**Student Name:** Andres Ardila – 40274138

**Journal URL:** https://github.com/AndresArdila544/SOEN-6841-LJ

**Dates Range of activities:** 9-16 September 2024

**Date of the journal: 17 September 2024**

**Key Concepts Learned**

A **project** is a temporary endeavor with specific goals and scope. A **software project** is the development and delivery of software. A **project manager** integrates people, processes, tools, and technology to drive successful project completion. **Processes** in software projects include *planning*, *execution*, and *monitoring*. **Project initiation** involves tasks (broken down into smaller tasks) for setting the foundation through the project charter and objectives. Objectives should be SMART, specific, measurable, achievable, relevant and time constrained.

**Effort, cost, schedule, and resource estimates** are made using various techniques like:

**Experience-based techniques**: **Estimation by Analogy**, results from similar previous projects, project size comparison, to build a correct size and effort estimate for the new project; and **Expert judgments**.

**Function point analysis** provide a standardized method for measuring the various functions of a software application, measuring functionality from the user’s point of view, based on what the user requests and receives in return. Establishing a *boundary for the project*, is determining the *unadjusted function point* with Data and Transaction functions: *Internal Logical File, External Interface File, External Input, External Output and External Inquiry*, to determine the weight of each function, with a final assessment of the environment and processing complexity of the project.

**Algorithmic cost modeling**: mathematical function of product, project and process attributes. Complex and difficult to use. **COCOMO cost modeling**: based on project experience. Estimate number of lines of code in a system. COCOMO 2 incorporates a range of sub-models: Application composition, Early design (person-months), Reuse, Post-architecture. It considers various scale factors like Development flexibility, Architecture/risk resolution, Team cohesion, Process maturity and Precedentedness.

We also need to use expert judgments and project and product characteristics. The size of a software system can only be known accurately when it is finished. Several factors influence the final size. Different approach to effort estimation for each type of software development lifecycle model projects.

**Application in Real Projects:**

In real-world software development, a well-defined project charter and setting SMART objectives are critical. These steps lay the foundation by clearly outlining goals, deliverables, stakeholders, and roles, ensuring that all team members are aligned from the beginning. Breaking down tasks into smaller, manageable components during project initiation facilitates better tracking and monitoring, which is particularly useful in Agile methodologies, where work is organized into sprints.

Experience-based techniques like Estimation by Analogy are useful for projects with available historical data. Function Point Analysis is effective for larger software systems, especially where measuring delivered functionality is essential. For complex projects, Algorithmic cost modeling, particularly COCOMO 2, is valuable in estimating effort and cost. This model offers scalability, allowing for continuous adjustments as the project evolves, thereby supporting strategic decision-making. However, the accuracy of such models depends on having sufficient historical data to feed into them.

While these techniques offer precision, they can also be complex and time-consuming to implement. For smaller projects or fast-paced environments, they may seem unnecessary or overly detailed. Nevertheless, their application can lead to more accurate effort and cost estimates, reducing risks such as budget overruns and missed deadlines. However, expert judgment, while useful, can be subjective and prone to bias.

One of the major challenges in software development is that the size of a system and its required effort often cannot be fully known until the project is completed. This uncertainty is further compounded when \*\*requirements change frequently\*\*, which can disrupt timelines, increase costs, or necessitate additional resources, even with meticulous planning and estimation.

**Peer Interactions**

We did an exercise in class where we would create an example of a project charter identifying its scope and objectives. I discussed with a class mate the correct way of achieving a successful project charter for the exercise and discussed why its important to clearly state the scope and out of scope for the project, and the difference between scope and objectives.

**Challenges Faced:**

The topics presented were greatly understood, however the effort and cost estimates require more practical work to understand them completely. The examples provided were clear but until we get our hands on applying them on a real project, they remain quite abstract.

**Personal development activities:**

These two weeks I actively took notes during the lectures as well as review the slides on my own time in preparation for next weeks quiz. I read the chapters as required highlighting key concepts to add to my notes and submitted two exercises that helped me get an understanding of IT projects nowadays.

**Goals for the Next Week:**

Understand the topics presented in chapter 4 and 5 through the upcoming lectures. Measure my understanding of the topics in this journal in next weeks quiz. Implement the key concepts learned in the project initiation phase with our group.