Student Name: Mohammed Huzaifa
Student ID: 40242080
Course: SOEN 6841 - Software Project Management
Journal URL: https://github.com/MHUZAIFA/SOEN-6841-Software-Project-Management
Duration: Feb 6 – Feb 20
Date: 20/02/2025
Learning journal: 3

Key Concepts Learned:

- **Three process levels**: Organization-level processes → Project processes → life-cycle processes.
- Software projects focus on making software design based on customer requirements, implementing source code and testing the software whereas Software maintenance focuses on already existing projects, removing defects, adding functionality, porting software on some other OS and more
- When to use which estimation technique:
  COCOMO - Use when Current project information is known
  FPA - Use when information for both Current and Previous projects is available
  Wide Band Delphie - Use when you have some or all information about the current project
  If we have **no data available** for current or previous projects, then it is impossible to estimate the effort.
- **Causes of risks** include bad negotiations, scope creep, cost and quality constraints, unrealistic estimates, poor management, resource unavailability, disinterest, attrition, and human error.
- **Resource risks** include the high cost of team members, availability issues due to health or attrition, and knowledge gaps. To mitigate these, use pipelines for quick replacements, keep reserve time for delays, and maintain a knowledge management system to store critical team knowledge.
- Product quality may suffer due to poor software design, construction, or defects arising from complexity, large integration interfaces, or numerous changes. To address this, integrate **quality checks** into the project schedule and ensure strict adherence to peer reviews, code reviews, and other formal quality reviews.
- **Best practices** for configuration management include centralized systems, secure access, continuous builds, easy branching, and audit tracking.
- If someone breaks the build, the configuration management system will not allow other developers to check their code until the build is rectified. Done using automatic **smoke testing** tools. If the build passes/fails, an email is sent to the developer and other listed recipients.
- A configuration management system stores build files, work products, documents, reviews, and reports generated throughout the software development lifecycle, with multiple versions of each document.
- **Risk reduction leverage** is a metric used to evaluate the effectiveness of risk mitigation strategies by comparing the reduction in risk exposure to the cost of implementing a mitigation or risk reduction countermeasure.
- **Risk Reduction Leverage (RRL) equals zero**, which indicates that the mitigation strategy has not reduced the risk exposure at all.
- Risks have a **"repel effect**". If quality doesn't meet the standard (quality risk) at the design step, this requires rework which leads to project schedule overrun risk. Similarly, a lot of bugs were found in testing (quality risk). This can overshoot the budgeted time. Another eg. A team member gets sick.
- **Top-Down Planning:** Start by defining the overall project and its timeline, then break it down into smaller tasks with their durations based on the larger task's allocated time.
- **Bottom-Up Planning:** Begin by estimating time durations for smaller tasks first, then aggregate these to calculate the time for larger tasks or the overall project.
- The schedule can be presented in 2 forms: **Calendar-based (Gantt Chart)** and **Activity networks (PERT Chart)**

Application in Real Projects:

- **Estimation Techniques**: In large-scale software projects, such as those undertaken by companies like Netflix or Amazon, selecting appropriate estimation methods like COCOMO or Function Point Analysis is crucial for accurate resource allocation and risk management.
- **Configuration Management**: Tech giants like Google and Microsoft utilize robust configuration management systems, including version control tools like Git and continuous integration platforms, to maintain code quality and streamline development processes.
- **Risk Management**: Companies like Slack have faced challenges like resource unavailability during rapid scaling. Implementing strategies such as maintaining reserve resources and cross-training employees can effectively mitigate these risks.

Peer Interactions:

- Collaborating with peers on software project management topics has been insightful. In group discussions, we debated the pros and cons of various risk management strategies, such as which types of risks should be classified as manageable versus unmanageable.
- One peer suggested a case where they had worked on a project with limited resources and how they used **bottom-up planning** to account for every small task before aggregating the total effort required. It helped me see how these methods could be tailored to different types of projects based on scope and team structure.
- Another peer highlighted how they have implemented **automated build checks** in their past projects to ensure code quality, which aligned with my own learning on configuration management.

Challenges Faced:

One of the challenges I faced during this period was understanding the complexities of **software maintenance**. I found it difficult to grasp how to balance the need for adding new features while maintaining a robust system. Some of the challenges encountered in real-world projects, such as managing **scope creep** and ensuring product quality during maintenance cycles, made me realize how critical it is to plan for the long-term sustainability of a software system. This realization motivated me to research more on the topic and look at case studies from companies like **Microsoft** and **IBM**, which frequently perform long-term software maintenance.

Personal development activities:

- Watched videos on how to present a project pitch effectively, focusing on how to communicate the value proposition and technical aspects of the project in a clear and concise manner.
- Read articles on configuration management tools such as Jenkins, GitLab, and Docker, understanding how these tools ensure smooth software deployment and testing across various environments.

Goals for the Next Week:

- Refine and prepare for the presentation for the project pitch.