# **MILESTONE 3** -- SFT221 SCRUM Report and Reflection

All students are expected to attend the SCRUM meetings and to participate. Failure to do so will result in greatly reduced grades.

**GROUP**: 2

**Members Present**:

|  |  |
| --- | --- |
| 1. Joon Dong | 4. Heqing Xu |
| 2. Doris Chai | 5. |
| 3. Xiaopeng Liu | 6. |

## 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 4 days after your lab day:**

* A set of AT LEAST 4 function specifications added to a new header file and stored in the repository.
* A set of blackbox tests as test documents (in an Excel file) with test data for the functions you created. At least 4 sets of test data are required for each function. You must have test cases for at least 6 functions (including all your custom function). Stored in the repository.
* **Create and add a C++ testing project to your solution.**
* Start writing blackbox test code (for the functions above) and store in repository (at least 1 is required for this milestone).
* Start implementing the functions and store them in repository (optional).
* A requirements traceability matrix added to the repository and shows the mapping between the requirements and test cases.
* Updated Jira project to show activities and progress.
* Completed scrum report including reflection questions answered.

**Rubric**

|  |  |  |
| --- | --- | --- |
| **Individual** | Group participation (includes GitHub commits and Jira usage) | 80% |
| Teamwork | 20% |
| **Group** | Function specifications (documented, complete, well-written, added to the project) | 10% |
| Blackbox test cases document (well-written, complete, good test data) | 15% |
| Blackbox test code (in the C++ project) well-designed and documented | 15% |
| Functions implementation (coded in the C project & well documented) | 10% |
| Requirements traceability matrix (complete and added to GitHub) | 10% |
| Git usage (used properly with good structure) | 10% |
| Jira usage (creates issues, tracks progress) | 10% |
| Scrum report & reflections | 20% |
| **Deadline** | 20% deduction for each day you are late |  |

**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** |
| **All** | **Test plan and SCRUM report 2** |  |
| **Xiaopeng** | **MS2 Reflection Q2** |  |
| **Doris** | **MS2 Reflection Q1** |  |
| **Heqing** | **MS2 Reflection Q3.B** |  |
| **Joon** | **MS2 Reflection Q3.A** |  |
|  |  |  |
|  |  |  |

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 |
| Functions | **What functions should be implemented** | **Each group member will be responsible for their own** |
| Tasks | **How to fairly distribute tasks** | **Tasks distributed** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**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 |
| Each group member will be responsible for a function | This will help simplify what each member should do as we will be contributing the same thing. (e.g. test your own function) |
| Communicate for peer review | Peer review is needed as we will be defining the implementation of these functions later. We also need to make sure we don’t come up with the same functions as each other. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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? |
| All | **SCRUM report** | **10min** | **No** |
| All | **Function definitions** | **10min** | **No** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**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 |
| Joon | Finalize scrum report, define one function, blackbox test cases for designed function, unit test for designed function |
| Doris | Reflection Q1, define one function, blackbox test cases for designed function AND for 1 given function |
| Xiaopeng | Reflection Q2, define one function, blackbox test cases for designed function AND for 1 given function |
| Heqing | Reflection Q2, define one function, blackbox test cases for designed function |
| All | Contribute to traceability report |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Major Outcomes of Meeting:**

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

|  |  |
| --- | --- |
| Outcome | Impact on Project |
| Tasks successfully distributed | **We were able to quickly distribute tasks fairly so that we could start working on our project functions right away** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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 |
| Noting down task list on notepad | **We broke down the list of tasks and subdivided those tasks. We can distribute tasks more easily if we start small and work our way up** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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 |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Reflections**:

Answer the following questions using your own words. Make sure that each answer comprises a minimum of 100 words.

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.

In this milestone our group created function signatures which contains a return type, function identifier and the parameter list. For each function, we have an overall understanding of what its supposed to do. However, we have not fully implemented each function yet, and without knowing how the function is implemented, we cannot do whitebox testing. We are not able to traverse all the possible routes of the function with only function signatures. On the other hand, blackbox testing is the perfect choice for us. We create blackbox test cases when only the logic of the functions is known to us. When we fully implement each function next week, we will be able to write whitebox test cases to make our test cases more comprehensive.

1. Explain why we need the function-test matrix and why it is important in a large project.  
     
   This is a testing that to make sure all requirements are covered by test cases. The function-test matrix improved efficiency and effectiveness in testing. Testers pay more attention to test cases and focus their efforts on areas with higher risk or complexity. In large projects involving multiple teams, the function-test matrix serves as a valuable communication and collaboration tool. It provides a common reference point for discussing requirements, test coverage, and testing progress. It also has great traceability. This means that testers can easily trace back from a specific test case. It also can give more helps in analysis, case management and test maintenance.
2. 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?

Agile model minimizes team downtime and guarantee that members are consistently engaged through iterative development, cross-functional teams, continuous communication, and customer involvement. They decompose projects into manageable segments, enabling team members to work flexibly across various tasks and stages. Agile underscores tight collaboration and swift adaptation to changes, enhancing the projects' adaptability and success rates. However, these methodologies might also heighten the complexity of project management, demanding that teams possess enhanced self-management skills and quick adaptability to alterations. Yet, when executed properly, Agile can render the project management process more adaptable and efficient.