# SFT221 SCRUM Report and Reflections

This report should be completed in the class and submitted at the end of class. Late submissions cannot be accepted without prior approval of the instructor. All students are expected to attend the in-class SCRUM meetings and to participate. Failure to do so will result in greatly reduced grades.

**GROUP : 1**

**Members Present**:

|  |  |
| --- | --- |
| 1. Gyeongrok, Oh | 4. Pui Wai, Tse |
| 2. Hoi Kit, Cheung | 5. Sau Ching Yuki, Wong |
| 3. Lap Chi, Wong | 6. Yonghun Won |

## Milestone 4 Tasks

**Deliverables Due at end of Lab:**

* Completed SCRUM report and reflections

**Deliverables Due at 23:59 6 Days after Lab:**

* Implemented Functions
* Implemented blackbox tests (store in repo), executed (results in Jira and on corresponding test documents) and debugged,
* whitebox tests written and stored in repository.
* whitebox tests implemented (store in repo), executed (results in Jira and on corresponding test documents) and debugged.
* Updated function-test matrix stored in the repository.
* Completed hook for test automation

**Rubric**

|  |  |  |
| --- | --- | --- |
| Individual | Group Participation | 75% |
| Teamwork | 5% |
| SCRUM Report | 10% |
| Automation Hook | 10% |
| Group | Implemented Functions (well-designed, written and documented) | 20% |
| Whitebox tests (well-designed, written and documented) | 20% |
| Test Execution (performed, results recorded, issues created) | 20% |
| Debugging (Bugs fixed, documented, Jira updated) | 5% |
| Git Usage (used properly with good structure) | 5% |
| Jira Usage (creates issues, tracks progress) | 5% |
| Meets Deadlines | 5% |
| SCRUM Report and Reflections | 20% |

**SCRUM Report**

**Summary of Tasks Completed or Delayed in the last week:**

Here you can list all of the tasks completed in the last week along with any tasks which could not be completed with a reason why they could not be completed.

|  |  |  |
| --- | --- | --- |
| **Member** | **Tasks Completed** | **Tasks Delayed/Blocked** |
| Hoi Kit, Cheung | Reflections,  Fill in the function-test matrix,  Functions specifications for part 1  (part 1) :   1. struct Map addRoute(const struct Map\* map, const struct Route\* route) 2. void printMap(const struct Map\* map, const int base1, const int alphaCols) 3. double calculateMinDistance(const struct Truck\* truck, const struct Point\* dest) **[New function]** 4. void listShortestDiversion(const struct Route\* diversion) **[New function]** | None |
| Lap Chi, Wong | Reflections,  Write the blackbox test code,  Review what functions need to be written,  Functions specifications for part 2  (part 2) :   1. int getClosestPoint(const struct Route\* route, const struct Point pt) 2. struct Route shortestPath(const struct Map\* map, const struct Point start, const struct Point dest) 3. int getShipmentFromInput() **[New function]** 4. void storeShipment(int weight, double size, const char\* dest[]) **[New function]** 5. int dispatchTruck(const struct Truck[]) **[New function]** | None |
| Gyeongrok, Oh | Reflections,  Functions specifications for part 3  (part 3) :   1. struct Map populateMap() 2. struct Route getBlueRoute() 3. struct Route getGreenRoute() 4. struct Route getYellowRoute() 5. struct Point convertStrToPoint(const char\* dest) **[New function]** 6. const char\* convertPointToStr(const struct Point\* point) **[New function]** | None |
| Pui Wai, Tse | Reflections,  Blackbox test case for part 1 | None |
| Sau Ching Yuki, Wong | Reflections,  Blackbox test case for part 2 | None |
| Yonghun Won | Reflections,  Blackbox test case for part 3 | None |

For every task delayed or blocked, describe the reason for the delay or block, how it impacts the project and the proposed solution or workaround**.**

|  |  |
| --- | --- |
| **Delayed or Blocked Task** |  |
| **Reason for delay or block** |  |
| **Impact on Project** |  |
| **Solution or work-around** |  |
|  |  |
| **Delayed or Blocked Task** |  |
| **Reason for delay or block** |  |
| **Impact on Project** |  |
| **Solution or work-around** |  |

**Summary of Meeting:**

A summary of the main points discusses in the meeting and the outcomes of the discussions.

|  |  |  |
| --- | --- | --- |
| Topic | Discussion Summary | Outcome |
| Workload distribution | To distribute the tasks for milestone 4 to team members so that each member has a fair share of workload | The workload distribution is agreed in the meeting |
| Discussed project timeline | Defined project timeline and deadlines. Adjusted the timeline based on everyone's needs and ensured that all team members agreed on the deadlines. | Project timeline finalized with adjustments based on team consensus. |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Summary of Decisions Made:**

This will include major architecture and design decisions, testing decisions, prioritization of tasks, dealing with problems encountered and other major outcomes from the meeting.

|  |  |
| --- | --- |
| Decision | Rationale |
| Implement a Communication Improvement Plan for Milestone | In the feedback from the last milestone 2, the professor requested more effective communication using Jira. Keeping this in mind, we made efforts among the milestone 3 team members to achieve this goal, and thankfully, we were able to receive an impressive score of 100%. We have agreed to maintain this momentum in milestone 4, aiming to further improve effective communication through more active Jira usage and frequent GitHub commits. |
| Prioritization of Tasks | The team has decided to prioritize the development of core functions that are critical for the project's functionality and build upon them. Non-essential features will be deferred to later stages of the project to ensure timely delivery of the primary functionality. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Tasks Attempted During Meeting:**

Each member is assumed to participate in the SCRUM meeting and contribute to the completion of the SCRUM report and reflections. Since the SCRUM meeting will not take more than 20-30 minutes, there is lots of time left to undertake some of the actual work tasks. In the table below, each member should list what they did to complete the SCRUM report, the reflections, and 1-4 other tasks they completed during the class period. If a task could not be completed, the student should indicate why this was not possible.

|  |  |  |  |
| --- | --- | --- | --- |
| Member | Task Attempted | Time Spent | Complete? |
| Gyeongrok, Oh | Reflections, function review | 30 mins | done |
| Hoi Kit, Cheung | Reflections, function review | 30 mins | done |
| Lap Chi, Wong | Reflections, function review | 30 mins | done |
| Pui Wai, Tse | Reflections, function review | 30 mins | done |
| Sau Ching Yuki, Wong | Reflections, function review | 30 mins | done |
| Yonghun Won | Reflections, function review | 30 mins | done |

**SCRUM Tasks Selected for Next Week**:

The tasks each member has selected to pursue for this class or the next week.

|  |  |
| --- | --- |
| Group Member | Task Description |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Major Outcomes of Meeting:**

This is where you should highlight the major accomplishments of the class.

|  |  |
| --- | --- |
| Outcome | Impact on Project |
| Defined Project Scope and Objectives | By establishing a well-defined project scope and objectives, the team can now focus their efforts on delivering the essential features within the specified boundaries. This clarity will prevent scope creep and ensure a more efficient use of resources. |
| Assigned Roles and Responsibilities | The meeting resulted in the assignment of roles and responsibilities to each team member based on their expertise and interests. Clear role assignments enhance accountability and productivity. Each team member now knows their specific tasks and contributions, reducing overlaps. |
| Planned Task Prioritization and Scheduling | The team collectively prioritized project tasks and developed a timeline for their completion. Task prioritization and scheduling allow the team to allocate resources effectively, focusing on critical components first and meeting project milestones in a timely manner. |
|  |  |
|  |  |
|  |  |
|  |  |

**Things That Went Well in This Meeting:**

Here you can highlight things which worked well. This indicates that the way you worked on these items is working and should be continued.

|  |  |
| --- | --- |
| Topic/Work Item | Reason for Success |
| Workload distribution | Requirements of Test Case and code are fully understood by all team members to create reasonable sub-tasks |
| Project Timeline | Realistic and achievable milestones set |
| Responsibilities | Clear understanding of individual tasks and roles |
|  |  |
|  |  |
|  |  |
|  |  |

**Things That Did NOT go Well in This Meeting:**

This is where you can list things which did not go well in the class. You should analyze why this happened and suggest how you can improve it next time. This will lead to the goal of *continuous process improvement*.

|  |  |
| --- | --- |
| Topic/Work Item | Reason for Problem and How to do Better |
| NIL |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Reflections**:

1. After you run your blackbox and whitebox tests you are asked to record the results in both the original test document as well as in Jira. Explain why it is a good idea to record the results in both places.

There are several advantages which enhances the overall testing process. Documenting the results in the original test document ensures that all the testing details, methodologies, and outcomes are well-organized in one central location. This makes it easier for testers, developers, and stakeholders to access comprehensive testing information. The test document provides a reference of testing which can be used for future planning.

Besides, recording the results in Jira allow tracking history, while issues found during testing, the team can create tickets and links to related cases, and make discussions through the tool. brings the related parties to the relative tasks directly. Jira enable traceability between test cases and requirements, so that they are better facilitated and managed. Developers can easily refer to Jira to understand the test outcomes and quickly address any identified issues. It also enables higher visibility of testing efforts and allows project managers to make decisions based on the recorded results.

Recording the test results in both the original test document and Jira combines the benefits streamlined issue tracking and project management, which make the testing process m more efficient and well-organized.

1. Why did we wait until the fourth milestone to write the whitebox tests?  
     
   Each of the milestone in this project resembles each step in the Software Testing Life Cycle (STLC). The last milestone (MS3) resembles the “Test case development" stage, in which implementation is not started yet. Without the presence of actual code, it is impossible to write any whitebox test case, because such test cases can only be designed after inspecting and understanding the actual code, and these test cases will attempt to expose the logical flaws of the coding logic. Blackbox test cases can be written once there are requirements gathered, since it disregards the coding logic and only focus on the actual result with the given input parameters. Therefore, MS3 is the stage in which blackbox test cases can be produced without the codes implemented.

MS4 resembles the “Implementation" stage, in which implementation of all proposed functions supposed to be finished. It is the appropriate timing for developing whitebox test cases with actual logic known.

1. For a given function did you produce more blackbox or whitebox tests? Explain why your answer (more blackbox or more whitebox) happens for most functions.

In most cases, there tend to be more blackbox test cases than whitebox test cases for functions. The reason for this is that blackbox testing is relatively more accessible and easier to conduct compared to whitebox testing.

Blackbox testing allows testers to focus on the external behavior of the function without needing to understand its internal implementation details. Testers can create test cases based on the function's specifications and requirements, making it more user-centric. This approach is especially useful when testing software products, as it emulates how end-users would interact with the function without needing to delve into the code.

On the other hand, whitebox testing requires a deeper understanding of the function's source code. Testers need to analyze the internal logic, branches, and paths in the code to design test cases that ensure thorough coverage. This process can be more time-consuming and requires technical expertise in programming.

Due to these factors, testers often prioritize blackbox testing, and as a result, more blackbox test cases are typically produced. However, it's essential to strike a balance between the two approaches, as whitebox testing can be valuable for uncovering issues that might be missed through blackbox testing alone.

In summary, the abundance of blackbox test cases is mainly due to its simplicity and user-centric nature, making it a popular choice for testing functions, especially when the internal implementation details are not immediately relevant or accessible to the testing process.

1. Explain the purpose of the automation hook for GIT and explain how it can improve the quality of the software in the project.

The automation hook for GIT serves as a safety measure to automatically run tests before code is pushed to the repository. It ensures that the code passes a suite of tests before being added to the repository. This hook is installed in the project's GIT directory, and each team member sets it up on their own computer. The hook runs a script that checks if the tests are successful. If the tests fail, the push to the repository is stopped, prompting the developer to fix the code before trying to push it again.

By automatically running tests before each push, it prevents the introduction of bugs or issues into the shared codebase. This ensures that the code is reliable and stable, reducing the likelihood of causing problems for other team members. It encourages developers to thoroughly test their changes before attempting to integrate them with the rest of the project. As a result, the team can have more confidence in the codebase, and the risk of breaking existing functionality is minimized. The automation hook also promotes a disciplined testing approach, encouraging developers to write comprehensive test cases for their code. This leads to a more robust and maintainable software project, as any issues are caught early in the development process, making them easier to fix.