**Software Evolution**

Developers Inc.

**Assignment # 3**

**Comp3520**

Jagjit Singh Bilkhu - T00533766

# Table of Contents:

[**Table of Contents:**](#_908xr6nkiee5) **2**

[**Summary:**](#_c8l5r4cyfyvf) **3**

[**Software Evolution Process:**](#_nniidlhmup82) **3**

[Feature Implementation And Bug Fixes:](#_bp5cm2ov0l2j) 4

[Environmental Adaptation:](#_hi4zgk50evji) 4

[**Conclusion:**](#_l6vztxyq4jtv) **5**

[**References:**](#_imvvyx63bxz7) **5**

# Summary:

Developers Inc. focuses on developing and delivering native mobile applications to provide cost-effective and robust solutions to the needs of our customers from different business fields. To develop and implement the solutions, we focus on a combination of scrum agile development with bits of extreme agile programming. The agile flavours allow us to break down the projects into smaller projects that are completed iteratively in sprint cycles.

For evolving and servicing our applications, we will group and mix sprints based on a specific focus area such as feature implementation, bug fixing, environmental adaptation/hardware and tool improvements. The feature implementation and bug fixes would be done interchangeably in the same sprint and the adaptations/reengineering would be done in special sprints dedicated to them when needed.

# Software Evolution Process:

The development process would be a mixture of the the types of sprints where they would be done interchangeably for example: We would begin by implementing some features and then do some refactoring to reduce the repeatedness of code and improve the readability of the code and then go back to doing more development sprints. To keep control and control of progress(i.e. To be able to roll back) we will be using GIT version controlling system. Incase any changes made have to be got rid of, GIT would be used to go back to previous versions during maintenance. The timeline of the sprints would be such that implementation and bug fixing sprints take about 85% of the whole time and the adaptations would take 15% of the time followed by the reengineering and environmental adaptation.

The length of the individual sprints of each type would be as such: The implementation sprints would be ideally 2 to 4 weeks long, and the re-engineering and environmental adaptation of system environment would be longer taking 4 to 5 weeks and would ideally be done towards the end days of a project but would be done rarely or when needed most.

## Feature Implementation And Bug Fixes:

These will be the major sprints during which the process of coding, bug fixing, testing, verification and validation will take place. During these sprints, we would be focusing on implementing new features, bug fixing and error correction, and verification of the progress made. Also, after some features and bug fixes are made, refactoring of code would be done to reduce and control the complexity. The progress made during these sprints would be recorded and monitored using JIRA where the tasks completed and features implemented would be added on to the application. Regular releases would also be done to provide updates to the clients and get validation feedback from them.

During these sprints, the majority of time spent would be feature implementation and the rest would be devoted towards bug fixing and error corrections. To specify, about 3 weeks of feature implementations and one week of bug fixing and error correction.

However, we will occasionally devote some time to do some refactoring and code cleanup to reduce complexity of code and keep it readable.

## Environmental Adaptation:

These sprints would be done if any major changes are made by the tools or frameworks used in development such as obsolete libraries, changes in languages eg android changes to Kotlin and iOS changed to Swift or if the clients require major redesign and modifications of the system. Also, incase the number of users using the applications rises above what the current servers can support, the adaptation would enable us to scale the application.

These sprints would be ideally scarce in the development process but when done, they would last longer than other sprints like 6 to 8 weeks after which strong emphasis would be given to testing and validation of the applications.

# Conclusion:

Overall, this software evolution process would be ideal for smaller software applications. They would allow us the flexibility to adapt to changing requirements by clients and also have time to verify and validate the progress. Refactoring regularly would prolong the life and readability of the code thus making it easier to change or add on to the code in the future. Also, after every refactoring sprint, strong testing would be done to ensure the consistency in the behaviour and functionality of the application.

Towards the phasing out periods, the environmental adaptation would enable generalising, adopting and scaling the application for larger number of users or larger amounts of data. Also, the generalising to allow reusability of a similar infrastructure to develop and deploy similar apps.

# References:

Software Maintenance Overview - Date unknown.

<https://www.tutorialspoint.com/software_engineering/software_maintenance_overview.htm>

Software Evolution - 24 November 2017

<https://en.wikipedia.org/wiki/Software_evolution>