# **Milestone 3 Scrum Report**

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

**GROUP**: \_\_\_\_\_02\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Members Present**:

|  |  |
| --- | --- |
| 1. Agenor Dionizio da Silva Junior | 4. Renan De Alencar Queiroz |
| 2. Ashley Shin | 5. Thanh Dat Trinh |
| 3. Ian Hartog | 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) | 10% |
| Blackbox test code (in the C++ project) well-designed and documented | 10% |
| Functions implementation (coded in the C project & well documented) | 15% |
| Requirements traceability matrix (complete and added to GitHub) | 15% |
| Git usage (used properly with good structure) | 10% |
| Jira usage (creates issues, tracks progress) | 15% |
| Scrum report & reflections | 15% |
| **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** |
| Agenor Junior | - Create 1 function and explain it.  - Create 4 Blackbox tests for this function.  - Fill the traceability matrix  - Answer the question 3 from Reflections |  |
| Ashley Shin | - Create 1 function and explain it.  - Create 4 Blackbox tests for this function.  - Fill the traceability matrix  - Answer the question 3 from Reflections |  |
| Ian Hartog | - Create 1 function and explain it.  - Create 4 Blackbox tests for this function.  - Fill the traceability matrix  - Answer the question 3 from Reflections |  |
| Renan Queiroz | - Choose 2 functions from mapping.c and write 4 Blackbox tests for each function.  - Add the test method  - Fill the traceability matrix  - Create and add the solution files  - Fill the scrum report and Reflections |  |
| Thanh Dat Trinh | - Create 1 function and explain it.  - Create 4 blackbox tests for this function.  - Fill the traceability matrix  - Answer the question 3 from Reflections |  |

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 |
| Division of work | We divided the MS3 in two parts. 1 person would do the work related to the functions from mapping.c file. The other 4 members would create 1 function and do the work about its own function. | Agreed |
| Organization | Each person would create and upload its own file to the GitHub. In the end, all the files would be joined to one main file. | Agreed |
| Scrum Report | Only one person would fill the report, but each person would answer the question 3 from the reflection part. | Agreed |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**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 |
| Create functions | Everyone registered in our Teams Group about the function each one was going to write about it, so we do not have duplicated functions. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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? |
| Agenor Dionizio da Silva Junior | discussion about the description of the function that would be created | 1h | Yes |
| Ashley Shin | discussion about the description of the function that would be created | 1h | Yes |
| Ian Hartog | discussion about the description of the function that would be created | 1h | Yes |
| Renan de Alencar Queiroz | discussion about the description of the function that would be created | 1h | Yes |
| Thanh Dat Trinh | discussion about the description of the function that would be created | 1h | Yes |
|  |  |  |  |
|  |  |  |  |

**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 |
| Created 4 new functions | **These functions will be needed for the rest of the project** |
| Created 4 Blackbox tests for the functions from mapping.c | **These tests give us certainty that functions are working correctly** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

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

**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. What is the difference between blackbox tests cases and blackbox test code? Explain how we use assertion in Visual Studio to execute tests.  
     
   Blackbox tests case are specific scenarios to test the functionality of software without knowing its internal code. It’s used to validate that the function behaves as expected. On the other hand, Blackbox test code refers to the actual scripts written to automate the execution of Blackbox test cases. It’s used to ensure consistent and repeatable execution of Blackbox test cases.

Assertions are used in test methods to verify that the output of a function or method is as expected. For example, in our test method, we did:

TEST\_METHOD(distance\_T001) {

Point p1 = { 0, 0 };

Point p2 = { 0, 0 };

Assert::AreEqual(0.0, distance(&p1, &p2));

}

For this test method, we are checking if the distance is equal to zero, so we used the Assert::AreEqual, and used as parameter what we are expecting (distance = 0.0) and the other parameter was the actual distance.

1. How can a traceability matrix help in the testing process?  
     
   For the x-axis in our matrix we have all the business requirements. So, this helps us to understand and see the outcome of our project. In our y-axis we have all our tests cases. When we cross the information, the first thing we can see is if all the business requirements are being covered. The second outcome we get from the traceability matrix is, in case we get a failure in our test, we know exactly which business requirement that test is covering, and we also know what the parameters are. For example, our addPackage\_T001 test covers requirements R002, R004 and R005. In case of failure, we should check which requirement was not met and review it.
2. Write down two of the function prototypes you submitted. Why did do you need each one of them and how will each one help you achieve the project needs?

- Function isValidBoxSize(): checks if a specific box size meets the acceptable shipping parameters, which are 1, 3, or 5 cubic meters. It accepts a single argument, boxSize, of type int, representing the volume of the box in cubic meters. The function then compares this value with the three allowed values (1, 3, and 5). If the provided value matches one of these, the function returns 1, indicating that the box size is valid. Otherwise, it returns 0, indicating that the size is not permitted.