Chapter 3: Managing Software Project

Project Management Concepts

- Involves the planning, monitoring, and control of the people, process and events that occur as software evolves from a preliminary concept of full operational deployment.
- Aimed to ensure that the software is delivered on time, within budget and schedule constraints, and satisfies the requirements of the client.

Key Concept:

- **People** - the most important element of a successful project
- **Product** - the software to be built
- **Process** -the set of framework activities and software engineering tasks to get the job done
- **Project** - all work required to make the product a reality

1. People:

Stakeholders:

❖ Senior Managers, Project Managers, Practitioners, Customers, End users

Team leaders:

❖ Motivation, Organization, Ideas or innovation

❖ Project solving, Managerial Identity, Achievement, Influence and Team building.

Software team

Software Agile Team

Communication and Coordination

1) Software Metrics (Process, Product and Project Metrics)

Software metric is a measure of software characteristics which are measurable or countable. Software metrics are valuable for many reasons, including measuring software performance, planning work items, measuring productivity, and many other uses.

Within the software development process, many metrics are all connected. Software metrics are similar to the four functions of management: Planning, Organization, Control, or Improvement.

- **Direct Measures:** Immediately measurable Eg: Cost, effort, LOC, speed, memory
- **Indirect Measures:** Not immediately quantifiable. Eg: Functionality, quality, complexity, efficiency, reliability, maintainability

Faults:

- **Errors:** Faults found by the practitioners during software development
- **Defects:** Faults found by the customers after release

Classification of Software Metrics

Processes– Activities related to production of software

Products- Explicit results of software development activities.

-Deliverables, documentation, by products

Project

-Inputs into the software development activities

-hardware, knowledge, people

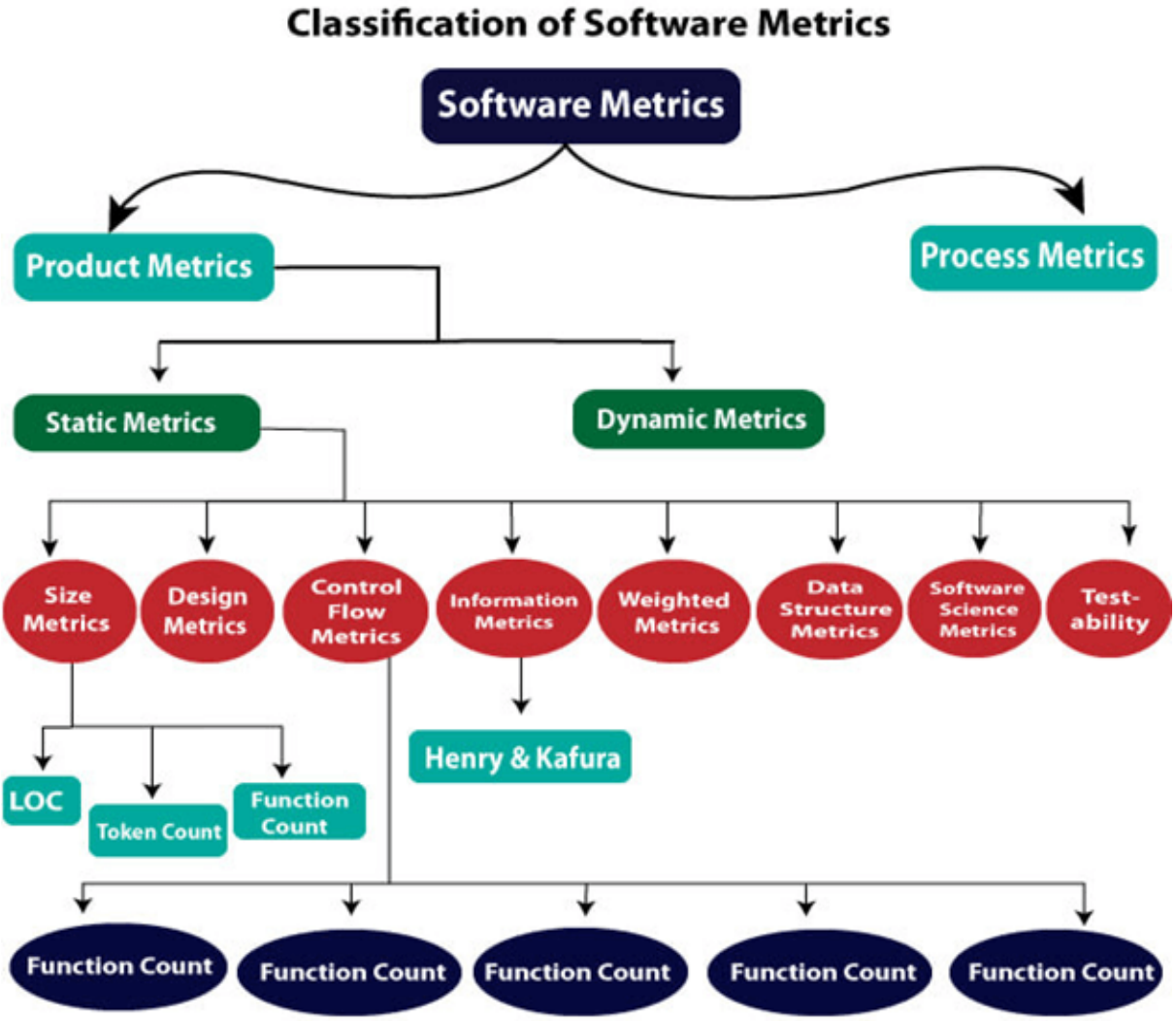
Software metrics can be classified into two types as follows:

Product Metrics: These are the measures of various characteristics of the software product. The two important software characteristics are:

1. Size and complexity of software.
2. Quality and reliability of software.

These metrics can be computed for different stages of SDLC.

Process Metrics: These are the measures of various characteristics of the software development process. For example, the efficiency of fault detection. They are used to measure the characteristics of methods, techniques, and tools that are used for developing software.



Types of Metrics

Internal metrics: Internal metrics are the metrics used for measuring properties that are viewed to be of greater importance to a software developer. **For example, Lines of Code (LOC) measure.**

External metrics: External metrics are the metrics used for measuring properties that are viewed to be of greater importance to the user, e.g., portability, reliability, functionality, usability, etc.

Hybrid metrics: Hybrid metrics are the metrics that combine product, process, and resource metrics. For example, cost per FP where FP stands for Function Point Metric.

Project metrics: Project metrics are the metrics used by the project manager to check the project's progress. Data from the past projects are used to collect various metrics, like time and cost; these estimates are used as a base of new software. Note that as the project proceeds, the project manager will check its progress from time-to-time and will compare the effort, cost, and time with the original effort, cost and time. Also understand that these metrics are used to decrease the development costs, time efforts and risks. The project quality can also be improved. As quality improves, the number of errors and time, as well as cost required, is also reduced. The project matrix describes the project characteristic and execution process.

- Number of software developer
- Staffing patterns over the life cycle of software
- Cost and schedule
- Productivity

Metrics for Software Cost and Effort Estimation

1.Size Oriented Metrics(KLOC Measurement)

- Measure the size of the software that has been produced.
- Any lines in code apart from comments and blanks, includes header, declaration as well as executable & non executable statements.
- Normalized by counting in Thousand Lines of Code(KLOC).

Size oriented metrics of project computed by

- Errors per KLOC (thousand lines of code)
- Defects per KLOC
- $ per KLOC
- Pages of documentation per KLOC
- Errors per person-month
- KLOC per person-month
- $ per page of documentation

Advantages:

- Easy to count or calculate from developed count.

Disadvantages:

- Programming language dependent.
- Penalize well designed but short programs.
- Not universally accepted as a best way to
- measure the software process.

2.Function Oriented Metrics

- Use a measure of the functionality delivered by the application as a normalization value.
- The most widely used function-oriented metric is the function point (FP).
- Computation of the function point is based on characteristics of the software's information domain and complexity.

Using historical data it can:

- Estimate the cost or effort required to design, code and test the software.
- Predict the number of errors that will be encountered during testing.
- Forecast the number of components and/or the number of projected source lines in the implemented system.
- **FP = Count Total * [0.65 + 0.01 * Sum (Value Adjustment Factors)]**
- Function Point values on past projects can be used to compute,
- for example, the average number of lines of code per function point

Advantages

- FP is programming language independent
- FP is based on data that are more likely to be known in the early stages of a project, making it more attractive as an estimation approach

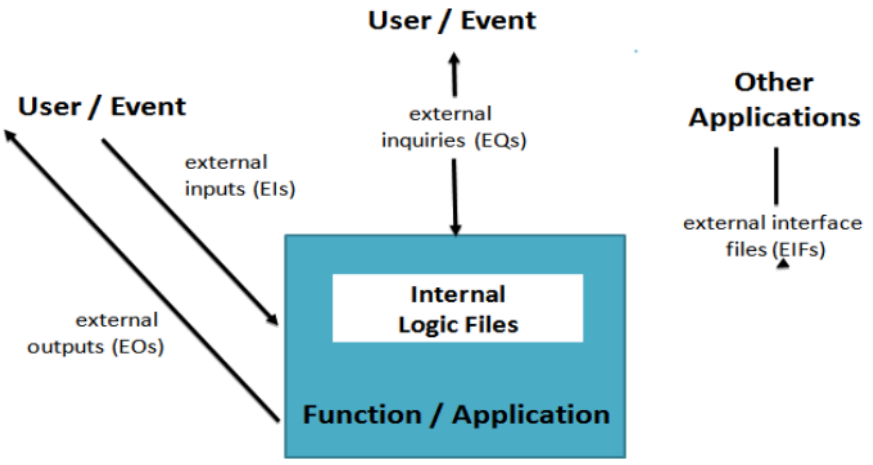
Disadvantages

- FP requires some "sleight of hand" because the computation is based on subjective data
- Counts of the information domain can be difficult to collect.
- FP has no direct physical meaning, it's just a number

Principle FP Analysis(FPA): System is decomposed into 5 functional units.

- **External inputs(EI)-** Distinct input from user External Outputs(EO)- Reports, screens, error messages, etc
- **External Enquiries(EQ)-** On line input that generates some results.
- **Internal Logic Files(ILF)-** Logical file (database)
- **External Interface Files(EIF)-** Data files/connections as interface to other systems

Function point Components:



Compute Function Points:

$$FP = Count\ Total * [\ 0.65 + 0.01 * \sum(F_i)\ ]$$

Count Total is the sum of all FP entries

Fi (i=1 to 14) are complexity value adjustment factors (**VAF**).

Value adjustment factors are used to provide an indication of problem complexity

Information Domain Value	Count		Weighting factor Simple	Average	Complex		
External Inputs (EIs)		×	3	4	6	=	
External Outputs (EOs)		×	4	5	7	=	
External Inquiries (EQs)		×	3	4	6	=	
Internal Logical Files (ILFs)		×	7	10	15	=	
External Interface Files (EIFs)		×	5	7	10	=	
Count total	→						

Value Adjustment Factors:

- F1. Data Communication
- F2. Distributed Data Processing
- F3. Performance
- F4. Heavily Used Configuration
- F5. Transaction Role
- F6. Online Data Entry
- F7. End-User Efficiency
- F8. Online Update
- F9. Complex Processing
- F10. Reusability
- F11. Installation Ease
- F12. Operational Ease
- F13. Multiple Sites
- F14. Facilitate Change

Function Point Calculation Example:

Information Domain Value	Count		Weighting factor: Simple	Average	Complex		
External Inputs (EIs)	3	×	3	4	6	=	9
External Outputs (EOs)	2	×	4	5	7	=	8
External Inquiries (EQs)	2	×	3	4	6	=	6
Internal Logical Files (ILFs)	1	×	7	10	15	=	7
External Interface Files (EIFs)	4	×	5	7	10	=	20
Count total							50

Used Adjustment Factors and assumed values are,

F09. Complex internal processing = **3**

F10. Code to be reusable = **2**

F13. Multiple sites = **3**

F03. High performance = **4**

F02. Distributed processing = **5**

Project Adjustment Factor (VAF) = 17

$$FP = \text{Count Total} * [\ 0.65 + 0.01 * \sum(F_i)\ ]$$

$$FP = [50]* [0.65 + 0.01 * 17]$$

$$FP = [50]* [0.65 + 0.17]$$

$$FP = [50]* [0.82] = 41$$

Example 2:

Study of requirement specification for a project has produced the following results: 7 inputs, 10 outputs, 6 inquiries, 17 files and 4 external interfaces.

- Input and external interface function point attributes are of average complexity and all other function points attributes are of low complexity.
- Determine adjusted function points assuming complexity adjustment value is 32.

Information Domain Value	Count		Weighting factor Simple	Average	Complex		
External Inputs (EIs)		×	3	4	6	=	
External Outputs (EOs)		×	4	5	7	=	
External Inquiries (EQs)		×	3	4	6	=	
Internal Logical Files (ILFs)		×	7	10	15	=	
External Interface Files (EIFs)		×	5	7	10	=	
Count total	→						

Value adjustment factors (VAF) = 32 given

$$FP = \text{Count Total} * [\ 0.65 + 0.01 * \sum(F_i)\ ]$$

$$= 233 * [\ 0.65 + 0.01 * 32]$$

$$= 233 * \ 0.97 = 226.01$$

3.Object-Oriented Metrics

- Conventional software project metrics (LOC or FP) can be used to estimate object-oriented software projects
- ▪ However, these metrics do not provide enough granularity (detailing) for the schedule and effort adjustments that are required as you iterate through an evolutionary or incremental process

Lorenz and Kidd suggest the following set of metrics for OO projects

- Number of scenario scripts: Detailed sequence of steps that describe the interaction between user and the application.
- Number of key classes (the highly independent components)
- Number of support classes: Developed for each of the key classes.
- Average number of support classes per key class.
- Number of subsystems. Subsystem is an aggregation of classes that support a function that is visible to the end user of a system.

4.Use Case Oriented Metrics

- Use cases describe user-visible functions and features that are basic requirements for a system.
- Like FP, the use case is defined early in the software process, allowing it to be used for estimation before significant modeling and construction activities are initiated.

- The use case is independent of programming language, because use cases can be created at vastly different levels of abstraction, there is no standard “size” for a use case
- Without a standard measure of what a use case is, its application as a normalization measure is suspect (doubtful).
- · Ex., effort expended / use case
- No. of use cases is directly proportional to the Size of the application in LOC .
- No. of test cases that will be designed to fully exercise the application.

5.Webapp Project Metrics

Deliver a combination of content and functionality to the end user.

- Number of static web pages
- Number of Dynamic web pages
- Number of internal page links.
- Number of persistent data objects.
- Number of external systems interfaced.
- Number of static content objects.
- Number of dynamic content objects.
- Number of executable functions.
- Customization index C=(Ndp/Ndp+Nsp)

Advantage of Software Metrics

- Comparative study of various design methodology of software systems.
- For analysis, comparison, and critical study of different programming languages concerning their characteristics.
- In comparing and evaluating the capabilities and productivity of people involved in software development.
- In the preparation of software quality specifications.
- In the verification of compliance of software systems requirements and specifications.
- In making inference about the effort to be put in the design and development of the software systems.
- In getting an idea about the complexity of the code.
- In taking decisions regarding further division of a complex module is to be done or not.
- In guiding resource managers for their proper utilization.
- In comparison and making design tradeoffs between software development and maintenance cost.
- In providing feedback to software managers about the progress and quality during various phases of the software development life cycle.
- In the allocation of testing resources for testing the code.

Disadvantage of Software Metrics

- The application of software metrics is not always easy, and in some cases, it is difficult and costly.
- The verification and justification of software metrics are based on historical/empirical data whose validity is difficult to verify.
- These are useful for managing software products but not for evaluating the performance of the technical staff.

- The definition and derivation of Software metrics are usually based on assuming which are not standardized and may depend upon tools available and working environment.
- Most of the predictive models rely on estimates of certain variables which are often not known precisely.

Decomposition Techniques

1. Software Sizing
2. Problem based Estimation
    a. LOC (Lines of Code) based
    b. FP (Function Point) based
3. Process based Estimation
4. Estimation with Use-cases

LOC (Lines of Code) based (Lines of code) :

A line of code (LOC) is any line of text in a code that is not a comment or blank line, and also header lines, in any case of the number of statements or fragments of statements on the line. LOC clearly consists of all lines containing the declaration of any variable, and executable and non-executable statements. As Lines of Code (LOC) only counts the volume of code, you can only use it to compare or estimate projects that use the same language and are coded using the same coding standards.

Source LOC are software metrics that are used to measure size of a software program by just counting a line of code inthe source code of the program.

Features :

- Variations such as "source lines of code", are used to set out a codebase.
- LOC is frequently used in some kinds of arguments.
- They are used in assessing a project's performance or efficiency.

Advantages :

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

Disadvantages :

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

Types of SLOC Measures:

1. **Physical LOC :** It is text of the physical code like comment lines, declarations and blank lines.
2. **Logical LOC:** LLOC is only the executable "statement" and comment lines, declarations and blank lines aren't counted in this purpose.
3. **Comment LOC:** consists only of comments in the source code. Code written without comment counts as poor code. Good codes have appropriate comments.
4. **Reused LOC:** reused in the particular code. People encourage writing and using reusable code which saves a lot of effort during the development process.

Problem-Based Estimation

Start with a bounded statement of scope. Decompose the software into problem functions that can each be estimated individually. Compute an LOC or FP value for each function. Derive cost or effort estimates by applying the LOC or FP values to your baseline productivity metrics. Ex., LOC/person-month or FP/person-month.

Combine function estimates to produce an overall estimate for the entire project. In general, the LOC/pm and FP/pm metrics should be computed by project domain. Important factors are team size, application area and complexity.

Process Based Estimation

Process-based estimation is obtained from "process framework" This is one of the most commonly used techniques.

- Identify the set of functions that the software needs to perform as obtained from the project scope.
- Identify the series of framework activities that need to be performed for each function.
- Estimate the effort (in person months) that will be required to accomplish each software process activity for each function.
- Apply average labor rates (i.e., cost/unit effort) to the effort estimated for each process activity.
- Compute the total cost and effort for each function and each framework activity.
- Compare the resulting values to those obtained by way of the LOC and FP estimates.
- If both sets of estimates agree, then your numbers are highly reliable. Otherwise, conduct further investigation and analysis concerning the function and activity breakdown.

Estimation with Use Cases

Developing an estimation approach with use cases is problematic for the following reasons:

- Use cases are described using many different formats and styles—there is no standard form.
- Use cases represent an external view (the user's view) of the software and can therefore be written at many different levels of abstraction.
- Use cases do not address the complexity of the functions and features that are described.
- Use cases can describe complex behaviour (Ex., interactions) that involve many functions and features.

- Although a number of investigators have considered use cases as an estimation input.
- Before use cases can be used for estimation,
- The level within the structural hierarchy is established.
- The average length (in pages) of each use case is determined.
- The type of software (e.g., real-time, business, engineering/scientific, WebApp, embedded) is defined.
- A rough architecture for the system is considered.
- Once these characteristics are established empirical data may be used to establish the estimated number of LOC or FP per use case (for each level of the hierarchy).
- Historical data is then used to compute the effort required to develop the system.

2) Software Project Estimates

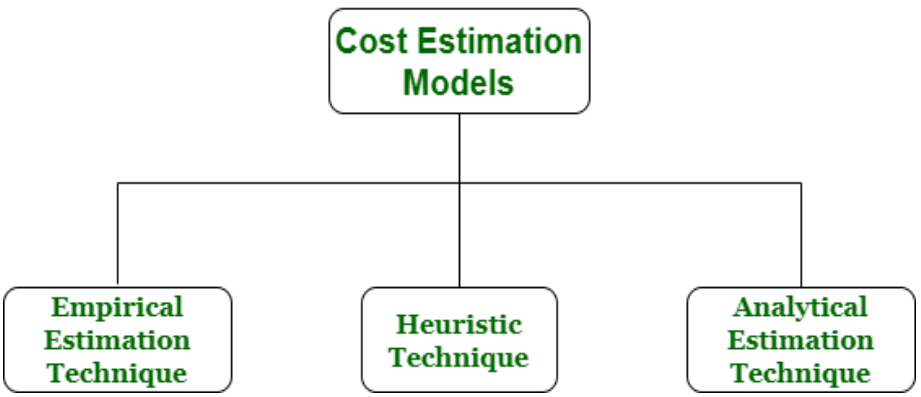
For any new software project, it is necessary to know how much it will cost to develop and how much development time it will take. These estimates are needed before development is initiated, but how is this done? Several estimation procedures have been developed and have the following attributes in common.

1. Project scope must be established in advance.
2. Software metrics are used as a support from which evaluation is made.
3. The project is broken into small PCs which are estimated individually.
   To achieve true cost & schedule estimate, several options arise.
4. Delay estimation
5. Used symbol decomposition techniques to generate project cost and schedule estimates.
6. Acquire one or more automated estimation tools.

Uses of Cost Estimation

1. During the planning stage, one needs to choose how many engineers are required for the project and to develop a schedule.
2. In monitoring the project's progress, one needs to assess whether the project is progressing according to the procedure and take corrective action, if necessary.

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 the Constructive Cost Model (COCOMO). This technique is also used to increase or speed up the analysis and investment decisions.

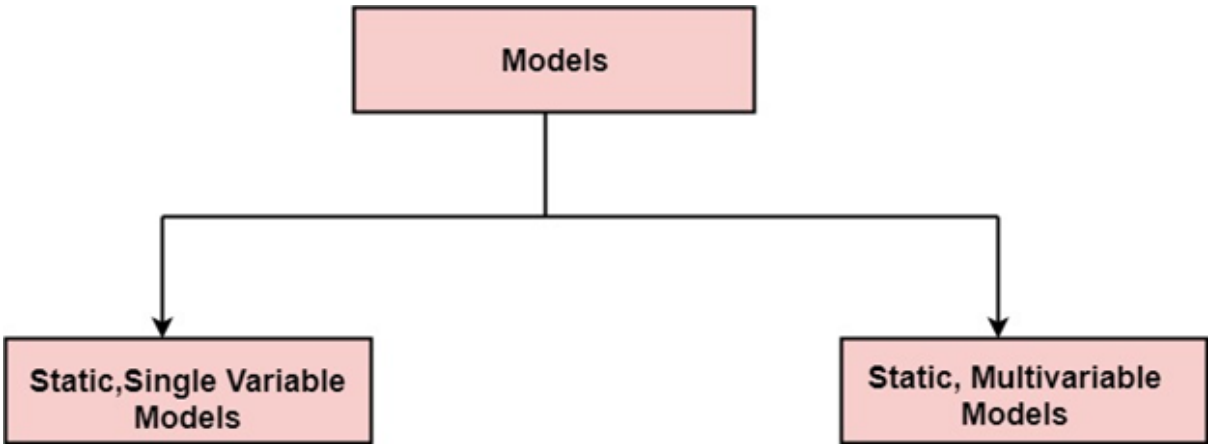
3. **Analytical Estimation Technique –**

   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 software science is based on an analytical estimation model.

A model may be static or dynamic. In a static model, a single variable is taken as a key element for calculating cost and time. In a dynamic model, all variable are interdependent, and there is no basic variable.



Static, Single Variable Models: When a model makes use of single variables to calculate desired values such as cost, time, efforts, etc. is said to be a single variable model. The most common equation is: $C=aL^b$

Where C = Costs
L= size
a and b are constants

The Software Engineering Laboratory established a model called SEL model, for estimating its software production. This model is an example of the static, single variable model.

The Software Engineering Laboratory established a model called SEL model, for estimating its software production. This model is an example of the static, single variable model.

$E=1.4L^{0.93}$
$DOC=30.4L^{0.90}$
$D=4.6L^{0.26}$

Where E= Efforts (Person Per Month)
DOC=Documentation (Number of Pages)
D = Duration (D, in months)
L = Number of Lines per code

Static, Multivariable Models: These models are based on method (1), they depend on several variables describing various aspects of the software development environment. In some models, several variables are needed to describe the software

development process, and the selected equation combines these variables to give the estimate of time & cost. These models are called multivariable models.

WALSTON and FELIX develop the models at IBM provide the following equation gives a relationship between lines of source code and effort: **$E=5.2L^{0.91}$**

The productivity index uses 29 variables which are found to be highly correlated productivity as follows:

$$I = \sum_{i=1}^{29} W_i X_i$$

Where **W_i** is the weight factor for the **ith variable** and **X_i={-1,0,+1}** the estimator gives **X_i**one of the values **-1, 0 or +1** depending on whether the variable decreases, has no effect or increases productivity.

Example: Compare the Walston-Felix Model with the SEL model on a software development expected to involve 8 person-years of effort.

- Calculate the number of lines of source code that can be produced.
- Calculate the duration of the development.
- Calculate the productivity in LOC/PY
- Calculate the average manning

Ans:The amount of manpower involved = 8PY=96persons-months

(a)Number of lines of source code can be obtained by reversing equation to give:

$$L = \left(\frac{E}{a}\right) 1/b$$

L (SEL) = (96/1.4)¹⁄₀.93=94264 LOC

L (SEL) = (96/5.2)¹⁄₀.91=24632 LOC

(b)Duration in months can be calculated by means of equation

D (SEL) = 4.6 (L) 0.26

= 4.6 (94.264)0.26 = 15 months

D (W-F) = 4.1 $L^{0.36}$

= 4.1 (24.632)0.36 = 13 months

(c) Productivity is the lines of code produced per persons/month (year)

$$P\ (SEL) = \frac{94264}{8} = 11783\ \frac{LOC}{Person} - Years$$

$$P\ (Years) = \frac{24632}{8} = 3079\ \frac{LOC}{Person} - Years$$

(d)Average manning is the average number of persons required per month in the project

$$M\ (SEL) = \frac{96P-M}{15M} = 6.4 Persons$$

$$M\ (W\text{-}F) = \frac{96P-M}{13M} = 7.4 Persons$$

COCOMO Model:

Boehm proposed COCOMO (Constructive Cost Estimation Model) in 1981.COCOMO is one of the most generally used software estimation models in the world. COCOMO predicts the efforts and schedule of a software product based on the size of the software.

The necessary steps in this model are:

1. Get an initial estimate of the development effort from evaluation of thousands of delivered lines of source code (KDLOC).
2. Determine a set of 15 multiplying factors from various attributes of the project.
3. Calculate the effort estimate by multiplying the initial estimate with all the multiplying factors i.e., multiply the values in step1 and step2.

The initial estimate (also called nominal estimate) is determined by an equation of the form used in the static single variable models, using KDLOC as the measure of the size. To determine the initial effort E_i in person-months the equation used is of the type is shown below

E_i=a*(KDLOC)b

The value of the constant a and b are depends on the project type.

In COCOMO, projects are categorized into three types:

1. Organic
2. Semi Detached
3. Embedded

1.Organic: A development project can be treated of the organic type, if the project deals with developing a well-understood application program, the size of the development team is reasonably small, and the team members are experienced in developing similar methods of projects. **Examples of this type of projects are simple business systems, simple inventory management systems, and data processing systems.**

2. Semidetached: A development project can be treated with semidetached type if the development consists of a mixture of experienced and inexperienced staff. Team members may have finite experience in related systems but may be unfamiliar with some aspects of the order being developed. **Example of Semidetached system includes developing a new operating system (OS), a Database Management System (DBMS), and complex inventory management system.**

3. Embedded: A development project is treated to be of an embedded type, if the software being developed is strongly coupled to complex hardware, or if the stringent regulations on the operational method exist. **For Example:** ATM, Air Traffic control.

For three product categories, Bohem provides a different set of expression to predict effort (in a unit of person month)and development time from the size of estimation in KLOC(Kilo Line of code) efforts estimation takes into account the productivity loss due to holidays, weekly off, coffee breaks, etc.

According to Boehm, software cost estimation should be done through three stages:

1. Basic Model
2. Intermediate Model
3. Detailed Model

1. Basic COCOMO Model: The basic COCOMO model provide an accurate size of the project parameters. The following expressions give the basic COCOMO estimation model:

Effort=a_1*(KLOC) a_2 PM

Tdev=b_1*(efforts)b_2 Months

Where

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

a_1,a_2,b_1,b_2 are constants for each group of software products,

Tdev is the estimated time to develop the software, expressed in months,

Effort is the total effort required to develop the software product, expressed in **person months (PMs)**.

Estimation of development effort

For the three classes of software products, the formulas for estimating the effort based on the code size are shown below:

Organic: Effort = 2.4(KLOC) 1.05 PM

Semi-detached: Effort = 3.0(KLOC) 1.12 PM

Embedded: Effort = 3.6(KLOC) 1.20 PM

Estimation of development time

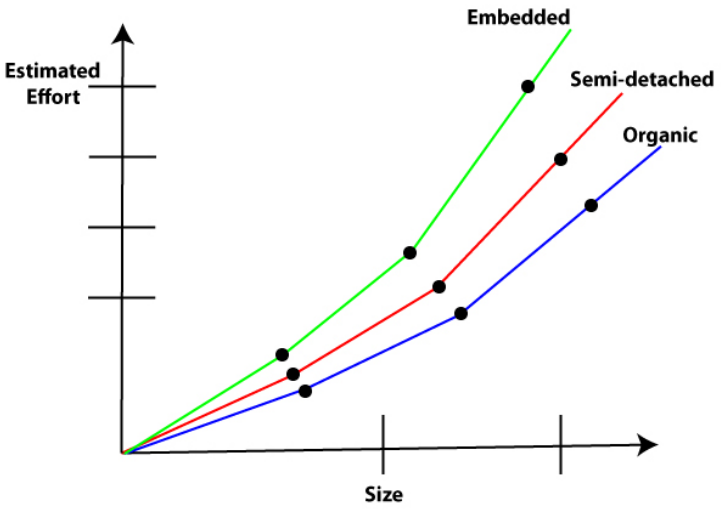
For the three classes of software products, the formulas for estimating the development time based on the effort are given below:

Organic: Tdev = 2.5(Effort) 0.38 Months

Semi-detached: Tdev = 2.5(Effort) 0.35 Months

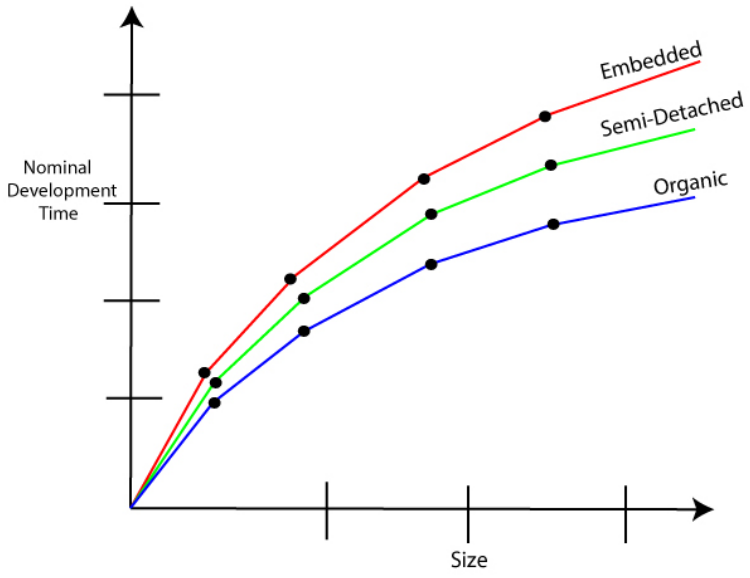
Embedded: Tdev = 2.5(Effort) 0.32 Months

Some insight into the basic COCOMO model can be obtained by plotting the estimated characteristics for different software sizes. Fig shows a plot of estimated effort versus product size. From fig, we can observe that the effort is somewhat superliner in the size of the software product. Thus, the effort required to develop a product increases very rapidly with project size.



Effort versus product size

The development time versus the product size in KLOC is plotted in fig. From fig it can be observed that the development time is a sub linear function of the size of the product, i.e. when the size of the product increases by two times, the time to develop the product does not double but rises moderately. This can be explained by the fact that for larger products, a larger number of activities which can be carried out concurrently can be identified. The parallel activities can be carried out simultaneously by the engineers. This reduces the time to complete the project. Further, from fig, it can be observed that the development time is roughly the same for all three categories of products. For example, a 60 KLOC program can be developed in approximately 18 months, regardless of whether it is of organic, semi detached, or embedded type.



Development time versus size

Example1: Suppose a project was estimated to be 400 KLOC. Calculate the effort and development time for each of the three models i.e., organic, semi-detached & embedded.

Solution: The basic COCOMO equation takes the form:

Effort=a_1*(KLOC) a_2 PM

Tdev=b_1*(efforts)b_2 Months

Estimated Size of project= 400 KLOC

(i)Organic Mode

E = 2.4 * (400)1.05 = 1295.31 PM

D = 2.5 * (1295.31)0.38=38.07 PM

(ii)Semidetached Mode

E = 3.0 * (400)1.12=2462.79 PM

D = 2.5 * (2462.79)0.35=38.45 PM

(iii) Embedded Mode

E = 3.6 * (400)1.20 = 4772.81 PM

D = 2.5 * (4772.8)0.32 = 38 PM

Example2: A project size of 200 KLOC is to be developed. Software development team has average experience on similar type of projects. The project schedule is not very tight. Calculate the Effort, development time, average staff size, and productivity of the project.

Solution: The semidetached mode is the most appropriate mode, keeping in view the size, schedule and experience of development time.

Hence E=3.0(200)1.12=1133.12PM

D=2.5(1133.12)0.35=29.3PM

$$\text{Average Staff Size (SS)} = \frac{E}{D} \text{ Persons}$$

$$= \frac{1133.12}{29.3} = 38.67 \text{ Persons}$$

$$\text{Productivity} = \frac{KLOC}{E} = \frac{200}{1133.12} = 0.1765 \text{ KLOC/PM}$$

P = 176 LOC/PM

2. Intermediate Model: The basic Cocomo model considers that the effort is only a function of the number of lines of code and some constants calculated according to the various software systems. The intermediate COCOMO model recognizes these facts and refines the initial estimates obtained through the basic COCOMO model by using a set of 15 cost drivers based on various attributes of software engineering.

2) Software Project Planning (MS Project Tool)

A Software Project is the complete methodology of programming advancement from requirement gathering to testing and support, completed by the execution procedures, in a specified period to achieve the intended software product.

Need of Software Project Management

- Software development is a sort of all new stream in world business, and there's next to no involvement in structure programming items.
- Most programming items are customized to accommodate customer's necessities.
- The most significant is that the underlying technology changes and advances so generally and rapidly that experience of one element may not be connected to the other one.
- All such business and ecological imperatives bring risk in software development; hence, it is fundamental to manage software projects efficiently.

Software project management is an art and discipline of planning and supervising software projects. It is a sub-discipline of software project management in which software projects are planned, implemented, monitored and controlled.

It is a procedure of managing, allocating and timing resources to develop computer software that fulfills requirements. There are three needs for software project management. These are:

- Time
- Cost
- Quality

It is an essential part of the software organization to deliver a quality product, keeping the cost within the client's budget and deliver the project as per schedule. There are various factors, both external and internal, which may impact this triple factor. Any of the three-factors can severely affect the other two.

Project Manager: A project manager is a character who has the overall responsibility for the planning, design, execution, monitoring, controlling and closure of a project. A project manager represents an essential role in the achievement of the projects.

A project manager is a character who is responsible for giving decisions, both large and small projects. The project manager is used to manage the risk and minimize

uncertainty. Every decision the project manager makes must directly profit their project.

Role of a Project Manager:

1. Leader: A project manager must lead his team and should provide them direction to make them understand what is expected from all of them.

2. Medium: The Project manager is a medium between his clients and his team. He must coordinate and transfer all the appropriate information from the clients to his team and report to the senior management.

3. Mentor: He should be there to guide his team at each step and make sure that the team has an attachment. He provides a recommendation to his team and points them in the right direction.

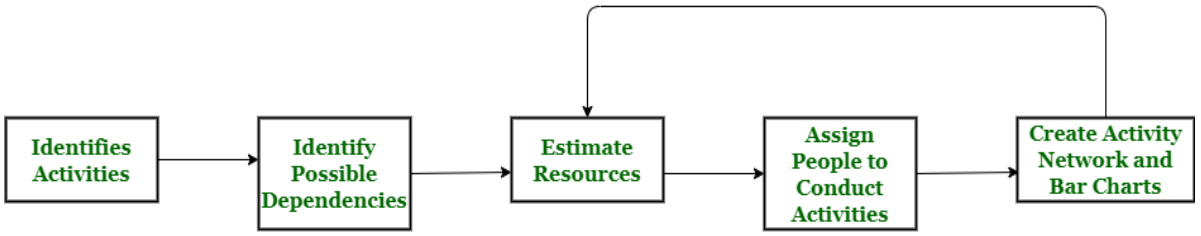
Responsibilities of a Project Manager:

- Managing risks and issues.
- Create the project team and assign tasks to several team members.
- Activity planning and sequencing.
- Monitoring and reporting progress.
- Modifies the project plan to deal with the situation.

3) Project Scheduling & Tracking

Project 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 that tasks are needed and have 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 the 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 the Gantt chart.



Project Scheduling Process

Process :

The manager needs to estimate time and resources of the project while scheduling the project. All activities in a 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 a 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 the project schedule. Then, the Project manager will decide the 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 the fact that each stage of the project is not problem-free.

Problems arise during Project Development Stage :

- People may leave or remain absent during a particular stage of development.
- Hardware may fail while performing.
- Software resources that are required may not be available at present, etc.

The project schedule is represented as a set of charts in which work-breakdown structure and dependencies within various activities are represented. To accomplish and complete a 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.

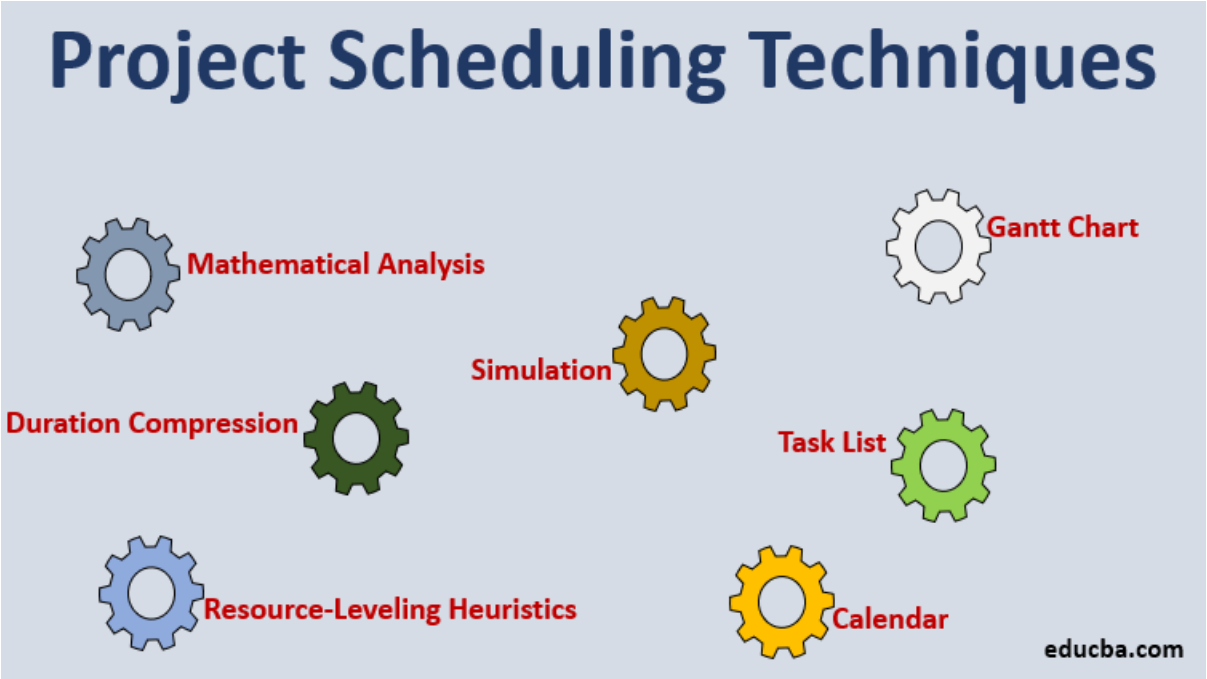
Advantages of Project Scheduling :

There are several advantages provided by project schedule in our project management:

- It simply ensures that everyone remains on the 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 processes.
- It provides effective budget management and risk mitigation.

Scheduling methods:

We need to use scheduling techniques in a project to align all its aspects so as to work corresponding to each other. A schedule should be proportionate with the time set for the project and all its resources should be used in an optimum manner. Given the variable nature of the project and its scope, it is hard to plan it, but we are expected to do it because it is we who will be held responsible.



1. Mathematical Analysis

Critical Path Method (CPM) and Program Evaluation and Review Technique (PERT) are the two most commonly used techniques by project managers. These methods are used to calculate the time span of the project through the scope of the project.

a. Critical Path Method

Every project's tree diagram has a critical path. The Critical Path Method estimates the maximum and minimum time required to complete a project. CPM also helps to identify critical tasks that should be incorporated into a project. Delivery time changes do not affect the schedule. The scope of the project and the list of activities necessary for the completion of the project are needed for using CPM. Next, the time taken by each activity is calculated. Then, all the dependent variables are identified. This helps in identifying and separating the independent variables. Finally, it adds milestones to the project.

b. Program Evaluation and Review Technique (PERT)

PERT is a way to schedule the flow of tasks in a project and estimate the total time taken to complete it. This technique helps represent how each task is dependent on the other. To schedule a project using PERT, one has to define activities, arrange them in an orderly manner and define milestones. You can calculate timelines for a project on the basis of the level of confidence:

- Optimistic timing
- Most-likely timing
- Pessimistic timing

Weighted average duration and not estimates are used by PERT to calculate different timeframes.

2. Duration Compression

Duration compression helps to cut short a schedule if needed. It can adjust the set schedule by making changes without changing the scope in case, the project is running late. Two methodologies that can be applied: fast tracking and crashing.

a. Fast Tracking

Fast-tracking is another way to use CPM. Fast-tracking finds ways to speed up the pace at which a project is being implemented by either simultaneously executing many tasks or by overlapping many tasks to each other. CPM helps us identify activities that can be used to speed up the pace of the project. Although it is an appealing technique, it has its own share of risks too. As many activities will be simultaneously implemented, it is highly likely to make mistakes and compromise on quality.

b. Crashing

Crashing deals with involving more resources to finish the project on time. For this to happen, you need spare resources to be available at your disposal. Moreover, all the tasks cannot be done by adding extra resources. Need to add new team members to a project and limited divisibility of tasks leads to increased communication and is the basic reason behind it. The crashing technique can also be used by adding time, paid overtime, but it should stay within the decided deadline. It, unfortunately, leads to raising the cost of the project.

3. Simulation

The expected duration of the project is calculated by using a different set of tasks in simulation. The schedule is created on the basis of assumptions, so it can be used even if the scope is changed or the tasks are not clear enough.

4. Resource-Leveling Heuristics

Cutting the delivery time or avoiding under or overutilization of resources by making adjustments with the schedule or resources is called resource leveling heuristics. Dividing the tasks as per the available resources, so that no resource is under or over-utilized. The only demerit of this methodology is it may increase the project's cost and time.

5. Task List

The task list is the simplest project scheduling technique of all the techniques available. Documented in a spreadsheet or word processor is the list of all possible tasks involved in a project. This method is simple and the most popular of all methods. It is very useful while implementing small projects. But for large projects with numerous aspects to consider, a task list is not a feasible method.

6. Gantt Chart

For tracking progress and reporting purposes, the Gantt Chart is a visualization technique used in project management. It is used by project managers most of the time to get an idea about the average time needed to finish a project. A project schedule

Gantt chart is a bar chart that represents key activities in sequence on the left vs time. Each task is represented by a bar that reflects the start and date of the activity, and therefore its duration.

7. Calendar

Many don't consider scheduling tasks on a calendar for their project requirements – when they should! Most of the calendars can be curated with names of their own. In this case, you can create one calendar per project and scheduled events for that project. The calendar shows a timeline for the entire project. The major advantage is that it can be subjected to change as it is shareable. While it seems to be a great technique for tracking a project, it does have certain limitations: you cannot assign tasks to certain people and you cannot see task dependencies.

Project Tracking:

Project Tracking is a method of project management for following the progress (or lack thereof) of activities involved in projects. Potential issues can be spotted and solved by team members and leaders. Tracking projects from the beginning, dealing with problems quickly, and proactively making decisions is what successful project managers do.

Managing all tasks and activities involved, handling multiple files involved, and most importantly, the people who make up the team make this incredibly challenging.

Project manager takes the control of the schedule in the aspects of:

- Project staffing
- Project problems
- Project resources
- Reviews
- Project budget

4) Risk Analysis & Management

- "Tomorrow's problems are today's risks." Hence, a clear definition of a "risk" is a problem that could cause some loss or threaten the progress of the project, but which has not happened yet.
- These potential issues might harm cost, schedule or technical success of the project and the quality of our software device, or project team morale.
- Risk Management is the system of identifying, addressing and eliminating these problems before they can damage the project.
- We need to differentiate risks, as potential issues, from the current problems of the project.
- Different methods are required to address these two kinds of issues.
- For example, staff storage, because we have not been able to select people with the right technical skills is a current problem, but the threat of our technical persons being hired away by the competition is a risk.

Risk Management

A software project can be concerned with a large variety of risks. In order to be adept at systematically identifying the significant risks which might affect a software project, it is essential to classify risks into different classes. The project manager can then check which risks from each class are relevant to the project.

There are three main classifications of risks which can affect a software project:

- Project risks
- Technical risks
- Business risks

1. Project risks: Project risks concern differ forms of budgetary, schedule, personnel, resource, and customer-related problems. A vital project risk is schedule slippage. Since the software is intangible, it is very tough to monitor and control a software project. It is very tough to control something which cannot be identified. For any manufacturing program, such as the manufacturing of cars, the plan executive can recognize the product taking shape.

2. Technical risks: Technical risks concern potential method, implementation, interfacing, testing, and maintenance issues. It also consists of an ambiguous specification, incomplete specification, changing specification, technical uncertainty, and technical obsolescence. Most technical risks appear due to the development team's insufficient knowledge about the project.

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

Other risk categories

1. **Known risks:** Those risks that can be uncovered after careful assessment of the project program, the business and technical environment in which the plan is being developed, and more reliable data sources (e.g., unrealistic delivery date)
2. **Predictable risks:** Those risks that are hypothesized from previous project experience (e.g., past turnover)
3. **Unpredictable risks:** Those risks that can and do occur, but are extremely tough to identify in advance.

Principle of Risk Management

1. **Global Perspective:** In this, we review the bigger system description, design, and implementation. We look at the chance and the impact the risk is going to have.
2. **Take a forward-looking view:** Consider the threat which may appear in the future and create future plans for directing the next events.
3. **Open Communication:** This is to allow the free flow of communications between the client and the team members so that they have certainty about the risks.
4. **Integrated management:** In this method risk management is made an integral part of project management.

5. **Continuous process:** In this phase, the risks are tracked continuously throughout the risk management paradigm.

Risk Management Activities:

Risk Assessment

The objective of risk assessment is to divide the risks in the condition of their loss, causing potential. For risk assessment, first, every risk should be rated in two methods:

- The possibility of a risk coming true (denoted as r).
- The consequence of the issues relates to that risk (denoted as s).

Based on these two methods, the priority of each risk can be estimated: **p = r * s**

Where p is the priority with which the risk must be controlled, r is the probability of the risk becoming true, and s is the severity of loss caused due to the risk becoming true. If all identified risks are set up, then the most likely and damaging risks can be controlled first, and more comprehensive risk abatement methods can be designed for these risks.

- **Risk Identification**

The project organizer needs to anticipate the risk in the project as early as possible so that the impact of risk can be reduced by making effective risk management planning.

A project can be of use by a large variety of risks. To identify the significant risk, this might affect a project. It is necessary to categorize the different risks of classes.

There are different types of risks which can affect a software project:

1. **Technology risks:** Risks that assume from the software or hardware technologies that are used to develop the system.
2. **People risks:** Risks that are connected with the person in the development team.
3. **Organizational risks:** Risks that assume from the organizational environment where the software is being developed.
4. **Tools risks:** Risks that assume from the software tools and other support software used to create the system.
5. **Requirement risks:** Risks that assume from the changes to the customer requirement and the process of managing the requirements change.
6. **Estimation risks:** Risks that assume from the management estimates of the resources required to build the system

2. Risk Analysis: During the risk analysis process, you have to consider every identified risk and make a perception of the probability and seriousness of that risk.

There is no simple way to do this. You have to rely on your perception and experience of previous projects and the problems that arise in them.

It is not possible to make an exact, numerical estimate of the probability and seriousness of each risk. Instead, you should authorize the risk to one of several bands:

- The probability of the risk might be determined as very low (0-10%), low (10-25%), moderate (25-50%), high (50-75%) or very high (+75%).
- The effect of the risk might be determined as catastrophic (threaten the survival of the plan), serious (would cause significant delays), tolerable (delays are within allowed contingency), or insignificant.

Risk Control

It is the process of managing risks to achieve desired outcomes. After all, the identified risks of a plan are determined; the project must be made to include the most harmful and the most likely risks. Different risks need different containment methods. In fact, most risks need ingenuity on the part of the project manager in tackling the risk.

There are three main methods to plan for risk management:

1. **Avoid the risk:** This may take several ways such as discussing with the client to change the requirements to decrease the scope of the work, giving incentives to the engineers to avoid the risk of human resources turnover, etc.
2. **Transfer the risk:** This method involves getting the risky element developed by a third party, buying insurance cover, etc.
3. **Risk reduction:** This means planning a method to include the loss due to risk. For instance, if there is a risk that some key personnel might leave, new recruitment can be planned.

Risk Leverage: To choose between the various methods of handling risk, the project plan must consider the amount of controlling the risk and the corresponding reduction of risk. For this, the risk leverage of the various risks can be estimated.

Risk leverage is the variation in risk exposure divided by the amount of reducing the risk.

Risk leverage = (risk exposure before reduction – risk exposure after reduction) / (cost of reduction)

1. Risk planning: The risk planning method considers each of the key risks that have been identified and develops ways to maintain these risks.

For each of the risks, you have to think of the behavior that you may take to minimize the disruption to the plan if the issue identified in the risk occurs.

You also should think about data that you might need to collect while monitoring the plan so that issues can be anticipated.

Again, there is no easy process that can be followed for contingency planning. It rely on the judgment and experience of the project manager.

2. Risk Monitoring: Risk monitoring is the method king that your assumption about the product, process, and business risks has not changed.