Unit : 3

Software Project Management

3.1 Responsibilities of software project manager.

- Responsible for **accomplishing the stated project objectives**.
- Take the **overall responsibility** of project success.
- The **job responsibility** → Planning to Deployment
- **Bridging gap** between the production team and client.
- **Manage project management triangle** which are Cost, Time, Scope and Quality.
- **General activities:** 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.
- Mainly → project planning and controlling.
- **Risk management.**
- **Time and cost estimation.**

3.1 Responsibilities of software project manager.

- Project manager take the decision that directly affects the benefits of the projects.
- He **sets up development milestones** and entry or exit criteria.

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

- Must have **theoretical knowledge**.
- A good **decision making** capabilities.
- He should be **client representative**.
- Should have **management skill**.
- Focuses on **risk management**.
- He should have **team leadership skill**.
- He should have the **experience** in the related area.
- Monitoring and scheduling capability.

3.1 Responsibilities of software project manager.

- **Evaluating performance** of each milestone
- Some skills like tracking and controlling the progress of the project, customer interaction, managerial presentations, and team building are acquired through experience.

3.2 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.

3.2 Metrics for project size estimation.

- **The project size is a measure of the problem complexity** in terms of the effort and time required to develop the product.
- There are several metrics to measure problem size. Each of them has its own advantages and disadvantages.
- Two important metrics to estimate size: ***Lines of code (LOC)* and *Function point*.**

❖ Lines of Code (LOC)

- Simplest measure, Very popular due to simplicity.
- The project size is estimated by **counting the number of source instructions** in the developed program. (No commonest line)
- To estimate LOC → project manager divides the problem into modules and sub modules, until leaf level modules can be estimated.

3.2 Metrics for project size estimation.

↪ **Disadvantages of LOC:**

- It is **language dependent**.
- LOC **gives only numerical values** of problem size.
- LOC metric **suffer from accuracy**.
- 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.
- LOC **doesn’t work well with non-procedural languages**.

❖ **Function point metric (FP)**

- It 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**.

3.2 Metrics for project size estimation.

- 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**.
- FP is directly dependent on the number of different functions or features it supports.
- A software supporting many features or functions should have larger in size.
- 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).

3.2 Metrics for project size estimation.

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: - - -
- 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:** $FP = UFP * TCF$

3.2 Metrics for project size estimation.

➥ Advantages of FP:

- Not restricted to code	- Language independent
- More accurate than LOC	-We can measure from specification

➥ Disadvantages:

- It **ignores quality** of output.
- It is fully oriented to traditional data processing system.
- **Major shortcoming:** it doesn't take algorithmic complexity into account.
- **To overcome this problem**, an extension of the function point metric called ***feature point metric*** is proposed, which incorporates an extra parameter for algorithm complexity.
- Not applicable universally, not good for real time software.

3.3 Project Estimation Techniques.

- It is one of important activities.
- The **important project parameters** that are estimated include: project size, effort required to develop the software, project duration, and cost.
- There are **three main categories** of project estimation techniques:
  1. Empirical estimation techniques
  2. Heuristic techniques
  3. Analytical estimation techniques

◈ Empirical estimation techniques

- It is **based on making an educated guess** of the project parameters.
- The project managers use their **past experiences** to make guess.

3.3 Project Estimation Techniques.

- There are **two popular empirical estimation techniques:**

I. <u>Expert judgment.</u>

- Most widely used technique.
- In which, an **expert makes an educated guess** of the problem size after analyzing the problem in detail.
- Expert **estimates the costs of different modules (or units)** then combined 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.
- So, the **chance of human error or individual bias** should be there.
- **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 there.**

3.3 Project Estimation Techniques.

II. 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 characteristic of the product/project in their estimation report.
- The coordinator prepares summary and allocate to estimators.
- Based on this summary, the estimators re-estimate.

3.3 Project Estimation Techniques.

- 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, coordinators prepares the final estimate.

◈Heuristic estimation techniques.

- **Mathematical techniques are used** in this techniques.
- Concept of ***Independent and dependent variables.***
- Two classes ***static single variable model*** and ***static multi variable model.***
- A single variable estimation model takes the following form:

$$\textbf{Estimated Parameter} = \mathbf{c1 * e^{d1}}$$

- Example → basic COCOMO model.

3.3 Project Estimation Techniques.

- The multivariable cost estimation model takes the following form:

  Estimated Parameter = $c1*e1^{d1} + c2*e2^{d2} + \ldots$

- It gives more accurate estimation compare to single variable model.
- Example → intermediate COCOMO model.
- Boehm classified a software into one of the following three categories based on the development complexity:

  I. Organic

  II. Semi detached

  III. Embedded

- This classification done using characteristics of the product, development team and development environment.

3.3 Project Estimation Techniques.

- 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.

➥ **Organic**

- Small group of team.
- Develop **well understood application** programs and having experienced developers.
- Little or **no innovation**.
- Project size should be **2-50 KLOC**.
- **For example** data processing programs, payroll, inventory management system.

3.3 Project Estimation Techniques.

➥ Semidetached

- Consists of a **mixture of experienced and inexperienced staff**.
- Team members may have **limited experience.**
- **Medium innovation**.
- Project size should be **50-300 KLOC**.
- **For example** compilers, interpreters, assemblers.

➥ Embedded

- Strongly coupled to **complex hardware**.
- Tight or inflexible (rigid) regulations.
- Great deal with **innovation** and requirements constraints, and strict deadlines.
- Large project size and development team, few experience.
- Project size is approximately **over 300 KLOC**.
- **Example** → air traffic control, ATMs, Real time operating systems
- Brooks stated 1 : 3 : 9 ratio for all three categories.

3.3 Project Estimation Techniques.

COCOMO (A heuristic estimate techniques)

- **COCOMO *(COnstructive COst estimation MOdel)*** was proposed by Boehm.
- It is ***regression based model*** provides hierarchy of software cost estimation models.
- ***Three stages:*** *Basic COCOMO, Intermediate COCOMO, and complete COCOMO.*
- Each level achieves **successive accuracy.**

❖ Basic COCOMO model

- **Single valued model**, computing s/w development efforts and time.
- It is ***quick and rough*** estimation technique.
- It gives an ***approximate estimate*** of the project parameters.

3.3 Project Estimation Techniques.

- The basic COCOMO estimation model is given by the following expressions:

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

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

✓ ***Estimation of development effort in basic COCOMO model***

Organic	:	$\text{Effort (E)} = 2.4\ (\text{KLOC})^{1.05}$	PM
Semidetached	:	$\text{Effort (E)} = 3.0\ (\text{KLOC})^{1.12}$	PM
Embedded	:	$\text{Effort (E)} = 3.6\ (\text{KLOC})^{1.20}$	PM

✓ ***Estimation of development time in basic COCOMO model***

Organic	:	$T_{dev} = 2.5\ (\text{effort})^{0.38}$	Months
Semidetached	:	$T_{dev} = 2.5\ (\text{effort})^{0.35}$	Months
Embedded	:	$T_{dev} = 2.5\ (\text{effort})^{0.32}$	Months

3.3 Project Estimation Techniques.

- 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.

❖ Intermediate COCOMO model.

- The basic COCOMO model considers efforts and time development are the alone functions of product size.
- But some factors from other projects may also affect the efforts and time.
- The intermediate COCOMO model doing this by using a set of 15 cost drivers (multipliers) based on various attributes of software development.
- The Intermediate COCOCMO model consumes these "cost drivers"

3.3 Project Estimation Techniques.

- 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.

- Product attributes
- Computer attributes
- Personal attributes
- Project attributes

❖ Complete COCOMO model

- It is also known as **detailed 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.

3.3 Project Estimation Techniques.

- For example, some subsystems may be considered as organic type, some semidetached, and some embedded.
- And these subsystems may be differ in requirements, development complexities and may not have previous experience of similar developing.
- 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) consists of different categories of the software.

3.3 Project Estimation Techniques.

➥ Advantages of COCOMO model

- It is easy to use.
- It is repeatable process.
- It is versatile enough.
- 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

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

3.3 Project Estimation Techniques.

◈Analytical estimation techniques

- 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.**
- **Example** → Halstead's software science
- In s/w estimation prediction, this technique is better than both empirical and heuristic techniques.

3.4 Scheduling

- It is important project planning activity.
- 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**.

3.4 Scheduling

❖ Work breakdown structure (WBS)

- 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.**
- **Provides systematic planning**, key project elements and simplifies the project by dividing into small manageable units.
- **Provide roadmap for** → resource allocation, scheduling, budgeting, productivity, performance etc.
- WBS can be **graphically shown in a hierarchical tree structure.**
- In which, root node is labeled by a problem name. Each node of the tree is broken down into smaller activities that are made the children of the node.
- WBS can be done by the **decision of the project manager.**

3.4 Scheduling

➥ **Types of WBS**

- ***Process WBS:*** decomposes large processes into smaller ones.
- ***Product WBS:*** decomposes large entities into smaller ones.
- ***Hybrid WBS:*** in includes both process and product elements into single WBS.
- There are two methods of WBS presentation:

✓ **Tree structure**

✓ **Indented list form**

➥ **Disadvantages**

- It **specifies tasks only**, not the process.
- It **doesn't specify the person** who is doing the task.
- WBS is **project specific**, so inter project comparison is difficult.
- Not describe *how the particular task will be completed.*
- WBS **doesn't provide any sequence** or plan for the tasks.

3.4 Scheduling

❖ Activity network

- 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.
- It find out the most efficient sequence of events needed to complete a project.
- **Activity diagram 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.

3.4 Scheduling

- In it, each activity is represented by a rectangular node and the duration of the activity is shown alongside each task.
- Figure:
- Activity Network Diagram Drawing Rules:

⮩ **Application**

- Used to **document complex projects** in a visual manner.
- Used to **establish the critical path** of a project.

❖ **Critical Path Method (CPM)**

- It is network analysis technique was discovered by M.R.Walker in 1957.
- Critical path is the **sequence of activities with the longest duration and critical activities.**
- CPM used to **calculate project completion time**.
- **The project manager** identifies the critical path for the project.

3.4 Scheduling

- CPM deals with **both cost and time**.
- 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 of CPM**

- 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.

3.4 Scheduling

- It displays dependencies which help in scheduling.
- It determines slack time.
- It can display parallel running activity.
- It is widely used in industry purpose.

➥ **Disadvantages of CPM**

- **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.

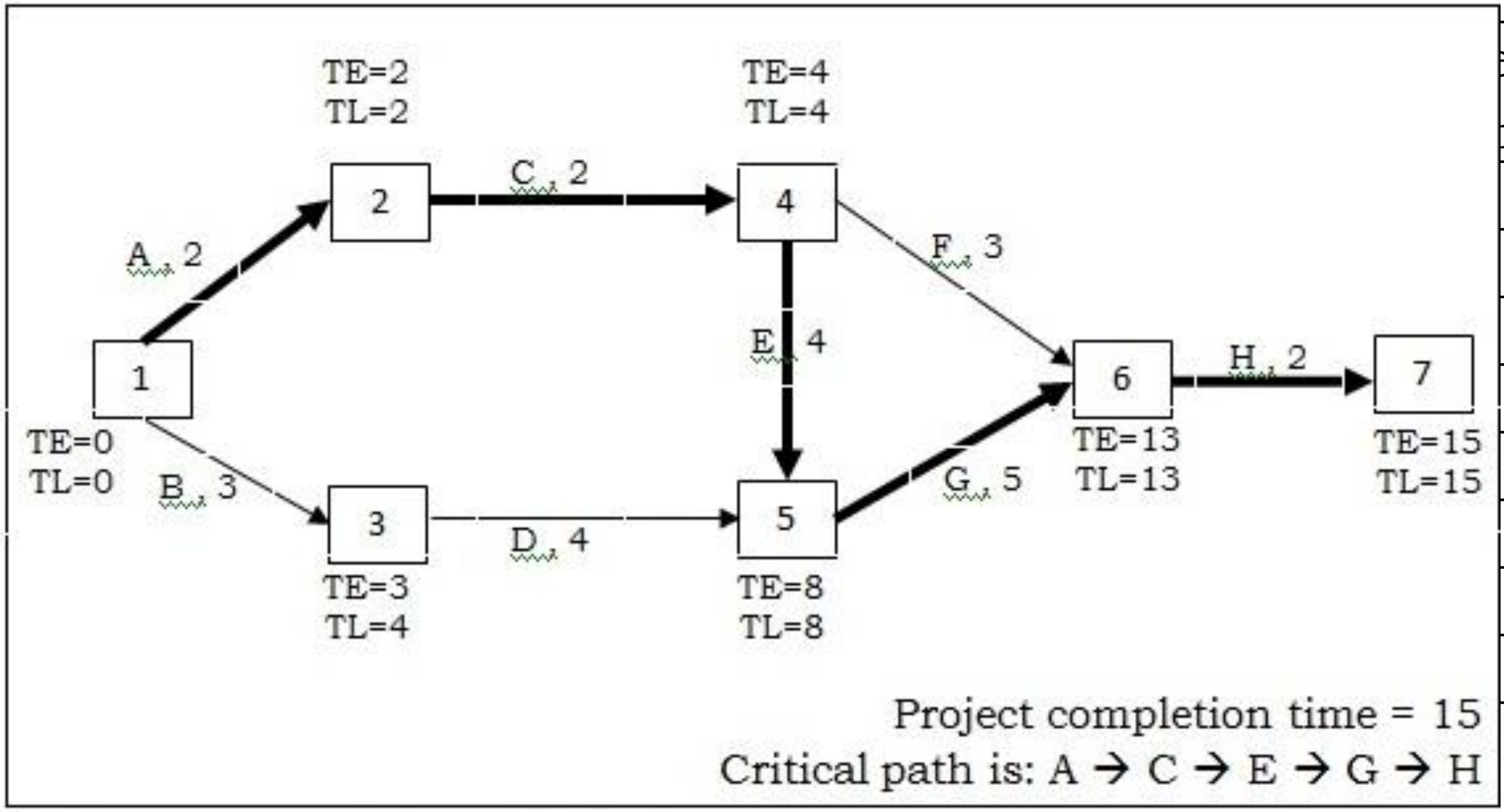
➥ **Application**

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

3.4 Scheduling

➥ **Example.**

- Find critical path (project completion time) for the air pollution control system having different activities listed below:



Project completion time = 15

Critical path is: A → C → E → G → H

3.4 Scheduling

❖ GANTT Chart

- It was proposed by Henry Gantt in 1914, also called **time line chart.**
- It is 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. Bars are drawn along time line. Length of bar proportional to the duration of time planned.
- It shows **activities against time**.
- **Slack time** is also shown in the bars.
- The chart is **prepared by the project manager**.

3.4 Scheduling

↪ **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.**

- Simple to understand and easy to use.
- Useful for planning and guiding projects, understating critical paths & planning resources.
- It is used in monitoring the progress of the project.

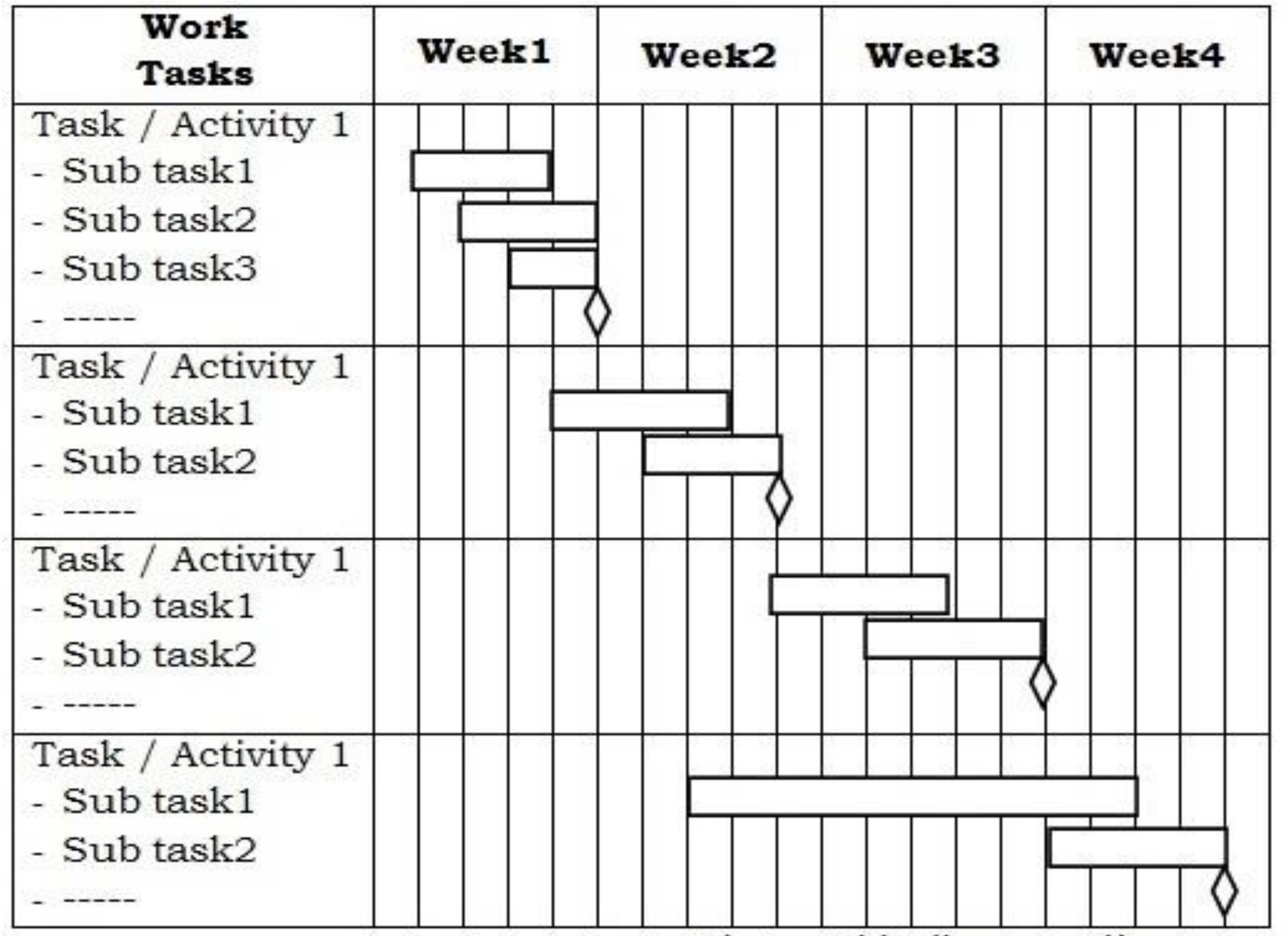
↪ **Disadvantages.**

- It doesn't show interdependencies and precedence of 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.

3.4 Scheduling

➥ **GANTT chart example:**



Diamond indicates milestones.

GANTT chart example.

3.4 Scheduling

❖ **Project monitoring and control.**

- Planning is one of the most important activities of project.
- ***Monitoring*** – collecting, recording, and reporting information
- ***Controlling*** – uses data from monitor 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 make sure that the project plan is being followed and also forecast future performance.

↪ **Need of PMC**

- To detect and react appropriately to deviations and changes to plans.

3.4 Scheduling

➥ **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 used** for PMC.

➥ **PMC activities:**

- Comparing the work.
- Assessing work performance.
- Analyzing, tracking, monitoring, and reporting on project risks.
- Providing reports.
- Monitoring the implementation of approved changes.
- Do scope verification, quality and cost control.
- Ensuring of defect repairs.
- Take correct actions when needed.

3.4 Scheduling

- Output of PMC update risk register.
- **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*.
- EVA particularly used for large projects.
- The critical ratio is: $\frac{\text{Actual progress}}{\text{Scheduled progree}} * \frac{\text{Budgeted cost}}{\text{Actual cost}}$
- If ratio is 1 everything is probably on target.
- The further away from 1 the ratio is, the more we may need to investigate.

3.5 Risk Management

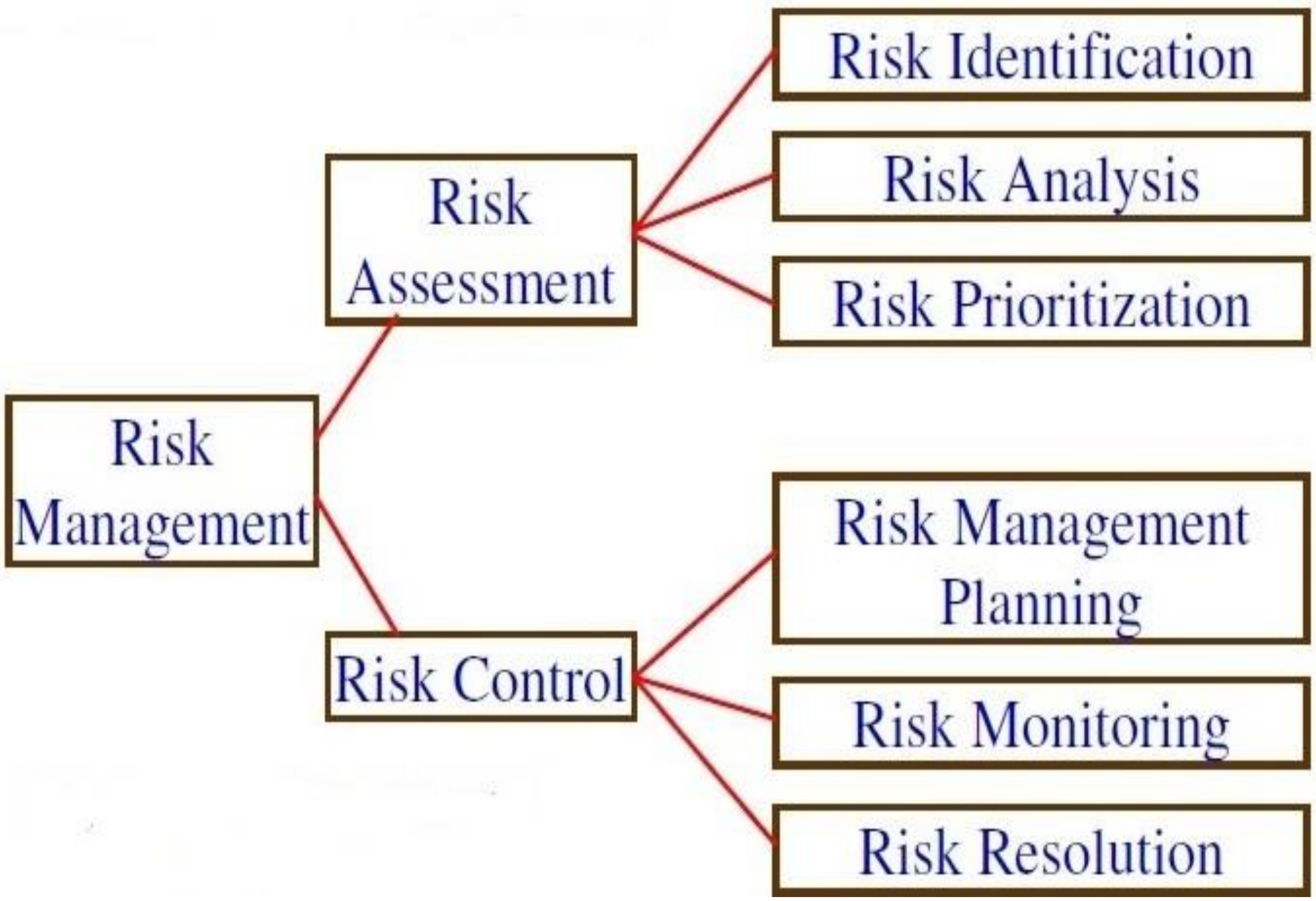
- ***Tomorrow's problem is today's risk.***
- ***Software risk*** is a problem that could cause some loss or threaten the success of software project, but which hasn't happened yet.
- This risk may **affect negatively** to the cost, schedule, technical success or quality of the project.
- **Risk management** is the process of identifying, addressing and eliminating these problems 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 after.
- **Allow early identification of risks** and provide management.
- Increase the chance of project success.

3.5 Risk Management

➥ **Risk management activities**



3.5 Risk Management

❑ **Risk assessment**

- It is the process of examining a project and identifying areas of potential risk.
- It includes the following activities:
  - **Risk identification**
  - **Risk analysis**
  - **Risk prioritization**

✓ **Risk identification**

- It is a systematic attempt to specify threats of the project plan.
- **Purpose** → to develop list of risk items also called *risk statement*
- Some common risks areas are found and checklist is prepared.
- *Categorize risks* into different classes.
- The project manager can then examine which risks from each class are relevant to the project.

3.5 Risk Management

- There are three main categories of risks:

I. ***Project risks:*** budgetary, schedule, personnel, resource etc. Schedule slippage.

II. ***Technical risks:*** Risks that threaten the quality of the product. design, implementation, interfacing, testing, and maintenance problems etc.
   - Most technical risks occur due to improper knowledge.

III. ***Business risks:*** Risks that threaten the development (or client) organization. Include risks of building an excellent product that no one wants, losing budgetary or personnel commitments, etc.

✓ **Risk analysis**

- When the risks have been identified, all risks are analyzed using different criteria.

3.5 Risk Management

- **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.
- Main activities of this process are:
  - Group similar risks
  - Determine risk drivers
  - Determine source of risks
  - Estimate risk exposure
  - Evaluate against criteria

✓ **Risk prioritization**

- It **focuses on its most severe risks** by assessing the risk.
- Risks are **estimated in probability** of (0.1 – 1.0).
- The higher the exposure, the more the risk should be tackled.
- Another way is → **risk avoidance** (don't do risky things).

3.5 Risk Management

❑ Risk Control

- It is the process of managing risks to achieve the desired outcomes.
- It includes:
    - **Risk management planning**
    - **Risk monitoring**
    - **Risk resolution**

✓ ***Risk management planning***

- Provides plan to deal with risk.
- Identify different strategies, like:
    - Risk avoidance (don’t do risky things)
    - Risk minimization [risk reduction] (reduce impact of risks)
    - Risk contingency plan (deals with risks if it occurs)

3.5 Risk Management

✓ ***Risk monitoring.***

- It is the continuous process of reassessing risks when project get changed.
- Projects can be evaluated periodically to manage the risks.
- Each key risk should be discussed at top level.

✓ ***Risk resolution.***

- Risk resolution is the execution of the plans for dealing with each risk.
- Decided by project manager.
- The input of this activity is → risk action plan
- And out puts are→

- Risk status	- Acceptable risks	- Reduced rework
- Corrective actions	- Problem prevention	

Thank you…