Learning Journal - IV

Student Name: Dakshina Ravishankar
Course: SOEN 6841- Software Project Management
Journal URL: https://github.com/dashi1601/SOEN6841LJ_40267664/

Dates Range of activities: 4/11/2024 – 8/11/2024
Date of the journal: 4/11/2024

Key Concepts Learned:	**Application in Real Projects:**	**Peer Interactions:**	**Challenges Faced:**	**Personal development activities:**	**Goals for the Next Week:**
Project closure involves steps that are executed when the project is coming to an end. It includes assessing whether everything in project went according to plan, archiving project data, lessons learnt and releasing unused resources. Each artifact produced is assessed and evaluated to improve future software projects	Closure fosters continuous improvement of software project execution by finding areas to work on so that future projects will not undergo the same problems. This step does not directly affect the software product but may help improve future products. Further, configuration management system helps in archiving all versions of artifact.	Discussed with my classmate who was a project manager about how closure is implemented in real projects. Documentation of analysis is very important at this stage; it must be consistent and accurate.	Difficult to understand how releasing software resources (tools, licences, etc.) worked in real like projects.	Answered exercise questions from textbook from chapter 8 to improve understanding on closure	Review case study from textbook
Lessons learnt facilitate better ways of doing a task, areas to improve on, figure out what went wrong, better negotiation with customers and what techniques worked better than others. It also helps assess risks that were handled better than others and what contingency or mitigation plan worked the best for a risk.	Lessons learnt are formulated by assessing different versions of artifacts produced in project lifetime. Software measurements, analysis and visualisation techniques can be used to form lessons learnt. These are then archived and can be referred to during future projects to improve decision making.	-		Referred to sample lessons learnt template	Read case studies from textbook
Software lifecycle refers to the sequence of process executed while developing a software. Software lifecycle models include waterfall, iterative, incremental, etc. Each model is based on its unique principal thus, requiring careful analysis before choosing a software lifecycle model for a software project. Selection of lifecycle	Waterfall models are used for business-critical projects and when most steps and requirements are determined and will not encounter much changes. On the other hand, iterative and agile models are suitable for projects that need to delivered in small increments and is susceptible to change. It	Discussed with my classmate who was a project manager about how a decision is made on the type of model to use. It usually involves comparing similar past projects and reviewing empirical studies that suggest	Difficulty in understanding the why sometimes waterfall model is preferred over iterative model that seem to	Referred to some research papers that compared different aspects of iterative and waterfall model on domain	Read case studies from textbook

model changes every aspect of software project management.	is also ideal when requirements are not clear at the start.	what model is better for a particular use case.	have more advantages .	specific software project.	
Software lifecycle metrics are measurement strategies used to measure software process and product attributes. Essentially, they are used to measure the work products. It also involves finding relationship between different work products, degree of association and impact, etc.	All the versions of work products are stored in configuration management. Lifecycle metrics helps assess the progress and quality of processes and products. It is used to determine whether the quality metrics estimated in software project planning phase is going according to plan.	-	Difficulty to understand process measurement in iterative model as each process is executed more than once.	Read through a common metrics used to measure processes and products and how it is measured in real time.	Read case studies from textbook
There are two types of software requirements, functional and non functional (quality attributes). Requirements are finetuned overtime by following requirement development steps like interacting with customers and stakeholders frequently.	A detailed SRS document is drafted that mentions a list of all functional and non-functional requirements, its measurability, why the requirement arose, who drafted the requirement, etc. This document is updated as requirements are finetuned, thus, multiple versions exist.	Discussed with my classmate who was a project manager about whether amount of refinement possible depends on the lifecycle model. Yes, it does in some cases.	Difficulty in understanding measurability of functional requirements.	Referred to sample SRS documents for web-based software projects.	Read case studies from textbook

Final Reflections:

Overall Course Impact:
Software closure is crucial in software project as it ensures success of future project success. It involves analysing all software artifacts, forming lessons learnt and closing unused resources like budget and resources. Lessons learnt help understand what went wrong and how this situation can be avoided in future projects. Further, software lifecycle models have an impact on project outcome as different projects work better for different lifecycles. Finally, software requirements are developed and refined overtime to ensure maximum customer satisfaction.

Application in Professional Life:
By applying the skills and knowledge from these chapters, I can improve my effectiveness in project closure process and requirement development. Additionally, have an understanding as to what type of lifecycle model will be suitable for a software project. These practices will not only streamline project workflows but also create a more resilient and responsive team environment.

Peer Collaboration Insights:
I learned how theory applies to real-life projects as each project is unique, requiring thorough analysis of its complexity before making any critical decision. Requirement development is crucial as any misunderstanding in requirement may lead to project failure.

Personal Growth:
I gained a clearer understanding of project closure, software lifecycle models and software requirement development, recognizing that it is not straightforward and that project managers play a critical role in a project's success.