**Learning Journal**

**Student Name:** Amal Saji

**Course:** Software Project Management (SOEN – 6841)

**Journal URL:** https://github.com/amalsaji25/SOEN-6841-Software-Project-Management

**Week4:** 11/02/2024 – 17/02/2024

**Date:** 17/02/2024

**Key Concepts Learned:**

During the fourth week focus was given to understand Project Plan Planning in a software project.

* In a Software project we have to balance the quality, budget, timeline and the value it brings to the company. Knowing the project’s goal, like boosting employee productivity helps tailor the software to meet those needs. In outsourced projects, the service provider aims for profit, so this is to be factored in the budget planning.
* Software project planning starts with rough initiation, with estimates refined later once details emerge. Two planning approaches are top down, having fixed release date, features that fit the team and bottom-up with team estimates based on requirements, set release date. Large outsourced projects use software engineering practices and service level agreements (SLA) for defined processes, good quality, competitive costs, and acceptable schedules. Project plans dice deeper into various aspects like risk, resources, tasks and communication.
* For both software vendors and companies with market deadlines, top-down project planning is the best approach. With a fixed release date in mind, teams fit features in the allotted time. It requires upfront details like project start/end dates, budget, and requirements. Outputs include plans for managing suppliers, configuration, communication, risk and resources.
* For large software projects, bottom-up planning is the best suited approach. Initially, the team identifies needed tasks and maps out the project's potential flow. It is crucial to gather the information, focusing on scope, requirements, and service level agreements (SLAs). Then, a software engineering model is picked (waterfall, agile, etc.) and development tasks are defined. With these inputs, the project plan is built, encompassing supplier management, communication, configuration, risk, resources, and more. The final plan lays out project duration, cost, work breakdown structure, and key dates.
* A project plan lists all tasks with start and end dates. However, this raw format makes it hard to see dependencies, critical paths, and milestones. To improve readability, related tasks are grouped under "phase" headings. The last task in each group signifies a milestone. Project management software like Microsoft Project allows to expand and collapse these groups for easier navigation and understanding of the plan's overall structure.
* Software projects has fluctuating resource demands. Design and requirement phases require fewer people, while construction and testing phases demand a considerable people. Additionally, specialized skills often limit cross-functionality. To address this unevenness, concurrent engineering models have emerged. By creating separate teams for construction and testing that work simultaneously, these models help compress project timelines.
* For outsourced projects, a supplier management plan becomes crucial. This plan ensures Service Level Agreement (SLA) compliance and oversees the quality of outsourced work. Integration is a key concern, so a central build approach is recommended where outsourced teams integrate their code early on, minimizing future integration issues.
* In multi-team projects, configuration management is key. A centralized system ensuring everyone has the same code and documents minimizes rework. Robust security and access control are essential for secure work and data protection.
* Successful communication in any project heavily relies on its structure, customer approach, and supplier interactions. A communication management strategy is crucial, outlining who, what, and when information needs to flow between stakeholders. Utilizing project templates across teams fosters consistent and efficient communication, minimizing confusion and streamlining collaboration.

**Reflections on Case Study:**

In the case study we can observe the feature selection clashes often occur between marketing and development in the SaaS company. To resolve these, the CTO creates a preliminary feature list during annual planning. Before each iteration (3 months), marketing prioritizes features, and the project manager balances resource availability and effort to determine which features make the cut. This locked list becomes the development focus for the upcoming iteration. Essentially, these features get prioritized and delivered within a defined timeframe of three months. The traditional waterfall model of software development is plan-driven, while pure agile models are not. This can create challenges for planning and budgeting iterations. The vendor used a time-boxing concept to address this, with a fixed release date and prioritized feature lists. This allows for flexibility and responsiveness while still enabling planning and resource allocation.

**Collaborative Learning:**

This week we were able to discuss and delve into project planning for software development, comparing top-down and bottom-up approaches. We acknowledged fluctuating resource needs across phases and the importance of clear communication strategies, especially in multi-team projects. Finally, we explored how agile projects can leverage time-boxed iterations with prioritized features to achieve flexibility while maintaining resource and budget awareness. The best approach depends on the project's context and goals. By understanding these methods and their issues, we can navigate the complexities of software development with informed decisions.

**Further Research/Readings:**

Project planning is not just about timelines and budgets, it's about fostering collaboration, anticipating risk, and embracing adaptability. The best plan can shift as the project evolves, so agility and communication are key ingredients for success in a project planning. In project planning it requires to embrace diverse perspectives and involve stakeholders early into discussions.

**Adjustments to Goals:**

I will need to figure out few project planning case studies and understand which approach suites here the best, i.e. if it’s the top down or bottom-up approach. I also need to explore some specialized planning techniques for mobile app development, aiming to apply them to a personal project I am working on.