Student Name: Zaziba Ahmed
Course: Software Project Management
Journal URL: https://github.com/ZazibaAhmed/SOEN6841-Learning-Journals
Dates Range of activities: 16th Jan 2025 – 30th Mar 2025
Date of the journal: March 30, 2025

---

1. Key Concepts Learned

Over the semester, we have covered a wide range of critical concepts across 14 chapters that laid a strong foundation for understanding the lifecycle and management of software projects:

- **Chapter 1 – Introduction to Software Project Management**
  Examined roles such as Project Manager, Scrum Master, and Stakeholder; defined what qualifies as a project; and explored the scope of software project management.
- **Chapter 2 – Project Initiation**
  Discussed the project scope, SMART objectives, project charters, and initial task scheduling. This established the foundation for understanding how software projects are initiated in a professional manner.
- **Chapter 3 – Effort and Cost Estimation**
  Learned estimation methods including COCOMO II, analogy-based estimation, and function point analysis. Highlighted the impact of these models on resource planning and budgeting.
- **Chapter 4 – Risk Management**
  Explored types of risk, probability matrices, impact analyses, risk classifications, and mitigation techniques. I became more conscious of planning and uncertainty as a result of this chapter.
- **Chapter 5 – Configuration Management**
  Discussed the necessity of managing change requests, maintaining product consistency, and using version control. Also discussed Git and other real-world CM tools as well.
- **Chapter 6 – Project Planning**
  Centered on project scheduling, baseline planning, work breakdown structures (WBS), and methods for estimating time and resources. Strengthened the notion that planning is a continuous process.
- **Chapter 7 – Monitoring and Control**
  Taught how to use milestone tracking, baselines and metrics, and corrective actions in iterative development environments to measure project progress.
- **Chapter 8 – Project Closure**
  Discussed how to formally end a project, document lessons learned, and archive deliverables and KPIs for later use.
- **Chapter 9 – Software Lifecycle Management (SLM)**
  Introduced software engineering metrics, quality control throughout lifecycle phases, and SDLC models such as Waterfall and Agile.
- **Chapter 10 – Requirement Management**
  Focused on collecting, analyzing, and managing requirements, including using configuration management to address changing requirements.
- **Chapters 11–12 – Design & Construction Management**
  Through architecture patterns, design prioritized quality, modularity, and abstraction; construction addressed programming standards and build techniques.

- **Chapter 13 – Testing and Quality Assurance**
  Discussed test planning, types of testing, verification and validation, and the function of QA in ensuring stakeholder satisfaction and software dependability.
- **Chapter 14 – Release and Maintenance**
  Talked about patching, post-release maintenance, deployment tactics, and how to guarantee support for the duration of the software's life.

2. Application in Real Projects

Using estimating models like COCOMO II and FPA in the early stages of budgeting has given me greater confidence. In a similar vein, risk management strategies such as earned value project tracking (Chapter 7) and probability-impact charts (Chapter 4) can be immediately applied in professional settings. The design, testing, and release chapters (11–14) also offered a useful guide for overseeing software development from conception to completion.

3. Peer Interactions

Particularly beneficial were group activities on stakeholder prioritizing and project charters. Classmates asked thoughtful questions about choosing a lifecycle model and handling unclear requirements (Chapter 10). My opinions were improved by peer discussions, particularly with regard to hybrid Agile-Waterfall approaches.

4. Challenges Faced

At first, I thought cost modeling and effort estimating models like COCOMO II were complicated. Additionally, risk assessment seemed abstract until we discussed actual cases in class. It also took careful consideration to determine when, in a hybrid team configuration, to use Agile vs Waterfall.

5. Personal Development Activities

I studied more publications on software quality assurance, worked on estimating on mock projects, and investigated tools like Git and Trello to mimic configuration and task management in order to move beyond the slides. To improve my conceptual foundation, I also read publications on risk frameworks and lifecycle models.

6. Goals for the Future

My goals include:

- Creating a comprehensive software project plan that incorporates risk assessment, scheduling, and estimation.
- Using JIRA or Azure DevOps to practice requirement traceability and prioritization.
- Using a combination of Agile and conventional project management techniques for group capstone projects.
- Using industry-aligned standards, I am improving my practical QA and testing abilities.