

PROJECT PLANNING

- Begins with a set of activities that are collectively called **project planning**.
- **Estimating the work** to be done, **the resources** that will be required and **the time** that will elapse from start to finish.
- In short, project planning help us to manage time, cost, quality, change, risk and issues.
- Once these tasks are accomplished, the software team should develop a project schedule.

Project planning-objective

- To provide a framework that enables the manager to make reasonable estimates of resources, cost and schedule.
- It attempts to define best case and worst case scenario so that project outcomes can be bounded.
- Project plan must be adapted and updated as project proceeds because of uncertainty.

Tasks set of project planning

1. Establish project scope
2. Determine feasibility
3. Analyze risks
4. Define required resources
 - a. Determine required human resources
 - b. Define reusable software resources
 - c. Identify environmental resources

Tasks set of project planning

5. Estimate cost and effort
 - a. Decompose the problem
 - b. Develop two or more estimates using size, function, process tasks, or use cases
 - c. Reconcile the estimates
6. Develop a project schedule
 - a. Establish a meaningful task set
 - b. Define a task network
 - c. Use scheduling tools to develop a time-line chart
 - d. Define schedule tracking mechanism.

Software scope

- Describes the functions and features that are to be delivered to end users;
- The data that are input and output;
- The content that is presented to users as a consequence of using the software;
- And the performance, constraints, interfaces, and reliability that bound the system.
- In short, it involves determining and documenting a list of specific goals, deliverables, tasks, costs and deadlines.
- It also explains boundaries of the project, establishes responsibilities of each member and sets up procedures for how completed work will be verified and approved.

Software Scope (continued)

- After the scope has been identified, two questions are asked
 - Can we build software to meet this scope?
 - Is the project feasible?

Feasibility

- After the scope is resolved, feasibility is addressed
- Software feasibility has four dimensions
 - **Technology** – Is the project technically feasible?
 - **Finance** – Is it financially feasible? Can development be completed at a cost that the software organization, its client, or the market can afford?
 - **Time** – Will the project's time-to-market beat the competition?
 - **Resources** – Does the software organization have the resources needed to succeed in doing the project?

Project Resources

Resource Estimation

- Three major categories of software engineering resources
 - **People**
 - **Development environment**
 - **Reusable software components**
- Each resource is specified with
 - A description of the resource
 - A statement of availability
 - The time when the resource will be required
 - The duration of time that the resource will be applied

Categories of Resources

People

- Number required
- Skills required
- Geographical location

Development Environment

- Software tools
- Computer hardware
- Network resources

The
Project

```
graph TD; People[People] --> Project((The Project)); DevEnv[Development Environment] --> Project; Reusable[Reusable Software Components] --> Project;
```

Reusable Software Components

- Off-the-shelf components
- Full-experience components
- Partial-experience components
- New components

Human Resources

- Planners need to select the number and the kind of people skills needed to complete the project
- They need to specify the organizational position and job specialty for each person
- The number of people required can be determined only after an estimate of the development effort

Development Environment Resources

- A software engineering environment (SEE) incorporates hardware, software, and network resources that provide platforms and tools to develop and test software work products
- Planners must identify the time window required for hardware and software and verify that these resources will be available

Reusable Software Resources

- **Off-the-shelf components**

- Components are from a third party or were developed for a previous project
- Ready to use; fully validated and documented;

- **Full-experience components**

- Components are similar to the software that needs to be built
- Software team has full experience in the application area of these components
- Modification of components will incur relatively low risk

- **Partial-experience components**

- Components are related somehow to the software that needs to be built but will require substantial modification
- Software team has only limited experience in the application area of these components
- Modifications that are required have a fair degree of risk

- **New components**

- Components must be built from scratch by the software team specifically for the needs of the current project
- Software team has no practical experience in the application area
- Software development of components has a high degree of risk