extension detailed the most important and relevant development steps whilst linking them to the GitHub commit ID's and also any relevant requirements (ID of changes in change table: ARC_JUSTIFY).

3. Method Selection and Planning

Original deliverable: https://kitchentossups.github.io/ass2oldDocs/Plan1.pdf
Updated deliverable: https://kitchentossups.github.io/ass2newDocs/Plan2.pdf

Team 8 decided to use a project management application to keep track of what tasks need to be completed and when. They originally used ‘Trello’ and then switched to ‘Monday’. Both pieces of software were unfamiliar to everyone in the group. We agreed that the system we used in assessment one worked well for us. We used weekly Gantt charts that kept track of how complete a task was. The advantage to using ‘Trello’ or ‘Monday’ is that you label tasks with different priorities. As the project is relatively small, we have found it easy to keep track of tasks and have all found it easy to identify the priority tasks without the need to explicitly mention them. It is in our opinion then, that having to learn how to use unfamiliar software would be redundant (ID of changes in change table: MSP_SOFT).

The LucidChart app was also used to make their entity component diagrams. We found it difficult to edit the submitted architecture diagrams. The decision was made to remake their diagrams in PlantUML and then build upon them. The disadvantage to doing this is a lot of time was wasted having to redone completed work. We want to keep changes to a minimum to reduce workload, but we felt in this case it was necessary because the new PlantUML diagrams are easier to read and follow (ID of changes in change table: MSP_SOFT).

In team 8’s method selection and planning document, they discuss originally assigning group members the roles of: secretary, meeting chair, librarian, and report editor. What each role entails is a mystery. It is not elaborated upon. We are therefore scrapping this bit altogether and are keeping our previous system of splitting the team into two and dividing the work equally amongst ourselves (ID of changes in change table: MSP_ROLES).

Going forward with team 8’s project, we decided to switch to using PlantUML instead of LucidChart app to create our Gantt Charts. This is because we were already familiar with the format of plantUML from assessment 1. We also agreed that plantUML Gantt charts were easier to read and understand than the counterpart. To keep with the consistency of the project we also converted their previous Gantt charts into plantUML equivalents. By doing so, we are ensuring the project plan and execution can be followed and understood with ease and consistency (ID of changes in change table: MSP_GANTT).

Team 8 had used a waterfall methodology for their project development; whereas we had used a plan driven one and borrowed some of the principles from an agile methodology. As a group, we assessed the fitness of using a waterfall method. We came to the mutual agreement that continuing to use the waterfall method going forward was sensible. This is for two reasons: Firstly, the waterfall method is an appropriate choice for this project and covers everything we need to finalise Piazza Panic; Secondly, we want to ensure consistency between Team 8’s work and our team’s work (ID of changes in change table: MSP_METHO).

Adobe defines the waterfall methodology as “a project management approach that emphasises a linear progression from beginning to end of a project”. The stages we chose for our waterfall method are as followed: requirements, design, implementation, testing and then maintenance. Initially we had to establish what we needed to do and what goals had to be met. This was done by discussing and evaluating what Team 8 had/had not already done and meeting with the customer to discuss requirements and carefully reading through the assessment documents. Once we had come to a consensus, we split the project into well-defined tasks and subtasks shown in the ‘PlanEvolution’ document.. Team 8 had not done this in their original plan. We felt it was appropriate because it clearly lays out what needs to be completed as opposed to their seemingly random 6 segments that contain a wide variety of tasks. For a linear methodology such as the waterfall one, splitting the project into clear sections and subsections allows for a more linear approach to completing each task (ID of changes in change table: MSP_TASKS).

We like how they have tabulated their tasks. We will therefore be keeping this part of their plan. This is so we can track which tasks have been done, who is meant to be doing what and the priority of each task. We didn’t do this in assessment 1 and think it can be valuable information to keep track of. We have removed their ‘file’ column however. Only a few rows had anything in this column so we felt it was redundant. (ID of changes in change table: MSP_TABLE)

The naming conventions for the 6 segments of tasks they used in the Gantt charts is confusing. They have named the segments after dates. Some of the dates used don’t even line up with when the start date of the segment. We have therefore renamed the segments and numbered them from 1 to 6. (ID of changes in change table: MSP_SEGMENTS)

We also swapped their second and third segment around, so the plan was in chronological order. This makes their plan easier to follow. (ID of changes in change table: MSP_CHRONO)

4. Risk Assessment and Mitigation

Original deliverable: https://kitchentossups.github.io/ass2oldDocs/Risk1%20(1).pdf
Updated deliverable: https://kitchentossups.github.io/ass2newDocs/Risk2.pdf

In the risk assessment inherited from team 8's deliverables, we immediately identified some duplicated - or extremely similar - risks in the document. For example, it contained two for lack of work completion on time. Any duplicates were removed. There was also the removal of an unused row at the bottom of the table (ID of changes in change table: RSK_DUPLICATE, RSK_XTRA_ROW).

Due to the removal of a risk, there was subsequently a necessity to then change the ID numbers of the following risks so there are no gaps in the sequential IDs. (ID of changes in change table: RSK_NUMBERS). Another edit made, was the altering of the colour coding of the severity ratings in order to get them all the same for each rating to add consistency to the risk assessment (ID of changes in change table: RSK_COLOUR). We felt this was appropriate because consistency means it is easier for people to follow and understand your work.

After that, we decided that many of the risks we received were a bit generic and needed the descriptions fleshing out, and so risk 3 (unclear schedule), risk 4 (repeated tasks), risk 5 (conflicts between group members), risk 6 (unreadable code), risk 7 (bugs and errors), risk 9 (upcoming assessment and exams getting in the way) and risk 10 (lost work) were all extended to provide more context and even examples for the risks. (ID of changes in change table: RSK_BULK). We wanted to make sure it was clear what we meant by each risk and that we ourselves fully understood what each risk meant.

Another change made to the received document was the changing of the names of who was handling each error as they were originally the names of the former group members. Since they were no longer involved in this specific Piazza Panic project, we saw it fit to change them to our group members and assigned them as we saw fit based on each component of the assessment we were each completing (ID of changes in change table: RSK_OWNER).

Furthermore, we altered some of the severity ratings for some of the risks as many were - in our opinion – slightly misrepresented. For example, risk 13 (misuse of licensed material and resources) was originally set to be of moderate severity. However due to the potential for academic misconduct and potentially not legally being allowed to use such material, we changed the rating to severe (ID of changes in change table: RSK_RATING).

5. General Changes

Overall, some general changes were made to all documents we received. Firstly, on the front page of each document the team number, name and group member names were all changed to correspond to our own (ID of changes in change table: GEN_NAMES). Next, spelling, punctuation and grammar was checked over to make sure no mistakes were present in the final draft(ID of changes in change table: GEN_SPaG).

6. Change Table

Change ID	Brief Description of the Change
Requirements	
REQ_ENDLESS	Added a requirement to correspond to the need for an endless mode.
REQ_CONSISTANCY	Updated any requirements that included old information (such as 2 cooks not 3)
REQ_REP	Added a requirement for reputation points
REQ_RELATIONS	Added the new UR's to the functional relationship parts of the table
REQ_DIFF	Added a requirement for difficulty
REQ_POWER-UPS	Added a requirement for power-ups
REQ_SAVE	Added a save game requirement
REQ_CTRL_SCN	Added an NFR for controls screen
REQ_STATIONS	Update the stations FR to include unlocking
REQ_SHOULD/SHALL	Altered priority of some URs
REQ_MOVE	Added an FR for moving cooks
REQ_STATEMENT	Extended requirements statement
REQ_FLESHING	Fleshed out some requirements' descriptions
Risk Assessment	
RSK_COLOUR	Changed the non-consistent colour
RSK_DUPLICATE	Removed duplicate risks
RSK_XTRA_ROW	Removed unnecessary row with nothing it
RSK_NUMBERS	Changed the designated numbers of risks after removing duplicates and extra row
RSK_BULK	Fleshed out some risks
RSK_OWNER	Changed the names of the owners of each risk.
RSK_RATING	Lowered/ raised the severity of each risk
Method Selection and Planning	
MSP_GANTT	Changed format and composition of Gantt charts
MSP_METHO	Used different methodologies for development

MSP_SOFT	Changed the software used to create plan
MSP_ROLES	Removed the need for roles such as secretary, meeting chair
MSP_TASKS	Changed visualisations of how tasks were carried out.
MSP_TABLE	Removed the ‘file’ column from their tabulated tasks
MSP_SEGMENTS	Renamed the 6 segments of tasks team 8 had made
MSP_CHRONO	Swapped around the order of team 8’s original segment 2 and segment 3 to put them in chronological order
Architecture	
ARC_JUSTIFY	Added a new section in the justification section to designate our extension work
ARC_UML_SIZE	Increased the clarity of the UML diagram by decreasing its size and zoom
ARC_UML_CONTENTS	Removed unnecessary inclusions in the UML diagrams
ARC_USE_CASE	Made the received use-case diagram readable
ARC_CRC	Added CRC cards for planning instead of ECS diagram as entities and component weren’t used
ARC_UML_ORDER	Changed the order UML diagrams are in, adding more in after
General Changes	
GEN_SPaG	General spelling, punctuation and grammar corrections
GEN_NAMES	Changed the names on the front page of all documents.