Unit-8 Project Management and Planning

Introduction to Project Management

Software project management is an art and discipline of planning and supervising software projects. It is a sub discipline of software project management in which software projects planned, implemented, monitored and controlled. It is a procedure of managing, allocating and timing resources to develop computer software that fulfills requirements.

Software project management is an essential part of software engineering. Good management cannot guarantee project success however bad management usually results in project failure i.e. software delivered late, cost more than originally estimated and fails to meet its requirement. Software Project Management (SPM) is a proper way of planning and leading software projects. It is a part of project management in which software projects are planned, implemented, monitored and controlled.

Software project management and developed populated by people of any organization that can categories into various parts:

1) Senior manager:

Who define the business issue that often has significance influence on the project.

2) Project manager:

Who must plans, motive, organize and control the practitioners who do software works.

3) Practitioners:

Who specify the required and other for the software to be developed and other stock holders who have a peripheral interest in the outcome

4) Customers:

Who interact with the software once, it is realized for the production use.

Activities in project management

Software Project Management consists of many activities that include planning of the project deciding the scope of product, estimation of cost in different terms, scheduling of tasks, etc.

The list of activities is as follows:

- Project planning and Tracking
- Project Resource Management
- Scope Management
- Estimation Management
- Project Risk Management
- Scheduling Management
- Project Communication Management
- Configuration Management

Project Planning: It is a set of multiple processes, or we can say that it a task that performed before the construction of the product starts.

Scope Management: It describes the scope of the project. Scope management is important because it clearly defines what would do and what would not. Scope Management create the project to contain restricted and quantitative tasks, which may merely be documented and successively avoids price and time overrun.

Estimation management: This is not only about cost estimation because whenever we start to develop software, but we also figure out their size(line of code), efforts, time as well as cost.

Scheduling Management: Scheduling Management in software refers to all the activities to complete in the specified order and within time slotted to each activity. Project managers define multiple tasks and arrange them keeping various factors in mind.

Project Resource Management: In software Development, all the elements are referred to as

Resources for the project. It can be a human resource, productive tools, and libraries.

Project Risk Management: Risk management consists of all the activities like identification, analyzing and preparing the plan for predictable and unpredictable risk in the project.

Project Communication Management: Communication is an essential factor in the success of the project. It is a bridge between client, organization, team members and as well as other stakeholders of the project such as hardware suppliers.

Project Configuration Management: Configuration management is about to control the changes in software like requirements, design, and development of the product.

Project Planning

Software project planning is task, which is performed before the production of software actually starts. It is there for the software production but involves no concrete activity that has any direction connection with software production; rather it is a set of multiple processes, which facilitates software production. The project plan sets out the resources available to the project the work breakdown and a schedule for carrying out the work. So, effective management of software project depends on the thoroughly planning the progress of project. A plan drawn up at the starting of a project should be used as driver for the project. The planning is an iterative. Process, which is only complete when the project itself is complete. As the project information becomes available during the project. The plan should be regularly revised. The details of the project plan vary depending on the type of project organization. However, most plans should include the following sections:

Introduction: It describes the objectives of the projects and sets out the constraints like budget, time etc that affect the project management.

Project organization: It describes the way in which development team is organized, the people involved and their role in the team.

Risk analysis: This describes the possible project risk, the likelihood of these risks arising and the risk reduction strategies that are proposed

Hardware and software resource requirement: This specifies the hardware and software required to carry out the development. If the hardware has to be bought estimates of prices and delivery schedule may be included.

Work breakdown: In this phase the project is breakdown into activities and identifies the milestones and deliverables associated with each activity.

Project schedule: This phase shows the dependencies between activities the estimated time required to reach each milestone and allocation of people to each activity.

Monitoring and reporting mechanism: This defines the management report that should be produced.

Plan-Driven Development (PDD)

Plan-driven development is a software development method which attempts to plan and develop all of the features a user might want in the final product and determines how all those features are to be developed. Actually this action plan is based on the execution of an ordered set of task-specific levels.

We can say it is an attitude to software engineering wherever it is planned in feature how the development procedure will take place. It is based on the administration methods of project engineering & an old-style method of handling big software development ventures. It makes the use of management planning unfolds in measuring progress and making project decisions.

A plan-driven development project sets out a program for the project, including the available resources, work breakdowns, work completion schedules etc.

- Organizing the team and roles of different members.
- Risk Analysis and Probabilities, Prospects, Strategies to Arise.
- Hardware as well as software resource requirements.
- Deciding Activities, Deliveries, and Milestones.
- Allotment of people and allocation of their time.
- Monitoring and Reporting systems

Advantages of PDD:

- Require knowledgeable personnel at the beginning.
- Appropriate for big software development/extensive systems and sides.
- Grips hazardous systems efficiently.
- Suitable for a stable growth environment.
- Achievement attained through construction and directive.

Disadvantages of PDD:

- Cannot lodge changes any time.
- Longer distance in each repetition or increase.
- The shoulder that, future variations will not occur.
- Lack of user participation through the life cycle of the creation.
- Expensive for the dynamic expansion environment.

Project Scheduling

Project scheduling is concerned with the techniques that can be employed to manage the activities that need to be undertaken during the development of a project. Project scheduling is the process of spate the total work involve in project into separate activities and estimating the time and resources required to complete activities and organize them into a coherent sequence. It is one of the most difficult jobs for a project manager, manager estimate the time and resources required. Project scheduling means separating the total work involved in a project into separate activities and organize the work so that the work force is optimal. Following are the principle for scheduling.

- Compartmentalization: Define district task
- Interdependency: Parallel and sequential task
- Time allocation: Assigned, personals, resources, days, start time and end time
- Effort validation: Be sure Resources are available
- Defined responsibilities
- Defined outcome: Each task must have an outcome

Scheduling is carried out in advance of the project commencing and involves:

- identifying the tasks that need to be carried out;
- estimating how long they will take;
- allocating resources (mainly personnel);

Project scheduling means preparing various graphical representations and showing project activities, their duration and staffing. Minimize the task dependencies to avoid delays cause by one task waiting for another task to complete. Usually some of the activities are carried out in parallel so co-ordinate these parallel activities and organize the work so that the workforce is used optimally.

Estimation Techniques

Estimation techniques refer to the methods and approaches used to approximate the time, effort, resources, or cost required for a specific task, project, or activity. These techniques are essential in project management, software development, engineering, construction, and various other fields where planning and resource allocation are critical.

The primary goal of estimation techniques is to provide reasonably accurate predictions of the resources and time needed for a project to help with effective planning, budgeting, and decision-making. Different projects may require different estimation approaches based on factors such as project complexity, available information, and the level of detail required.

Here are some common types of estimation techniques:

Expert Judgment: Involves seeking input from individuals with expertise and experience in the relevant field to estimate project parameters.

Analogous Estimation (Top-down Estimation): Relies on historical data from similar projects to estimate the parameters of the current project.

Parametric Estimation: Uses statistical relationships between project parameters and historical data to calculate estimates.

Three-Point Estimation (PERT - Program Evaluation and Review Technique): Utilizes three estimates (optimistic, most likely, and pessimistic) to calculate the expected value and account for uncertainty.

Bottom-Up Estimation: Involves estimating the individual components or tasks of a project and aggregating them to derive the overall estimate.

Expert Delphi Technique: Gathers estimates from a group of experts through a series of iterations and feedback until a consensus is reached.

Reserve Analysis: Adds contingency reserves to the estimated values to account for uncertainties and risks.

Historical Data Analysis: Analyzes data from previous projects to identify patterns, trends, and historical performance metrics.

Monte Carlo Simulation: Runs multiple simulations based on probability distributions to generate a range of possible outcomes and assess project risks.

Vendor Bid Analysis: Analyzes bids and proposals from vendors to estimate the cost and effort involved in outsourcing certain aspects of the project.

Each technique has its strengths and weaknesses, and the choice of method often depends on the specific characteristics of the project, the availability of data, and the level of accuracy required. Effective estimation is crucial for successful project planning, resource allocation, and risk management.

COCOMO Model

The Constructive Cost Model (COCOMO) is a software development cost estimation model that was first introduced by Barry Boehm in 1981. The model is widely used in the software engineering industry as it provides an accurate estimate of the cost, effort, and duration required to complete a software project. COCOMO takes into account a variety of factors, such as project size, complexity, and development environment, to generate a detailed estimate of the resources required for a project. The model has evolved over time, with several versions being developed to address different types of software projects. Despite its limitations, COCOMO remains a

valuable tool for software developers and project managers looking to estimate the cost and effort involved in their projects.

Effort and Schedule are the two essential parameters that determine the quality of software products and are also the primary outcomes of COCOMO

- **Effort** refers to the amount of labor required to complete a software development task and is typically measured in person months.
- **Schedule** refers to the time required to complete a job, which is directly proportional to the effort invested. The schedule is measured in units of time, such as weeks and months, and is crucial for successful project management.

COCOMO has several models that predict cost estimates at different levels, depending on the required level of accuracy and correctness. These models are applicable to various software development projects, with the characteristics of each project determining the constant values used in subsequent calculations. The specific system type also influences these characteristics, with Boehm's classification of organic, semi-detached, and embedded systems being particularly relevant. These classifications provide guidelines for understanding the characteristics of a software system and enable software developers to choose the appropriate COCOMO model for accurate cost estimation.

COCOMO classifies projects into three distinct types:

- Organic
- Semi-detached
- Embedded Systems

Organic

Organic development projects fall into the category of well-understood application programs with reasonably small development teams comprised of experienced members with similar project experience. These types of projects are typically straightforward and include simple business systems, inventory management systems, and data processing systems. Because of their relative simplicity, these projects often require less effort and time than other project types and

can be completed more quickly. COCOMO classifies such projects as organic, given their relative simplicity and smaller size, making them easier to manage and develop efficiently.

Semi-detached

Semidetached development projects involve a mix of experienced and inexperienced team members who may not be familiar with all aspects of the project being developed. Examples of such projects include developing new operating systems, database management systems, and complex inventory management systems. Due to their moderate complexity and the need to manage both experienced and inexperienced team members, these projects often require more effort and time than simpler organic projects. COCOMO classifies such projects as semidetached to help developers estimate the effort and time required to complete them accurately.

Embedded Systems

Embedded development projects are characterized by strong coupling between software and complex hardware or the need to adhere to stringent operational regulations. Examples of such projects include software development for Automated Teller Machines (ATMs) and Air Traffic Control systems. Due to their high complexity and the need to interface with complex hardware, embedded projects often require significant effort and time to complete successfully. COCOMO classifies such projects as embedded to help developers accurately estimate the effort and time required for project completion.