**Unit 3**

**Software Project Management (SPM) – Software Engineering**

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. This article focuses on discussing [Software Project Management (SPM)](https://www.geeksforgeeks.org/software-engineering-software-project-management-spm/).

**Need for Software Project Management**

Software is a non-physical product. [Software development](https://www.geeksforgeeks.org/what-is-software-development/) is a new stream in business and there is very little experience in building software products. Most of the software products are made to fit clients’ requirements.

**Types of Management in SPM**

**1. Conflict Management**

[Conflict management](https://www.geeksforgeeks.org/organisational-conflicts-consequences-and-types/) is the process to restrict the negative features of conflict while increasing the positive features of conflict. The goal of conflict management is to improve learning and group results including efficacy or performance in an organizational setting. Properly managed conflict can enhance group results.

**2. Risk Management**

[Risk management](https://www.geeksforgeeks.org/risk-management-software-engineering/) is the analysis and identification of risks that is followed by synchronized and economical implementation of resources to minimize, operate and control the possibility or effect of unfortunate events or to maximize the realization of opportunities.

**3. Requirement Management**

It is the process of analyzing, prioritizing, tracking, and documenting requirements and then supervising change and communicating to pertinent stakeholders. It is a continuous process during a project.

**4. Change Management**

[Change management](https://www.geeksforgeeks.org/change-management-in-software-engineering/) is a systematic approach to dealing with the transition or transformation of an organization’s goals, processes, or technologies. The purpose of change management is to execute strategies for effecting change, controlling change, and helping people to adapt to change.

**5. Software Configuration Management**

Software configuration management is the process of controlling and tracking changes in the software, part of the larger cross-disciplinary field of configuration management. [Software configuration management](https://www.geeksforgeeks.org/software-engineering-system-configuration-management/) includes revision control and the inauguration of baselines.

**6. Release Management**

[Release Management](https://www.geeksforgeeks.org/release-management-in-software-engineering/) is the task of planning, controlling, and scheduling the built-in deploying releases. Release management ensures that the organization delivers new and enhanced services required by the customer while protecting the integrity of existing services.

**Aspects of Software Project Management**

The list of focus areas it can tackle and the broad upsides of Software Project Management is:

**1. Planning**

The[software project manager](https://www.geeksforgeeks.org/software-engineering-role-and-responsibilities-of-a-software-project-manager/) lays out the complete project’s blueprint. The project plan will outline the scope, resources, timelines, techniques, strategy, communication, testing, and maintenance steps. SPM can aid greatly here.

**2. Leading**

A software project manager brings together and leads a team of engineers, strategists, programmers, designers, and data scientists. Leading a team necessitates exceptional communication, interpersonal, and leadership abilities. One can only hope to do this effectively if one sticks with the core SPM principles.

**3. Execution**

SPM comes to the rescue here also as the person in charge of software projects (if well versed with SPM/[Agile methodologies](https://www.geeksforgeeks.org/software-engineering-agile-software-development/)) will ensure that each stage of the project is completed successfully. measuring progress, monitoring to check how teams function, and generating status reports are all part of this process.

**4. Time Management**

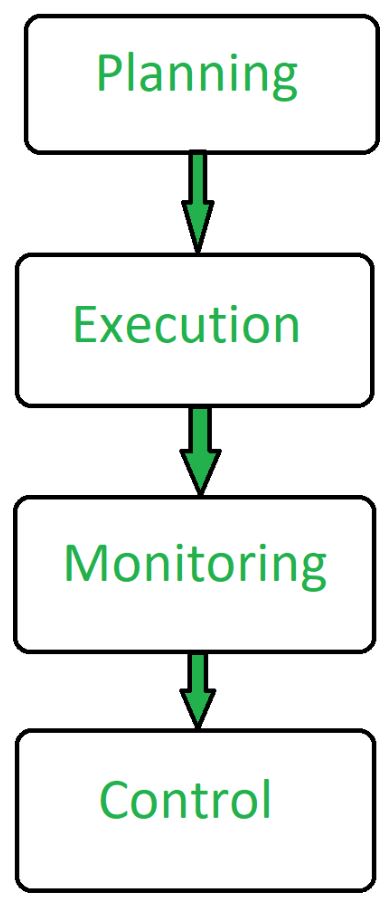
Abiding by a timeline is crucial to completing deliverables successfully. This is especially difficult when managing software projects because changes to the original project charter are unavoidable over time. To assure progress in the face of blockages or changes, software project managers ought to be specialists in managing risk and emergency preparedness. This [Risk Mitigation and   
management](https://www.geeksforgeeks.org/short-note-on-risk-assessment-and-risk-mitigation/) is one of the core tenets of the philosophy of SPM.

**5. Budget**

[Software project managers](https://www.geeksforgeeks.org/software-engineering-role-and-responsibilities-of-a-software-project-manager/), like conventional project managers, are responsible for generating a project budget and adhering to it as closely as feasible, regulating spending, and reassigning funds as needed. SPM teaches us how to effectively manage the monetary aspect of projects to avoid running into a financial crunch later on in the project.

**6. Maintenance**

Software project management emphasizes continuous product testing to find and repair defects early, tailor the end product to the needs of the client, and keep the project on track. The [software project manager](https://www.geeksforgeeks.org/a-day-in-the-life-of-a-project-manager/) makes ensuring that the product is thoroughly tested, analyzed, and adjusted as needed. Another point in favor of SPM.



*Aspects of Project Management*

**Downsides of Software Project Management**

Numerous issues can develop if a Software project manager lacks the necessary expertise or knowledge. Software [Project management](https://www.geeksforgeeks.org/project-management-tutorial/) has several drawbacks, including resource loss, scheduling difficulty, data protection concerns, and interpersonal conflicts between Developers/Engineers/Stakeholders. Furthermore, outsourcing work or recruiting additional personnel to complete the project may result in hefty costs for one’s company.

**1. Costs are High**

Consider spending money on various kinds of project management tools, software, & services if ones engage in [Software Project Management](https://www.geeksforgeeks.org/phases-project-management-processes/) strategies. These initiatives can be expensive and time-consuming to put in place. Because your team will be using them as well, they may require training. One may need to recruit subject-matter experts or specialists to assist with a project, depending on the circumstances. Stakeholders will frequently press for the inclusion of features that were not originally envisioned. All of these factors can quickly drive up a project’s cost.

**2. Complexity will be increased**

Software Project management is a multi-stage, complex process. Unfortunately, some specialists might have a propensity to overcomplicate everything, which can lead to confusion among teams and lead to delays in project completion. Their expressions are very strong and specific in their ideas, resulting in a difficult work atmosphere. Projects having a larger scope are typically more arduous to complete, especially if there isn’t a dedicated team committed completely to the project. Members of cross-functional teams may lag far behind their daily tasks, adding to the overall complexity of the project being worked on.

**3. Overhead in Communication**

Recruits enter your organization when we hire software project management personnel. This provides a steady flow of communication that may or may not match a company’s culture. As a result, it is advised that you maintain your crew as   
small as feasible. The communication overhead tends to skyrocket when a team becomes large enough. When a large team is needed for a project, it’s critical to identify software project managers who can conduct effective communication with a variety of people.

**4. Lack of Originality**

Software Project managers can sometimes provide little or no space for creativity. Team leaders either place an excessive amount of emphasis on management processes or impose hard deadlines on their employees, requiring them to develop and operate code within stringent guidelines. This can stifle innovative thought and innovation that could be beneficial to the project. When it comes to Software project management, knowing when to encourage creativity and when to stick to the project plan is crucial. Without Software [project management](https://www.geeksforgeeks.org/project-management-tutorial/) personnel, an organization can perhaps build and ship code more quickly. However, employing a trained specialist to handle these areas, on the other hand, can open up new doors and help the organization achieve its objectives more quickly and more thoroughly.

# Role and Responsibilities of a software Project Manager – Software Engineering

A software project manager is the most important person inside a team who takes the overall responsibilities to manage the software projects and plays an important role in the successful completion of the projects. This article focuses on discussing the role and responsibilities of a[software project manager](https://www.geeksforgeeks.org/software-engineering-software-project-management-spm/).

## Who is a Project Manager?

A [project manager](https://www.geeksforgeeks.org/6-steps-to-become-a-project-manager/) has to face many difficult situations to accomplish these works. The job responsibilities of a project manager range from invisible activities like building up team morale to highly visible customer presentations. Most of the managers take responsibility for writing the project proposal, project cost estimation, scheduling, project staffing, software process tailoring, [project monitoring](https://www.geeksforgeeks.org/risk-mitigation-monitoring-and-management-rmmm-plan/) and control, software configuration management, risk management, managerial report writing, and presentation, and interfacing with clients.

The tasks of a project manager are classified into two major types:

1. [Project planning](https://www.geeksforgeeks.org/project-plan/)
2. [Project monitoring and control](https://www.geeksforgeeks.org/what-is-monitoring-and-control-in-project-management/)

## Project Planning

Project planning is undertaken immediately after the feasibility study phase and before the starting of the requirement analysis and specification phase. Once a project is feasible, Software project managers start project planning. Project planning is completed before any development phase starts.

1. Project planning involves estimating several characteristics of a project and then plan the project activities based on these estimations.
2. Project planning is done with most care and attention.
3. A wrong estimation can result in schedule slippage.
4. Schedule delay can cause customer dissatisfaction, which may lead to a project failure.
5. Before starting a software project, it is essential to determine the tasks to be performed and properly manage allocation of tasks among individuals involved is the software development.
6. Hence, planning is important as it results in effective software development.
7. Project planning is an organized and integrated management process, which focuses on activities required for successful completion of the project.
8. It prevents obstacles that arise in the project such as changes in projects or organizations objectives, non-availability of resources, and so on.
9. Project planning also helps in better utilization of resources and optimal usage of the allotted time for a project.
10. For effective project planning, in addition to a very good knowledge of various estimation techniques, experience is also very important.

## Objectives of Project Planning

1. It defines the roles and responsibilities of the[project managemen](https://www.geeksforgeeks.org/project-management-tutorial/)t team members .
2. It ensures that the project management team works according to the business objectives.
3. It checks feasibility of the schedule and user requirements.
4. It determines project constraints, several individuals help in planning the project.

## Activities Performed by Project Manager

### 1. Project Estimation

[Project Size Estimation](https://www.geeksforgeeks.org/software-engineering-project-size-estimation-techniques/) is the most significant parameter based on which all other estimations like cost, duration and effort are made.

* **Cost Estimation:** Total expenses to develop the software product is estimated.
* **Time Estimation:** The total time required to complete the project.
* **Effort Estimation:** The effort needed to complete the project is estimated.

### 2. Scheduling

After the completion of the estimation of all the project parameters, scheduling for manpower and other resources is done.

### 3. Staffing

Team structure and staffing plans are made.

### 4. Risk Management

The project manager should identify the unanticipated risks that may occur during [project development risk](https://www.geeksforgeeks.org/different-types-of-risks-in-software-project-development/), analyze the damage that might cause these risks, and take a risk reduction plan to cope with these risks.

### 5. Miscellaneous Plans

This includes making several other plans such as quality assurance plans, configuration management plans, etc.

* **Lead the team:** The project manager must be a good leader who makes a team of different members of various skills and can complete their individual tasks.
* **Motivate the team-member:** One of the key roles of a software project manager is to encourage team members to work properly for the successful completion of the project.
* **Tracking the progress:** The project manager should keep an eye on the progress of the project. A project manager must track whether the project is going as per plan or not. If any problem arises, then take the necessary action to solve the problem. Moreover, check whether the product is developed by maintaining correct coding standards or not.
* **Liaison:** The project manager is the link between the development team and the customer. Project manager analysis the customer requirements and convey it to the development team and keep telling the progress of the project to the customer. Moreover, the project manager checks whether the project is fulfilling the customer’s requirements or not.
* **Monitoring and reviewing:**Project monitoring is a continuous process that lasts the whole time a product is being developed, during which the project manager compares actual progress and cost reports with anticipated reports as soon as possible. While most firms have a formal system in place to track progress, qualified project managers may still gain a good understanding of the project’s development by simply talking with participants.
* **Documenting project report:** The [project manager](https://www.geeksforgeeks.org/project-manager-salary/) prepares the documentation of the project for future purposes. The reports contain detailed features of the product and various techniques. These reports help to maintain and enhance the quality of the project in the future.
* **Reporting:**Reporting project status to the customer and his or her organization is the responsibility of the project manager. Additionally, they could be required to prepare brief, well-organized pieces that summarize key details from in-depth studies.

## Features of a Good Project Manager

1. Knowledge of project estimation techniques.
2. Good decision-making abilities at the right time.
3. Previous experience managing a similar type of projects.
4. Good communication skills to meet the customer satisfaction.
5. A [project manager](https://www.geeksforgeeks.org/6-steps-to-become-a-project-manager/) must encourage all the team members to successfully develop the product.
6. He must know the various type of risks that may occur and the solution to these problems.

# Project planning

# What is Project Plan?

Once a project is found to be possible, computer code project managers undertake project design. Project designing is undertaken and completed even before any development activity starts. Project designing consists of subsequent essential activities: Estimating the subsequent attributes of the project:

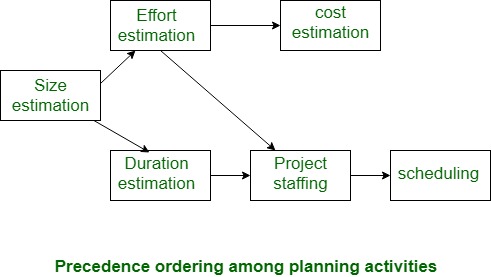
* **Project size:** What’s going to be the downside quality in terms of the trouble and time needed to develop the product?
* **Cost:** What proportion is it reaching to value to develop the project?
* **Duration:** How long is it to reach design plateamended development?
* **Effort:** What proportion of effort would be required?

The effectiveness of the following design activities relies on the accuracy of those estimations.

* planning force and alternative resources
* workers organization and staffing plans
* Risk identification, analysis, and abatement designing
* Miscellaneous arrangements like quality assurance plans, configuration, management arrangements, etc.

**Precedence ordering among project planning activities:**

The different project-connected estimates done by a project manager have already been mentioned. The below diagram shows the order in which vital project coming up with activities is also undertaken. It may be simply discovered that size estimation is the 1st activity. It’s conjointly the foremost basic parameter supported that all alternative coming up with activities square measure dispensed, alternative estimations like the estimation of effort, cost, resource, and project length also are vital elements of the project coming up with.



# Project Size Estimation Techniques – Software Engineering

In the dynamic field of **Software Engineering**, the accurate estimation of project size is a fundamental aspect that influences the success of software projects. **Project Size Estimation Techniques** are essential tools that help in predicting the required resources, time, and cost, thus ensuring the project’s feasibility and efficiency from the onset. It is a crucial aspect of software engineering, as it helps in planning and allocating resources for the project.

## What is Project Size Estimation?

Project size estimation is determining the scope and resources required for the project.

1. It involves assessing the various aspects of the project to estimate the effort, time, cost, and resources needed to complete the project.
2. Accurate project size estimation is important for effective and efficient project planning, management, and execution.

## Importance of Project Size Estimation

Here are some of the reasons why project size estimation is critical in project management:

1. **Financial Planning:**Project size estimation helps in planning the financial aspects of the project, thus helping to avoid financial shortfalls.
2. **Resource Planning:**It ensures the necessary resources are identified and allocated accordingly.
3. **Timeline Creation:**It facilitates the development of realistic timelines and milestones for the project.
4. **Identifying Risks:**It helps to identify potential risks associated with overall project execution.
5. **Detailed Planning:**It helps to create a detailed plan for the project execution, ensuring all the aspects of the project are considered.
6. **Planning Quality Assurance:**It helps in planning quality assurance activities and ensuring that the project outcomes meet the required standards.

## Who Estimates Projects Size?

Here are the key roles involved in estimating the project size:

1. **Project Manager:**Project manager is responsible for overseeing the estimation process.
2. **Subject Matter Experts (SMEs):**SMEs provide detailed knowledge related to the specific areas of the project.
3. **Business Analysts:**Business Analysts help in understanding and documenting the project requirements.
4. **Technical Leads:**They estimate the technical aspects of the project such as system design, development, integration, and testing.
5. **Developers:**They will provide detailed estimates for the tasks they will handle.
6. **Financial Analysts:**They provide estimates related to the financial aspects of the project including labor costs, material costs, and other expenses.
7. **Risk Managers:**They assess the potential risks that could impact the projects’ size and effort.
8. **Clients:**They provide input on project requirements, constraints, and expectations.

## ****Different Methods of Project Estimation****

1. **Expert Judgment:** In this technique, a group of experts in the relevant field estimates the project size based on their experience and expertise. This technique is often used when there is limited information available about the project.
2. **Analogous Estimation:** This technique involves estimating the project size based on the similarities between the current project and previously completed projects. This technique is useful when historical data is available for similar projects.
3. **Bottom-up Estimation:** In this technique, the project is divided into smaller modules or tasks, and each task is estimated separately. The estimates are then aggregated to arrive at the overall project estimate.
4. **Three-point Estimation:** This technique involves estimating the project size using three values: optimistic, pessimistic, and most likely. These values are then used to calculate the expected project size using a formula such as the PERT formula.
5. **Function Points:** This technique involves estimating the project size based on the functionality provided by the software. Function points consider factors such as inputs, outputs, inquiries, and files to arrive at the project size estimate.
6. **Use Case Points:** This technique involves estimating the project size based on the number of use cases that the software must support. Use case points consider factors such as the complexity of each use case, the number of actors involved, and the number of use cases.
7. **Parametric Estimation:** For precise size estimation, mathematical models founded on project parameters and historical data are used.
8. **COCOMO (Constructive Cost Model):** It is an algorithmic model that estimates effort, time, and cost in software development projects by taking into account several different elements.
9. **Wideband Delphi:**Consensus-based estimating method for balanced size estimations that combines expert estimates from anonymous experts with cooperative conversations.
10. **Monte Carlo Simulation:** This technique, which works especially well for complicated and unpredictable projects, estimates project size and analyses hazards using statistical methods and random sampling.

Each of these techniques has its strengths and weaknesses, and the choice of technique depends on various factors such as the project’s complexity, available data, and the expertise of the team.

## Challenges in Project Size Estimation

Project size estimation can be challenging due to multiple factors. Here are some factors that can affect the accuracy and reliability of estimates:

1. **Unclear Requirements:**Initial project requirements can be vague or subject to change, thus making it difficult to estimate accurately.
2. **Lack of Historical Data:**Without access to the data of similar past projects, it becomes difficult to make informed estimates, thus estimates becoming overly optimistic or pessimistic and leading to inaccurate planning.
3. **Interdependencies:**Project with numerous interdependent tasks are harder to estimate due to the complicated interactions between components.
4. **Productivity Variability:**Estimating the productivity of resources and their availability can be challenging due to fluctuations and uncertainties.
5. **Risks:** Identifying and quantifying risks and uncertainties is very difficult. Underestimating the potential risks can lead to inadequate contingency planning, thus causing the project to go off track.

## Improving Accuracy in Project Size Estimation

Improving the accuracy of project size estimation involves a combination of techniques and best practices. Here are some key strategies to enhance estimation accuracy:

1. **Define Clear Requirements:**Ensure all project requirements are thoroughly documented and engage all stakeholders early and frequently to clarify and validate the requirements.
2. **Use Historical Data:**Use data from similar past projects to make informed estimates.
3. **Use Estimation Techniques:**Use various estimation techniques like Analogue Estimation, Parametric Estimation, Bottom-Up Estimation, and Three-Point Estimation.
4. **Break Down the Project:**Use Work Breakdown Structure (WBS) and detailed take analysis to make sure that each task is specific and measurable.
5. **Incorporate Expert Judgement:**Engage subject matter experts and experienced team members to provide input on estimates.

## Future of Project Size Estimation

The future of project size estimation will be shaped by the advancements in technology and methodologies. Here are some key developments that can define the future of project size estimation:

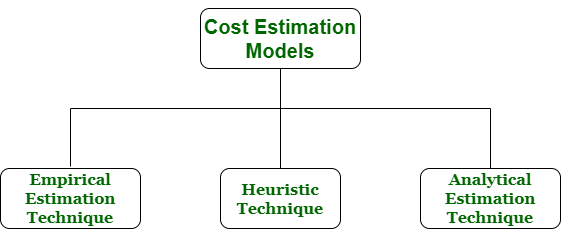
1. **Smarter Technology:** Artificial intelligence (AI) could analyze past projects and code to give more accurate forecasts, considering how complex the project features are.
2. **Data-Driven Insights:** Instead of just lines of code, estimates could consider factors like the number of users, the type of software (mobile app vs. web app), and how much data it handles.
3. **Human-AI Collaboration:**Combining human expertise with AI can enhance the decision-making process in project size estimation.
4. **Collaborative Platforms:**Tools that facilitate collaboration among geographically dispersed teams can help to enhance the project size estimation process.
5. **Agile Methodologies:**The adoption of agile methodologies can promote continuous estimation and iterative refinement.

# Empirical Estimation Techniques

# Cost Estimation Models in Software Engineering

**Cost estimation** simply means a technique that is used to find out the cost estimates. The cost estimate is the financial spend that is done on the efforts to develop and test software in [Software Engineering](https://www.geeksforgeeks.org/software-engineering/). Cost estimation models are some mathematical algorithms or parametric equations that are used to estimate the cost of a product or a project. Various techniques or models are available for cost estimation, also known as Cost Estimation Models.

**Cost Estimation Models as shown below :**

Cost Estimation Models

1. **Empirical Estimation Technique –**Empirical estimation is a technique or model in which empirically derived formulas are used for predicting the data that are a required and essential part of the software project planning step. These techniques are usually based on the data that is collected previously from a project and also based on some guesses, prior experience with the development of similar types of projects, and assumptions. It uses the size of the software to estimate the effort. In this technique, an educated guess of project parameters is made. Hence, these models are based on common sense. However, as there are many activities involved in empirical estimation techniques, this technique is formalized. For example Delphi technique and Expert Judgement technique.
2. **Heuristic Technique –**Heuristic word is derived from a Greek word that means “to discover”. The heuristic technique is a technique or model that is used for solving problems, learning, or discovery in the practical methods which are used for achieving immediate goals. These techniques are flexible and simple for taking quick decisions through shortcuts and good enough calculations, most probably when working with complex data. But the decisions that are made using this technique are necessary to be optimal. In this technique, the relationship among different project parameters is expressed using mathematical equations. The popular heuristic technique is given by [Constructive Cost Model (COCOMO)](https://www.geeksforgeeks.org/software-engineering-cocomo-model/). This technique is also used to increase or speed up the analysis and investment decisions.
3. **Analytical Estimation Technique –** Analytical estimation is a type of technique that is used to measure work. In this technique, firstly the task is divided or broken down into its basic component operations or elements for analyzing. Second, if the standard time is available from some other source, then these sources are applied to each element or component of work. Third, if there is no such time available, then the work is estimated based on the experience of the work. In this technique, results are derived by making certain basic assumptions about the project. Hence, the analytical estimation technique has some scientific basis. [Halstead’s](https://www.geeksforgeeks.org/software-engineering-halsteads-software-metrics/) software science is based on an analytical estimation model.

## Other Cost Estimation Models

1. **Function Point Analysis (FPA):** This technique counts the number and complexity of functions that a piece of software can perform to determine how functional and sophisticated it is. The effort needed for development, testing and maintenance can be estimated using this model.
2. **Putnam Model:**This model is a parametric estimation model that estimates effort, time and faults by taking into account the size of the the programme, the expertise of the development team and other project-specific characteristics.
3. **Price-to-Win Estimation:** Often utilized in competitive bidding, this model is concerned with projecting the expenses associated with developing a particular software project in order to secure a contract. It involves looking at market dynamics and competitors.
4. **Models Based on Machine Learning:**Custom cost estimating models can be built using machine learning techniques including neural networks, regression analysis and decision trees. These models are based on past project data. These models are flexible enough to adjust to changing data and project-specific features.
5. **Function Points Model (IFPUG):**A standardized technique for gauging the functionality of software using function points is offered by the International Function Point Users Group (IFPUG). It is employed to calculate the effort required for software development and maintenance.

# COCOMO Model – Software Engineering

The Constructive Cost Model (COCOMO) is a software cost estimation model that helps predict the effort, cost, and schedule required for a software development project. Developed by Barry Boehm in 1981, COCOMO uses a mathematical formula based on the size of the software project, typically measured in lines of code (LOC).

## What is the COCOMO Model?

The COCOMO Model is a procedural cost estimate model for [software projects](https://www.geeksforgeeks.org/top-software-development-project-ideas/) and is often used as a process of reliably predicting the various parameters associated with making a project such as size, effort, cost, time, and quality. It was proposed by Barry Boehm in 1981 and is based on the study of 63 projects, which makes it one of the best-documented models.

The key parameters that define the quality of any [software product](https://www.geeksforgeeks.org/software-engineering-software-product/), which are also an outcome of COCOMO, are primarily effort and schedule:

1. **Effort:** Amount of labor that will be required to complete a task. It is measured in person-months units.
2. **Schedule:** This simply means the amount of time required for the completion of the job, which is, of course, proportional to the effort put in. It is measured in the units of time such as weeks, and months.

## Types of Projects in the COCOMO Model

In the COCOMO model, software projects are categorized into three types based on their complexity, size, and the development environment. These types are:

1. **Organic:**A software project is said to be an organic type if the team size required is adequately small, the problem is well understood and has been solved in the past and also the team members have a nominal experience regarding the problem.
2. **Semi-detached**: A software project is said to be a Semi-detached type if the vital characteristics such as team size, experience, and knowledge of the various programming environments lie in between organic and embedded. The projects classified as Semi-Detached are comparatively less familiar and difficult to develop compared to the organic ones and require more experience better guidance and creativity. Eg: Compilers or different Embedded Systems can be considered Semi-Detached types.
3. **Embedded:** A software project requiring the highest level of complexity, creativity, and experience requirement falls under this category. Such software requires a larger team size than the other two models and also the developers need to be sufficiently experienced and creative to develop such complex models.

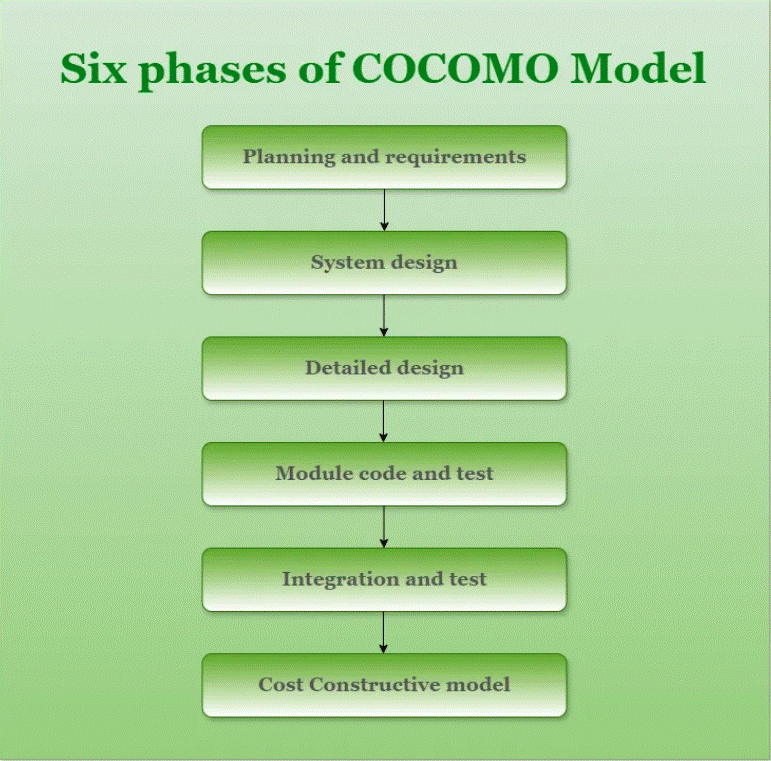
### Comparison of these three types of Projects in COCOMO Model

| **Aspects** | **Organic** | **Semidetached** | **Embedded** |
| --- | --- | --- | --- |
| **Project Size** | 2 to 50 KLOC | 50-300 KLOC | 300 and above KLOC |
| **Complexity** | Low | Medium | High |
| **Team Experience** | Highly experienced | Some experienced as well as inexperienced staff | Mixed experience, includes experts |
| **Environment** | Flexible, fewer constraints | Somewhat flexible, moderate constraints | Highly rigorous, strict requirements |
| **Effort Equation** | E = 2.4(400)1.05 | E = 3.0(400)1.12 | E = 3.6(400)1.20 |
| **Example** | Simple payroll system | New system interfacing with existing systems | Flight control software |

## Detailed Structure of COCOMO Model

Detailed COCOMO incorporates all characteristics of the intermediate version with an assessment of the cost driver’s impact on each step of the[software engineering process.](https://www.geeksforgeeks.org/software-processes-in-software-engineering/) The detailed model uses different effort multipliers for each cost driver attribute. 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.

**The Six phases of detailed COCOMO are:**



*Phases of COCOMO Model*

1. **Planning and requirements:**This initial phase involves defining the scope, objectives, and constraints of the project. It includes developing a project plan that outlines the schedule, resources, and milestones
2. **System design:**: In this phase, the high-level architecture of the software system is created. This includes defining the system’s overall structure, including major components, their interactions, and the data flow between them**.**
3. **Detailed design:**This phase involves creating detailed specifications for each component of the system. It breaks down the system design into detailed descriptions of each module, including data structures, algorithms, and interfaces.
4. **Module code and test:**This involves writing the actual source code for each module or component as defined in the detailed design. It includes coding the functionalities, implementing algorithms, and developing interfaces.
5. **Integration and test:**This phase involves combining individual modules into a complete system and ensuring that they work together as intended.
6. **Cost Constructive model:**The **Constructive Cost Model (COCOMO)**is a widely used method for estimating the cost and effort required for [software development projects](https://www.geeksforgeeks.org/top-software-development-project-ideas/).

Different models of **COCOMO** have been proposed to predict the cost estimation at different levels, based on the amount of accuracy and correctness required. All of these models can be applied to a variety of projects, whose characteristics determine the value of the constant to be used in subsequent calculations. These characteristics of different system types are mentioned below. Boehm’s definition of organic, semidetached, and embedded systems:

## Importance of the COCOMO Model

1. **Cost Estimation:** To help with resource planning and project budgeting, COCOMO offers a methodical approach to [software development cost estimation.](https://www.geeksforgeeks.org/cost-estimation-models-in-software-engineering/)
2. **Resource Management:** By taking team experience, project size, and complexity into account, the model helps with efficient resource allocation.
3. **Project Planning**: COCOMO assists in developing practical project plans that include attainable objectives, due dates, and benchmarks.
4. **Risk management**: Early in the development process, COCOMO assists in identifying and mitigating potential hazards by including risk elements.
5. **Support for Decisions**: During project planning, the model provides a quantitative foundation for choices about scope, priorities, and resource allocation.
6. **Benchmarking**: To compare and assess various software development projects to industry standards, COCOMO offers a benchmark.
7. **Resource Optimization:** The model helps to maximize the use of resources, which raises productivity and lowers costs.

## Types of COCOMO Model

There are three types of COCOMO Model:

* Basic COCOMO Model
* Intermediate COCOMO Model
* Detailed COCOMO Model

### 1. Basic COCOMO Model

The Basic COCOMO model is a straightforward way to estimate the effort needed for a software development project. It uses a simple mathematical formula to predict how many person-months of work are required based on the size of the project, measured in thousands of lines of code (KLOC).

It estimates effort and time required for development using the following expression:

*E = a\*(KLOC)b PM*

*Tdev = c\*(E)d*

*Person required = Effort/ Time*

*Where,*

*E is effort applied in Person-Months*

*KLOC is the estimated size of the software product indicate in Kilo Lines of Code*

*Tdev is the development time in months*

*a, b, c are constants determined by the category of software project given in below table.*

The above formula is used for the cost estimation of 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 software projects are:

| **Software Projects** | **a** | **b** | **c** | **d** |
| --- | --- | --- | --- | --- |
| 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 |

1. The effort is measured in Person-Months and as evident from the formula is dependent on Kilo-Lines of code. The development time is measured in months.
2. These formulas are used as such in the Basic Model calculations, as not much consideration of different factors such as reliability, and expertise is taken into account, henceforth the estimate is rough.

### ****Example of Basic COCOMO Model****

Suppose that a Basic project was estimated to be 400 KLOC (kilo lines of code). Calculate effort and time for each of the three modes of development. All the constants value provided in the following table:

**Solution**

From the above table we take the value of constant a,b,c and d.

1. For organic mode,
   * effort = 2.4 × (400)1.05  ≈ 1295 person-month.
   * dev. time = 2.5 × (1295)0.38 ≈ 38 months.
2. For semi-detach mode,
   * effort = 3 × (400)1.12  ≈ 2462 person-month.
   * dev. time = 2.5 × (2462)0.35 ≈  38 months.
3. For Embedded mode,
   * effort = 3.6 × (400)1.20  ≈ 4772 person-month.
   * dev. time = 2.5 × (4772)0.32 ≈ 38 months.

**Below are the programs for Basic COCOMO Model:**

C++JavaPythonC#JavaScript

*// C++ program to implement basic COCOMO*

#include *<bits/stdc++.h>*

**using** **namespace** **std**;

*// Function For rounding off float to int*

int fround(float x)

{

int a;

x = x + 0.5;

a = x;

**return** (a);

}

*// Function to calculate parameters*

*// of Basic COCOMO*

void calculate(float table[][4], int n,

char mode[][15], int size)

{

float effort, time, staff;

int model;

*// Check the mode according to size*

*// organic*

**if** (size >= 2 && size <= 50)

model = 0;

*// semi-detached*

**else** **if** (size > 50 && size <= 300)

model = 1;

*// embedded*

**else** **if** (size > 300)

model = 2;

cout << "The mode is " << mode[model];

*// Calculate Effort*

effort = table[model][0] \* pow(size,

table[model][1]);

*// Calculate Time*

time = table[model][2] \* pow(effort,

table[model][3]);

*// Calculate Persons Required*

staff = effort / time;

*// Output the values calculated*

cout << "**\n**Effort = " << effort <<

" Person-Month";

cout << "**\n**Development Time = " << time <<

" Months";

cout << "**\n**Average Staff Required = " <<

fround(staff) << " Persons";

}

*// Driver code*

int main()

{

float table[3][4] = {2.4, 1.05, 2.5, 0.38, 3.0, 1.12,

2.5, 0.35, 3.6, 1.20, 2.5, 0.32};

char mode[][15]

= {"Organic", "Semi-Detached", "Embedded"};

int size = 4;

calculate(table, 3, mode, size);

**return** 0;

}

**Output**

The mode is Organic

Effort = 10.289 Person-Month

Development Time = 6.06237 Months

Average Staff Required = 2 Persons

### 2. Intermediate COCOMO Model

The basic COCOMO model assumes that the effort is only a function of the number of lines of code and some constants evaluated according to the different software systems. However, in reality, no system’s effort and schedule can be solely calculated based on Lines of Code. For that, various other factors such as reliability, experience, and Capability. These factors are known as **Cost Drivers (multipliers)**and the Intermediate Model utilizes 15 such drivers for cost estimation.

**Classification of Cost Drivers and their Attributes:**

The cost drivers are divided into four categories

**Product attributes:**

* Required [software reliability](https://www.geeksforgeeks.org/software-engineering-hardware-reliability-vs-software-reliability/) extent
* Size of the application database
* The complexity of the product

**Hardware attributes**

* Run-time performance constraints
* Memory constraints
* The volatility of the virtual machine environment
* Required turnabout time

**Personal attributes**

* Analyst capability
* Software engineering capability
* Application experience
* Virtual machine experience
* Programming language experience

**Project attributes**

* Use of [software tools](https://www.geeksforgeeks.org/what-is-software-tool/)
* Application of [software engineering methods](https://www.geeksforgeeks.org/5-most-commonly-used-software-development-methodologies/)
* Required development schedule

Each of the 15 such attributes can be rated on a six-point scale ranging from “very low” to “extra high” in their relative order of importance. Each attribute has an effort multiplier fixed as per the rating. Table give below represents Cost Drivers and their respective rating:

*The****Effort Adjustment Factor (EAF)****is determined by multiplying the effort multipliers associated with each of the 15 attributes.*

The Effort Adjustment Factor (EAF) is employed to enhance the estimates generated by the basic COCOMO model in the following expression:

**Intermediate COCOMO Model equation:**

*E = a\*(KLOC)b \* EAF PM*

*Tdev = c\*(E)d*

*Where,*

* *E is effort applied in Person-Months*
* *KLOC is the estimated size of the software product indicate in Kilo Lines of Code*
* *EAF is the Effort Adjustment Factor (EAF) is a multiplier used to refine the effort estimate obtained from the basic COCOMO model.*
* *Tdev is the development time in months*
* *a, b, c are constants determined by the category of software project given in below table.*

The constant values a, b, c, and d for the Basic Model for the different categories of the software projects are:

| **Software Projects** | **a** | **b** | **c** | **d** |
| --- | --- | --- | --- | --- |
| Organic | 3.2 | 1.05 | 2.5 | 0.38 |
| Semi-Detached | 3.0 | 1.12 | 2.5 | 0.35 |
| Embedded | 2.8 | 1.20 | 2.5 | 0.32 |

### 3. Detailed COCOMO Model

Detailed COCOMO goes beyond Basic and Intermediate COCOMO by diving deeper into project-specific factors. It considers a wider range of parameters, like team experience, development practices, and software complexity. By analyzing these factors in more detail, Detailed COCOMO provides a highly accurate estimation of effort, time, and cost for software projects. It’s like zooming in on a project’s unique characteristics to get a clearer picture of what it will take to complete it successfully.

## CASE Studies and Examples

1. **NASA Space Shuttle Software Development:** NASA estimated the time and money needed to build the software for the Space Shuttle program using the COCOMO model. NASA was able to make well-informed decisions on resource allocation and project scheduling by taking into account variables including project size, complexity, and team experience.
2. **Big Business Software Development:**The COCOMO model has been widely used by big businesses to project the time and money needed to construct intricate business software systems. These organizations were able to better plan and allocate resources for their software projects by using COCOMO’s estimation methodology.
3. **Commercial Software goods:**The COCOMO methodology has proven advantageous for software firms that create commercial goods as well. These businesses were able to decide on pricing, time-to-market, and resource allocation by precisely calculating the time and expense of building new software products or features.
4. **Academic Research Initiatives:**To estimate the time and expense required to create software prototypes or carry out experimental studies, academic research initiatives have employed COCOMO. Researchers were able to better plan their projects and allocate resources by using COCOMO’s estimate approaches.

## Advantages of the COCOMO Model

1. **Systematic cost estimation:**Provides a systematic way to estimate the cost and effort of a software project.
2. **Helps to estimate cost and effort:**This can be used to estimate the cost and effort of a software project at different stages of the development process.
3. **Helps in high-impact factors:** Helps in identifying the factors that have the greatest impact on the cost and effort of a software project.
4. **Helps to evaluate the feasibility of a project:**This can be used to evaluate the feasibility of a software project by estimating the cost and effort required to complete it.

## Disadvantages of the COCOMO Model

1. **Assumes project size as the main factor:**Assumes that the size of the software is the main factor that determines the cost and effort of a software project, which may not always be the case.
2. **Does not count development team-specific characteristics:**Does not take into account the specific characteristics of the development team, which can have a significant impact on the cost and effort of a software project.
3. **Not enough precise cost and effort estimate:**This does not provide a precise estimate of the cost and effort of a software project, as it is based on assumptions and averages.

## Best Practices for Using COCOMO

1. **Recognize the Assumptions Underpinning the Model:**Become acquainted with the COCOMO model’s underlying assumptions, which include its emphasis on team experience, size, and complexity. Understand that although COCOMO offers useful approximations, project results cannot be predicted with accuracy.
2. **Customize the Model**: Adapt COCOMO’s inputs and parameters to your project’s unique requirements, including organizational capacity, development processes, and industry standards. By doing this, you can be confident that the estimations produced by COCOMO are more precise and appropriate for your situation.
3. **Utilize Historical Data:** To verify COCOMO inputs and improve estimating parameters, collect and examine historical data from previous projects. Because real-world data takes project-specific aspects and lessons learned into account, COCOMO projections become more accurate and reliable.
4. **Verify and validate:** Compare COCOMO estimates with actual project results, and make necessary adjustments to estimation procedures in light of feedback and lessons discovered. Review completed projects to find errors and enhance future project estimation accuracy.
5. **Combine with Other Techniques:** To reduce biases or inaccuracies in any one method and to triangulate results, add COCOMO estimates to other estimation techniques including expert judgment, similar estimation, and bottom-up estimation.

**Halstead‟s software science**

Halstead’s Software metrics are a set of measures proposed by Maurice Halstead to evaluate the complexity of a software program. These metrics are based on the number of distinct operators and operands in the program and are used to estimate the effort required to develop and maintain the program. These metrics provide a quantitative assessment of software complexity, aiding in software development, maintenance, and quality assurance processes. They include measures such as program length, vocabulary, volume, difficulty, and effort, calculated based on the number of unique operators and operands in a program. Halstead’s metrics help developers understand and manage software complexity, identify potential areas for optimization, and improve overall software quality.

## What is Halstead’s Software Metrics?

Halstead’s Software Metrics, developed by Maurice Halstead in 1977, are a set of measures used to quantify various aspects of software programs. According to Halstead’s, “A computer program is an implementation of an algorithm considered to be a collection of tokens which can be classified as either operators or operand”. This means that the program consists of various symbols and data elements that are either performing actions (operators) or upon which actions are performed (operands). This distinction helps in understanding and analyzing the structure and behavior of the program.

## Token Count

In Halstead’s Software metrics, a computer program is defined as a collection of tokens that can be described as operators or operands. These tokens are used to analyze the complexity and volume of a program. Operators are symbols that represent actions, while operands are the entities on which the operators act. All software science metrics can be specified using these basic symbols. These symbols are referred to as tokens. By counting and analyzing these tokens, Halstead’s metrics provide insights into the complexity, effort, and quality of software code.

In Halstead’s Software Metrics:

***n1 = Number of distinct operators.***

***n2 = Number of distinct operands.***

***N1 = Total number of occurrences of operators.***

***N2 = Total number of occurrences of operands.***

## Field of Halstead Metrics

### ****Program length (N):****

This is the total number of operator and operand occurrences in the program.

### ****Vocabulary size (n):****

This is the total number of distinct operators and operands in the program.

### ****Program volume (V):****

This is the product of program length (N) and the logarithm of vocabulary size (n),

***i.e., V = N\*log2(n)***

### ****Program level (L):****

This is the ratio of the number of operator occurrences to the number of operand occurrences in the program,

***i.e., L = n1/n2***

where n1 is the number of operator occurrences and n2 is the number of operand occurrences.

### ****Program difficulty (D):****

This is the ratio of the number of unique operators to the total number of operators in the program,

***i.e., D = (n1/2) \* (N2/n2)***

### ****Program effort (E):****

This is the product of program volume (V) and program difficulty

***(D), i.e., E = V\*D***

### ****Time to implement (T):****

This is the estimated time required to implement the program, based on the program effort (E) and a constant value that depends on the programming language and development environment.

Halstead’s software metrics can be used to estimate the size, complexity, and effort required to develop and maintain a software program. However, they have some limitations, such as the assumption that all operators and operands are equally important, and the assumption that the same set of metrics can be used for different programming languages and development environments.

## Halstead’s Software Metrices

Halstead’s Software Metrics are:

### ****Halstead Program Length****

Halstead Program Length (N) in Halstead’s Software Metrics refers to the total number of tokens in a program. Where tokens are the smallest individual units of code such as operators, operands, keywords, and identifiers.

***N = N1 + N2***

The estimated program length is denoted by N^ and is given by the formula:

***N^ = n1log2n1 + n2log2n2***

Several alternative formulas have been proposed to estimate program length, including:

***NJ = log2(n1!) + log2(n2!)NB = n1 \* log2n2 + n2 \* log2n1NC = n1 \* sqrt(n1) + n2 \* sqrt(n2)NS = (n \* log2n) / 2***

*Halstead’s,*

### ****Halstead Vocabulary****

The total number of unique operators and unique operand occurrences.

***n = n1 + n2***

### ****Program Volume****

Proportional to program size, represents the size, in bits, of space necessary for storing the program. This parameter is dependent on specific algorithm implementation. The properties V, N, and the number of lines in the code are shown to be linearly connected and equally valid for measuring relative program size.

***V = Size \* (log2 vocabulary) = N \* log2(n)***

The unit of measurement of volume is the common unit for size “bits”. It is the actual size of a program if a uniform binary encoding for the vocabulary is used. And

***error = Volume / 3000***

### ****Potential Minimum Volume****

The potential minimum volume V\* is defined as the volume of the most succinct program in which a problem can be coded.

***V\* = (2 + n2\*) \* log2(2 + n2\*)***

Here, n2\* is the count of unique input and output parameters

### ****Program Level****

To rank the programming languages, the level of abstraction provided by the programming language, Program Level (L) is considered. The higher the level of a language, the less effort it takes to develop a program using that language.

### *L = V\* / V*

The value of L ranges between zero and one, with L=1 representing a program written at the highest possible level (i.e., with minimum size).   
And estimated program level is

***L^ =2 \* (n2) / (n1)(N2)***

### ****Program Difficulty****

This parameter shows how difficult to handle the program is.

***D = (n1 / 2) \* (N2 / n2)******D = 1 / L***

As the volume of the implementation of a program increases, the program level decreases and the difficulty increases. Thus, programming practices such as redundant usage of operands, or the failure to use higher-level control constructs will tend to increase the volume as well as the difficulty.

### ****Programming Effort****

Measures the amount of mental activity needed to translate the existing algorithm into implementation in the specified program language.

***E = V / L = D \* V = Difficulty \* Volume***

### ****Language Level****

Shows the algorithm implementation program language level. The same algorithm demands additional effort if it is written in a [low-level program language](https://www.geeksforgeeks.org/difference-between-high-level-and-low-level-languages). For example, it is easier to program in Pascal than in[Assembler](https://www.geeksforgeeks.org/introduction-of-assembler).

***L’ = V / D / D lambda = L \* V\* = L2 \* V***

### ****Intelligence Content****

Determines the amount of intelligence presented (stated) in the program This parameter provides a measurement of program complexity, independently of the programming language in which it was implemented.

***I = V / D***

### ****Programming Time****

Shows time (in minutes) needed to translate the existing algorithm into implementation in the specified program language.

***T = E / (f \* S)***

The concept of the processing rate of the human brain, developed by psychologist John Stroud, is also used. Stoud defined a moment as the time required by the human brain to carry out the most elementary decision. The Stoud number S is therefore Stoud’s moments per second with:   
5 <= S <= 20. Halstead uses 18. The value of S has been empirically developed from psychological reasoning, and its recommended value for programming applications is 18.   
Stroud number S = 18 moments / second   
seconds-to-minutes factor f = 60

## Example of Halstead’s Software Metrices

Before we look at the example, let’s review the counting rules for a C program.

### Counting Rules for C Language

1. Comments are not considered.
2. The identifier and function declarations are not considered
3. All the variables and constants are considered operands.
4. Global variables used in different modules of the same program are counted as multiple occurrences of the same variable.
5. Local variables with the same name in different functions are counted as unique operands.
6. Functions calls are considered operators.
7. All looping statements e.g., do {…} while ( ), while ( ) {…}, for ( ) {…}, all control statements e.g., if ( ) {…}, if ( ) {…} else {…}, etc. are considered as operators.
8. In control construct switch ( ) {case:…}, switch as well as all the case statements are considered as operators.
9. The reserve words like return, default, continue, break, size, etc., are considered operators.
10. All the brackets, commas, and terminators are considered operators.
11. GOTO is counted as an operator and the label is counted as an operand.
12. The unary and binary occurrences of “+” and “-” are dealt with separately. Similarly “\*” (multiplication operator) is dealt with separately.
13. In the array variables such as “array-name [index]” “array-name” and “index” are considered as operands and [ ] is considered as operator.
14. In the structure variables such as “struct-name, member-name” or “struct-name -> member-name”, struct-name, and member-name are taken as operands, and ‘.’, ‘->’ are taken as operators. Some names of member elements in different structure variables are counted as unique operands.
15. All the hash directives are ignored.

**Let’s examine the following C program**

int sort (int x[ ], int n)  
  
{  
 int i, j, save, im1;  
 /\*This function sorts array x in ascending order \*/  
 If (n< 2) return 1;  
 for (i=2; i< =n; i++)  
 {  
 im1=i-1;  
 for (j=1; j< =im1; j++)  
 if (x[i] < x[j])  
 {  
 Save = x[i];  
 x[i] = x[j];  
 x[j] = save;  
 }  
 }  
 return 0;  
}

### ****Explanation****

| **Operators** | **Occurrences** | **Operands** | **Occurrences** |
| --- | --- | --- | --- |
| int | 4 | sort | 1 |
| () | 5 | x | 7 |
| , | 4 | n | 3 |
| [] | 7 | i | 8 |
| if | 2 | j | 7 |
| < | 2 | save | 3 |
| ; | 11 | im1 | 3 |
| for | 2 | 2 | 2 |
| = | 6 | 1 | 3 |
| – | 1 | 0 | 1 |
| <= | 2 | – | – |
| ++ | 2 | – | – |
| return | 2 | – | – |
| {} | 3 | – | – |
| n1=14 | N1=53 | n2=10 | N2=38 |

**Here are the calculated Halstead metrics for the given C program:**

*Program Length (N) = 91 Vocabulary (n) = 24 Volume (V) = 417.23 bits Estimated Program Length (N^) = 86.51 Unique Operands Used as Both Input and Output (n2\* = 3 (x: array holding integer to be sorted. This is used both as input and output) Potential Volume (V\*) = 11.6 Program Level (L) = 0.027 Difficulty (D) = 37.03 Estimated Program Level (L^) = 0.038 Effort (T) = 610 seconds*

## ****Advantages of Halstead Metrics****

* It is simple to calculate.
* It measures the overall quality of the programs.
* It predicts the rate of error.
* It predicts maintenance effort.
* It does not require a full analysis of the programming structure.
* It is useful in scheduling and reporting projects.
* It can be used for any programming language.
* Easy to use: The metrics are simple and easy to understand and can be calculated quickly using automated tools.
* Quantitative measure: The metrics provide a quantitative measure of the complexity and effort required to develop and maintain a software program, which can be useful for project planning and estimation.
* Language independent: The metrics can be used for different programming languages and development environments.
* Standardization: The metrics provide a standardized way to compare and evaluate different software programs.

## Disadvantages of Halstead Metrics

* It depends on the complete code.
* It has no use as a predictive estimating model.
* Limited scope: The metrics focus only on the complexity and effort required to develop and maintain a software program, and do not take into account other important factors such as reliability, maintainability, and usability.
* Limited applicability: The metrics may not be applicable to all types of software programs, such as those with a high degree of interactivity or real-time requirements.
* Limited accuracy: The metrics are based on a number of assumptions and simplifications, which may limit their accuracy in certain situations.

## Conclusion

Halstead’s software metrics offer a quantitative approach to assessing a program’s complexity. They include measures like program length (total operators and operands), vocabulary (unique operators and operands), and volume (a measure of the program’s size in bits). These metrics help understand the diversity and size of the code, aiding in evaluating its complexity and potential maintainability. Overall, Halstead’s metrics provide valuable insights into the code’s structure and complexity.

**Staffing level estimation**

Staffing is the art of acquiring, developing, and maintaining a satisfactory and satisfied workforce. Staffing is that function by which a manager builds an organization through the recruitment, selection, and development of the individual, which also includes a series of activities. It ensures that the organization has the right number of people at the right places, at the right time, and performing the right thing.

As we know, the prime concern of the staffing function in the management process is in the fulfilment of the manpower requirements within an organization. These requirements may arise in the case of starting a new enterprise or expanding the existing one. It may also arise as the need for replacing those who quit, retire, transfer, or are promoted from the job. In any case, the need for ‘the right person for the right job, at the right time’ needs an emphasis.

## Staffing Process

The process of staffing consists of several interrelated activities, such as planning for human resources requirements, recruitment, selection, training development, remuneration, and so on. These activities together make the staffing process. Therefore, these are called elements or steps of the staffing process.

## ****1. Manpower Planning****

Human resource management is a process of determining the number and type of personnel required for filling the vacant job in an organization. Manpower requirements involve two kinds of analysis, i.e., workload analysis and workforce analysis. Workload analysis involves determining the number and type of employees required to perform various jobs and achieve organizational objectives. Workforce analysis shows the number and type of human resources available with an organization.  
The difference between workload and workforce is calculated to determine shortage and surplus of manpower. Excess workload indicates understaffing, i.e., the need of appointing more people and excess workforce indicates overstaffing, i.e., need to remove or transfer some employees to other places.

## ****2. Recruitment****

After estimating manpower requirements, the second step in the process of staffing is recruitment. Recruitment refers to a process of searching for prospective employees and encouraging them to apply for jobs in the organization. It involves identifying various resources of human force and attracting them to apply for the job. The main purpose of a requirement is to create a pool of applicants by a large number of qualified candidates. Recruitment can be done by both internal and external sources of recruitment. Internal sources may be used to a limited extent, and to get fresh talent and a wider choice, external sources can be used.

## ****3. Selection****

Selection is the process of choosing and appointing the right candidates for various job positions in the organization. It is treated as a negative process because it involves the rejection of some candidates. There are many steps involved in the process of employee selection. These steps include preliminary screening, filling-in application, written test, interviews, medical examination, checking references, and issuing a letter of appointment to the candidates. The most suitable candidates who meet the requirement of the vacant job are selected. The process of selection serves two important purposes, firstly, it ensures that the organization gets the best among the available candidates, and secondly, it boosts ups the self-esteem and prestige of the candidates.

## ****4. Placement and Orientation****

After selection, an appropriate job is assigned to each selected person. Placement is the process of matching the candidates with the jobs in the organization. Under this process, every selected candidate is assigned a job most suitable for him. The purpose of placement is to fit the right person to the right job so that the efficiency of work is high and the employees get personal satisfaction. Correct placement helps to reduce labour turnover and absenteeism. Here, orientation means introducing new employees to the organization. It is the process of introducing and familiarizing newly appointed candidates with their job, work groups and the organization so that they may feel at home in the new environment.

## ****5. Training and Development****

People are in search of careers and not jobs. Every individual must be given a chance to rise to the top. The most favourable way for this to happen is to promote employee learning. For this, organizations either provide training themselves within the organization or through external institutions. This is beneficial for the organization as well. If the employees are motivated enough, it will increase their competence and will be able to perform even better for the organization with greater efficiency and productivity. By providing such opportunities to its employees for career advancement, the organization captivates the interest and holds on of its talented employees. The majority of the organization has a distinct department for this purpose, that is, the Human Resource Department. Though in small organizations, the line manager has to do all the managerial functions viz, planning, organizing, staffing, controlling, and directing. The process of staffing further involves three more stages.

## ****6. Performance appraisal****

After training the employees and having them on the job for some time, there should be an evaluation done on their performance. Every organization has its means of appraisal whether formal or informal. Appraisal refers to the evaluation of the employees of the organization based on their past or present performance by some pre-decided standards. The employee should be well aware of his standards and his superior is responsible for proving feedback on his performance. The process of performance appraisal, thus includes specifying the job, performing appraisal performance, and providing feedback.

## ****7. Promotion and Career planning****

It has now become important for all organizations to deal with career-related issues and promotional routes for employees. The managers should take care of the activities that serve the long-term interests of the employees. They should be encouraged from time to time, which will help the employees to grow and find their true potential. Promotions are an essential part of any employee’s career. Promotion refers to the transferring of employees from their current positions to a higher level increasing their responsibilities, authority and pay.

## ****8. Compensation****

Every organization needs to set up plans for the salary and wages of the employees. There are several ways to develop payment plans for the employees depending upon the significance of the job. The worth of the job needs to be decided. Therefore, all kinds of payments or rewards provided to the employees is referred to as compensation. The compensation may be in the form of direct financial payments, such as salary, wages, bonuses, etc., or indirect payments like insurance or vacations provided to the employee.

## Aspects or Components of Staffing

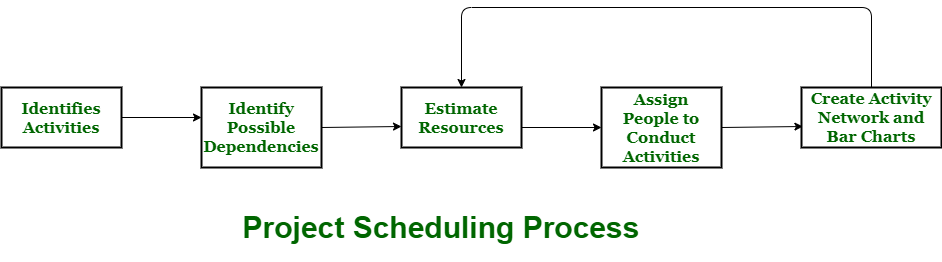
There are three aspects or components of staffing, namely, recruitment, selection, and training. They are defined below:

* **Recruitment:**It is the process of finding potential candidates for a particular job in an organization. The process of recruitment involves persuading people to apply for the available positions in the organization.
* **Selection:**It is the process of recognizing potential and hiring the best people out of several possible candidates. This is done by shortlisting and choosing the deserving and eliminating those who are not suitable for the job.
* **Training:** It is the process that involves providing the employees with an idea of the type of work they are supposed to do and how it is to be done. It is a way of keeping the employees updated on the way of work in an organization and the new and advanced technologies.

**Scheduling**

A schedule in your project’s time table actually consists of sequenced activities and milestones that are needed to be delivered under a given period of time.

**Project schedule** simply means a mechanism that is used to communicate and know about that tasks are needed and has to be done or performed and which organizational resources will be given or allocated to these tasks and in what time duration or time frame work is needed to be performed. Effective project scheduling leads to success of project, reduced cost, and increased customer satisfaction. Scheduling in project management means to list out activities, deliverables, and milestones within a project that are delivered. It contains more notes than your average weekly planner notes. The most common and important form of project schedule is Gantt chart.



**Process :**  
The manager needs to estimate time and resources of project while scheduling project. All activities in project must be arranged in a coherent sequence that means activities should be arranged in a logical and well-organized manner for easy to understand. Initial estimates of project can be made optimistically which means estimates can be made when all favorable things will happen and no threats or problems take place.

The total work is separated or divided into various small activities or tasks during project schedule. Then, Project manager will decide time required for each activity or task to get completed. Even some activities are conducted and performed in parallel for efficient performance. The project manager should be aware of fact that each stage of project is not problem-free.

**Problems arise during Project Development Stage :**

* People may leave or remain absent during particular stage of development.
* Hardware may get failed while performing.
* Software resource that is required may not be available at present, etc.

The project schedule is represented as set of chart in which work-breakdown structure and dependencies within various activities are represented. To accomplish and complete project within a given schedule, required resources must be available when they are needed. Therefore, resource estimation should be done before starting development.

**Resources required for Development of Project :**

* Human effort
* Sufficient disk space on server
* Specialized hardware
* Software technology
* Travel allowance required by project staff, etc.

**Advantages of Project Scheduling :**  
There are several advantages provided by project schedule in our project management:

* It simply ensures that everyone remains on same page as far as tasks get completed, dependencies, and deadlines.
* It helps in identifying issues early and concerns such as lack or unavailability of resources.
* It also helps to identify relationships and to monitor process.
* It provides effective budget management and risk mitigation.

**Organization and Team Structures**

# Software Project Team Organization

Effective software project team organization is crucial for project success. Common structures include hierarchical, chief-programmer, matrix, egoless, and democratic teams, each with unique benefits and challenges. The right structure depends on project needs and team dynamics. Choosing the appropriate organization enhances communication, productivity, and software quality.

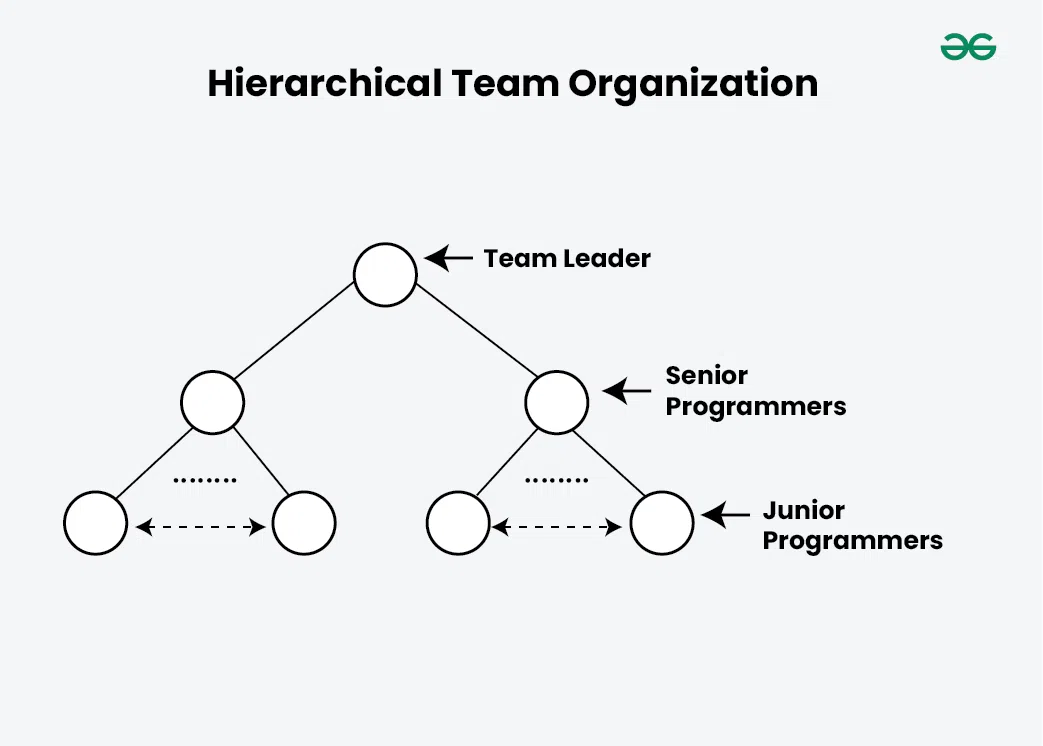
## Software Project Team Organization

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

1. Hierarchical team organization
2. Chief-programmer team organization
3. Matrix team, organization
4. Egoless team organization
5. Democratic team organization

### ****Hierarchical team organization****

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



*Hierarchical Team Organization*

**Below are some benefits of Hierarchical Team Organization:**

* It limits the number of communication paths and still allows for the needed communication.
* It can be expanded over multiple levels.
* It is well suited for the development of hierarchical software products.
* Large software projects may have several levels.

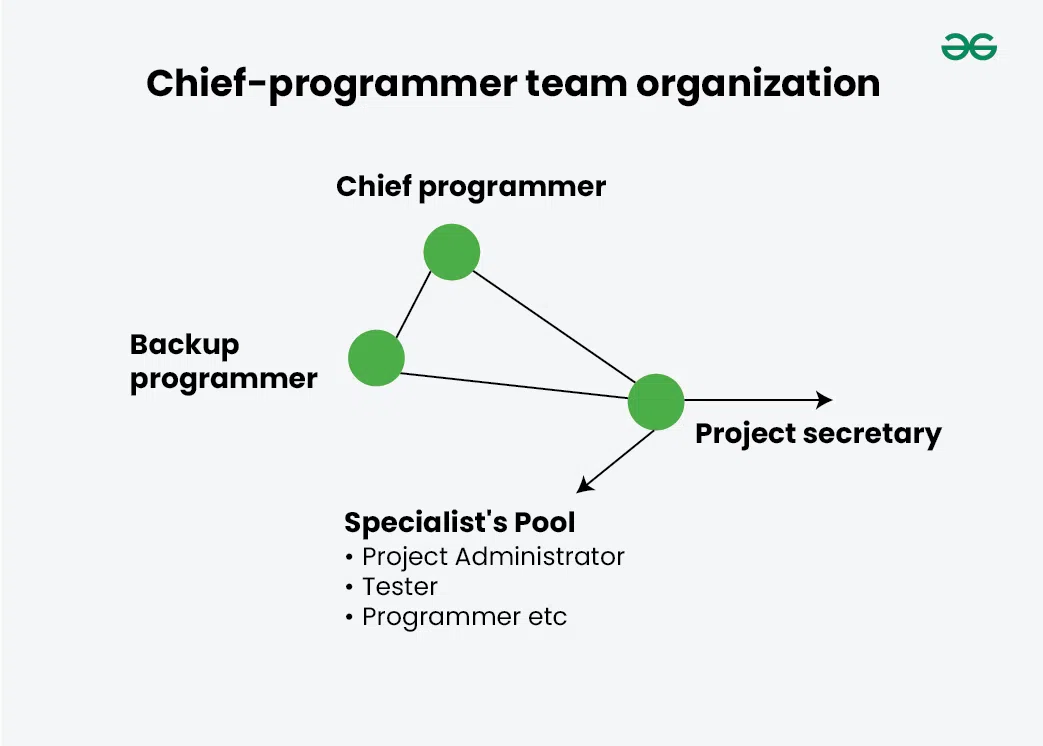
**Below are some Limitations of hierarchical team organization:**

* As information has to travel up the levels, it may get distorted.
* Levels in the hierarchy often judge people socially and financially.
* Most technically competent programmers tend to be promoted to management positions which may result in the loss of good programmers and also bad managers.

### ****Chief-programmer team organization****

The chief programmer team organization is composed of a small team consisting of the following team members :

* **The Chief programmer** is the person who is actively involved in the planning, specification and design process and ideally in the implementation process.
* **The project assistant**is the closest technical co-worker of the chief programmer.
* **The project secretary**relieves the chief programmer and all other programmers of administration tools.
* **Specialists:** These people select the implementation language, implement individual system components employ software tools, and carry out tasks.



*Chief-programmer team organization*

**Advantages of Chief-programmer team organization**

* Centralized decision-making
* Reduced communication paths
* Small teams are more productive than large teams
* The chief programmer is directly involved in system development and can exercise better control functions.

**Disadvantages of Chief-programmer team organization:**

* Project survival depends on one person only.
* This can cause psychological problems as the “chief programmer” is like the “king” who takes all the credit and other members are resentful.
* Team organization is limited to only a small team and a small team cannot handle every project.
* The effectiveness of the team is very sensitive to the Chief programmer’s technical and managerial activities.

### ****Matrix Team Organization****

In a [matrix team organization](https://www.geeksforgeeks.org/matrix-organizational/), team members are grouped based on their specialties, such as developers, testers, or designers. Each specialist group is led by a[Project manager](https://www.geeksforgeeks.org/software-engineering-role-and-responsibilities-of-a-software-project-manager/) who oversees their work. This structure allows team members to focus on their areas of expertise while managers coordinate tasks and resources across different projects.

### ****Egoless Team Organization****

*Egoless programming is a state of mind in which programmer is supposed to separate themselves from their product. In this team organization goals are set and decisions are made by group consensus. Here group, ‘leadership’ rotates based on tasks to be performed and differing abilities of members.*

In this organization work products are discussed openly and freely examined by all team members. There is a major risk which such organization if teams are composed of inexperienced or incompetent 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 cannot be reached.

#### **Advantages of Democratic Team Organization**

* Each member can contribute to decisions.
* Members can learn from each other.
* Improved job satisfaction.

#### **Disadvantages of Democratic Team Organization**

* Communication overhead increased.
* Need for compatibility of members.
* Less individual responsibility and authority.

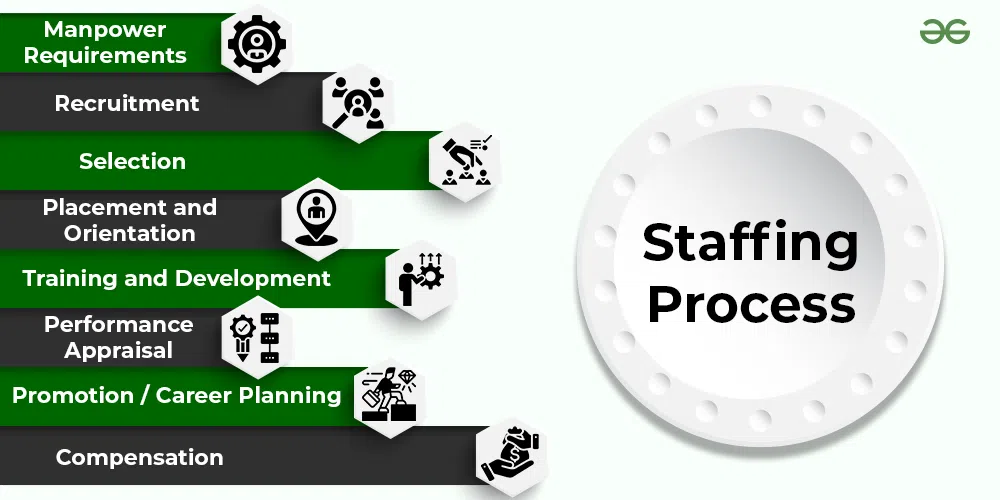
## Conclusion

Each Structure of Software Project Team Organization i.e. hierarchical, chief-programmer, matrix, egoless, and democratic has is own strengths and weaknesses. Hierarchical teams provide clear communication paths, while chief-programmer teams benefit from centralized decision-making. Matrix teams use specialized skills, egoless teams promote collaboration, and democratic teams ensure broad participation.

# Staffing Process

Staffing is the art of acquiring, developing, and maintaining a satisfactory and satisfied workforce. Staffing is that function by which a manager builds an organization through the recruitment, selection, and development of the individual, which also includes a series of activities. It ensures that the organization has the right number of people at the right places, at the right time, and performing the right thing.

As we know, the prime concern of the staffing function in the management process is in the fulfilment of the manpower requirements within an organization. These requirements may arise in the case of starting a new enterprise or expanding the existing one. It may also arise as the need for replacing those who quit, retire, transfer, or are promoted from the job. In any case, the need for ‘the right person for the right job, at the right time’ needs an emphasis.



## Staffing Process

The process of staffing consists of several interrelated activities, such as planning for human resources requirements, recruitment, selection, training development, remuneration, and so on. These activities together make the staffing process. Therefore, these are called elements or steps of the staffing process.

## ****1. Manpower Planning****

Human resource management is a process of determining the number and type of personnel required for filling the vacant job in an organization. Manpower requirements involve two kinds of analysis, i.e., workload analysis and workforce analysis. Workload analysis involves determining the number and type of employees required to perform various jobs and achieve organizational objectives. Workforce analysis shows the number and type of human resources available with an organization.  
The difference between workload and workforce is calculated to determine shortage and surplus of manpower. Excess workload indicates understaffing, i.e., the need of appointing more people and excess workforce indicates overstaffing, i.e., need to remove or transfer some employees to other places.

## ****2. Recruitment****

After estimating manpower requirements, the second step in the process of staffing is recruitment. Recruitment refers to a process of searching for prospective employees and encouraging them to apply for jobs in the organization. It involves identifying various resources of human force and attracting them to apply for the job. The main purpose of a requirement is to create a pool of applicants by a large number of qualified candidates. Recruitment can be done by both internal and external sources of recruitment. Internal sources may be used to a limited extent, and to get fresh talent and a wider choice, external sources can be used.

## ****3. Selection****

Selection is the process of choosing and appointing the right candidates for various job positions in the organization. It is treated as a negative process because it involves the rejection of some candidates. There are many steps involved in the process of employee selection. These steps include preliminary screening, filling-in application, written test, interviews, medical examination, checking references, and issuing a letter of appointment to the candidates. The most suitable candidates who meet the requirement of the vacant job are selected. The process of selection serves two important purposes, firstly, it ensures that the organization gets the best among the available candidates, and secondly, it boosts ups the self-esteem and prestige of the candidates.

## ****4. Placement and Orientation****

After selection, an appropriate job is assigned to each selected person. Placement is the process of matching the candidates with the jobs in the organization. Under this process, every selected candidate is assigned a job most suitable for him. The purpose of placement is to fit the right person to the right job so that the efficiency of work is high and the employees get personal satisfaction. Correct placement helps to reduce labour turnover and absenteeism. Here, orientation means introducing new employees to the organization. It is the process of introducing and familiarizing newly appointed candidates with their job, work groups and the organization so that they may feel at home in the new environment.

## ****5. Training and Development****

People are in search of careers and not jobs. Every individual must be given a chance to rise to the top. The most favourable way for this to happen is to promote employee learning. For this, organizations either provide training themselves within the organization or through external institutions. This is beneficial for the organization as well. If the employees are motivated enough, it will increase their competence and will be able to perform even better for the organization with greater efficiency and productivity. By providing such opportunities to its employees for career advancement, the organization captivates the interest and holds on of its talented employees. The majority of the organization has a distinct department for this purpose, that is, the Human Resource Department. Though in small organizations, the line manager has to do all the managerial functions viz, planning, organizing, staffing, controlling, and directing. The process of staffing further involves three more stages.

## ****6. Performance appraisal****

After training the employees and having them on the job for some time, there should be an evaluation done on their performance. Every organization has its means of appraisal whether formal or informal. Appraisal refers to the evaluation of the employees of the organization based on their past or present performance by some pre-decided standards. The employee should be well aware of his standards and his superior is responsible for proving feedback on his performance. The process of performance appraisal, thus includes specifying the job, performing appraisal performance, and providing feedback.

## ****7. Promotion and Career planning****

It has now become important for all organizations to deal with career-related issues and promotional routes for employees. The managers should take care of the activities that serve the long-term interests of the employees. They should be encouraged from time to time, which will help the employees to grow and find their true potential. Promotions are an essential part of any employee’s career. Promotion refers to the transferring of employees from their current positions to a higher level increasing their responsibilities, authority and pay.

## ****8. Compensation****

Every organization needs to set up plans for the salary and wages of the employees. There are several ways to develop payment plans for the employees depending upon the significance of the job. The worth of the job needs to be decided. Therefore, all kinds of payments or rewards provided to the employees is referred to as compensation. The compensation may be in the form of direct financial payments, such as salary, wages, bonuses, etc., or indirect payments like insurance or vacations provided to the employee.

Direct financial payments are of two kinds, that is, performance-based and time-based. In a time-based payment plan, the salary or wages are paid daily, weekly, monthly, or annually, whereas, the performance-based payment plan is the payment of salary or wages according to the set task. There are many ways in which the compensation of the employee based on their performance can be calculated. There are also plans, which are a combination of both time-based and performance-based. There are a few factors that affect the payment plan, such as legal, company policy, union, and equity. Thus, staffing is the process that includes possession, retention, promotion, and compensation of the human capital, that is, the most important resource of the organization. There are several factors such as the supply and demand of specific skills in the labour market, legal and political considerations, the company’s image, policy, unemployment rate, human resource planning cost, labour market conditions, technological developments, general economic environment, etc., that may affect the execution of recruitment, selection, and training.

## Aspects or Components of Staffing

There are three aspects or components of staffing, namely, recruitment, selection, and training. They are defined below:

* **Recruitment:**It is the process of finding potential candidates for a particular job in an organization. The process of recruitment involves persuading people to apply for the available positions in the organization.
* **Selection:**It is the process of recognizing potential and hiring the best people out of several possible candidates. This is done by shortlisting and choosing the deserving and eliminating those who are not suitable for the job.
* **Training:** It is the process that involves providing the employees with an idea of the type of work they are supposed to do and how it is to be done. It is a way of keeping the employees updated on the way of work in an organization and the new and advanced technologies.

**Risk management**

A risk is a probable problem; it might happen, or it might not. There are main two characteristics of risk.

* **Uncertainty:** the risk may or may not happen which means there are no 100% risks.
* **Loss:** If the risk occurs in reality, undesirable results or losses will occur.

In this Article we will understand Risk Management in detail.

## What is Risk Management?

Risk Management is a systematic process of recognizing, evaluating, and handling threats or risks that have an effect on the finances, capital, and overall operations of an organization. These risks can come from different areas, such as financial instability, legal issues, errors in strategic planning, accidents, and natural disasters.

*The main goal of risk management is to predict possible risks and find solutions to deal with them successfully.*

## Why is risk management important?

Risk management is important because it helps organizations to prepare for unexpected circumstances that can vary from small issues to major crises. By actively understanding, evaluating, and planning for potential risks, organizations can protect their financial health, continued operation, and overall survival.

Let’s Understand why risk management important with an example.

Suppose In a software development project, one of the key developers unexpectedly falls ill and is unable to contribute to the product for an extended period.

One of the solution that organization may have , The team uses collaborative tools and procedures, such as shared work boards or project management software, to make sure that each member of the team is aware of all tasks and responsibilities, including those of their teammates.

An organization must focus on providing resources to minimize the negative effects of possible events and maximize positive results in order to reduce risk effectively. Organizations can more effectively identify, assess, and mitigate major risks by implementing a consistent, systematic, and integrated approach to risk management.

## The risk management process

Risk management is a sequence of steps that help a software team to understand, analyze, and manage uncertainty. Risk management process consists of

* Risks Identification.
* Risk Assessment.
* Risks Planning.
* Risk Monitoring



*Risk Management Process*

**Risk Identification**

Risk identification refers to the systematic process of recognizing and evaluating potential threats or hazards that could negatively impact an organization, its operations, or its workforce. This involves identifying various types of risks, ranging from IT security threats like viruses and phishing attacks to unforeseen events such as equipment failures and extreme weather conditions.

**Risk analysis**

Risk analysis is the process of evaluating and understanding the potential impact and likelihood of identified risks on an organization. It helps determine how serious a risk is and how to best manage or mitigate it. Risk Analysis involves evaluating each risk’s probability and potential consequences to prioritize and manage them effectively.

**Risk Planning**

Risk planning involves developing strategies and actions to manage and mitigate identified risks effectively. It outlines how to respond to potential risks, including prevention, mitigation, and contingency measures, to protect the organization’s objectives and assets.

**Risk Monitoring**

Risk monitoring involves continuously tracking and overseeing identified risks to assess their status, changes, and effectiveness of mitigation strategies. It ensures that risks are regularly reviewed and managed to maintain alignment with organizational objectives and adapt to new developments or challenges.

## ****Understanding Risks in Software Projects****

A computer code project may be laid low with an outsized sort of risk. To be ready to consistently establish the necessary risks that could affect a computer code project, it’s necessary to group risks into completely different categories. The project manager will then examine the risks from every category square measure relevant to the project.

There are mainly 3 classes of risks that may affect a computer code project:

1. **Project Risks:**   
   Project risks concern various sorts of monetary funds, schedules, personnel, resources, and customer-related issues. A vital project risk is schedule slippage. Since computer code is intangible, it’s tough to observe and manage a computer code project. It’s tough to manage one thing that can not be seen. For any producing project, like producing cars, the project manager will see the merchandise taking form.

For example, see that the engine is fitted, at the moment the area of the door unit is fitted, the automotive is being painted, etc. so he will simply assess the progress of the work and manage it. The physical property of the merchandise being developed is a vital reason why several computer codes come to suffer from the danger of schedule slippage.

1. **Technical Risks:**   
   Technical risks concern potential style, implementation, interfacing, testing, and maintenance issues. Technical risks conjointly embody ambiguous specifications, incomplete specifications, dynamic specifications, technical uncertainty, and technical degeneration. Most technical risks occur thanks to the event team’s lean information concerning the project.
2. **Business Risks:**   
   This type of risk embodies the risks of building a superb product that nobody needs, losing monetary funds or personal commitments, etc.

## ****Classification of Risk in a project****

**Example:**Let us consider a satellite-based mobile communication project. The project manager can identify many risks in this project. Let us classify them appropriately.

* What if the project cost escalates and overshoots what was estimated? – **Project Risk**
* What if the mobile phones that are developed become too bulky to conveniently carry? **Business Risk**
* What if call hand-off between satellites becomes too difficult to implement? **Technical Risk**

## Risk management standards and frameworks

Risk management standards and frameworks give organizations guidelines on how to find, evaluate, and handle risks effectively. They provide a structured way to manage risks, making sure that everyone follows consistent and reliable practices. Here are some well-known risk management standards and frameworks:

**1. COSO ERM Framework:**

**COSO ERM Framework was introduce in 2004 and updated in 2017. Its main purpose is to a**ddresses the growing complexity of Enterprise Risk Management (ERM).

* **Key Features**:
  + 20 principles grouped into five components: Governance and culture, Strategy and objective-setting, Performance, Review and revision, Information, communication, and reporting.
  + It promote integrating risk into business strategies and operations.

2. **ISO 31000**:

ISO 31000 was introduce in 2009, revised in 2018. It provides principles and a framework for ERM.

* **Key Features**:
  + It offers guidance on applying risk management to operations.
  + It focuses on identifying, evaluating, and mitigating risks.
  + It promote senior management’s role and integrating risk management across the organization.

3. **BS 31100**:

## Benefits of risk management

Here are some benefits of risk management:

* Helps protect against potential losses.
* Improves decision-making by considering risks.
* Reduces unexpected expenses.
* Ensures adherence to laws and regulations.
* Builds resilience against unexpected challenges.
* Safeguards company reputation.

## Limitation of Risk Management

Here are Some Limitation of Risk Management

* Too much focus on risk can lead to missed opportunities.
* Implementing risk management can be expensive.
* Risk models can be overly complex and hard to understand.
* Having risk controls might make people feel too safe.
* Relies on accurate human judgment and can be prone to mistakes.
* Some risks are hard to predict or quantify.
* Managing risks can take a lot of time and resources.

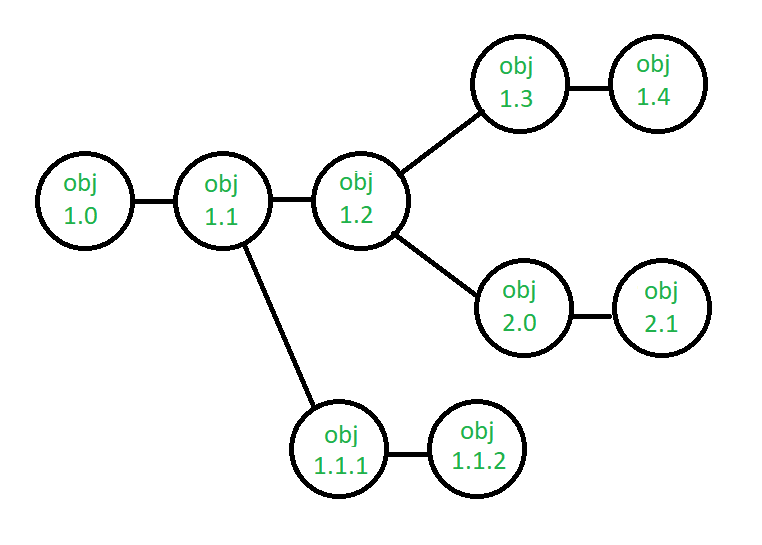
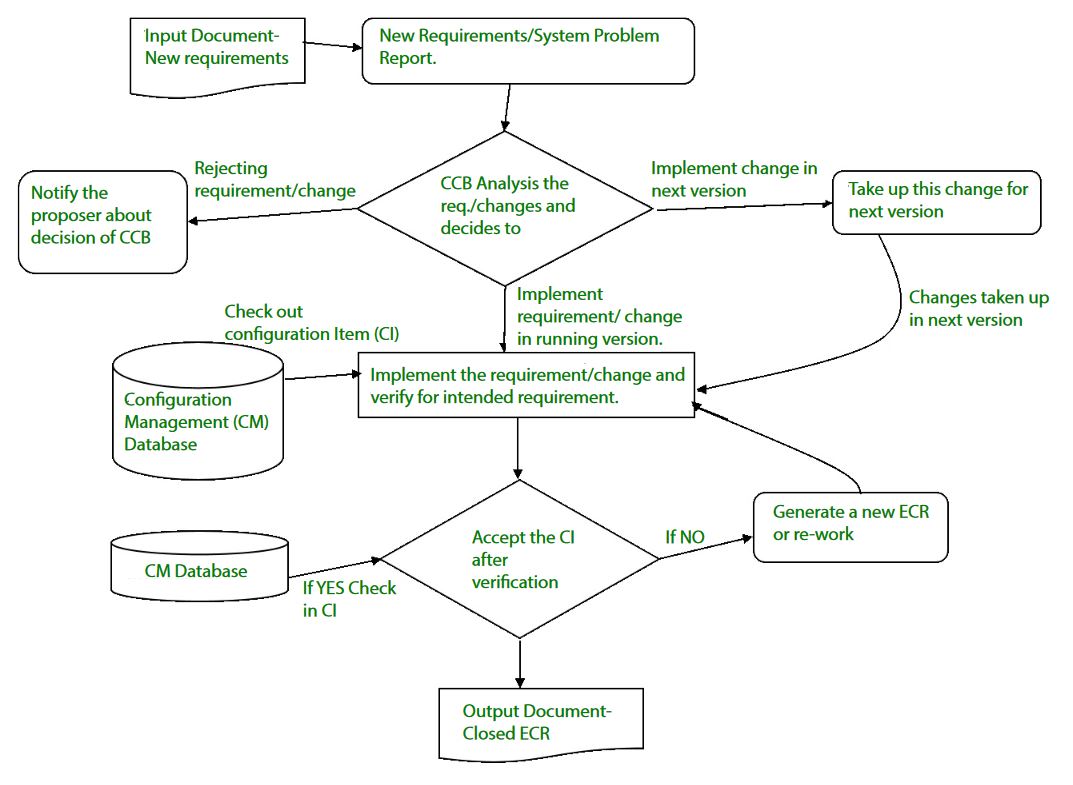
## Conclusion

Risk management is important for protecting businesses from potential problems and helping them make better decisions. While it has many advantages, like saving money and keeping things running smoothly, it also has challenges, such as cost and complexity. Overall, using risk management wisely can help businesses succeed by preparing for the unexpected and making the most of opportunities.

**Software Configuration Management**

**System Configuration Management (SCM)** is an arrangement of exercises that controls change by recognizing the items for change, setting up connections between those things, making/characterizing instruments for overseeing diverse variants, controlling the changes being executed in the current framework, inspecting and revealing/reporting on the changes made. It is essential to control the changes because if the changes are not checked legitimately then they may wind up undermining a well-run programming. In this way, SCM is a fundamental piece of all project management activities.

**Processes involved in SCM –** Configuration management provides a disciplined environment for smooth control of work products. It involves the following activities:

1. **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 relationships among items, creating a mechanism to manage multiple levels of control and procedure for the change management system.
2. **Version control –** Creating versions/specifications of the existing product to build new products with the help of the SCM system. A description of the version is givenbelow:Suppose after some changes, the version of the configuration object changes from 1.0 to 1.1. Minor corrections and changes result in versions 1.1.1 and 1.1.2, which is followed by a major update that is object 1.2. The development of object 1.0 continues through 1.3 and 1.4, but finally, a noteworthy change to the object results in a new evolutionary path, version 2.0. Both versions are currently supported.
3. **Change control –** Controlling changes to Configuration items (CI). The change control process is explained in Figure below:A change request (CR) is submitted and evaluated to assess technical merit, potential side effects, the overall impact on other configuration objects and system functions, and the projected cost of the change. The results of the evaluation are presented as a change report, which is used by a change control board (CCB) —a person or group who makes a final decision on the status and priority of the change. An engineering change Request (ECR) is generated for each approved change. Also, CCB notifies the developer in case the change is rejected with proper reason.

**4.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. The audit confirms the completeness, correctness, and consistency of items in the SCM system and tracks action items from the audit to closure.

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

System Configuration Management (SCM) is a software engineering practice that focuses on managing the configuration of software systems and ensuring that software components are properly controlled, tracked, and stored. It is a critical aspect of [software development](https://www.geeksforgeeks.org/software-development/?ref=lbp), as it helps to ensure that changes made to a software system are properly coordinated and that the system is always in a known and stable state.

SCM involves a set of processes and tools that help to manage the different components of a software system, including source code, documentation, and other assets. It enables teams to track changes made to the software system, identify when and why changes were made, and manage the integration of these changes into the final product.

## Importance of Software Configuration Management

1. Effective Bug Tracking: Linking code modifications to issues that have been reported, makes bug tracking more effective.
2. Continuous Deployment and Integration: SCM combines with continuous processes to automate deployment and testing, resulting in more dependable and timely software delivery.
3. Risk management: SCM lowers the chance of introducing critical flaws by assisting in the early detection and correction of problems.
4. Support for Big Projects: Source Code Control (SCM) offers an orderly method to handle code modifications for big projects, fostering a well-organized development process.
5. Reproducibility: By recording precise versions of code, libraries, and dependencies, source code versioning (SCM) makes builds repeatable.
6. Parallel Development: SCM facilitates parallel development by enabling several developers to collaborate on various branches at once.

## Why need for System configuration management?

1. **Replicability:** Software version control (SCM) makes ensures that a software system can be replicated at any stage of its development. This is necessary for testing, debugging, and upholding consistent environments in production, testing, and development.
2. **Identification of Configuration:** Source code, documentation, and executable files are examples of configuration elements that SCM helps in locating and labeling. The management of a system’s constituent parts and their interactions depend on this identification.
3. **Effective Process of Development:** By automating monotonous processes like managing dependencies, merging changes, and resolving disputes, SCM simplifies the development process. Error risk is decreased and efficiency is increased because of this automation.

## Key objectives of SCM

1. **Control the evolution of software systems:**SCM helps to ensure that changes to a software system are properly planned, tested, and integrated into the final product.
2. **Enable collaboration and coordination:**SCM helps teams to collaborate and coordinate their work, ensuring that changes are properly integrated and that everyone is working from the same version of the software system.
3. **Provide version control:** SCM provides version control for software systems, enabling teams to manage and track different versions of the system and to revert to earlier versions if necessary.
4. **Facilitate replication and distribution:** SCM helps to ensure that software systems can be easily replicated and distributed to other environments, such as test, production, and customer sites.
5. SCM is a critical component of [software development](https://www.geeksforgeeks.org/what-is-software-development/?ref=lbp), and effective SCM practices can help to improve the quality and reliability of software systems, as well as increase efficiency and reduce the risk of errors.

## The main advantages of SCM

1. Improved productivity and efficiency by reducing the time and effort required to manage software changes.
2. Reduced risk of errors and defects by ensuring that all changes were properly tested and validated.
3. Increased collaboration and communication among team members by providing a central repository for software artifacts.
4. Improved quality and stability of software systems by ensuring that all changes are properly controlled and managed.

## The main disadvantages of SCM

1. Increased complexity and overhead, particularly in large software systems.
2. Difficulty in managing dependencies and ensuring that all changes are properly integrated.
3. Potential for conflicts and delays, particularly in large development teams with multiple contributors.

**Miscellaneous Plan.**

**Software Project Management Plan (SPMP) – Software Engineering**

Once project designing is complete, project managers document their plans during a software package Project Management setup (SPMP) document. The SPMP document ought to discuss an inventory of various things that are mentioned below. This list will be used as a doable organization of the SPMP document. Organization of the software package Project Management set up (SPMP) document.

**Introduction:**

* Objectives
* Major Functions
* Performance Issues
* Management and Technical Constraints

**Project Estimates:**

* Historical Data Used
* Estimation Techniques Used
* Effort, Resource, Cost, and Project Duration Estimates

**Schedule:**

* Work Breakdown Structure
* Task Network Representation
* Gantt Chart Representation
* PERT Chart Representation

**Project Resources:**

* People
* Hardware and Software
* Special Resources

**Staff Organization:**

* Team Structure
* Management Reporting

**Risk Management Plan:**

* Risk Analysis
* Risk Identification
* Risk Estimation
* Risk Abatement Procedures

**Project Tracking and Control Plan:**

**Miscellaneous Plans:**

* Process Tailoring
* Quality Assurance Plan
* Configuration Management Plan
* Validation and Verification
* System Testing Plan
* Delivery, Installation, and Maintenance Plan