Lecture:23

- **Reference to Previous Topics**

I. Overview of Management

II. Software Developmental Fundamentals

III. Process

IV. **Planning**

- **Planning is one of the most important management activities and is an on-going effort throughout the life of the project.**
- **Software project management begins with a set of activities that are collectively called Project Planning.**
- **Preliminary planning starts on day one even in the pre-project phase**
- **It should not be conducted "in secret"**
- **It needs buy-in and approval**

1. **PROJECT PLANNING**

1.1 Planning Objectives

1.2 Project Planning - Definition

1.3 Key Tasks

1.4 Project Management Process

1.5 Planning Puzzle

1.6 Primary Planning Steps

- **1.1 Project Planning Objectives**
- **The objective of software project planning is to provide a framework that enables the manager to make reasonable estimates of:**
    - **Resources**
    - **Cost, and**
    - **Schedule**

- **These estimates are made within a limited time frame at the beginning of a software project and should be updated regularly as the project progresses.**
- **In addition, estimates should attempt to define best case and worst-case scenarios so that project outcomes can be bounded.**
- **1.1 Project Planning Objectives**
- **Planning is one of the most important management activities and is an ongoing effort throughout the life of the project.**
- **Software project management begins with a set of activities that are collectively called Project Planning.**
- **The software project planner must estimate following things before a project begins:**
    - **How much will it cost?**
    - **How long will it take?**
    - **How many people will it take?**
    - **What might go wrong?**
- **1.2 Project Planning - Definition**
- **What is it?**

Software project planning involves estimation - your attempt to determine how much money, how much effort, how many resources, and how much time it will take to build a specific software-based system or product.

- **Who does it?**

Software Project Managers- Using information solicited from customers and software engineers and software metrics data collected from past projects.

- **1.2 Project Planning - Definition**
- **Why is it important?**

Would you build a house without knowing how much you were about to spend? Of course not, and since most computer-based systems and products cost considerably more to build than a large house, it would seem reasonable to develop an estimate before you start creating the software.

- **1.2 Project Planning - Definition**

What are the steps?

- **Estimation begins with a description of the scope of the product.**
- **Until the scope is “bounded" it's not possible to develop a meaningful estimate.**
- **The problem is then decomposed into a set of smaller problems and each of these is estimated using historical data and experience as guides.**
- **1.2 Project Planning - Definition**

What are the steps?

- **It is advisable to generate your estimates using at least two different methods (as a cross check).**
- **Problem complexity and risk are considered before a final estimate is made.**
- **1.2 Project Planning - Definition**

What is the work product?

- **A Simple table *delineating***
  - **the tasks to be performed,**
  - **the functions to be implemented and**
  - **the cost, effort and time involved for each is generated.**
- **A list of required project resources is also produced**
- **1.2 Project Planning - Definition**

How do I ensure that I've done it right?

- **That’s hard, because you won't really know until the project has been completed.**
- **However, if you have experience and follow a**
  - **Systematic approach,**
  - **Generate estimates using solid historical data,**
  - **Create estimation, data points using at least two Different methods, and**
  - **Factor in complexity and risk,**

you can feel confident that you've done a right job to achieve the targets.

- **1.3 Project Planning: Key Tasks**

- **1.4 Project Management Process**
- **Why**
  - **Deliverable**
- **What**
  - **SOW, Requirements**
- **How**
  - **Design Specification, SDP, Lifecycle**
- **Do**
  - **Execution, Control**
- **Done**
  - **Project Closing and Completion Report**
- **1.5 Planning Puzzle**
- **1.6 Primary Planning Steps**

1. Project Plan: Outline the activities, tasks, dependencies and timeframes
2. Resource Plan: List the labor, equipment and materials required
3. Financial Plan: Identify the labor, equipment and materials costs

- **1.6 Primary Planning Steps**

4. Quality Plan: Provide quality targets, assurance and control measures
5. Risk Plan: Highlight potential risks and actions taken to mitigate them
6. Acceptance Plan: List the criteria to be met to gain customer acceptance

- **1.6 Primary Planning Steps**

7. Communications Plan: List the information needed to inform stakeholders
8. Procurement Plan: Identify products to be sourced from external suppliers.

   (At this point the project has been planned in detail and is ready to be executed)

2.1 Information needs

2.2 Steps/items required

2.3 Inputs to SDP

2.4 Tools and Techniques for SDP

2.5 Outputs from SDP

2.6 SDP Execution

2.7 Inputs to SDP execution

2.8 Tools and Techniques for SDP execution

2.9 Outputs from SDP execution

2.10 Elements of SDP

- **The project development plan is one of the first formal documents produced by the project. Within this document, the project manager describes in detail:**

    1. **How the project will be developed?**
    2. **What resources will be required?**
    3. **How these resources will be used?**

- **2.1 SDP Information needs**

1. **Introduction and status of the plan**
2. **The authorisation procedures**
3. **Statement of project objectives**
4. **Statement of requirement**
5. **Deliverables in the project**
6. **A Work Breakdown Structure**
7. **The project milestones**

- **2.1 SDP Information needs**

8. **The resource requirements**
9. **Interdependencies of work**
10. **The timetable of events**

11. **Staffing, organisation and responsibilities**
12. **Development methods and toolsets to be used**
13. **Source documentation**
14. **Resource and financial summary**

- **2.2 SDP Steps/items required**

1. **System overview**
    - **Situation Audit**
    - **Gap analysis**
2. **Software development management**
    - **Project organization and resources**
    - **Development facilities**
    - **Project organizational structure**
    - **Personnel**

- **2.2 SDP Steps/items required**

3. **Schedule and milestones**
    - **Scheduled activities**
    - **Milestones and baselines**
    - **Activity network diagrams**
    - **System component source**
    - **Budget administration**
    - **Milestone payments**
    - **Major budgetary expenditures**
    - **Expenditure authorization procedure**

- **2.2 SDP Steps/items required**

4. **Risk analysis**

5. **Security**
6. **Interface with external sources**
7. **Procedure for formal reviews**
8. **Corrective action process**
9. **Problem change report**

- **2.2 SDP Steps/items required**

10. **Software engineering**
    - **Standards and procedures**
    - **Development methodology**
    - **Development resources**
    - **Personnel -qualifications and function**
11. **Testing procedure**
12. **Software configuration management**
13. **Software quality assurance**

- **2.3 Inputs to SDP**

a) **Other planning outputs.**

- **All of the outputs of the planning processes in the other knowledge areas are inputs for developing the project plan.**

(Other planning outputs include both base documents, such as the WBS, and the supporting detail.)

- **Many projects will also require application area-specific inputs (e.g., most major projects will require a cash-flow forecast).**

- **2.3 Inputs to SDP**

b) **Historical information.**

- **The available historical information (e.g., estimating data-bases, records of past project performance) should have been consulted during the other project planning processes.**

- **This information should also be available during project plan development to assist with verifying assumptions and assessing alternatives that are identified as part of this process.**

- **2.3 Inputs to SDP**

c) **Organizational policies.**

- **Any and all of the organizations involved in the project may have formal and informal policies whose effects must be considered.**

- **2.3 Inputs to SDP**

- **Organizational policies that typically must be considered include, but are not limited to:**
  - **Quality management —process audits, continuous improvement targets**
  - **Personnel administration —hiring and firing guidelines, employee performance reviews**
  - **Financial controls —time reporting, required expenditure and disbursement reviews, accounting codes, and standard contract provisions.**

- **2.3 Inputs to SDP**

d) **Constraints.**

- **A constraint is an applicable restriction that will affect the performance of the project.**

Example, a predefined budget is a constraint that is highly likely to limit the team's options regarding scope, staffing, and schedule.

When a project is performed under contract, contractual provisions will generally be constraints.

- **2.3 Inputs to SDP**

e) **Assumptions.**

- **Assumptions are factors that, for planning purposes, are considered to be true, real, or certain.**

- **Assumptions affect all aspects of project planning, and are part of the progressive elaboration of the project.**

- **2.3 Inputs to SDP**

e) **Assumptions.**

- **Project teams frequently identify, document, and validate assumptions as part of their planning process.**

Example: If the date that a key person will become available is uncertain, the team may assume a specific start date. Assumptions generally involve a degree of risk.

1. **PROJECT PLANNING**
    - **1.1 Planning Objectives**
    - **1.2 Project Planning - Definition**
    - **1.3 Key Tasks**
    - **1.4 Primary Planning Steps**
    - **1.5 Project Management Process**
    - **1.6 Planning Puzzle**
2. **SOFTWARE DEVELOPMENT PLAN (SDP)**
    - **2.1 Information needs**
    - **2.2 Steps/items required**
    - **2.3 Inputs to SDP**

Lecture:24

- **Reference to Previous Session**

1. **PROJECT PLANNING**

1.1 Planning Objectives

1.2 Project Planning - Definition

1.3 Key Tasks

1.4 Primary Planning Steps

1.5 Project Management Process

1.6 Planning Puzzle

- **Reference to Previous Session**

2. **SOFTWARE DEVELOPMENT PLAN (SDP)**

2.1 Information needs

2.2 Steps/items required

2.3 Inputs to SDP

- **2.3 Inputs to SOFTWARE DEVELOPMENT PLAN (SDP)**
- **2.3 Inputs to SOFTWARE DEVELOPMENT PLAN (SDP)**

a) **Other planning outputs**

b) **Historical information**

c) **Organizational policies**

d) **Constraints**

e) **Assumptions**

- **2.4 Tools and Techniques for SDP**
- **2.4 Tools and Techniques for SDP**

a) **Project planning methodology.**

- **A project planning methodology is any structured approach used to guide the project team during development of the project plan.**
- **It may be as simple as standard forms and templates (whether paper or electronic, formal or informal) or as complex as a series of required simulations (e.g., Monte Carlo analysis of schedule risk).**

- **2.4 Tools and Techniques for SDP**

a) **Project planning methodology (Cont.)**

- **Most project planning methodologies make use of a combination of "hard" tools, such as project management software, and "soft" tools, such as facilitated startup meetings.**
- **2.4 Tools and Techniques for SDP**

b) **Stakeholder skills and knowledge.**

- **Every stakeholder has skills and knowledge that may be useful in developing the project plan.**

- **2.4 Tools and Techniques for SDP**

b) **Stakeholder skills and knowledge.**

- **The project management team must create an environment in which the stakeholders can contribute appropriately.**

Who contributes

What they contribute

When they contribute

- **2.4 Tools and Techniques for SDP**
- **Example:**

On a construction project being done under a lump-sum contract, the professional cost engineer will make a major contribution to the profitability objective during proposal preparation when the contract amount is being determined.

- **2.4 Tools and Techniques for SDP**
- **On a project where staffing is defined in advance, the individual contributors may contribute significantly to meeting cost and schedule objectives by reviewing duration and effort estimates for reasonableness.**
  - **Project Management Information System (PMIS)**
  - **Earned Value Management (EVM)**
- **2.4 Tools and Techniques for SDP**
  - **PMIS. It consists of the tools and techniques used to:**
    - **gather**
    - **integrate**
    - **disseminate the outputs**

It is used to support all aspects of the project from initiating through closing, and can include both manual and automated systems.

- **2.4 Tools and Techniques for SDP**
- **Earned Value Management (EVM).**

A technique used to integrate the project's:

- **scope**
- **schedule**
- **resources**
- **measure and report project performance**

- **2.5 Outputs from SDP**
- **2.5 Outputs from SDP**
- **The project plan is a formal, approved document used to manage project execution.**
- **The project schedule lists planned dates for performing activities and meeting milestones identified in the project plan.**
- **2.5 Outputs from SDP**
- **The project plan and schedule should be distributed as defined in the communications management plan (e.g., management of the performing organization may require broad coverage with little detail, while a contractor may require complete details on a single subject).**
- **In some application areas, the term integrated project plan is used to refer to this document.**
- **2.5 Outputs from SDP**
- **A clear distinction should be made between the project plan and the project performance measurement baselines.**
- **The project plan is a document or collection of documents that should be expected to change over time as more information becomes available about the project.**
- **The performance measurement baselines will usually change only intermittently and then generally only in response to an approved scope of work or deliverable change.**
- **2.6 SDP Execution**
- **2.6 SDP Execution**
- **Project plan execution is the primary process for carrying out the project plan the vast majority of the project's budget will be expended in performing this process.**
- **In this process, the project manager and the project management team must coordinate and direct the various technical and organizational interfaces that exist in the project.**

- **2.6 SDP Execution**
- **It is the project process that is most directly affected by the project application area in that the product of the project is actually created here.**
- **Performance against the project baseline must be continuously monitored so that corrective actions can be taken based on actual performance against the project plan.**
- **Periodic forecasts of the final cost and schedule results will be made to support the analysis.**
- **2.7 Inputs to SDP Execution**
- **2.7 Inputs to SDP Execution**
- **Project plan: The subsidiary management plans (scope management plan, risk management plan, procurement management plan, configuration management plan, etc.) and the performance measurement baselines are key inputs to project plan execution.**
    - **2.7 Inputs to SDP Execution**
    - **Supporting detail**
    - **Organizational policies: Any and all of the organizations involved in the project may have formal and informal policies that may affect project plan execution.**
- **2.7 Inputs to SDP Execution**
    - **Preventive action: Preventive action is anything that reduces the probability of potential consequences of project risk events.**
- **2.7 Inputs to SDP Execution**
    - **Corrective action: Corrective action is anything done to bring expected future project performance in line with the project plan. Corrective action is an output of the various control processes—as an input here it completes the feedback loop needed to ensure effective project management.**
- **2.8 Tools and Techniques for SDP Execution**
- **2.8 Tools and Techniques for SDP Execution**
- **General management skills:** General management skills such as leadership, communicating, and negotiating are essential to effective project plan execution.
- **Product skills and knowledge:** The project team must have access to an appropriate set of skills and knowledge about the project's product.
- **2.9 Outputs from SDP Execution**

- **2.9 Outputs from SDP Execution**
- **Work results: Work results are the outcomes of the activities performed to accomplish the project.**
- **Information on work results: which deliverables have been completed and which have not, to what extent quality standards are being met, what costs have been incurred or committed, etc.—is collected as part of project plan execution and fed into the performance reporting process.**
- **2.9 Outputs from SDP Execution**
- **It should be noted that although outcomes are frequently tangible deliverables such as buildings, roads, etc., they are also often intangibles such as people trained who can effectively apply that training.**
- **2.9 Outputs from SDP Execution**
- **Change requests: Change requests (e.g., to expand or contract project scope, to modify cost [budgets], or schedule estimates [dates, etc.]) are often identified while the work of the project is being done.**
- **Summary**
- **Summary**

2. **SOFTWARE DEVELOPMENT PLAN (SDP)**

2.4 Tools and Techniques for SDP

2.5 Outputs from SDP

2.6 SDP Execution

2.7 Inputs to SDP Execution

2.8 Tools and Techniques for SDP Execution

2.9 Outputs from SDP Execution

Lecture:25

- **Reference to Previous Session**
- **Reference to Previous Session**

1. **PROJECT PLANNING**
2. **SOFTWARE DEVELOPMENT PLAN (SDP)**

2.4 Tools and Techniques for SDP

2.5 Outputs from SDP

2.6 SDP Execution

2.7 Inputs to SDP Execution

2.8 Tools and Techniques for SDP Execution

2.9 Outputs from SDP Execution

- **2.10 Elements of SDP**
- **2.10 Elements of SDP**
- Project Plan is iteratively defined through Concept & Requirements Phase
- Initial estimates are refined as scope and requirements become clearer
- Two phases of project plan
  - **Preliminary**
  - **Final**
- **2.10 Elements of SDP**

a) **Scope Planning**

i. **Scope planning is the process of progressively elaborating and documenting the project work (project scope) that produces the product of the project.**

ii. **Project scope planning starts with the initial inputs of product description, the project charter, and the initial definition of constraints and assumptions.**

iii. **Note that the product description incorporates product requirements that reflect agreed-upon customer needs and the product design that meets the product requirements.**

iv. **The outputs of scope planning are the scope statement and scope management plan, with the supporting detail.**

v. **The scope statement forms the basis for an agreement between the project and the project customer by identifying both the project objectives and the project deliverables.**

vi. **Project teams develop multiple scope statements that are appropriate for the level of project work decomposition.**

- **2.10 Elements of Project Plan**
- **2.10 Elements of Project Plan**

i. Could be “Mission Statement”

ii. Defines Business objectives for project

iii. Includes Business Case (NPV Model) & Detailed Description of assumptions.

- 2.10 **Elements of Project Plan**
- 2.10 **Elements of Project Plan**
- **Description of how new system is to be developed**
  - **Technologies**
  - **In House vs. Consultants**
  - **Derivatives of existing (i.e. use existing object model)**
  - **Architectural Layout - Layers**
- 2.10 **Elements of Project Plan**
- 2.10 **Elements of Project Plan**
- **Specifies general needs from outside sources**
  - **Consultants**
  - **Software Suppliers**
  - **Hardware Suppliers**
  - **Network/Infrastructure Suppliers**
  - **There are two types of contracts**
    1. **Cost -Plus**
    2. **Fixed Price**
- **Most other relationships are some kind of combination of these two**
  - 2.10 **Elements of Project Plan**
- **Cost-plus (also called Time and material)**

- **Cost-plus is a contractual relationship where the developer is paid for the cost of the service provided and in addition is allowed an agreed profit margin.**
- **This is rather like renting a car the customer pays for the time that the car is used (by the hour, day, week etc.), and for any other expenses such as insurance and gasoline.**
- 2.10 **Elements of Project Plan**
- **Fixed price**
- **A fixed price contract is a commitment by the developer to provide an agreed product or service for an agreed fee, within an agreed schedule.**
- **This is similar to purchasing a bus ticket, when the bus company agrees to take the customer to a specific destination within a published timetable, and for an agreed fee.**
- 2.10 **Elements of Project Plan**
- 2.10 **Elements of Project Plan**
- Defines specific dates for milestones components
- Work Breakdown Structure
- Use Scheduling Engines like MS project etc.
- 2.10 **Elements of Project Plan**
- 2.10 **Elements of Project Plan**
- Resources such as licenses, servers, or other software/hardware related items
- Personnel – how many people of what type and for how long

(Remember these are initial estimates in preliminary project plan)

- 2.10 **Elements of Project Plan**
- 2.10 **Elements of Project Plan**
- What methods to be used to validate performance
  - **Testing for adherence to spec**
  - **Monitoring usage**
    - **Web trends**
    - **Database Logs**

  - **Transaction logs**
  - **Setup schedule for reviews**

- **2.10 Elements of Project Plan**
- **2.10 Elements of Project Plan**
- Define where possible problems can occur
  - **New technologies**
  - **Business risks**
  - **Resources**
- Define Contingency Plans
  - **Development methods**
  - **If the worst happens, what to do**
- **Summary**
- **Summary**

2. **SOFTWARE DEVELOPMENT PLAN (SDP)**

   2.10 Elements of SDP

2. **Lifecycle Planning**

Lecture:26

- **Reference to Previous Session**

2. **SOFTWARE DEVELOPMENT PLAN (SDP)**

   2.10 Elements of SDP

2. **LIFECYCLE PLANNING**

Examine:

1. **How well requirements are understood at the beginning? Is it going to change when moving through the project?**
2. **Is system architecture understood? Any changes on the way?**
3. **Level of reliability?**

4. **Level of re-design for future versions?**

Examine:

1. **Risk Level?**
2. **Stuck With Predefined Schedule?**
3. **Need For Midcourse Correction?**
4. **Customer Informed Through The Project?**
5. **Visible Management Through The Project?**
6. **If a Model is chosen, How much it needs Modification?**

4. **PLANNING DOCUMENTS**

5. **PLANNING DOCUMENTS**

- You (the PM) need to choose which documents are appropriate
- Docs do not have to be lengthy
- Small Set

4. **PLANNING DOCUMENTS**

1. **Software Development Plan (SDP)**
2. **The Statement of Requirement (SOR)**
3. **System Specification**
4. **Design Specification**
5. **The Data Item Descriptions (DID s)**
6. **The software configuration management plan (SCMP)**
7. **Statistical software process improvement (SSPI)**
8. **Communications management plan**

4. **PLANNING DOCUMENTS**

9. **The software quality assurance plan (SQAP)**
10. **The Risk Mitigation, Monitoring and Management Plan (RMMM)**

11. **Project Development Budget**
12. **Maintenance Documents**
13. **Statement of Work (SOW)**
14. **Assignment Matrix (RAM)**
15. **Project Charter**
16. **Risk Management Plan**
4. **PLANNING DOCUMENTS**
17. **Software Project Management Plan (SPMP)**
18. **Statement of Need**
19. **System Interface Specification**
20. **Software Validation & Verification Plan**
21. **The Software Test Plan**
22. **User Documentation**
23. **Support Plan**
24. **Migration Plan**
25. **Operations Plan**
2. **The Statement of Requirement (SOR)**
3. **The Statement of Requirement (SOR)**

- **The project manager (or the advising consultant adopting this role) must ensure that the project sponsor has produced a written statement of requirements (SOR).**

2. **The Statement of Requirement (SOR)**

This must be a thorough document which is:

- **Unambiguous**
- **Fully defined or complete**
- **Verifiable deliverables**
- **No conflicts**

- **Consistent**
- **Auditable**

- **The SOR should be closely matched to the contract and there should be no conflict of interests between the two.**

3. **System Specification**

4. **System Specification**

- **If a *System Specification* has been properly developed, nearly all information required for a description of software scope is available and documented before software project planning begins.**
- **In cases where a specification has not been developed, the planner must take on the role of system analyst to determine attributes and bounds that will influence estimation tasks.**

4. **Design Specification**

5. **Design Specification**

- **The project design specification is a detailed plan of how the requirements are to be implemented.**
- **The implementation decisions involved may contain potential problems.**
- **Example: Problems will arise if the selected hardware turns out to be inadequate, such as if the CPU is too slow, the LAN is not sufficiently reliable, or the maximum available memory is insufficient.**

5. **The Data Item Descriptions (DID s)**

6. **The Data Item Descriptions (DID s)**

- **The DID s are a comprehensive set of documentation standards that cover all phases of software development, maintenance and the production of user reference manuals.**
- **The DID s include a section called *preparation instructions* that provide a large degree of freedom by permitting tailoring of the document format and the use of alternate presentation styles.**

6. **The software configuration management plan**

7. **The software configuration management plan**

- **The software configuration management plan (SCMP) is part of the project's software development plan.**

- **The SCMP may appear as a separate document or as a section within the project development plan.**
- **The SCMP documents the resources that are needed, how they are to be used, and which standards and procedures will be applied during the project.**

7. **Statistical software process improvement (SSPI)**
8. **Statistical software process improvement (SSPI)**

- **SSPI uses software failure analysis to collect information about all errors and defects encountered as an application, system, or product is developed and used.**

7. **Statistical software process improvement (SSPI)**

- **Failure analysis works in the following manner:**

1. **All errors and defects are categorized by origin (e.g., flaw in specification, flaw in logic, non conformance to standards).**
2. **The cost to correct each error and defect is recorded.**
3. **The number of errors and defects in each category is counted and ranked in descending order.**
4. **The overall cost of errors and defects in each category is computed.**
5. **Resultant data are analyzed to uncover the categories that result in highest cost to the organization.**
6. **Plans are developed to modify the process with the intent of eliminating or reducing the frequency of the class of errors and defects that is most costly.**
8. **Communications management plan**
9. **Communications management plan**

- **A communications management plan is a document that provides:**

1. **Details of methods to be used to gather and store various types of information.**
2. **A distribution structure that, details to whom information (status reports, data, schedule, technical documentation, etc. will flow, and what methods (written reports, meetings, etc.) will be used to distribute various types of information. This structure must be compatible with the responsibilities and reporting relationships described by the project organization chart.**
8. **Communications management plan**
3. **A description of the information to be distributed.**

4. **Production schedules showing when each type of communication will be produced.**
5. **Methods for accessing information between scheduled communications.**
6. **A method for updating and refining the communications management plan as the project progresses and develops.**
9. **The software quality assurance plan**
10. **The software quality assurance plan**

- **Every project must have a quality plan.**
- **It is drawn up by the project manager at the start of the project and should be agreed with the project sponsor.**

9. **The software quality assurance plan**

- **You would expect the quality plan to contain the following elements:**

1. **Statement of the quality control organisation**
2. **Identification of specific standards and methods that will be used**
3. **Definition of the quality control procedures; this is aligned to the Work Breakdown Structure**
4. **Specification of quality milestones**
5. **Detail of unusual features**
6. **Change control and configuration management**
7. **Detail of acceptance procedures**
8. **Specification of quality assurance procedures**
10. **The RMMM Plan**
11. **The RMMM Plan**

- **A collection of risk information sheets developed for all risks that lie above the cut off.**
- **A risk management strategy can be included in the software project plan or the risk management steps can be organized into a separate *Risk Mitigation, Monitoring and Management Plan.***
- **The RMMM plan documents all work performed as part of risk analysis and are used by the project manager as part of the overall project plan.**

11. **Project Development Budget**

12. **Project Development Budget**

- **Good estimates are important, as they form the foundations of a good project development plan.**
- **This plan, prepared by the project manager, is produced during the initial stages of the project and includes estimates related to:**

1. **The project development budget**
2. **The project development schedule**
3. **The required development resources (development staff, development equipment etc.)**

11. **Project Development Budget**

- **In parallel with integration and testing, the following managerial and activities take place:**

1. **Final budgeting of the project; the cost of changes is determined, risk contingency activities are evaluated, and the budget is updated.**
2. **Training is conducted for users, operators, customers, installers, maintenance engineers, and marketing engineers.**
3. **Installation sites are prepared, and the infrastructure for hardware and special equipment is planned and installed.**
4. **The development team size is reduced.**

12. **Maintenance Documents**

13. **Maintenance Documents**

- At the conclusion of the integration and test phase all documentation must be complete and ready for delivery, including:

1. **Maintenance documentation**
2. **Final user documentation**
3. **All updated development documentation**
4. **Test documentation and test reports**

13. **The statement of work (SOW)**

14. **The statement of work (SOW)**

- **The statement of work is the basis of the contract between the pro-poser and the customer, and is often incorporated into the contract.**
- **The SOW contains a detailed list of all work to be performed by the pro-poser for the benefit of the customer.**
- **It is a narrative description of *products* or *services* to be supplied by the project. The SOW indicates a:**
- ***Business need***
- ***Product scope description***
- ***Strategic plan***

14. **Responsibility Assignment Matrix (RAM)**

15. **Responsibility Assignment Matrix (RAM)**

- Project roles and responsibilities should be closely linked to the project scope definition.
- A Responsibility Assignment Matrix (RAM) is often used for this purpose.

15. **Project Charter**

16. **Project Charter**

- A document issued by senior management that formally authorizes the existence of a *project*.
- And it provides the *project manager* with the authority to apply organizational resources to project activities.

16. **Risk management plan**

17. **Risk management plan**

- **The risk management plan describes how risk identification, qualitative and quantitative analysis, response planning, monitoring, and control will be structured and performed during the project life cycle.**
- **The risk management plan may include the following:**

16. **Risk management plan**

1. ***Methodology.*** **Defines the approaches, tools, and data sources that may be used to perform risk management on this project. Different types of assessments may be appropriate, depending upon the project stage, amount of information available, and flexibility remaining in risk management.**

16. **Risk management plan**

2. ***Roles and responsibilities.*** **Defines the lead, support, and risk management team membership for each type of action in the risk management plan. Risk management teams organized outside of the project office may be able to perform more independent, unbiased risk analyses of project than those from the sponsoring project team.**

3. ***Budgeting.*** **Establishes a budget for risk management for the project.**

16. **Risk management plan**

4. ***Timing.*** **Defines how often the risk management process will be performed throughout the project life cycle. Results should be developed early enough to affect decisions. The decisions should be revisited periodically during project execution.**

5. ***Scoring and interpretation.*** **The scoring and interpretation methods appropriate for the type and timing of the qualitative and quantitative risk analysis being performed. Methods and scoring must be determined in advance to ensure consistency.**

16. **Risk management plan**

6. ***Thresholds.*** **The threshold criteria for risks that will be acted upon, by whom, and in what manner. The project owner, customer, or sponsor may have a different risk threshold. The acceptable threshold forms the target against which the project team will measure the effectiveness of the risk response plan execution.**

7. ***Reporting formats.*** **Describes the content and format of the risk response plan. Defines how the results of the risk management processes will be documented, analyzed, and communicated to the project team, internal and external stakeholders, sponsors, and others.**

8. ***Tracking.*** **Documents how all facets of risk activities will be recorded for the benefit of the current project, future needs, and lessons learned. Documents if and how risk processes will be audited.**

5. **SCHEDULING**

6. **SCHEDULING**

- Any project can be completed, given:
  - **an infinite amount of time and resources**.
- Realistically:
  - **the amount of time available for project development is always finite.**
- In fact, in most cases it is less than the project manager considers sufficient.

- Few projects are completed ahead of time; many projects overrun their schedule.

5. **SCHEDULING**

- **The project schedule is one of the most important parts of the project development plan.**
- **The plan includes:**
  - **scheduling of development activities and**
  - **scheduling of project resources, particularly people**

5. **SCHEDULING**

- **The project development plan describes in detail:**
  - **how the project manager plans to develop the project**
  - **what resources will be required and**
  - **how these resources will be applied**

5. **SCHEDULING**

- **No matter how well the project schedule is prepared, that schedule is useless unless it is adhered to.**
- **It is the project manager's responsibility to withstand pressure and to assure that the project is developed in an orderly fashion, according to the schedule.**
- **Whenever circumstances change, the project schedule should then be updated to reflect the new situation.**

5. **SCHEDULING**

- **A schedule is a list of:**
  - **activities and**
  - **anticipated time of implementation of these activities**
- **There are many ways of representing a schedule:**
  - **lists of activities,**
  - **diagrams,**
  - **graphs etc.**

5. **SCHEDULING**

- **The most common methods of schedule representation are :**
  - **precedence network diagrams (such as PERT),**
  - **Gantt charts and**
  - **lists of milestones.**

6. **GUIDELINES FOR SUCCESSFUL PLANNING**

7. **GUIDELINES FOR SUCCESSFUL PLANNING**

- **A common failure in many kinds of planning is that the plan is never really implemented. Instead, all focus is on writing a plan document.**
- **Most of the following guidelines help to ensure that the planning process is carried out completely and is implemented completely -- or, deviations from the intended plan are recognized and managed accordingly.**

6. **GUIDELINES FOR SUCCESSFUL PLANNING**

i. **Involve the Right People in the Planning Process**

- **It's critical that all parts of the system continue to exchange feedback in order to function effectively. This is true no matter what type of system.**
- **When planning, get input from everyone who will be responsible to carry out parts of the plan, along with representative from groups who will be effected by the plan.**

6. **GUIDELINES FOR SUCCESSFUL PLANNING**

i. **Involve the Right People in the Planning Process**

- **Of course, people involved should be responsible to review and authorize the plan.**

6. **GUIDELINES FOR SUCCESSFUL PLANNING**

ii. **Write Down the Planning Information and Communicate it Widely**

- **New managers, in particular, often forget that others don't know what these managers know.**
- **Even if managers do communicate their intentions and plans verbally, chances are that others won't completely hear or understand what the manager wants done.**

6. **GUIDELINES FOR SUCCESSFUL PLANNING**

ii. **Write Down the Planning Information and Communicate it Widely**

- **Also, as plans change, it's extremely difficult to remember who is supposed to be doing what and according to which version of the plan.**

6. **GUIDELINES FOR SUCCESSFUL PLANNING**

ii. **Write Down the Planning Information and Communicate it Widely (Cont.)**

- **Key stakeholders (employees, management, board members, sponsors, customers, clients, etc.) may request copies of various types of plans.**
- **Therefore, it's critical to write plans down and communicate them widely.**

6. **GUIDELINES FOR SUCCESSFUL PLANNING**

iii. **Goals and Objectives Should Be SMARTER**

- **Specific**
- **Measurable**
- **Acceptable**
- **Realistic**
- **Time frame**
- **Extending**
- **Rewarding**

6. **GUIDELINES FOR SUCCESSFUL PLANNING**

iv. **Build in Accountability (Regularly Review Who's Doing What and By When?)**

- **Plans should specify who is responsible for achieving each result, including goals and objectives.**
- **Dates should be set for completion of each result, as well.**
- **Responsible parties should regularly review status of the plan.**

6. **GUIDELINES FOR SUCCESSFUL PLANNING**

vi. **Build in Accountability (Regularly Review Who's Doing What and By When?) (Cont.)**

- **Be sure to have someone of authority "sign off" on the plan, including putting their signature on the plan to indicate they agree with and support its contents.**

- **Include responsibilities in policies, procedures, job descriptions, performance review processes, etc.**

6. **GUIDELINES FOR SUCCESSFUL PLANNING**

v. **Note Deviations from the Plan and Re-plan Accordingly**

- **It's OK to deviate from the plan.**
- **The plan is not a set of rules.**
- **It's an overall guideline.**
- **It is equally important to notice deviations and adjust the plan accordingly.**

6. **GUIDELINES FOR SUCCESSFUL PLANNING**

vi. **Evaluate Planning Process and the Plan**

- **During the planning process, regularly collect feedback from participants.**
- **Do they agree with the planning process?**

  If not, what don't they like and how could it be done better?
- **In large, ongoing planning processes (such as strategic planning, business planning, project planning, etc.), it's critical to collect this kind of feedback regularly.**

6. **GUIDELINES FOR SUCCESSFUL PLANNING**

vi. **Evaluate Planning Process and the Plan (Cont.)**

- **During regular reviews of implementation of the plan, assess if goals are being achieved or not.**
- **If not, were goals realistic? Do responsible parties have the resources necessary to achieve the goals and objectives?**

6. **GUIDELINES FOR SUCCESSFUL PLANNING**

vi. **Evaluate Planning Process and the Plan (Cont.)**

- **Should goals be changed? Should more priority be placed on achieving the goals? What needs to be done?**
- **Write down how the planning process could have been done better. File it away and read it the next time you conduct the planning process.**

6. **GUIDELINES FOR SUCCESSFUL PLANNING**

vii. **Recurring Planning Process is at Least as Important as Plan Document**

- **Far too often, primary emphasis is placed on the plan document.**
- **This is extremely unfortunate because the real treasure of planning is the planning process itself.**

6. **GUIDELINES FOR SUCCESSFUL PLANNING**

vii. **Recurring Planning Process is at Least as Important as Plan Document (Cont.)**

- **During planning, planners learn a great deal from ongoing analysis, reflection, discussion, debates and dialogue around issues and goals in the system.**
- **The ongoing communications are what sensitize people to understanding and following the values and behaviors described**

6. **GUIDELINES FOR SUCCESSFUL PLANNING**

viii. **Nature of the Process Should Be Compatible to Nature of Planners**

- **A prominent example of this type of potential problem is when planners don't prefer the "top down" or "bottom up", "linear" type of planning**
- **For example, going from general to specific along the process of an environmental scan, SWOT analysis, mission/vision/values, issues and goals, strategies, objectives, timelines, etc.**

6. **GUIDELINES FOR SUCCESSFUL PLANNING**

viii. **Nature of the Process Should Be Compatible to Nature of Planners (Cont.)**

- **There are other ways to conduct planning. For an overview of various methods, see (in the following, the models are applied to the strategic planning process, but generally are eligible for use elsewhere):**
- **Summary**
- **Summary**

1. **PROJECT PLANNING**
2. **SOFTWARE DEVELOPMENT PLAN (SDP)**
3. **LIFE CYCLE PLANNING**
4. **PLANNING DOCUMENTS**
5. **SCHEDULING**

6. **GUIDELINES FOR SUCCESSFUL PLANNING**

Lecture:27

- Reference to Previous Topics

I. Overview of Management

II. Software Developmental Fundamentals

III. Processes

IV. Planning

V. **Organization**

I. **Organization Perspective**

1. **Basic definition**
2. **Organization as a System**
3. **Structural Dimensions**

II. **Project Organization**

1. **Organizational Systems**
2. **Organizational Cultures and Styles**
3. **Organizational Structure**
    - **Traditional Structures of Business Organization**
4. **Organizational Planning**
5. **Staff Acquisition**
6. **Team Development**
7. **Organizational Management Tools**
8. **Key Concept in Design of Organization**

III. **Contractual Obligations**

1. **Fixed Cost**
2. **Cost plus**

IV. **SOW**

1. **SOW Template**

V. **Software built around a Scheduling Engine**

1. **Primavera**
2. **TurboProject**
3. **OpenPlan**
4. **Microsoft Project**
5. **AutoPlan**
6. **Project Scheduler 8**
7. **CA SuperProject**
8. **Timeline**

1. **Basic Definition**

- **Basically, an organization is a group of people intentionally organized to accomplish an overall, common goal or set of goals.**
- **Business organizations can range in size from two people to tens of thousands.**
- **People are managed through an organizational structure.**

1. **Basic Definition**

- **This hierarchical structure is based on the four cornerstones of management:**
  - **Delegation**
  - **Authority**
  - **Responsibility and**
  - **Supervision.**
- **Delegation allows authority, and authority produces (and requires) responsibility.**
- **Both authority and responsibility require supervision, and effective supervision requires a suitable organizational structure:**

2. **Organization as a System**

- **It helps to think of organizations as systems. A system is an organized collection of parts that are highly integrated in order to accomplish an overall goal.**
- **The system has various inputs which are processed to produce certain outputs, that together, accomplish the overall goal desired by the organization.**

2. **Organization as a System**

- **There is ongoing feedback among these various parts to ensure they remain aligned to accomplish the overall goal of the organization.**
- **There are several classes of systems, ranging from very simple frameworks all the way to social systems, which are the most complex. Organizations are, of course, social systems.**

2. **Organization as a System**

- **Systems have inputs, processes, outputs and outcomes.**
- **Inputs to the system include resources such as people, materials, technologies and money.**
- **These inputs go through a process where they're aligned, moved along and carefully coordinated, ultimately to achieve the goals set for the system.**
- **Outputs are tangible results produced by processes in the system, such as products or services for consumers.**

2. **Organization as a System**

- **Another kind of result is outcomes, or benefits for consumers, e.g., jobs for workers, enhanced quality of life for customers, etc.**
- **Systems can be the entire organization, or its departments, groups, processes, etc.**

2. **Organization as a System**

- **Feedback comes from:**
  - **Employees who carry out processes in the organization**
  - **Customers/clients using the products and services, etc.**
  - **Larger environment of the organization, such as;**
    - **Influences from government**
    - **Society**
    - **Economics and**

- **Technologies**

2. **Organization as a System**

- **Each organization has numerous subsystems, as well.**
- **Common examples of subsystems are:**
  - **Departments**
  - **Programs**
  - **Projects, teams**
  - **Processes to produce products or services, etc**

2. **Organization as a System**

- **The organizational system is defined by its:**
  - **Legal documents (articles of incorporation, by laws, roles of officers, etc.)**
  - **Mission**
  - **Goals and strategies**
  - **Policies and procedures**
  - **Operating manuals, etc**

3. **Structural dimensions**

   a) **Centralization -the extent to which functions are dispersed in the organization, either in terms of integration with other functions or geographically**

   b) **Formalization - regarding the extent of policies and procedures in the organization**

   c) **Hierarchy - regarding the extent and configuration of levels in the structure**

4. **Structural dimensions**

   a) **Standardization - regarding the extent that organizational processes are standardized**

   b) **Specialization - regarding the extent to which activities are refined**

   c) **Training - regarding the extent of activities to equip organization members with knowledge and skills to carry out their roles**

II. **Project Organization**

III. **Project Organization**

1. **Organizational Systems**
2. **Organizational Cultures and Styles**
3. **Organizational Structure**
4. **Organizational Planning**
5. **Staff Acquisition**
6. **Team Development**
7. **Organizational Management Tools**
8. **Key Concept in Design of Organization**

1. **Organizational Systems**

- **Project-based organizations are those whose operations consist primarily of projects.**
- **These organizations fall into two categories:**

a) **Organizations that derive their revenue primarily from performing projects for others — architectural firms, engineering firms, consultants, construction contractors, government contractors, nongovernmental organizations, etc.**

1. **Organizational Systems**

b) **Organizations that have adopted *management by projects.***

These organizations tend to have management systems in place to facilitate project management.

Example: The financial systems of these organizations are often specifically designed for accounting, tracking, and reporting on multiple simultaneous projects.

1. **Organizational Systems**

- **Non-project-based organizations often lack management systems designed to support project needs efficiently and effectively.**
- **The absence of project-oriented systems usually makes project management more difficult.**
- **In some cases, non-project based organizations will have departments or other subunits that operate as project-based organizations with systems to match.**

- **The project management team should be acutely aware of how the organization's systems affect the project.**
- **Example: If the organization rewards its functional managers for charging staff time to projects, then the project management team may need to implement controls to ensure that assigned staff members are being used effectively on the project.**

2. **Organizational Cultures and Styles**

- **Most organizations have developed unique and describable cultures.**
- **These cultures are reflected in their:**
  - **shared values,**
  - **norms,**
  - **beliefs, and**
  - **expectations;**
  - **policies and**
  - **procedures;**
  - **view of authority relationships;**
  - **and in numerous other factors.**

2. **Organizational Cultures and Styles**

- **Organizational cultures often have a direct influence on the project.**
  - **A team proposing an unusual or high-risk approach is more likely to secure approval in an aggressive or entrepreneurial organization.**
  - **A project manager with a highly participative style is apt to encounter problems in a rigidly hierarchical organization, while a project manager with an authoritarian style will be equally challenged in a participative organization.**

3. **Organizational Structure**

- **The structure of the performing organization often constrains the availability of or terms under which resources become available to the project.**
- **Organizational structures can be characterized as spanning a spectrum from functional to projectized, with a variety of matrix structures in between.**

- **The following figure shows key project related characteristics of the major types of enterprise organizational structures:**

- **3.1 Traditional Structures of Business Organization**

i. **Functional Structure**

ii. **Projectized Organization**

iii. **Matrix organizations**

iv. **Project Office**

- **3.1 Traditional Structures of Business Organization**

iv. Project Office

- There is a range of uses for what constitutes a project office. A project office may operate on a continuum from providing support functions to project managers in the form of training, software, templates, etc. to actually being responsible for the results of the project.

I. **Organization Perspective**

1. **Basic definition**
2. **Organization as a System**
3. **Structural Dimensions**

II. **Project Organization**

1. **Organizational Systems**
2. **Organizational Cultures and Styles**
3. **Organizational Structure**

- **Traditional Structures of Business Organization**

Lecture28

I. **Organization Perspective**

II. **Basic definition**

III. **Organization as a System**

IV. **Structural Dimensions**

V. **Project Organization**

1. **Organizational Systems**
2. **Organizational Cultures and Styles**
3. **Organizational Structure**
    - **Traditional Structures of Business Organization**

VI. **Project Organization**

1. **Organizational Planning**
2. **Staff Acquisition**
3. **Team Development**
4. **Organizational Management Tools**
5. **Key Concept in Design of Organization**

VII. **Contractual Obligations**

1. **Fixed Cost**
2. **Cost plus**

VIII. **SOW**

1. **SOW Template**

IX. **Software built around a Scheduling Engine**

1. **Primavera**
2. **TurboProject**
3. **OpenPlan**
4. **Microsoft Project**
5. **AutoPlan**
6. **Project Scheduler 8**
7. **CA SuperProject**
8. **Timeline**

4. **Organizational Planning**

5. **Organizational Planning**

- **Involves identifying, documenting, and assigning:**
  - **Project roles**
  - **Responsibilities, and**
  - **Reporting relationships**
- **Roles, responsibilities, and reporting relationships may be assigned to individuals or to groups.**
- **The individuals and groups may be part of the organization performing the project, or they may be external to it.**

4. **Organizational Planning**

- **Internal groups are often associated with a specific functional department such as engineering, marketing, or accounting.**
- **On most projects, the majority of organizational planning is done as part of the earliest project phases.**
- **However, the results of this process should be reviewed regularly throughout the project to ensure continued applicability.**
- **If the initial organization is no longer effective, then it should be revised promptly.**
- **4.1 Inputs to Organizational Planning**
- **4.1 Inputs to Organizational Planning**

i. **Project interfaces**

ii. **Staffing requirements**

iii. **Constraints**

- **4.1 Inputs to Organizational Planning**

i. **Project interfaces: generally fall into one of three categories:**

   a) **Organizational interfaces**

   b) **Technical interfaces**

   c) **Interpersonal interfaces**

ii. **Project interfaces**

a) **Organizational interfaces:**

- **Formal and informal reporting relationships among different organizational units.**

- **Organizational interfaces may be highly complex or very simple.**

i. **Project interfaces**

b) **Technical interfaces:**

Formal and informal reporting relationships among different technical disciplines.

i. **Project interfaces**

c) **Interpersonal interfaces:**

d) **Formal and informal reporting relationships among different individuals working on the project.**

e) **Staffing requirements**

- Define what kinds of competencies are required from what kinds of individuals or groups and in what time frames.
- Staffing requirements are a subset of the overall resource requirements identified during resource planning .

iii. **Constraints**

- **Factors that limit the project team's options.**
- **A project's organizational options may be constrained in many ways.**
- **Common factors that may constrain how the team is organized include, but are not limited to, the following:**

a) **Organizational structure of the performing organization**

b) **Collective bargaining agreements**

c) **Preferences of the project management team**

d) **Expected staff assignment**

iii. **Constraints**

a) **Organizational structure of the performing organization**

- **An organization whose basic structure is a strong matrix means a relatively stronger role for the project manager than one whose basic structure is a weak matrix.**

iii. **Constraints**

iv. **Collective bargaining agreements**

v. **Contractual agreements with unions or other employee groups may require certain roles or reporting relationships (in essence, the employee group is a stakeholder).**

iii. **Constraints**

c) **Preferences of the project management team**

d) **If members of the project management team have had success with certain structures in the past, then they are likely to advocate similar structures in the future.**

iv. **Constraints**

c) **Expected staff assignment**

d) **How the project is organized is often influenced by the competencies of specific individuals.**

- **4.2 Tools and Techniques for Organizational Planning**
- **4.2 Tools and Techniques for Organizational Planning**

i. **Templates**

ii. **Human resource practices**

iii. **Organizational theory**

iv. **Stakeholder analysis**

v. **Templates**

- **Although each project is unique, most projects will resemble another project to some extent.**

- **Using the role and responsibility definitions or reporting relationships of a similar project can help expedite the process of organizational planning.**

ii. **Human resource practices**

- **Many organizations have a variety of:**
  - **Policies**
  - **guidelines, and**
  - **procedures that can help the project management team with various aspects of organizational planning.**
- **Example: An organization that views managers as "coaches" is likely to have documentation on how the role of "coach" is to be performed.**

iii. **Organizational theory**

- **There is a substantial body of literature describing how organizations can and should be structured.**
- **Although only a small subset of this body of literature is specifically targeted toward project organizations, the project management team should be generally familiar with the subject of organizational theory so as to be better able to respond to project requirements.**

iv. **Stakeholder analysis**

- Identification of stakeholders and the needs of the various stakeholders should be analyzed to ensure that their needs will be met.
- **4.3 Outputs from Organizational Planning**
- **4.3 Outputs from Organizational Planning**

i. **Role and responsibility assignments**

ii. **Staffing management plan**

iii. **Organization chart**

iv. **Supporting detail**

v. **Role and responsibility assignments (RAM)**

- **Project roles (who does what) and responsibilities (who decides what) must be assigned to the appropriate project stakeholders.**
- Project roles and responsibilities should be closely linked to the project scope definition.

- **A Responsibility Assignment Matrix (RAM) is often used for this purpose.**

ii. **Staffing management plan**

- **The staffing management plan describes when and how human resources will be brought onto and taken off of the project team.**
- **The staffing plan may be:**
  - **Formal or informal,**
  - **Highly detailed or broadly framed, based on the needs of the project**
  - **It is a subsidiary element of the overall project plan.**

iii. **Organization chart**

- **An organization chart is any graphic display of project reporting relationships.**
- **It may be:**
  - **Formal or informal,**
  - **Highly detailed or broadly framed, based on the needs of the project.**

iv. **Supporting detail**

- Supporting detail for organizational planning varies by application area and project size.
- Information frequently supplied as supporting detail includes, but is not limited to:
  - **Organizational impact —what alternatives are precluded by organizing in this manner.**

iv. **Supporting detail**

  - **Job descriptions —written outlines by job title of the competencies, responsibilities, authority, physical environment, and other characteristics involved in performing a given job. Also called position descriptions.**
  - **Training needs —if the staff to be assigned is not expected to have the competencies needed by the project, those competencies will need to be developed as part of the project.**

5. **STAFF ACQUISITION**

6. **STAFF ACQUISITION**

- **Staff acquisition involves getting the needed human resources (individuals or groups) assigned to and working on the project.**

- **In most environments, the "best" resources may not be available, and the project management team must take care to ensure that the resources that are available will meet project requirements.**

6. **TEAM DEVELOPMENT**

7. **TEAM DEVELOPMENT**

- **Team development includes both:**
  - **enhancing the ability of stakeholders to contribute as individuals as well as**
  - **enhancing the ability of the team to function as a team.**
- **Individual development (managerial and technical) is the foundation necessary to develop the team.**
- **Development as a team is critical to the project's ability to meet its objectives.**

6. **TEAM DEVELOPMENT**

- **Principles Of Staffing A Project**
  - **Top Talent**
  - **Job Matching**
  - **Career Progression**
  - **Team Balance**
  - **Misfit Elimination**
- **Team Organization**
  - **Important who works with whom**

7. **Organizational Management Tools**

8. **Organizational Management Tools**

i. **Management Development**

ii. **Supervisory Training**

iii. **Team Building**

iv. **Vital Statistics**

v. **Progress reporting**

vi. **Compliance of Rule of Business**

vii. **Trade Offs**

viii. **Quality Assurance**

ix. **Beneficiaries Concern**

7. **Organizational Management Tools**

i. **Management Development**

- **Responsibility**
- **Authority**
- **Competence**
- **Resource Distribution**
- **Pre-requisites**
- **Constraints**
- **Calendar**

7. **Organizational Management Tools**

ii. **Supervisory Training**

- **Field/on site operations**
- **Concept clearance**
- **Procedural details**
- **Resource management**
- **Activity Scheduling**

7. **Organizational Management Tools**

iii. **Team Building**

- **Managers**
- **Professionals**
- **Technical support group**

- **Logistical support group**
- **Skill set evaluation**

7. **Organizational Management Tools**

iv. **Vital Statistics**

- **Historical facts**
- **Technical data**
- **Direct concerns**
- **Lesson learned**
- **Identification of missing links**
- **Reliability of data**
- **Relevance of data**

7. **Organizational Management Tools**

v. **Progress reporting**

- **Mandatory periodic reports**
- **Exception reports**
- **Event reporting**
- **Current status reporting**
- **Reporting formats**
- **Reporting frequency**
- **Report recipient**
- **Reporting officer**

7. **Organizational Management Tools**

v. **Progress reporting**

- **Reporting Media**
- **Response analysis (of previous reporting)**

- **Review of Reports**
- **Signing of Reports**
- **Tracking of Reports**
- **Mitigations offered**
- **Corresponding the dead lines**

7. **Organizational Management Tools**

vi. **Compliance of Rule of Business**

- **Financial**
- **Procedural**
- **Administrative**
- **Justifications for deviations**
- **Acceptance of derailments**

7. **Organizational Management Tools**

vii. **Trade Offs**

- **Trade off between DO or DON'T**
- **Quality of job (in the light of constraints)**
- **Limiting the scope/deliverables**
- **Meeting targets with minimum standards**
- **Unavoidable mandatory deliverables**

viii. **Quality Assurance**

- **Standard QA procedures**
- **Self defined measures**
- **Task-specific controls**

7. **Organizational Management Tools**

ix. **Beneficiaries Concern**

- **Acceptance**
- **Enthusiasm**
- **Adding comforts in terms of use, cost or time**
- **Confidence building-The Reliability**

III. **Contractual Obligations**

IV. **Contractual Obligations**

1. **Types of Contracts**
    a) **Fixed price or lump sum**
    b) **Cost reimbursable**
    c) **Time & material contracts**
    d) **Unit price contracts**
2. **Types of Contracts**

a) **Fixed price or lump sum**: **Involves a fixed total price for a well-defined product or service**

b) **Cost reimbursable: Involves payment to the seller for direct and indirect costs**

c) **Time & material contracts: Hybrid of both fixed price and cost reimbursable, often used by consultants**

d) **Unit price contracts: Requires the buyer to pay the seller a predetermined amount per unit of service**

e) **Cost Reimbursable Contracts**

- **Contract Types Versus Risk**

IV. **Statement of Work (SOW)**

1. **Definition**
2. **Template**
3. **Statement of Work (SOW)- Definition**

- A description of the work required for the project
- Sets the “boundary conditions”

- SOW vs. CSOW (Contract SOW)
  - Latter: uses legal language as part of a competitive bidding scenario
- Can be used in the final contract – be careful, be specific, be clear

1. **Statement of Work (SOW)- Definition**

- Typically done after approval (after "Go")
- Can be multiple versions

i. **List of deliverables for an RFP**

ii. **More detailed within final RFP**

iii. **Binding version from contract**

2. **SOW Template**
3. **SOW Template**
4. **SOW Template**
5. **SOW Template**

V. **Software built around a Scheduling Engine**

VI. Software built around a Scheduling Engine

- **These packages were originally created to manage one project at a time, but over the years have been enhanced to handle multiple projects.**
- **Well known examples include :**

1. **Primavera**
2. **TurboProject**

V. **Software built around a Scheduling Engine**

5. **OpenPlan**
6. **Microsoft Project**
7. **AutoPlan**
8. **Project Scheduler 8**

9. **CA SuperProject**
10. **Timeline**
    - I. **Organization Perspective**
        1. **Basic definition**
        2. **Organization as a System**
        3. **Structural Dimensions**
    - II. **Project Organization**
        1. **Organizational Systems**
        2. **Organizational Cultures and Styles**
            - **Traditional Structures of Business Organization**
        3. **Organizational Planning**
        4. **Staff Acquisition**
        5. **Team Development**
        6. **Organizational Management Tools**
        7. **Key Concepts in Design of Organization**
    - III. **Contractual Obligations**
        1. **Fixed Cost**
        2. **Cost plus**
    - IV. **SOW**
        1. **SOW Template**
    - V. **Software built around a Scheduling Engine**
        1. **Primavera**
        2. **TurboProject**
        3. **OpenPlan**
        4. **Microsoft Project**

5. **AutoPlan**
6. **Project Scheduler 8**
7. **CA SuperProject**
8. **Timeline**

Lecture :29

Software Project Management

Software Project Management

Software Project Management

"***If you don't know where you are, a map won't help.***"

Watts Humphrey

i. **Software cost and effort estimation will never be an exact science.**

ii. **Too many variables can affect the ultimate cost of software and effort applied to develop it, these variables are:**

a) **Human**

b) **Technical**

c) **Environmental**

d) **Political**

iii. **However, software project estimation can be transformed from a black art to a series of systematic steps that provide estimates with acceptable risk.**

iv. **To achieve reliable cost and effort estimates, a number of options available:**

a) **Delay estimation until late in the project (obviously, we can achieve 100% accurate estimates after the project is complete!).**

b) **Base estimates on similar projects that have already been completed.**

c) **Use relatively simple decomposition techniques to generate project cost and effort estimates.**

d) **Use one or more empirical models for software cost and effort estimation.**

a. **Delay estimation until late in the project (obviously, we can achieve 100% accurate estimates after the project is complete!).**

Unfortunately, the first option, however attractive, is not practical. Cost estimates must be provided 'up front.'

However, we should recognize that the longer we wait, the more we know, and the more we know, the less likely we are to make serious errors in our estimates.

b. **Base estimates on similar projects that have already been completed.**

The second option can work reasonably well, if the current project is quite similar to past efforts and other project influences (e.g., the customer, business conditions, deadlines) are equivalent.

Unfortunately, past experience has not always been a good indicator of future results.

c. **Use relatively simple decomposition techniques to generate project cost and effort estimates.**

d. **Use one or more empirical models for software cost and effort estimation.**

The remaining options are viable approaches to software project estimation.

Ideally, the techniques noted for each option should be applied in tandem; each used as a cross-check for the other.

v. **Software project estimation is a form of problem solving, and in most cases, the problem to be solved (i.e., developing a cost and effort estimate for a software project) is too complex to be considered in one piece.**

vi. **For this reason, we decompose the problem, re-characterizing it as a set of smaller (and hopefully, more manageable) problems.**

vii. **By decomposing a project into major functions and related software engineering activities, cost and effort estimation can be performed in a stepwise fashion.**

Very difficult to do, but often needed

Created, used or refined during

Strategic planning

Feasibility study and/or SOW

Proposals

Vendor and sub-contractor evaluation

Project planning (iteratively)

Basic process

1) **Estimate the size of the product**
2) **Estimate the effort (man-months)**
3) **Estimate the schedule**

NOTE: Not all of these steps are always explicitly performed

Cone of Uncertainty

In a software project, unless you are sure that your estimation is accurate, you cannot make much progress.

Estimation of factors such as

cost,

effort,

risks, and

resources is crucial.

It gives you a fair idea of the size of the project. You can use the information about size to estimate the cost, effort, and duration of the project.

This further helps plan for resources and schedule the project.

i. **Estimation of resources, cost, and schedule for a software engineering effort requires:**

- **Experience,**
- **Access to good historical information**
- **Courage to commit to quantitative predictions when qualitative information is all that exists.**

ii. **Estimation carries inherent risk and this risk leads to uncertainty.**

a) **Project complexity**

b) **Project size**

c) **The degree of structural uncertainty**

d) **The availability of historical information**

e) **Risk**

f) **Project complexity**

Has a strong effect on the uncertainty, inherent in planning.

Complexity, however, is a relative measure that is affected by familiarity with past effort.

a) **Project complexity**

The first-time developer of a sophisticated e-commerce application might consider it to be exceedingly complex.

However, a software team developing its tenth e-commerce Web site would consider such work run of the mill.

a) **Project complexity**

A number of quantitative software complexity measures can be applied as per the need of project.

Such measures are applied at the design or code level and are therefore difficult to use during Software planning (before a design and code exist).

a) **Project complexity**

However, other, more subjective assessments of complexity (e.g., the function point complexity adjustment factors) can be established early in the planning process.

b) **Project size is another important factor that can affect the accuracy and efficacy of estimates.**

As size increases, the interdependency among various elements of the software grows rapidly.

Problem decomposition, an important approach to estimating, becomes more difficult because decomposed elements may still be alarming.

To paraphrase Murphy's Law: "What can go wrong will go wrong"- and if there are more things that can fail, more things will fail.

c) **The degree of structural uncertainty**

has an effect on estimation risk.

In this context, structure refers to the degree to which requirements have been solidified, the ease with which functions can be compartmentalized and the hierarchical nature of the information that must be processed.

d) **The availability of historical information**

has a strong influence on estimation risk.

By looking back, we can emulate things that worked and improve areas where problems arose.

e) **When comprehensive software metrics are available for past projects, estimates can be made with greater assurance, schedules can be established to avoid past difficulties, and overall risk is reduced.**

f) **Risk**

Is measured by the degree of uncertainty in the quantitative estimates established for resources, cost, and schedule.

If project scope is poorly understood or project requirements are subject to change, uncertainty and risk become dangerously high.

The software planner should demand completeness of function, performance, and interface definitions (contained in a *System Specification).*

The planner, and more important, the customer should recognize that variability in software requirements means instability in cost and schedule.

Summary

Summary

Lecture:30

Reference to Previous Session

Reference to Previous Session

3. **DECOMPOSITION TECHNIQUES**

Software Sizing

Software Sizing

The accuracy of a software project estimate is predicated on a number of things:

1. **The degree to which the planner has properly estimated the size *of* the product to be built.**

Software Sizing

2. **The ability to translate the size estimate into human effort, calendar time, and money (a function *of* the availability of reliable software metrics from past projects).**
3. **The degree to which the project plan reflects the abilities of the software team.**

Software Sizing

4. **The stability of product requirements and the environment that supports the software engineering effort.**

As project estimate is only as good as the estimate of the size of the work to be accomplished, sizing represents the project planner's first major challenge.

Software Sizing

In the context of project planning, size refers to a quantifiable outcome of the software project.

If a direct approach is taken, size can be measured in LOC. If an indirect approach is chosen, size is represented as FP.

3. **DECOMPOSITION TECHNIQUES**

Software Sizing

1. **"Fuzzy logic" sizing.**
2. **Standard component sizing.**

3. **Change sizing.**

Software Sizing

There are four different approaches to the sizing problem:

1. **"Fuzzy logic" sizing. This approach uses the approximate reasoning techniques that are the cornerstone of fuzzy logic.**

Software Sizing

1. **"Fuzzy logic" sizing.**

To apply this approach, the planner must identify the type of application, establish its magnitude on a qualitative scale, and then refine the magnitude within the original range.

Although personal experience can be used, the planner should also have access to a historical database of projects so that estimates can be com- pared to actual experience.

Software Sizing

2. **Function point sizing.**

The planner develops estimates of the information domain. Its characteristics will be discussed later in the session.

Software Sizing

3. **Standard component sizing.**

Software is composed of a number of different "standard components" that are generic to a particular application area.

Software Sizing

3. **Standard component sizing.**

Example: The standard components for an information system are subsystems, modules, screens, reports, interactive programs, batch programs, files, LOC, and object-level instructions.

The project planner estimates the number of occurrences of each standard component and then uses historical project data to determine the delivered size per standard component.

Software Sizing

3. **Standard component sizing.**

To illustrate, consider an information systems application:

The planner estimates that 18 reports will be generated.

Historical data indicates that 967 lines of Program code are required *per* report.

Software Sizing

3. **Standard component sizing.**

This enables the planner to estimate that 17,000 LOC will be required for the reports component. Similar estimates and computation are made for other standard components, and a combined size value (adjusted statistically) results.

Software Sizing

4. **Change sizing.**

This approach is used when a project encompasses the use of existing software that must be modified in some way as part of a project.

The planner estimates the number and type (e.g., reuse, adding code, changing code, and deleting code) of modifications that must be accomplished.

Software Sizing

4. **Change sizing.**

Using an "effort ratio" for each type of change, the size of the change may be estimated.

4. **Estimation – Tools**

5. **Estimation – Tools**

i. **Work Breakdown Structure (WBS)**

a) **Dividing into Logical Units/Tasks**

b) **Benefits of using a WBS**

c) **Measuring Effort for a Project**

d) **SLOC Technique**

e) **FP Technique**

f) **COCOMO Technique**

g) **Delphi Technique**

ii. **Work Breakdown Structure (WBS)**

a) **Dividing into Logical Units/Tasks**

i. **To correctly estimate the effort, size, or cost of completing a project, it is important to be aware of the multiple tasks that comprise a project.**

ii. **You can divide a project into logical units or tasks by using the WBS technique.**

iii. **Work Breakdown Structure (WBS)**

a) **Dividing into Logical Units/Tasks**

iii. **Creating a WBS is a prerequisite for any estimation activity.**

iv. **It enables you to conceptualize an abstract entity, such as a project, into distinct, independent units.**

i. **Work Breakdown Structure (WBS)**

a) **Dividing into Logical Units/Tasks**

v. **After dividing a project into tasks, you can categorize them as logical, broad tasks.**

Example: These tasks can be compiled under a common category, Marketing.

Drawing up a marketing strategy,

Planning a phase-wise product release &

Interacting with media agencies and the production department

b) **Benefits of Using a WBS**

Using a WBS provides a number of benefits to the management and to the development teams.

1. **First, it gives the management an idea about the size and complexity of the project.**
2. **Second, it helps in planning, scheduling, and monitoring a project realistically. This is possible because all the tasks in the project can be preformed measurable targets for each task.**

b) **Benefits of Using a WBS**

Helps in:

planning,

scheduling, and

monitoring a project

Tools :

Program Evaluation and Review Techniques (PERT),

Critical Path Method (CPM),

Timeline charts, and Gantt charts

b) **Benefits of Using a WBS**

These tools use WBS as the fundamental basis for:

Assessing resources to tasks,

Computing the number of days needed, and the

Cost required to complete the tasks.

ii. **Measuring Effort for a Project**

Measuring the effort for a project is a specialized activity.

It enables you to derive cost estimates that are critical for project management.

ii. **Measuring Effort for a Project**

An incorrect measurement of effort at the beginning of project can result in inaccurate project plans and frequent slippages.

Note: It can also lead to inaccurate cost estimates, which can cause steep cost deviations between estimated and actual cost values.

ii. **Measuring Effort for a Project**

There are many techniques, that you can use to accurately estimate effort, such as:

i. **Source Lines of Code (SLOC),**

ii. **Functional Point (FP), and**

iii. **Constructive Cost Model (COCOMO)**

These are quantitative estimation techniques because they rely on the use of formulae to calculate effort.

ii. **Measuring Effort for a Project**

iv. **Delphi Technique:**

In addition to quantitative estimation techniques, there is a human-based technique known as the Delphi technique.

This technique is based on soft skills and relies more on human factors, such as collecting information during group discussions.

a) **SLOC- Technique**

At the beginning of a software project, it is important to determine the size of the project.

The project size helps determine the:

Resources

Effort, and

Duration of the project.

There are many techniques to calculate the size of a software project.

a) **SLOC- Technique**

You can calculate the size by using a directly measurable technique, the SLOC technique.

It is defined as the source lines of code that are delivered as part of the product.

The effort spent on creating the source lines of code is expressed in relation to thousand lines of code (KLOC).

a) **SLOC- Technique**

The SLOC technique is an objective method of estimating the size because there are no multiple ways of calculating the lines of code.

Therefore, the effort estimate is close to being accurate.

a) **SLOC- Technique**

This technique includes:

The calculation of lines of code,

Documentation of pages,

Inputs, outputs, and

Components of a software program.

a) **SLOC- Technique**

Components are again of multiple types, such as:

reports,

screens, and

files.

It is also used to directly calculate the effort to be spent on a project.

Example of counting source lines of code:

The code displayed here contains four lines of code:

If (emp_code<=1200)

Print ("Welcome to the Inventory database.");

Else

Print ("Access denied to the Inventory databases.");

Counting SLOC

You can use the SLOC technique to estimate the effort required for a project when the programming language and the technology to be used are predefined.

In addition to the programming language and technology, the complexity and effort required to write a program should be easily predictable.

Counting SLOC

The use of the SLOC technique requires that the technology or language remain unchanged throughout the project.

Generally, you can use the SLOC technique when you are using third-generation languages, such as FORTRAN or COBOL.

Counting SLOC

While counting the source lines of code, there are some general considerations that you need to keep in mind:

i. **Only the delivered lines of code are included in SLOC calculation.**

Example: Test drivers and other support software are not part of the number of lines developed for a project.

Counting SLOC

ii. **Only the source lines of code written only by the development team as counted.**

This excludes the code created by applications generators.

ii. **Only declaration statements are counted as source lines of code.**

This excludes comments inserted to improve the readability of programs.

Disadvantages of Using SLOC

Despite being accurate in providing figures to calculate the effort required for a project, the SLOC technique has a drawback.

The SLOC technique is language-dependent.

The effort required to calculate source lines of code may not be the same for all languages.

Disadvantages of Using SLOC

Example: To conceive and write 8 lines of code that accomplish a task in the assembly language may require 15 minutes.

However, you may need only five minutes to complete the same lines of code if it is written in Visual Basic.

b) **FP Technique**

The FP technique is a direct indicator of the functionality of a software application from the user's perspective.

This is the most popular technique used to estimate the size of a software project.

b) **FP Technique**

“80% of the Fortune 500 are using function points, at least somewhere in their application development organizations".

You use the FP technique to estimate the total size of a project. The total size of a project is estimated as a single FP value.

After calculating the total size of a project in FP, you divide the total FP into the different phases of the SDLC.

b) **FP Technique**

This way, you can determine how much effort per FP is required in that particular phase.

Example: The testing phase is planned for 20 FP of work.

Similarly, you can express the cost required to complete FP of work for a particular phase.

At the end of a project, you can also express the number of defects reported in terms of per FP for a phase.

b) **FP Technique**

Size:

Small projects (10-99 Functional Points), variance of 7% from post-requirements estimates

Medium (100-999 Functional Points), 22% variance

Large (1000-9999 Functional Points) 38% variance

Very large (> 10K Functional Points) 51% variance

Features of Function Points

The total size of a software project is expressed in total function points.

It is independent of the:

computer language,

development methodology,

technology, or

capability of the project team developing the software project.

Features of Function Points

The specific user functionality of the application is evaluated in terms of relation to what is delivered by the application and, not how it is delivered.

Only user-requested and user-defined components are counted.

Features of Function Points

To calculate FP for a project, some major components are required.

The major components and their relationships are represented in the following figure:

Advantages of Using Function Points

i. **Function points are language-and technology-independent. Therefore, you can use them to estimate any kind of project.**

ii. **They can also be used to estimate the effort, cost, and schedules of projects that use the Prototyping and Spiral models because such projects have uncertain user and project requirements.**

Advantages of Using Function Points

iii. **You can use function points as a project estimation technique when you anticipate changes in the middle of a project.**

iv. **These changes may disturb the estimates if, you had used SLOC to estimate the effort, cost, or size of a project.**

v. **The FP estimation uses a subjective and holistic approach for project estimation.**

vi. **Consequently, the estimates calculated by using the FP are unlikely to be incorrect.**

Disadvantages of Using Function Points

i. **Estimation by using FP generally uses data from past projects for assigning weights to GSCs and the information domain values.**

ii. **To be able to do this realistically, it is important for the organization to have developed similar projects in the past.**

Lecture31

Reference to Previous Session

Reference to Previous Session

1. **Estimation – Concepts**
2. **Cone of uncertainty**
3. **Estimation – A Critical Factor**
4. **Decomposition Techniques**
5. **Estimation – Tools**

Work Breakdown Structure (WBS)

Measuring Effort for a Project

i. **Source Lines of Code (SLOC)**

ii. **Functional Point (FP)**

Estimation (Cont.)

c) **COCOMO Technique**

d) **COCOMO Technique**

COCOMO uses cost driver attributes to calculate the effort and duration of a project.

The COCOMO technique has three levels of implementation.

With each level, the complexity of the model increases. The levels of the COCOMO technique are:

i. **Basic**

ii. **Intermediate**

iii. **Advanced**

ii. **Basic COCOMO**

Estimates the effort and cost of a software project by using only the lines of code.

You, use basic COCOMO when you need a rough estimate of effort, such as during maintenance projects.

This is because in such projects, a majority of the work is already completed.

i. **Basic COCOMO**

Estimating the effort in the basic COCOMO technique involves three steps.

1. **Estimating the total delivered lines of code**
2. **Determining the effort constants based on the type of the project**
3. **Substituting values for lines of code and effort constant in a formula**

i. **Basic COCOMO**

The next step in the COCOMO model is to determine the type of the project being developed.

The basic COCOMO technique considers three types of projects to calculate effort.

a) **Organic**

b) **Embedded**

c) **Semidetached**

i. **Basic COCOMO**

a) **Organic projects have sufficient and defined objectives.**

The organizations that undertake organic projects have ample experience in development and use small development teams.

These are simple business and financial applications; such as a banking system and inventory system

i. **Basic COCOMO**

b) **Embedded projects have stringent and specialized hardware, software, and human resources requirements.**

Organizations usually have less experience in developing such projects.

Example: Real-time operating systems (RTOS), industrial automation systems, and sophisticated space and aviation systems.

i. **Basic COCOMO**

c) **Semidetached projects are a combination of the preceding two types of software projects.**

A new operating system and a database management system (DBMS) are examples of such projects.

i. **Basic COCOMO**

The last step in calculating effort by using the COCOMO technique is to substitute the values of lines of code and effort constants in the following formula:

$E = a1 * (KLOC)a2$

In the formula, E is the effort for a project. The effort constants, a1 and a2, depend on the type of project being developed.

ii. **Intermediate COCOMO**

Calculation of effort by using the intermediate COCOMO technique involves an additional step of calculating the effort adjustment factor (EAF).

The effort adjustment factor is calculated by assigning ratings to following cost driver attributes.

ii. **Intermediate COCOMO**

Cost Drivers

1. **Analyst Capability (ACAP)**
2. **Programmer Capability (PCAP)**
3. **Programming Language Experience (LEXP)**
4. **Virtual Machine Experience (VEXP)**

5. **Required Software Reliability (RELY)**
6. **Database Size (DATA)**
7. **Software Product Complexity (CPLX)**

ii. **Intermediate COCOMO**

Cost Drivers

8. **Execution Time Constraint (TIME)**
9. **Main Storage Constraint (STOR)**
10. **Computer Turnaround Time (TURN)**
11. **Virtual Machine Volatility (VIRT)**
12. **Use of Software Tools (TOOL)**
13. **Modern Programming Practices (MODP)**
14. **Required Development Schedule (SCED)**

ii. **Intermediate COCOMO**

These cost driver attributes relate to the various aspects of a software project, such as project, product, personnel, and computer attributes.

Using the intermediate COCOMO technique, you can accurately estimate effort and cost required for a project.

Accurate estimates are very helpful to start new development projects.

iii. **Advanced COCOMO**

The advanced COCOMO technique uses the steps of the intermediate COCOMO technique.

In addition, it uses costs driver attributes assigned to each phase of the SDLC such as analysis and design.

Applicability of COCOMO

COCOMO is flexible and capable of using SLOC, FP, and even object points.

Object points are measurable code sections in an object-oriented programming language, such as C++, and Java.

You can use COCOMO when the size of a project is extensive and the requirements of the project are vague.

In contrast, SLOC and FP can be used for projects where either the requirements are more or less known or developers possess the relevant experience in developing projects.

Applicability of COCOMO

COCOMO is suitable for complex and sophisticated projects that are expected to operate within intensive hardware, software, and personnel constraints.

Generally, you can use COCOMO when the software development environment is new to an organization.

In addition, you can use COCOMO when you do not have baseline data about past projects.

Applicability of COCOMO

However, you need complete data about your current project to assign weightage to each cost driver attribute.

You can use FP or SLOC techniques when you have enough past project data to assign accurate weightage to the 14 GSCs (General System Characteristics) and the various information domain value elements.

Applicability of COCOMO

General System Characteristic

1. **Data Communication**
2. **Distributed Functions**
3. **Performance**
4. **Heavily Used Configuration**
5. **Transaction Rate**
6. **On-line Data Entry**

7. **End-use Efficiency**

Applicability of COCOMO

8. **On-Line Update**
9. **Complex Processing**
10. **Reusability**
11. **Installation Ease**
12. **Operational Ease**
13. **Multiple Sites**
14. **Facilitates Change**

d) **Delphi Technique**

e) **Delphi Technique**

It's a Human-based estimation technique.

Human-based estimation techniques use human experience and analytical skills to estimate the size, productivity, and effort required for a project.

This is a trusted technique and is widely used in many established organizations to facilitate practical and reasonable estimation.

d) **Delphi Technique**

The rationale of using the Delphi technique is that when many experts independently arrive at the same estimate on the basis of similar assumptions, the estimate is likely to be correct.

Steps of the Delphi Technique

Three distinct groups of people need to be present for estimation

Estimation experts

Estimation coordinator

Author

The Delphi technique has eight basic steps.

Applicability of the Delphi Technique

The Delphi technique is a simple and subjective method of estimation.

However, it is a very effective method because most of the estimates are tried and tested.

You can use this method if the project is small or if you have the data and expertise that can enable unambiguous estimates.

5. **Key Estimation Methodologies**

6. **Key Estimation Methodologies**

i. **Top-down**

ii. **Bottom-up**

iii. **Expert Judgment**

iv. **Estimation by Analogy**

v. **Top-down Estimation**

Based on overall characteristics of project

Some of the these other techniques can be "types" of top-down:

Analogy,

Expert Judgment, and

Algorithmic methods

i. **Top-down Estimation**

Advantages

Easy to calculate

Effective early on (like initial cost estimates)

Disadvantages

Some models are questionable or may not fit

Less accurate because it doesn't look at details

ii. **Bottom-up Estimation**

Create WBS

Add from the bottom-up

Advantages

Works well if activities are well understood

Disadvantages

Specific activities not always known

More time consuming

iii. **Expert Judgment**

Use somebody who has recent experience on a similar project

Accuracy depends on their 'real' expertise

Comparable application(s) must be accurately chosen

Systematic

Can use a weighted-average of opinions

iv. **Estimation by Analogy**

Use past project

Must be sufficiently similar (technology, type, organization)

Find comparable attributes (ex: # of inputs/outputs)

Can create a function

iv. **Estimation by Analogy**

Advantages

Based on actual historical data

Disadvantages

Difficulty 'matching' project types

Prior data may have been mis-measured

How to measure differences – no two exactly same

6. **Over and Under Estimation**

Over estimation issues

The project will not be funded

Conservative estimates guaranteeing 100% success may mean funding probability of zero.

Parkinson's Law: Work expands to take the time allowed

Danger of feature and scope creep

Be aware of "double-padding": team members & manager

6. **Over and Under Estimation**

Under estimation issues

Quality issues (short changing key phases like testing)

Inability to meet deadlines

Morale and other team motivation issues

7. **Estimation "Presentation"**

How you present the estimation can have huge impact

- **Techniques**

i. **Plus-or-minus qualifiers**

6 months +/-1 month

i. **Ranges**

6-8 months

8. **Net Present Value Analysis: NPV**

NPV: a method of calculating the expected net monetary gain or loss from a project by discounting all expected future cash inflows and outflows to the present point in time

Projects with a positive NPV should be considered if financial value is a key criterion

The higher the NPV, the better

Summary

Summary

Lecture:32

- **Reference to Previous Session**
- **Software Project Management**
  Key Topics

1. **Overview of Project Management**
2. **Software Development Fundamentals**
3. **Processes**
4. **Organization**
5. **Planning**
6. **Estimation**
7. **Work Breakdown Structure**
8. **Scheduling**
9. **Risk and Change Management**
10. **Quality**
11. **Application Tools**
12. **Commissioning & Migration**

- **Software Project Management**
  Key Topics
- **Work Breakdown Structure (WBS)**

1. **WBS Preview**
    i. **Salient features**
    ii. **What is it?**
    iii. **When is it produced?**
    iv. **What it contains?**

v. **Use of WBS**

2. **WBS-A Mandatory Management Tool**

i. **The WBS as a project management tool**

ii. **Characteristics of a High-Quality WBS**

iii. **Sample WBS**

iv. **Types of WBS**

- **Work Breakdown Structure (WBS)**

3. **WBS- Major Steps**

i. **Project Decomposition**

ii. **Functional Decomposition**

iii. **Design Decomposition**

iv. **Develop Project Tasks**

v. **Define Project Development Phases**

vi. **Define Task Relationships**

vii. **Defining Deliverables**

viii. **Creating a WBS**

a) **Brainstorming to arrive at board tasks for a project**

b) **Refine the board tasks**

c) **Categorize tasks into logical task headers**

4. **WBS Implementation**

i. **WBS & Methodology**

ii. **WBS Techniques**

5. **Guidelines for decomposition of work tasks**

1. **WBS Preview**

2. **Salient features**

3. **What is it?**
4. **When is it produced?**
5. **What it contains?**
6. **Use of WBS**

i. **WBS - Salient Features**

a. **Large, complex projects are organized and comprehended by breaking them into progressively smaller pieces until they are a collection of defined "work packages" that may include a number of tasks.**

- **A $1,000,000,000 project is simply a lot of $50,000 projects joined together.**

b. **Work breakdown structure (WBS) is a technique to decompose the project for the purpose of management and control.**

i. **WBS - Salient Features**

c. **It provides the framework for organizing and managing the work.**

i. **A project may be divided into subprojects.**

ii. **Subprojects are subdivided into smaller, more manageable work components (work packages) at the lowest level.**

iii. **The work packages may be further decomposed to project activities.**

i. **WBS - Salient Features**

d. **The WBS is commonly used at the beginning of a project for defining project scope, organizing Gantt schedules and estimating costs.**

e. **It lives on, throughout the project, in the project schedule and often is the main path for reporting project costs.**

f. **Includes activities like management, procurement, installation, software development etc.**

i. **WBS - Salient Features**

g. **Many of the WBS development tasks are derived from the development method that will be used, and from the design and architecture of the system.**

h. **WBS is related to planning and scheduling a project**

i. **It is a functional decomposition of the tasks of the project.**

i. **WBS - Salient Features**

j. **It starts with the end objective required and successively subdividing it into manageable components in terms of size and complexity of:**

- **Program,**
- **Project,**
- **System,**
- **Subsystem,**
- **Components,**
- **tasks, subtasks, and**
- **work elements.**

i. **WBS - Salient Features**

k. **The WBS is first and foremost a technical data gathering structure, developed so that the achievement in technical progress can be measured and analyzed against a formal baseline plan.**

l. **The WBS aids the customer in understanding the status of the project as time elapses.**

i. **WBS - Salient Features**

m. **The WBS aids the customer's customer in understanding the status of the project.**

n. **All managers, internal and external need to use the planning and status information within the WBS structure to:**

- **Aid in the adjustment to the current program paths and**
- **For maximizing the attainment of short term and long term goals**.

ii. **WBS-What is it?**

a. **A definition of a project in terms of it's work**

b. **A deliverable-oriented grouping of project elements that organizes and defines the total scope of the project.**

c. **Each descending level represents an increasingly detailed definition of the project's work.**

d. **An outline of the work of the project, not the work itself.**

e. **Created by those doing the work –may include all functional stakeholders.**

iii. **WBS- What it Contains?**

a. **Maps all contractual obligations (SOW) regarding deliverables**

b. **Details project objectives**

c. **Detailed enough to meet performance (measurable) objectives**

d. **Contains built-in WBS and Project Plan review and update**

iv. **WBS- When is it produced?**

- **The WBS is produced following the development of the scope statement, before the schedule.**
- **It is a "bridging" document between the Scope and Schedule. (what and when)**

v. **Uses of WBS**

a. **Defines 100% of the scope and can communicate the scope of the project without the presence of the scope statement or document.**

b. **Communicates effectively to all stakeholders**

c. **Defines and clarifies**

- **Boundaries (Life cycle of the project – not the product)**
- **Deliverables**
- **Refines Scope**

v. **Uses of WBS**

d. **Defines scope in project management language (perhaps down to work package level which may lead to development of project schedule)**

e. **Is not a single document that can be mistaken for the project plan, schedule or scope statement**

2. **WBS- A Mandatory Management Tool**

3. **WBS- A Mandatory Management Tool**

i. **The WBS as a project management tool**

ii. **Characteristics of a High-Quality WBS**

iii. **Sample WBS**

iv. **Types of WBS**

i. **WBS as a Project Management Tool**

- **A WBS is a valuable management tool that is used throughout all life-cycle phases to:**

a. **Manage Risk**

- **It helps in managing risk by providing insight into technical aspects of program management.**
- **WBS as a Project Management Tool**

b. **Manage Risk**

- **Benefits all acquisition disciplines (e.g., program management, contracting, logistics, finance and budgeting, etc.)**
- **A WBS can be used to identify issues and concerns. By identifying all possible issues and concerns, Program Managers can reduce risk.**

i. **WBS as a Project Management Tool**

b. **Manage Costs**

- **A WBS can be used to help make program management decisions.**
- **For example, if the costs of an element in the WBS are too high, the WBS can be used to identify possible tradeoffs.**
- **Identifying and analyzing tradeoffs can help the manager decide how best to stay within budget.**

i. **WBS as a Project Management Tool**

c. **Assign Work**

- **The WBS is also useful for determining an acquisition strategy and/or assigning work.**
- **The information contained in the WBS can help a Program Manager develop a statement of work that describes what products or services are to be delivered.**

d. **Schedule and Track activities**

- **A schedule of key events can be developed for each element in the WBS.**
- **Completion of these key events is then tracked.**

i. **WBS as a Project Management Tool**

d. **Schedule and Track activities**

e. **The work breakdown structure defines all tasks to be performed during the development of the project. This will include tasks from such project categories as:**

- **Software development**
- **Installation**
- **Maintenance**
- **Management**
- **Training**
- **Procurement**
- **Documentation**

ii. **WBS as a Project Management Tool**

e. **Align with Terms of Reference and scope of project**

- **At all times, any, work being performed by a member of the software project team must be part of a WBS task: No member of the team should ever perform any task that does not appear in the WBS list of tasks.**
- **The WBS is essentially a management tool that provides the ability to assign well-define tasks to members of the development team.**

i. **WBS as a Project Management Tool**

e) **Align with Terms of Reference and scope of project**

f) **It is through the WBS that progress is monitored as tasks are completed and potential problems are discovered.**

- **New tasks that were overlooked are identified, and estimates are revised based on the actual resources used for completed tasks.**

i. **WBS as a Project Management Tool**

f. **Report Expense**

- **The WBS is also a budgetary tool that provides a means of charging each development activity to the appropriate section in the project budget.**

- **This is one of the basic methods for planning and monitoring project expenditure.**

i. **WBS as a Project Management Tool**

f. **Report Expense**

- **There are mainly computerized utilities available to support the maintenance of the WBS.**
- **These utilities run both on small PC type computers and large mainframes.**
- **WBS utilities are often available as part of a manager's general planning utility, and provide other scheduling and monitoring features, such as PERT analysis and report generation.**

- **SUMMARY**
- **SUMMARY**

1. **WBS Preview**
    i. **Salient features**
    ii. **What is it?**
    iii. **When is it produced?**
    iv. **What it contains?**
    v. **Use of WBS**
2. **WBS-A Mandatory Management Tool**
    i. **The WBS as a project management tool**

Lecture:33

- **Reference to Previous Session**
- **WBS**

1. **WBS Preview**
    i. **Salient features**
    ii. **What is it?**
    iii. **When is it produced?**
    iv. **What it contains?**

v. **Use of WBS**

2. **WBS-A Mandatory Management Tool**

i. **The WBS as a project management tool**

g. **Mapping WBS for Cost Management**

ii. **Characteristics of a High-Quality WBS**

iii. **Sample WBS**

iv. **Types of WBS**

- **WBS (Cont.)**

3. **WBS- Major Steps**

i. **Project Decomposition**

ii. **Functional Decomposition**

iii. **Design Decomposition**

iv. **Develop Project Tasks**

v. **Define Project Development Phases**

vi. **Define Task Relationships**

vii. **Defining Deliverables**

viii. **Creating a WBS**

a) **Brainstorming to arrive at board tasks for a project**

b) **Refine the board tasks**

c) **Categorize tasks into logical task headers**

4. **WBS Implementation**

i. **Implementation**

ii. **Methodology**

5. **Guidelines for decomposition of work tasks**

i. **General Guidelines**

ii. **A typical list of high level WBS tasks**

iii. **Management and Administration Tasks**

- **Reference to Previous Session**

1. **WBS Preview**

i. **Salient features**

ii. **What is it?**

iii. **When is it produced?**

iv. **What it contains?**

v. **Use of WBS**

i. **WBS as a Project Management Tool**

g) **Mapping WBS for Cost Management**

– **In a product-oriented WBS, functional categories of work may form "cost accounts" within a WBS element.**

– **Cost account managers are responsible for a functional area's contribution to a WBS element.**

– **Cost accounts from several departments or functions may combine into one WBS element.**

i. **WBS as a Project Management Tool**

g) **Mapping WBS for Cost Management**

– **Internal department planning for a cost account will be made up of individual work packages.**

– **A work package will typically have its own budget and schedule.**

i. **WBS as a Project Management Tool**

g) **Mapping WBS for Cost Management**

– **Work packages should be small enough to be executed by individuals or small groups in a single department, and they should be of relatively short schedule duration.**

– **A small project might define a maximum work package size as two weeks of effort.**

– **Larger projects will assemble larger work packages that can be appropriately managed and controlled.**

i. **WBS as a Project Management Tool**

g) **Mapping WBS for Cost Management**

– **The project manager will have to decide to what degree employment of various details of WBS implementation will benefit the efficient management of the project.**

– **On a very small project, a formal WBS may serve no useful purpose, but it can become valuable if project size or complexity start to increase.**

i. **WBS as a Project Management Tool**

g) **Mapping WBS for Cost Management**

– **As an organization's project management environment matures, or as larger size and complexity are encountered, application of the WBS concept can evolve from an *ad hoc* list of tasks, to time-phased activity lists, task lists clustered by project deliverables and services, or an end-product focused WBS fed by cost accounts and work packages.**

i. **WBS as a Project Management Tool**

WORD OF CAUTION

i. **WBS as a Project Management Tool**

WORD OF CAUTION

- **The WBS is a useful tool provided it is constantly updated.**
- **It should be updated periodically together with the project development plan and the project schedule.**
- **It is reasonable to expect the WBS list to have tasks added, modified or even removed as project development progresses.**
- **Summary**
- Summary

2. **WBS-A Mandatory Management Tool**

    i. **The WBS as a project management tool**

        g. **Mapping WBS for Cost Management**

Lecture 34:

- **Reference to Previous Session**
- **WBS**

1. **WBS Preview**
    i. **Salient features**
    ii. **What is it?**
    iii. **When is it produced?**
    iv. **What it contains?**
    v. **Use of WBS**
2. **WBS-A Mandatory Management Tool**
    i. **The WBS as a project management tool**
        g. **Mapping WBS for Cost Management**
    ii. **Characteristics of a High-Quality WBS**
    iii. **Types of WBS**
    iv. **Sample WBS**
    v. **WBS (Cont.)**
3. **WBS- Major Steps**
    i. **Project Decomposition**
    ii. **Functional Decomposition**
    iii. **Design Decomposition**
    iv. **Develop Project Tasks**
    v. **Define Project Development Phases**
    vi. **Define Task Relationships**
    vii. **Defining Deliverables**
    viii. **Creating a WBS**
        a) **Brainstorming to arrive at board tasks for a project**
        b) **Refine the board tasks**
        c) **Categorize tasks into logical task headers**

4. **WBS Implementation**

    i. **Implementation**

    ii. **Methodology**

5. **Guidelines for decomposition of work tasks**

    i. **General Guidelines**

    ii. **A typical list of high level WBS tasks**

    iii. **Management and Administration Tasks**

        - **Reference to Previous Session**

2. **WBS-A Mandatory Management Tool**

    i. **The WBS as a project management tool**

        g. **Mapping WBS for Cost Management**

a. **Review and signoff from top to bottom**

b. **Includes logical flow and is hierarchical in nature**

c. **Clear and concise**

d. **Provides ability to roll-up information to higher levels**

e. **Receives 100% team buy-in**

f. **Should have at least 2 levels**

- **Level 1 defines 100% of the service/product/result**
- **Level 2 defines the deliverables in terms of work (groupings)**
- **Project Management (and sub-contract management) at Level 2.**
- **The deliverables in the WBS must match the scope or contract (WBS should not contain work that is not defined in the scope – Scope should not describe work not contained in the WBS)**

h. **All deliverables should be accounted for regardless of responsibility**

i. **Every WBS element should be clearly defined – or should be clarified in the WBS Dictionary**

ii. **Characteristics of a High Quality WBS**

k. **Features**

- **The WBS should contains 100% of the work defined by the scope or contract**
- **Development of WBS should involve the entire project team**
- **Should be deliverable-oriented**
- **Should captures all deliverables (Internal, External, Interim) in terms of work to be completed**

ii. **Characteristics of a High Quality WBS**

l. **Usefulness**

- **Should defines the context of the project and clarifies the work**
- **Should communicates project scope to all stakeholders in terms of the work to be completed**
- **Is "in sync" with the scope statement and project schedule**
- **Implies and allows for continual improvement/update of the WBS to maintain current-ness and "vitality" within the project**

ii. **Characteristics of a High Quality WBS**

m. **WBS Is Not**

- **A single document that substitutes for the project schedule or project plan**
- **A listing of tasks or activities**

iv. **Types of WBS**

a) **Contract WBS (CWBS)**

- **First 2 or 3 levels**
- **High-level tracking**

b) **Project/Product WBS (PWBS)**

- **Defined by PM and team members**
- **Tasks tied to deliverables**
- **Lowest level tracking**

iv. **Types of WBS**

a) **Contract WBS:**

i. **If a project involves several organizational participants or contractors, guidance for one contractor can be provided in a Contract WBS (CWBS).**

ii. **The project manager may provide a high-level CWBS for each developer, perhaps to level 2 or level 3.**

iv. **Types of WBS**

a) **Contract WBS:**

iii. **The developer will then fill in the details of lower WBS levels to reflect the work to be accomplished and the data flow in that organization.**

iv. **The extended CWBS must include the levels at which required reporting information is summarized for submittal to the Owner (The Owner). One and only one CWBS exists for each contract.**

iv. **Types of WBS**

a) **Contract WBS:**

v. **The adequate number of levels of each CWBS leg extension is determined by the contractual work scope, the negotiated cost performance report**

vi. **Extends the Program WBS to a lower level in order to provide management and cost information to the Government.**

vii. **Includes all the elements for products (e.g., hardware, software, data, or services) that are the responsibility of the contractor.**

viii. **Must be consistent with the Program WBS.**

iv. **Types of WBS**

a) **Contract WBS:**

ix. **Contractor's Responsibility**

- **Contractors may extend the work breakdown structure to whatever level they feel is necessary to manage the program.**
- **Contractors also use the Contract WBS to define work packages.**
- **Example:**
  - **Discrete portions of the project that can be charged to a single organization.**

- **Building blocks for program management and cost reporting.**

iv. **Types of WBS**

a) **Contract WBS:**

Contractor's Responsibility

- **Remember that a Contract WBS:**
  - **Is prepared and maintained by each contractor working on different portions of the program.**
  - **Includes the number of levels thought sufficient by the contractor to manage the program.**
  - **Must be updated if changes are made to the Program WBS.**

iv. **Types of WBS**

b) **Product/Project breakdown structure (PWBS)**

i. **A Product/Project breakdown structure (PWBS) is a hierarchy where the project is decomposed into product-oriented components when defining project scope.**

ii. **It defines and describes project deliverable products and product constituent components.**

- Summary
- Summary
  i. **Characteristics of a High-Quality WBS**
  ii. **Types of WBS**
  iii. **Sample WBS**

Lecture35:

- **Software Project Management**
- **Reference to Previous Session**

1. **WBS Preview**
2. **WBS-A Mandatory Management Tool**
   i. **The WBS as a project management tool**

ii. **Characteristics of a High-Quality WBS**

iii. **Types of WBS**

iv. **Sample WBS**

- **WBS (Cont.)**

2. **WBS-A Mandatory Management Tool**

iv. **Sample WBS (Cont.)**

3. **WBS- Major Steps**

iv. **Key Steps**

a) **Project Decomposition**

b) **Functional Decomposition**

c) **Design Decomposition**

d) **Develop Project Tasks**

0.0 Retail Web Site

1.0 Project Management

2.0 Requirements Gathering

3.0 Analysis & Design

4.0 Site Software Development

4.1 HTML Design and Creation

4.2 Backend Software

4.2.1 Database Implementation

4.2.2 Middleware Development

4.2.3 Security Subsystems

4.2.4 Catalog Engine

4.2.5 Transaction Processing

4.3 Graphics and Interface

4.4 Content Creation

5.0 Testing and Production

1.10 MANAGEMENT

1.1 Plan Project

1.1.1 Develop Project Plan

1.1.2 Update Project Plan

1.2 Track Project

1.2.1 Prepare status reports

1.2.2 Collect/analyze project metrics

1.3 Perform Quality Activities

1.3.1 Prepare QA Plan

1.3.2 Conduct Reviews

1.3.3 Conduct Audits

1.4 Perform Configuration Management

1.4.1 Prepare CM Plan

1.4.2 Develop Project Library

1.4.3 Manage Change Board

1.4.4 Maintain Configuration Items

2.0 DESIGN

2.1 Prepare Preliminary Design

2.1.1 Develop Enterprise Architecture

2.1.2 Prepare Data Flow Diagrams

2.1.3 Prepare Logical Data Model

2.2 Prepare Detailed Design

2.2.1 Prepare Physical Data Model

2.2.2 Prepare Data Dictionary

2.3 Document Design

2.3.1 Develop Design Specification

2.4 Review Design

3.0 DEVELOPMENT/INTEGRATION

3.1 Develop Software

3.1.1 Develop Server Application

3.1.2 Develop User Interface

3.1.3 Develop XYZ Interface

3.2 Procure Hardware

3.2.1 Procure Server

3.2.2 Procure Workstations

3.3 Procure Software Packages

3.3.1 Procure Database

3.3.2 Procure User Interface Building Tool

3.3.3 Procure Operating System

3.4 Perform Integration Testing

3.5 Convert Data

3.5.1 Develop Conversion Plan

3.6 Develop User Manual

3.7 Transition Management

4.0 ACCEPTANCE TESTING

4.1 Plan Acceptance Test

4.2 Conduct Acceptance Test

4.3 Develop Test Report

5.0 INSTALLATION

5.1 Develop Installation Plan

5.2 Site Preparation

5.3 Install at Locations

5.3.1 Headquarters

5.3.2 Site 1

6.0 MAINTENANCE

6.1 Hardware Maintenance

6.2 Software Maintenance

- **Tasks shown as blue are fixed costs of the project. These should be estimated soon.**
- **Tasks shown as black are proportional to the size of the requirements. A more detailed work breakdown structure will be produced for these tasks when the requirements are better known.**
- **Tasks shown as red depend on the defect rate in the delivered code. These will be estimated after 12 months experience.**
- **TIP:**
- **Label each step uniquely to show its position in the WBS, e.g., Step 1.1.4.A.**
- **Use numbers for steps that you intend to do in sequence, and use letters for steps that you intend to do in parallel.**
- **Ex: Step 1.1 comes before Steps 1.2.A and 1.2.B, but those two steps may be done in parallel, and Step 1.3 will be done after all 1.2.* steps have been finished. Don't worry about renumbering if you delete a step.**

i. **Key steps**

a) **Project Decomposition**

b) **Functional Decomposition**

c) **Design Decomposition**

d) **Develop Project Tasks**

e) **Define Project Development Phases**

f) **Define Task Relationships**

g) **Defining Deliverables**

ii. **Creating a WBS**

a) **Brainstorming to arrive at board tasks for a project**

b) **Refine the board tasks**

c) **Categorize tasks into logical task headers**

a) **Project Decomposition**

i. **Just like any other large complex task, the development of a software project is more easily managed with the divide and conquer approach.**

ii. **Stepwise refinement, when applied to a software project, produces all the low level work tasks and includes:**

- **Development tasks**
- **Managerial tasks**
- **Support tasks and**
- **Administrative tasks**

a) **Project Decomposition**

iii. **The WBS list of project tasks is derived from the project's statement of work (the SOW) that defines the scope of the project.**

iv. **The SOW is usually prepared before the official launching of the project, and is often part of the project contract between the customer and the developer.**

a) **Project Decomposition**

v. **For internal projects, when an organization is funding its own development work, the SOW becomes synonymous with the Project definition specification or a similar document that defines the scope of work for the software project manager.**

b) **Functional Decomposition**

i. **The functional decomposition of a software project is a division of the system into its operational components as they are seen by the user.**

ii. **Functional decomposition is part of the requirements phase of a project.**

iii. **The objective of this phase is to define the characteristics of the system from the user's perspective.**

b) **Functional Decomposition (Cont.)**

iv. **Ex: Automatic bank teller system**

- **Ability to communicate on-line between remote automatic tellers and the bank's central computer in order to provide updated account information is a functional characteristic or the system.**
- **This will usually be defined during the requirements phase of the development cycle.**

b) **Functional Decomposition (Cont.)**

iv. **Ex: Automatic bank teller system**

- **However, the method of transmission between the automatic teller and the central computer is not a functional characteristic of the system, as this is internal to the design and implementation of the system and is not apparent to the user.**
- **The method of transmission, including the communications protocol, will usually be defined during the design phase of the development of the system.**
- **Functional Decomposition**

b) **Functional Decomposition (Cont.)**

v. **Just as the requirements phase precedes the design phase, so the functional decomposition of a software system will usually precede the design decomposition.**

vi. **The functional decomposition will often provide much of the information necessary for the subsequent division of the system into the implementation components.**

b) **Functional Decomposition (Cont.)**

vii. **In fact, the functional decomposition is often a good place to start when designing a software system, as the major functional components of a system will often correspond to the initial division of the system into subsystems or high level components.**

c) **Design Decomposition**

i. **The design decomposition of a software system is a division of the system into lower level components that coincide with the programming components of the system.**

ii. **The work breakdown structure (WBS) is the decomposition of a software project into low level work tasks.**

iii. **An important point to remember is that in design decomposition, only the lower level components are actually implemented.**

c) **Design Decomposition (Cont.)**

iv. **Higher level components represent a group of lower level components.**

v. **Design decomposition basically produces two types of system component:**

- **high level components**
- **low level modules**

v. **Different software development standards use different terminology to identify the various levels of decomposition.**

c) **Design Decomposition (Cont.)**

vii. **Following Figure presents an example of the design decomposition of an automatic bank teller system into lower levels of design components.**

viii. **On the third level, the Automatic teller component decomposes into the Hardware interfaces, and the Teller logic.**

ix. **The next level may then decompose the Hardware interfaces into the Keyboard driver, the Display driver, the Printer, driver and the Beeper.**

x. **At this level, these drivers may represent actual software modules.**

c) **Design Decomposition**

d) **Design Decomposition (Cont.)**

xi. **A fully decomposed system, with all its low level components, is not always easy to grasp.**

xii. **This is especially true during the presentation of the system at a project review, when the system needs to be quickly understood by people who have not been involved in its design.**

c) **Design Decomposition (Cont.)**

xiii. **On such occasions, the stepwise refinement technique is a convenient method for gradually presenting progressive detail by initially showing the first decomposition level, and then slowly revealing subsequent levels.**

(Guidelines for decomposition to be discussed at end of topic)

c) **Design Decomposition (Cont.)**

xiv. **This is demonstrated in following figure.**

- **At a convenient intermediate decomposition step we can divide the design in two: the upper levels and the lower levels.**
- **This is used particularly when the design phase is implemented in two distinct stages: top level design and detailed design (see Fig.).**

d) **Develop Project Tasks**

i. **WBS tasks are developed by asking,**

- **"What tasks need to be done to accomplish the project objective?"**

ii. **The choice of WBS structure is subjective and reflects the preferences and judgment of the project manager.**

iii. **As levels of the WBS become lower, the scope, complexity, and cost of each subtask become smaller. The lowest level tasks, or work packages, are independent, manageable units that are planned, budgeted, scheduled, and controlled on their own.**

d) **Develop Project Tasks (Cont.)**

iv. **As efforts of similar scope and type are planned, the basic WBS tasks remain fairly similar, but each project requires a specific set of tasks that address the uniqueness of the project's requirements.**

v. **Certain top level elements, such as project management, are included in the WBS of every project, regardless of its type, size, or complexity. Other items, like installation, may not apply to every project.**

d) **Develop Project Tasks (Cont.)**

vi. **The initially developed WBS evolves over the course of the planning.**

vii. **It is highly probable that it will look quite different as the scheduling, estimation, and resource allocation portions of the plan are completed.**

d) **Develop Project Tasks (Cont.)**

viii. **One of the difficult parts of talking about IT projects generically is the wide range of such projects. Typically, in a small project, there is a single project development phase.**

ix. **In large or complex systems, however, there are often multiple development phases, where different functional requirements are met.**

d) **Develop Project Tasks (Cont.)**

x. **Sometimes these phases are driven by the need to achieve certain levels of functionality prior to the availability of the complete system.**

xi. **Other times, the phases are defined to partition the development effort and to reduce the risks associated with larger project efforts.**

- **Summary**

2. **WBS-A Mandatory Management Tool**

    iv. **Sample WBS**

3. **WBS- Major Steps**

    iv. **Key Steps**

        a) **Project Decomposition**

        b) **Functional Decomposition**

        c) **Design Decomposition**

        d) **Develop Project Tasks**

Lecture:36

- **Software Project Management**
- **Reference to Previous Session**

2. **WBS-A Mandatory Management Tool**

    iv. **Sample WBS (Cont.)**

3. **WBS- Major Steps**

    iv. **Key Steps**

        a) **Project Decomposition**

        b) **Functional Decomposition**

        c) **Design Decomposition**

        d) **Develop Project Tasks**

- **WBS (Cont.)**

3. **WBS- Major Steps**

    i. **Key Steps**

e) **Define Project Development Phases**

f) **Define Task Relationships**

g) **Defining Deliverables**

ii. **Creating a WBS**

e) **Brainstorming to arrive at board tasks for a project**

f) **Categorize tasks into logical task headers**

g) **Refine the board tasks**

- **WBS (Cont.)**

4. **WBS Implementation**

i. **Implementation**

ii. **Methodology**

5. **Review WBS**

e) **Define Project Development Phases**

i. **For large systems, the decomposition of the system into smaller components needs to be done early in the planning cycle.**

ii. **The rationale for the decomposition must be known, otherwise, different results derived from different reasons for the system decomposition may occur.**

- **Example: If a phase is defined to accommodate user needs, the phase may cross multiple functional areas of the system.**

e) **Define Project Development Phases (Cont.)**

iii. **If, on the other hand, a system is divided into phases simply to reduce risk, a functional division might occur where the phases represent completion of entire functional areas of the system.**

iv. **The way in which the phases are handled, differs with each application.**

v. **Often, phases are handled as top level WBS elements, with tasks associated with each phase defined.**

f) **Define Task Relationships**

i. **If a project is broken down into phases, be sure that the WBS reflects this.**

ii. **The WBS structure denotes a hierarchy of task relationship. Subtask completion eventually rolls up into task completion, which ultimately results in project completion.**

iii. **There can, however, also be relationships between tasks that are not within the outlined hierarchy.**

f) **Define Task Relationships (Cont.)**

iv. **These relationships need to be noted, and the ultimate structuring of the tasks optimized to favor a minimum of horizontal dependencies and relationships.**

v. **If the tasks are not organized efficiently, it becomes difficult to schedule and allocate resources to the tasks.**

f) **Define Task Relationships (Cont.)**

vi. **The project scope of work is an iterative process that is generally done by the project team with the use of a Work Breakdown Structure (WBS), allowing the team to capture and then decompose all of the work of the project.**

f) **Define Task Relationships (Cont.)**

vii. **A WBS is a deliverable-oriented grouping of project components that organizes and defines the total scope of the project; work not in the WBS is outside the scope of the project.**

viii. **As with the scope statement, the WBS is often used to develop or confirm a common understanding of project scope.**

ix. **Each descending level represents an increasingly detailed description of the project deliverables.**

f) **Define Task Relationships (Cont.)**

x. **A WBS is normally presented in chart form, however, the WBS should not be confused with the method of presentation — drawing an unstructured activity list in chart form does not make it a WBS.**

xi. **Each item in the WBS is generally assigned a unique identifier; these identifiers can provide a structure for a hierarchical summation of costs and resources.**

f) **Define Task Relationships (Cont.)**

xii. **The items at the lowest level of the WBS are work packages, especially in organizations that follow earned value management practices.**

xiii. **These work packages may in turn be further decomposed in a subproject work break-down structure.**

f) **Define Task Relationships (Cont.)**

xiv. **Generally, this type of approach is used when the project manager is assigning a scope of work to another organization, and this other organization must plan and manage the scope of work at a more detailed level than the project manager in the main project.**

g) **Defining Deliverables**

i. **Deliverables associated with each task are shown in the WBS and are reflected in the Deliverables portion of the Project Plan.**

ii. **A sample of a Deliverables template is shown next.**

iii. **All deliverables are listed as they are identified.**

iv. **As the schedule is completed, the due date is filled in, and responsibility for the deliverable is assigned as it is known (typically when the organization chart is defined).**

g) **Defining Deliverables (Cont.)**

v. **The date delivered is a field that is filled in as deliveries are made.**

vi. **Over the course of the project, a comparison of the due date and the date delivered provides a metric for how well deliverable dates are met by the project team.**

g) **Defining Deliverables (Cont.)**

h) **Defining Deliverables (Cont.)**

vii. **While the deliverables list is a compilation of information identified in the WBS and the project schedule, it is useful to maintain a separate list since deliverable completion can be a key metric of project progress.**

viii. **Separate tracking of deliverables can help keep a project on track.**

ix. **It also serves as a useful communication tool with both stakeholders and the project team.**

ii. **Creating a WBS**

i. **A project can be compared to a large system.**

ii. **A large system consists of multiple, independent subsystems that achieve a common goal.**

iii. **Similarly, a project consists of small, independent tasks.**

iv. **Each task can be subdivided into sub tasks.**

ii. **Creating a WBS (Cont.)**

v. **For example, in a general software project, a task is to perform project analysis.**

vi. **You may also consider studying the organizational objectives and preparing a project proposal to present to the client.**

vii. **Therefore, in a project analysis task, there are three subtasks. A subtask is also known as a work package.**

viii. **A work package is a unit-level entity in a project system.**

ii. **Creating a WBS (Cont.)**

ix. **Three steps are generally used to create WBS but they can vary in relation to an individual or an organization.**

a) **Brainstorming to arrive at broad tasks for a project**

b) **Refine the broad tasks**

c) **Categorize tasks into logical task headers**

d) **Brainstorming to Arrive at Broad Tasks**

- **The first step in creating a WBS is to organize a meeting of all senior managers, system analysts, and prospective developers.**
- **The objective of the meeting is to brainstorm and come up with a set of broad tasks that need to be performed in a project.**
- **Note of all possible tasks is made and subsequently, the feasibility of each task is assessed and any conflict of common tasks is eliminated.**

a) **Brainstorming to Arrive at Broad Tasks (Cont.)**

- **During the session the project managers and the analysts come up with tasks on the basis of prior project experience focusing on:**
    - **Determining the scope of a project**
    - **Drafting the software specifications**
    - **Securing resources for the project**

a) **Brainstorming to Arrive at Broad Tasks (Cont.)**

- **Some additional tasks are also determined based on client requirements.**
    - **Preparing the initial project budget**

- **Estimating the approximate project timeline etc**

- **In addition, personnel responsible to complete each task are also determined.**
- **The subtasks are subsequently arranged in the order in which they are executed.**

b) **Refining Broad Tasks**

- **In the second step of creating a WBS, you refine the list of tasks that was compiled during brainstorming.**
- **Refining the tasks may include adding more tasks or combining the existing ones.**
- **You may also change the arrangement of tasks.**
- **A change in the arrangement of tasks can occur on the basis of two theories of WBS.**
    - **Deliverable-based theory**
    - **Project life-cycle-based theory**

b) **Refining Broad Tasks (Cont.)**

- **You use the deliverable-based theory when the deliverables of a project are more important than its phases.**
- **This normally happens when the deliverables are decided before the project begins.**
- **However, if the phases of a project are completely defined, you use the project -life-cycle-based theory.**
- **You can use the project life-cycle-based theory to arrange the tasks in a project.**

c) **Categorizing Tasks into Logical Headers**

- **Finally, after defining tasks and arranging them, you categorize each task into a logical task header.**
- **Example: preparing test plans and test cases and drawing up the test plans can be categorized as Testing.**
- **This activity provides another chance to ensure that you have not missed any task during brainstorming.**
- **In addition, you can also consult an expert such as a senior manager to review and validate the tasks identified.**

4. **WBS Implementation**

i. **Implementation Strategy**

a) **When you set up a project WBS, think about how you will be using it later in the project.**

b) **Consider how you will organize the WBS, schedule format, manager assignments, and charge numbers, in your early project planning.**

c) **It will be helpful if you can map the charge numbers, managers, and task groups to each other.**

d) **This will help you track costs and progress for each manager.**

4. **WBS Implementation (Cont.)**

i. **Implementation Strategy (Cont.)**

e) **If your project schedule will be on MS-Project, you may want to insert "text" columns into your schedule (Gantt View) for project charge numbers and manager names.**

f) **Some project management environments have definite conventions for grouping items in a WBS.**

g) **The best method is to have a WBS that works for your particular project environment.**

h) **The WBS should be designed with consideration for its eventual uses.**

4. **WBS Implementation (Cont.)**

i. **Implementation Strategy (Cont.)**

i) **Your WBS design should try to achieve certain goals:**

- **Be compatible with how the work will be done and how costs and schedules will be managed.**
- **Give visibility to important or risky work efforts.**
- **Allow mapping of requirements, plans, testing, and deliverables.**
- **Foster clear ownership by managers and task leaders.**
- **Provide data for performance measurement and historical databases.**
- **Make sense to the workers and accountants.**

4. **WBS Implementation (Cont.)**

ii. **Methodology**

a) **PM must map activities to chosen lifecycle as each lifecycle has different sets of activities**

b) **Integral process activities occur for all**

- **Planning, configuration, testing**

c) **Operations and maintenance phases are not normally in plan (considered post-project)**

4. **WBS Implementation (Cont.)**

ii. **Methodology (Cont.)**

c) **Some models are "straightened" for WBS**

- **Spiral and other iterative models**
- **Linear sequence several times**

e) **Deliverables of tasks vary by methodology**

5. **Review-WBS**

- **Types: Process, product, hybrid**
- **Formats: Outline or graphical org chart**
- **Shows: Hierarchical task relationships**
- **High-level version: Does not show dependencies or duration**
- **What hurts most is: What's missing**
- **Becomes input to many things especially: Schedule**
- **Summary**

1. **WBS Preview**

   i. **Salient features**

   ii. **What is it?**

   iii. **When is it produced?**

   iv. **What it contains?**

   v. **Use of WBS**

2. **WBS-A Mandatory Management Tool**

   i. **The WBS as a project management tool**

ii. **Characteristics of a High-Quality WBS**

iii. **Types of WBS**

iv. **Sample WBS**

- **Summary**

3. **WBS- Major Steps**

ii. **Key Steps**

a) **Project Decomposition**

b) **Functional Decomposition**

c) **Design Decomposition**

d) **Develop Project Tasks**

e) **Define Project Development Phases**

f) **Define Task Relationships**

g) **Defining Deliverables**

iii. **Creating a WBS**

a) **Brainstorming to arrive at board tasks for a project**

b) **Refine the board tasks**

c) **Categorize tasks into logical task headers**

- **Summary**

4. **WBS Implementation**

i. **Implementation**

ii. **Methodology**

5. **WBS Preview**

Lecture37

- **Software Project Management**

I. **Overview of Management**

II. **Software Developmental Fundamentals**

III. **Process**

IV. **Planning**

V. **Organization**

VI. **Estimation**

VII. **Work Breakdown Structure**

VIII. **Scheduling**

- **Scheduling**

I. **Understanding Schedule**

1. **Salient Features**
2. **Need for Project Scheduling**
3. **What Delays Software Projects?**
4. **How to Prevent Delays?**

II. **Scheduling Fundamentals**

1. **Basics**
2. **Creating a Schedule**

III. **Scheduling Tools**

1. **Gantt Chart**
2. **Network Scheduling Techniques**

IV. **Rules for Creating a Network Schedule**

V. **Comparison Analysis of Project Management Tools**

VI. **Scheduling Personnel**

VII. **Scheduling Resources**

VIII. **Monitoring and Updating the Schedule**

IX. **General Guidelines for Scheduling**

X. **Understanding Schedule**

1. **Salient Features**

    i. **The project manager's objective is to:**

        a. **Define all project tasks**

        b. **Build a network depicting their interdependencies**

        c. **Identify the tasks that are critical within the network**

        d. **Track their progress to ensure that delay is recognized**

        e. **Understanding Schedule**

2. **Salient Features (Cont.)**

    i. **As a project manager, you need to assign / understand durations to all activities and monitor their progress.**

    ii. **You also need to plan the order in which the activities will be performed as well as the start and end dates for each activity.**

    iii. **The plan includes not only the scheduling of development activities, but also the scheduling of project resources, particularly people.**

    iv. **Understanding Schedule**

3. **Salient Features (Cont.)**

    i. **To accomplish this, the manager must have a schedule that has been defined at a degree of resolution that enables the manager to monitor Progress and control the project.**

    ii. **The project schedule is one of the most important parts of the project development plan.**

I. **Understanding Schedule**

1. **Salient Features (Cont.)**

    vii. **Software project scheduling is an activity that distributes estimated effort across the planned project duration by allocating the effort to specific software engineering and other related tasks.**

    viii. **The project schedule reflects the start dates and end dates for each activity in the project.**

ix. **Understanding Schedule**

2. **Salient Features (Cont.)**

vii. **To create a schedule, you can use project management tools, e.g.:**

- **Gantt charts**
- **Network-scheduling techniques**

viii. **No matter how well the project schedule is prepared, that schedule is useless unless it is adhered to.**

- **Understanding Schedule**

3. **Salient Features (Cont.)**

vii. **It is the project manager's responsibility to withstand pressure and to assure that the project is developed in an orderly fashion, according to the schedule.**

viii. **Whenever circumstances change, the project schedule should then be updated to reflect the new situation.**

I. **Understanding Schedule**

2. **Need for Project Scheduling**

i. **Software projects have a tendency to get out of control because of the multiple activities that need to be monitored, tracked, and controlled.**

ii. **When a project goes out of control, the original deadlines, the budget, and the effort required overshoot the initial estimates.**

iii. **This not only impacts the product, but also the credibility of the development team.**

I. **Understanding Schedule**

3. **What Delays Software Projects?**

- **There are various reasons why a software project is delayed. Some of the reasons are listed below:**

i. **The initial estimate of the effort and resources required to complete the software project was incorrect.**

ii. **Changes in client requirements were not accounted for in the project-schedule.**

I. **Understanding Schedule**

3. **What Delays Software Projects? (Cont.)**

iii. **Known risks were not mitigated.**

iv. **Technical difficulties surfaced during software product development.**

v. **Human difficulties, such as interpersonal problems and team attitude, became unmanageable.**

vi. **The initial deadline set for the project was unrealistic.**

I. **Understanding Schedule**

5. **How to Prevent Delays?**

i. **While most of the above reasons result in project delays, unrealistic deadlines cause the maximum damage.**

ii. **Unrealistic deadlines are caused either by the client wanting a quick delivery or the management wanting a return business from a client.**

iii. **However, unrealistic deadlines are the norm for software projects.**

I. **Understanding Schedule**

5. **How to Prevent Delays? (Cont.)**

iv. **You need to resolve the conflict between an unrealistic deadline and the constraints of the software development team.**

v. **As a project manager you need to balance the limited resources and the final deliverable committed to the client.**

vi. **To do this, you can follow the points listed below:**

a) **Study similar past projects for the method of dealing with similar situations. Use the historical information to project time and effort estimates for the current software project.**

I. **Understanding Schedule**

5. **How to Prevent Delays? (Cont.)**

b) **Use an incremental process model to create a schedule to deliver the required functional module of the software product.**

c) **Present the schedule created on the basis of past projects to the client. Explain the projected delays and the reasons for the same.**

d) **Present the incremental model as a strategy for developing the software product.**

e) **However, all the strategies have their advantages and disadvantages.**

II. **Scheduling Fundamentals**

1. **Basics**

i. **Only a small percentage of all the activities in a software project have a direct impact on the on-time completion of the software project.**

ii. **Therefore, you need to identify the activities that are critical to the completion of the project.**

iii. **In addition, you ensure that all inputs are available to complete the critical activities.**

II. **Scheduling Fundamentals**

1. **Basics (Cont.)**

iv. **This is where a software project schedule comes into the picture.**

v. **The goal of a software project schedule is to determine the duration of the software project and the phases within the project.**

vi. **A software project schedule enables you to distribute the estimated effort to be spent in performing the critical activities.**

II. **Scheduling Fundamentals**

1. **Basics (Cont.)**

vii. **There are a few basic principles that determine how a software project schedule is created.**

viii. **Group similar activities together.**

ix. **Determine the dependencies of the different activities.**

x. **Allocate the estimated time and resources to each activity.**

xi. **Define the roles and responsibilities and the output and validation criteria.**

II. **Scheduling Fundamentals**

2. **Creating a Schedule**

- **Some of the guidelines for creating a software project schedule are discussed below:**

i. **Classification**

ii. **Interdependence**

iii. **Time and effort allocation**

iv. **Validation criteria**

v. **Defined responsibilities and outputs**

II. **Scheduling Fundamentals**

2. **Creating a Schedule (Cont.)**

i. **Classification**

ii. **While managing a software project, you need to group similar tasks and activities so that they are completed successfully.**

iii. **The primary tools used to group similar tasks are the WBS and decomposition technique.**

iv. **Using these tools, you can divide a software project into different phases.**

II. **Scheduling Fundamentals**

2. **Creating a Schedule (Cont.)**

i. **Classification (Cont.)**

ii. **A software project is composed of multiple phases and each phase is composed of multiple activities.**

iii. **Although each activity is treated separately, it is linked to other activities.**

II. **Scheduling Fundamentals**

3. **Creating a Schedule (Cont.)**

i. **Interdependence**

ii. **A software project is composed of multiple phases and each phase is composed of multiple activities.**

iii. **Although each activity is treated separately, it is linked to other activities.**

II. **Scheduling Fundamentals**

4. **Creating a Schedule (Cont.)**

i. **Interdependence (Cont.)**

ii. **As a project manager, you determine the interdependence and sequence of activities.**

iii. **Example: Some activities can be completed without any inputs from other activities, whereas other activities cannot start unless a preceding activity is completed.**

II. **Scheduling Fundamentals**

2. **Creating a Schedule (Cont.)**

iii. **Time and Effort Allocation**

iv. **Each activity in a software project needs a certain amount of time and effort for completion.**

v. **To manage the project, you assign start and end dates to each activity.**

vi. **You also need to allocate appropriate effort to each activity.**

vii. **Most software projects operate with time and effort constraints.**

viii. **Therefore, managing within the available resources is very important for a software project manager.**

II. **Scheduling Fundamentals**

2. **Creating a Schedule (Cont.)**

iv. **Validation Criteria**

- **You also determine the validation criteria for time and effort allocation in a software project.**
- **Determining the validation criteria allows you to ensure that the optimal level of resources is available for a particular activity.**
- **Ex: Suppose 5 people are assigned to an activity that requires an effort of 3.5 person days. This means resources allocated are more than the actual requirement.**

II. **Scheduling Fundamentals**

2. **Creating a Schedule (Cont.)**

v. **Defined Responsibilities and outputs**

- **As a software project manager, you assign roles and responsibilities to all people assigned to a software project.**

- **This defines the hierarchy in the development team.**

- **You also define the outputs from each activity.**
  - **This helps in identifying the results expected at the end of every activity.**

II. **Scheduling Fundamentals**

2. **Creating a Schedule (Cont.)**

v. **Defined Responsibilities and outputs (Cont.)**

- **Next, you link the roles to the outputs.**
- **When each role is linked to the expected results, you can track each person's effort and the progress of each activity towards closure.**

- **Summary**

I. **Understanding Schedule**

- **Salient Features**
- **Need for Project Scheduling**
- **What Delays Software Projects?**
- **How to Prevent Delays?**

II. **Scheduling Fundamentals**

- **Basics**
- **Creating a Schedule**

Lectu38

- **Software Project Management**

I. **Overview of Management**

II. **Software Developmental Fundamentals**

III. **Process**

IV. **Planning**

V. **Organization**

VI. **Estimation**

VII. **Work Breakdown Structure**

VIII. **Scheduling**

- **Reference to Previous Session**

I. **Understanding Schedule**

1. **Salient Features**
2. **Need for Project Scheduling**
3. **What Delays Software Projects?**
4. **How to Prevent Delays?**

II. **Scheduling Fundamentals**

1. **Basics**
2. **Creating a Schedule**

- **Scheduling (Cont.)**

III. **Scheduling Tools**

1. **Gantt Chart**
2. **Network Scheduling Techniques**

IV. **Comparison Analysis of Project Management Tools**

V. **Scheduling Personnel**

VI. **Scheduling Resources**

VII. **Monitoring and Updating the Schedule**

VIII. **General Guidelines for Scheduling**

IX. **Scheduling Tools**

1. **Gantt Chart**

- **There are various tools that help you create a schedule.**
- **One of the simplest project management tools used to represent the timeline of activities is the Gantt chart.**

- **A Gantt chart has horizontal bars plotted on a chart to represent a schedule.**
- **In a Gantt chart, you plot time on the horizontal axis and activities on the vertical axis.**
- **High level Gantt chart -*Project Development Schedule***

1. **Gantt Chart**

- **You represent an activity by a horizontal bar on the Gantt chart.**
- **The position of a horizontal bar shows the start and end time of an activity and the length of the bar shows its duration.**
- **You can have one look at the Gantt chart and make out the progress of the project.**

1. **Gantt Chart**

- **Activity D starts on day 1 of the project. The planned duration of this activity is 20 days.**
- **The planned duration is shown in a lighter shade on the Gantt chart.**
- **This is done to differentiate planned duration from the current status of the activity.**

1. **Gantt Chart**

- **In the case of activity D, the current duration of the activity is 19 days.**
- **Therefore, activity D is still one day short of completion. This can be discerned from the length of the gray and black bars.**
- **Activity C, planned for completion on day 40, is much behind schedule.**
- **This can be observed from the smaller length of the black bar in the Gantt chart.**
- **Table 1: Project Activities and Time Allocation Details**

1. **Gantt Chart**

- **The Gantt chart displays the week-wise status of a software project.**
- **A vertical dotted line is drawn through the chart to represent the current date.**
- **This line indicates the status of the project on a specific day.**
- **The left side of the line indicates the tasks that are completed.**
- **On-going activities run across the line.**

1. **Gantt Chart**

- **The future activities lie completely to the right of the line.**
- **After drawing the Gantt chart you can extend the bars every week to a length proportional to the work completed during the week.**
- **Therefore, the bars describe the status of the-project at any point of time.**
- **This process of creating a Gantt chart is manual.**
- **To create a project schedule automatically various automated tools are available like MS project**

2. **Network Scheduling Techniques**

- **Network scheduling techniques use network schedules to trace the completion of predetermined activities.**
- **There are two basic network-scheduling techniques:**
    - **PERT**
    - **CPM**
- **Each technique depicts a project as a sequence of activities.**
- **This helps you perform an analysis of individual activities or the complete project.**

2. **Network Scheduling Techniques**

- **The network-scheduling techniques also enable you to analyze the dependencies that exist between the activities.**
- **Using PERT and CPM, individually or in combination, helps you complete a project on time.**
- **By using these techniques, you can determine the latest time by when an activity should start to be completed on time.**

2. **Network Scheduling Techniques**

- **Despite the different approaches followed by PERT and CPM, both techniques have some common components like:**
    i. **Activities**
    ii. **Nodes**
    iii. **Network**
    iv. **Critical path**

i. **Activities**

- **Activities are the basic building blocks of network schedules.**
- **An activity is defined as a task that consumes time, effort, money, or any other resource.**
- **It is necessary to specify all the activities of a project by breaking down a project into several steps.**

i. **Activities**

- **You need to define the steps in such a way that they are distinct, homogeneous tasks for which you can estimate resource requirements.**
- **Each activity is represented on a network schedule by using an arrow with its head indicating the direction in which the project will progress.**
- **Each activity is identified by a description or an alphabet.**
- **In addition, the estimated duration of each activity is placed below the activity.**

▪ **An Activity**

i. **Activities**

- **After identifying all the activities in a project, it is necessary to schedule them.**

- **This enables you to arrange the activities in the order of completion.**
- **You sequence the activities based on their types.**

i. **Activities**

- **Three kinds of activities that are possible in a project are predecessor, successor, and concurrent activities.**
    - **Predecessor activities need to end before the next can begin**
    - **After a predecessor activity is completed, the successor activity becomes the predecessor for another activity.**
    - **However, unlike the predecessor and successor activities, the concurrent activities can be completed simultaneously with other activities.**

i. **Activities**

- **Some activities in a project, irrespective of whether they are predecessor, successor, or concurrent, may depict a float period.**
- **Float is the amount of time by which an activity may be delayed without affecting the completion date of the entire project.**
- **However, the complex dependencies that exist between activities result in sequencing constraints for projects.**
- **To a large extent, these constraints limit the flexibility that you may otherwise have in project planning.**

ii. **Nodes**

- **A node on a network schedule is that point in time at which an activity either begins or ends.**
- **The point where an activity begins is called a tail node and the point where it end is called its head node.**
- **On a network schedule, a circle represents a node.**
- **A number identifies each node in a network schedule.**

iii. **Network**

- **A network is the graphic representation of the activities in a project.**
- **It depicts all the activities and nodes in the project.**
- **On- a network, the arrows terminating at a node need to be completed before the following activity can begin.**
- **Sample Network Schedule**

iv. **Critical Path**

- **The critical path is the longest path through a network.**
- **It consists of those activities that cannot be completed concurrent1y. In other words, the critical path represents the maximum duration for a project.**
- **You can determine the maximum duration by adding the duration of each activity on the critical path.**
- **Typically, a double line in a network schedule represents the critical path for that project.**
- **Network Schedule with Critical Path Identified**

iv. **Critical Path**

- **All the activities on the critical path are critical for a project.**
- **If an activity on the critical path is delayed, the entire project is delayed by the same amount of time.**
- **You need to monitor the activities on the critical path because the project depends on the successful completion of these activities.**
- **If required, additional resources can be applied to these activities to shorten the project duration.**
- **However, some of these activities may also depict a float period.**
- **Rules for Creating a Network Schedule**
- **There are a few basic rules that are followed while constructing a network schedule.**

    1. **Each activity has a preceding node and a succeeding node.**
    2. **Each node has a distinct number. As a convention, the number that is assigned to the head of the arrow is greater than the number that is assigned to the tail.**
    3. **The network schedule has no loops. For example, in Figure 5, activity 1 is the predecessor of both activity 2 and activity 3. This places activities l, 2, and 3 in a loop.**
    4. **Fig 5: A loop is not permitted**

- **Rules for Creating a Network Schedule**

    1. **Each activity has a unique preceding and succeeding event associated with it.**

- **For example, in Figure 6, activities A and B have common preceding and succeeding events associated with them. This is not allowed in a network schedule.**
- **Figure 6**
- **Rules for Creating a Network Schedule**
- **At times, you need to introduce a dummy activity in a network schedule.**
- **This is an imaginary activity that enables the network schedule to display parallel activities.**
- **For example, in Figure 7, consider A and B as two parallel activities that can be executed simultaneously.**
- **Rules for Creating a Network Schedule**
- **Both A and B must end so that activity C can begin.**

- **A dummy activity is introduced as a dashed arrow to mark the start of activity C.**
- **The dummy activity is introduced to show the dependency between activities in the network schedule and does not have a description or duration.**
- **Figure 7**
- **PERT**
- **Program Evaluation and Review Technique: based on idea that estimates are uncertain**
    1. **Therefore uses duration ranges**
    2. **And the probability of falling to a given range**
- **Uses an "expected value" (or weighted average) to determine durations.**
- **PERT was developed in 1957 to cater to the needs of the Polaris Fleet Ballistic Missile project of the US government.**
- **PERT uses a probabilistic approach to time estimates.**
- **You normally apply it to projects that are characterized by uncertainty.**
- **For example, in a complex software project where you require research to identify activities, PERT allows you to account for the uncertainties that are common to most software projects.**
- **PERT uses the network schedule to represent a project schedule while taking the uncertainties into account.**
- **Time Estimates in PERT**
- **PERT is a probabilistic technique that uses three time estimates.**
- **It assumes that activity times are represented by a probability distribution.**
- **Time Estimates in PERT**
- **To finish an activity, it bases the probability distribution of activity time on three time estimates:**
    1. **Optimistic time**
        - **Would likely occur 1 time in 20**
    2. **Pessimistic time**
        - **Would be exceeded only one time in 20**

3. **Most likely time**

   - **Modal value of the distribution**

- **Time Estimates in PERT**
- **The optimistic time is the shortest time period within which an activity can end if everything goes well.**
- **The pessimistic time is the time that an activity takes to complete if everything that can go wrong goes wrong.**
- **This is the longest time that an activity can take to complete.**
- **Time Estimates in PERT**
- **The most likely time is the estimate of the normal time that an activity takes to complete.**
- **From these three estimates, you derive the expected time to complete an activity.**
- **The expected time is also referred to as the average time for the activity.**
- **PERT Formula**
- **To calculate the expected time for each activity, you use the following equation:**
- **PERT Formula**
- **Confidence Interval can be determined**
- **Based on a standard deviation of the expected time**

1. **Using a bell curve (normal distribution)**

- **For the whole critical path use**
- **PERT Example**
- **Confidence interval for P2 is 4 times wider than P1 for a given probability**
- **Ex: 68% probability of 9.7 to 11.7 days (P1) vs. 9.5-13.5 days (P2)**
- **Time Estimates in PERT**
- **According to this equation, you calculate the expected time for an activity as a weighted mean of the optimistic, most likely, and pessimistic times.**
- **The weights attached to these times are 1, 4, and 1, respectively.**
- **Consider the example of the activities in a software project as given in Table 1.**

- **Following table provides the break-up of the estimated times into the optimistic, pessimistic, and most likely times for each activity.**
- **Table 2: Optimistic, Most Likely Time, and Pessimistic Estimates for Activities**
- **Fig 8: A typical PERT chart**
- **Fig 9:Pert Network Schedule for Activities**
- **PERT packages and enhancements**
- **Some enhanced versions of the PERT chart support additional planning activities, such as personnel assignment, resource allocation and cost analysis.**
- **An interesting adaptation of PERT, called flow graph representation uses precedence networks to perform project life cycle cost analysis.**
- **PERT packages and enhancements**
- **The flow graph technique analyzes project costs based on relationships between quantities, unit cost, time variables, staffing costs and learning etc., all of which are represented on the PERT- like chart.**
- **PERT software packages take much of the tedium out of the preparation of PERT charts, and also come with additional features such as various planning analyzers for activity assignment, what if scenarios and resource allocation.**
- **PERT packages and enhancements**
- **Computer utilities have been developed to perform flow graph representation analysis which produces scheduled costs for major project activities.**
- **These utilities have proven invaluable for project managers and release managers from laborious desk work, providing them with more time to actively manage the project.**
- **Summary**

III. **Scheduling Tools**

1. **Gantt Chart**
2. **Network Scheduling Techniques**
    - **Rules for Creating a Network Schedule**
    - **PERT**

- **Software Project Management**

I. **Overview of Management**

II. **Software Developmental Fundamentals**

III. **Process**

IV. **Planning**

V. **Organization**

VI. **Estimation**

VII. **Work Breakdown Structure**

VIII. **Scheduling**

- **Reference to Previous Session**

III. **Scheduling Tools**

  - **Gantt Chart**
  - **Network Scheduling Techniques**
    - **Rules for Creating a Network Schedule**
    - **PERT**

- **CPM**
- **CPM was developed in the 1950s by DuPont to solve scheduling problems in an industrial setting.**
- **This method was used successful1y by a number of projects that use a stable technology and are relatively risk-free.**
- **CPM is also based on the concept of a network schedule.**
- **CPM uses the time estimates that are calculated using the PERT technique.**
- **CPM**
- **CPM treats the time estimates from, PERT as a known value.**
- **This means that CPM is a deterministic technique whereas PERT is a probabilistic technique that estimates the duration of activities.**

- **CPM and PERT were developed as two separate techniques.**
- **However, over time, both the techniques evolved into one scheduling and tracking tool for project managers.**
- **CPM**
- **CPM allows you to derive the critical path for a project schedule.**
- **The critical path is the longest path in a software project.**
- **The critical path is composed of critical activities in a project.**
- **These activities are well defined and independent.**
- **CPM**
- **This means no activity in the critical path is dependent on another activity for successful completion.**
- **To derive the critical path for a network schedule, you need to calculate the earliest finish (EF) time and the latest finish (LF) time for each activity.**
- **CPM**
- **In order to examine this phenomenon we must first understand that in all non-trivial networks there are many ways of moving from the source node to the sink node.**
- **For example, in following figure, a possible path runs from node S to 1 to 3 to 6 to 9 to 10 to E.**
- **Another possible path runs from node S to 1 to 2 to 5 to 8 to 10 to E.**
- **Possible Paths from Source to Sink Nodes**
- **CPM**
- **Each path can be characterized by a number that represents the total duration of the path, calculated by summing the durations for all activities along the path.**
- **Table 1 contains a list of all possible paths from node S to node E in the PERT chart in following figure, together with the length of each path. Path 2 is the longest, 52 weeks.**
- **CPM**
- **The longest path is referred to as *the critical path* and it determines the duration of the project.**
- **By shortening the duration of an activity along the critical path we can usually shorten the duration of the whole project.**

- **There are a few extreme cases where this will not occur, notably when there are two critical paths.**
- **However, one thing is certain; shortening an activity that is not on the certain path will not shorten the length of the whole project.**
- **CPM**

a) **Calculation of the Earliest Finish Time**

- **The earliest finish time for an activity is the minimum time required to complete it.**
- **The EF time of the first node is always zero.**
- **To calculate the EF time for a network schedule, you need to move along the schedule from left to right.**
- **Starting from the first node, you add the EF time of each node to the duration required by the immediately succeeding activity.**
- **CPM**

b) **Calculation of the Earliest Finish Time**

- **In case a node has more than one preceding activities, you use the largest EF time of the preceding activities.**
- **This rule also applies in case one of the preceding activities is a dummy activity.**
- **For example, apply the rules for calculating the EF time to the activities listed in Table 1.**
- **The EF time of node 1 is 0.**
- **Project Activities and Time Allocation Details**

- **CPM**

a) **Calculation of the Earliest Finish Time**

- **Therefore, the EF time for node 2 is 10 person days.**
- **The EF time for node 4 is calculated by adding the duration of the software construction activity, which is 80 person days, to the EF time of the preceding node 2, which is 10 person days.**
- **Therefore, the EF time of node 4 is 90 person days.**

- **CPM**

a) **Calculation of the Earliest Finish Time**

- **This means that the earliest possible time within which the software construction activity can be finished from the start of the project is 90 person days.**
- **The EF time for the rest of the activities in the network schedule is calculated in a similar manner. Following figure shows the network schedule with the EF time calculated for all activities**
- **CPM**

b) **Calculation of the Earliest Finish Time**

- **This means that the earliest possible time within which the software construction activity can be finished from the start of the project is 90 person days.**
- **The EF time for the rest of the activities in the network schedule is calculated in a similar manner. Following figure shows the network schedule with the EF time calculated for all activities**
- **Fig 10: Network Schedule with EF time calculated**

- **CPM**

b) **Calculation of the Latest Finish Time**

- **The latest finish time for an activity is the maximum time required to complete it without delaying the project.**
- **To calculate the LF time for a network schedule, you move along the schedule from the right to the left.**
- **Starting from the last node, you subtract the duration of the following activity from the LF time.**
- **If a node has more than one following activity, subtract the smallest activity duration from the LF time to obtain the LF time of the next node.**

- **CPM**

b) **Calculation of the Latest Finish Time**

- **This rule also applies if the following activity is a dummy activity.**
- **For example, apply the rules for calculating the LF time to the activities listed in Table 1.**

- **The LF time for node 8 is calculated by subtracting the duration of the user acceptance testing activity, which is 20 person days, from the LF of node 10, which is 155 person days.**
- **CPM**

c) **Calculation of the Latest Finish Time**

- **Therefore, the LF of node 8 is 135 person days. Similarly, the LF for node 5 is calculated subtracting the duration of the user training activity from the LF time of node 8.**
- **The LF time for node 5 is 130 person days.**
- **The LF time for node 6, which has a dependent relationship with node 8, is the same as the LF time of node 8.**
- **In a similar manner you can calculate the LF time for the rest of the activities in the network schedule.**
  - **Fig 11: Network Schedules with LF Times calculated**

- **CPM**

c) **Deriving Critical Path**

- **To determine the critical path or a network; you identify the nodes where the EF time and the LF time are equal.**
- **Then, compare the EF time and the LF time calculated for the network schedule.**
- **For example, combining figures 8.10 and 8.11 determines the critical path for the network schedule with the activities defined in Table 1. This is displayed in Figure 8: 12.**
- **Fig 12: Network Schedule with Critical Path Determined**

- **Comparison Analysis of Project Management Tools**
- **The three project management tools discussed are used for a variety of software-related and non software-related projects.**
- **However, each tool is suited to a particular application.**
- **In addition, each tool has its advantages and disadvantages that permit or limit you from using the tool in specific conditions.**
- **Comparison Analysis of Project Management Tools**

- **You use Gantt charts in simple software projects where the software development life cycle is linear.**
- **Example: The structure of a software product for the banking industry is well defined because of the numerous banking applications that already exist.**
- **However, Gantt charts have a few disadvantages. The three primary disadvantages of a Gantt chart are related to:**

    1. **Interdependence of activities**
    2. **Tracking the project progress**
    3. **Representing uncertainties**

- **Disadvantages of Gantt Chart**

1. **Interdependence of activities**

    - **A Gantt chart cannot represent the interdependence of activities in a software project properly.**
    - **The bar graph format is unsuited if you want to display a large number of activities that are concurrent.**
    - **Example: In a software project, there are various activities that happen either simultaneously or overlap. Using a Gantt chart, you cannot distinguish between the activities that occur simultaneously from those that share some dependency.**

- **Disadvantages of Gantt Chart**

2. **Tracking the Project Progress**

    - **You also cannot use a Gantt chart as a control device to track a software project.**
    - **The bar graph format of the Gantt chart does not provide you with any method of tracking the progress of a software project.**
    - **You can only check the different activities within the software project, their start dates, and their end dates.**

- **Disadvantages of Gantt Chart**

3. **Representing uncertainties**

    - **A Gantt chart cannot reflect the uncertainties within a complex software project.**

- **You can only use it for simple software projects where all the" activities are known and for which the start and end dates can be easily determined.**
- **However, uncertainties are the norm in modern software projects, and project schedules often require rescheduling to adjust for uncertainties.**

- **Disadvantages of Gantt Chart**

3. **Representing uncertainties (Cont.)**
    - **If a project is large and complex, such as creating automated error-checking software for a communications satellite, the different activities and sub-activities cannot be determined at the time the project commences.**
    - **There is an uncertainty about the activities in the later stages of the project until that stage is reached. In addition, Gantt charts cannot reflect the tolerance levels during various activities.**

- **Advantages of PERT/CPM**
- **PERT and CPM are useful techniques that allow you to manage the complexity of interdependencies within a software project.**
- **In addition, both techniques allow you to schedule and track a software project more efficiently than Gantt charts.**
- **Both PERT and CPM use network schedules to represent a project schedule.**
- **Although PERT and CPM were developed separately, both techniques were combined overtime and began to be referred to as PERT/CPM.**
- **Advantages of PERT/CPM**
- **The advantages of PERT/CPM relate to:**
    1. **Precedence relationship**
    2. **Interdependencies**
    3. **Large projects**
    4. **Simple structure**
    5. **Preplanning**
    6. **Project tracking**

1. **Precedence relationship**

- **PERT/CPMI allows you to organize the activities in a project according to a sequence.**
- **In a network schedule, each activity must have a preceding activity and a succeeding activity.**
- **This rule is applicable to all activities except the first and last activities.**
- **Therefore, PERT/CPM allows you an advantage over Gantt charts that arranges the activities in terms of their duration.**

2. **Interdependencies**

- **Unlike the Gantt chart, which is unable to depict interdependencies between various activities, PERT/CPM allows you to show the interdependencies.**
- **For example, you can use the network schedule to specify that an activity will begin only after specific preceding activities are complete.**
- **Similarly, you can also show that two activities need to be executed simultaneously for the succeeding activity to start.**

3. **Large Projects**

- **PERT/CPM is suited for one-time, large, and complex projects that have a lot of uncertainties.**
- **In a complex software project, it is very difficult to identify each activity in the software project in advance.**
- **Most activities are identified and defined in terms of their start and end dates only during the -relevant stage of the project.**
- **PERT/CPM allow you to create a project schedule while taking project uncertainties into consideration.**

4. **Simple Structure**

- **PERT/CPM is based on simple mathematical concepts.**
- **The technique uses network schedules to describe a project schedule.**
- **The graphical nature of the network schedule is easy to understand.**
- **Even a layperson can understand the scope of the entire software project by looking at a summary network schedule.**
- **The PERT/CPM schedule can also be used as a medium of communication between groups within and across projects to convey the project status and progress.**

5. **Preplanning**

- **PERT/CPM forces you to plan the software project in advance and achieve better coordination of work.**
- **As the network schedule demands that each activity must have a preceding and succeeding activity, you are forced to identify as many activities as possible and sequence them.**
- **The identification of as many activities as possible reduces the chance of important activities being omitted from consideration.**

5. **Preplanning (Cont.)**

- **The identification and sequencing exercise helps identify potential problems and bottlenecks in project execution.**
- **This allows you to resolve the problems on paper before they materialize.**

6. **Project Tracking**

- **The network schedule focuses your attention on the critical path and critical activities.**
- **This enables you to complete the project within the shortest time and least cost.**
- **You can track the progress of the project because the start and end dates are known for each critical activity.**
- **PERT/CPM also allows you to monitor costs and check if the project is within the allocated budget.**
- **Disadvantages of PERT/CPM**
- **Although PERT/CPM provides many benefits in creating a project schedule, the technique is not foolproof.**
- **The disadvantages of PERT/CPM relate to:**

    1. **Activity definition**
    2. **Precedence relationships**
    3. **Time estimates**

1. **Activity Definition**

- **PERT/CPM requires that each-activity in a software project must-be clearly defined.**
- **In addition, no activity can overlap with another activity in terms of the task being achieved.**
- **However, in a large and complex software project, it is not always possible for you to identify all the activities and sub-activities.**

- **If you attempt to create a network schedule with an incomplete list of activities, important activities may be left out.**

2. **Precedence Relationships**

- **As a software project manager, you may not always be able to identify all the activities and sub-activities in a software project.**
- **When a software project commences, you have clearly defined activities only for a few phases of the project.**
- **The rest of the activities are defined when the project reaches to the relevant stage.**
- **Lack of clarity on all activities in the project prevents you from defining the precedence relationships between all activities.**
- **This hampers the creation of a network schedule.**

3. **Time Estimates**

- **In PERT, you estimate the time required to complete an activity.**
- **Therefore, the PERT network schedule represents only a probabilistic estimation of time.**
- **A project schedule created on the basis of probabilistic time estimates can give you only a rough idea of the project timelines.**
- **CPM vs. PERT**
- **Scheduling Personnel**
- **Essentially, the development team is a resource, just as development equipment is a resource.**
- **However, scheduling people is not the same as scheduling equipment.**
- **The project manager's most important project resource is the development team, and therefore special attention must be given to the scheduling of the activities of team members.**
- **Scheduling Personnel**
- **As the number of project activities varies, so the size of the development team varies throughout the project development life cycle.**
- **The team's organizational structure becomes more important as the size of the team grows.**

i. **The development team size**

- **The size of the development team is influenced not only by the number of activities, but also by the intensity of the activities.**

- **Some activities are intense at the beginning of the project and decline toward the end, and vice versa.**
    - **Example: Planning requires more human resources at the beginning of the project and fewer at the end, while configuration control requires fewer at the beginning and more at the end.**

i. **The development team size (Cont.)**

- **The team size is determined by the total number of people assigned to the project.**
- **However, in scheduling the team size, the allocation of two people, each for half of a full-time assignment, does not necessarily equal the assignment of one person full-time.**
- **It is difficult to deal with the allocation of half a person to an assignment.**
- **In such cases, such as quality assurance or configuration control, the expense to the project of assigning these activities can be reduced by sharing these functions with other projects.**

i. **The development team size (Cont.)**

- **This is especially true for small projects.**
- **Testing is another example of personnel resource sharing between projects.**
- **Many organizations have an independent test team that is not directly part of the project team.**
- **The independent test team becomes involved in the project mainly toward the end of the development cycle (although some test activities do start much earlier).**
- **Such teams can then move from one project to another provided the test activities have been scheduled appropriately.**

ii. **The infamous man month**

- **Another common source of error, when scheduling people, is the difference in the way the term *man month* (or, as it is now called, *work month)* is used.**
- **If a project manager calculates that a specific scheduled activity requires six work months to complete, does that mean that if a suitable engineer is assigned for six months then the activity can be completed in six months?**

ii. **The infamous man month (Cont.)**

- **In many cases the activity cannot be completed in six months, because a person seldom provides six months of work during a six month calendar period.**

- **People take vacations, they celebrate holidays, and they are occasionally sick. Generally, people provide between eight and ten months work during a twelve month period.**
- **This must be taken into consideration when preparing a schedule.**

ii. **The infamous man month (Cont.)**

- **The schedule should take into account absenteeism, overhead (discussions, meetings and just plain talking) and it should explain which activities cannot be shortened and which can.**
- **Also, there is a maximum number of people that can be assigned to a project.**
- **Three hundred and sixty-five engineers cannot complete a one year project in one day!**

ii. **The infamous man month (Cont.)**

- **Therefore the significance of the duration of the project as well as the duration of each major activity must be clearly understood and presented as part of the schedule.**
- **Scheduling Resources**
- **Without the necessary tools, the development team cannot be expected to do a good job.**
- **If the project's target computer is not available when the integration phase is scheduled to begin, then the integration phase may very well not begin.**
- **We therefore see that a form of the *critical path* principle is also applicable to the availability of development resources.**
- **Scheduling Resources**
- **This means that if the availability of a critical scheduled resource is delayed it will delay the completion of the project.**
- **There are ways of dealing with availability problems for critical resources, such as *risk analysis.***
- **The objective is to provide contingency plans in the event that critical resources are not available according to schedule.**

i. **Scheduling workspace**

- **At the outset of the project, workspace, particularly office space, is usually one of the first major resources required.**
- **The need for adequate workspace is often overlooked when scheduling development facilities.**

- **The project area should be pre assigned and well defined.**
- **As the project progresses, the required work area will increase.**
- **A well-defined and separate work area is an important factor in forming a development team.**

ii. **Scheduling equipment**

- **With, the right tools, the job can always be done better.**
- **But as we have seen, having the right tools is not enough: the tools must also be available at the right time.**
- **The objective of the scheduling of equipment is to assure that adequate development tools are available in sufficient quantity, when needed.**

ii. **Scheduling equipment**

- **Assuring the availability of adequate development equipment is an important part of good planning.**
- **Bad scheduling can lead to situations where members of the development team are left idle or partly idle waiting for the delivery of equipment.**
- **Even if team members can be reassigned temporarily, their efficiency and effectiveness as developers will be significantly reduced.**

iii. **Vendors and subcontractors**

- **Not all scheduled activities are directly dependent on the project manager and the development earn.**
- **Frequently, outside parties are also involved in project development.**
- **As we have seen the timely delivery of equipment is crucial to the development schedule, and this often requires procurements from outside parties.**

iii. **Vendors and subcontractors (Cont.)**

- **It is not uncommon, especially in large or complex projects, to subcontract certain parts of the project to companies who have specific expertise in relevant areas.**
- **This means that direct: control of development may be delegated to the subcontractor.**

iii. **Vendors and subcontractors (Cont.)**

- **It is difficult to schedule resources and activities over which the project manager does not' 1ave full control. In such situations the project manager has two alternatives:**
    - **Leave the scheduling to the subcontractor or vendor**
    - **Retain control over the subcontractor or vendor**

iii. **Vendors and subcontractors (Cont.)**

- **In the first case, when scheduling is left to the outside party.**
- **The project manager is at the mercy of a party over which he has no control.**
- **If the other party slips the delivery schedule, it may cause a schedule slippage for the whole project. This is best handled by:**
    - **Motivating the outside party to deliver on time (e.g. penalizing the party for late delivery)**
    - **Identifying late delivery as a project risk, and preparing contingency plans to handle the situation should it occur**

iii. **Vendors and subcontractors (Cont.)**

- **In the second case, when the project manager retains control over the outside party, many of the benefits of subcontracting are lost.**
- **For large projects, a position must be created for a supervisor of subcontractors and vendors.**
- **It is the responsibility of the supervisor to be constantly aware of the work being performed by outside parties, through:**
    - **Visits to subcontractor and vendor sites**
    - **Reviews and milestone evaluations**
    - **Periodic reports from subcontractors and vendors**
- **Monitoring and Updating the Schedule**
- **Periodic reports**
- **Other schedule monitoring activities**
- **Updating the schedule**
- **General Guidelines for Scheduling**

1. **Promote team involvement**

2. **Iterate from high level to detailed schedule**
3. **Be aware of needs of customer, management, users and marketing**
4. **Schedule not only activities, but resources and personnel too**
5. **Resist pressure to commit to an unreasonable schedule**
6. **Use computerized scheduling tools**
7. **Schedule contingency plans for risks**
8. **Update the schedule periodically or after major project events**

- **Summary**

III. **Scheduling Tools**

2. **Network Scheduling Techniques**

- **CPM**

IV. **Comparison Analysis of Project Management Tools**

V. **Scheduling Personnel**

VI. **Scheduling Resources**

VII. **Monitoring and Updating the Schedule**

VIII. **General Guidelines for Scheduling**

Lecture40

- **Software Project Management**

I. **Overview of Management**

II. **Software Developmental Fundamentals**

III. **Process**

IV. **Planning**

V. **Organization**

VI. **Estimation**

VII. **Work Breakdown Structure**

VIII. **Scheduling**

IX. **Risk And Change Management**

- **Risk & Change Management**

1. **Fundamentals**
    - **What is it?**
    - **Why is it necessary?**
    - **Who does it?**
    - **Why is it important?**
    - **What are the steps?**
    - **What is the work product?**
    - **How do I ensure that I've done it right?**
2. **Risk & Change Management Concepts**
3. **Types of Risk**

- **Risk & Change Management**

2. **Risk Management Process**
3. **Risk Components and Drivers**
4. **Developing a Risk Table**
5. **Reactive VS. Proactive Risk Strategies**
6. **Risk Mitigation, Monitoring, and Management**
7. **Safety Risks and Hazards**
8. **The RMMM Plan**

- **What is it?**
- **Risk Management is the process by which risks to the project (e.g. to the scope, deliverables, timescales or resources) are formally identified, quantified and managed during the project.**
- **A project risk may be identified at any stage of the project by completing a Risk Form and recording the relevant risk details within the Risk Register.**

- **What is it? (Cont.)**
- **Change Management is the process by which changes to the Project's scope, deliverables, timescales or resources are formally defined, evaluated and approved prior to implementation.**
- **A core aspect of the Project Manager's role is to manage change within the project successfully.**
- **This is achieved by understanding the business and system drivers requiring the change, documenting the benefits and costs of adopting the change and formulating a structured plan for implementing the change.**
- **What is it? (Cont.)**
- **To formally request a change, it is often necessary to complete a Change Form.**
- **The change request details may then be recorded within a Change Register.**
- **Why is it necessary?**
- **Risk analysis and management are a series of steps that help a software team to understand and manage uncertainty.**
- **Many problems can plague a software project.**
- **A risk is a potential problem - it might happen, it might not.**
- **But regardless of the outcome, it's a really good idea to identify it, assess its probability of occurrence, estimate its impact, and establish a contingency plan should the problem actually occur.**
- **Who does it?**
- **Everyone involved in the software process participate in risk analysis and management:**
    - **Managers**
    - **Software engineers**
    - **Customers**
- **Why is it important?**
- **Software is a difficult undertaking.**
- **Lots of things can go wrong, and frankly, many often do.**

- **It's for this reason that being prepared, understanding the risks and taking preventive measures to avoid or manage them is a key element of good software project management.**
- **What are the steps?**
- **Recognizing what can go wrong is the first step, called - risk identification.**
- **Next, each risk is analyzed to determine the likelihood that it will occur and the damage that it will do if it does occur.**
- **Once this information is established, risks are ranked by probability and impact.**
- **Finally, a plan is developed to manage those risks with high probability and high impact.**
- **What is the work product?**
- **A risk mitigation monitoring and management (RMMM) plan or a set of risk information sheets is produced.**
- **How do I ensure that I've done it right?**
- **The risks that are analyzed and managed should be derived from thorough study of the people, the product, the process and the project.**
- **The RMMM should be revisited as the project proceeds to ensure that risks are kept up to date.**
- **Contingency plans for risk management should be realistic.**

2. **Risk & Change Management Concepts**

- **Foresight is an excellent management quality that can often be cultivated with experience.**
- **Indeed, in many cases, problems can be anticipated.**
- **In such cases, the manager can plan for the possibility that a problem will occur by estimating its probability, evaluating its impact, and preparing solutions in advance.**

2. **Risk & Change Management Concepts (Cont.)**

- **Performing risk analysis means being prepared.**
- **It is a form of insurance, the basic idea being that if a problem occurs, a solution is readily available.**
- **Like all insurance, risk analysis usually comes with a price.**
- **The cost of preparing for the occurrence of a problem is primarily the cost of having the alternative solution at hand while the problem may or may not occur.**

2. **Risk & Change Management Concepts (Cont.)**

- **In some cases, the cost may be minimal, the time needed to analyze and document the solution, and the time to track the problem.**
- **In other cases the cost may be substantial, for example, the price of an alternative piece of development equipment.**
- **In any case, a problem that has been analyzed and resolved ahead of time is far simpler to resolve than a problem that occurs unexpectedly.**

2. **Risk & Change Management Concepts (Cont.)**

- **Any project can encounter uncertainties in the form of increased costs, schedule delays, and diminished qualities.**
- **Unless tackled, these uncertainties can lead to major project disasters.**
- **The uncertainties encountered during project execution are the potential project risks.**

2. **Risk & Change Management Concepts (Cont.)**

- **Every software project has to grapple with the new risks threatening information security along with the conventional risks, such as hardware failure, time and cost escalation, defects, or resource crunch.**
- **Risk can be defined as the possibility of loss.**
- **Risk arises due to the inability to achieve objectives within defined cost, schedule, and technical constraints.**

2. **Risk & Change Management Concepts (Cont.)**

- **Risk has two components.**
- **The possibility of not achieving a particular outcome is one, and the result of failing to achieve the outcome is the other.**
- **The former is the probability of loss, and the latter is the loss.**
- **Software project management deals with managing both these components of risk.**

2. **Risk & Change Management Concepts (Cont.)**

- **"Risk management focuses the project manager's attention on those portions of the project most likely to cause trouble and compromise participants' win conditions."**

- **In other words, risk management is a set of actions that helps the project manager plan to deal with uncertain occurrences.**
- **It is through risk management that project managers assess risks and manage to reduce risks to an acceptable level.**

3. **Types of Risks**

i. **Development Process Risks**

a) **Developer errors**

b) **Project risks**

c) **Business risks**

d) **Unsatisfied employees**

e) **Poor management objectives**

ii. **Product Risks**

iii. **Development Process Risks**

a) **Developer errors**

- **Poor training due to budgetary constraints**
- **Inadequate skills and software tools.**
- **Environment problems**
- **Interruptions or distractions at office.**
- **Problems in personnel acquisition and retention.**
- **Natural disasters such as earth quake, fire**

i. **Development Process Risks**

b) **Project Risks**

- **If project risks become real, it is likely that project schedule will slip and that costs will increase.**
- **Project risks identify:**
  - **Potential budgetary constraints**
  - **Schedule**

- **Personnel (staffing and organization)**
- **Resource, customer, and requirements problems and their impact on a software project**

i. **Development Process Risks**

c) **Business risks**

- **Business risks threaten the viability of the software to be built.**
- **Business risks often jeopardize the project or the product.**
- **Candidates for the top five business risks are:**

1. **Building an excellent product or system that no one really wants (market risk)**
2. **Building a product that no longer fits into the overall business strategy for the company (strategic risk)**

i. **Development Process Risks**

c) **Business risks (Cont.)**

- **Candidates for the top five business risks are:**

3. **Building a product that the sales force doesn't understand how to sell**
4. **Losing the support of senior management due to a change in focus or a change in people (management risk), and**
5. **Losing budgetary or personnel commitment (budget risks). It is extremely important to note that simple categorization won't always work. Some risks are simply unpredictable in advance.**

i. **Development Process Risks**

d) **Unsatisfied employees can also become a risk to an organization.**

- **For example:**

- **A sacked employee can use password snuffers to gain unauthorized access.**
- **A dismissed person can flood the system with senseless messages.**
- **An unsatisfied employee can also try to sabotage the project work by destroying files and programs.**

i. **Development Process Risks**

e) **Poor Management Objective**

- **If the language in the management objective is ambiguous and not stated clearly, the risk management program will not function properly.**
- **Narrowly focused and changing objectives that are not updated can also be counted as risks.**
- **Lack of contingency plans, incomplete cost estimates, and unrealistic schedules are also potential risks in a project.**
- **Similarly, unrealistic performance standards are also potential risks to the development process.**

3. **Types of Risks**

i. **Development Process Risks**

a) **Developer errors**

b) **Project risks**

c) **Business risks**

d) **Unsatisfied employees**

e) **Poor management objectives**

ii. **Product Risks**

ii. **Product Risks**

- **Changing requirements during product development.**
- **Incomplete and unclear requirements**
- **Problems in meeting design specifications**
- **Project deliverables or objectives are not clearly defined**
- **Technical data is missing**

ii. **Product Risks**

- **Several alternatives at any given time during the project**
- **If errors are not recognized during the design phase**
- **Size and complexity of the product**

- **Achieving client acceptance of the product.**

4. **Risk Management Process**

i. Risk Identification

ii. Risk Analysis

iii. Risk Mitigation

- **Fig 1: Risk Management Process**

4. **Risk Management Process**

- **The process of risk management begins during the analysis phase of software development life cycle.**
- **However, the actual process of managing risks continues throughout the product development phase.**
- **Risk management is a dynamic process because it deals with the activities that are yet to happen.**

4. **Risk Management Process**

- **Risk management has a two-fold agenda.**
    - **Deciding actions for preventing risks from happening**
    - **Deciding actions for tackling risks that materialize.**
- **Therefore, risk management is all about pre-empting a risk, coming up with a plan for resolving the risk, and finally executing the plan.**
- **Summary**

1. **Fundamentals**
    - **What is it?**
    - **Why is it necessary?**
    - **Who does it?**
    - **Why is it important?**
    - **What are the steps?**
    - **What is the work product?**

- **How do I ensure that I've done it right?**

2. **Risk & Change Management Concepts**
3. **Types of Risk**
4. **Risk Management Process**

Lecture41

- **Software Project Management**
- **Reference to Previous Session**

1. **Fundamentals**
    - **What is it?**
    - **Why is it necessary?**
    - **Who does it?**
    - **Why is it important?**
    - **What are the steps?**
    - **What is the work product?**
    - **How do I ensure that I've done it right?**
2. **Risk & Change Management Concepts**
3. **Types of Risk**
4. **Risk Management Process**

- **Risk & Change Management (Cont.)**

4. **Risk Management Process**
5. **Risk Components and Drivers**
6. **Developing a Risk Table**
7. **Reactive VS. Proactive Risk Strategies**
8. **Risk Mitigation, Monitoring, and Management**
9. **Safety Risks and Hazards**

10. **The RMMM Plan**

11. **Risk Management Process**

a) **Risk Identification**

b) **Risk Analysis**

c) **Risk Mitigation**

4. **Risk Management Process**

a) **Risk Identification**

- **Risk Identification involves:**
  - **Identifying risks that may occur on a particular project**
  - **Determining which risks might affect the project**
  - **Documenting their characteristics**
- **Risk Management Process**

b) **Risk Identification (Cont.)**

- **Participants in risk identification activities can include the following, where appropriate:**
  - **Project manager**
  - **Project team leaders**
  - **Project team**
  - **Risk management team if assigned**
  - **Subject matter experts from outside the project team**
  - **Customers**
  - **End users**
  - **Other project managers**
  - **Stakeholders**
  - **Outside risk management experts**
- **Risk Management Process**

c) **Risk Identification (Cont.)**

- **Risk Identification is an iterative process because new risks may become known as the project progresses through its life cycle.**
- **The frequency of iteration and who participates in each cycle will vary from case to case.**
- **The project team should be involved in the process so that they can develop and maintain a sense of ownership of and responsibility for the risks and associated risk response actions.**

4. **Risk Management Process**

a) **Risk Identification (Cont.)**

- **Persons outside the team may provide additional objective information.**
- **The Risk Identification process usually leads to the Qualitative Risk Analysis process.**
- **Alternatively, it can lead directly to the Quantitative Risk Analysis process when conducted by an experienced risk manager.**
- **On some occasions simply the identification of a risk may suggest its response, and these should be recorded for further analysis and implementation in the Risk Response Planning process .**

4. **Risk Management Process**

a) **Risk Identification (Cont.)**

i. **The project manager gathers information about the potential risks in the project.**

ii. **The project manager plans the strategies for avoiding risks or controlling them.**

iii. **The project team conducts brainstorming sessions and discussions among team members about the requirements document.**

iv. **They discuss the available technology, manpower, prevailing environment, and all project-related factors.**

4. **Risk Management Process**

a) **Risk Identification (Cont.)**

v. **The project manager picks up the thread from these and creates a risk log.**

vi. **After the risk log is prepared, the project manager calls a meeting within the team and technical experts to discuss the risk log and the mitigation plans.**

vii. **An effective way of identifying' risks is using a questionnaire.**

4. **Risk Management Process**

a) **Risk Identification (Cont.)**

vii. **One method for identifying risks is to create a risk item checklist.**

viii. **The checklist can be used for risk identification and focuses on some subset of known and predictable risks in the following generic subcategories:**

ix. ***Product size*** **-risks associated with the overall size of the software to be built or modified.**

x. ***Business impact*** **-risks associated with constraints imposed by management or the marketplace.**

xi. ***Customer characteristics*** **-risks associated with the sophistication of the customer and the developer's ability to communicate with the customer in a timely manner.**

4. **Risk Management Process**

a) **Risk Identification (Cont.)**

d) ***Process definition*** **-risks associated with the degree to which the software process has been defined and is followed by the development organization.**

e) ***Development environment*** **-risks associated with the availability and quality of the tools to be used to build the product.**

f) ***Technology to be built*** **-risks associated with the complexity of the system to be built and the newness of the technology that is packaged by the system.**

g) ***Staff size and experience*** **-risks associated with the overall technical and project experience of the software engineers who will do the work.**

4. **Risk Management Process**

a) **Risk Identification (Cont.)**

viii. **During risk identification, you obtain answers to the following queries:**

ix. **Why is the risk important?**

x. **What information is needed to track the status of the risk?**

xi. **Who is responsible for the risk management activity?**

xii. **What resources are needed to perform the activity?**

xiii. **What is the detailed plan to improve the risk and / or mitigate it?**

4. **Risk Management Process**

b) **Risk Analysis**

i. **After identifying the risks, the project manager needs to analyze the risks.**

ii. **Uncertainty and loss are the two characteristics of risk.**

iii. **The uncertainty factor in risk means that the unknown event mayor may not happen.**

iv. **While analyzing risks, the project manager needs to quantify the level of uncertainty and the degree of loss.**

v. **Risk Management Process**

c) **Risk Analysis (Cont.)**

i. **Based on this, the project manager plans schedules and costs.**

ii. **During analysis, information on risk is converted into information on decision-making.**

iii. **Analysis provides the basis for the project manager to work on the "right" risks.**

iv. **Risk Management Process**

d) **Risk Analysis (Cont.)**

i. **Various tasks involved in risk analysis are:**

ii. **WBS elements are identified**

iii. **One of the tasks in the risk analysis phase is to describe the risk. The risk can be related to:**

- **Product**
- **Process**
- **Organization**
- **Clients**
- **Infrastructure**

iv. **Risk Management Process**

e) **Risk Analysis (Cont.)**

i. **Various tasks involved in risk analysis are:**

ii. **WBS elements are evaluated to determine the risk events.**

iii. **Then the project manager quantifies the probability of occurrence of risk.**

- **The project manager can assign probability values between 0 and 1. For example, a risk with a low probability of occurrence is marked 0.2 while that with a high probability of occurrence is marked 0.8. The reason why a particular risk has a high or low probability depends on the actual circumstance of the project.**

iv. **Risk Management Process**

f) **Risk Analysis (Cont.)**

i. **Various tasks involved in risk analysis are:**

ii. **Risks are rated depending on their probability of occurrence.**

iii. **Based on the probability of risk, the project manager identifies the impact of the risk.**

iv. **The impact of risk on cost, schedule, and quantity needs to be calculated and graded.**

v. **The impact of risk can be graded on a scale of 1 to 10, 1 being the lowest, and 10 being the highest.**

4. **Risk Management Process**

b) **Risk Analysis (Cont.)**

viii. **Various tasks involved in risk analysis are:**

ix. **Then the risk factor is calculated by multiplying the probability of risk and the impact of risk.**

x. **Finally, each risk is prioritized relative to other risks. The risk factor is used to prioritize the identified risks.**

Example:

- **The risk with a probability value 0.1 and an impact value 2 will have minimal impact.**
- **While risks close to probability value 0.8 and with an impact value 9 will have greater impact**.

4. **Risk Management Process**

c) **Risk Mitigation**

i. **Risk mitigation is the best possible approach adopted by the project manager to avoid risks from occurring.**

ii. **The probability of the risk occurring and the potential impact of the risk can be mitigated by dealing with the problem early in the project.**

iii. **Essentially, risk mitigation involves three possibilities and the project manager needs to adopt a risk mitigation strategy aimed at them.**

4. **Risk Management Process**

c) **Risk Mitigation (Cont.)**

iv. **The three possibilities include:**

v. **Risk avoidance**

vi. **Risk monitoring**

vii. **Contingency planning**

4. **Risk Management Process**

c) **Risk Mitigation (Cont.)**

1. **Risk Avoidance**

2. **To avoid risks from occurring, the project team prepares the risk plan before the commencement of the project.**

3. **The project team identifies the potential risks and prioritizes them based on their probability of occurrence and impact.**

4. **Then, the team prepares a plan for managing risks. In most software projects, this plan is popularly called the risk management plan (RMMM).**

4. **Risk Management Process**

c) **Risk Mitigation (Cont.)**

1. **Risk Avoidance (Cont.)**

2. **To prepare the risk management plan:**

3. **The project team first identifies and assesses the risks associated with the project.**

4. **Then, the probability of occurrence of each risk is estimated and the possible impact is calculated. In the plan, the probable cost and damage is also quantified.**
5. **The project team also identifies the contingency plans for all the identified risks.**

4. **Risk Management Process**

c) **Risk Mitigation (Cont.)**

1. **Risk Avoidance (Cont.)**
2. **To prepare the risk management plan:**
3. **The contingency plan for each risk is:**
   - **Based on the project's defined operational software process.**
   - **Modified throughout the software development life cycle of the project based on the changes taking place.**
   - **Risk Management Process**

d) **Risk Mitigation (Cont.)**

1. **Risk Avoidance (Cont.)**
2. **To prepare the risk management plan:**
3. **The contingency plan for each risk is: (Cont.)**
   - **Includes the cost, in terms of effort, in carrying out the plan.**
   - **The software risks identified are tracked, reassessed, and re-planned at the end of each phase.**
   - **The project manager revisits the plan if significant changes are introduced in the software project.**

4. **Risk Management Process**

c) **Risk Mitigation (Cont.)**

2. **Risk Monitoring**
3. **As the project proceeds, risk-monitoring activities commence.**
4. **It is not possible to monitor closely all the risks that are identified for the project.**
5. **For example, if 100 risks are identified for a project, only top 20 risks are monitored.**

6. **There are re-planning checkpoints where the information obtained from monitoring the risks is used to refine the risk assessments and management plan.**

4. **Risk Management Process**

c) **Risk Mitigation (Cont.)**

2. **Risk Monitoring (Cont.)**

3. **The project manager monitors the top 20 percent of the factors that may indicate the status of the risks in the project.**

4. **In the case of large teams, the project manager also needs to monitor the attitude of the team members and their problems.**

5. **This helps the project manager monitor any possible team-related risks.**

4. **Risk Management Process**

c) **Risk Mitigation (Cont.)**

2. **Risk Monitoring (Cont.)**

3. **Besides monitoring the top 20 percent of the risks, the project manager needs to monitor the mitigation steps also.**

4. **Consider an example: To ensure that particular software is browser-independent, the software is created on the lowest compatible browser.**

5. **Such software will work on any browser thus making it browser-independent.**

6. **Risk Management Process**

d) **Risk Mitigation (Cont.)**

2. **Risk Monitoring (Cont.)**

3. **Therefore, mitigating a project risk involves working hard at reducing the possibility that the risk will ever occur.**

4. **Mitigation includes nearly all actions that a project team takes to overcome risks.**

5. **For example, choosing a more expensive but proven technology over, a newer, less expensive technology is a step toward mitigating project risks.**

4. **Risk Management Process**

c) **Risk Mitigation (Cont.)**

3. **Contingency Planning**
4. **The possibility of contingency planning arises when mitigation efforts fail and risk becomes a reality.**
5. **Contingency planning is used to monitor risks and invoke a predetermined response.**
6. **According to the plan, a trigger is set up. If the trigger is reached, the contingency plan is put into effect.**
7. **Risk Management Process**

d) **Risk Mitigation (Cont.)**

3. **Contingency Planning (Cont.)**
4. **Contingency planning involves maintaining an alternative plan if the original plan fails.**
5. **A simple example could be the savings people make for a rainy day.**
6. **Contingency plans are a must for the top 20 percent of the risks identified.**
7. **These plans are put to use after the risks become a reality.**
8. **Risk Management Process**

e) **Risk Mitigation (Cont.)**

3. **Contingency Planning (Cont.)**
4. **The importance of contingency planning can be realized from this example.**
5. **Despite the massive attack on WTC, the stock markets in the US resumed functioning within a few days.**
6. **This was possible because the finance companies had backed up their data and information on computers elsewhere.**
7. **The contingency planning of finance companies prevented the risk of huge data loss for the stock market.**
8. **Risk Management Process**

d) **Managing Risks: An Example**

- **Your organization is a vendor of software solutions.**
- **A bus transport company wants you to develop a Schedule Adherence system.**

- **The team that will develop this software is new and the platform selected for development is also new to your organization.**
- **The project team needs to be trained intensively for this.**
- **Risk Management Process**

e) **Managing Risks: An Example (Cont.)**

- **During this project, the team is expected to manage a large volume of data.**
- **The team has never had any experience in managing such a large volume of data.**
- **The system also needs to use this data to generate various MIS reports related to delays or adherence of bus services.**

4. **Risk Management Process**

d) **Managing Risks: An Example (Cont.)**

- **The performance requirement is less than fifteen seconds for all popular browsers.**
- **Your organization is anticipating numerous requirement changes during the development process.**
- **The system needs to be implemented across several states in the country.**

4. **Risk Management Process**

e) **Managing Risks: An Example**

- **The data related to the system is highly confidential because it can provide an edge to the competitors.**
- **Now, as a project manager, you need to prepare a risk management plan for this project.**
- **The project starts on May 15 and should be completed on November 15.**
- **First, you need to identify the potential risks involved in the project. The potential risks to the project are described in following table.**

▪ **Potential Risks to a Project**

4. **Risk Management Process**

• **Managing Risks: An Example (Cont.)**

- **After identifying the risks, you need to estimate the probability of their occurrence and their impact on product development.**
- **Based on this, you calculate the risk factor and plan the mitigation steps.**
- **Your risk management plan is displayed in Table 1.**
- **In Table 1, the risk factors with high values are the high priority risks.**
- **Risk Management Process**

- **Managing Risks: An Example (Cont.)**
  - **Here, the high priority risks are design changes during development, performance risk due to high volume of data to be processed, and cross-browser compatibility.**
  - **The high priority risks need to be monitored closely and continuously.**
  - **The rest of the risks can be monitored periodically.**
  - **Certain risks need contingency planning despite being low in the priority list.**

4. **Risk Management Process**

- **Managing Risks: An Example (Cont.)**

▪ **Example:**

- **The risk due to involvement of new technology is a low priority risk.**
- **However, the probability of occurrence of this risk is 0.5, which is fairly high.**
- **This calls for contingency planning because you may not be aware of all the details of the new technology.**
- **Suppose the system fails to respond if the number of users exceeds a certain number.**
- **Risk Management Process**

- **Managing Risks: An Example (Cont.)**
  - **Example: (Cont.)**
  - **In such a situation, you need to have a contingency plan ready.**
  - **You need to discuss with the vendor regarding a workaround for such a situation.**
  - **The workaround should be ready in the case of failure of the system.**

5. **Risk Components and Drivers**

- **The U.S. Air Force has written a pamphlet that contains excellent guidelines (or software risk identification and abatement.**
- **The Air Force approach requires that the project manager identify the risk drivers that affect software risk components:**
    a) **Performance**
    b) **Cost**
    c) **Support**
    d) **Schedule**

5. **Risk Components and Drivers**

- **In the context of this discussion, the risk components are defined in the following manner:**
    a) ***Performance risk*** **- the degree of uncertainty that the product will meet its requirements and be fit for its intended use.**
    b) ***Cost risk*** **- the degree of uncertainty that the project budget will be maintained.**
    c) ***Support risk*** **- the degree of uncertainty that the resultant software will be easy to correct, adapt, and enhance.**
    d) ***Schedule risk*** **- the degree of uncertainty that the project schedule will be maintained and that the product will be delivered on time.**

5. **Risk Components and Drivers**

- **The impact of each risk driver on the risk component is divided into one of four impacts categories:**
    - **Negligible**
    - **Marginal**
    - **Critical**
    - **Catastrophic**

5. **Risk Components and Drivers**

- **Referring to table 1, a characterization of the potential consequences of errors (rows labeled 1) or a failure to achieve a desired outcome (rows labeled 2) are described.**
- **The impact category is chosen based on the characterization that best fits the description in the table.**

- **Table 1: Risk Components and Drivers**
- **Table 2: Sample risk table prior to sorting**

6. **Developing a Risk Table**

- **A risk table provides a project manager with a simple technique for risk projection.**
- **A sample risk table is illustrated on screen.**

i. **A project team begins by listing all risks (no matter how remote) in the first column of the table.**

ii. **This can be accomplished with the help of the risk item check-lists referenced in Section 6.3.**

iii. **Each risk is categorized in the second column (e.g., PS implies a project size risk, BU implies a business risk).**

6. **Developing a Risk Table**

iv. **The probability of occurrence of each risk is entered in the next column of the table.**

v. **The probability value for each risk can be estimated by team members individually.**

vi. **Individual team members are polled in round-robin fashion until their assessment of risk probability begins to converge.**

7. **Developing a Risk Table**

- **A project team begins by listing all risks (no matter how remote) in the first column of the table.**
- **This can be accomplished with the help of the risk item check-lists.**
- **Each risk is categorized in the second column (e.g., PS implies a project size risk, BU implies a business risk).**

6. **Developing a Risk Table**

- **The probability of occurrence of each risk is entered in the next column of the table.**
- **The probability value for each risk can be estimated by team members individually.**
- **Individual team members are polled in until their assessment of risk probability begins to converge.**

6. **Developing a Risk Table**

- **Next, the impact of each risk is assessed.**
- **Each risk component is assessed using the characterization presented in table 2, and an impact category is determined.**
- **The categories for each of the four risk components are averaged to determine an overall impact value.**
    - **Performance**
    - **Support**
    - **Cost**
    - **Schedule**
- **Risk and management concern**
- **Risk and management concern**
- **Risk impact and probability have a distinct influence on management concern.**
- **A risk factor that has a high impact but a very low probability of occurrence should not absorb a significant amount of management time.**
- **However, high-impact risks with moderate to high probability and low-impact risks with high probability should be carried forward into the risk analysis steps that follow.**

7. **Reactive VS. Proactive Risk Strategies**

- **At best, a reactive strategy monitors the project for likely risks.**
- **Resources are set aside to deal with them, should they become actual problems.**
- **More commonly, the software team does nothing about risks until something goes wrong.**
- **Then, the team flies into action in an attempt to correct the problem rapidly.**
- **This is often called a *fire fighting mode.***
- **When this fails, "Crisis Management takes over, and the project is in real jeopardy.**

7. **Reactive VS. Proactive Risk Strategies**

- **A considerably more intelligent strategy for risk management is to be proactive.**
- **A proactive strategy begins long before technical work is initiated.**

- **Potential risks are identified, their probability and impact are assessed and they are ranked by importance.**
- **Then, the software team establishes a plan for managing risk.**
- **The primary objective is to avoid risk, but because not all risks can be avoided, the team works to develop a contingency plan that will enable it to respond in a controlled and effective manner.**

8. **Risk, Mitigation, Monitoring and Management**

- **All of the risk analysis activities presented to this point have a single goal to assist the project team in developing a strategy for dealing with risk.**
- **An effective strategy must consider three issues:**
    - **Risk avoidance**
    - **Risk monitoring**
    - **Risk management and contingency planning**
    - **If a software team adopts a proactive approach to risk, avoidance is always the best strategy.**
- **This is achieved by developing a plan for risk mitigation.**

8. **Risk, Mitigation, Monitoring and Management**

- **Example: Assume that high staff turnover is noted as a project risk.**
- **Based on past history and management intuition, the likelihood of high turnover is estimated to be 0.70 (70 per cent, rather high) and the impact is projected at level 2.**
- **That is, high turnover will have a critical impact on project cost and schedule.**

8. **Risk, Mitigation, Monitoring and Management**

- **To mitigate this risk, project management must develop a strategy for reducing turnover. Among the possible steps to be taken are**
- **Meet with current staff to determine causes for turnover (e.g., poor working conditions, low pay, and competitive job market).**
- **Mitigate those causes that are under our control before the project starts.**
- **Once the project commences, assume turnover will occur and develop techniques to ensure continuity when people leave.**

8. **Risk, Mitigation, Monitoring and Management**

- **Organize project teams so that information about each development activity is widely dispersed.**
- **Define documentation standards and establish mechanisms to be sure that documents are developed in a timely manner.**
- **Conduct peer reviews of all work (so that more than one person is “up to speed").**
- **Assign a backup staff member for every critical technologist.**

8. **Risk, Mitigation, Monitoring and Management**

- **In addition to monitoring these factors, the project manager should monitor the effectiveness of risk mitigation steps.**
- **This is one mechanism for ensuring continuity, should a critical individual leave the project.**
- **The project manager should monitor documents carefully to ensure that each can stand on its own and that each imparts information that would be necessary if a newcomer were forced to join the software team somewhere in the middle of the project.**

8. **Risk, Mitigation, Monitoring and Management**

- **Risk management and contingency planning assumes that mitigation efforts have failed and that the risk has become a reality.**
- **Continuing the example, the project is well underway and a number of people announce that they will be leaving.**
- **If the mitigation strategy has been followed, backup is available, information is documented, and knowledge has been dispersed across the team.**

8. **Risk, Mitigation, Monitoring and Management**

- **In addition, the project manager may temporarily refocus resources (and readjust the project schedule) to those functions that are fully staffed, enabling newcomers who must be added to the team to "get up to speed."**
- **Those individuals who are leaving are asked to stop all work and spend their last weeks in "knowledge transfer mode.**
- **This might include video-based knowledge capture, the development of "commentary documents," and/or meeting with other team members who will remain on the project.**

8. **Risk, Mitigation, Monitoring and Management**

- **It is important to note that RMMM steps incur additional project cost.**
    - **For example, spending the time to "backup" every critical technologist costs money.**
    - **Part of risk management, therefore, is to evaluate when the benefits accrued by the RMMM steps are outweighed by the costs associated with implementing them.**

8. **Risk, Mitigation, Monitoring and Management**
    - **In essence, the project planner performs a classic cost/benefit analysis.**
    - **If risk aversion steps for high turnover, it will increase both project cost and duration by an estimated 15 %, but the predominant cost factor is "backup," management may decide not to implement this step.**
    - **On the other hand, if the risk aversion steps are projected to' increase costs by 5 % and duration by only 3% management will likely put all into place.**

9. **Risk, Mitigation, Monitoring and Management**
    - **For a large project, 30 or 40 risks may be identified.**
    - **If between three and seven risk management steps are identified for each, risk management may become a project in itself!**
    - **For this reason, we adapt the Pareto 80-20 rule to software risk.**
    - **Experience indicates that 80% of the overall project risk (i.e., 80% of the potential for project failure) can be accounted for by only 20% of the identified risks.**

10. **Risk, Mitigation, Monitoring and Management**
    - **The work performed during earlier risk analysis steps will help the planner to determine which of the risks reside in that 20% (e.g., risks that lead to the highest risk exposure.**
    - **For this reason, some of the risks identified, assessed, and projected may not make it into the RMMM plan - they don't fall into the critical 20% (the risks with highest project priority).**

9. **Safety Risks and Hazards**

- **Risk is not limited to the software project itself.**
- **Risks can occur after the software has been successfully developed and delivered to the customer.**
- **These risks are typically associated with the consequences of software failure in the field.**

- **In the early days of computing, there was reluctance to use computers (and soft- ware) to control safety critical processes such as nuclear reactors, aircraft flight control, weapons systems, and large-scale industrial processes.**

9. **Safety Risks and Hazards**

- **Although the probability .of failure of a well-engineered system was small, an undetected fault in a computer- based control or monitoring system could result in enormous economic damage or, worse, significant human injury or loss of life.**
- **But the cost and functional benefits of Computer-based control and monitoring far outweigh the risk.**
- **Today, computer hardware and software are used regularly to control safety critical systems.**
- **When software is used as part of a control system, complexity can increase by an order of magnitude or more.**

9. **Safety Risks and Hazards**

- **Subtle design faults induced by human error-something that can be uncovered and eliminated in hardware-based conventional control-become much more difficult to uncover when software is used.**
- **Software safety and hazard analysis are software quality assurance activities that focus on the identification and assessment of potential hazards that may affect software negatively and cause an entire system to fail.**
- **If hazards can be identified early in the software engineering process, software design features can be specified that will either eliminate or control potential hazards.**

10. **The RMMM Plan**

- **A risk management strategy can be included in the software project plan or the risk management steps can be organized into a separate Risk Mitigation, Monitoring and Management Plan.**
- **The RMMM plan documents all work performed as part of risk analysis and are used by the project manager as part of the overall project plan.**
- **Some software teams do not develop a formal RMMM document.**
- **Rather, each risk is documented individually using a risk information sheet (RIS).**

10. **The RMMM Plan**

- **In most cases, the RIS is maintained using a database system, so that creation and information entry, priority ordering, searches, and other analysis may be accomplished easily.**
- **Once RMMM has been documented and the project has begun, risk mitigation and monitoring steps commence.**
- **As we have already discussed, risk mitigation is a problem avoidance activity.**

10. **The RMMM Plan**

- **Risk monitoring is a project tracking activity with three primary objectives:**
    1. **To assess whether predicted risks do, in fact, occur.**
    2. **To ensure that risk aversion steps defined for the risk are being properly applied.**
    3. **To collect information that can be used for future risk analysis.**
- **In many cases, the problems that occur during a project can be traced to more than one risk.**
- **Another job of risk monitoring is to attempt to allocate origin (what risk (s) caused which problems throughout the project).**
- **Summary**

2. **Risk Management Process**
3. **Risk Components and Drivers**
4. **Developing a Risk Table**
5. **Reactive VS. Proactive Risk Strategies**
6. **Risk Mitigation, Monitoring, and Management**
7. **Safety Risks and Hazards**
8. **The RMMM Plan**

Lecture42

- **Software Project Management**

I. **Overview of Management**

II. **Software Developmental Fundamentals**

III. **Process**

IV. **Planning**

V. **Organization**

VI. **Estimation**

VII. **Work Breakdown Structure**

VIII. **Scheduling**

IX. **Risk And Change Management**

X. **Quality**

1. **Quality Concept**
2. **Producing quality software**
3. **Quality Control**
4. **Quality Control Myths**
5. **Resources for quality control**
6. **Quality Assurance**
7. **Software Quality Assurance plan**
8. **Example of contents of a Software Quality Assurance plan**
9. **Software quality metrics**
10. **Some General Guidelines**

- **The problem of quality management is not what people don't know about it. The problem is what they think they do know…**

Philip Crosby

1. **Quality Concepts**

- **Quality is conformance to:**
    - **Explicitly stated functional and performance requirements**
    - **Explicitly documented development standards**
    - **Implicit characteristics that are expected of all professionally developed software.**

1. **Quality Concepts**

- **Quality is difficult to define, especially when applied to a product development contract.**
- **Though not all software is developed under contract, quality still primarily remains a concern of the customer**

1. **Quality Concepts**

- **The IEEE's standard glossary of software engineering terminology (IEEE 1987b), contains four separate definitions for software quality:**

    1. **The totality of features and characteristics of a software product that bear on its ability to satisfy given needs; for example, conform to specifications.**
    2. **The degree to which software possesses a desired combination of attributes.**

1. **Quality Concepts**

    3. **The degree to which a customer or user perceives that software meets his or her composite expectations.**
    4. **The composite characteristics of software that determine the degree to which the software in use will meet the expectations of the customer.**
    5. **The US DOD (1988b) defines software quality as: The ability of a software product to satisfy its specified requirements.**

2. **Quality Concepts**

- **Remember**

    - **Quality should be measured in terms of the customer's expectations.**
    - **However, the customer's perspective is subjective.**
    - **In this session we look at quality control from the project manager's perspective, which must regard the achievement of quality control as an objective and systematic process.**

1. **Quality Concepts**

What is it?

- **It's not enough to talk the talk by saying that *soft* ware quality is important, you have to:**

    1. **Explicitly define what is meant when you say 'software quality'.**

2. **Create a set of activities that will help ensure that every software engineering Work product exhibits high quality**
3. **Perform quality assurance activities on every software project**
4. **Use metrics to develop strategies to improving your software process and as a consequence the quality of the end product.**

1. **Quality Concepts**

Who does it?

- **Everyone involved in the software engineering process is responsible for quality.**

1. **Quality Concepts**

Why is it important?

- **You can do it right, *or* you can do it over again.**
- **If a software team stresses quality in all software engineering activities, it reduces the amount of rework that it must do that results in lower costs, and more importantly, improved time-to-market.**

1. **Quality Concepts**

What are the steps?

- **Before software quality assurance activities can be initiated, it is important to define 'software quality' at a number of different levels of abstraction.**
- **Once you understand what quality is, a software team must identify a set of SQA activities that will filter errors out of work products before they are passed on.**

1. **Quality Concepts**

What is the work product?

- **A Software Quality Assurance Plan is created to define a software team's SQA strategy.**
- **During analysis, design, and code generation, the primary SQA work product is the formal technical review summary report.**
- **During testing test plans and procedures are produced.**
- **Other work products associated with process improvement may also be generated.**

1. **Quality Concepts**

How do I ensure that I've done it right?

- **Find errors before they become defects!**
- **That is, work to improve your defect removal efficiency, thereby reducing the amount of rework that your software team has to perform.**

1. **Quality Concepts**

- **When we examine an item based on its measurable characteristics, two kinds of quality may be encountered:**

    a) **Quality of design**

    b) **Quality of conformance**

1. **Quality Concepts**

a) ***Quality of design*** **refers to the characteristics that designers specify for an item.**

- **The grade of materials, tolerances, and performance specifications all contribute to the quality of design.**
- **As higher-grade materials are used, tighter tolerances and greater levels of performance are specified, the design quality of a product increases, if the product is manufactured according to specification.**

1. **Quality Concepts**

b) ***Quality of conformance*** **is the degree to which the design specifications are followed during manufacturing.**

- **Again, the greater the degree of conformance, the higher is the level of quality of conformance.**

1. **Quality Concepts**

- **In software development, quality of design encompasses requirements, specifications, and the design of the system.**
- **Quality of conformance is an issue focused primarily on implementation.**
- **If the implementation follows the design and the resulting system meets its requirements and performance goals, conformance quality is high.**

1. **Quality Concepts**

- **But are quality of design and quality of conformance the only issues that software engineers must consider?**
- **There is a more "intuitive" relationship of quality of a product:**

  User satisfaction = compliant product + good quality + delivery within budget and schedule
- **Quality is important, but if the user isn't satisfied, nothing else really matters.**

2. **Producing Quality Software**

- **As we have seen, one of the main problems in producing quality software is the difficulty in determining the degree of quality within the software.**
- **As there is no single widely accepted definition for quality, and because different people perceive quality in different ways, both the developer and the customer must reach agreement on metrics for quality.**
- **The method of measuring quality may differ for different projects.**

2. **Producing Quality Software**

i. **Three distinct components of quality are:**

a) **An objectively assessable component:**

The quality of a product is objectively assessable when the characteristics of the product, as stated in the requirements specification, can be identified.

2. **Producing Quality Software**

i. **Three distinct components of quality are:**

b) **A subjectively assessable component:**

The quality of a product is subjectively assessable when the characteristics of the product comply with the customer's expectations.

a) **A non-assessable component:**

The quality of a product is non-assessable when it behaves according to our expectations in situations that have not been foreseen.

2. **Producing Quality Software**

ii. **For the quality of a software product to be assessable, as many characteristics as possible should be moved from the subjective and non-assessable components to the assessable component.**

iii. **Essentially, this means that the requirements specification must describe as many measurable characteristics of the product as possible.**

2. **Producing Quality Software**

iv. **Badly defined requirements are always a source of dispute between developer and customer.**

v. **Well-defined, detailed and measurable requirements minimize disputes and disagreements when the development of the product is complete.**

vi. **However, many development methods have a prolonged interval between the specification of requirements and the delivery of the product.**

2. **Producing Quality Software**

vii. **The determination of quality should not be postponed until development is complete.**

viii. **Effective software quality control requires frequent assessments throughout the development cycle.**

ix. **Thus, effective quality control coupled with a good requirements specification will clearly increase the quality of the final product.**

3. **Quality control**

i. **Quality control involves the series of inspections, reviews, and tests used throughout the software process to ensure each work product meets the requirements placed upon it.**

ii. **Quality control includes a feedback loop to the process that created the work product.**

iii. **The combination of measurement and feedback allows us to tune the process when the work products created fail to meet their specifications.**

iv. **This approach views quality control as part of the manufacturing process.**

3. **Quality control**

v. **Quality control activities may be fully automated, entirely manual, or a combination of automated tools and human interaction.**

vi. **A key concept of quality control is that all work products have defined; measurable specifications to which we may compare the output of each process.**

vii. **The feedback loop is essential to minimize the defects produced.**

4. **Quality Control Myths**

- **The establishment of effective quality control frequently encounters various misconceptions and myths:**

1. **Myth: Quality costs money**

    - **This is one of the most common myths (not only in software development).**

        - **In fact, quality in software usually saves money**

    - **Poor quality breeds failure**

        - **There is a positive correlation between failures and cost in that it is more expensive to remove execution failures designed into software than to design software to exclude execution failures.**

4. **Quality Control Myths**

- **Myth: Software failures are unavoidable**

    - **This is one of the worst myths because the statement is partly true, and is therefore often used as an excuse to justify poor quality software.**

    - **The claim that 'there is always another bug' should never be a parameter in the design or implementation of software**

5. **Resources for quality control**

i. **When the SQA (Software Quality Control) mandate includes configuration control activities, the required resources will also include those required for configuration control.**

ii. **Merging SQA and configuration control is not uncommon, and can eliminate some duplication of assignments and activities.**

iii. **Two alternative organizational charts are shown in Fig. Note that for small projects, merging the two groups may mean simply assigning both responsibilities to the same person.**

- **Configuration control and software quality control: *(a)* separate groups; *(b)* combined groups**

5. **Resources for quality control**

iv. **Though many tools are common to both quality control and configuration control, few tools are specifically designed for quality control.**

v. **The following are some of the general support tools that can be useful in supporting SQA activities:**

a) **Documentation utilities**

b) **Software design tools**

c) **Debugging aids**

d) **Structured preprocessors**

e) **File comparators**

5. **Resources for quality control**

f) **Structure analyzers**

g) **Standards auditors**

h) **Simulators**

i) **Execution analyzers**

j) **Performance monitors**

k) **Statistical analysis packages**

l) **Integrated CASE tools**

m) **Test drivers**

n) **Test case generators**

6. **Resources for quality control**

vi. **These tools support quality control in all phases of software development.**

vii. **Documentation aids can provide partially automatic document writers, spelling checkers and thesauruses etc.**

viii. **Structured preprocessors (such as the UNIX utility lint*)* are useful both to standardize code listings and to provide additional compile-time warnings that compilers often overlook.**

5. **Resources for quality control**

ix. **Early warnings regarding possible execution time problems can be provided by simulators, execution time analyzers and performance monitors.**

x. **Substantial software system testing can often be performed automatically by test suite generators and automatic test executors.**

6. **Quality Assurance**

i. **IEEE Standard 12207 defines QA this way:**

- **“The quality assurance process is a process for providing adequate assurance that the software products and processes in the product life cycle conform to their specific requirements and adhere to their established plans.”**

ii. **This sentence uses the word process three times. That is a key aspect of QA—it is not a single technology, but also a method and approach.**

6. **Quality Assurance**

iii. **Another key point is that quality is not treated as a philosophical issue, but, rather, as measurably meeting expectations and conforming to requirements. The rigor of the process should be chosen to suit the needs of the product and organization.**

iv. **Finally, QA is about providing assurance and credibility: the product should work right, and people should believe that it will work right.**

6. **Quality Assurance**

v. **What goes into QA? Testing, of course, is a key activity.**

vi. **There is, however, an adage that “you can’t test quality into a product.” A solid test plan should catch errors and give a measure of quality.**

vii. **A good QA plan ensures that the design is appropriate, the implementation is careful, and the product meets all requirements before release.**

viii. **An excellent QA plan in an advanced organization includes analysis of defects and continuous improvement. (This feedback loop is characteristic of mature organizations.)**

6. **Quality Assurance**

ix. **Quality assurance consists of the auditing and reporting functions of management.**

x. **The goal of quality assurance is to provide management with the data necessary to be informed about product quality, thereby gaining insight and confidence that product quality is meeting its goals.**

xi. **Of course, if the data provided through quality assurance identify problems, it is management's responsibility to address the problems, and apply the necessary resources to resolve quality issues.**

7. **The Software Quality Assurance plan**

i. **All SQA tools to be used during software development should be identified and described in the SQA plan.**

ii. **This plan includes a description of all required quality assurance resources and details of how they will be applied.**

iii. **Thus, at the start of the project SQA resources can be budgeted and procured as part of the required project development resources**

7. **The Software Quality Assurance plan**

iv. **The software quality assurance plan (SQAP), like the software configuration plan, is also part of the overall software project development plan.**

v. **The SQAP documents which resources are needed, how they should be used and which standards and procedures will be applied during the project.**

vi. **The SQAP then becomes the mandate for the quality assurance group during project development.**

7. **The Software Quality Assurance plan**

vii. **The issuance of this plan is the responsibility of the project manager, though in large projects it will usually be delegated to the quality assurance manager.**

viii. **The SQAP may appear as a separate document or as a section within the project development plan, and may include the configuration management plan (if this has not been documented separately).**

7. **The Software Quality Assurance plan**

ix. **The SQAP must cover subcontractors, vendors and suppliers, irrespective of whether or not these external entities have their own quality assurance organization.**

x. **In any project, the quality of external components is ultimately the concern of the project manager and the SQA organization.**

7. **The Software Quality Assurance plan**

xi. **When a system fails, it usually makes little difference whether the failure is due to an externally developed component or an in-house developed component.**

xii. **The plans for supervising these external groups must be adapted to the type of external components being provided *(off the shelf or new development)* and the type of organization *(do they have their own quality assurance organization?).***

7. **The Software Quality Assurance plan**

xiii. **The SQAP, as part of the project development plan, should be reviewed and updated periodically and whenever any requirements, project development procedures, methodologies or other relevant activities are changed.**

xiv. **The IEEE SQAP guide recommends periodic evaluation of two aspects of the plan:**

a) **The plan's content**

b) **The plan's implementation**

7. **The Software Quality Assurance plan**

xv. **The plan's content should be evaluated with regard to the specific SQAP standard used, to assure the plan's continuing compliance with the standard even when the characteristics of the software project change.**

xvi. **The plan's implementation should be evaluated in terms of the changing scope of the project, including the tasks and responsibilities referenced in the plan, and other new or changed characteristics of the project.**

7. **The Software Quality Assurance plan**

xvii. **When updating the SQAP the following project activities and events should be considered:**

a) **New or changed contractual requirements**

b) **Additional standards and policies**

c) **Additional project documents**

d) **Changes in the project's organizational structure**

e) **New tools and utilities**

f) **Additional subcontractors and vendors**

8. **Example of contents of a Software Quality Assurance plan**

1. **Software quality assurance organization and resource**

- **Organization structure**
- **Personnel skill level and qualifications**
- **Resources**

2. **SQA standards, procedures, policies and guidelines**

8. **Example of contents of a Software Quality Assurance plan**

3. **SQA documentation requirements**

    a) **List of all documentation subject to quality control**

    b) **Description of method of evaluation and approval**

4. **SQA software requirements**

    a) **Evaluation and approval of software**

    b) **Description of method of evaluation**

    c) **Evaluation of the software development process**

    d) **Evaluation of non-deliverable software**

    e) **Evaluation of reused software**

8. **Example of contents of a Software Quality Assurance plan**

5. **Evaluation of storage, handling and delivery of:**

▪

- **Project documents**
- **Software**
- **Data files**

6. **Reviews and audits**

8. **Example of contents of a Software Quality Assurance plan**

7. **Software configuration management (when not addressed in a separate document)**

8. **Problem reporting and corrective action**

9. **Evaluation of test procedures**

10. **Tools techniques and methodologies**

11. **Quality control of subcontractors, vendors and suppliers**

8. **Example of contents of a Software Quality Assurance plan**

12. **Additional control**

- **Miscellaneous control procedures**

- **Project specific control**

13. **SQA reporting, records and documentation**
    - **Status reporting procedures**
    - **Maintenance**
    - **Storage and security**
    - **Retention period**

9. **Software quality metrics**

i. **The quality of two products can be compared, and it is perfectly acceptable to claim that the quality of one product is greater than the quality of another.**

ii. **It is also acceptable to measure quality and deduce the extent of expected faults based on the measured result.**

iii. **The set of measurable values associated with the quality of a product is referred to as the product's quality *metrics.***

9. **Software quality metrics**

iv. **Software quality metrics can be used to determine the extent to which a software product meets its requirements.**

v. **The use of quality metrics increases the objectivity of the evaluation of product quality.**

9. **Software quality metrics**

iv. **Human evaluation of quality is subjective, and is therefore a possible source of disagreement, particularly between customer and developer.**

v. **Quality metrics, once defined, do indeed increase objectivity, but the definition itself is not necessarily objective and greatly depends upon the needs of the organization that produces the definition.**

9. **Software quality metrics**

viii. **The basic approach for applying software quality metrics requires:**

a) **The identification of all required software quality attributes. This is usually derived from the software requirements specification.**

b) **Determination of measurable values to be associated with each quality attribute. A description of the method by which each measurable value will be measured.**

c) **A procedure for documenting the results of measuring the quality of the software product.**

9. **Software quality metrics**

ix. **A set of many values can be used to determine the overall quality of a software product.**

x. **However, a single measure can be created to represent the overall quality of the software product.**

This requires:

a) **A weighted method for combining the measured quality attributes into a single measure of quality for the product.**

9. **Software quality metrics**

xi. **Some examples of software metrics are:**

- ***Reliability:*** **The percentage of time that the system is successfully operational (e.g. 23 out of 24 hours produces: 100 x (23/24) percent)**
- ***Recoverability:*** **The amount of time it takes for the system to recover after failure (e.g. 1hour to reload from backups and 30 minutes to reinitialize the system)**
- ***User-friendliness:*** **The amount of training time needed for a new user**

10. **Some general guidelines**

i. **The measurement of software quality should not be performed only at the end of a project.**

ii. **The degree of quality should be measured at regular intervals during development.**

iii. **Thus, any major reduction in the overall measure of quality should act as a warning for the project manager that collective action is required.**

iv. **High quality at the end of the project is achieved by assuring high quality throughout the development of the project.**

10. **Some general guidelines**

v. **In small projects, many SQA activities can be performed by the project manager.**

This includes the organization and supervision of reviews and audits, the evaluation and selection of development tools, and the selection and application of standards.

10. **Some general guidelines**

vi. **Test procedures and testing are always best when conducted by a separate independent team.**

The decision on whether supervision of testing activities can be assigned to SQA depends on many factors, including:

- **The independence of the SQA team,**
- **The size of the project and**
- **The 'complexity of the project**

10. **Some general guidelines**

vii. **When testing is performed by an independent test team, SQAS involvement will be minimal.**

In most other cases it will be the responsibility of the SQA team to plan and supervise the testing of the system.

10. **Some general guidelines**

viii. **As a general guideline, it is usually undesirable for SQA to be performed by a member of the development team.**

However, small projects often cannot justify the cost of a dedicated SQA engineer.

This problem can be solved by having a single SQA engineer responsible for two or three small projects (with each project funding its share of the SQA services).

10. **Some general guidelines**

ix. **The ability to control software quality is directly linked to the quality of the software requirements specification.**

Quality control requires the unambiguous specification of as many of the required characteristics of the software product as possible.

1. **Quality Concept**
2. **Producing quality software**
3. **Quality Control**

4. **Quality Control Myths**
5. **Resources for quality control**
6. **Quality Assurance**
7. **Software Quality Assurance plan**
8. **Example of contents of a Software Quality Assurance plan**
9. **Software quality metrics**
10. **Some General Guidelines**

Lectur43

- **Software Project Management**

I. **Overview of Management**

II. **Software Developmental Fundamentals**

III. **Process**

IV. **Planning**

V. **Organization**

VI. **Estimation**

VII. **Work Breakdown Structure**

VIII. **Scheduling**

IX. **Risk And Change Management**

X. **Quality**

XI. **Application Tools**

XII. **Commissioning & Migration**

XI. **Application Tools**

1. **Managing Tasks in Microsoft Project**
2. **Creating Tasks**
3. **Types of Tasks**

4. **Adding Constraints to Tasks**
5. **Adding Deadlines to Tasks**
6. **Setting Dependencies between Tasks**
7. **Lag and Lead Time between Tasks**
8. **Splitting Tasks**
9. **Assigning Resources & Costs to Tasks**
10. **Assigning Costs to Tasks**
11. **Tracking a Project Plan**
12. **Displaying Project Information Using Reports**
13. **Advantages of MS Project**
14. **Disadvantages of MS Project**

1. **Managing Tasks in MS Project**

i. **Project management is a highly specialized and professional task.**

ii. **The chances that a project can go haywire are very high because project management includes numerous activities, such as**

- **Planning**
- **Resource allocation**
- **Risk management**
- **Estimation**

1. **Managing Tasks in MS Project**

iii. **You can automate all these activities using a tool like *MS Project* to schedule and track your project plan.**

iv. **Specify tasks and the expected duration of each task and MS Project prepares a project schedule.**

v. **The schedule displays a plan consisting of start and end dates of each task.**

vi. **In Addition, it plots a pictorial Gantt chart. The Gantt chart view depicts the extension of tasks across days along with the corresponding resources.**

2. **Creating Tasks**

i. **To manage a project efficiently, treat it as a decomposable entity that can be analyzed and resolved.**

ii. **A problem first needs to be broken into units and then analyzed for solutions because it is always easier to focus on micro level units.**

iii. **Microsoft Project enables you to perform these tasks.**

2. **Creating Tasks**

iv. **Using Microsoft Project, you can create tasks for each phase, specify their duration, and assign resources to them.**

v. **You can also set relationships between tasks to identify the successor and predecessor tasks.**

3. **Types of Tasks**

- **Microsoft Project enables you to create three types of tasks:**

a) **General**

b) **Milestone**

c) **Recurring**

i. **The usual tasks that are independent and have no special characteristics related to their duration or time.**

ii. **Example: Analysis is a general task that XYZ Inc. plans to conduct.**

iii. **The duration of a task signifies the period for which the task lasts in a project schedule.**

iv. **The start date is preset as the current date unless you enter a specific date.**

v. **The finish date for the task is displayed as xxx days after the current date.**

vi. **Duration can be specified in minutes, hours, days, weeks, months, or years.**

vii. **If you are not sure of a confirmed duration, you can specify an estimated duration of a task.**

viii. **After a task is created, you; can specify the subtasks below it.**

i. **These tasks have no duration.**

ii. **They serve as markers or reference nodes for significant achievements in a project.**

iii. **The only difference is that after the task is created, you select an additional option named *Mark task as milestone* in the Task Information dialog box. Like:**

- **Project startup**
- **Project End**
- **Pilot Implemented**
- **SRS Document Sign Off**

iv. **Recurring tasks occur repeatedly at regular intervals. For example, a weekly project meeting with all the team members is a recurring task.**

v. **Frequency depends upon nature of project.**

4. **Adding Constraints to Tasks**

i. **Constraints are defined as limitations that negatively affect the schedule and quality of a project.**

ii. **A constraint also affects other constraints.**

iii. **For example, you may need more resources if you reduce the project duration.**

- **Duration of a project**
- **Resources of a Project**
- **Performance goals of a project**
- **Adding Deadlines to Tasks**

iv. **Deadlines are strict finish dates for a task. Target date to complete a task.**

v. **You add a deadline to a specific task if you track the progress of that task.**

vi. **Setting realistic deadlines to tasks also enables you to schedule the subsequent tasks.**

vii. **Therefore, you can track whether a task has exceeded the planned time or is completed before time.**

6. **Setting Dependencies between Tasks**

- **Means a particular task cannot begin until its preceding task is completed.**
- **For example, the creation of the high-level design of a project cannot start until the client signs off the SRS document.**

7. **Lag and Lead Time between Tasks**

i. **While adding dependencies to tasks, you can also specify lag and lead times for any two interdependent tasks.**

ii. **Lag time is defined as the excess time that is allowed to lapse between predecessor and the successor tasks.**

iii. **This results in a delay in the start of the successor task.**

7. **Lag and Lead Time between Tasks**

iv. **For example, you may use this option if you feel that the tasks of the analysis phase are taking more than the planned time to complete.**

v. **Consequently, you might not want to start the high-level design task as planned but postpone it.**

vi. **In this case, you would calculate the maximum lag time for the high-level design to start.**

8. **Splitting Tasks**

i. **A split in a task is defined as an abrupt and sudden interruption in the planned schedule.**

ii. **The need to split tasks arises because practically no task can be completed without a break.**

iii. **Due to resource crunch, a planned task is normally halted for some time.**

iv. **For example, at XYZ Inc., both the tasks to train developers and to test a software module require a Windows 2000 Server.**

8. **Splitting Tasks**

v. **However, depending upon the criticality of the tasks, you can split one of the non critical tasks.**

vi. **In this case, you decide to stop testing the software module and continue training the developers.**

vii. **After creating Testing task, you create Unit testing task under the Testing task.**

viii. **Next, you split the Unit testing task.**

9. **Assigning Resources to Tasks**

i. **Resources are assigned to tasks for:**

- **To track the costs of each resource**

- **To track the work completed and the material resources used in a project**

ii. **There are two types of resources:**

- **Work**
- **Material**

9. **Assigning Resources to Tasks**

iii. **Work resources refer to people &equipment.**

iv. **At end of a project, these resources are not consumed.**

v. **Unlike work resources, material resources are used completely by the end of the project.**

vi. **For example, to complete the task of writing the code for a module, you need human and material resources.**

vii. **You need material resources such as hardware, software, and connectivity.**

10. **Assigning Costs to Tasks**

i. **Controlling costs ensures the optimal use of resources.**

ii. **For example, to complete a specific task in a project plan, you can assign more resources to that task.**

iii. **Alternatively, you can increase the duration of that task depending on the cost of the resources that are used by that task.**

10. **Assigning Costs to Tasks**

iv. **You can assign two rates for any type of a resource.**

v. **The rate for work resources are calculated on an hourly basis.**

vi. **However, the rate at which the materia1 resources are calculated can be:**

- ***Standard rate:*** **The normal rate at which a resource is procured**
- ***Overtime rate:*** **The rate that you pay if you overuse a work resource**

11. **Tracking a Project Plan**

i. **Project management is a multiple-phase undertaking:**

- **The first phase is the project initiation or the project planning phase.**
- **This phase is followed by the project-tracking phase.**

- **Finally, you wrap up a project with the-project-end phase.**

11. **Tracking a Project Plan**

ii. **Project tracking is an extremely in important activity in an organization.**

iii. **Unless a project is tracked effectively, it might never come close to termination.**

iv. **Project tracking also ensures strict adherence to project plans, requirements, and schedules.**

12. **Displaying Project Information Using Reports**

- **You create reports to display project-related information in a tabular format, which is easily comprehensible.**
- **Project reports provide a picture of the status of a project.**
- **The status of a project helps measure the progress of it.**
- **Reports can be prepared for a specific phase or for all the phases at one time.**

13. **Advantages of MS Project**

- **Easy outlining of tasks**
- **Resource management**
- **Accuracy: baseline vs. actual; various calculations**
- **Easy charting and graphics**
- **Cost management**
- **Capture historical data**

14. **Disadvantages of MS Project**

- **Illusion of control**
- **Workgroup features ok, still in-progress**
- **Scaling**
- **No estimation features**
- **Remember:**
    - **Being a MS-Project expert does not make you an expert project manager!**
    - **No more so than knowing MS-Word makes you a good writer.**

- **Summary - Application Tools**

1. **Managing Tasks in Microsoft Project**
2. **Creating Tasks**
3. **Types of Tasks**
4. **Adding Constraints to Tasks**
5. **Adding Deadlines to Tasks**
6. **Setting Dependencies between Tasks**
7. **Lag and Lead Time between Tasks**
8. **Splitting Tasks**
9. **Assigning Resources and Resource Costs to Tasks**
10. **Assigning Costs to Tasks**
12. **Displaying Project Information Using Reports**
13. **Advantages of MS Project**
14. **Disadvantages of MS Project**

XII. **Commissioning & Migration**

1. **Software implementation**

i. **Implementation Prerequisites**

ii. **Implementation Key Plans**

iii. **Implementation Activities**

iv. **Software Maintenance**

2. **Software Implementation**

- **After the software product is developed, the challenge lies in seeing the system work successfully at the user-end.**
- **Product implementation is the crucial activity of actually handing over the product to the users.**
- **Implementation begins with the preparatory activities of planning, training the user staff, and testing the system.**

- **After the preparatory activities are complete, the new system is installed and configured at the user end.**

1. **Software Implementation**

i. **Implementation Prerequisites**

- **The purpose of implementation is to organize and implement the new or updated system.**
- **Most of the time, implementation is a contract-based activity.**
- **The contract provides details of the implementation activity.**
- **Implementation procedure includes details such as the scope, method, and duration of implementation.**

1. **Software Implementation**

i. **Implementation Prerequisites (Cont.)**

- **They are finalizing the number of licenses, procedures for error logging, scheduling system changeover, and procedure for maintaining data backups.**
- **However, the primary task is to organize a team to monitor implementation.**
- **An implementation coordination committee is set up to monitor implementation.**

1. **Software Implementation**

i. **Implementation Prerequisites (Cont.)**

- **Committee identifies various requirements of implementation process, such as arranging for software licenses, project software, and other standby facilities.**
- **The committee looks into the implications of the system environment, staff allocation for implementation tasks, and internal communication.**
- **The committee also devises alternative methods facilitating system changeover.**

1. **Software Implementation**

ii. **Key Implementation Plans**

- **The various key plans performed during implementation are:**

    a) ***Implementation plan***

    b) ***Training plan***

c) ***Acceptance plan***

d) ***System support plan***

1. **Software Implementation**

ii. **Key Implementation Plans**

a) ***Implementation Plan***

- **The first task of implementation is the creation of the implementation plan.**
- **The implementation plan contains details of the methods of system transfer and the timescale for implementation.**
- **Details of hardware and software needed for implementation are also mentioned in the plan.**

1. **Software Implementation**

ii. **Key Implementation Plans**

a) ***Implementation Plan (Cont.)***

- **The plan contains details of date, location, and timescale for implementation.**
- **The plan also mentions the prerequisites of implementation, such as the existing network environment and the availability of related software.**

1. **Software Implementation**

ii. **Key Implementation Plans**

a) ***Implementation Plan (Cont.)***

- **In the implementation plan, Implementation Schedule section records details such as location and duration of implementation.**
- **The Resource List section has a list of the resources and their contact numbers.**
- **The plan has details of the server and clients in the Components of Installation section.**
- **Finally, the plan also contains data migration details.**

1. **Software Implementation**

ii. **Key Implementation Plans**

b) ***Training Plan***

- **Preparing a training plan is one of the tasks of implementation.**

- **The training plan contains details of the participants to be trained.**
- **It also mentions the focus area and the audience for the training session.**

1. **Software Implementation**

ii. **Key Implementation Plans**

b) ***Training Plan (Cont.)***

- **Training schedule containing the training venue, training name, start date and end date is a part of the training plan.**
- **The plan conveys budgetary details and the roles and responsibilities of participants as well as the faculty.**

1. **Software Implementation**

ii. **Key Implementation Plans**

b) ***Training Plan (Cont.)***

- **The plan has details of the focus areas of training, courses identified for training, and training schedules.**
- **Roles and responsibilities of trainer are also included in the training plan.**

1. **Software Implementation**

ii. **Key Implementation Plans**

c) ***Acceptance Plan***

- **The purpose of an acceptance plan is to determine the details of acceptance testing.**
- **The plan contains details such as the order of testing of functions, location, date, and duration of testing.**
- **Details of participating members, coordinators, and users are also mentioned in the plan.**

1. **Software Implementation**

ii. **Key Implementation Plans**

c) ***Acceptance Plan (Cont.)***

- **The acceptance plan lists the procedure for bug fixing, version control, and verification and validation details.**

- **Procedure for the creation of acceptance test cases and setting up software and hardware environment is also mentioned in the plan.**
- **The acceptance plan contains details of the acceptance schedule, acceptance criteria, list of resources, and a bug report form.**

1. **Software Implementation**

ii. **Key Implementation Plans**

d) ***System Support Plan***

- **The implementation team prepares the system support plan either during or before product installation.**
- **The system support plan contains details of how the system will be supported after it is implemented.**
- **Typically, the support plan has a warranty clause and usually lasts for a year.**
- **The warranty covers details of people in the support activities, details of agreement, the response time, and support details.**

1. **Software Implementation**

iii. **Implementation Activities**

- **After the prerequisites are in place, implementation activities begin in right earnest.**
- **Now, all the plans created for implementation are put into practice.**

1. **Software Implementation**

iii. **Implementation Activities**

- **Acceptance testing**
- **User and operator training**
- **Data migration**
- **Installation and configuration**
- **Product sign-off**
- **Product wind-up**
- **Post mortem**

- **Product delivery**

i. **According to surveys, changeability is one of the most important attributes of complex, multifunction software systems.**

ii. **This calls for continuous adjustments of the software to suit the changing environment.**

iii. **While changes are inevitable, it is not always possible to create products foreseeing the changes.**

iv. **Therefore, the product needs to be changed according to the changing scenario.**

v. **You might need to upgrade the software product based on new system requirements or remove redundant functionality.**

vi. **The need for a change brings software maintenance into the picture.**

vii. **Maintenance activities can either be in-house or outsourced.**

1. **Software implementation and Maintenance**

    i. **Software Implementation**

    ii. **Implementation Prerequisites**

    iii. **Implementation Key Plans**

    iv. **Implementation Activities**

2. **Software Maintenance**

    i. **Maintenance Activities**

    ii. **Maintenance Curve**

    iii. **Maintenance Process**