# **MILESTONE 2** -- 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**: C

**Members Present**:

|  |  |
| --- | --- |
| 1.Manav Zadafiya | 4.Sunny Vavadiya |
| 2.Fenil Soni | 5. |
| 3.Ashraf Bharot | 6. |

## Milestone 2 Tasks

Some of the software for the project has already been written for you and is available on Blackboard. You must use this in your project and every team should add it to the source code for their repository. Anything in the main function is simply for demonstration purposes and can be replaced. The software you are being given has not been tested and you will need to test it.

You need to study the problem and the code provided for you and then:

* Add any new data structures you will require This will require a thorough analysis of the problem and the existing software. This should be done by creating a new header file in the directory where the rest of the source code has been placed. You do not want to go back and modify it later if you can avoid it as it will slow the project.
* Create a test plan for the project by replacing the text in the supplied test plan template with your test plan.

**Deliverables Due at End of Lab**

* Completed SCRUM report & reflections

**Deliverables Due within 48 hours of lab**

* An analysis of the problem (no written artifacts produced),
* A series of data structures created as header files and stored in the repository,
* A test plan stored in the repository.

**Rubric**

|  |  |  |
| --- | --- | --- |
| Individual | Group Participation | 75% |
| Teamwork | 10% |
| SCRUM Report | 15% |
| Group | Data structures (complete, correct and well-designed) | 20% |
| Test Plan (complete, well-written) | 20% |
| Git Usage (used properly with good structure) | 10% |
| Jira Usage (creates issues, tracks progress) | 10% |
| Meets Deadlines | 15% |
| SCRUM Report and 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** |
| **Manav, Fenil** | **Analysis problem, define and design new data Structures** | **Delayed by day** |
| **Ashraf** | **Define functions and its specifications** | **On time** |
| **Sunny** | **Test Plan** | **On time** |
| **All** | **Reviewing each other’s work** | **On time** |
|  |  |  |
|  |  |  |
|  |  |  |

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** | **Designing new data structure** |
| **Reason for delay or block** | **Confusion with deadlines** |
| **Impact on Project** | **No Major Impact as we catch up it quickly, but overall team performance affected** |
| **Solution or work-around** | **Follow strict deadline and give some extra effort for something left out.** |
|  |  |
| **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 |
| MS2 | **What mistake did we make in milestone 1? And what are the areas of improvement.** | **what we can improve to increase productivity.** |
| Data Structure | **What is role of these newly designed data structure?** | **More deeper understanding of problem** |
| Test Plan | **How we are going to execute test** | **Test plan analysis** |
| MS 3 | **What is requirement of milestone 3 and how work will be divided?** | **Planned MS3** |
| deliverables | **Review submissions and deliverables** | **Document major events** |
|  |  |  |
|  |  |  |

**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 |
| MS2 | As it is important to identify our weakness which help us as team to co-ordinate and can assist with each other when needed. Moreover, it will increase overall team performance and moral. |
| Data Structure | It is important that every team member understand newly define data structure as we all are working around it. |
| Test Plan | As a team, understanding how software is being tested can help us to know the actual task of program and how it is expected to work. |
| MS 3 | Understanding MS3 helps us to prepare for upcoming tasks and forces us to come up with new and better ideas to get work done. |
| Deliverables | Understanding what needs to be deliver helps us to document important events. |
|  |  |
|  |  |

**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? |
| Manav | **SCRUM REPORT and Points of discussion and plan for MS3** | **30 min** | **Completed** |
| Fenil | **Keep record of important points in meeting and help in completing scrum report** | **30 min** | **Completed** |
| Sunny | **Checklist all tasks that needs to be completed from ms1-scrum report and help in completing scrum report** | **30 min** | **Completed** |
| Ashraf | **Review work done by team members for deployment and help in completing scrum report** | **30 min** | **Completed** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**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 |
| Manav | Converting pseudo code into working program |
| Sunny | Creating pseudo code for functions |
| Fenil | Defining function’s parameter, return type and its descriptions |
| Ashraf | Test each function by unit testing |
| All | Working together to solve problems |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Major Outcomes of Meeting:**

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

|  |  |
| --- | --- |
| Outcome | Impact on Project |
| ms-2 almost completed | **Meet the project requirement and worked almost as planned.** |
| Problems identify | **Help us to improve for upcoming milestone** |
| ms-3 discussed | **Help us to prepare for next meeting** |
|  |  |
|  |  |
|  |  |
|  |  |

**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 |
| Milestone 1 | Contribution from all team members |
| Meeting and work to be done | Dedication of team members towards project |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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 |
| Time management | Due to a bit of confusion with deadlines, we were running late but we caught up as quickly as possible by making some extra efforts. From next time we will double check the deadlines and work accordingly to meet projects requirement. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Reflections**:

1. In this milestone you have been asked to analyze a problem and design software(functions) to complete the solution without actually writing the software.
   1. Is this process more difficult than just writing the software to complete the project? If so, why is it more difficult? If not, why is it easier than just writing the software?

We believe this process is as difficult as writing software. Although designing functions is one of challenging tasks as the software will be written on bases of these functions only, converting it to working code will came up with the same level of challenges as implementing logic into code is not easy. Hence, we consider both analyzing problem and designing functions for solution as difficult as converting it to actual working program.

* 1. Describe two advantages of developing software in this manner rather than just moving on to writing the functions without writing specifications first.

1. Improved Clarity and Precision:

When we analyze a problem and design software functions beforehand, it helps us to create a clear and detailed blueprint of the software's functionality. This approach ensures that whoever is working on project, including developers, testing team have a common understanding of the project's objectives. Precise specifications provide a well-defined road map that helps in identifying potential issues, ambiguities, and edge cases before any code is written. This clarity and precision lead to fewer misunderstandings, reduced scope creep, and ultimately a more successful and efficient software development process.

1. Reduced Development Time:

Taking the time to analyze a problem and design software functions thoroughly can help in identifying potential bottlenecks, inefficient processes, and unnecessary features early in the development cycle. By addressing these issues at the design stage, like we did, can avoid costly and time-consuming rework during the coding phase. In the end, this leads to a more cost-effective and efficient development process as we already have plan and just have to execute it.

To sum up, this approach of analyzing a problem and designing software functions before coding not only enhances the clarity and precision of the project but also results in reduced development costs and time, ultimately leading to a more successful and efficient software development process.

1. Why is it a good idea to create a test plan? Describe at least 3 advantages of test plans.

Creating a test plan is a fundamental step in the software development and testing process, and it offers several advantages.

Structured Testing Approach:

A test plan provides a structured and organized approach to testing. It outlines the scope, objectives, and strategies for testing a software application or system. This structured approach helps testing teams and developers understand what needs to be tested, the testing methodologies to be employed, and the expected outcomes. It ensures that testing is conducted systematically, reducing the likelihood of overlooking critical test cases, features, or requirements.

Risk Reduction:

Test plans help in identifying and minimizing risks early in the development process. By defining test objectives, scope, and criteria, a test plan enables project stakeholders to assess the potential risks associated with the software project. It allows teams to prioritize critical areas for testing, allocate resources accordingly, and focus on high-impact test cases. Moreover, a well-documented test plan serves as a reference point for evaluating whether testing goals are met, which aids in risk assessment and management throughout the project.

Communication and Collaboration:

Test plans serve as a communication tool among various stakeholders, including developers, testers, and project managers. They provide a common reference point for everyone involved in project, ensuring that there is a shared understanding of the testing process and its objectives. Test plans also facilitate collaboration by defining roles and responsibilities, specifying the test environment and data requirements, and outlining the schedule and timeline for testing activities. This clarity in communication and collaboration can lead to smoother project execution and help in avoiding misunderstandings and conflicts.

In addition, test plans can also improve test coverage, aid in resource allocation, and provide a basis for tracking progress and reporting results. Overall, a well-crafted test plan is an essential tool for effective software testing and quality assurance, as it helps ensure that the software meets the required standards and fulfills its intended purpose while managing risks and enhancing collaboration.

1. Describe the process you used to analyze and understand the existing software.

The following steps outline the process we used to analyze and understand existing software:

Reading and understanding Documentation:

Start by reviewing available documentation, i.e. project-student-2023.v2.pdf, related to the software. These documents provide us with valuable insights into the software's intended functionality, design, and purpose. By reviewing it we understand intend of software and how it works and what problem it is addressing.

Review Source Code:

Analyzing the source code to understand how the software is implemented. Paying attention to the code structure, naming conventions, comments, and coding patterns. How it works and what are the features.

Trace Functionality:

Create a rough map of the software's features and processes. This helps in visualizing how different components of the software interact and understanding the core functionality.

Identify Logic:

Focus on the logic of the software. This includes understanding how the software processes data, enforces constraints, and manages workflows.

Performance Assessment:

Assess the software's performance characteristics, such as response times, resource utilization, and scalability. This is vital for identifying potential performance bottlenecks and optimization opportunities.

Report and Recommendations:

Document our analysis in a comprehensive report. Include findings, recommendations for improvements or updates, and a rough roadmap for any necessary actions.

After analyzing the software, we learn that analyzing and understanding existing software is a complex and iterative process that requires a combination of technical skills, domain knowledge, and attention to detail. The depth of analysis may vary based on the context of the assessment.