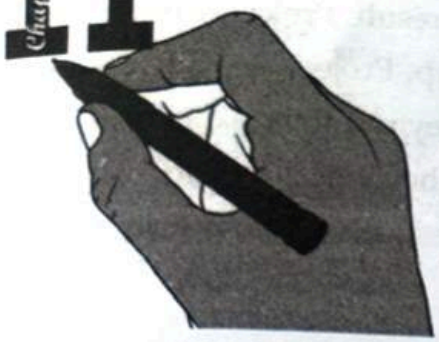
Chapter 11



MANAGING SOFTWARE PROJECT

CHAPTER OUTLINE

After studying this chapter, the reader will be able to understand the

- Introduction
- Software Project Manager
- Management Activities
- Project Planning
- Mile Stones and Deliverable
- Project Scheduling
- Risk Management
- Cost Estimation Techniques

INTRODUCTION

A project is a group of tasks that need to complete to reach a clear result. Projects can vary from simple to difficult and can be operated by one person or in a group. Projects usually described and approved by a project manager or team executive. They go beyond their expectations and objects, and it's up to the team to handle logistics and complete the project on time. For good project development, some teams split the project into specific tasks so they can manage responsibility and utilize team strengths.

A Software Project is the complete procedure of software development from requirement gathering to testing and maintenance, carried out according to the execution methodologies, in a specified period of time to achieve intended software product.

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

SOFTWARE PROJECT MANAGER

A software project manager is a person who undertakes the responsibility of planning, executing and controlling the software project. A project manager closely monitors the development process, prepares and executes various plans, arranges necessary and adequate resources, maintains communication among all team members in order to address issues of cost, budget, resources, time, quality and customer satisfaction.

Project managers are responsible for planning and scheduling project development. The managerial person who supervise the development work to ensure that, it is carried out to the required standards and monitor progress to check that the development is on time and within budget. The manager's job is to ensure that software project meets these constraints and delivers the software that contributes to the goal of organization. Software project management is differing from other type of project management due to following reasons:

- **The product is intangible**

  The software product is intangible in nature and it can be realized only, it cannot be seen or touched. So, it is difficult for project manager to measure progress and should rely on documentation produced by others.

- **There are no standard software processes**

  With a long history in engineering discipline, the process is tried and tested. The engineering process for some types of system, such as building construction is well understood but in software project there is no universally accepted fixed standard ,process change from one organization to another and project to project.

- **Large software Projects are often One-off Projects**

  Software technology is changing very rapidly and it makes the manager's experience and skill obsolete. Lessons learned from previous project may not be transferrable to new projects. So, managers having massive volume of experience may find it difficult to anticipate problems.

MANAGEMENT ACTIVITIES

Project manager's job varies depending on type of organization and software product being developed. So it is difficult to write standard job description for a software manager. Some common activities of project managers are as follows:

1. **Proposal writing**

   Project manager have to write the project proposal to win a contract from the customer. Proposal writing is a skill that acquire through practice and experience. Proposal should include the objective of the project and how it will be carried out. It also includes cost and schedule estimates and justification why the project contract awarded to a particular organization or team.

2. **Project planning and scheduling**

   Project planning is concerned with identifying activities, milestones and deliverable produced by a project. A plan is drawn up to guide the development team towards the project goals.

3. **Project cost**

Project cost includes cost estimation activity that is concerned with estimating the effort, time and resources require accomplishing the project development. The parameters involved in computing the total cost of a software development project includes: hardware and software costs including maintenance, travel and training costs and Effort costs i.e. the costs of paying software engineers and others.

Estimation involves answering the following questions.

- How much effort is required to complete each activity?
- How much calendar time is needed to complete each activity?
- What is the total cost of each activity?

4. **Project monitoring and reviews**

It is the continuing project activity. In this the managers must keep the track of progress of project and compare actual and planned progress and costs. Managers clear that what is going on through informal discussion with project staff and review concerned with overall progress and technical development of system and checking whether the project and goals of the organization paying for the software are still aligned.

5. **Personnel selecting and evaluation**

Project managers usually have to select people to work on their project. Ideally, skilled staff with appropriate experience will be available to work on their project but in most cases, managers have to settle for a less than ideal project team members because.

- The project budget may not cover the use of highly paid staff. Less experienced, less well paid staff may have to be used.
- Staff with appropriate experience may not be available either within an organization or externally. It may be impossible to recruit new staff to the project.
- The organization may wish to develop a skill of its employees. Inexperienced staff may be assigned to a project to learn and to gain experiences.

6. **Report writing and presentation**

Project managers are usually responsible for reporting on the project to both the client and contractor organization. They have to write concise, coherent documents that abstract critical information to detailed project report.

PROJECT PLANNING

Software project planning is task, which is performed before the production of software actually starts. It is there for the software production but involves no concrete activity that has any direction connection with software production; rather it is a set of multiple processes, which facilitates software production. The project plan sets out the resources available to the project the work breakdown and a sche4dule for carrying out the work. So, effective management of software project depends on the thoroughly planning the progress of project. A plan drawn up at the starting of a project should be used as driver for the project. The planning is an iterative. Process, which is only complete when the project itself is complete. As the project information becomes available during the project. The plan should be regularly revised. The details of the project plan vary depending on the type of project organization. However, most plans should include the following sections:

1. **Introduction**

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

2. **Project organization**

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

3. **Risk analysis**

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

4. **Hardware and software resource requirement**

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

5. **Work breakdown**

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

6. **Project schedule**

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

7. **Monitoring and reporting mechanism**

   This defines the management report that should be produced.

Types of Plan

- Quality Plan
- Validation Plan
- Configuration management plan
- Maintenance plan
- Staff development Plan

1. **Validation plan**

   Validation plan describes the approach, resources and schedule used for system validation. It is intended to show that the program does what the user requires.

2. **Configuration and management plan**

   This plan includes the management of system change. When a system is maintained the role of the CM team is to ensure that changes are incorporated in a controlled way. This plan includes configuration management procedures and structures to be used.

3. **Maintenance plan**
   It predicts the maintenance requirements of the system, maintenance cost and effort required.

4. **Staff development plan**
   It describes how the skill and experience of the project team member will be developed.

MILE STONES AND DELIVERABLE

Because software is intangible, information can only be provided as reports and documents that describe the state of software being developed without this information, it is impossible to access how well the impossible to access how well the project is progressing, cost estimates and schedule can be not be updated.

Milestones

Project milestones are the predictable outcome of an activity where some formal report of progress. Milestones are the recognizable end point of software process activity, They may be short report of what has been completed and presented to the management. Milestones should represent the end of a distinct, logical stage in the project.

1. **Deliverables**

   A deliverable is a project report (result) that is delivered to the customer at the end of tome major project phase. Deliverables are usually milestones but milestones need to be deliverables. The project deliverable, which are delivered to customer are the requirement definition and requirement specification.

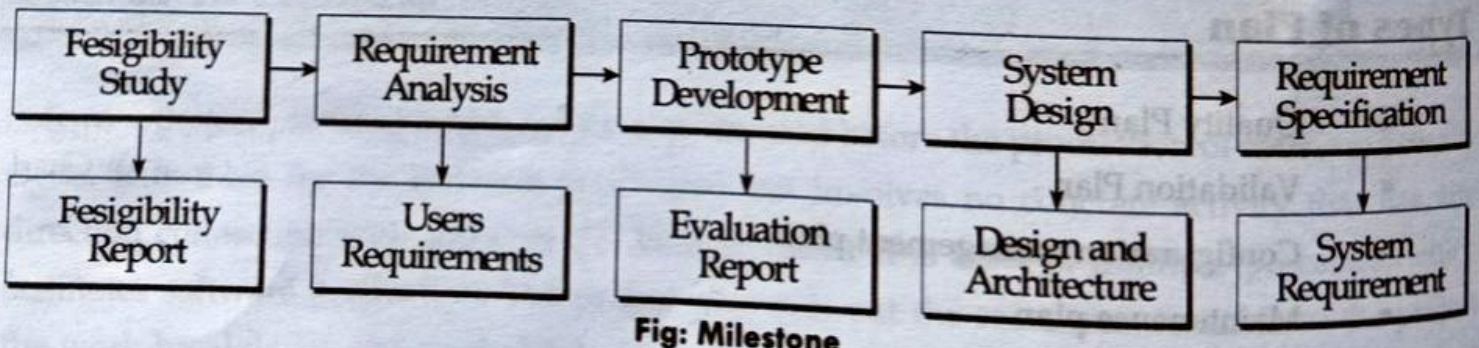


Fig: Milestone

PROJECT SCHEDULING

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

required. Project scheduling means separating the total work involved in a project into separate activities and organize the work so that the work force is optimal following are the principle for scheduling.

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

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

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

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

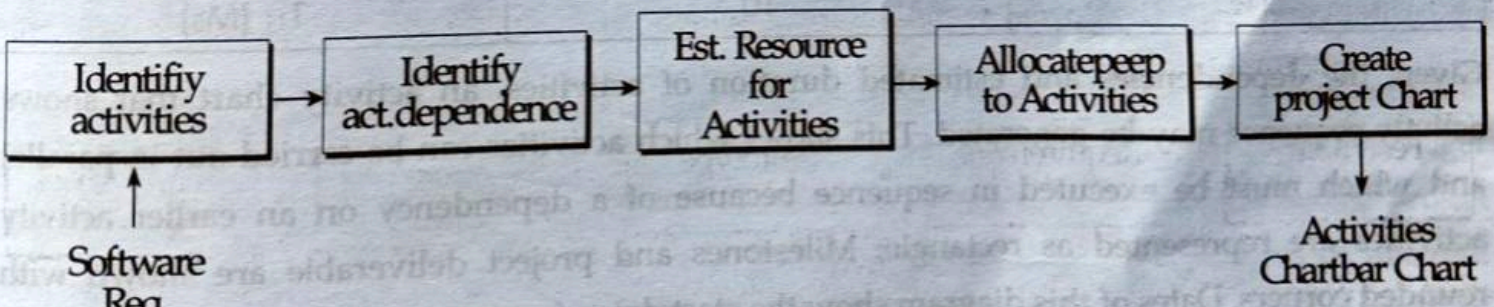


Fig: Project Scheduling Process

Bar Charts and Activity Networks

Project schedules are usually represented as a set of chars and network diagram showing the work breakdown activities dependencies and staff allocations and time allocations. Bar charts and activity network are graphical notations that are used to illustrate the project schedule.

Bar chart shows the schedule against the calendar. i.e. it shows who is responsible for each activity and when the activity is scheduled to being and end.

But "Activity network show the dependencies between the different activity making up a project and critical path. Bar charts and activity charts can be generate automatically from a database of project information using a project management tool.

Example:

Task	Duration Days	Dependencies
T_1	8	
T_2	15	
T_3	15	T_1 (M_1)
T_4	10	
T_5	10	T_2, T_4 (M_2)
T_6	5	T_1, T_2 (M_3)
T_7	20	T_1 (M_1)
T_8	25	T_4 (M_5)
T_9	15	T_3, T_6 (M_4)
T_{10}	15	T_5, T_7 (M_7)
T_{11}	7	T_9 (M_6)
T_{12}	10	T_{11} (M_8)

Given, the dependencies and estimated duration of activities, an activity chart that shows activity sequence may be generated. This shows which activities can be carried out in parallel and which must be executed in sequence because of a dependency on an earlier activity activities are represented as rectangle; Milestones and project deliverable are shown with rounded corners. Dates of this diagram show the start date of activity is written in British style.

"The minimum time required to finish the project can be estimated by considering the longest path in the activity graph (the critical path). "Critical path is the sequence of dependence activates that defines the time required to complete the project." Any slippage in the completion in any critical activity cause project delays because the following activities can not start until the delayed activity has been completed.

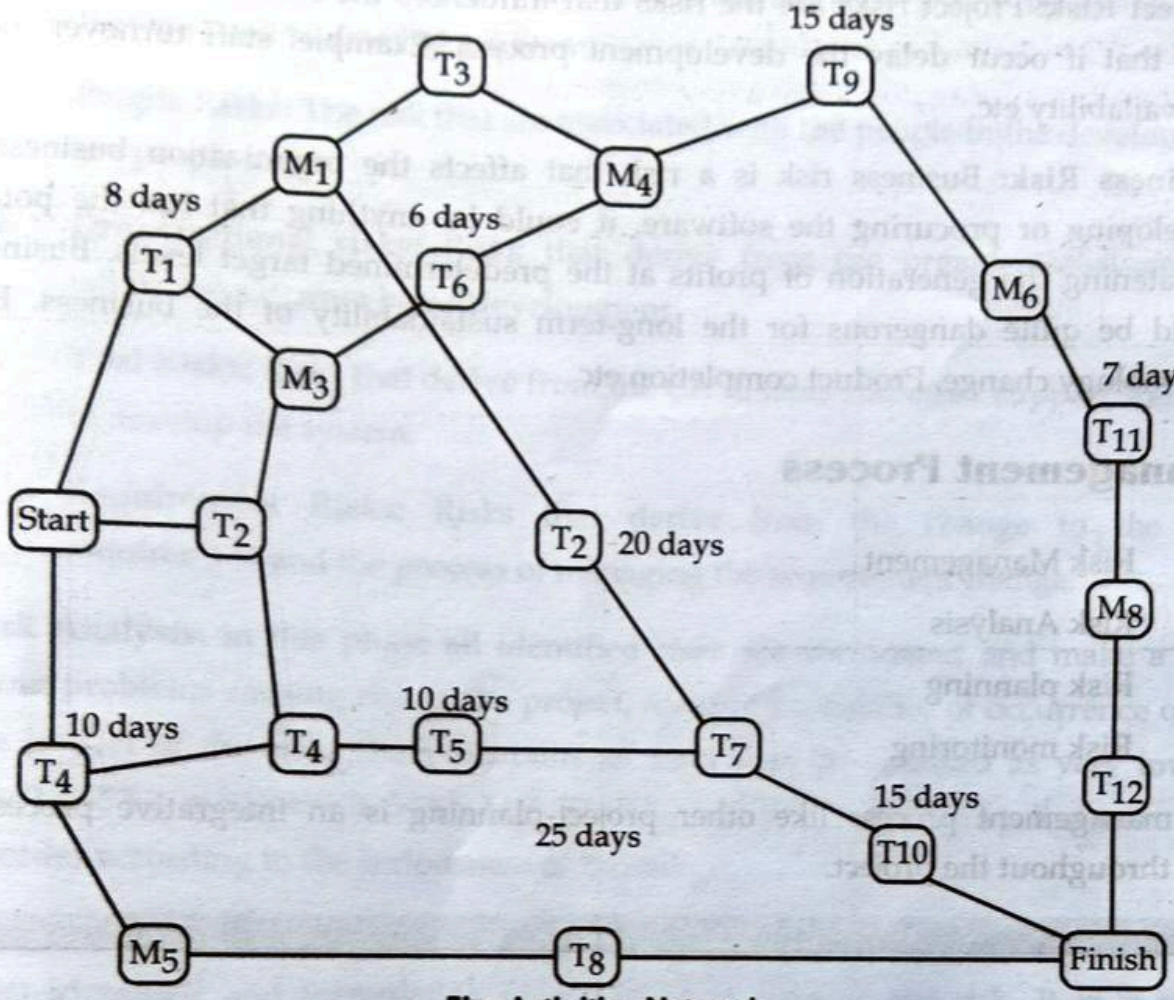


Fig: Activities Network

RISK MANAGEMENT

Risk means something that if occur threaten the project development process regularly and influence the project schedule and product quality. Risks are the factor that the project manager would prefer not to have happen. The risk that may affect the project depends on the project and organizational environment where the software is being developed. Because of inherent uncertainties that most projects face risk management is important.

The risk management is process of identifying risks that might affect the project schedule or quality of the project/software being developed and taking action to avoid these risks. Risk management involves identifying and assessing major project risks to establish the probability that they will occur and the consequences for the project if that risk does arise.

There are mainly three categories of risks:

1. **Product Risk**: Product risks are the risks that influence the quality attributes as well as performance of software being developed. **Example:** failure of purchase component to perform as expected

2. **Project Risk**: Project risks are the risks that influences the schedule of software project and that if occur delay the development process. Example: staff turnover, hardware unavailability etc.

3. **Business Risk:** Business risk is a risk that affects the organization business that is developing or procuring the software. it could be anything that has the potential of threatening the generation of profits at the predetermined target levels. Business risks could be quite dangerous for the long-term sustainability of the business. **Example:** Technology change, Product completion etc.

Risk Management Process

- Risk Management
- Risk Analysis
- Risk planning
- Risk monitoring

The risk management process like other project-planning is an integrative process which continues throughout the project.

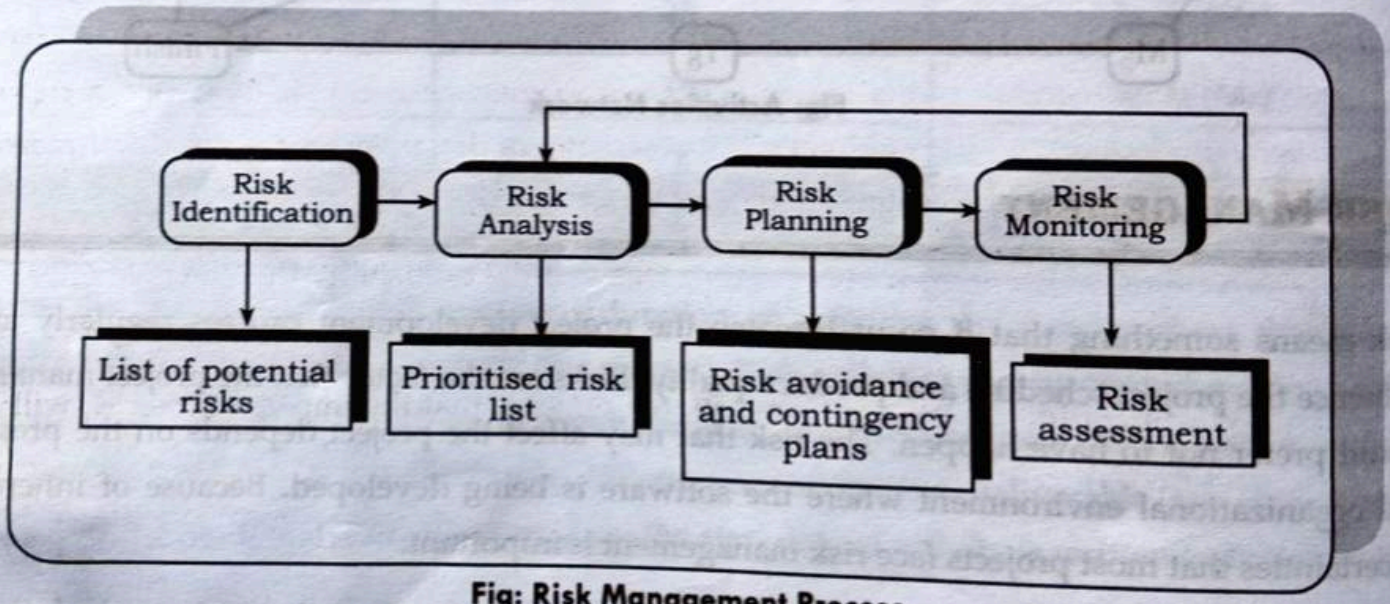


Fig: Risk Management Process

1. **Risk identification**

Risk identification involves brainstorming activities. it is the first stage of the risk management. It concerned with discovering possible risk to the project and may be carried out as a team process using past experience. It also involves preparation of risk list. Brainstorming is a group discussion technique where all the stakeholders meet together. This technique produces new ideas and promotes creative thinking. To help the process, a checklist of different types of risks may be prepared. Preparation of risk list involves identification of risks that is occurring continuously in previous software projects. There are mainly six types of risk that can arise in project.

i. **Technology Risk:** The risk that derive from the software or hardware technology that are used to develop a system.

ii. **People Risks:** The risk that are associated with the people in the development term are people risks.

iii. **Organizational risks:** Risks that derive from the organizational environment where the software being development.

iv. **Tool Risks:** Risks that derive from the CASE tools and other support software used to develop the system.

v. **Requirement Risks:** Risks that derive from the change to the customer requirement and the process of managing the requirement change.

2. **Risk Analysis:** in this phase all identified risks are considered and make a judgment about problems causing risk in the project, identify probability of occurrence of risk and the impact of the risk. The probability of risks may be accessed as very low (<10%), low(10-25%), moderate(26-50%), high(51-75%) and very high(>75%). Risks are tabulated in order according to the seriousness of the risk.

3. **Risk planning:** Risk planning is the process of consideration of the key risks that have been identified and formulation of strategies to manage the risk. It is not a simple process; it depends on the judgment and experience of the project manager. the strategies to manage the risk are as follows:

i. **Avoidance Strategies:** This strategy tells that the probability that the risk will arise will be reduced. **Example:** The strategy for dealing with defective components.

ii. **Minimization Strategies:** These strategies mean that the impact of the risk will be reduced.

Example: Staff illness, Reorganize team so that there is more overlap of work and people therefore understand each other's job.

iii. **Contingency Plans:** These strategies means that you are prepared for the worst and have a strategy in place to deal with it.

4. **Risk monitoring:** Risk monitoring means regularly assessing each of the identified risk to decide whether or not the risk is becoming more or less portable and whether the effect of the risk have changed. It should be a continuous process, and at every management progress review.

COST ESTIMATION TECHNIQUES

Software projects are notorious for going past their deadline, going over budget, or both. The problem lies in the estimation of the amount of effort required for the development of a project. While developing a software project, the project is split in to a number of separate activities. So cost estimation concerned with estimates of effort and time with the project activities. It is the process of predicting the effort required to develop a software system. Many estimation models have been proposed over the years. As a number of these models rely on a software size estimate as input, we first provide an overview of common size metrics. The cost estimation is usually dependent upon the size estimate of the project, which may use lines of code or function points as metrics. Cost estimation is usually measured in terms of effort. The most common metric used is person months or years (or man months or years). The effort is the amount of time for one person to work for a certain period of time. For any new software project, it is necessary to know how much it will cost to develop and how much development time will it take. These estimates are needed before development is initiated. There are several different techniques for performing software cost estimation, including expert judgment and algorithmic models. Software cost estimation is the process of predicting the effort required to develop a software system. Many estimation models have been proposed over the years. As a number of these models rely on a software size estimate as input, we first provide an overview of common size metrics. Estimation is done to answer the following questions:

- How much effort is required complete an activity?
- How much calendar time is needed to complete each activity?
- What is the total cost of an activity?

Since the cost of development is primarily the cost of the effort. There are three parameters involves in computing the total cost of a software development.

- Hardware and software cost including maintenance.
- Travel and training cost
- Effort cost i.e. the cost of paying software engineers.

The effort cost is calculated by calculating following:

- The cost of providing heating and lighting office space.
- Cost of support staffs.
- Cost of networking and communications
- Cost of social security and employee benefits.

Software Pricing Factors

Software pricing must take into account broader organizational, economic, political and business consideration. The major factors that affect the software pricing are as follows:

- Development organization quotes a low price if want to move into a new segment of software market. Accepting low profit in one project may give the opportunity of more profit later.
- If an organization has no idea about cost estimation then it may increase its price by some contingency.
- It the customer is not clear about their requirement then there is chance of changing requirement at that time organization may lower its price. After contract is awarded then high prices may be charged for changes to the requirement.
- If the financial health of the organization is not good then they may lower their price to gain contract and establish themselves in business.

Estimation Techniques

Cost estimation is one of the most challenging tasks in project management. It is to accurately estimate needed resources and required schedules for software development projects. The software estimation process includes estimating the size of the software product to be produced, estimating the effort required, developing preliminary project schedules, and finally, estimating overall cost of the project.

It is very difficult to estimate the cost of software development. Many of the problems that plague the development effort itself are responsible for the difficulty encountered in estimating that effort. Software, however, is intangible, invisible, and intractable. It is inherently more difficult to understand and estimate a product or process that cannot be seen and touched. Software grows and changes as it is written. When hardware design has been inadequate, or when hardware fails to perform as expected, the solution is often attempted through changes to the software. This change may occur late in the development process, and sometimes results in unanticipated software growth.

The approaches to cost estimation can be tackled using wither top-down or a bottom up approach.

Top down Approach

Top-down estimating method is also called Macro Model. Using top-down estimating method, an overall cost estimation for the project is derived from the global properties of the software project, and then the project is partitioned into various low-level components. This method is

more applicable to early cost estimation when only global properties are known. In the early phase of the software development, It is very useful because there are no detailed information available. This approach starts at system level. The estimator starts by examining the overall functionality of the product and how that functionality is provided by interacting sub systems.

Advantages

- It focuses on system-level activities such as integration, documentation, configuration management, etc., many of which may be ignored in other estimating methods and it will not miss the cost of system-level functions.
- It requires minimal project detail, and it is usually faster, easier to implement.

Disadvantages

- It often does not identify difficult low-level problems that are likely to escalate costs and sometime tends to overlook low-level components.
- It provides no detailed basis for justifying decisions or estimates.

Bottom up Approach

Using bottom-up estimating method, the cost of each software components is estimated and then combines the results to arrive at an estimated cost of overall project. It aims at constructing the estimate of a system from the knowledge accumulated about the small software components and their interactions. The leading method using this approach is COCOMO's detailed model.

Advantages

- It permits the software group to handle an estimate in an almost traditional fashion and to handle estimate components for which the group has a feel.
- It is more stable because the estimation errors in the various components have a chance to balance out.

Disadvantages

- It may overlook many of the system-level costs (integration, configuration management, quality assurance, etc.) associated with software development.
- It may be inaccurate because the necessary information may not available in the early phase.
- It tends to be more time-consuming.
- It may not be feasible when either time or personnel are limited.

The main cost estimation techniques are:

1. **Algorithm Cost Modeling:** The algorithmic method is designed to provide some mathematical equations to perform software estimation. These mathematical equations are based on research and historical data and use inputs such as Lines of Code (LOC), number of functions to perform, and other cost drivers such as language, design methodology, skill-levels, risk assessments, etc. The algorithmic methods have been largely studied and there are a lot of models have been developed, such as COCOMO

   Algorithmic cost model can be built by analyzing the cost and attributes of similar projects by using an empirical formula

   Effort = A x $Size^B$ x M, where

   A: depends on organizational practice and type of software that is developed

   B: 1-1.5 reflects disproportionate effort for large projects

   M: reflects product, process and people attributes

   Size: size may be assessment of code size expressed in function or object points.

COCOMO Model

The Constructive Cost Model (COCOMO) is an algorithmic software cost estimation model developed by Barry Boehm in 1981. This model is an empirical model derived by collecting data from a large number of software projects, then analyzing that data to discover formulae. The model uses a basic regression formula, with parameters that are derived from historical project data and current project characteristics. The first model of COCOMO model known as COCOMO 81 assumes that the software would be developed using waterfall model and from scratch level.

There have been radical changes in software development practice. Software can be created by using reusable components and linking them by scripting language. Existing software reengineering to create new software, CASE tools are used to generate source code automatically from the design, prototyping and incremental development are heavily in practice. To take these changes in account COCOMO 2 model is introduced.

In COCOMO, projects are categorized into three types:

i. **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.

ii. **Moderate:** A development project can be treated with moderate 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 moderate (Semidetached) system includes developing a new operating system (OS), a Database Management System (DBMS), and complex inventory management system.

iii. **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 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*

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*

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

Solution: The basic COCOMO equation takes the form:

Effort=a1*(KLOC) a2 PM

Tdev=b1*(efforts)b2 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

Example: 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

P = 176 LOC/PM

The Basic COCOMO

It is the one type of static model to estimates software development effort quickly and roughly. It mainly deals with the number of lines of code and the level of estimation accuracy is less as we don't consider the all parameters belongs to the project. The estimated effort and scheduled time for the project are given by the relation:

Effort (E) = a*(KLOC)b MM

Scheduled Time (D) = c*(E)d Months(M)

Where,

- E = Total effort required for the project in Man-Months (MM).
- D = Total time required for project development in Months (M).
- KLOC = the size of the code for the project in Kilo lines of code.
- a, b, c, d = The constant parameters for a software project.

PROJECT TYPE	a	b	c	d
Organic	2.4	1.05	2.5	0.38
Semidetached	3	1.12	2.5	0.35
Embedded	3.6	1.2	2.5	0.32

Example: For a given project was estimated with a size of 300 KLOC. Calculate the Effort, Scheduled time for development. Also, calculate the Average resource size and Productivity of the software for Organic project type.

Ans: Given estimated size of project is: 300 KLOC

For Organic

Effort (E) = a*(KLOC)b = 2.4*(300)1.05 = 957.61 MM

Scheduled Time (D) = c*(E)d = 2.5*(957.61)0.38 = 33.95 Months(M)

Avg. Resource Size = E/D = 957.61/33.95 = 28.21 Mans

Productivity of Software = KLOC/E = 300/957.61 = 0.3132 KLOC/MM = 313 LOC/MM

For Moderate

Effort (E) = a*(KLOC)b = 3.0*(300)1.12 = 1784.42 MM

Scheduled Time (D) = c*(E)d = 2.5*(1784.42)0.35 = 34.35 Months(M)

For Embedded

Effort (E) = a*(KLOC)b = 3.6*(300)1.2 = 3379.46 MM

Scheduled Time (D) = c*(E)d = 2.5*(3379.46)0.32 = 33.66 Months(M)

COCOMO 2

This model incorporates a range of sub-models that produce increasingly detailed software cost estimates. The sub-models in COCOMO 2 are:

a. **Application composition model:** It is used when software is composed from existing parts. Supports prototyping projects and projects where there is extensive reuse based on standard estimates of developer productivity in application (object) points/month. It Takes CASE tool use into account. The formula is:

PM = NOP/PROD = (Object Points x (1 - %reuse/100)) / PROD

Where, PM is the effort in person-months

NOP is the new object points

PROD is the productivity.

b. **Early design model:** This model is used when requirements are available but design has not yet started. In this model, Estimates can be made after the requirements have been agreed. The empirical formula used to calculate person month is: $PM = A \times Size^{B} \times M$

Where, $M = PERS \times RCPX \times RUSE \times PDIF \times PREX \times FCIL \times SCED$

A= 2.94 in initial calibration

Size in KLOC

B varies from 1.1 to 1.24 depending on novelty of the project, development flexibility, risk management approaches and the process maturity.

c. **Reuse oriented Model:** This model is used to compute the effort of integrating reusable components. Major effort is required to integrate automatically generated code. The empirical formula is:

PMAuto = (ASLOC x (AT/100)) / ATPROD

Where, ASLOC – No. LOC that have to be adapted

AT - % of adapted code that is automatically generated

ATPROD – engineer productivity in adapting code (2400 LOC/month)

d. **Post-architecture model.** This model is used once the system architecture has been designed and more information about the system is available. In this model we uses same formula as early design estimates: $PM = A \times Size^B \times M$)

Where, Size estimate for the software should be more accurate at this stage. It takes into consideration the factors like: New code to be developed, rework required to support change and extent of possible reuse.

2. **Expert judgment:** Expert judgment techniques involve consulting with software cost estimation expert or a group of the experts to use their experience and understanding of the proposed project to arrive at an estimate of its cost. The estimating steps using this method are:

- Coordinator presents each expert with a specification and an estimation form.
- Coordinator calls a group meeting in which the experts discuss estimation issues with the coordinator and each other.
- Experts fill out forms anonymously
- Coordinator prepares and distributes a summary of the estimation on an iteration form.
- Coordinator calls a group meeting, specially focusing on having the experts discuss points where their estimates varied widely.
- Experts fill out forms, again anonymously, and steps 4 and 6 are iterated for as many rounds as appropriate.

3. **Estimation by analogy:** Estimating by analogy means comparing the proposed project to previously completed similar project where the project development information is known. Actual data from the completed projects are extrapolated to estimate the proposed project. This method can be used either at system-level or at the component-level. Estimating by analogy is relatively straightforward. Actually in some respects, it is a systematic form of expert judgment since experts often search for analogous situations so as to inform their opinion.

The steps using estimating by analogy are:

- Characterizing the proposed project.
- Selecting the most similar completed projects whose characteristics have been stored in the historical data base.
- Deriving the estimate for the proposed project from the most similar completed projects by analogy.

4. **Parkinson's law:** Parkinson's Law states that work expands to fill the time available. In this method, the cost is determined by available resources rather than by objective assessment.

5. **Pricing to win:** In this method, the software cost is estimated to be whatever the customer has available to spend on the project. The estimated effort depends on customer's budget not on the software functionality.

EXERCISE

1. What is software project management? Explain the management activities in brief.
2. Managing software project is difficult than other project management. Why?
3. What is project planning? Project planning is iterative. Explain.
4. Explain different sections of project plan.
5. What is project scheduling? Explain the importance of activity network and bar chart with example.
6. What is risk management? Explain risk management process with block diagram.
7. What do you mean by cost estimation? Explain different software cost estimation techniques.
8. What is COCOMO model? Explain different sub models of COCOMO 2 model.

❑❑❑