Evaluation and Testing

Module	SEPR
Year	2019/20
Assessment	4
Team	York Fire Marshalls
Members	Samuel Whitehead, Andrew Connor, Alex Dawson, Finn Jackson, Fred Dodd, Phuong Kha
Deliverable	Evaluation and Testing

Product Brief

As part of assessment 4, we have been tasked with taking over another groups project and have been given new requirements for a product brief. These new requirements have been listed below.

- Implement five special power ups that Fire Engines can obtain on the journey (e.g. granting temporary immunity, repairing damages/refilling their tanks without returning to the Fire Station)
- Implement support for different levels of difficulty in the game (e.g. easy, normal, hard)
- Implement facilities that allow players to save the state of the game at any point and resume a saved game later

To implement these requirements, we decided it was important to update the previous groups project requirements to include these new additions for the continuity of testing and for each member of our group to know the exact details that we had decided on with the changes. For example, the requirement to add five special power ups did not go into detail as to what the power ups should do, so it was important to outline this in the updated requirements to make sure our team knew what these power ups would do. A link to this updated requirements requirements can be found on our website here https://fpd500.github.io/SEPR16.github.io/.
The document itself has been formatted in the same way that the previous groups formatted into a table, splitting each requirement into the separate categories User, Functional and Non Functional. Each of these have been added for the continuity of the testing of the project, and because of this, have been formatted in the same way that the previous groups had done. Each of these requirements have also been added to the updated traceability matrix and new test cases have been created to ensure that each requirement is met without any issues. In total, we have added 18 new requirements that we need to have met. To ensure that all of these requirements have been fully met we have used testing.

Testing

For the testing of our new additions to the software, we found that continuing on from the previous groups testing report was the way forward. This is because it would allow us to continue using the exhaustive testing table that has complete coverage of all of the previous requirements. All of these tests including the report for assessment 3 can be found at https://sepr-docs.firebaseapp.com/testing. To continue their testing methodology, it was important to follow the ISO/IEC/IEEE 29119 standard[1] just as the previous group outlined in their testing report. To do this, it is important for our testing to be preplanned, and to continue throughout the process of the development cycle, so once we had written our new requirements into the project requirements table, we wrote tests into the previous groups test table to ensure that each requirement would be met. We used black-box testing to ensure that our requirements would be met for the end user, with many of these tests covering more than one requirement. The previous group had used a mix of unit testing and black box testing, but due to difficulties surrounding unit testing in libgdx, only the core features of the game have been tested using unit tests. As we will not be changing these core features, and all new code will be extended from these objects, we will not be adding any new unit tests.

As this is the case the previous code coverage will be the same as before and we will be concentrating on creating tests that ensure our requirements are met for the user. Each test all has a level of severity attached to it, which is based on how severe it is for the test not to pass, for some of our tests this is very high as if the test fails, an entire requirement will not be met, whilst others are of low severity as if they fail, at least part of the requirement is met.

To ensure complete coverage of our testing, we have used the same traceability matrix as the previous group to ensure each requirement is being tested so that once we have completed our project, can we say for certain in our evaluation that all requirements have been met.
In total we have written 13 new tests and will be implementing these tests after each new feature is added into our game, and will only accept that a feature has been completed once all tests that are linked to it will pass. After each new feature has been added and tested successfully, a code review will be undertaken by each member of our team that has not themselves produced the code, this will consist of improving the quality of the code and its readability or attempting to find better solutions improving the quality of our code and making tests more likely to pass.

Testing of final project
On our website it is possible to find both the testing table and our traceability matrix at https://fpd500.github.io/SEPR16.github.io/. As shown in the testing table, here are the results of our 13 new tests

.
Test results
Number of tests passed: 12
Number of partial test passes: 1
Number of tests failed: 0

In total we have used 13 tests to evaluate whether we have met the new requirements of the project, of these all 12 have passed, and one has partially passed. It is important to note however that this does not mean that there are no more errors in our software, as it is impractical to have exhaustive testing.
The pesticide paradox also restricts us from finding every bug, stating that every method used to find a bug, leaves a small residue bug for which that method is useless. Meaning that we can never definitely say our code is bug free. Overall, we believe that our testing has been suitable and that it covers enough that the game should be functional. Since it tests all the major interactions a user will take, there shouldn't be any problems when a real user plays the game. As shown by our traceability matrix, each requirement has been tested successfully, showing each new feature has been implemented except from one.
The test that had only partially passed is T.N.010, which tests that the loaded games are in the same state as when they were saved. This only partially passed as whilst the loaded game is nearly exactly the same as the game that was saved, enemy patrols are not saved. This can be seen as a problem as it can be used as an exploit to cheat in the game.

Evaluation of final product

The initial product brief stated that the final version of the game must be “playable and enjoyable by your SEPR cohort “. We feel that we have been extremely successful at this, having created a game that is easily played, lacking many noticeable bugs and is extremely fun to play. The requirements for Assessment 4 specified 3 major changes, these being five different types of power ups, three different modes of difficulty, and the ability to save and load the game at any point while playing. All requirements have been met as the game now contains 5 powerups, the three difficulty modes and the ability to save the game. As shown by the final testing of the project, all of the tests passed except one. These requirements for assessment 4 can also be found with our traceability matrix on our website at https://fpd500.github.io/SEPR16.github.io/, which is evidence that our final product has met the criteria of the requirements given to us.

One way that our final product has not met the specification of one of our stakeholders is that it was specified in an email that the game should be able to save during the mini game. This was not achieved as we implemented the saves game option into our pause screen, and the project we had chosen did not allow for the game to be paused whilst in the mini game. We felt that this would not be a big problem because of the style of the minigame, as it is fast paced, and pausing the game would allow for the player to have an advantage that we consider to be cheating.

Another way in which we have not met the final specification is that alien patrols would not be saved when the user attempts to save the game. When we would try to fix this, our game would break and we decided to spend our time completing the other requirements instead of breaking the game further.

Bibliography

[1] ‘ISO/IEC/IEEE International Standard - Software and systems engineering –Software testing –Part 1:Concepts and definitions’, ISOIECIEEE 29119-12013E, pp. 1–64, Sep. 2013 [Online]. Available: 10.1109/IEEESTD.2013.6588537