Student Name: Nisarg Shah (40264902)

Course: SOEN 6841 Software Project Management

Journal URL: https://github.com/Nisarg-18/SOEN-6841-Software-Project-Management

Week 1: January 18 - January 24

Date: 23 January 2024

Key Concepts Learned:

1. What is project management?
    a. It is defined as an activity started to reach a specific goal in a certain time using the available resources.
    b. A project consumes: Resources, Budget and Time

2. What is software project management?
    a. It can be defined as applying project management and software engineering methods to develop or maintain a software product to reach its end goal in minimum time and budget and with all the available resources.
    b. The process involves Project Initiation, Project Planning, Project Monitoring and Project closure.

3. Importance and problems in project management
    a. Software projects are growing rapidly and are contributing to the economy and helping people improve their lives.
    b. There are still many challenges faced when building software, for example, lack of proper skills, immature tools, etc.
    c. Despite many problems, it is being done and can be done by using proper software project management techniques.

4. Processes in software projects
    a. Processes in software are defined as the way of doing things.

5. Factors influencing a software project
    a. Customer Management, Team Management, Software development model, Supplier Management, Technology Management

6. Requirements to be a Successful Software Project Manager
    a. Understands the project in and out
    b. Understands team management
    c. Understands tools and technology

7. Software project initiation
    a. Software project initiation involves project charter, project scope, project objectives, initial time and effort estimation

8. Project Charter, Project Scope and Objective
    a. It is a crucial document that defines the project's purpose, scope, objectives, stakeholders, and overall goals.

b. Project scope refers to the detailed definition of the deliverables, features, functions, and characteristics of a project.
c. Project objectives are the set of goals to be met when the project is completed. If they are not met, the project will be considered a failure.

Application in Real Projects:

1. This week mainly consisted of the basics of project management and how it is crucial to use project management, It sets the foundation for effective planning and execution.

2. I will try and implement these project management practices in the next software side project I start.

Peer Interactions:

1. Discussed the case study from chapter one in the class.

2. Had a healthy discussion on why project management is required in the field of software.

Challenges Faced:

1. Being a software developer, learning these concepts is a bit challenging because it involves a whole lot of things to take care of while starting a project.

Personal development activities:

1. Started exploring more about software project management and how it makes the whole development journey easy for both developers and customers.

Goals for the Next Week:

1. Start with the project initiation and market analysis of the course project.

2. Go through chapters 3 and 4.

Week 2: January 28 - February 3

Date: 3rd February 2024

Key Concepts Learned:

1. Effort Estimation for a Project:
    a. It is the process of predicting the amount of human effort required to complete a project.
    b. Estimation techniques: Experience-based and algorithmic cost modelling.

2. Experienced-based techniques:
    a. Estimation by analogy: Estimate new projects by comparing them to similar past projects
    b. Estimation by expert judgment: Experienced individuals or teams provide their opinions based on their knowledge of similar projects.

3. Algorithmic cost modelling:
    a. Cost is estimated as a mathematical function of product, project and process attributes whose values are estimated by project managers

4. COCOMO Model:
    a. Stands for Constructive Cost Model.
    b. It incorporates a range of sub-models that produce increasingly detailed software estimates.

5. Risks management:
    a. It is an uncertain event or condition that, if it occurs, could have a large impact on one or more project objectives.
    b. Major types of risks: Technology risks, Budget risks, Quality, risks, Time risks and Resource risks.

6. Steps involved in risk assessment:
    a. Risk Identification: Collecting all risk items.
    b. Risk Analysis: Analysing all the collected risks based on likelihood of occurrence, impact on the cost and efforts, etc.
    c. Risk Prioritization

7. Strategies for risk control:
    a. Acceptance: It involves acknowledging the existence of a risk without actively attempting to change or mitigate it.
    b. Avoidance: It aims to eliminate the risk by changing project plans or avoiding certain activities that pose the risk.
    c. Risk transfer: It involves shifting the responsibility for the risk and its potential impact to another party.
    d. Mitigation: It involves taking proactive measures to reduce the probability or impact of a risk.

Reflections on Case Study/course work:

1. This week mainly consisted of learning about estimating efforts and costs. Also, I learned about risk assessment and what are the different strategies available to deal with it.

2. Assessing the efforts and risks for the course project.

Collaborative Learning:

1. Discussed project objectives with the team members.

2. Discussed different features to add to our project's product.

Challenges Faced:

1. Finding the unique selling points for our intelligent tutoring system.

Further Research/Readings:

1. I read about various existing intelligent tutoring systems and understood their unique selling points.

2. Go through Chapter 5.

Adjustments to Goals:

1. Start with the market analysis of the course project.

Week 3: February 4 - February 10

Date: 10th February 2024

Key Concepts Learned:

1. Configuration Management: A configuration management system (CMS) is a set of tools, processes, and policies used to manage and control changes to software, hardware, documentation, and other configuration items throughout their lifecycle. The primary goal of a CMS is to ensure consistency, reliability, and traceability of configurations across different environments and versions.

2. Benefits of CMS:
    a. Reduces confusion and establishes order.
    b. Organizes the activities necessary to maintain product integrity.
    c. Ensures correct product configurations.
    d. Limits legal liability by providing a record of actions.
    e. Reduces life-cycle costs.
    f. Enables consistent conformance with requirements.
    g. Provides a stable working environment.
    h. Enhances compliance with standards.
    i. Enhances status accounting.

3. The parts of a configuration management system typically include:
    a. Version Control System (VCS): This is the core component that manages changes to source code, documents, and other files. It allows multiple developers to collaborate on the same project while keeping track of changes and versions.

    b. Build Management: This involves automating the process of compiling source code, running tests, and packaging the software into deployable artifacts. Build management ensures that software can be consistently built and deployed across different environments.

    c. Release Management: This part of the CMS is responsible for planning, scheduling, and coordinating the release of software to different environments, such as development, testing, staging, and production.

    d. Configuration Item Identification: This involves identifying and labeling all configuration items within the system, including software components, hardware devices, and documentation.

    e. Change Control: This process manages requests for changes to configuration items, evaluates their impact, and ensures that changes are implemented in a controlled manner.

4. Four Key Functions of CM:

    a. Version Control: One of the primary functions of a CMS is to manage version control. This involves tracking changes made to source code, documents, and other artifacts over time. Version control allows developers to keep track of different versions of files, revert to previous versions if necessary, and collaborate

effectively on shared projects without the risk of overwriting each other's work. By maintaining a history of changes, version control also enables traceability, which is crucial for understanding the evolution of the codebase and identifying the source of bugs or issues.

b. Configuration Identification: Another important function of a CMS is configuration identification. This involves identifying and labeling configuration items within the system, including software components, hardware devices, documentation, and other related assets. By establishing a clear and consistent naming convention for configuration items, a CMS helps ensure that all stakeholders have a common understanding of the components and their relationships within the system. Configuration identification is essential for managing dependencies, tracking changes, and maintaining consistency across different environments.

c. Change Management: CMS facilitates change management by providing mechanisms for managing requests for changes to configuration items. This includes documenting change requests, evaluating their impact, obtaining approvals from relevant stakeholders, and implementing changes in a controlled manner. By enforcing change control procedures, a CMS helps prevent unauthorized or unplanned changes, reduces the risk of introducing errors or inconsistencies, and ensures that changes are implemented in accordance with established policies and procedures.

d. Configuration Status Accounting: The fourth key function of a CMS is configuration status accounting. This involves maintaining accurate and up-to-date records of the status and configuration of all items within the system. Configuration status accounting tracks the current version, location, and status of each configuration item, as well as any related documentation, changes, or approvals. By providing visibility into the status of configuration items, a CMS enables stakeholders to assess the impact of changes, track progress against project milestones, and make informed decisions about the release and deployment of software products.

Reflections on Case Study/course work:

1. Learned about configuration management systems and how they play an important role in project management.

Collaborative Learning:

1. Discussed the findings for the market analysis with the project team.

2. Listed out the existing intelligent tutoring systems and unique selling points for our product.

Challenges Faced:

1. Finding out existing intelligent tutoring systems.

Further Research/Readings:

1. Reading chapter 6.

Adjustments to Goals:

1. Completing the problem identification and market analysis documents.

Week 4: February 11 - February 17

Date: 17th February 2024

Key Concepts Learned:

1. Project Planning: A software project plan is a comprehensive document outlining various aspects of a software development project. It serves as a roadmap for the entire project team, detailing tasks, schedules, resources, milestones, and deliverables to ensure the successful completion of the project within the defined constraints of time, budget, and scope.

2. Parts of a Software Project Plan:
    a. Introduction: Overview of the project, objectives, and scope.
    b. Project Organization: Roles and responsibilities of team members, organizational structure.
    c. Project Schedule: Timeline with milestones, task dependencies, and critical path.
    d. Resource Management: Allocation of human, financial, and technical resources.
    e. Risk Management: Identification, assessment, and mitigation strategies for project risks.
    f. Quality Management: Processes and metrics for ensuring software quality.
    g. Communication Plan: Strategies for internal and external communication.
    h. Change Management: Procedures for handling changes to project scope, schedule, or resources.
    i. Monitoring and Control: Methods for tracking progress and managing deviations from the plan.
    j. Closure: Procedures for project handover, documentation, and post-implementation review.

3. Types of Software Project Plans:
    a. Strategic Project Plan: Aligns with organizational objectives and long-term goals.
    b. Tactical Project Plan: Focuses on day-to-day tasks and short-term objectives.
    c. Detailed Project Plan: Provides granular details on specific tasks, resources, and timelines.
    d. High-Level Project Plan: Offers a broad overview of project objectives and major milestones.

4. Inputs for Making a Software Project Plan:
    a. Project Requirements: Detailed description of the features, functionalities, and constraints of the software.
    b. Stakeholder Inputs: Feedback and expectations from key stakeholders, such as clients, users, and project sponsors.
    c. Resource Availability: Information about the human, financial, and technological resources available for the project.
    d. Organizational Policies and Standards: Guidelines and procedures set by the organization governing project management, development, and quality assurance.
    e. Industry Best Practices: Knowledge and insights from industry standards and best practices in software development and project management.

5. Project Pitch: In this pitch, we address the pressing need for personalized educational support with an Intelligent Tutoring System (ITS). Highlighting the challenges faced by learners and educators, we emphasize the urgency of our solution. Our ITS offers a transformative approach, providing personalized learning experiences, adaptive feedback mechanisms, and data-driven insights. Market analysis demonstrates a growing demand for educational technology solutions, and we stand out with our unique features. Our value proposition lies in improved learning outcomes, increased efficiency, and cost savings for stakeholders. With a clear business model focusing on revenue streams and scalability, we ensure long-term viability.

Reflections on Case Study/course work:

1. The project proposal exercise provided a comprehensive understanding of designing an ITS to address educational challenges. It highlighted the importance of clearly defining objectives, scope, methodology, expected outcomes, and market considerations. Crafting a compelling pitch involved articulating the problem, proposing a solution, and outlining the value proposition and business model.

Collaborative Learning:

1. Preparing project proposal pitch with the team members.

2. Each member contributed unique insights, which enriched the proposal and improved its quality.

Challenges Faced:

1. Bringing together different viewpoints into a clear proposal was tough. It required us to communicate well and find compromises.

Further Research/Readings:

1. Go through chapters 1 to 6 again to prepare for the mid-term exam.

2. Additional research on emerging trends in educational technology could provide valuable insights for future projects.

Adjustments to Goals:

1. Prioritizing time for exam preparation while maintaining progress on the project.