**Student Name:** Kevin Wadera

**Course:** SOEN-6841 – Fall2024 – Software Project Management

**Journal URL:** [40261875\_Learning\_Journal\_4](https://github.com/kevwad/SOEN-6841/blob/main/Learning_Journal_4_40261875.docx)

**Dates Rage of activities:** 28- October to 08-November 2024

**Date of the journal:** 09-Nov-2024

**Key Concepts Learned**

This week I learned the importance of closure process for a software development project. In addition, I also learned about various components of the closure process which include much more than delivery of the product as it includes, among others, user training, releasing resources, and lessons learned. For an organisation, it is also important to document all project related data which are structured relevant and clean. This serves as baseline for future projects.

Next, I learned about the life-cycle management of a software project. We learn about the various phases during the development which broadly falls into the following categories: Requirements, Design, Construction, Testing, Release and lastly Maintenance. There are different methodologies in which these phases can be managed differently. The simplest one is the ‘Waterfall Model’ where there is a strict delineation and handover between the mentioned phases. This tends to be very rigid in its implementation but are best suited for very large projects where requirements are well-defined from the very start and are unlikely to change during the life cycle. On the other hand, we have Iterative development which allows the software vendors to reduce the risks by continuously refining the software based on customers feedback. We also learn about other models including Incremental, and Extreme Programming among others. We also learn about how Testing and Quality assurances can vary for each of these different models. Thus, I got an overview of various methodologies and steps of software development process.

**Application in Real Project**

Post the completion of a software project, the closure process is a vital as it ensures that all the aspects of project are finalized. Additionally, for a software manager, it becomes important to document the artifacts produced during the software development as it helps in improving transparency for the customer as well as it can be used as a baseline for future projects.

From the perspective of Software Manager, effective life-cycle management is crucial for ensuring the successful delivery of the project. Also, based on the type of software project and customer requirements, appropriate methodologies can be selected that utilises the various phases in an effective manner.

**Peer Interactions/collaboration**

I had the opportunity to interact with my fellow student who has experience of working in an Agile environment. I learned about the process practised by them as well as tools used by them in Agile development. I learned about how tools like ‘**Jira’** are used in companies to translate the requirements of customers to actual implementation during a sprint. I learned that a developer can find all the information related to the task in their hand like requirements, design, expectation, testing information, acceptance criteria, among others could all be found in a single place.

**Challenges Faced**

Having limited experience, it was difficult to visualise how different methodologies are implemented in real life projects. However, learning from my peer about their experience in Agile environment helped me grasp the fundamentals and I’m sure that it will provide me a good starting point when working in professional environment.

**Personal Development Activities**

I intend to learn more about the Agile methodologies and its practical application to prepare myself for industry.

**Goals for the Next Week**

I intend to apply my learnings to the group project. Similar to the learnings about breaking the work by its component and iterating through them to achieve the result, I plan to break down the project requirements into its components and divide the work among peers so that tasks could be covered concurrently and then integrated at the end.