**Learning Journal – Week 3**

**Student Name:** Amanpreet Kaur

**Course:** Software Project Management

**Journal URL:** <https://github.com/Amanpreet1304/SOEN6841-Software_Project_Management>

**Dates Rage of activities:** 30th January 2025 – 5th February 2025

**Date of the journal:** 2nd February 2025

**Key Concepts Learned:**

This week, I explored Configuration Management (CM) and Project Planning, which are essential for maintaining software integrity and ensuring smooth execution of projects. Key learnings include:

* **Configuration Management (CM):**
* Ensures version control, systematic change management, and documentation tracking to maintain software consistency.
* Prevents issues like version conflicts, feature loss, and reappearance of previously fixed bugs.
* Tools like Git, SVN, and Perforce help maintain version control.
* **Change Management & Configuration Control:**
* Involves structured handling of software changes, ensuring impact analysis and approval before implementation.
* **Software Configuration Control Board (SCCB)** reviews and approves changes to avoid unintended consequences.
* **Project Planning & Scheduling:**
* Defines scope, milestones, resource allocation, and communication plans to ensure successful project completion.
* **Work Breakdown Structure (WBS)** is used to divide projects into manageable tasks.
* **Scheduling Techniques:**
* **Top-Down Planning:** Defines the overall project timeline first, then assigns tasks within it.
* **Bottom-Up Planning:** Estimates individual task durations first, then sums them to determine the project timeline.
* **Critical Path Method (CPM):** Identifies the longest sequence of dependent tasks, determining the minimum project duration.
* **Risk & Resource Management:**
* Proper resource allocation ensures team members work efficiently without overload.
* Supplier Management involves handling external vendors and service providers effectively.

**Application in Real Projects:**

The learned concepts can be applied to our project - Intelligent Tutoring System (ITS) or any real time project as follows:

* **Applying Configuration Management:**
* Since our ITS project will undergo multiple updates, we plan to use GitHub for version control and JIRA for tracking changes.
* CM practices will help ensure that each team member works on the correct software version, avoiding conflicts when merging code.
* **Change Control Process for ITS:**
* We plan to implement a formal change request process to track any updates or bug fixes in our project.
* Any major change will require approval and proper documentation.
* **Project Planning for ITS Development:**
* Work Breakdown Structure (WBS) can be used to divide the project into various tasks and CPM will help identify critical tasks, ensuring dependencies are properly scheduled.
* **Scheduling Challenges & Solutions:**
* Since the AI-based tutoring model is complex, we can allocate extra buffer time for algorithm fine-tuning.
* Supplier management will be crucial as we rely on third-party APIs for speech recognition and NLP processing.

**Peer Interactions:**

* **Discussion on Configuration Management Tools:**
* Debated whether Git or SVN would be a better choice for version control in our project.
* Some team members suggested Perforce for handling large datasets, but we decided GitHub is more suitable for our needs.
* **Project Planning Debate:**
* Compared Top-Down vs. Bottom-Up Planning for software projects.
* Top-down was preferred for defining project milestones, while bottom-up was found useful for estimating task durations in Agile sprints.

**Challenges Faced:**

* Initially, I struggled with how Configuration Status Accounting works in tracking software versions. After reviewing real-world case studies, I realized its importance in traceability and compliance tracking.
* Our team faced challenges in assigning developers to different tasks while ensuring workload balance. We resolved this by creating a Gantt Chart to visualize task dependencies.

**Personal development activities:**

* **Explored Git Workflows:**
* Studied branching strategies (Git Flow, Feature Branching) to implement efficient code management in our project.
* **Learned CPM & Gantt Chart Tools:**
* Practiced creating critical path diagrams using Microsoft Project to understand task dependencies.
* **Analyzed Real-World Configuration Failures:**
* Researched NASA’s Mars Climate Orbiter failure, which was caused by a configuration error (unit mismatch between metric & imperial systems), highlighting the importance of CM in mission-critical software.

**Goals for the Next Week:**

* Analyze the Work Breakdown Structure (WBS) for better task tracking.
* Explore Gantt Charts to schedule project milestones effectively.
* Research best practices for managing software change requests in Agile projects.