Learning Journal 2

Student Name: Dev Bhupendra Pandya
Course: SOEN 6841: Software Project Management
Journal URL: https://github.com/DP8801/SOEN_6841_SPM.git
Dates Rage of activities: 3rd Feb, 2025 to 7th Feb
Date of the journal: 8th Feb, 2025

Key Concepts Learned:

Chapter 5: Configuration Management (CM)

So, this week, I learned about Configuration Management (CM) and why it's such a big deal in software projects. Essentially, CM is all about managing changes efficiently, ensuring every version is traceable, and preventing the chaos of uncontrolled changes. I found out that without CM, you can end up testing the wrong version of code, losing track of documentation, or even deploying broken features. That's a nightmare for any project!

Core Parts of CM:

1. Configuration Identification: Establishes baseline components.
2. Configuration Control: Manages change requests and approvals.
3. Configuration Status Accounting: Keeps track of all changes.
4. Configuration Auditing: Ensures that everything matches project requirements.

Biggest Takeaway - Change Impact Analysis:

One cool thing I learned is that every change request should go through an impact analysis before approval. This helps determine how a change might affect the entire system, which reduces risks. [Link]

Real-World Example:

A well-documented failure due to poor CM practices is the Knight Capital Group trading disaster, where a mismanaged software update caused a $440 million loss in just 45 minutes! This case highlights the importance of rigorous version control and change tracking.

Chapter 6: Software Project Planning

Okay, planning a software project isn't as simple as setting deadlines and hoping everything works out. It's a continuous process from start to finish.

There are two main approaches:

- Top-Down Planning: Start with the total project duration, then break it into phases.
- Bottom-Up Planning: Estimate smaller tasks first, then combine them into an overall schedule.

Work Breakdown Structure (WBS):

I finally understood why breaking a project into smaller, manageable tasks is so important. WBS helps maintain logical dependencies, meaning you can see which tasks rely on others. This prevents bottlenecks and keeps projects flowing smoothly.

Project Scheduling Techniques:

- Gantt Charts: Simple, visual representation of project timelines.
- Critical Path Method (CPM): Identifies the longest sequence of dependent tasks.
- Goldratt's Critical Chain Method (CCM): Focuses on managing resource dependencies and removing unnecessary buffers.

Cool New Concept - Critical Chain Method (CCM):
Unlike CPM, CCM optimizes scheduling by identifying and eliminating unnecessary time buffers. [Link]
Real-World Example:
While working on an ETL pipeline project, I applied WBS by structuring the workflow as: Data Extraction → Transformation → Loading → Testing
This structure allowed the team to work in parallel and reduce waiting times between phases.

Application in Real Projects:

- CM in DevOps: I realized how important CM is in modern CI/CD pipelines. Tools like Git, Jenkins, and Docker help automate CM by tracking versions and managing deployments.
- Project Planning in Real Life: I applied WBS while planning an ETL pipeline project. Breaking the project into Data Extraction → Transformation → Loading → Testing helped structure my workflow efficiently.

Peer Interactions:

I had an interesting discussion with classmates about whether **manual tracking** of software versions can ever be eliminated. Some argued that **GitOps** (managing infrastructure as code) fully automates CM, but others pointed out that **manual reviews** are still crucial in regulated industries.
Project Planning Failures:
We explored NASA's Mars Climate Orbiter failure, which was caused by a miscommunication between teams using different unit systems (metric vs imperial). This highlighted the need for clear documentation and configuration control.

Challenges Faced:

1. Estimating Project Timelines: Struggled to decide when to use top-down vs. bottom-up planning. Turns out, top-down works best for fixed deadlines, while bottom-up is ideal for Agile teams.

Personal development activities:

Researched Feature Flags & Trunk-Based Development as alternatives to traditional CM.
Took a Coursera course on Project Scheduling Techniques.
Practiced making Gantt charts using JIRA Roadmaps and MS Project.
These are the videos are:

1. **https://www.youtube.com/watch?v=r_Jf_lRpYDc**
2. **https://www.youtube.com/watch?v=4oDLMs11Exs**

Goals for the Next Week:

1. Work on a mock software development timeline using CPM & Gantt charts.