Unit-4

Software Project Management

❖ RESPONSIBILITIES OF SOFTWARE PROJECT MANAGER

- Software project managers take the overall responsibility of project success.
- A project manager is the person who is responsible for accomplishing the stated project objectives.
- The job responsibility of a project manager ranges from invisible activities like building up team spirit to highly visible customer presentations. (Planning to Deployment).
- A project manager bridging the gap between the production team and client.
- He managing the constraints of the project management triangle, which are cost, time, scope, and quality.
- General activities of manager like → project proposal writing, project cost estimation, scheduling, project staffing, software process tailoring, project monitoring and control, software configuration management, risk management, interfacing with clients, managerial report writing and presentations, etc.
- All the above activities are mainly classified into: *project planning, project monitoring and control activities.* Project planning activity starts before development, while monitoring and control starts after development.
- Key among his or her duties is the recognition of risks that affect the success of the running project (risk management). It follows that a project manager is one who is responsible for making decisions both large and small, in such a way that risk is controlled and minimized.
- Time and cost estimation are also important factors of project manager responsibilities.
- Follow project status and modify it to ensure success.
- Every decision taken by the project manager should be taken in such a way that it directly benefits the project.
- He sets up development milestones and entry or exit criteria.

➢ **The skills required in project manager to manage the project, are:**

- He must have theoretical knowledge of different project management techniques.
- A good decision-making capability is also required in project manager.
- He should be client representative and has to determine and implement the exact needs of the client, and capable of understanding and discussing the problems with customers.
- He should have the management skills like ask the questions and resolve conflicts regarding project.
- Successful project manager one who focuses on risk management.
- He should have team leadership skill so the project which is divided into different persons are managed well and completed as per schedule.
- He should have the experience in the related area of the developing project.
- Monitoring and scheduling the progress of the project.
- Evaluating performance of each milestone of the project.
- Some skills like tracking and controlling the progress of the project, customer interaction, managerial presentations, and team building are acquired through experience

❖ METRICS FOR PROJECT SIZE ESTIMATION

- Metrics are the tools that help in better monitoring and control.
- Size of the program (or project) is neither the number of bytes that the source code occupies nor the byte size of the executable code.
- But it is an indicator of the effort and time required to develop the project. So, it indicates project development complexity.
- The project size is a measure of the problem complexity in terms of the effort and time required to develop the entire product.
- There are several metrics to measure problem size. Each of them has its own advantages and disadvantages.
- We will consider two important metrics to estimate size: *Lines of code (LOC) and Function point (FP).*

➢ **Lines of code (LOC):**

- The simplest measure of problem size is Lines Of Code (LOC) or Source Lines Of Code (SLOC).
- It is very popular because of its simplicity.
- LOC is a software metric used to measure the size of a software program by counting the number of lines in the text of the program's source code. Comments line and headers lines are ignored at counting the source code.
- In other term, it is software metric that measures the size of software in terms of lines in the program.
- In order to estimate the LOC count at the beginning of a project, project managers usually divide the problem into modules and each module into sub modules and so on, until the sizes of the different leaf-level modules can be approximately predicted.
- According to Boehm, every line of source text should be calculated as one LOC.
- By using the estimation of the lowest level modules, project managers arrive at the total size estimation.
- One main advantage of LOC is that it is very simple to understand and directly relates to end product.

➢ **However, LOC has several disadvantages.**

- LOC is language dependent. LOC focuses on coding activity only, while a good problem size measure should consider the overall complexity of the problem.
- LOC gives only numerical values of problem size, that is depend on coding style. It can't measure the size of specification.
- LOC does not measure with the quality and efficiency of the code.
- LOC metric suffer from accuracy when use of high-level languages, code reuse, library subroutine etc.
- LOC doesn't issue logical and structural complexities.
- It is very difficult to accurately estimate the LOC in the final product from the problem specification. Accurate LOC can be computed only after code has been fully developed.
- LOC doesn't work well with non-procedural languages.

➢ **Function Point metric (FP):**

- Function point metric was first proposed by Albrecht in 1979 and internationally approve modified model in 1983. This metric overcomes many of the shortcomings of the LOC metric.
- Function point metrics, measure functionality from the users' point of view, that is, on the basis of what the user requests and receives in return

- One of the important advantages of using the function point metric is that it can be used to easily estimate the size of a software product directly from the problem specification.
- In LOC metric, the project size can be accurately determined only after the product has fully been developed. While in case of FP, the size of a software product is directly dependent on the number of different functions or features it supports.
- Software supporting many features or functions should have larger FP in size.
- Each function when invoked reads some input data and transforms it to the corresponding output data.
- By using the number of input and output data values, function point metric computes the size of a software product.
- FP considers five different characteristics of the product to calculate the size. The function point (FP) of given software is the weighted sum of these five items, and it will give unadjusted function point (UFP).

  UFP = (Number of inputs)*4 +
  (Number of outputs)*5 +
  (Number of inquiries)*4 +
  (Number of files)*10 +
  (Number of interfaces)*10
- The meaning of each parameter is as follow:

(1) Number of inputs

- Each data item input by the user is counted.
- Group of related inputs are considered as a single input.
- For example, while entering the data concerning student to student information system software; the data items name, age, gender, address, phone number, etc. are together considered as a single input.

(2) Number of outputs

- It refers to reports printed, screen outputs, error messages produced, etc.
- The set of related data items is counted as one input.

(3) Number of inquiries

- It is the number of distinct interactive queries which can be made by the users.
- These should be user commands for specific actions.

(4) Number of files

- Each logical file is counted. A logical file means groups of logically related data is counted as a file.
- The files can be data structures or physical files.

(5) Number of interfaces

- The interfaces used to exchange information with other systems.
- Examples of such interfaces are data files on tapes, disks, communication links with other systems etc.
- Once the unadjusted function point (UFP) is computed, the technical complexity factor (TCF) is computed next.
- TCF refines the UFP by considering 14 factors which assigns 0 (no influence) to 6 (strong influence). These numbers are summed and yielding DI (Degree of Influence).
- Now TCF is computed as

TCF = (0.65 + 0.01 * DI

- As DI can vary from 0 to 70, TCF can vary from 0.65 to 1.35.

Finally: FP = UFP * TCF

Advantages of function point (FP)

- It is not restricted to code.
- It is language independent.
- It is more accurate than estimate LOC.
- We can have the measure from the specification that can't be done with LOC.

Shortcomings of function point (FP) metric

- It ignores quality of output.
- It is fully oriented to traditional data processing system.
- Major shortcoming of FP is it does not calculate the algorithmic complexity of software.
- Function point has been criticized as not being universally applicable to all types of software. Because FP doesn't capture all functional characteristics of real-time software.
- The FP is originally designed for business information system, and it is not suited for many engineering and embedded systems.
- To overcome the problem of FP, an extension of the function point metric called feature point metric is proposed.
- Feature point metric incorporates an extra parameter algorithm complexity.
- This parameter ensures that the computed size using the feature point metric shows the fact that the more is the complexity of a function, the greater is the effort required to develop it and therefore its size should be larger.

❖ PROJECT ESTIMATION TECHNIQUES

The Estimation is prediction or a rough idea to determine how much effort would take to complete a defined task.

- Project estimation is a very important activity.
- The important project parameters that are estimated include: project size, effort required to develop the software, project duration, and cost.
- Project estimation determines how much money, effort. Resources and time it will take to build a specific system or product.
- These estimates help in project resource planning and scheduling.
- There are three main categories of project estimation techniques:
  1. Empirical estimation techniques
  2. Heuristic techniques
  3. Analytical estimation techniques

➢ **Empirical estimation techniques:**

- These techniques do not use mathematical equations to estimate the cost of the project. As it is done using prior experience and common sense over the years.
- Empirical estimation techniques are based on making an educated guess of the project parameters.

- The project managers use their past experiences to make guess.
- There are two popular empirical estimation techniques:
  1. Expert judgment
  2. Delphi cost estimation

1. Expert judgment

- This is most widely used empirical estimation technique.
- In which, an expert makes an educated guess of the problem size after analysing the problem in detail.
- Expert estimates the costs of different modules (or units) then combines them for overall estimation.
- But in some situation, an expert may not have knowledge of all aspects of project or he may overlook some factors. For simple example, the expert may have very good knowledge of database and analysis part but may not be aware with testing or presentation part.
- So, the chance of human error or individual bias should be there in this technique.
- To solve the above problem → a more refined form of expert judgment is the estimation made by group of experts. It should minimize the factors of above problem.
- Even after this, chance of biasing is also there in expert group judgment also. Because sometimes unusual decision taken by one member can affect the whole group.

2. Delphi cost estimation

- It provides solution to the shortcomings of the expert judgment approach.
- Delphi estimation is carried out by a team having a group of experts (estimators) and a coordinator.
- In it, the coordinator provides each estimator with a copy of the software requirements specification (SRS) document and a form for recording his cost estimate.
- Estimators complete their individual estimates anonymously (unidentified) and submit to the coordinator.
- The estimators mention any unusual characteristics of the product/project in their estimation report.
- The coordinator prepares and distributes the summary of the responses of all the estimators and include any unusual noted by any estimator.
- Based on this summary, the estimators re-estimate.
- This process is iterated for several rounds.
- Discussion among the estimators is not allowed during the entire estimation process.
- After the completion of several iterations of estimations, the coordinator takes the responsibility of compiling the results and preparing the final estimate.

➢ **Heuristic estimation techniques:**

- In these techniques, project parameters are modelled using mathematical expressions.
- Once the basic (independent) parameters are known, the other (dependent) parameters can be easily determined by substituting the value of the basic parameters in the mathematical expression.
- Various heuristic estimation models can be divided into the following two classes: static single variable model and static multi variable model.
- In single variable estimation models, estimate the desired characteristics of a problem such as software product size.

- A single variable estimation model takes the following form :

  Estimated Parameter = c1 * ed1

  (where e is the basic characteristic of the software (independent variable) already estimated, Estimated Parameter is the dependent parameter to be estimated, c1 and d1 are constants.)
- The example of static single variable cost estimation variable is the basic COCOMO model.
- The multivariable cost estimation model takes the following form :

  Estimated Parameter = c1*e1d1 + c2*e2d2+ ...

  (where e1, e2… are the basic characteristic of the software (independent variable) already estimated and c1, c2, d1, d2, … are constants.)
- Multivariable estimation models are expected to give more accurate estimates compared to the single variable models.
- The example of a multivariable estimation model is the intermediate COCOMO model.

  After analyzing various types of projects size using LOC, Boehm postulated that, any software development project can be classified into one of the following three categories based on the development complexity :

  (i) Organic

  (ii) Semi detached

  (ii) Embedded
- This classification is based on characteristics of the product, development team and development environment.
- These three product classes correspond to application, utility and system programs, respectively.
- Data processing programs are considered to be application programs. Compilers, linkers, etc., are utility programs. Operating systems and real-time system programs, etc. are system programs.

➢ **Analytical estimation techniques:**

- This Technique derives the required results starting from certain simple assumptions.
- Analytical estimating is a structured estimating technique often used in work measurement.
- In this technique, the required results are derived with basic assumptions regarding the project.
- Thus, unlike empirical and heuristic techniques, analytical techniques do have scientific basis.
- Halstead's software science is an example of an analytical technique which is especially useful for estimating software maintenance efforts.
- However, Halstead's software science is not very useful for planning development projects because this technique derives the required results after program has been developed. (It depends on completed code). But it is very useful in estimation software maintenance efforts.
- So, for predicting software estimation, it outperforms both empirical and heuristic techniques.

Boehm's definitions of organic, semidetached, and embedded systems are explained below.

Organic

- This type of project has small group of teams.
- Develop well understood application programs and having experienced developers with similar types of programs development.
- Little or no innovation is there in this type of projects.

- Project size should be 2-50 KLOC in organic type.
- For example, data processing programs, payroll, inventory management system.

Semidetached

- This type of development consists of a mixture of experienced and inexperienced staff.
- Team members may have limited experience on related systems.
- Medium innovation is there in this type of projects.
- Project size should be 50-300 KLOC in organic type.
- For example, compilers, interpreters, assemblers.

Embedded

- This type of software strongly coupled to complex hardware.
- Tight or inflexible (rigid) regulations on the operational procedures exist.
- In this projects, great deal with innovation and requirements constraints. Deadlines are strict and development consists of complex interface.
- Project size is large and development team is also large among which few are experienced with same type of development.
- Project size is approximately over 300 KLOC.
- For example, real time system like air traffic control, ATMs etc.
- Brooks states that utility programs are three times as difficult to write as application programs and system programs are three times as difficult as utility programs. So, the relative level of complexity ratio for these three categories of products is 1:3: 9.
- For the three product categories, Boehm provides different sets of expressions to predict the effort (in unit of person-months) and development time from the size estimation given in KLOC.

➢ **COCOMO (A Heuristic Estimation Technique) :**

- COCOMO (COnstructive COst estimation MOdel) was proposed by Boehm.
- It is regression-based model provides hierarchy of software cost estimation models.

(**Note:** *regression method* used to identify the relationship between one dependent variable and several independent variables.)

- Software cost estimation should be done through three stages: Basic COCOMO, Intermediate COCOMO, and complete COCOMO.
- Each level achieves successive accuracy.

➢ **Basic COCOMO model**

- It is static, single valued model that computes software development efforts using program size (expressed in LOC).
- It is quick and rough estimation technique.
- The basic COCOMO model gives an approximate estimate of the project parameters.
- The basic COCOMO estimation model is given by the following expressions:

$$\textbf{Effort (E)} = a1 * (KLOC)^{a2} \text{ PM}$$

$$\textbf{Tdev} = b1 * (Effort)^{b2} \text{ Months}$$

Where :

KOLC is Kilo Lines of Code.

a1, a2, b1, b2 are constants for each category 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).

- The effort estimation is expressed in units of person-months (PM).

Estimation of development effort in basic COCOMO model.

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

Organic **:** Effort (E) = $2.4 (KLOC)^{1.05}$ PM

Semidetached : Effort (E) = $3.0 (KLOC)^{1.12}$ PM

Embedded **:** Effort (E) = $3.6 (KLOC)^{1.20}$ PM

Estimation of development time in basic COCOMO model

- 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

Semidetached : Tdev = $2.5 (effort)^{0.35}$ Months

Embedded **:** Tdev = $2.5 (effort)^{0.32}$ Months

- It is important to note that the effort and the duration estimations obtained using the COCOMO model are called as nominal effort estimate and nominal duration estimate.

Basic COCOMO Model				
Type of System	a1	a2	b1	b2
Organic (2-50 KLOC)	2.4	1.05	2.5	0.38
Semi-detached (50-300 KLOC)	3.0	1.12	2.5	0.35
Embedded (Approx over 300 KLOC)	3.6	1.20	2.5	0.32

Example: Assume that the size of an organic type software product has been estimated to be 2,000 lines of source code. Assume that the average salary of software engineers be Rs. 20,000/- per month. Determine the effort required to develop the software product and the nominal development time.

Solution: Given problem is in basic COCOMO model and source code lines are 2000 (means 2 KLOC), so it is coming in organic type of development.

For the basic COCOMO model :

Effort(E) = 2.4 $(KLOC)^{1.05}$PM

= 2.4 $(2)^{1.05}$

= 5 PM

Nominal development time = 2.5 * $(effort)^{0.38}$ Months

= 2.5 * $(5)^{0.38}$

= 4.6 Months

Cost required for developing the product : = 4.6 * 20000

= 92000 /-

- **Intermediate COCOMO model**

- The basic COCOMO model assumes that effort and time development are functions of the product size alone. Its result in lack of accuracy.
- But some factors from other projects may also affect the efforts and time.
- Therefore, in order to obtain an accurate estimation of the effort and project duration, the effect of all relevant parameters must be taken into account.
- The intermediate COCOMO model doing this by using a set of 15 cost drivers (multipliers) based on various attributes of software development.
- So, Intermediate COCOMO model - computes software development effort as function of program size and a set of "cost drivers" that include subjective assessment of product, hardware, personnel and project attributes.
- Project manager doing the task of rating these 15 parameters in scale 1 to 3 and then cost driver values are estimated.
- The cost drivers can be grouped into four categories.

1. **Product attributes:**
   - Required software reliability (RELY)
   - Database size (DATA)
   - Product complexity (CPLX)
2. **Computer attributes:**
   - Execution time constraint (TIME)
   - Main storage constraint (STOR)
   - Virtual machine volatility (VIRT)
   - Computer turnaround time (TURN)
3. **Personal attributes:**
   - Analytical capability (ACAP)
   - Application experience (AEXP)
   - Programmer capability (PCAP)
   - Virtual machine experience (VEXP)
   - Programming language experience (LEXP)
4. **Project attributes:**
   - Modern programming practice (MODP)
   - Use of software tools (TOOL)
   - Requirement development schedule (SCED)

➢ Complete COCOMO model

- It is also known as detailed COCOMO model or advanced COCOMO model.
- A major shortcoming of both the basic and intermediate COCOMO models is that they consider a software product as a single homogeneous entity.
- But most large systems are made up several smaller sub-systems.
- For example, some subsystems may be considered as organic type, some semidetached, and some embedded.
- And these subsystems may be different requirements, development complexities and may not have previous experience of similar developing.
- The complete COCOMO model considers these differences in characteristics of the subsystems.
- Detailed COCOMO model - incorporates all characteristics of the intermediate version with an assessment of the cost driver's impact on each step (analysis, design, etc.) of the software engineering process.
- It estimates the effort and development time as the sum of the estimates for the individual subsystems.
- The cost of each subsystem is estimated separately.
- This approach reduces the margin of error in the final estimate.
- For example, a distributed Management Information System (MIS). Because it can have following sub components:
  - Graphical User Interface (GUI) part (organic)
  - Database part (semidetached)
  - Communication part (embedded)
- The costs for these three components can be estimated separately, and summed up to give the overall cost of the system.

Advantages of COCOMO model

- It is easy to use.
- It is repeatable process.
- It is versatile enough to support different modes and levels.
- It works well on projects that are not so different in size, process and complexity.
- It is highly standardized, based on previous experience.
- It can be detailed documented.

Disadvantages of COCOMO model

- COCOMO models are not accurate and not good for scientific justification.
- It ignores software safety issues.
- It ignores personal turnover levels.
- All the levels of COCOMO are dependent on size estimates.
- It also ignores many hardware issues.

❖ SCHEDULING

- Project-task scheduling is an important project planning activity. It involves deciding which tasks would be taken up when.
- Project scheduling is a tool that distributes estimated efforts across the project duration and allocates these efforts to specific tasks.
- In order to schedule the project activities, a software project manager needs to do the following :
  - Identify all the tasks needed to complete the project.
  - Break down large tasks into small activities.
  - Determine the dependency among different activities.
  - Establish the most likely estimates for the time durations necessary to complete the activities.
  - Allocate resources to activities.
  - Plan the starting and ending dates for various activities.
  - Determine the critical path.
- Among all of the above activities: identifying tasks and breaking down them into small activities done through work breakdown structure (WBS). PERT and CPM used to sequence the activity and estimating time and project schedule is developed through Microsoft project tool.

➢ **Work breakdown structure (WBS) :**

- Work Breakdown Structure (WBS) is used to decompose a given task set recursively into small activities.
- The WBS is a uniform, consistent, and logical method for dividing the project into small, manageable components for purposes of planning, estimating, and monitoring.
- An effective WBS encourages a systematic planning process, reduces the omission of key project elements, and simplifies the project by dividing it into manageable units.
- A WBS will provide a roadmap for planning, monitoring, and managing all factors of the project like ' resource allocation, scheduling, budgeting, productivity, performance etc.
- WBS can be shown graphically in a hierarchical tree structure and developed top to bottom manner.
- The root of the tree is labelled by the problem name.
- Each node of the tree is broken down into smaller activities that are made the children of the node.
- Each activity is recursively decomposed into smaller sub-activities until at the leaf level.
- WBS can be done by the decision of the project manager.

Types of WBS:

- There are three types of WBS as follows:

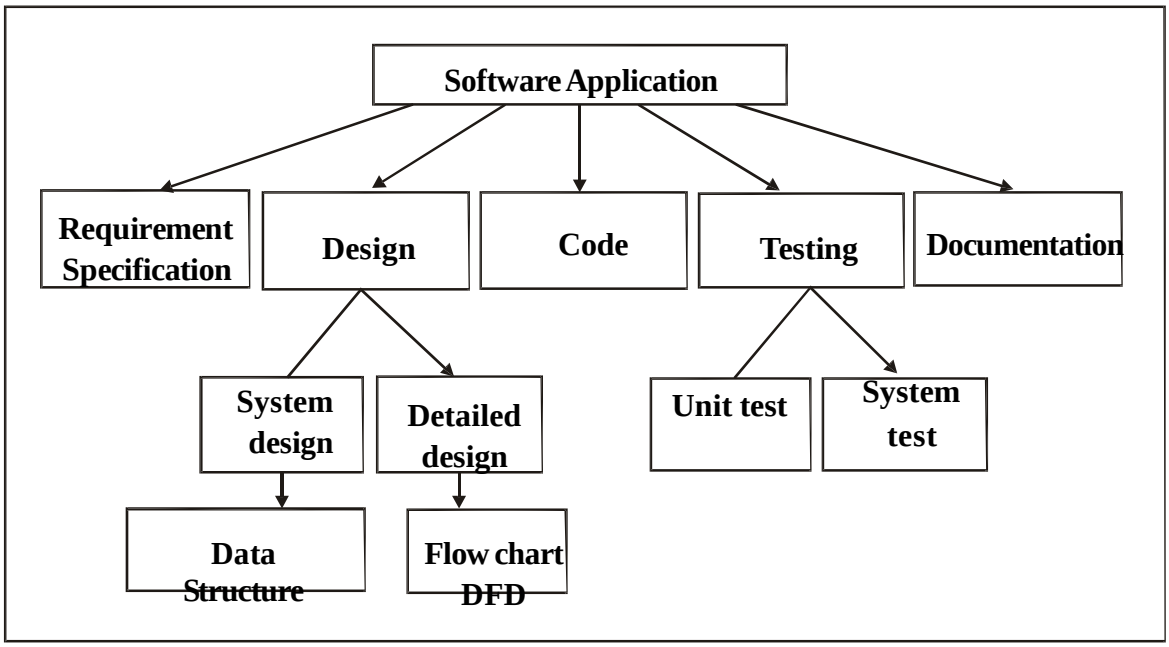
(i) Process WBS: it decomposes large processes into smaller ones. Each process finally decomposed in the task.

(ii) Product WBS: it decomposes large entity into smaller components. It is used by system engineers.

(iii) Hybrid WBS: in includes both process and product elements into single WBS.

- There are two methods of WBS presentation:

1. Tree structure :



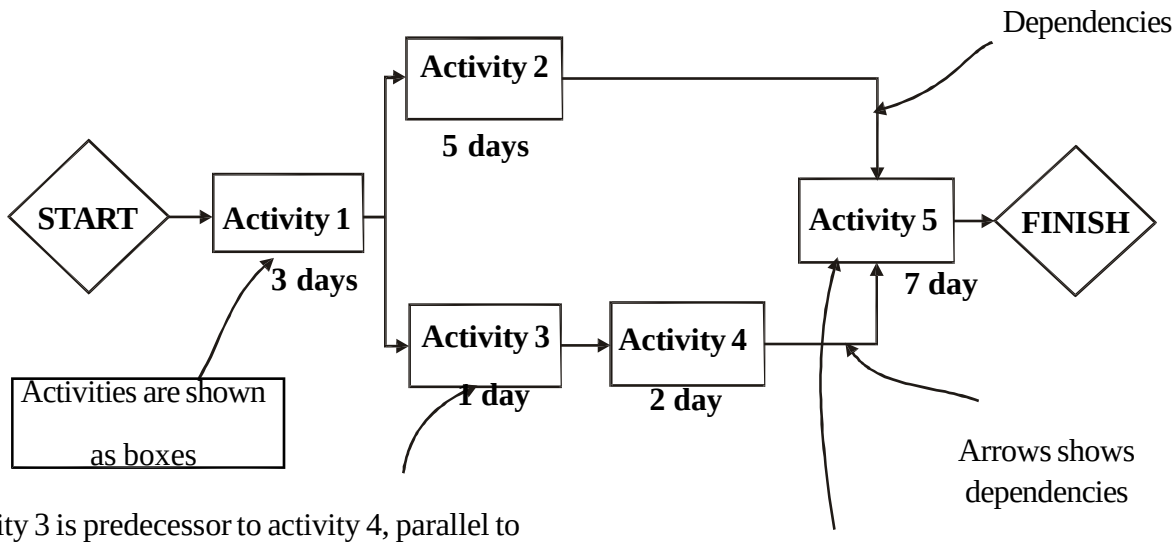
2. Indented list form :

Level	Tasks
0	Top level Process (Product) (Software application)
1	Requirement Specification
2	Design 2.1 System Design 2.1.1 Data structure 2.2 Detailed design 2.2.1 Flow chart, DFD
3	Code
4	Testing 4.1 Unit test 4.2 System test
5	Documentation

Disadvantages of WBS

- It specifies tasks only, not the process by which task is carried out.
- It doesn't specify the person who is doing the task.
- WBS is project specific, so inter project comparison is difficult.
- In WBS, only the list of tasks and deliverables are prescribed, not how the particular work will be completed.
- WBS doesn't provide any sequence or plan for the tasks.

- **Activity network:**
  - In project management, an activity is a task that needs to be accomplished within defined period of time or by deadline to achieve goal.
  - Activities in a project are graphically represented using activity network diagram.
  - Activity network is a network graph using nodes with interconnecting edges to represent tasks and their planned sequence of completion, interdependence and interrelationship that must be accomplished to reach the project goals.
  - An Activity Network Diagram helps to find out the most efficient sequence of events needed to complete any project. It enables you to create a realistic project schedule by graphically showing:
    - The total amount of time needed to complete the project
    - The sequence in which tasks must be carried out
    - Which tasks can be carried out at the same time
    - Which are the critical tasks that you need to keep an eye on.
  - At the time of drawing activity network diagram, each activity is represented by a rectangular node and the duration of the activity is shown alongside each task.



Activity 3 is predecessor to activity 4, parallel to activity 2 and successor to activity 1.

Activity 5 cannot begin until activity 2 and 4 have been completed.

  - The Activity Network diagram displays interdependencies between tasks through the use of boxes and arrows. Arrows pointing into a task box come from its predecessor tasks, which must be completed before the task can start. Arrows pointing out of a task box go to its successor tasks, which cannot start until at least this task is complete.
  - **Activity Network Diagram Drawing Rules:**
    - All the preceding activities must be completed before the project completed.
    - The arrows represent the logical precedence of the project.
    - The task which is dependent on other tasks cannot start until the tasks on which it depended are completed.

Applications

- Activity Networks and PERT (Programming Evaluation and Review Technique) Charts are typically used to document complex projects in a visual manner.
- They are also used to establish the critical path of a project.

➢ **Critical Path Method (CPM):**

- The CPM method was discovered by M.R.Walker in 1957. Critical path method is a network analysis technique.
- Critical path is the sequence of activities with the longest duration. A delay in any activity on this path will result in a delay for the whole project. The activities on the critical path are critical activities.
- CPM used to calculate project completion time.
- CPM used to predict the project duration by finding out sequence of activities has the least amount of scheduling flexibilities.
- The project manager identifies the critical activities of the project.
- CPM deals with both cost and time.
- CPM is used to calculate expected completion time of the project.
- It is based on single time estimation.

Need of CPM

- Planning resource requirements.
- Control resource allocation.
- Prediction of deliverables.
- Internal and external program review.
- Performance evaluation.

Advantages

- It provides clear, concise and unambiguous way of documenting project plans, schedules, time and cost.
- It is mathematically easy and simple.
- It is useful to new project managers.
- It displays dependencies which help in scheduling.
- It determines slack time. Which is the total time for that task may be delayed to complete.
- It can display parallel running activity.
- It is widely used in industry purpose.

Disadvantages

- It is too complex for large projects.
- It doesn't handle the scheduling of people and resource allocation.
- Critical path should be calculated carefully.
- Calculation of estimating the completion time is difficult.
- Activity time estimates are subjective and depend on judgment.

Applications

- Used in construction activities.
- Used in medical and surgical sector.
- Used in transportation activities and oil refineries.

➢ **GANTT chart:**

- It was proposed by Henry Gantt in 1914.
- It is also called time line chart.
- It's mainly used to allocate resources to activities (Resource Planning).
- The resources allocated to activities include staff, hardware, and software etc.
- A Gantt chart is a special type of bar chart where each bar represents an activity.
- It is one of the most popular and useful ways of showing activities displayed against time.
- The bars are drawn along a time line. Activities against time are drawn in bar chart. The length of each bar is proportional to the duration of time planned for the corresponding activity.
- The slack time for a particular task is also shown in the bars.
- The chart is prepared by the project manager.

How to plan GANTT chart

- Identify all the tasks.
- If possible, break down the tasks into smaller tasks.
- Determine the total estimated completion time for each task.
- Plot activities on GANTT chart. Draw milestones at applicable places.

Advantages

- It is very simple to understand and easy to use.
- It is used in monitoring the progress of the project.
- GANTT chart mainly used for resource allocation.
- Useful for planning and guiding projects, understating critical paths & planning resources.

Disadvantages

- It doesn't show interdependencies of activities.
- It doesn't show precedence relationship among activities.
- Not suitable for large projects.
- It can't calculate shortest time for any activity in the project.

Application

- Used in industry to plan and schedule the activities and milestones.

Example (a simple frame work that shows the GANTT chart):

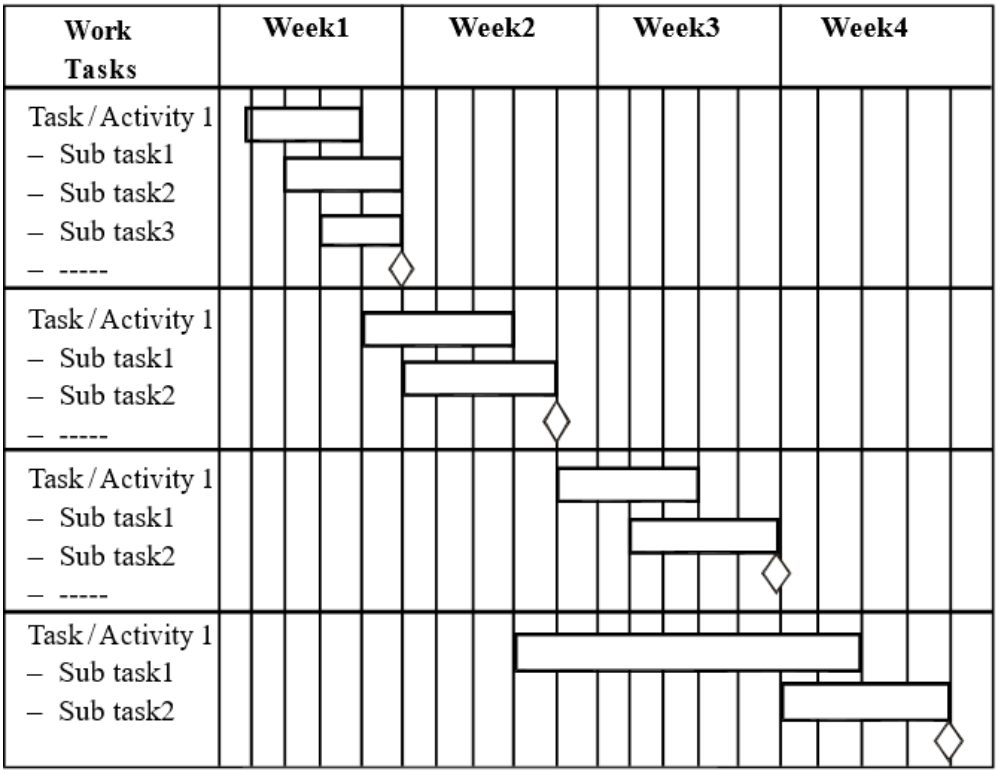


Figure: GANTT chart example

- **Project Monitoring and Control (PMC) :**

- Planning is one of the most important project activities. And without proper planning, project monitoring and control is not possible.
- **Monitoring** - collecting, recording, and reporting information concerning project performance that project manager and others wish to know.
- **Controlling** - it uses data from monitoring activity to bring actual performance from planned performance.
- Project monitoring and planning (PMC) activities take place in parallel with project execution, so the implementation should be corrective at appropriate level.
- The main purpose is to - *to ensure quality of final product.*

Why there is a need of PMC

- Simply because we know that things don't always go according to plan.
- To detect and react appropriately to deviations and changes to plans.

What do we monitor and control

- We need to monitor → men, machine, money, material, space, time, tasks, quality, performance and we need to control' time, cost and performance.
- PERT chart is mainly used for project monitoring and control.

Monitoring and controlling project work includes following activities:

- Comparing the work that is occurring to the project management plan.
- Assessing work performance information to determine if any corrective or preventative actions are necessary.

- Analyzing, tracking, monitoring, and reporting on project risks.
- Providing status reports, accomplishments, and issue reports.
- Monitoring the implementation of approved changes.
- Do scope verification process, schedule control, quality control and cost control.
- Making sure that approved defect repairs have been made.
- Take corrective actions when needed.
- Monitoring and controlling outputs related to risk management include updating the risk register.
- Monitoring and controlling outputs related to communications management include performance reports, forecasts, and resolved issues.
- Techniques used for project monitoring and control activity are: Earned Value Analysis (EVA) and Critical ratio.
- A way of measuring overall performance (not individual task) is using an aggregate performance measure - Earned Value.
- **Earned Value Analysis** (EVA) is an industry standard method of measuring project's progress at any given point of time.
- Earned value of work performed (value completed) for those tasks in progress, found by multiplying the estimated percent physical completion of work for each task by the planned cost for those tasks. The result is amount that should be spent on the task so far. This can be compared with actual amount spent.
- Especially for large projects, it may be worthwhile calculating a set of critical ratios for all project activities
- The critical ratio is :

$$\frac{\textbf{Actual progress}}{\textbf{Scheduled progress}} * \frac{\textbf{Budgeted cost}}{\textbf{Actual cost}}$$

- If ratio is 1 everything is probably on target.
- The further away from 1 the ratio is, the more we may need to investigate.
- Continuous monitoring gives the project management team insight the health of the project, and identifies any areas that can require special attention.

❖ RISK MANAGEMENT

- Tomorrow's problem is today's risk.
- Software risk is a problem that could cause some loss or threaten the success of software project, but which hasn't happened yet.
- This risk may affect negatively to the cost, schedule, technical success or quality of the project.
- Risk management is the process of identifying, addressing and eliminating the problems of risks before they can damage the project.

Objectives of the risk management

- Identify potential problems and deal with them when they are easier to handle before they become critical.
- Focus on the project's objectives and consciously look for things that may affect project quality.
- Allow early identification of risks and provide management decisions to the solutions.
- Increase the chance of project success.

Risk management activities:

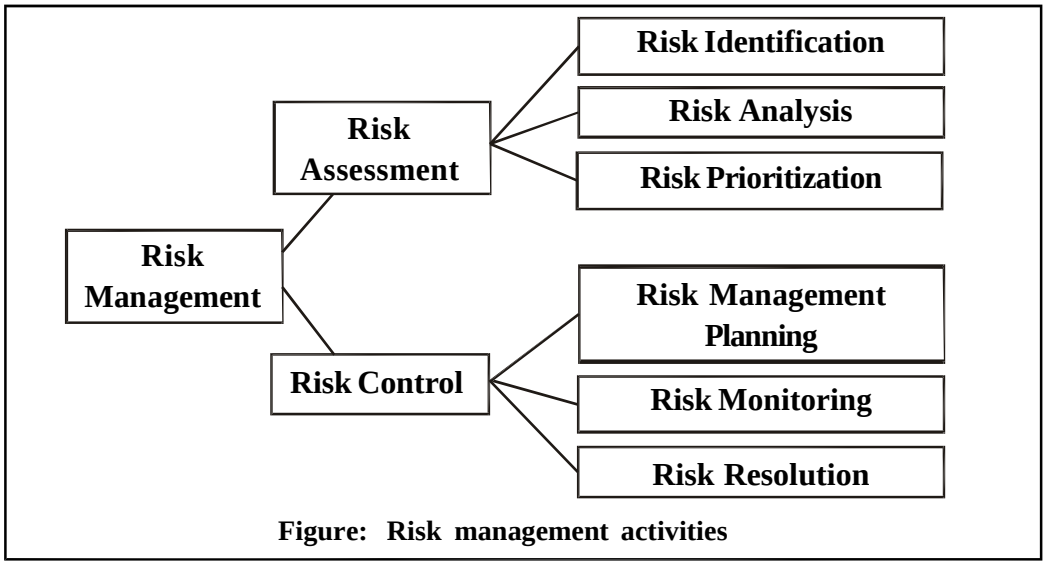


Figure: Risk management activities

Risk assessment:

- It is the process of examining a project and identifying areas of potential risk.
- It includes the following activities:
  - (i) Risk identification
  - (ii) Risk analysis
  - (iii) Risk prioritization

(i) Risk identification

- It is a systematic attempt to specify threats of the project plan.
- The purpose of risk identification is → to develop list of risk items also called risk statement.
- So, some common risks areas are found and checklist is prepared.
- If we want to identify the important risks which may affect the project, it is necessary to categorize risks into different classes.
- The project manager can then examine which risks from each class are relevant to the project.
- There are three main categories of risks which can affect a software project:

1. Project risks

→ Project risks threaten the project. They concerned various forms of budgetary, schedule, personnel, resource, and customer-related problems.

→ An important project risk is schedule slippage. Because it is very difficult to monitor and control this risk.

2. Technical risks

→ Risks that threaten the quality of the product. These risks concern potential design, implementation, interfacing, testing, and maintenance problems.

→ It also covers the specification risks.

→ Most technical risks occur due to the development team's insufficient knowledge about the project.

3. Business risks

→ Risks that threaten the development (or client) organization.

→ This type of risks includes risks of building an excellent product that no one wants, losing budgetary or personnel commitments, etc.

- The output of this activity is the list of risks.

(ii) Risk analysis

- When the risks have been identified, all risks are analyzed using different criteria.
- The purpose of this activity is → to examine how project outcomes might change with modification of risk input variables.
- The input of this activity is the list of risks developed in risk identification and output is the ranking of the risks and further analysis of description and probability of the risks.
- The main activities of this process are:
  - Group similar risks - detect duplicates and group similar risk items.
  - Determine risk drivers - determine risk parameters that affect identified risks.
  - Determine source of risks - find out causes of risks.
  - Estimate risk exposure - doing by measuring the probability and consequences of a risk.
  - Evaluate against criteria - each risk item is evaluated using the predefined criteria, which are important for the specific risk.

(iii) Risk prioritization

- It helps the project focus on its most severe risks by assessing the risk exposure.
- This process can be done in a quantitative way, by estimating the probability (0.1 - 1.0) and relative loss, on a scale of 1 to 10.
- The higher the exposure, the more the risk should be tackled.
- Another way of handling risk is the risk avoidance (do not do risky things).

Risk control:

- Risk control is the process of managing risks to achieve the desired outcomes.
- Risk control activity includes the following: (i) Risk management planning
  (ii) Risk monitoring
  (iii) Risk resolution

(i) Risk management planning

- It produces a plan for dealing with each significant risk.
- In involves identification of strategies to deal with risk. These strategies fall into three categories.
  - Risk avoidance - simply "don't do the risky things". We may avoid risks by not undertaking certain projects.
  - Risk avoidance attempts to reduce the probability of a risk.
  - Risk minimization (risk reduction) - to reduce the impact of the risk.
  - Risk contingency plan - which deals with a risk if it occurs.

(ii) Risk monitoring

- Risk monitoring is the continuous process of reassessing risks as the project proceeds and when the conditions change.
- Projects can be evaluated periodically to manage the risks and reduce their impact on the project.
- Each key risk should be discussed at management progress meetings.

(iii) Risk resolution

- When a risk occurred, it has to be solved. Risk resolution is the execution of the plans for dealing with each risk.
- The project manager has to decide the plan for resolving the risks.
- The input of this activity is → risk action plan and the outputs are:
  - Risk status
  - Acceptable risks
  - Reduced rework
  - Corrective actions
  - Problem prevention