# **MILESTONE 3** -- 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.

**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 3 Tasks

In this milestone you will create issues to design the functions, design all of the functions you need to complete the project and store the specifications in the repository. As soon as the specifications start to be produced, you can start to design the blackbox tests (what they test, how to perform them and test data). Once tests are written, they can be implemented and added to the repository and any team members not otherwise busy can start to implement the functions. You will also build a function-test matrix that shows the blackbox tests for each function. This will be maintained through the testing cycle as new tests are added.

**Deliverables Due at end of Lab:**

* Completed SCRUM report and reflections

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

* A set of function specifications stored in the repository,
* A set of blackbox tests as test documents with test data for the functions.
* Start writing blackbox test code and store in repository. (at least 1 required)
* Start implementing functions and store in repository. (optional)
* A function-test matrix added to the repository.
* Updated Jira project to show activities and progress.

**Rubric**

|  |  |  |
| --- | --- | --- |
| Individual | Group Participation | 75% |
| Teamwork | 10% |
| SCRUM Report | 15% |
| Group | Function Specs (documented, correct, complete, well-written) | 20% |
| Test documents (well-written, complete, good test data) | 20% |
| Test Code (well-designed, written and documented) | 10% |
| Git Usage (used properly with good structure) | 5% |
| Jira Usage (creates issues, tracks progress) | 10% |
| Meets Deadlines | 10% |
| SCRUM report & reflections | 25% |

**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** |
| Gyeongrok, Oh | Reflections,  Test plan (Control Procedures, Resources and Responsibilities, Tools) | None |
| Hoi Kit, Cheung | Reflections,  Test plan (Objective, Scope, Test Strategy) | None |
| Lap Chi, Wong | Reflections,  Test plan (Environment Requirements, Execution Strategy),  Data structure analysis, Header creation | None |
| Pui Wai, Tse | Reflections,  Test plan (Dependencies, Risks, Approvals) | None |
| Sau Ching Yuki, Wong | Reflections,  Test plan (Functions to Be Tested, Suspensions / Exit Criteria, Resumption Criteria) | None |
| Yonghun Won | Reflections,  Test plan (Test Schedule, Deliverables, Documentation) | 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 3 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 |
| Enhancing collaboration and communication by involving more group members on GitHub and encouraging comments in Jira | Professor's feedback on milestone 2 highlighted the need for active participation and effective communication within the team. By increasing group involvement on GitHub and promoting comments in Jira tickets, we can improve collaboration, share insights, and ensure that all team members contribute to the project. This will lead to better coordination, knowledge sharing, and transparency, resulting in more efficient development and a higher quality final product. |
| Different deadlines set for various tasks | Setting different deadlines allows for better task prioritization and ensures that dependent tasks can be started once their prerequisites are completed. This approach helps in maintaining a systematic and organized workflow. |
| Review of potential redundant functions | Identifying and reviewing potential redundant functions is essential to avoid wastage of time and resources. Eliminating duplicated or redundant functions ensures efficiency and streamlines development efforts. |
|  |  |
|  |  |
|  |  |
|  |  |

**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, code review | 30 mins | done |
| Hoi Kit, Cheung | Reflections, code review | 30 mins | done |
| Lap Chi, Wong | Reflections, code review | 30 mins | done |
| Pui Wai, Tse | Reflections, code review | 30 mins | done |
| Sau Ching Yuki, Wong | Reflections, code review | 30 mins | done |
| Yonghun Won | Reflections, code 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 |
| take turns | meeting record |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Major Outcomes of Meeting:**

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

|  |  |
| --- | --- |
| Outcome | Impact on Project |
| Requirements of Test Case and code analyzed | Sub-tasks created to share the workload equally among team members and meet the requirements |
| Enhanced understanding | Improved clarity and alignment among team members, leading to more effective task execution. |
| Increased collaboration | Better knowledge sharing and support within the team, fostering a collaborative working environment. |
|  |  |
|  |  |
|  |  |
|  |  |

**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. In this milestone, we write the blackbox tests but not the whitebox tests. Explain why we can write the blackbox tests but not the whitebox tests.

Until this milestone it is in analysis and planning stage. The proposed list of functions has been drafted with only parameters, return types and proposed designs. No actual implementation is involved.

Blackbox test can be planned once the input and output of a function are well defined without knowing how the logic within the function is written. Based on the function descriptions and the input and expected output, the reasonable sets of testing data can be prepared before implementation. In fact, it is the best timing to design blackbox test cases when the function logics are absent, as knowledge of the codes of the functions may cloud the judgement in designing effective blackbox test cases.

Whitebox test can only be planned based on the understanding of the existing logic, which is not applicable when the implementation of function is still absent. This can be done after implementation, according to any potential logical fault found upon code inspection.

1. Explain why we need the function-test matrix and why it is important in a large project.

The function-test matrix is a tool with a structured table that indicates whether a particular test case verifies a specific business requirement. In a large project, it is important that the requirements, functions, and test cases have traceable tracks. It ensures all business and functional requirements are covered by test scenarios and test cases. The traceability enables better understanding of the system and facilitates impact analysis when changes or updates are made.

Besides, the function-test matrix helps project testing team on test planning and prioritization. It provides information about the impact of changes on specific functions or features, helps the team in making decisions on testing priorities. The team can focus more on high-risk or critical functions, ensuring that testing efforts are allocated efficiently and improve efficiency.

The function-test matrix enhances test traceability, assists in test planning and prioritization. It improves overall test management and contributes to the quality and reliability of software systems, particularly in large projects where there is a need for systematic and structured testing.

The function-test matrix ensures that all functions or features of the system are adequately tested. By mapping test cases to specific functions, it helps identify any gaps in the testing process and ensures that all critical components are covered.

As the project evolves, the function-test matrix aids in maintaining and updating test cases. When changes are made to functions or features, the matrix helps identify the associated test cases that need to be reviewed or modified. It also supports regression testing by ensuring that existing test cases are re-executed to verify that changes haven't introduced any unintended side effects.

1. Other life cycle models left team members idle while waiting for parts of the project to be completed. Describe how an agile model, like the one we are using, avoids this problem and keeps the whole team busy all the time. Does this make managing the project simpler or more complex and why?

Traditional life cycle models often lead to periods of inactivity for team members, as they typically follow a sequential process, requiring one stage to be completed before the next can begin. This means team members may find themselves waiting for others to complete their tasks before they can progress.

In contrast, the Agile model like the one we are using for our project, employs an iterative and incremental approach that helps avoid this idle time. It allows some functions to be developed, tested, and revised accordingly, instead of completing implementation of all functions and then moving to testing. Therefore, agile model emphasizes frequent communication and collaboration, ensuring the entire team can continually contribute to the project's progress. If the developer implemented some functions, the testers can immediately test these functions, while the developer can quickly move to develop other functions when the testers are still testing the previous ones, therefore optimizing team productivity.

However, since Agile is adaptive and responsive, project priorities can change frequently based on user’s or tester’s feedback, which requires a high degree of flexibility and adaptability. Agile requires regular monitoring of project progress and continuous reprioritization of tasks. Regular team meetings, often daily stand-ups, are necessary to ensure everyone is on the same page, which can require more coordination and planning by the project manager.

As a result, while the Agile model promotes continuous team engagement, it adds more complexity to project management. To successfully utilize this model, it requires effective coordination, frequent communication, and the ability to adapt to changes quickly.