SEPR 2019/20 Assessment 3

Team CheatCodez

Jonathan Grout, George Lesbirel, Cheuk Wang Wu, Lauren Quarshie, Muaz Atif, Lillian Coultas

Change Report

Approach to Change Management

Our approach to change management uses a combination of a top-down approach, (Lewin's [1]), to provide focus on performance improvement, and a bottom-up approach, (Deming's [2]), to provide new approaches to solving the problems by encouraging innovation from all team members.

Change management approach to documentation and deliverables

Our team has decided to use a variation of Kurt Lewin's Change Management Model [1], which incorporates Lewin's Force Field Analysis [3] :

The "Unfreezing" stage will consist of multiple team sessions in which the team will look over all documents, and conduct a Force Field Analysis to understand the factors (or 'forces') for or against making changes to the documentation and deliverables. The Force Field Analysis will also involve an impact analysis of any potential changes, as well as the actual approval or denial of these changes. The "Change" stage will be used as a transition period involving the actual implementation of the change. The implementation will be tracked in the form of a Change Register (see example in **Appendix 1.2**), in which the changes will be categorized depending on which area of the project they affect; changes to **requirements: REQ_number**, changes to **risk assessment: RIS_number**, changes to **methods/planning: MET_number**, changes to **architecture: ARC_number**, changes to **software: SOF_number.** Each change will also have a requirement reference, which represents the corresponding requirement that the change would affect if it gets approval for implementation. The Change Register will include both approved and denied changes to ensure traceability for the stakeholder and clients. The "Refreezing" stage will involve internalising all approved changes by tracing, and consequently updating these changes throughout all documentation and deliverables.

Change management approach to code

Due to the iterative nature of this assessment, using an approach such as the Deming Cycle (PDCA) [2] allows for collective-decision making as a team. The **'Plan'** phase will consist of identifying any bugs or inefficient techniques used, as well as brainstorming potential solutions or improvements. The **'Do'** phase will involve implementing the suggested changes on a smaller scale for testing, until there is a level of certainty that means the change can be implemented project-wide. The **'Check'** phase will involve benchmarking the improvements and analysing how well the changes are performing compared to the old methods. This phase also involves looking at the long-term performance of the improvements i.e. checking if they are stable enough to endure multiple iterations. The **'Act'** phase will involve three potential actions:

Adopt : implement the change across the whole project if they perform well and improvement is clear

Adapt : revise the solution/improvement by going back to the beginning of the cycle and repeat the stages using a different method, or by conducting more research beforehand.

Abandon : if the proposed change creates a new problem or exacerbates the original problem then return to the 'Plan' stage and re-analyse the problem. Once a new plan is developed, continue with the rest of the cycle.

Any changes made to the code itself will also be documented in the Change Register.

Changes to Testing Report

Despite having taken over a new game, our approach to testing remains largely the same as it did before. We will still be performing unit, integration and manual tests using JUnit. Our team is experienced with JUnit, and it is an excellent tool to write unit tests in Java. Last time our tests achieved 41% class coverage (when run with coverage through Intellij), this time we're planning for higher coverage to ensure that all our requirements are tested.

We will be using a traceability matrix (see below for link) , to show the relationship between requirements and created tests. The matrix will make it clear which requirement each test is covering, and we are aiming for every requirement that is implemented at this stage to have at least one test covering it. Some requirements may be covered using unit and manual tests, whereas others may only be covered by one of the two; this is due to the structure of the requirements.

Our layout of the tests will be similar to before, in each test case we will describe what the test is doing, whether it passes or fails, and provide clarification as to why that is the case. This time we will make it clear what the inputs are for each test (where necessary) and what we expect the output to be. By seeing what the expected output is, we can make a judgement on whether we think the test has been implemented correctly. While a high test pass rate is good, if we expected many to fail then it may mean we have implemented the tests incorrectly and that they need editing.

To make sure we can test as much as is necessary, we will be using partitioning. By breaking each section of the code down into partitions, it makes it clear which areas need to be tested and how many tests need to be run on each section. Ideally we would want at least one test for every partition to achieve maximum test coverage.

The tests we have conducted can be found on the website (https://teamcheatcodez.github.io/project-kroy/supplements/testcases.pdf).

Traceability matrix referenced above can also be found on the website (https://teamcheatcodez.github.io/project-kroy/supplements/traceabilitymatrix.pdf)

Changes to Methods and Plans

For this stage of the project the tools we plan to use will remain largely similar to the previous assessment, with one new addition and a few tools now redundant. We are now using Firebase [https://firebase.google.com/], a platform for mobile and web development owned by Google. While we did not need to use it previously, the website we acquired from Salt N Sepr (SNS) requires us to use this to host the documentation on the website.

Previously we used Box2D to handle the physics and collisions of our game, but as we are now developing a different game, Box2D is no longer used in the implementation. There was no reason to add Box2D here as the physics and collisions work fine without it, for this reason we will no longer be using it. Additionally, we plan to keep using the same database we designed before for this game; as we do not need to design a database again, we will no longer be using MySQL workbench. MySQL workbench was useful for creating and generating our database, but as it is already set up we can remove it as a necessary tool. We will also no longer be using Swagger as the implementation for the new game does not require it to be used for database interaction, therefore it is redundant and no longer a required tool.

The minigame we are developing for this stage of the project requires additional assets to the ones provided. We are continuing to use Photoshop for asset development as it is a program we are familiar with and it allows us to create high quality assets for the game. The previous group designed their assets using Aseprite, so the assets for the minigame may be of a different art style to the rest of the game but it will still be fully functional.

Our approach to team management remains largely similar to the previous assessment. George and Muaz will continue to develop the game, this time with assistance from Cheuk. As this stage of the project has a shorter deadline than the previous one, we thought it prudent to allow Cheuk to assist with development to ensure the game is fully developed and tested in time. Jonathan, Lauren, and Lillian will continue with project documentation and testing where required. The Belbin team roles assigned during the first stage of the project will remain unchanged.

For this part of the project, we will still be using Jira to assist with managing our sprints. As there is less time available for this assessment we aim to complete more tasks in each sprint to make sure that everything is finished and reviewed before the deadline. This will help mitigate the risk *Estimation (R-01)*, as we are ensuring that the project will be completed on time.

As the project we are developing has now changed, so has our plan. Our approach to development will now be based on the Gantt chart created by Salt N Sepr rather than our own Gantt chart created previously. This Gantt chart suits our development targets and so we have no changes to make to it. This Gantt chart showing our plan can be found in the **Appendix 1.1 (https://teamcheatcodez.github.io/project-kroy/assets/docs3/appendix.pdf)**

References

[1] - Managementstudyguide.com. (2020). *Kurt Lewin's Change Management Model: The Planned Approach to Organizational Change*. [online] Available at: https://www.managementstudyguide.com/kurt-lewins-change-management-model.htm [Accessed 6 Feb. 2020].

[2] - Tallyfy. (2020). *6 Essential Change Management Models to Help Innovate & Grow - Tallyfy*. [online] Available at: https://tallyfy.com/change-management-models/#Bottom-Up_Approach [Accessed 6 Feb. 2020].

[3] - tutor2u. (2020). *Lewin's Force Field Model (Change Management)* | *Business* | *tutor2u*. [online] Available at: https://www.tutor2u.net/business/reference/models-of-change-management-lewins-force-field-model [Accessed 6 Feb. 2020].