## 3.1 Activities in project management

Software project management comprises of a number of activities, which contains planning of project, deciding scope of software product, estimation of cost in various terms, scheduling of tasks and events, and resource management. Project management activities may include:

- Project Planning
- Scope Management
- Project Estimation

### 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.

### Scope Management

It defines the scope of project; this includes all the activities, process need to be done in order to make a deliverable software product. Scope management is essential because it creates boundaries of the project by clearly defining what would be done in the project and what would not be done.

During Project Scope management, it is necessary to -

- Define the scope
- Decide its verification and control
- Divide the project into various smaller parts for ease of management.
- Verify the scope
- Control the scope by incorporating changes to the scope

## Project Estimation

For an effective management accurate estimation of various measures is a must. With correct estimation managers can manage and control the project more efficiently and effectively.

Project estimation may involve the following:

- Software size estimation

*Software size may be estimated either in terms of KLOC (Kilo Line of Code) or by calculating number of function points in the software. Lines of code depend upon coding practices and Function points vary according to the user or software requirement.*

- Effort estimation

*The managers estimate efforts in terms of personnel requirement and man-hour required to produce the software. For effort estimation software size should be known. This can either be derived by managers' experience, organization's historical data* *or software size can be converted into efforts by using some standard formulae.*

- Time estimation

*Once size and efforts are estimated, the time required to produce the software can be estimated. Efforts required is segregated into sub categories as per the requirement specifications and interdependency of various components of software.*

- Cost estimation

*This might be considered as the most difficult of all because it depends on more elements than any of the previous ones. For estimating project cost, it is required to consider -*

- Size of software
- Software quality
- Hardware
- Additional software or tools, licenses etc.
- Skilled personnel with task-specific skills
- Travel involved
- Communication
- Training and support

## 3.2 Software Project Planning

Project planning helps in better utilization of resources and optimal usage of the allotted time for a project. Project planning should be effective so that the project begins with well-defined tasks. Effective project planning helps to minimize the additional costs incurred on the project while it is in progress.

There are various activities involved in *project planning* as listed below:

1. **Project estimation** → When a project is finalized then several estimations need to be made. A proper ***cost estimation*** is required that would be required for project. ***Time estimation*** is needed that would propose the time in which the project would be in deliverable position. ***Effort estimation*** is needed to finalize the resource available and required for the project.
2. **Project Schedules** → Schedules for manpower and several other resources has to be prepared that would be needed as the project progress.
3. **Project staffing** → Staff needs to be organized and planned. Analysis has to be done to find out if the existing staff is sufficient for the project or new recruitment is required.
4. **Project risks and management** → Possible risks to the project needs to identified and its solution needs to found to let project progress smoothly.
5. **Project miscellenous management** → Quality of the product, configuration management plan, etc are some miscellenous task that needs to be planned accordingly.

## 3.3 Software Project Management Plan

Software project manager prepare a document on the basis of decision finalized during the project planning. This document is known as Software Project Management Plan Document or SPMP document.

SPMP document is a well organized document that contains the project planning in detail.

A SPMP document is prepared and organized in structure as shown below:

**Introduction**

- Objectives
- Functions
- Performance issues
- Constraints

**Project estimates**

- Historical data used
- Estimation techniques details
- Cost, duration, effort estimates

**Project Schedule**

- Work breakdown
- Gantt and PERT chart

**Project resource**

- Manpower
- Hardware and Software
- Highly skilled professionals

**Staff organization**

- Team formation and structure
- Management reporting

**Risk Management**

- Risk analysis
- Risk identification
- Risk abatement methods

**Project tracking**

- Project Control
- Miscellenous activities

All the above activities are documented in SPMP document by project manager.

## 3.4 Software Project Scheduling and Time Line Charts

Project Scheduling is a significant project planning activity. It comprises deciding which functions would be taken up when. To schedule the project plan, a software project manager wants to do the following:

- Divide the large works into smaller activities or module.
- Find out the various tasks/activities and correlate them.
- Estimate time frame required for each task.
- Allocate the resources to each activity.
- Plan when to start and when to end each activity.
- Determine the critical path of the project.

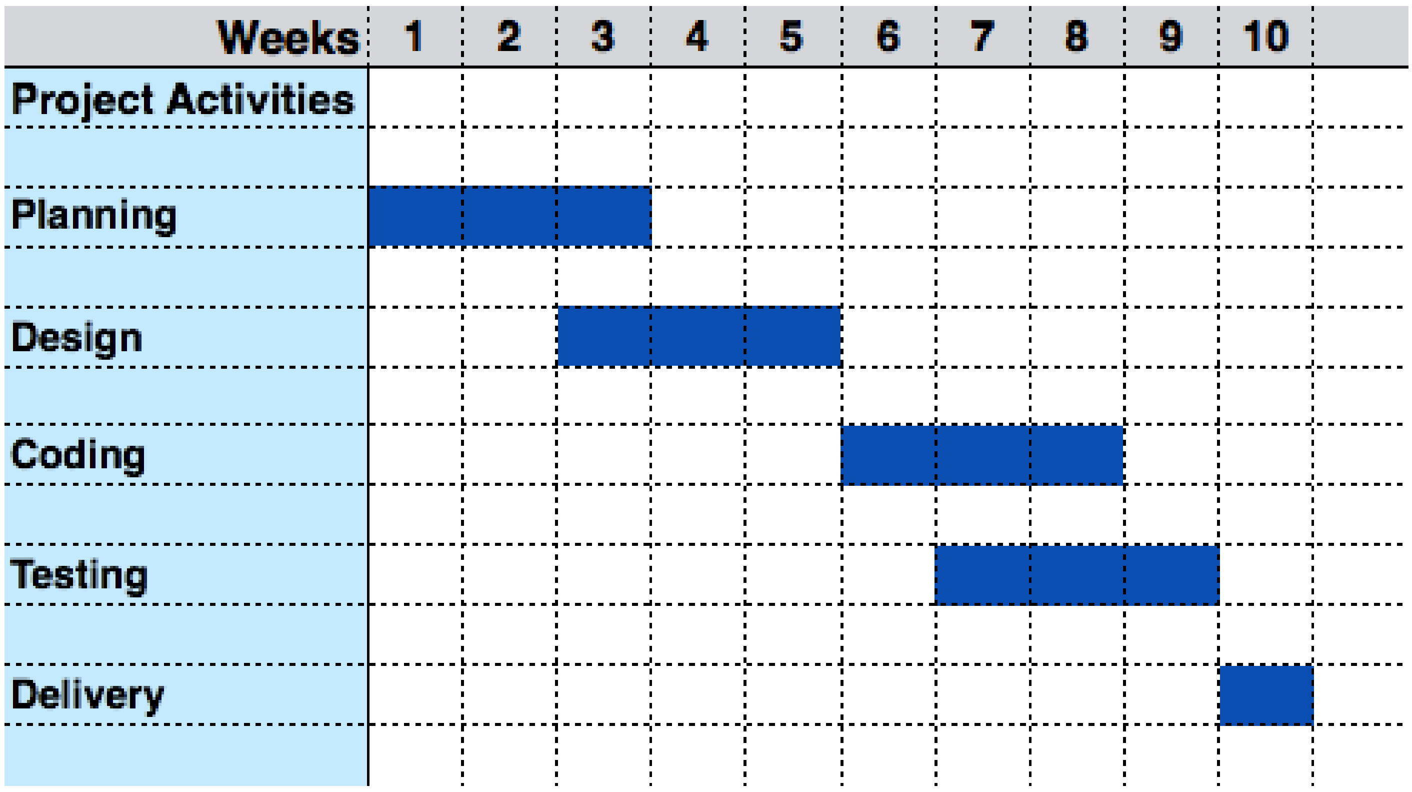
A critical path is the chain of important activities that would help in determining the duration of the project.

**<u>Techniques</u>**

**Gantt Charts**

- Gantt chart is a project management tool which enables you schedule your work over a period of time.
- Gantt Chart was devised by Henry Gantt (1917).
- It represents project schedule with respect to time periods.
- It is a horizontal bar chart with bars representing activities and time scheduled or the project activities.
- The Y-axis is made up of individual tasks and the X-axis represents time.
- The Gantt chart tool provides a visual timeline or the start and end of tasks, making it clear how task are

interrelated and perhaps rely on the completion of another before one can start.

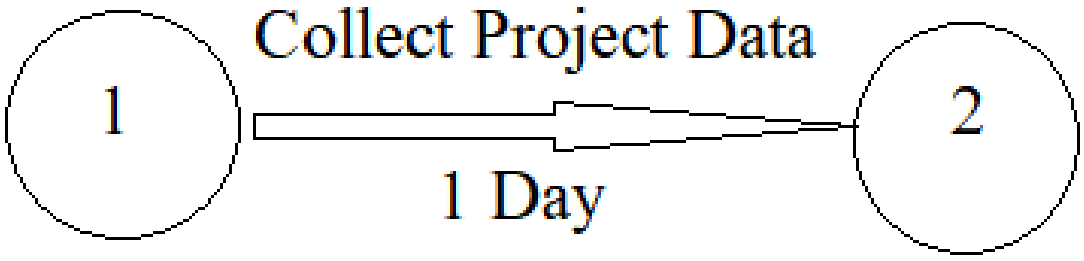


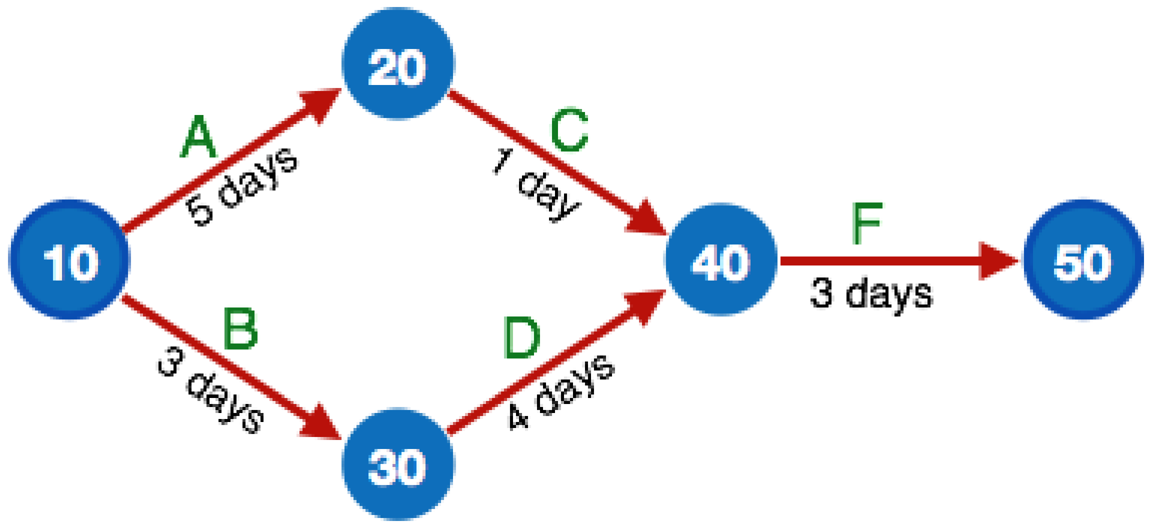
## Advantages and Disadvantages

- Gantt charts are usually the better option for monitoring projects.
- Gantt chats provide project transparency
- With a Gantt chart, PMs can quickly identity wherether a project is progressing as scheduled
- The disadvantages of Gantt chart is the time it can take to create and update.

## PERT Chart

- PERT (Program Evaluation & Review Technique) chart is a tool that depicts project as network diagram.
- It is capable of graphically representing main events of project in both parallel and consecutive way.
- The PERT methodology was developed in the 1950s by the U.S Navy to help manage large, complex projects. With the PERT chart, PMs can define all task and activities required for the project, and then develop a realistic timeframe for project completion.
- Events, which occur one after another, show dependency of the later event over the previous one.
- Events(Milestone dates) are shown as numbered nodes. They are connected by labelled arrows depicting sequence of tasks in the project.
- Examples :





## Advantages and Disadvantages

- A PERT chart is advantageous at the outset of a project's planning phase to get a high-level overview of all the task involved, their dependencies, and their estimated completion dates.
- A PERT chart provides an excellent visual representation of the relationship between each project activity.
- On the downside, PERT charts can quickly become complicated and even confusion when used with, complex projects with hundreds of individual tasks.

## 3.5 Software Project Team Management and Organization

### How to manage a software development team?

- Clearly define and map out expectations(requirements)
- Allocate developers to tasks accordingly
- Stay on top of deadlines
- Distribute and share files in one place

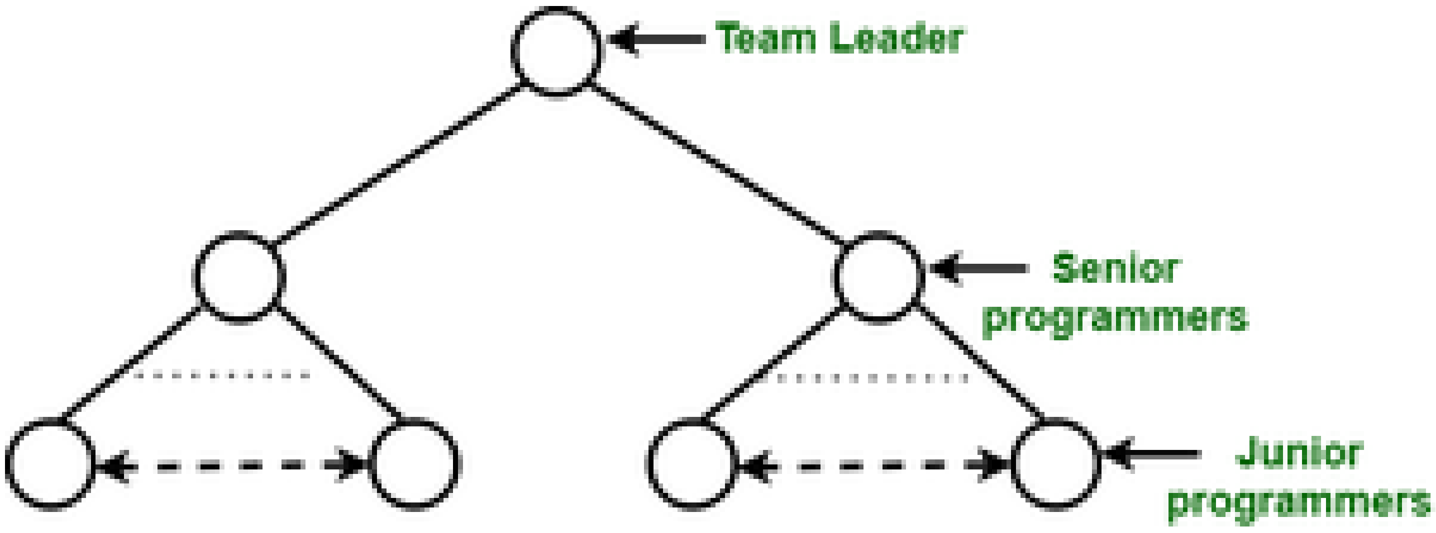
- Monitor real-time updates

There are many ways to organize the project team. Some important ways are as follows :

- Hierarchical team organization
- Chief-programmer team organization
- Matrix team, organization
- Egoless team organization
- Democratic team organization

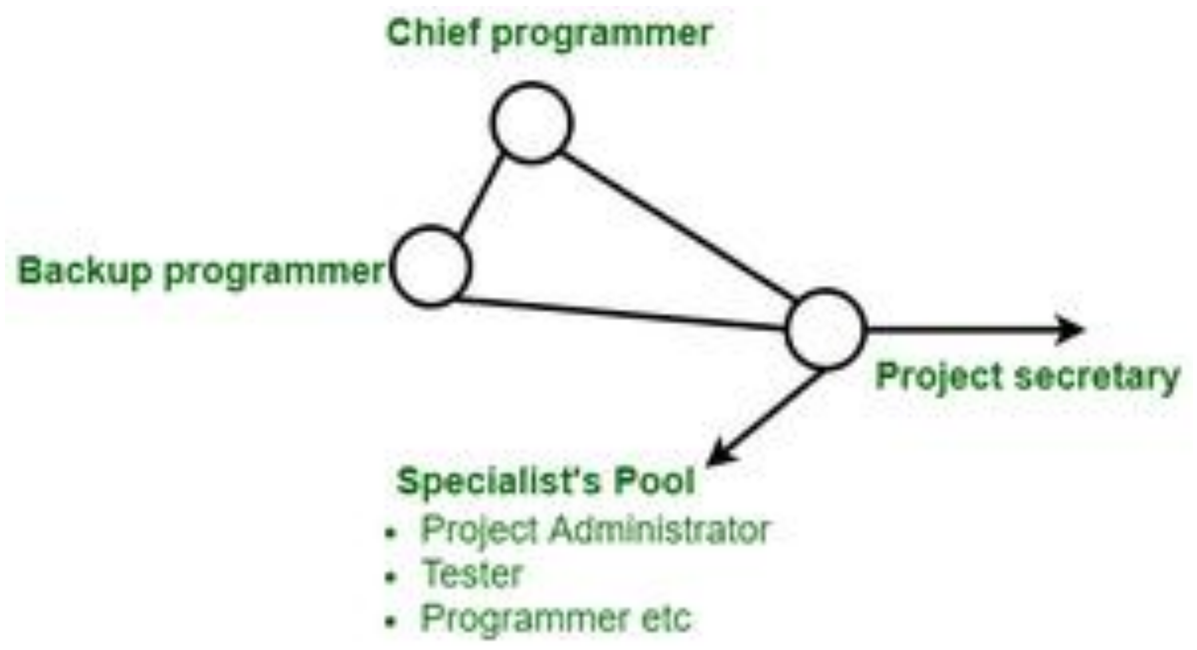
**Hierarchical team organization:**

In this, the people of organization at different levels following a tree structure. People at bottom level generally possess most detailed knowledge about the system. People at higher levels have broader appreciation of the whole project.



**Chief-programmer team organization :**

This team organization is composed of a small team consisting the following team members :



The Chief programmer : It is the person who is actively involved in the planning, specification and design process and ideally in the implementation process as well.

The project assistant : It is the closest technical co-worker of the chief programmer.

The project secretary : It relieves the chief programmer and all other programmers of administration tools.

Specialists : These people select the implementation language, implement individual system components and employ software tools and carry out tasks.

**Matrix Team Organization :**

The matrix organizational structure is a combination of two or more types of organizational structures. The matrix organization is the structure uniting these other organizational structures to give them balance. Usually, there are two chains of command, where project team members have two bosses or managers.

**Egoless Team Organization :**

Egoless programming is a state of mind in which programmer are supposed to separate themselves from their product. In this organization work products are discussed openly and all freely examined by all team members.

**Democratic Team Organization :**

It is quite similar to the egoless team organization, but one member is the team leader with some responsibilities :

- Coordination
- Final decisions, when consensus(general agreement) cannot be reached.

## 3.6 Software Project Estimation

**Estimation** is the process of finding an estimate, or approximation, which is a value that can be used for some purpose even if input data may be incomplete, uncertain, or unstable.

**Estimation** determines how much money, effort, resources, and time it will take to build a specific system or product.

**Software project estimation** is the process of estimating various resources (staff, cost, hardware) required for the completion of a project.

Effective software project estimation is an important activity in any software development project.

The four basic steps in Software Project Estimation are −

- Estimate the size of the development product.
- Estimate the effort in person-months or person-hours.
- Estimate the schedule in calendar months.
- Estimate the project cost in agreed currency.

Following Points Depicts Software Project Estimation:-

| Activity | Task Involved |
|---|---|
| Planning | Cost Estimation, planning for training of manpower, Project scheduling and budgeting the project. |
| Controlling | Size metrics and schedule metric help the manager to keep control of the project during execution |
| Monitoring/Improving | Metrics are used to monitor progress of the project and wherever possible sufficient resources are allocated to improve. |

## 3.6.1 LOC Based Estimation

A line of code (LOC) is any line of text in a code that is not a comment or blank line, and also header lines, in any case of the number of statements or fragments of statements on the line.

LOC clearly consists of all lines containing the declaration of any variable, and executable and non-executable statements. As Lines of Code (LOC) only counts the volume of code, you can only use it to compare or estimate projects that use the same language and are coded using the same coding standards.

Advantages :

- Most used metric in cost estimation.
- Its alternates have many problems as compared to this metric.
- It is very easy in estimating the efforts.

Disadvantages :

- Very difficult to estimate the LOC of the final program from the problem specification.
- It correlates poorly with quality and efficiency of code.
- It doesn’t consider complexity.

## 3.6.2 FP Based Estimation

Function Point Analysis was initially developed by Allan J. Albercht in 1979 at IBM and it has been further modified by the International Function Point Users Group (IFPUG).

Function Point Analysis (FPA) is a method or set of rules of Functional Size Measurement. It assesses the functionality delivered to its users, based on the user’s external view of the functional requirements. It measures the logical view of an application, not the physically implemented view or the internal technical view.

## Types of FPA:

### Transactional Functional Type –

**External Input (EI):** EI processes data or control information that comes from outside the application's boundary.

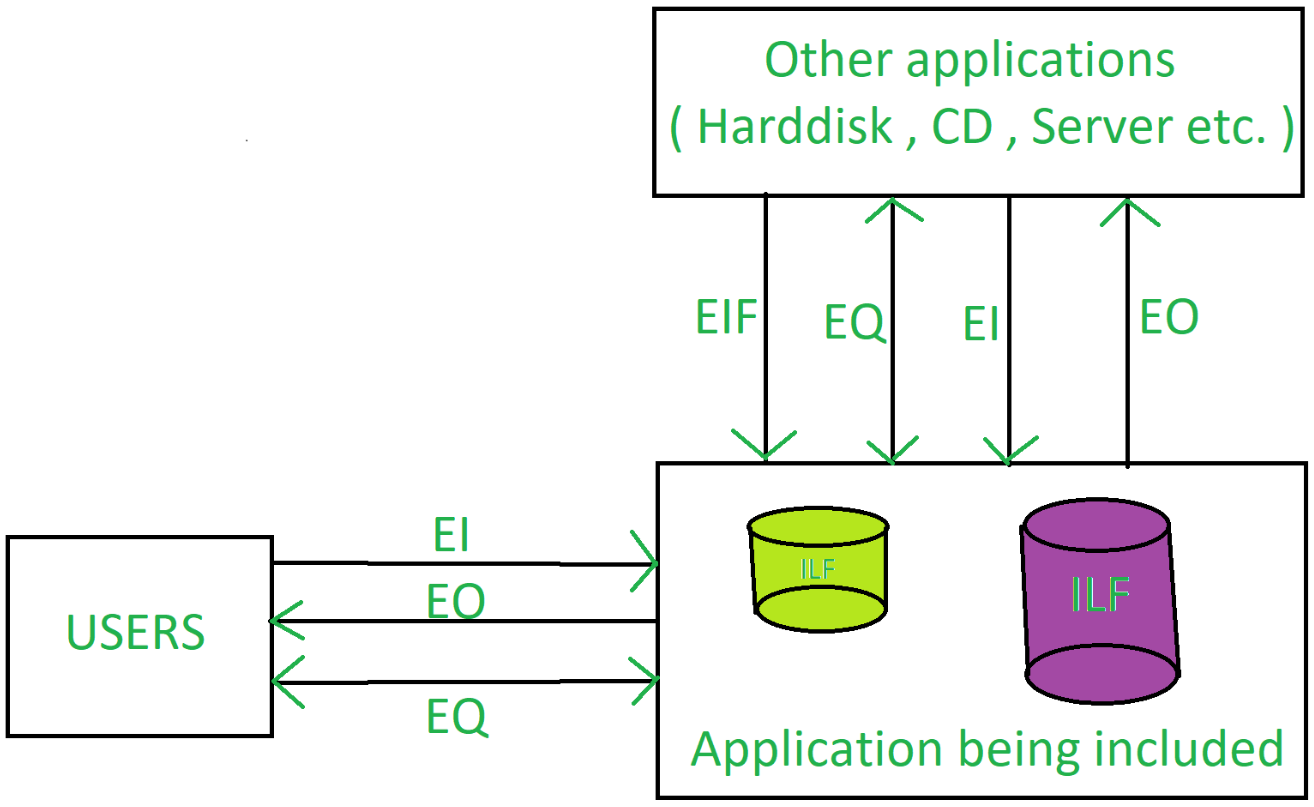
**External Output (EO):** EO is an elementary process that generates data or control information sent outside the application's boundary.

**External Inquiries (EQ):** EQ is an elementary process made up of an input-output combination that results in data retrieval.

### Data Functional Type –

**Internal Logical File (ILF):** A user identifiable group of logically related data or control information maintained within the boundary of the application.

**External Interface File (EIF):** A group of users recognizable logically related data allusion to the software but maintained within the boundary of another software.



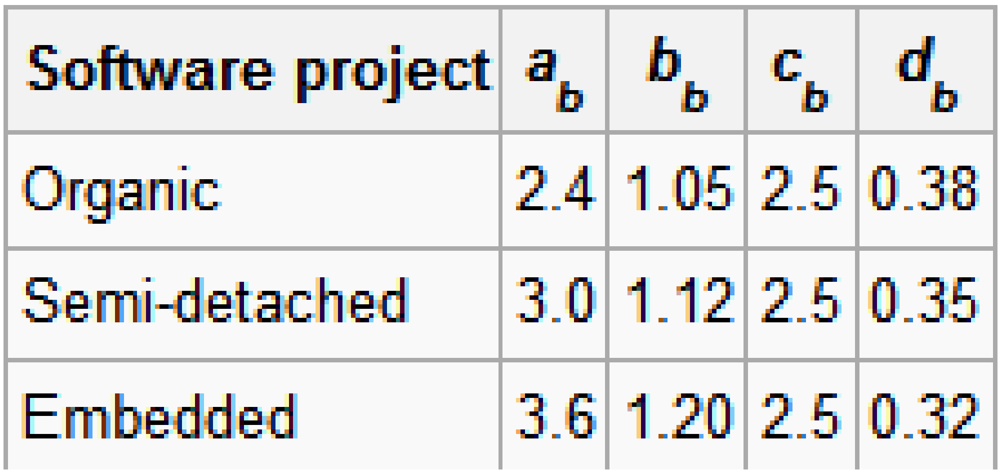
## 3.6.3 COCOMO Model

- ❖Boehm proposed COCOMO (Constructive Cost Estimation Model) in 1981.
- ❖COCOMO is one of the most generally used software estimation models in the world.
- ❖COCOMO predicts the efforts and schedule of a software product based on the size of the software.
- ❖Applies on 3 classes of software project.
  - ➢Organic Project

  - 2-50 KLOC | Small Team | Good Experience | not tight Deadline
- Semi Detach Project
  - 50-300 KLOC | Medium Team | Average Experience | Medium Deadline
- Embedded Project
  - 300 & above KLOC | Large Team | Very little previous Experience | Tight Deadline

❖Stages of COCOMO

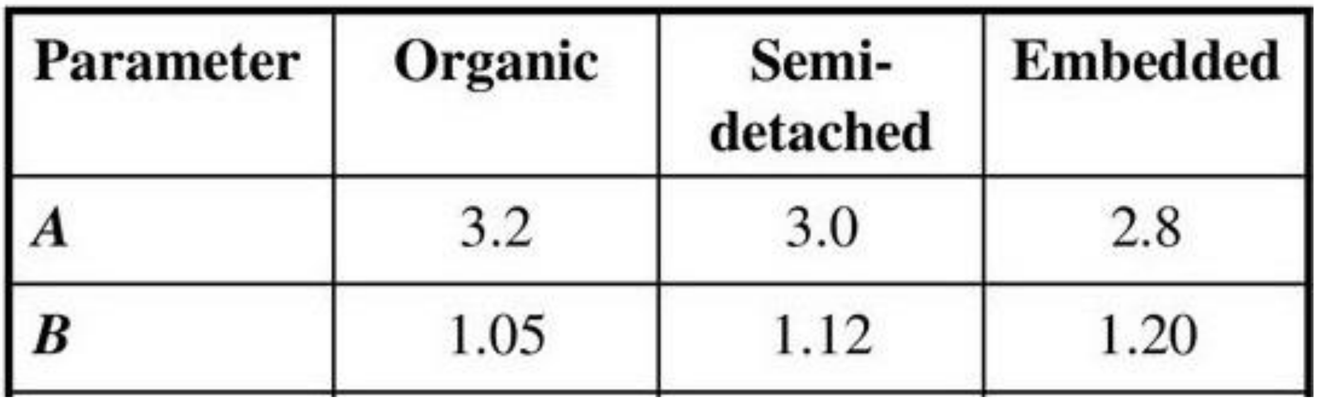
- **Basic COCOMO Model**
  - Approximate estimate of project parameters.
  - Software development effort is estimated using LOC (lines of code)
    - Effort (E) = a $(\text{KLOC})^b$
    - Time(T) = c$(\text{Effort})^d$
    - Person required (P) = E/D
  - The above formula is used for the cost estimation of for the basic COCOMO model, and also is used in the subsequent models. The constant values a,b,c, and d for the Basic Model for the different categories of the system:



| Software project | $a_b$ | $b_b$ | $c_b$ | $d_b$ |
|---|---|---|---|---|
| Organic | 2.4 | 1.05 | 2.5 | 0.38 |
| Semi-detached | 3.0 | 1.12 | 2.5 | 0.35 |
| Embedded | 3.6 | 1.20 | 2.5 | 0.32 |

## Intermediate COCOMO Model:

- It is extension of Basic COCOMO Model
- It refines estimate obtained by Basic COCOMO
- Effort Applied (E) = $a_i$ $(KLOC)^{b_i}$



| Parameter | Organic | Semi-detached | Embedded |
|---|---|---|---|
| *A* | 3.2 | 3.0 | 2.8 |
| *B* | 1.05 | 1.12 | 1.20 |

## Detail COCOMO Model:

In detailed cocomo, the whole software is divided into different modules and then we apply COCOMO in different modules to estimate effort and then sum the effort.It has 6 phases -

- Plan & requirement

- System Design
- Detail Design
- Module Code and Test
- Integration and test
- Cost Constructive

The effort is calculated as a function of program size and a set of cost drivers are given according to each phase of the software lifecycle.

COCOMO 1

- It is used in waterfall model of software development cycle.
- It helps give estimates required for the efforts and schedule.
- It is based on the linear reuse formula.
- It is based on the assumption of reasonable stable requirements.
- The number of sub-models are 3.
- It has 15 cost drivers assigned to it.
- The size of the software is measured in terms of lines of code.

COCOMO 2

- It is used in non-sequential, rapid development of models.
- It also helps reuse the models of software.
- It helps provide estimates which represent a standard deviation near the most likely estimate.

- It is based on the non-linear formula of reuse.
- It is based on the reuse model that focuses on effort required to understand and estimate.
- The number of sub-models required are 4.
- It is assigned with 17 cost drivers.
- The size of software is stated in terms of object points, function points and lines of code.

## 3.7 Risk analysis and management

- **Risk** is future uncertain events with a probability of occurrence and potential for loss.
- We need to differentiate risks, as potential issues, from the current problems of the project.
- For example, staff storage, because we have not been able to select people with the right technical skills is a current problem, but the threat of our technical persons being hired away by the competition is a risk.
- **Risk Management** is the system of identifying addressing and eliminating these problems before they can damage the project.
- There are three main classifications of risks which can affect a software project:
    - Project risks
    - Technical risks
    - Business risks

**1. Project risks**: Project risks concern differ forms of budgetary, schedule, personnel, resource, and customer-related problems.

**2. Technical risks:** Technical risks concern potential method, implementation, interfacing, testing, and maintenance issue. Most technical risks appear due to the development team's insufficient knowledge about the project.

**3. Business risks:** This type of risks contain risks of building an excellent product that no one need, losing budgetary or personnel commitments, etc.

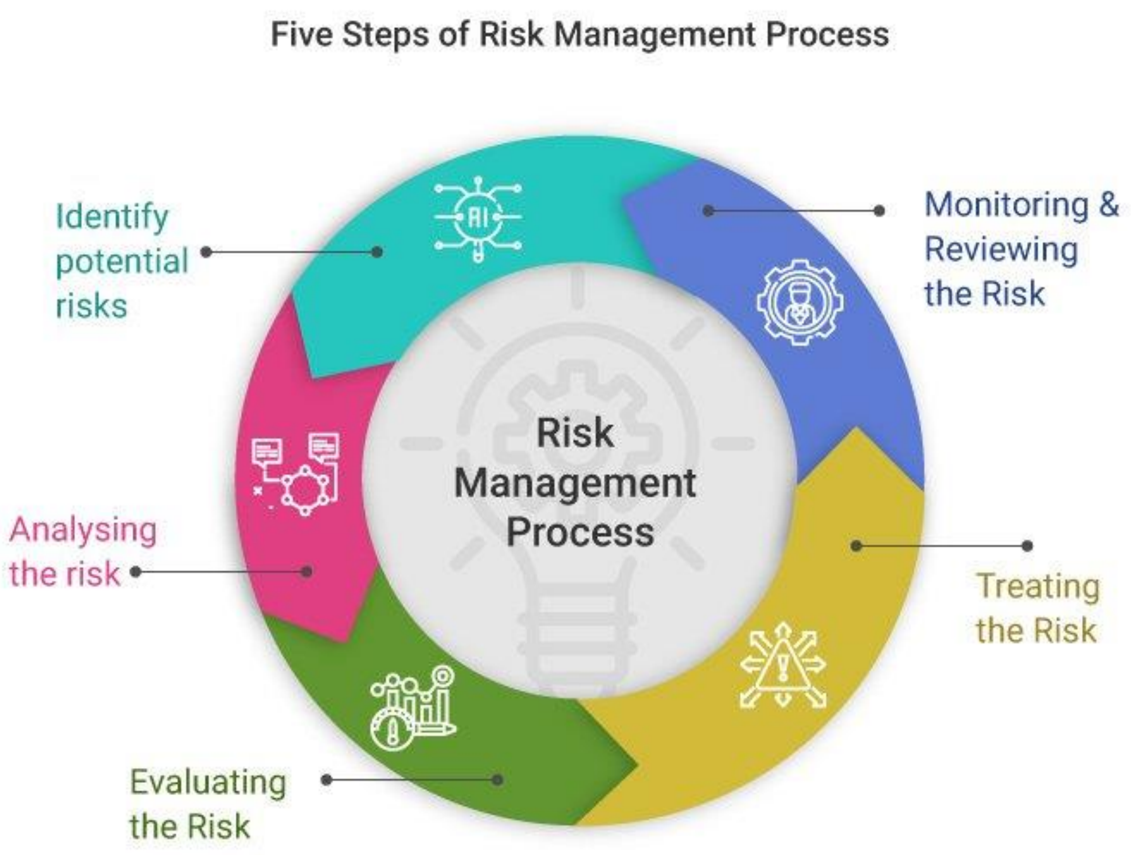
- **Other risk categories**

1. **Known risks**: Those risks that can be uncovered after careful assessment of the project program, the business and technical environment in which the plan is being developed, and more reliable data sources (e.g., unrealistic delivery date)

2. **Predictable risks:** Those risks that are hypothesized from previous project experience (e.g., past turnover)

3. **Unpredictable risks:** Those risks that can and do occur, but are extremely tough to identify in advance.

## 3.8 Risk Management Process



**Step 1: Identify the Risk.** You and your team uncover, recognize and describe risks that might affect your project or its outcomes. There are a number of techniques you can use to find project risks. During this step you start to prepare your Project Risk Register.

**Step 2: Analyze the risk.** Once risks are identified you determine the likelihood and consequence of each risk. You develop an understanding of the nature of the risk and its potential to affect project goals and objectives. This information is also input to your Project Risk Register.

**Step 3: Evaluate or Rank the Risk.** You evaluate or rank the risk by determining the risk magnitude, which is the combination of likelihood and consequence. You make decisions about whether the risk is acceptable or whether it is serious enough to warrant treatment. These risk rankings are also added to your Project Risk Register.

**Step 4: Treat the Risk.** This is also referred to as Risk Response Planning. During this step you assess (evaluate) your highest ranked risks and set out a plan to treat or modify these risks to achieve acceptable risk levels. You create risk mitigation(less severe) strategies, preventive plans and contingency (future) plans in this step. And you add the risk treatment measures for the highest ranking or most serious risks to your Project Risk Register.

**Step 5: Monitor and Review the risk.** This is the step where you take your Project Risk Register and use it to monitor, track and review risks.

## 3.9 Software Configuration Management

In Software Engineering, Software Configuration Management (SCM) is a process to systematically manage, organize, and control the changes in the documents, codes, and other entities during the Software Development Life Cycle.

It is practical in controlling and managing the access to various SCIs (s/w cofig items) e.g., by preventing the two members of a team for checking out the same component for modification at the same time.

- It provides the tool to ensure that changes are being properly implemented.
- It has the capability of describing and storing the various constituent of software.
- SCM is used in keeping a system in a consistent state by automatically producing derived version upon modification of the same component.

Process involved in SCM:

**Identification and Establishment** – Identifying the configuration items from products that compose baselines at given points in time (a baseline is a set of mutually consistent Configuration Items, which has been formally reviewed and agreed upon, and serves as the basis of further development). Establishing relationship among items, creating a mechanism to manages multiple levels of control and procedure for change management system.

**Version control –** Creating versions/specifications of the existing product to build new products from the help of SCM system.

**Change Control -** Change control is a systematic approach for managing all changed made to a product. The purpose is to <u>ensure that no unnecessary changed are made</u>, that all changes are documented, that services are not unnecessarily disrupted and that resources are used efficiently.

**Configuration auditing –** A software configuration audit complements the formal technical review of the process and product. It focuses on the technical correctness of the configuration object that has been modified. It ensures that what is built is what is delivered.

**Reporting –** Providing accurate status and current configuration data to developers, tester, end users, customers and stakeholders through admin guides, user guides, FAQs, Release notes, Memos, Installation Guide, Configuration guide etc .