Change Report

SEPRet Studios

Anthony Nel, Ben Silverman, Jake Billington, Leah Moss and Shouyi Yuan

Team's formal approach to change management

When changing the requirements deliverable, we looked through each requirement and decided if it was appropriate and should be removed or have its priority changed. Afterwards, having looked through all the requirements we discussed if we thought there were any requirements that were missing. We then added these missing requirements, and highlighted them as yellow to show that they have been added in this stage of the assessment.

Having updated the requirements deliverable we moved on to the architecture. Here we discussed what changes would need to be made to the architecture to fit with our updated requirements, and also to meet requirements that existed in the previous assessment but didn't need to be implemented till assessment 3. To highlight changes that have to be made to the games architecture we made all text in the added classes red, in order to make them easier to identify.

All changes in software development methods tools used, team management approaches and plan for Assessment 4, are documented towards the end of this change report. For the software development tools and management methods, we discussed before beginning to code the required changes for assessment 3 and decided to continue, for the most part, with the same methods and tools we used in assessment 2. We did this because we believed that this approach worked well for assessment 2 and adopting any changes that this group used would just slow us down as we tried to adapt to the different tools and methods.

All changes to the test report are documented below in this change report. Before we began writing any code for assessment 3 we decided that we would continue on with the same testing structure as the previous group did for assessment 2. We did this because we thought that the previous group's testing report was thorough and well structured, so we agreed to carry on new tests with a similar approach.

We didn't make many changes to the previous risk report, because we thought that most of the risks stayed the same. So we just decided upon who would be allocated to keep track of each risk set out in the previous report, and also made small changes where necessary to risk mitigation, where our methods were more amenable to a different approach. All changes are highlighted in yellow, to make them easier to identify.

We made changes to the code based on our changed architecture, however there were smaller changes to the code base that we couldn't have predicted we would have to make from the architecture. For these changes, we added these changes to a change log that we were keeping track of, so that we were all informed of changes to the code and weren't confused by it. Then following this, we commented above the changed and newly added code that it was done in assessment 3 so that it could be easily identified as code that had changed from the original version.

After choosing this game to develop on we looked at the code and its documentation, all classes and methods had been documented in the same manner that we did for our assessment 2 game allowing a javadoc to be created. We decided to make no changes to this approach, as it had worked for us previously so we continued documenting new methods and classes in the same fashion. However, we thought that code within methods was a bit sparsely commented, so we decided to add some more inline comments to existing code and code that we added as we went along.

Explanation of changes made

Testing

Test Plan & Report URL

https://docs.google.com/document/d/1wH6X1rLhD_Ar_ZQ3d5Hb482hpbPo01dKS_VQy0MXjt8/edit?usp=sharing

Note: All additional testing material (e.g. unit test results, coverage results, traceability matrix) has been linked within the above Test Plan.

Methods and Approaches

Upon reviewing our previous team's codebase, we made the decision to extend their current JUnit test classes and add our own, as we wanted to ensure the pre-existing tests would still pass upon completion of Assessment 3. These unit tests proved a great template for us to base further unit tests off, too; had we deleted everything and started again, it likely would have taken longer to reach a working state again.

Due to the increase in functionality of our game during this assessment, we added new tests to all three of the existing test classes (FireTruckTest, FireStationTest and FortressTest) as well as adding two more test classes (AlienTest and MinigameTest) for the alien interaction and minigame respectively.

Similar to the previous team's unit tests, we also decided to extend their manual tests to fit the new requirements in Assessment 3. While we had originally laid our manual tests out in a spreadsheet-style format, the previous team opted for writing each test out in a word document. The benefits to this were that it allowed for a "process" to be written out in plain English, so future developers would be more easily able to run the tests themselves. However, we decided to modify the style of the document somewhat to include tables under each test ID detailing the inputs, expected outputs and actual outputs. We felt this made it much clearer as to which tests passed and failed, as well as the specific inputs and outputs expected for each test.

Linked inside the Test Plan is a full changelog of how we extended and modified Assessment 2's testing material.

Presentation of tests and testing statistics

To produce the Test Report for Assessment 3 we modified the previous team's Assessment 2 Test Report. Due to the increase in number of unit tests, we have updated the statistics surrounding test outcomes, coverage and comments on further improvement. We also decided to remove the satisfied requirements for each unit test, since we did not feel it was actually viable for a single unit test to fulfill an entire requirement by itself.

As mentioned before, we also slightly adapted the Manual Test plan to include more detail within each test. The Traceability Matrix has also been updated to include the added manual and unit tests.

Methods and plans

Assessment 4 plan URL

https://drive.google.com/open?id=1fCXQEQoNH0DD1Nces8RC4s2h8QEYy6tA

Use of tools

Similarly to the previous team, we also used messenger as an existing tool to help with the delegation and organisation of work within the group, and to organise group meetings. However we didn't decide to continue with the previous team's use of weekly emails. We decided to do this because we already met at least twice a week, which for us was enough for each individual within the group to determine whether they were on track with regards to the completion of the project.

While the other group didn't specify within their method report which IDE they used, we decided upon intellij because we had already utilised it with relative ease for the previous stage of the assessment. Additionally, it provided assistance with handling most of our interactions with Git, allowing us to create and view pushes, commits, and merges to the version control system. Because our IDE offered a helpful interface with respect to controlling Git, we didn't need to use the software GitKraken that the previous team used in order to make using git easier.

Also unlike the other group we didn't use gitkraken IO in order to create to do's and flag bugs that need fixing to the other members of the group. Instead we decided to use trello in order to perform these tasks. We decided on this because we used trello for the previous stage of the assessment, giving us familiarity and saving us time that would have otherwise been spent learning how to use a new tool.

Development methodology

Similarly to the previous team working on this game, we decided to take an agile approach to the creation of the game. We decided upon this, because we took the same approach for assessment 2 and it worked well for us providing sufficient flexibility, with an acceptable amount of overhead with respects to time spent managing the team. However we did meet more frequently than the previous group, meeting twice a week. We decided upon this because it allowed us to respond more quickly and re-delegate work when shorter tasks had been completed sooner than expected, additionally it provided an opportunity to more frequently share development issues uncovered when creating the game.

Group roles

We also decided not to follow the other groups role scheme, because everyone felt comfortable in programming we all assumed a development role. Everyone also handled retrieving/creating the assets for our own development tasks that we chose. Ben Silverman was the tester, because he assumed this role for assessment 2 and was already familiar with testing procedures. And everyone took on the role of contributing to the reports.

Plan for assessment 4

We have decided to keep the previous groups plan the same, for the most part for assessment 4. We did this because we believe that the plan gave a sensible amount of time to each task. Additionally, we didn't need to add anything. This is because we have not gained any additional information as to what assessment 4 entails. However, we did move the start date for choosing the net project we will be working on, because the previous group wanted to start that after the deadline for choosing.