UNIT 1

Introduction and Software Project Planning

PART-1

Fundamentals of Software Project Management, Need Identification, Vision and Scope Document, Project Management Cycle, SPM Objectives, Management Spectrum, SPM Framework.

CONCEPT OUTLINE : PART-1

- Software project management is perhaps the most important factor in the outcome of a project.
- Project management cycle comprises of four phases :
  a. Initiation
  b. Planning
  c. Execution
  d. Closure
- Need and identification is the initial phase of project life cycle.
- The vision/scope document represents the ideas and decisions developed during envisioning phase.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 1.1. What is project ? Explain the characteristics of project.

Answer

1. In general, a project is understood to be a group of activities that are carried out to achieve a certain outcome.
2. Each project, thus, will have well defined objectives, which would measure success or failure of the project.
3. In IT industry, software projects involve the implementation of applications using new technology or improving upon the functionalities of existing applications using technology.
4. In essence, software projects use technology to develop or improve existing applications for meeting business requirements.
5. In software projects, the users will have a general idea about how the application would look like and its applicability.
6. The software projects sometimes deal with the abstract nature of the end result and get modified as time elapses.
7. Software projects should have certain activities and need to have minimum attributes for its successful completion.

Characteristics of project :

The characteristics which distinguish projects can be summarized as follows :

1. Non-routine tasks are involved.
2. Planning is required.
3. Specific objectives are to be met or a specified product is to be created.
4. The project has a pre-determined time span.
5. Work is carried out for someone other than yourself.
6. Work involves several specialism's.
7. Work is carried out in several phases.
8. The resources that are available for use on the project are constrained.
9. The project is large or complex.

Que 1.2. What is project management ?

Answer

1. Project management is the art of maximizing the probability that a project delivers its goals on time, within budget and at the required quality.
2. The art of planning for the future has always been a human trait.
3. In essence, a project can be captured on paper with a few simple elements : a start date, an end date, the tasks that have to be carried out and when they should be finished, and some idea of the resources (people, machines etc.) that will be needed during the course of the project.
4. Project management is the application of knowledge, skills, tools, and techniques to project activities to meet project requirements.
5. Project management is accomplished through the use of the processes such as : initiating, planning, executing, controlling, and closing.
6. It is important to note that many of the processes within project management are iterative in nature.

Que 1.3. Discuss various types of software projects.

Answer

The following are main application areas of software :

1. **System software :** System software is a collection of programs that are written to provide services to other programs. The examples of system software are compiler, editor, file management utilities etc.
2. **Business software :** Business information processing is the largest single software application area. The examples of business application area are payroll, inventory management, marketing, purchase etc.
3. **Embedded software :** Intelligent consumer products are becoming very popular in industrial market. Embedded software resides in read only memory and used to control products and systems for the consumer and markets. Examples of embedded software are washing machine, microwave oven, air conditioner, sleeping timer in television remote, fuel control keypad etc.

4. **Engineering and scientific software :** Engineering and scientific software have been categorized by "number crunching" algorithms. There are number of software which are used for scientific use. Examples of such types of software are Computer Aided Design (CAD), Computer Aided Manufacturing (CAM), simulation etc.
5. **Personal computer software :** Personal computer software may be said as small business applications which are used by single user. The examples of this type of software are word processor, spreadsheet, multimedia, entertainment, database management, small business financial application etc.
6. **Real time software :**
   a. Software that monitors/analyzes/controls real world events as they occur is known as real time software.
   b. Elements of real time software includes :
      i. data gathering components that collects and formats information from an external
      ii. environment, an analysis component that transform information as required by the application,
      iii. a control/output component that respond to the external environment, and
      iv. a monitoring component that coordinates all other components so that real time response (typically all ranging from 1 millisecond to 1 second) can be maintained.
7. **Artificial intelligence software :** Artificial intelligence software makes use of non-numerical algorithms to solve the complex problem that cannot solve by straight forward analysis. The examples of A.I. software are expert system, pattern recognition (image or voice), artificial neural network, theorem proving, game playing etc.

Que 1.4. How software projects are different from other types of project ?

Answer

Many of the techniques of general project management are applicable to software project management, but Fred Brooks pointed out that the products of software projects have certain characteristics which make them different.

1. **Invisibility :**
   a. When a physical artefact such as a bridge or road is being constructed the progress being made can actually be seen. With software, progress is not immediately visible.

b. One way of perceiving software project management is the process of making visible that which is invisible.

2. **Complexity :** Per dollar, pound or euro spent software products contain more complexity than other engineered artefacts.

3. **Conformity :**
   a. The traditional engineer is usually working with physical systems and physical materials like cement and steel.
   b. These physical systems can have some complexity, but are governed by physical laws that are consistent.
   c. Software developers have to conform to the requirements of human clients. It is not just that individuals can be inconsistent.
   d. Organizations, because of lapses in collective memory, in internal communication or ineffective decision-making, can exhibit remarkable organizational stupidity that developers have to cater for.

4. **Flexibility :**
   a. The ease with which software can be changed is usually seen as one of its strengths.
   b. However, this means that where the software system interfaces with a physical or organizational system, it is expected that, where necessary, the software will change to accommodate the other components rather than vice versa.
   c. This means the software systems are likely to be subjected to a high degree of change.

Que 1.5. **What are the activities or knowledge area involved in management/project management ?**

Answer

The open university software project management module suggested that management involves the following activities/knowledge area :

1. **Planning :** Deciding what is to be done.
2. **Organizing :** Making arrangements.
3. **Staffing :** Selecting the right people for the job, etc.
4. **Directing :** Giving instructions.
5. **Monitoring :** Checking on progress.
6. **Controlling :** Taking action to remedy hold-ups.
7. **Innovating :** Coming up with new solutions.
8. **Representing :** Liaising with users, etc.

Que 1.6. **Write a short note on softwar project management.**

Answer

1. The essence of project management is to make sure that all the agreed objectives are accepted, executed, and shared among all the stakeholders.
2. An effective project management will ensure that all the stakeholders are identified at the initial stage itself and then "buy in" the objectives.
3. The first step in project management would be to start planning.
4. A detailed planning will help identify minute details and plan for these details at that granular level.
5. The success of project management is entrusted to project manager (PM).
6. Some of the elements that contribute to the success of project management are as follows :
   a. clearly stated and understood objectives,
   b. role and responsibilities of each team member,
   c. communication protocol,
   d. escalation process,
   e. clearly defined project organization structure, which will show process of information flow,
   f. periodic status report review, and
   g. changed management process.
7. Predictability is an important aspect of project management; this ensures that customers as well as team members can predict the quality and the schedule of the deliverables.

Que 1.7. **What are the various activities covered by software project management ? Also, explain the life cycle of a software project.**

Answer

Various activities covered by software project management are :

1. **The feasibility study :**
   a. This is an investigation into whether a prospective project is worth starting.
   b. Information is gathered about the requirements of the proposed application.
   c. The probable developmental and operational costs, along with the value of the benefits of the new system, are estimated.
   d. With a large system, the feasibility study could be treated as a project in its own right and have its own planning sub-phase.

2. **Planning :**
   a. If the feasibility study produces results which indicate that the prospective project appears viable, then planning of the project can take place.
   b. However, for a large project, we would not do all our detailed planning right at the beginning.
   c. We would formulate an outline plan for the whole project and a detailed one for the first stage.
3. **Project execution :** The project can now be executed. The execution of a project often contains design and implementation sub-phases.

Stages involved in the life cycle of a software project are :

1. **Requirements analysis :** The user's requirements from the system are analyzed during the implementation of the project.
2. **Specification :** Detailed documentation of what the proposed system is to do.
3. **Design :** A design has to be drawn up which meets the specification. This design will be in two stages. One will be the external or user design concerned with the external appearance of the application. The other produces the physical design.
4. **Coding :** This may refer to writing code in a procedural language such as C or Ada, or could refer to the use of an application-builder such as Microsoft Access.
5. **Verification and validation :** Whether software is developed specially for the current application or not, careful testing will be needed to check that the proposed system meets its requirements.
6. **Implementation/Installation :** Some system development practitioners refer to the whole of the project after design as 'implementation' (that is, the implementation of the design) while others insist that the term refers to the installation of the system after the software has been developed.
7. **Maintenance and support :** Once the system has been implemented there is a continuing need for the correction of any errors that may have crept into the system and for extensions and improvements to the system.

Que 1.8. **Explain various problems with software projects in detail.** **UPTU 2014-15, Marks 05**

Answer

A survey of managers published by Thayer, Pyster and Wood identified the following commonly experienced problems :

1. Poor estimates and plans.
2. Lack of quality standards and measures.

3. Lack of guidance about making organizational decisions.
4. Lack of techniques to make progress visible.
5. Poor role definition who does what ?
6. Incorrect success criteria.

The above list looks at the project from the manager's point of view. What about the staff who make up the members of the project team ? Below is a list of the problems identified by a number of students on a computing and information systems course who had just completed a year's industrial placement :

1. Inadequate specification of work.
2. Management ignorance of ICT.
3. Lack of knowledge of application area.
4. Lack of standards.
5. Lack of up-to-date documentation.
6. Preceding activities not completed on time - including late delivery of equipment.
7. Lack of communication leading to duplication of work.
8. Lack of communication between users and technicians.
9. Lack of commitment especially when a project is tied to one person who then moves.
10. Narrow scope of technical expertise.
11. Changing statutory requirements.
12. Changing software environment.
13. Deadline pressure.
14. Lack of quality control.
15. Remote management.
16. Lack of training.

Que 1.9. **Explain need and identification of SPM. Also, write a vision and scope document according to IEEE standard in respect of SPM.** **UPTU 2013-14, Marks 10**

OR

What are the various factors which describe the scope of software ? Explain. **UPTU 2012-13, Marks 05**

Answer

Need and identification of SPM :

1. Need and identification is the initial phase of the project life cycle.

2. It starts with the recognition of a need, problem, or opportunity and ends with the issuance of a Request For Proposal (RFP).
3. The customer identifies a need, a problem, or an opportunity for a better way of doing something and therefore sees some benefit to undertaking a project that will result in an improvement or advantage over the existing condition.
4. For example, suppose a company's management recognizes that the time the company takes to issue invoices and collect payments from its customers is too long.
5. Furthermore, the fact that the company payment records are not up to date has caused second invoices to be sent to customers who have already paid, thus upsetting some good customers.
6. Management recognizes several problems and opportunities for improvement, so it develops an RFP asking contractors to submit proposals for implementing an automated billing and collection system.
7. In a different scenario, the company's management might request a proposal from an in-house individual or project team rather than from an external contractor.

The outline of vision and scope documents is as follows :

1. Problem statement
   a. Project background
   b. Stakeholders
   c. Users
   d. Risks
   e. Assumptions
2. Vision of the solution
   a. Vision statement
   b. List of features
   c. Scope of phased release (optional)
   d. Features that will not be developed

Que 1.10. **Describe project management life cycle.**

OR

Describe the terms project initiation, planning, execution and closure.

Answer

The project management life cycle comprises of four phases :

1. Initiation involves starting up the project, by documenting a business case, feasibility study, terms of reference, appointing the team and setting up a project office.

2. Planning involves setting out the roadmap for the project by creating the following plans : project plan, resource plan, financial plan, quality plan, acceptance plan and communications plan.
3. Execution involves building the deliverables and controlling the project delivery, scope, costs, quality, risks and issues.
4. Closure involves winding-down the project by releasing staff, handing over deliverables to the customer and completing a post implementation review.

A more detailed description of the MPMM project management methodology and life cycle is given below :

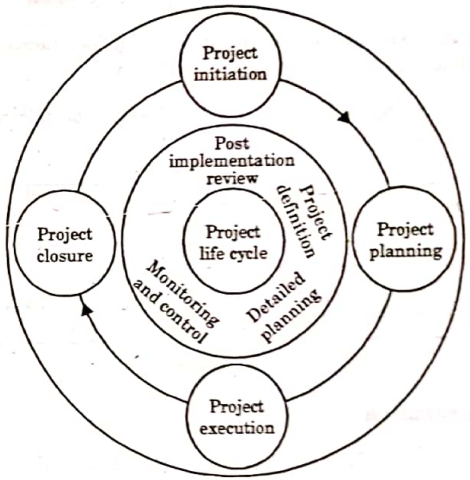


Fig. 1.10.1.

1. **Project initiation :**
   a. Project initiation is the first phase in the project life cycle and essentially involves starting up the project.
   b. Initiate a project by defining its purpose and scope, the justification for initiating it and the solution to be implemented.
   c. We will also need to recruit a suitably skilled project team, set up a project office and perform an end of phase review.
   d. The project initiation phase involves the following six key steps which are shown in Fig. 1.10.2.

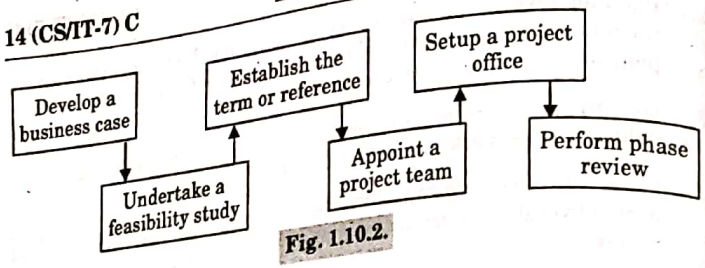


Fig. 1.10.2.

2. Project planning :

a. After defining the project and appointing the project team, we are ready to enter the detailed project planning phase.

b. This involves creating a suite of planning documents to help guide the team throughout the project delivery.

c. The planning phase involves completing the following ten key steps as shown in Fig. 1.10.3.

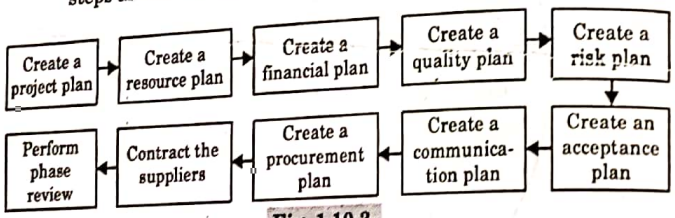


Fig. 1.10.3.

3. Project execution :

a. With a clear definition of the project and a suite of detailed project plans, we are now ready to enter the execution phase of the project.

b. This is the phase in which the deliverables are physically built and presented to the customer for acceptance.

c. While each deliverable is being constructed, a suite of management processes are undertaken to monitor and control the deliverables being output by the project.

d. These processes include managing time, cost, quality, change, risks, issues, suppliers, customers and communication.

e. Once all the deliverables have been produced and the customer has accepted the final solution, the project is ready for closure.

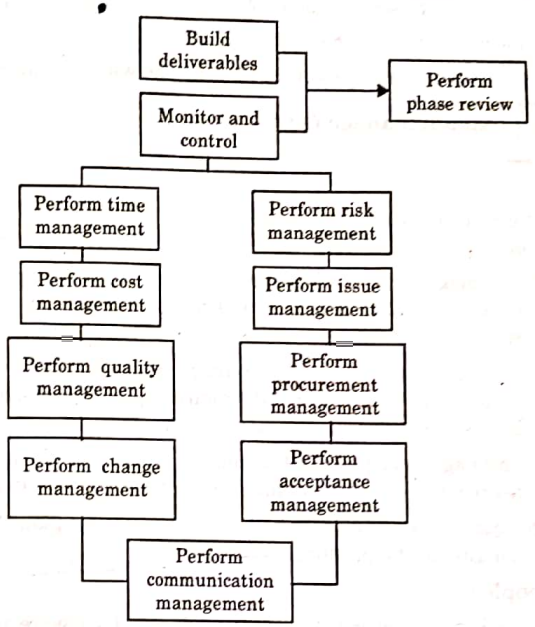


Fig. 1.10.4.

4. Project closure :

a. Project closure involves releasing the final deliverables to the customer, handing over project documentation to the business, terminating supplier contracts, releasing project resources and communicating project closure to all stakeholders.

b. The last remaining step is to undertake a post implementation review to identify the level of project success and note any lessons learned for future projects.

Que 1.11. What are the objectives of SPM ?

Answer

Basic objectives of SPM are as follows :

1. Define the project.
2. Reduce it to a set of manageable tasks.
3. Obtain appropriate and necessary resources.
4. Build a team or teams to perform the project work.
5. Plan the work and allocate the resources to the tasks.
6. Monitor and control the work.

7. Report progress to senior management and/or the project sponsor.
8. Close down the project when completed.
9. Review it to ensure the lessons are learnt and widely understood.

Que 1.12. Explain management spectrum.

Answer

i. Effective software project management focuses on the four P's : people, product, process, and project. The order is not arbitrary.

ii. The manager who forgets that software engineering work is an intensely human endeavour will never have success in project management.

iii. A manager who fails to encourage comprehensive customer communication early in the evolution of a project risks, building an elegant solution for the wrong problem.

iv The manager who pays little attention to the process runs the risk of inserting competent technical methods and tools into a vacuum.

v. The manager who embarks without a solid project plan jeopardizes the success of the product.

1. **The people :**
   a. The cultivation of motivated, highly skilled software people has been discussed since the 1960s.
   b. In fact, the people factor is so important that the Software Engineering Institute has developed a People Management Capability Maturity Model (PM-CMM), "to enhance the readiness of software organizations to undertake increasingly complex applications by helping to attract, grow, motivate, deploy, and retain the talent needed to improve their software development capability".
   c. The people management maturity model defines the following key practice areas for software people : recruiting, selection, performance management, training, compensation, career development, organization and work design, and team/culture development.
   d. Organizations that achieve high levels of maturity in the people management area have a higher likelihood of implementing effective software engineering practices.
2. **The product :**
   a. Before a project can be planned, product objectives and scope should be established, alternative solutions should be considered and technical and management constraints should be identified.
   b. Without this information, it is impossible to define reasonable (and accurate) estimates of the cost, an effective assessment of risk, a realistic breakdown of project tasks, or a manageable project schedule that provides a meaningful indication of progress.
   c. The software developer and customer must meet to define product objectives and scope.
   d. In many cases, this activity begins as part of the system engineering or business process engineering and continues as the first step in software requirements analysis.
   e. Objectives identify the overall goals for the product (from the customer's point of view) without considering how these goals will be achieved.
   f. Scope identifies the primary data, functions and behaviours that characterize the product, and more important, attempts to bound these characteristics in a quantitative manner.
   g. Once the product objectives and scope are understood, alternative solutions are considered.
   h. The alternatives enable managers and practitioners to select a best approach, given the constraints imposed by delivery deadlines, budgetary restrictions, personnel availability, technical interfaces. and myriad other factors.
3. **The process :**
   a. A software process provides the framework from which a comprehensive plan for software development can be established.
   b. A small number of framework activities are applicable to all software projects, regardless of their size or complexity.
   c. A number of different task sets–tasks, milestones, work products, and quality assurance points enable the framework activities to be adapted to the characteristics of the software project and the requirements of the project team.
   d. Finally, umbrella activities – such as software quality assurance, software configuration management, and measurement – overlay the process model.
   e. Umbrella activities are independent of any one framework activity and occur throughout the process.
4. **The project :**
   a. We conduct planned and controlled software projects for one primary reason.
   b. It is the only known way to manage complexity. In 1998, industry data indicated that 26 % of software projects failed outright and 46 % experienced cost and schedule overruns.
   c. Although the success rate for software projects has improved somewhat, our project failure rate remains higher than it should be.

d. In order to avoid project failure, a software project manager and the software engineers who build the product must avoid a set of common warning signs, understand the critical success factors that lead to good project management, and develop a common sense approach for planning, monitoring and controlling the project.

Que 1.13. What is software project management framework ?

Answer

1. A project is a synchronized event where there is perfect harmony and understanding between the participants.
2. The participants are equipped with the essential skills of planning, cooperating, helping, and communicating.
3. However, the most important activity here is to arrange the movement of the participants.
4. The responsibility lies with the project manager to synchronize the activities of the project to result in a perfect presentation.
5. Although each, project manager has a unique style of functioning, there are some fundamental approaches that guide a project manager.
6. These approaches are traditional project management concepts and software engineering concepts.
7. To understand software projects and their dynamics, we must be aware of the environment in which a software project is executed.
8. This further requires an understanding of the larger framework of software project management.

Que 1.14. List the various responsibilities of software project manager.

Answer

1. *Ownership of customer* relationship and business.
2. Analysis of project health (productivity and profitability) and report to business manager.
3. Managing the onsite team.
4. Maintain the consolidated delivery and billing plan.
5. Identif. ation and planning of new business with the customer.
6. Review of estimates and proposals.
7. Provide manpower requirements.
8. Maintain the project management plan.

PART-2

Software Project Planning, Planning Objectives, Project Plan, Types of Project Plan, Structure of Software Project Management Plan.

CONCEPT OUTLINE : PART-2

- Project planning is an aspect of project management which comprises of various processes.
- Types of project plans :
  a. Quality plan
  b. Validation plan
  c. Configuration management plan
  d. Maintenance plan
  e. Staff development plan

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 1.15. Write short note on software project planning.

Answer

1. Project planning is an aspect of project management, which comprises of various processes.
2. The aim of these processes is to ensure that various project tasks are well coordinated and they meet the various project objectives including timely completion of the project.
3. The project plan reflects the current status of all project activities and is used to monitor and control the project.
4. Project planning is an ongoing effort throughout the project life cycle.
5. The project planning tasks ensure that various elements of the project are coordinated and therefore guide the project execution.
6. Project planning helps in :
   a. Facilitating communication.
   b. Monitoring/measuring the project progress.
   c. Provides overall documentation of assumptions/planning decisions.
7. The project planning phases can be broadly classified as follows :
   a. Development of the project plan.

b. Execution of the project plan.
c. Change control and corrective actions.

Que 1.16. Write down the outline of stepwise planning activities.

Answer

The stepwise planning is discussed as follows :

0. Select project
1. **Identify project scope and objectives :**
   a. Identify objectives and measures of effectiveness in meeting them.
   b. Establish a project authority.
   c. Identify stakeholders.
   d. Modify objectives in the light of stakeholder analysis.
   e. Establish methods of communications with all parties.
2. **Identify project infrastructure :**
   a. Establish relationship between project and strategic planning.
   b. Identify installation standards and procedures.
   c. Identify project team organization.
3. **Analyze project characteristics :**
   a. Distinguish the project as either objective or product driven.
   b. Analyze other project characteristics.
   c. Identify high-level project risks.
   d. Take into account user requirements concerning implementation.
   e. Select general life cycle approach.
   f. Review overall resource estimates.
4. **Identify project products and activities :**
   a. Identify and describe project products (including quality criteria).
   b. Document generic product flows.
   c. Recognize product instances.
   d. Produce ideal activity network.
   e. Modify idea to take into account need for stages and checkpoints.
5. **Estimate effort for each activity :**
   a. Carry out bottom-up estimates.
   b. Revise plan to create controllable activities.
6. **Identify activity risks :**
   a. Identify and quantify activity-based risks.
   b. Plan risk reduction and contingency measures where appropriate.

   c. Adjust plans and estimates to take account of risks.
7. **Allocate resources :**
   a. Identify and allocate resources.
   b. Revise plans and estimates to account for resource constraints.
8. **Review/publish plan :**
   a. Review quality aspects of project plan.
   b. Document plans and obtain agreements.
9. **Execute plan/lower levels of planning :**
   a. This may require the reiteration of the planning process at a lower level.

Que 1.17. How do we identify the planning objectives ?

Answer

1. After all the stakeholders are identified, it is critical to identify what each stakeholder expects to gain from the project.
2. For instance, for an end user it might be that they are expecting a very user friendly and robust software with a number of features, while for a maintainer it is the quality of the documentation and the modifiability of the system that are important.
3. A manager would not want any overruns to the schedule, while the person who approves the budget might expect a low budget project.
4. By identifying each of the stakeholders "win" condition, the project's objectives are clear from the start.
5. In the negotiation with the customer, these objectives should be reinforced and documented.
6. Steps to identify the objectives are as follows :
   a. Identify and allocate resources.
   a. Understand how people will benefit from the project.
   b. Prioritize objectives for the project.
   c. Establish reasonable expectations on the parts of all the stakeholders.
   d. Transform these objectives into project activities.
   e. Identify and manage the risks.
   f. Keep people involved. Keep senior management and the customer aware of the status of the project at all times.

Que 1.18. When does the software planning activity start and end in software life cycle ? List some important activities that a software project manager performs during software project planning ?

UPTU 2014-15, Marks 05

OR

Write short note on software project planning and its scope.

UPTU 2013-14, Marks 05

Answer

Software project planning : Refer Q. 1.15, Page 19C, Unit-1.

i. Project planning spans across the various aspects of the project. Generally project planning is considered to be a process of estimating, scheduling and assigning the projects resources in order to deliver an end product of suitable quality.

ii. However, it is much more as it can assume a very strategic role, which can determine the success of the project.

iii. A project plan is one of the crucial steps in project planning. Typically, project planning can include the following types of project planning :

1. **Project scope definition and scope planning :**
   a. In this step, we document the project work that would help us achieve the project goal.
   b. We document the assumptions, constraints, user expectations, business requirements, technical requirements, project deliverables, project objectives and everything that defines the final product requirements.
   c. This is the foundation for a successful project completion.
2. **Quality planning :**
   a. The relevant quality standards are determined for the project.
   b. This is an important aspect of project planning.
   c. Based on the inputs captured in the previous steps such as the project scope, requirements, deliverables, etc., various factors influencing the quality of the final product are determined.
   d. The processes required to deliver the product as promised and as per the standards are defined.
3. **Project activity definition and activity sequencing :**
   a. In this step, we define all the specific activities that must be performed to deliver the product by producing the various product deliverables.
   b. The project activity sequencing identifies the interdependence of all the activities defined.
4. **Time, effort and resource estimation :**
   a. Once the scope, activities and activity interdependence is clearly defined and documented, the next crucial step is to determine the effort required to complete each of the activities.

   b. The effort can be calculated using one of the many techniques available such as function points, lines of code, complexity of code, benchmarks, etc.
   c. This step clearly estimates and documents the time, effort and resource required for each activity.
5. **Risk factors identification :**
   a. "Expecting the unexpected and facing it".
   b. It is important to identify and document the risk factors associated with the project based on the assumptions, constraints, user expectations, specific circumstances, etc.
6. **Schedule development :**
   a. The time schedule for the project can be arrived on the basis of the activities, interdependence and effort required for each of them.
   b. The schedule may influence the cost estimates, the cost benefit analysis and so on.
   c. Project scheduling is one of the most important task of project planning and also the most difficult task.
   d. In very large projects, it is possible that several teams work on developing the project.
   e. They may work on it in parallel. However, their work may be interdependent.
   f. Popular tools can be used for creating and reporting the schedules such as Gantt charts.
7. **Cost estimation and budgeting :**
   a. Based on the information collected in all the previous steps, it is possible to estimate the cost involved in executing and implementing the project.
   b. A cost benefit analysis can be arrived at for the project.
   c. Based on the cost estimates, budget allocation is done for the project.
8. **Organizational and resource planning :**
   a. Based on the activities identified, schedule and budget allocation resource types and resources are identified.
   b. One of the primary goals of resource planning is to ensure that the project is run efficiently.
   c. This can only be achieved by keeping all the project resources fully utilized as possible.
   d. The success depends on the accuracy in predicting the resource demands that will be placed on the project.
   e. Resource planning is an iterative process and necessary to optimize the use of resources throughout the project life cycle thus making the project execution more efficient.

f. There are various types of resources : equipment, personnel, facilities, money, etc.

9. **Risk management planning :**
   a. Risk management is a process of identifying, analyzing and responding to a risk.
   b. Based on the risk factors, identified risk resolution plan is created.
   c. The plan analyses each of the risk factors and their impact on the project. The possible responses for each of them can be planned.
   d. Throughout the lifetime of the project, these risk factors are monitored and acted upon as necessary.

10. **Project plan development and execution :**
   a. Project plan development uses the inputs gathered from all the other planning processes such as scope definition, activity identification, activity sequencing, quality management planning, etc.
   b. A detailed work breakdown structure comprising of all the activities identified is used.
   c. The tasks are scheduled based on the inputs captured in the steps previously described.
   d. The project plan documents all the assumptions, activities, schedule, timelines and drives the project.

11. **Performance reporting :**
   a. The progress of each of the tasks/activities described in the project plan is monitored.
   b. The progress is compared with the schedule and timelines documented in the project plan.
   c. Various techniques are used to measure and report the project performance such as EVM (Earned Value Management).
   d. A wide variety of tools can be used to report the performance of the project such as PERT charts, Gantt charts, Logical bar charts, Histograms, Pie charts, etc.

12. **Planning change management :**
   a. Analysis of project performance can necessitate that certain aspects of the project be changed.
   b. The Requests for Changes need to be analyzed carefully and its impact on the project should be studied.
   c. Considering all these aspects the project plan may be modified to accommodate this request for change.

Activities performed during software project planning are : Refer Q. 1.14, Page 18C, Unit-1.

Que 1.19. What are the various dimensions of software project planning ? Discuss the impact of each section of a software project planning on the successful completion of the software project.

UPTU 2012-13, Marks 10

Answer

The mnemonic SMART is sometimes used to describe well-defined objectives of software project planning such as :

1. **Specific :**
   a. Effective objectives are concrete and well defined. Vague aspirations such as 'to improve customer relations' are unsatisfactory.
   b. Objectives should be defined in such a way that it is obvious to all whether the project has been successful or not.
2. **Measurable :**
   a. Ideally there should be measures of effectiveness which tell us how successful the project has been.
   b. For example 'to reduce customer complaints' would be more satisfactory as an objective than 'to improve customer relations'.
   c. The measure can, in some cases, be an answer to simple yes/no question, for example 'Did we install the new software by 1 June ?'
3. **Achievable :** It must be within the power of the individual or group to achieve the objective.
4. **Relevant :** The objective must be relevant to the true purpose of the project.
5. **Time constrained :** There should be a defined point of time by which the objective should have been achieved.

Impact of each section of a software project planning on the successful completion of the software project : Refer Q. 1.18, Page 21C, Unit-1.

Que 1.20. What do you mean by project plan ? Also, discuss types of project plans.

UPTU 2014-15, Marks 05

Answer

Project plan : Refer Q. 1.15, Page 19C, Unit-1.

There are various types of project plan :

1. **Quality plan :** It describes the quality procedures and standards that will be used in project.
2. **Validation plan :** It describes the approach, resources and schedule used for system validation.
3. **Configuration management plan :** It describes the configuration management procedures and structures to be used.

4. **Maintenance plan :** It predicts the maintenance requirements of the system, maintenance costs and effort required.
5. **Staff development plan :** It describes how the skills and experience of the project team members will be developed.

Que 1.21. Write a short note on pragmatic planning.

UPTU 2012-13, Marks 05

Answer

a. In the pragmatic planning, the problem is solved in a practical and sensible way rather than by having ideas or theories.

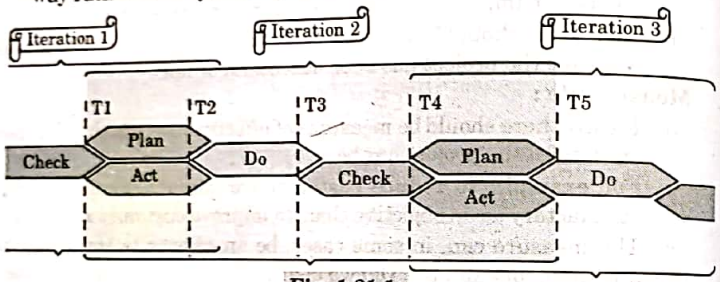


Fig. 1.21.1.

b. The set of activities are done in a particular period of time. The set of activities are called the iterations.

c. These iterations are overlapped in time in order to optimize people allocation, use of resources and guarantee the feedback from previous iterations.

d. Each iteration is performed in a fixed period of time. There are four set of activities : plan, do, check and act.

e. In the planning process there are following five events :

1. **T1 :** It represents the beginning of the iteration. In this, we do the initial planning. The scope of the planning covers only the period of the current iteration. All members of the team participate in the planning.
2. **T2 :** In this iteration, planned activities are executed. All members of the team must have something to do within the scope of the iteration. The activities for future iterations should not be planned in the current iteration. All sort of output may be produced such as documents, code, reports, meeting minutes, etc.
3. **T3 :** In this iteration, everything that is produced is checked. Documents are reviewed, codes are tested, user interfaces and integrations with other systems are tested, etc. All found issues must be registered to be solved in due time.

4. **T4 :**
   i. All issues found during the check phase are solved and all planned deliverables are released. Everybody should deliver something.
   ii. In case time is not enough to solve some issues, they are included in the planning of the next iteration with the highest priority.
   iii. The planning of the next iteration also starts at this point, taking advantage of the experience from the previous iteration.
5. **T5 :** Once everything is released, the iteration finishes. T2 of the next iteration immediately starts because most of people and resources are already available. T1 to T5 repeats several times, in a fixed period of time, until the end of the project.

Que 1.22. Discuss the structure of a Software Project Management Plan (SPMP) in detail.

UPTU 2014-15, Marks 05

Answer

The structure of software project plan is as follows :

1. **Overview**
   a. Project purpose, objectives, and success criteria
   b. Project deliverables
   c. Assumptions, dependencies, and constraints
   d. References
   e. Definitions and acronyms
   f. Evolution of the plan
2. **Project organization**
   a. External interfaces
   b. Internal structure
   c. Roles and responsibilities
3. **Managerial process plans**
   a. Start-up plans
      i. Estimation plan
      ii. Staffing plan
      iii. Staff training plan
      iv. Resource acquisition plan
      v. Project commitments
   b. Work plan
   c. Control plan

i. Data control plan
ii. Requirements control plan
iii. Schedule control plan
iv. Budget control plan
v. Communication, tracking, and reporting plan
vi. Metrics collection plan
vii. Risk management plan
viii. Issue resolution plan
ix. Project close-out plan

4. **Technical process plans**
   a. Process model
   b. Methods, tools, and techniques
   c. Configuration management plan
   d. Quality assurance plan
   e. Documentation plan
   f. Process improvement plan

PART-3

Software Project Estimation, Estimation methods, Estimation models, Decision Process.

CONCEPT OUTLINE : PART-3

- According to Kelkar, estimation is defined as process of reliably predicting the various parameters associated with making a project, *i.e.*, size, effort, cost, time and quality.
- Various software estimating techniques are :
  a. Parkinson's law
  b. Function point
  c. Top-down approach
  d. Bottom-up approach
  e. Estimation by analogy
- According to Boehm, software cost estimation should be done through three stages :
  a. Basic COCOMO model
  b. Intermediate COCOMO model
  c. Complete COCOMO model

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 1.23. **What do you mean by software project estimation ?**

UPTU 2014-15, Marks 05

OR

Explain software project estimation. Why software project estimation is needed ?

Answer

1. Delivery of an error free product *i.e.*, software within budget and schedule is the topmost priority of any organization.
2. This in turn calls for reasonably good estimates of effort, time and cost and of course quality which affects the schedule and effort.
3. It is seen that in order to produce good quality software, extra effort and longer schedule is required.
4. While computing the cost of project, one of the major components is the cost of effort applied by skilled software professionals in terms of person hours, person days or person-months.
5. Other factors which contribute to total cost are cost of hardware and software, traveling, cost of tools, training etc.
6. Hence, key to accurate estimation of cost is accurate estimation for the manpower that in turn also decides the team size.
7. Estimation of size of the software to be developed is also very important as it forms the basis for effort and schedule estimation.
8. A number of estimation methods to measure size in terms of number of lines of code, function points, object points etc. are proposed.
9. Software effort estimation is important because of following reasons :
   a. Organizations have proper control over project and they can plan systematically.
   b. There is a clear understanding of the product.
   c. Estimation also determines the project feasibility in terms of budget and time constraints.
   d. It helps in identification of resources to be used during the project.
   e. Estimation also helps management in taking decision for current as well as future projects.
   f. Estimation helps in quantifying the impact of risks and guides the project manager to take suitable decision.

Que 1.24. Is software project estimation essential for all kinds of software project ? Discuss various software project estimation techniques, together with their merits and demerits.

Answer

Need of software project estimation : Refer Q. 1.23, Page 29C, Unit-1.

Boehm has discussed seven techniques of software effort estimation. They are described briefly as follows :

1. **Algorithmic cost modeling :**
   a. This approach is based on historical cost information.
   b. In this, a model is developed which relates the project cost to some software metric which is usually the size in this case.
   c. The most common size metric is the number of Lines of Code (LOC) of the final product but that is not easy to predict in the beginning.
   d. Code size based estimation is also uncertain because number of factors contributes for computing the final number of LOC, for example, hardware used, software choice, type of DBMS used etc.
2. **Expert judgment :**
   a. In this approach, experience and judgment of one or more experts on that particular application domain is used for the project estimation by extrapolating expert's experience.
   b. Each of the expert estimates the project cost and final cost estimation is decided by consensus.
   c. The problem with this technique is the limited expertise of the experts and hence should be used when other options are not available.
3. **Estimation by analogy :**
   a. This technique is used when the organization in the past have completed the projects of the similar nature.
   b. For example, an organization for last few years has developed expertise in the banking domain, which can be used in future projects.
   c. In this technique, comparisons are made with the past projects and estimates are proposed.
4. **Parkinson's law :**
   a. According to Parkinson's law, work expands to fill the available time and budget.
   b. It means that the Parkinson's approach to cost estimation assumes that the time for completion and available resources are known prior to starting the project.

   c. As a result, in some cases it can also result into over estimates.
5. **Pricing to win approach :**
   a. In the pricing to win approach, cost of the project is proportional to the budget of the customer.
   b. As a result, the estimated effort does not depend upon the software functionality but also on the customer's capability to spend.
   c. This approach, therefore often results into poor quality product, schedule overrun and over budgeting.
6. **Top-down estimation :**
   a. In this approach, the whole project is decomposed into number of phases/tasks and estimation for each phase is done using different approach.
   b. In top-down estimation, whole functionality of the product is decomposed into sub-functions and cost estimation is done by using these logical sub-functions.
   c. It is therefore suitable for using early in software life cycle.
7. **Bottom-up estimation :**
   a. In this approach, instead of logical functions, components implementing these functions are used for cost estimation.
   b. Each component cost is estimated and then added to give the final cost estimate.
   c. Bottom-up estimation techniques are appropriate at detailed stages of project planning.
   d. In practice, both top-down as well as bottom-up approaches are used.

Que 1.25. Explain Bottom-up estimating.

Answer

i. Estimating methods can be generally divided into bottom-up and top-down approaches.

ii. With the bottom-up approach, the estimator breaks the project into its component tasks and then estimates how much effort will be required to carry out each task.

iii. With a large project, the process of breaking down into tasks would be a repetitive one : each task would be analyzed into its component subtasks and these in turn would be further analyzed.

iv. This is repeated until we get to components that can be executed by a single person in about a week or two.

v. The reader might wonder why this is not called a top-down approach : after all we are starting from the top and working down. Although this

top-down analysis is an essential precursor to bottom-up estimating, it is really a separate one -that of producing a Work Breakdown Structure (WBS).

vi. The bottom-up part comes in adding up the calculated effort for each activity to get an overall estimate.

vii. The bottom-up approach is most appropriate at the later, more detailed, stages of project planning. If this method is used early on in the project cycle then the estimator will have to make some assumptions about the characteristics of the final system, for example, the number and size of software modules.

viii. These will be working assumptions that imply no commitment when it comes to the actual design of the system.

ix. Where a project is completely novel or there is no historical data available, the estimator would be forced to use the bottom-up approach.

Que 1.26. Explain the top-down approach and parametric models.

Answer

i. The top-down approach is normally associated with parametric (or algorithmic) models. These may be explained using the analogy of estimating the cost of rebuilding a house.

ii. This would be of practical concern to a house-owner who needs sufficient insurance cover to allow for rebuilding their property if it were destroyed. Unless the house-owner happens to be in the building trade it is unlikely that he or she would be able to work out how many bricklayer-hours, how many carpenter hours, electrician-hours and so on would be required.

iii. Insurance companies, however, produce convenient tables where the house-owner can find an estimate of rebuilding costs based on such parameters as the number of storey's and the floor space that a house has.

iv. This is a simple parametric model.

v. The effort needed to implement a project will be related mainly to variables associated with characteristics of the final system.

vi. The form of the parametric model will normally be one or more formulae in the form :

$$\text{effort} = (\text{system size}) \times (\text{productivity rate})$$

vii. For example, system size might be in the form "thousands of lines of code" (KLOC) and have the specific value of 3 KLOC while the productivity rate 40 days per KLOC.

viii. The values to be used will often be matters of subjective judgment. A model to forecast software development effort therefore has two key components.

ix. The first is a method of assessing the size of the software development task to be undertaken. The second assesses the rate of work at which the task can be done.

x. For example, Amanda at IOE might estimate that the first software module to be constructed is 2 KLOC.

xi. She might then judge that if undertook the development of the code, with her expertise she could work at a rate of 40 days per KLOC per day and complete the work in 2×40 days, that is, 80 days, while Ken, who is less experienced would need 55 days per KLOC and take 2×55 that is, 110 days to complete the task.

xii. Some parametric models, such as that implied by function points, are focused on system or task size, while others, such as COCOMO, are more concerned with productivity factors.

xiii. Having calculated the overall effort required, the problem is then to allocate proportions of that effort to the various activities within that project.

Que 1.27. Why software project estimation is needed ? What are various estimation approaches and when these are used ? Explain any one estimation method with example.

UPTU 2012-13, Marks 10

OR

What do you understand by function point ? Discuss limitation and benefits of function point.

Answer

Need of software project estimation : Refer Q. 1.23, Page 29C, Unit-1.
Various estimation approaches : Refer Q. 1.24, Page 30C, Unit-1.
One estimation method with example :

1. Function point metric was proposed by Allan Albrecht.
2. This metric overcomes many of the shortcomings of the LOC metric.
3. One of the important advantages of using the function point metric is that it can be used to easily estimate the size of a software product directly from the problem specification.
4. This is contrast to the LOC metric, where the size can be accurately determined only after the product has been fully developed.
5. The conceptual idea behind the function point metric is that the size of a software product is directly dependent on the number of different functions or features it supports.
6. A software product supporting many features would certainly be of larger size than a product with less number of features. Each function when invoked reads some input data and transforms it to the corresponding output data.

7. Function point measures functionality from the user's point of view, that is, on the basis of what the user requests and receives in return from the system.

8. The principle of Albrecht's function point analysis (FPA) is that a system is decomposed into functional units.

a. Inputs	: information entering the system.
b. Outputs	: information leaving the system.
c. Enquiries	: requests for instant access to information.
d. Internal logical files	: information held within the system.
e. External interface files	: information held by other systems, that is, used by the system being analyzed.

9. The five functional units are divided in two categories :

a. **Data function type :**

i. **Internal Logical Files (ILF) :** A user identifiable group of logically related data or control information maintained within the system.

ii. **External Interface Files (EIF) :** A user identifiable group of logically related data or control information referenced by the system, but maintained within another system. This means that EIF counted for one system, may be ILF in another system.

b. **Transactional function types :**

i. **External Input (EI) :** An EI processes data or control information that comes from outside the system. The EI is an elementary process, which is the smallest unit of activity, that is, meaningful to the end user in the business.

ii. **External Output (EO) :** An EO is an elementary process that generates data or control information to be sent outside the system.

iii. **External Inquiry (EQ) :** An EQ is an elementary process, that is, made up of an input output combination that results in data retrieval.

Advantages of function point :

1. It is not restricted to code.
2. Language independent.
3. More accurate than estimated LOC.

Drawbacks of function point :

1. Subjective counting.
2. Hard to automate and difficult to compute.
3. Ignores quality of output.
4. Oriented to traditional data processing application.
5. Efforts prediction using the unadjusted function count is often no worse than when the TCF is added.

Que 1.28. What do you mean by the software project estimation ? Give various estimation models. Describe any one of the estimation models using suitable examples. **UPTU 2013-14, Marks 10**

OR

Give various estimation models. Describe any one of the estimation model using suitable examples. **UPTU 2014-15, Marks 05**

Answer

Software project estimation : Refer Q. 1.23, Page 29C, Unit-1.

Various estimation models are :

COCOMO model :

1. COCOMO (COnstructive COst estimation MOdel) was proposed by Boehm in 1981.
2. Boehm postulated that any software development project can be classified into one of the following three categories based on the development complexity : organic, semidetached, and embedded.
3. In order to classify a product into the identified categories, Boehm requires us to consider not only the characteristics of the product but also those of the development team and development environment.
4. These three product classes correspond to application, utility, and system programs, respectively.
5. Normally, data processing programs are considered to be application programs.
6. Compilers, linkers, etc. are utility programs.
7. Operating systems and real time system programs are system programs.
8. System programs interact directly with the hardware and typically involve meeting timing constraints and concurrent processing.
9. Boehm's definitions of organic, semidetached, and embedded systems are elaborated as follows :

a. **Organic :** We can consider a development project to be of organic type, if the project deals with developing a well-understood application program, the size of the development team is reasonably small, and the team members are experienced in developing similar types of projects.

b. **Semidetached :** A development project can be considered to be of semidetached type, if the development team consists of a mixture of experienced and inexperienced staff. Team members may have limited experience on related systems but may be unfamiliar with some aspects of the system being developed.

c. **Embedded :** A development project is considered to be of embedded type, if the software being developed is strongly coupled to complex hardware, or if stringent regulations on the operational procedures exist.

10. According to Boehm, software cost estimation should be done through three stages : Basic COCOMO, Intermediate COCOMO, and Detailed/ Complete COCOMO.

a. **Basic COCOMO model :**

i. The basic COCOMO model gives an approximate software development efforts and cost as function of program size expressed in estimated lines of code.

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

Effort (E) = $a^*(\text{KLOC})^b$

Development Time (T_{dev}) = $c^*(E)^d$

Where E is effort applied in person-month, T_{dev} is development time in months.

iii. The coefficients a, b, c, d are constant and can be calculated by given table :

Project	a	b	c	d
Organic	2.4	1.05	2.5	0.38
Semidetached	3.0	1.12	2.5	0.35
Embedded	3.6	1.20	2.5	0.32

iv. When effort and development time are known, the average staff size to complete the project may calculated as :

Average staff size (SS) = E/T_{dev} persons

v. When project size is known, the productivity level may be calculated as :

Productivity (P) = KLOC/E

b. **Intermediate COCOMO :**

i. The basic COCOMO model assumes that effort and development time are functions of the product size alone.

ii. However, a host of other project parameters besides that product size affect the effort required to develop the product as well as the development time.

iii. Therefore, in order to obtain an accurate estimation of the effort and project duration, the effect of all relevant parameters must be taken into account.

iv. The intermediate COCOMO model recognizes this fact and refines the initial estimate obtained through the basic COCOMO

expressions by using a set of fifteen cost drivers (multipliers) based on various attributes of software development.

v. For example, if modern programming practices are used, the initial estimates are scaled downwards by multiplication with a cost driver having a value less than one.

vi. If there are stringent reliability requirements on the software product, this initial estimate is scaled upward.

vii. Boehm requires the project manager to rate these fifteen different parameters for a particular project on a scale of one to three.

viii. Then depending on these ratings, he suggests appropriate cost driver values which should be multiplied with the initial estimate obtained using the basic COCOMO.

ix. In general, the cost drivers can be grouped into four categories :

1. **Product attributes :** The characteristics of the product that are considered include the inherent complexity of the product (CPLX), reliability requirements of the product (RELY) and database size (DATA).
2. **Computer attributes :** The characteristics of the computer that are considered include execution time constraints (TIME), main storage constraints (STOR), virtual machine volatility (VIRT) and computer turnaround time (TURN).
3. **Personnel :** The attributes of development personnel that are considered include the analyst capability (ACAP), application experience (AEXP), programmer capability (PCAP), virtual machine experience (VEXP) and programming language experience (LEXP).
4. **Project attribute :** The characteristics of the project that are considered includes modern programming practices (MODP), use of software tools (TOOL) and required development schedule (SCED).

x. The intermediate COCOMO equations are :

Effort (E) = $a^*(\text{KLOC})^{b**}$ EAF

Development Time (T_{dev}) = $c^*(E)^d$

xi. The EAF (Effort Adjustment Factor) is multiplication of different types of cost drivers. E is an effort applied in person-month, T_{dev} is development time in months.

xii. The coefficients a, b, c, d are constant and can be calculated by given table :

Project	a	b	c	d
Organic	3.2	1.05	2.5	0.38
Semidetached	3.0	1.12	2.5	0.35
Embedded	2.8	1.20	2.5	0.32

c. **Detailed/ complete COCOMO model :**

i. A major shortcoming of both the basic and the intermediate COCOMO models is that they consider a software product as a single homogeneous entity.

ii. However, most large systems are made up of several smaller subsystems. These subsystems may have widely different characteristics.

iii. For example, some subsystems may be considered as organic type, some semidetached, and some embedded.

iv. Not only that the inherent development complexity of the subsystems may be different, but also for some subsystems the reliability requirements may be high, for some the development team might have no previous experience of similar development, and so on.

v. The complete COCOMO model considers these differences in the characteristics of the subsystems and estimates the effort and development time as the sum of the estimates for the individual subsystems.

vi. The cost of each subsystem is estimated separately. This approach reduces the margin of an error in the final estimate.

2. **COCOMO-II :**

i. COCOMO-II is the revised version of the original COCOMO and is developed at University of Southern California under the leadership of Dr. Barry Boehm.

ii. The model is turned to the life cycle practices of the 21st century. It also provides a quantitative analytic framework, and set of tools and techniques for evaluating the effects of software technology improvements on software life cycle costs and schedules.

iii. COCOMO-II provides three detailed cost estimation models. These can be used to estimate project costs at different phases of the software.

iv. As the project progresses, these models can be applied at different stages of the same project.

a. **Application composition :** Here, the external features of the system that the users will experience are designed. Prototyping will typically be employed to do this with small applications that can be built using high-productivity application-building tools, development can stop at this point.

b. **Early design :** Here, the fundamental software structures are designed. With larger, more demanding systems, where, for example, there will be large volumes of transactions and performance is important, careful attention will need to be paid to the architecture to be adopted.

c. **Post architecture :** Here, the software structures undergo final construction, modification and tuning to create a system that will perform as required.

Que 1.29. Discuss the role of cost estimation in a software development project. Briefly explain COCOMO model for cost estimation for all category of projects.

UPTU 2014-15, Marks 05

Answer

In cost estimation, the number of estimation techniques have been developed and are having following attributes in common :

1. Project scope must be established in advance.
2. Software metrics are used as a basis from which estimates are made.
3. The project is broken into smaller pieces which are estimated individually.

To achieve reliable cost and schedule estimates, a number of options arise :

1. Delay estimation until late in project.
2. Use simple decomposition techniques to generate project cost and schedule estimates.
3. Develop empirical models for estimation.
4. Acquire one or more automated estimation tools.

COCOMO model : Refer Q. 1.28, Page 35C, Unit-1.

Que 1.30. Write short note on decision process.

UPTU 2013-14, Marks 05

Answer

1. Software project management (SPM) is basically defined by the ability of decision making.
2. It is the responsibility of managers to design this process and optimize it to minimize costs and maximize production.
3. Decision makings are based on resources and constraints which are planned for the target project and the plan could change hardly in line with new requirements during project progress.
4. But what could be done accordingly to confront changes which are contingent in every project, is to define an optimal plan with effective decisions.
5. Decision making is a cognitive process resulting in the selection of a course of action among several alternative scenarios.
6. This process finally leads the decision maker to take an action or make a choice.

7. Thereby, it is an ability based on experience and knowledge that enables a leader of a process to succeed.
8. The nature of software projects on the other side add other complexity in which development process has intangibility that makes it difficult for managers to design a suitable strategy for decision making.
9. SPM requires special mindset to make practitioners be able to conduct management process in an effective and efficient manner.
10. These mindsets are from any point of view considered as a high level experience and management capabilities, since it originates from a complex process and organizational understanding.
11. Therefore, management of this knowledge requires special strategies for an effective knowledge management.
12. It is evident that an effective approach for modeling the decision making process and redefining the decision structure over SPM is necessary.
13. This model should be able to deal with high level of intangibility and continuous change requests within the software project.

VERY IMPORTANT QUESTIONS

Following questions are very important. These questions may be asked in your SESSIONALS as well as UNIVERSITY EXAMINATION.

Q. 1. Write short note on :
i. Project
ii. Characteristics of project
iii. Project management

Ans.
i. Refer Q.1.1.
ii. Refer Q.1.1.
iii. Refer Q.1.2.

Q. 2. Define SPM and life cycle of software project.
Ans. Refer Q.1.6 and 1.7.

Q. 3. Write short note on :
i. Need and identification
ii. Vision and scope document
Ans. Refer Q. 1.9.

Q. 4. Explain project management cycle and SPM objectives.
Ans. Refer Q. 1.10 and 1.11.

Q. 5. What do you mean by software project planning and planning objectives ?
Ans. Refer Q. 1.15 and 1.17.

Q. 6. Discuss the following :
i. Project plan and its types.
ii. Structure of project management plan.
Ans. i. Refer Q. 1.20.
ii. Refer Q. 1.22.

Q. 7. What do you mean by software project estimation ? Also discuss various techniques and models associated to it.
Ans. Refer Q. 1.23, Q. 1.27 and 1.28.

Q. 8. Define decision process.
Ans. Refer Q. 1.30.

2 UNIT

Project Organization and Scheduling

PART-1

Project Elements, Work Breakdown Structure, Types of WBS, Functions, Activities and Tasks, Project Life Cycle and Product Life Cycle.

CONCEPT OUTLINE : PART-1

- The purpose of work breakdown structure (WBS) is to divide the program/project into manageable pieces of work to facilitate planning and control of cost, schedule and technical content.
- **Types of WBS :**
  a. Product breakdown
  b. Service project breakdown
  c. Result project breakdown
  d. Cross cutting element
  e. Project management
- Task is a generic term for work, that is, not included in the WBS but that potentially could be a further decomposition of work by the individual responsible for that work.
- Various phases of project life cycle are :
  a. Initiation b. Planning
  c. Executing d. Controlling
  e. Closing
- Various types of product life cycle model are :
  a. SDLC model
  b. Waterfall model
  c. Prototyping model
  d. V-process model
  e. Spiral model
  f. Iterative enhancement model
  g. Rapid application development (RAD) model

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 2.1. **What are the elements of project ?**

Answer

In order to ensure that all our projects reach the required level of success, here are the 5 essential elements that need to be included :

1. **Strategic planning :**
   a. The first stage of any project is to understand the need for the project and what it is trying to achieve. SMART (Specific, Measurable, Attainable, Relevant, Timely) objectives need to be established along with measures of success and key milestones where progress can be reviewed.
   b. Working as an internal project manager will require close liaison with key internal stakeholders and departments to establish their specific requirements and set commonly agreed objectives.
2. **Product development :**
   a. The variety of activities that are deemed to be projects are wide-ranging and varied, and can include new products, processes and services. The development of any of these needs to be closely linked to meeting defined business objectives and adding value to the organization.
   b. The benefits of a project should be well articulated at the beginning so there is a clear link to the success of the project and the impact on overall business aims.
3. **Communication :**
   a. It is vital to sell the benefits of any project to those who will be affected during the project or by the project's final outcome.. Implementing a new process requires that end users understand why the project is beneficial and potential buyers need to be convinced by the advantages of new products and services.
   b. In essence, communicating the message of why new or different is good will help counteract the typical human reluctance to change.
4. **Resources :**
   a. It is vital to ensure that adequate resources in terms of people, time, finances and equipment are in place. Internally, this could involve the IT department providing the appropriate hardware/ software, Human Resources recruiting the necessary people or the Facilities department providing offices or other relevant support.
   b. There also needs to be allocated budgets and finance as well as appropriate timelines for project completion.
5. **People :** No project manager works in isolation. There are many stakeholders involved in a project who all have a specific role to play and who all have a vested interest in the project's success. The key stakeholders who drive projects and help make them a success include:
   a. **Sponsor :** The project sponsor is the person who defines the business objectives that drive the project. The sponsor can be a member of the senior management team or someone from outside of the organization.

   b. **Project manager :** A professional project manager creates the project plan and ensures that it meets the budget, schedule and scope determined by the sponsors. The project manager is also responsible for risk assessment and management.
   c. **Project team members :** These can include subject area experts, members of departments, external professionals and new recruits. Anyone who can offer a positive contribution to the project in terms of their knowledge and capabilities makes a good team member.

Que 2.2. **Write short note on work breakdown structures.**

UPTU 2012-13, Marks 05

OR

What do you mean by Work Breakdown Structure (WBS) in context to software project and product ? Discuss with examples.

UPTU 2014-15, Marks 05

Answer

1. Work Breakdown Structures (WBS) were first used by the U.S. department of defense for the development of missile systems in mid-1960.
2. The purpose of a work breakdown structure (WBS) is to divide the program/project into manageable pieces of work to facilitate planning and control of cost, schedule and technical content.
3. It identifies the total work to be performed and divides the work into manageable elements, with increasing levels of detail.
4. Work breakdown structure (WBS) is a chart in which the critical work elements, called tasks, of a project are illustrated to describe their relationships to each other and to the project as a whole.
5. The graphical nature of the WBS can help a project manager predict outcomes based on various scenarios, which can ensure that optimum decisions are made about whether or not to adopt suggested procedures or changes.
6. A well-organized, detailed WBS can assist key personnel in the effective allocation of resources, project budgeting, procurement management, scheduling, quality assurance, quality control, risk management, product delivery and service oriented management.
7. When creating a WBS, the project manager defines the key objectives first and then identifies the tasks required to reach those goals.
8. A WBS takes the form of a tree diagram with the "trunk" at the top and the "branches" at the bottom.
9. The primary requirement or objective is shown at the top, with increasingly specific details shown as the observer reads down.

10. Fig. 2.2 shows how a typical WBS looks like.

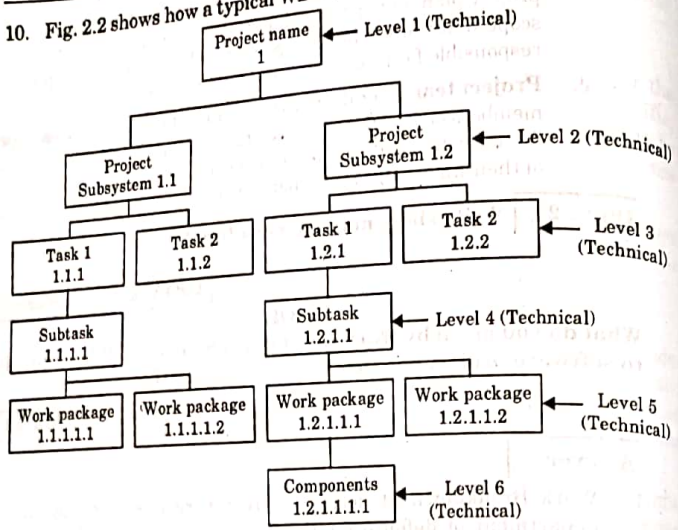


Fig. 2.2.1.

11. There are six levels till where we can drill down and breakdown the job structure :

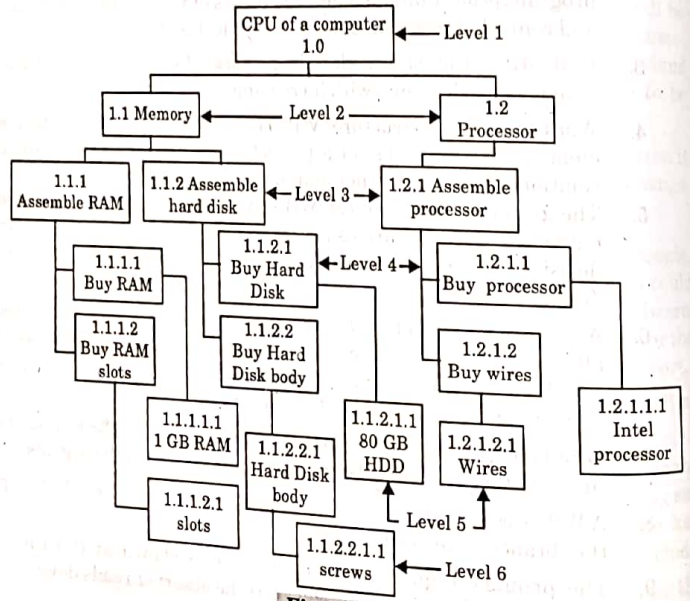


Fig. 2.2.2.

a. **Level 1 :** The first level is the project name or the name of the assignment.
b. **Level 2 :** Level 2 represents the subsystem which will make up the project.
c. **Level 3 :** Level 3 shows the task to be performed to complete the subsystem from managerial aspect.
d. **Level 4 :** The main task is further broken down into sub tasks from a technical aspect.
e. **Level 5 :** This is the final deliverable also termed as work package.
f. **Level 6 :** These are components needed to form the work package.

Example : The following figure shows 'WBS for CPU' for assembling a CPU of a computer.

Que 2.3. Write short note on work breakdown structure with its type.

UPTU 2013-14, Marks 05

Answer

Work breakdown structure : Refer Q. 2.2, Page 45C, Unit-2.

Types of WBS :

There are different types of projects and, therefore, different types of WBSs, each with unique elements. All WBSs have two or more of the five types of level 2 elements as shown in Fig. 2.3.1.

These five types of elements are as follows :

1. **Product breakdown :**
   a. The subdivision based on the physical structure of the product(s) being delivered (as the capital asset) is the most common basis for a WBS and the easiest WBS to develop.
   b. These projects have a tangible output product : software, a building, a dam, an airplane, a user's manual, etc., all have a natural structure.
   c. Alternatively, there may be multiple products delivered such as an airport ground surveillance system, an IT system for a centralized clearing house, an integrated deep water system, or an orbiting space laboratory system.
2. **Service project breakdown :**
   a. Service projects do not have a tangible, structured deliverable. The output is a defined body of work done for others : conference, party, wedding, vacation trip, etc.
   b. The work breakdown is a logical collection of related work areas.
3. **Results project breakdown :**
   a. Results projects do not have a tangible, structured deliverable. The output is the consequence of a process that results in a product or a conclusion : cancer research, new drug development, culture change, etc.
   b. The work breakdown is a series of accepted steps.

4. **Cross-cutting element :**
   a. This is a breakdown of items that cut across the product, such as architectural design, assembly or system test.
   b. These usually are technical and supportive in nature. There may be more than one element of this characteristic at level 2.
   c. While there is no restriction, these types of cross-cutting elements are rare in service or results projects.
5. **Project management :**
   a. This is a breakdown of the managerial responsibilities and managerial activities of the project.
   b. It includes items such as reports, project reviews, and other activities of the project manager or his or her staff (Conceptually, these are the overhead of the project).
   c. Normally, there is only one of these types of WBS element, but it exists on all projects as a level 2 element.

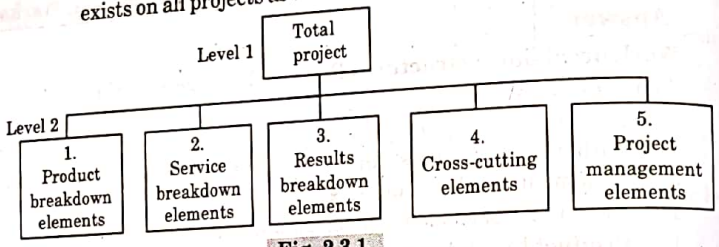


Fig. 2.3.1.

Que 2.4. What are the principles of WBS ? Discuss in detail.

Answer

WBS principles are as follows :

1. The WBS covers the total scope of the project. Work not involved in the WBS, is not in the project.
2. All deliverables or output products are represented in the WBS.
3. The sum of the elements at each level represents 100 percent of the work of the next higher level (The sum of the level 2 items is 100 percent of the project work or cost).
4. Work in each element is equivalent to the sum of the work in the subordinate elements.
5. The subdivisions should be logical and reflect the nature of the product, service, or result.
6. Each WBS element should represent a discrete element of work that can be described in the WBS dictionary.
7. Each WBS element should have a unique identifier.
8. WBS element descriptors preferably should be nouns, with adjective modifiers if necessary. For clarity or for cultural reasons, WBS descriptors may include verbs and modifiers. However, they should not be considered activities since activities are the action elements that occur below the WBS.
9. The work in each WBS element may be described in detail in a WBS dictionary, which may become the basis for statements of work or work-authorizing documents.
10. Project management is a level 2 element in all WBSs.
11. Stakeholders should participate in the development of the WBS.
12. The WBS should be baselined after approval by the stakeholders.
13. A formal change process should exist for baselined WBSs.
14. The WBS should focus on project output or deliverables; it is not an organization chart, a schedule, or a resource list.
15. The lowest level should be the level above the activities—the work package level.
16. The lowest level should permit adequate control and visibility for project management.
17. The lowest level need not be the same for all branches of the WBS.
18. The lowest level should not be so detailed as to create an administrative burden.
19. The WBS does not reflect time relationships or horizontal relationships between elements; all structural relationships are vertical.

Que 2.5. What are the recommended steps to develop a project WBS ?

Answer

The recommended steps to develop a project WBS are as follows :

Step 1 : Identify the project objectives (this will assist in steps 2 and 3).

Step 2 : Determine the general type of project by identifying specifically whether the primary output is a product, service, or result.

Step 3 :
   a. If the project output is a product, level 2 will include the product name, secondary product names, and cross-cutting elements. Make sure all project outputs can be related to a level 2 element (proceed to step 4).
   b. If the project output is a service, level 2 will include the top-level groupings of the various types of work and the project management element. Consider identifying as many activities as possible and grouping them by logical categories related to areas of work (bottom-up synthesis) (proceed to step 5).
   c. If the project output is a result, level 2 will consist of the major steps in the acknowledged process necessary to achieve the result plus the project management element (proceed to step 6).

Step 4 : For product WBSs, subdivide the product element into the logical physical breakdown of the product. Subdivide the cross-cutting elements into the supporting work (proceed to step 7).

Step 5 : For service WBSs, subdivide the level 2 WBS elements into logical functional work areas (proceed to step 7).

Step 6 : For results WBSs, subdivide each level 2 WBS elements into the standard processes specified to achieve the objective or output of the element (proceed to step 7).

Step 7 : Review the work at each level to make sure 100 percent of the work is identified; add elements as necessary. In a product WBS, make sure integrative elements are added as necessary.

Step 8 : Continue to subdivide the elements to the work package level. Further subdivision would violate the principles outlined above. Stop when the next level would be activities or is unknown until further analysis or planning is performed.

Step 9 : Review the WBS with stakeholders and adjust as necessary to make sure that all the project work is covered.

Que 2.6. What are the main features and uses of WBS ?

Answer

Features of WBS :

1. **Structure :** WBS diagram is drawn like the organization chart. Different desktop applications offer functionalities to easily create this kind of diagrams.
2. **Description :** Each WBS element should be described with a title. The meaning of each title should be clear.
3. **Coding :** One of the main features of WBS is the ability to uniquely code the different elements of the work. The coding system can be alphabetic, numeric or alphanumeric.
4. **Depth :**
   a. The recommended depth of a WBS diagram is three to four levels. If deeper hierarchies are required, the division into subprojects can be used and one element would then present one subproject.
   b. The downside adding of too many levels is firstly the readability of the diagram and secondly, the fact that the larger the diagram, the more troublesome it is to update when major changes occur in the project.
5. **Level of detail :** A rule of thumb is creating a WBS diagram is to make the lowest level element (often called work package) small enough to be considered a separate work element when estimating the ar ount of work in a project. This allows the work package later to be divided into a list of work activities and tasks.

Uses of WBS :

1. For dividing the project into phases.
2. For dividing the project into responsibility areas within the organization.
3. For dividing the schedule of the project into sub-schedules whose interrelations are known.
4. For giving grounds to following the cost of the project by defining clear targets to it.
5. For giving hierarchical outlining and coding for the work to be done.
6. For enabling integrating planning and managing of the project from both financial and scheduling perspective.

Que 2.7. What do you mean by activity ? Explain different characteristics of activity.

Answer

1. Before we try to identify the activities that make up a project, it is worth reviewing what we mean by a project and its activities and adding some assumptions that will be relevant when we start to produce an activity plan.
   a. A project is composed of a number of interrelated activities.
   b. A project may start when at least one of its activities is ready to start.
   c. A project will be completed when all of the activities it encompasses have been completed.
   d. An activity must have a clearly defined start and a clearly defined endpoint, normally marked by the production of a tangible deliverable.
   e. If an activity requires a resource, then that resource requirement must be forecastable and is assumed to be required at a constant level throughout the duration of the activity.
   f. The duration of an activity must be forecastable-assuming normal circumstances, and the reasonable availability of resources.
   g. Some activities might require that others are completed before they can begin (these are known as precedence requirements).
2. The activity-based approach consists of creating a list of all the activities that the project is thought to involve.
3. This might involve a brainstorming session involving the whole project team or it might stem from an analysis of similar past projects.
4. When listing activities, particularly for a large project, it might be helpful to subdivide the project into the main lifestyle stages and consider each of these separately.

5. Rather than doing this in an adhoc manner, with the obvious risks of omitting or double-counting tasks, a much favoured way of generating a task list is to create a Work Breakdown Structure (WBS).
6. This involves identifying the main (or high-level) tasks required to complete a project and then breaking each of these into a set of lower level tasks. Fig. 2.7.1 shows a fragment of a WBS where the design task has been broken down into three tasks and one of these has been further decomposed into two tasks.

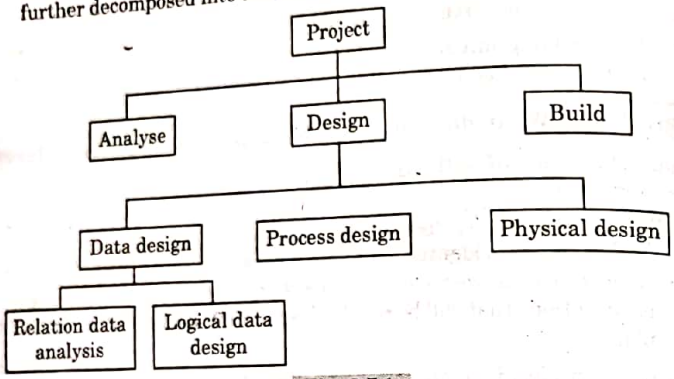


Fig. 2.7.1.

7. Activities are added to a branch in the structure if they directly contribute to the task immediately above if they do not contribute to the parent task, then they should not be added to that branch.
8. The tasks at each level in any branch should include everything, that is, required to complete the task at the higher level if they are not a comprehensive definition of the parent task, then something is missing.
9. When preparing a WBS, consideration must be given to the final level of detail or depth of the structure.
10. Each branch should, however, be broken down at least to a level where each leaf may be assigned to an individual or responsible section within the organization.
11. Advantages claimed for the WBS approach include the belief that it is much more likely to result in a task catalogue that is complete and is composed of non-overlapping activities.
12. Note that, it is only the leaves of the structure that comprise the list of activities in the project, higher level nodes merely represent collections of activities.

The characteristics of activities are as follows :

1. Work is performed and described in terms of a verb, adjective, and noun.

2. A single person or organization is responsible for the work — more than one resource may be assigned to an activity, but one person is in charge of delivering the output.
3. It has defined start and finish points — there is either a specific predecessor activity or event that must be completed first or a specific date on which the activity is scheduled to start; the scheduled end date is based on the estimated duration, baseline duration, or scheduled duration of the activity.
4. Usually, there is a tangible output or product at completion — projects occasionally have level-of-effort activities or support activities without clearly defined outputs.
5. It fits logically under an existing WBS element — if it does not, the activity is not part of the project, the WBS needs modification, or the activity is ambiguous and needs redefinition.
6. Actual schedule status data can be collected for the activity — for schedule control, the start and end points must be sufficiently defined so that the start and finish of the activity can be reported.
7. Actual cost (person-hour) data can be collected for the activity or work package that contains the activity — for cost or resource control, actual cost data or the actual expenditure of resources can be collected.
8. The labour and costs necessary to perform the activity can be estimated — the resource requirements must be able to be determined in the planning phase.
9. The output of the activity is known or can be identified — outputs are frequent pieces of paper or other tangible proof of the activity being completed.
10. The activity represents a significant effort in support of project objectives — trivial or incidental activities need not be included.
11. Zero duration activities are milestones or events and represent the start or completion of another activity or set of activities — they should be included at the start and finish of the project and included to identify completion of key activities or groups of activities.

Que 2.8. What are different product identification techniques ?

Answer

1. For identifying tasks and activities, product development technique needed is an awareness of the processes of software development using different life cycles.
2. Life cycles must be evaluated and tailored to the individual needs of each project.
3. A general understanding of software development activities (software engineering) and how to define the software product are competencies needed for task and activity identification.

4. The project management skills needed here are a continuation of those needed for building the WBS, such as documenting plans in an arranged structure and finding tasks and activities that can be used to create a schedule.
5. Activity ID requires the people skills of leadership, interface, and communication, and the ability to present ideas effectively throughout the identification process.
6. The product development techniques are as follows :
   a. **Assessing processes :** Defining criteria for reviews.
   b. **Defining the product :** Identifying customer environment and product requirements.
   c. **Evaluating alternative processes :** Evaluating various approaches.
   d. **Tailoring processes :** Modifying standard processes to suit a project.
   e. **Understanding development activities :** Learning the software development cycle.

Que 2.9. **Diagrammatically represent the product life cycle model.**

UPTU 2012-13, Marks 05

OR

Write short note on product life cycle.

OR

Write short note on project life cycle and product life cycle.

UPTU 2013-14, Marks 05

OR

What do you mean by project life cycle and product life cycle ?

UPTU 2014-15, Marks 05

Answer

The product life cycle is discussed as follows :

1. **Product life cycle :**
   a. The product life cycle describes the sales pattern of a product over time. Generally, the time span begins with product introduction and ends with its obsolescence and replacement.
   b. While the form of life cycle is fairly standard, it is subjected to variations.
   c. The concept underlying the premise of product life cycle is that all products pass through the stages outlined below.

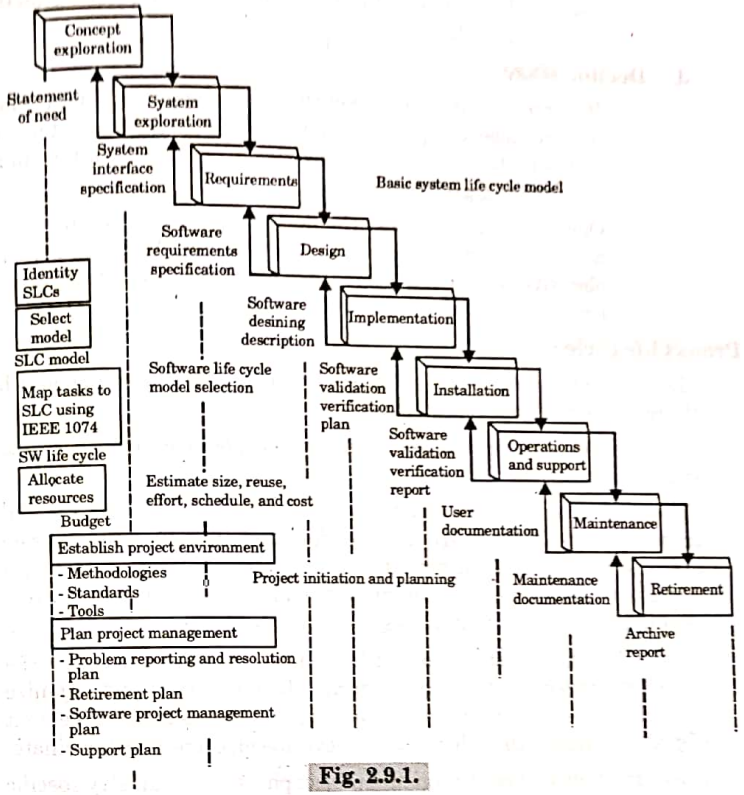


Fig. 2.9.1.

2. **Basic stages in the product life cycle :** There are basic four stages of PLC :
   a. **Development stage :** It is the first stage and represents a slow growth period. It is assumed that newly released products require some time to gain market acceptance, so sales in the initial period are slow.
   b. **Growth stage :** It is the second stage of PLC. Once the product introduction proved successful, rapid growth stages are reached and sales increase according to the market value.
   c. **Mature stage :**
      i. It is the third stage of PLC. According to the concept of life cycle, the market for any product is limited, and sales will generally fall short of their potential.
      ii. When this point is reached, the market enters the maturation stage. The life cycle goes further to assume that each product

eventually is replaced by another or that initial rapid growth will end in decline.

d. Decline stage :

i. If a product enters a market that has already moved into the mature stage, competition is intense because the product must compete for a share of an existing market that is not experiencing growth.

ii. Once the market enters the decline stage, new products do not enter the market and demand level falls. At this point, the objective is to increase market share to maintain stable sales level.

Project life cycle :

1. The project life cycle refers to a logical sequence of activities to accomplish the project's goals and objectives.
2. Regardless of scope or complexity, any project goes through a series of stages during its life.
3. There is first an initiation or birth phase, in which the outputs and critical success factors are defined, followed by a planning phase, characterized by breaking down the project into smaller parts/tasks, an execution phase, in which the project plan is executed, and lastly a closure or exit phase, that marks the completion of the project.
4. Project activities must be grouped into phases because by doing so, the project manager and the core team can efficiently plan and organize resources for each activity, and also objectively measure achievement of goals and justify their decisions to move ahead, correct, or terminate.
5. It is of great importance to organize project phases into industry specific project cycles.
6. Not only because each industry sector involves specific requirements, tasks, and procedures when it comes to project, but also because different industry sectors have different needs for life cycle management methodology. And paying close attention to such details is the difference between doing things well and excelling as project managers.
7. Diverse project management tools and methodologies prevail in the different project cycle phases.

Following are the stages of project life cycle :

1. Initiation :

a. In this stage, the scope of the project is defined along with the approach to be taken to deliver the desired outputs.

b. The project manager is appointed and in turn, he selects the team members based on their skills and experiences.

c. The most common tools or methodologies used in the initiation stage are project charter, business plan, project framework (or overview), business case justification and milestone reviews.

2. Planning :

a. The second phase should include a detailed identification and assignment of each task until the end of the project.

b. It should also include a risk analysis and a definition of criteria for the successful completion of each deliverable.

c. The governance process is defined, stakeholders identified and reporting frequency and channels agreed.

d. The most common tools or methodologies used in the planning stage are business plan and milestones reviews.

3. Execution and controlling :

a. The most important issue in this phase is to ensure project activities properly executed and controlled.

b. During the execution phase, the planned solution is implemented to solve the problem specified in the project's requirements.

c. In product and system development, a design resulting in a specific set of product requirements is created. This convergence is measured by prototypes, testing, and reviews.

d. As the execution phase progresses, groups across the organization become more deeply involved in planning for final testing, production, and support.

e. The most common tools or methodologies used in the execution phase are an update of risk analysis and score cards, in addition to business plan and milestone reviews.

4. Closure :

a. In this last stage, the project manager must ensure that the project is brought to its proper completion.

b. The closure phase is characterized by a written formal project review report containing the following components :

i. a formal acceptance of the final product by the client,

ii. weighted critical measurements (matching the initial requirements specified by the client with the final delivered product),

iii. rewarding the team, a list of lessons learned,

iv. releasing project resources and

v. a formal project closure notification to higher management.

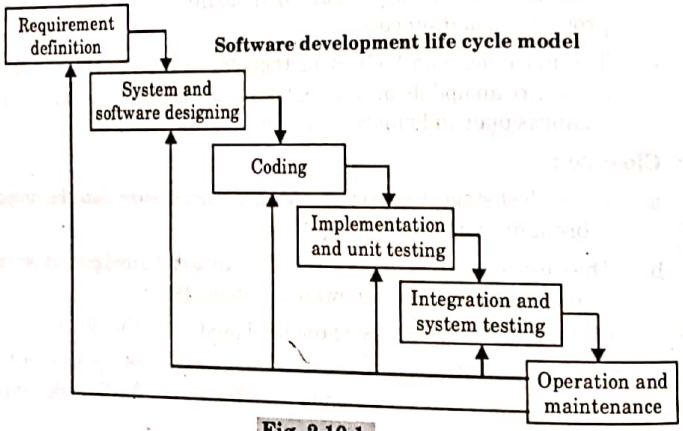
c. No special tool or methodology is needed during the closure phase.

Que 2.10. Discuss SDLC model in brief.

Answer

Software Development Life Cycle (SDLC) :

1. Software development life cycle represents number of identifiable stages under which software goes during its life.
2. It is a diagrammatic representation which also provides description of various phases and their sequence in life cycle of software product.
3. Software undergoes some basic stages during its life cycle *i.e.*, requirement analysis and specification, design, coding, testing and maintenance.
4. There are many software models which are used as per requirement of software product.
5. All models undergo these basic stages while their mapping of the stages may be different as per model requirement.
6. We have different life cycle models, each one have its own advantages and disadvantages. We can choose any one of them on the basis of :
   a. Development speed
   b. Product quality
   c. Project visibility
   d. Administrative overhead
   e. Risk exposure



Software development life cycle model

Fig. 2.10.1.

Phases of software development life cycle models :

1. Requirement definition (system analysis and system specification)
2. System and component (software) design
3. Coding
4. Implementation and unit testing
5. Integration and system testing
6. Operation and maintenance

Que 2.11. Explain waterfall model and its limitations.

UPTU 2014-15, Marks 05

Answer

Waterfall model :

1. Waterfall model is a theoretical software development model which was used in 70's.
2. It is also known as classical, traditional, conventional or linear segment model.
3. "There are different stages to the development and the output of first stage flow to the next (second) stage and output of second flows to third stage and so on.
4. It force on sequential phase development in which no phase can overlap another phase and so the developer must complete each phase before starting next phase.
5. Each phase of this model has a well defined starting and ending criteria which is to be documented by which the standard outputs (deliverables) to be produced by each phase can formulate.
6. This model does not allow to go back to the previous stage from one stage "one way street with no turning back" like waterfall that's why it is called waterfall model.
7. The different phases of this model are :

a. Feasibility study :

i. This phase is used to check whether the new proposed system is economically, technically and operationally feasible or not.

ii. In which information is gathered about what output to be produce, input required and process can be used and then different solution strategies are formulated.

iii. Finally, analysis of all solutions done on the basis of their cost and benefits and accordingly the best solution is selected.

b. Requirement analysis and specification :

i. This phase give specification about what is the system for.

ii. This phase analyze and specifies the requirement of user/customer and document them properly.

iii. It is a very critical phase of software development life cycle as whole project is planned and scheduled here for development.

iv. In requirement analysis, the data are gathered from users using different methods such as interviews, questionnaires, on site observation and through written document of the organization.

v. And, then it is to be assured that the requirements are understood properly and no inconsistency, incompleteness and ambiguity are left.

vi. Finally, the requirements are organized systematically in the form of document called software requirement specification (SRS) document.

c. **System and software designing phase :** In design phase, overall structure or architecture is developed which is transformation of requirement specified in SRS.

d. **Coding and module testing :**

i. In this phase, system design is translated into source code also called program code.

ii. Here, programming for different module is done in selected programming language. End product of coding phase is module testing, in which each module is tested individually whether they are working properly or not, this is also called unit testing.

iii. The output of this phase is programmed module.

e. **Integration and system testing :**

i. According to plan, individually tested module are integrated to develop the system.

ii. Generally, in this integration all the module are not joined together to form the system rather than it is done in various steps and during these steps the partially integrated system is tested and then the next module added to it and again the testing is done.

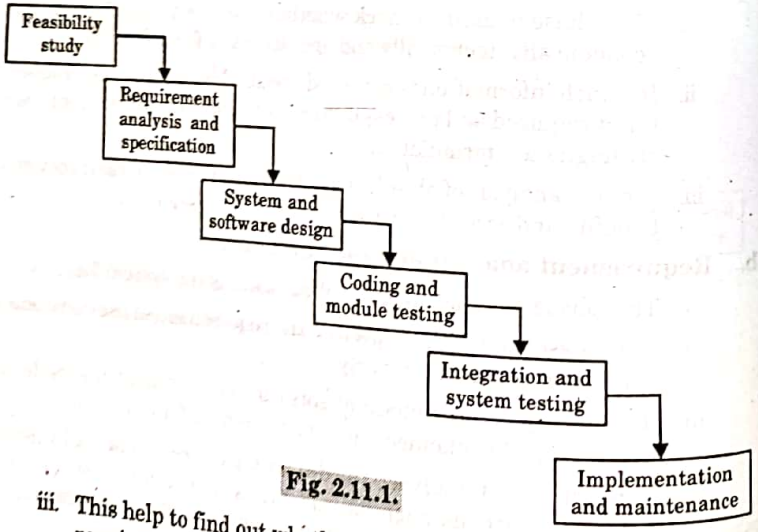


Fig. 2.11.1.

iii. This help to find out whether the developed system conforms the requirement or not.

iv. The output of this phase is testing and integration report.

f. **Implementation/Installation and maintenance :**

i. In this phase, system is installed at the user end and it is checked. If there is any up gradation required in hardware or software element at user end that is made available.

ii. Training is given to user staff for using the system.

iii. Once the software is properly installed there is need of maintaining the software.

iv. This ensures that software is working properly at user site.

v. The maintenance requires much more efforts than software development.

Advantages of waterfall model :

1. Easy to understand.
2. Each stage has well defined input and output.
3. Helps in project planning.
4. It provides a template into which models for analysis, design, code, test and support can be placed.
5. It provides structure to a technically weak or inexperienced staff.

Disadvantages of waterfall model :

1. Iteration not possible as it is one way street.
2. Requirements freezing at starting stage.
3. Sequencing, no stage can start until the previous stage is completed.
4. A rigid model.
5. Difficulty in accommodating changes after project development.
6. Customer gets opportunity very late to review the project so less user involvement during development process.

Que 2.12. What is prototyping model ? When it is used ?

Answer

Prototype model :

1. There are certain drawbacks in waterfall model.
2. This model developed to overcome from main two drawbacks of waterfall model.
3. They are :
   a. Difficult to predict how the new system will be.
   b. Difficulty in predicting the entire requirements at very beginning of project, because even end user don't know all requirements initially.

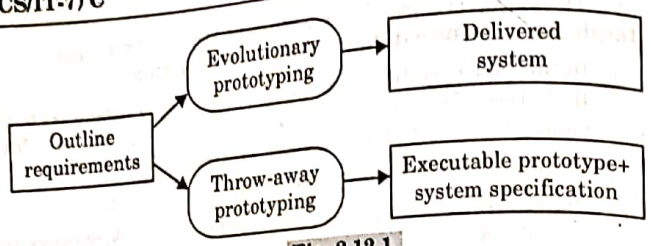


Fig. 2.12.1.

4. In prototype model, firstly a working prototype is developed instead of developing actual software.
5. This is developed according to available requirements which basically have limited functions, low reliability while it passes through all stages of development *i.e.*, design, coding, testing but is done formally.
6. Now, this model is used by developer and given to user for review which helps the user to better understand his need and requirement and then feedback from user is collected and given to developer that helps to remove uncertainties in the requirements of the software.
7. Prototype modeling is of two types :
   a. Evolutionary/Exploratory prototyping
   b. Throw-away prototyping

Que 2.13. Explain V-process model.

Answer

1. Figure 2.13.1 shows a diagrammatic representation of V-process model.

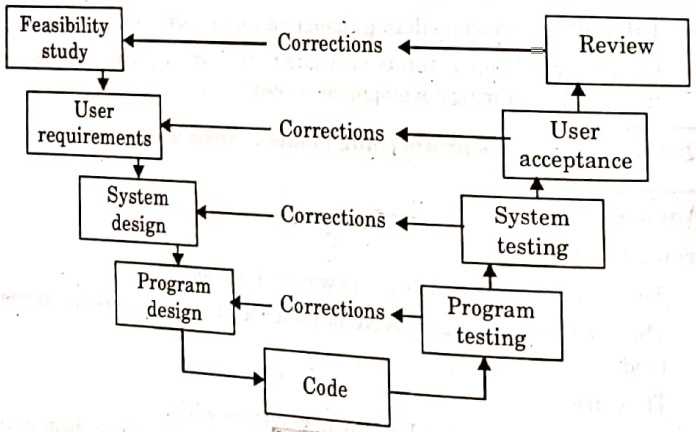


Fig. 2.13.1.

2. This is an elaboration of the waterfall model and stresses the necessity for validation activities that match the activities that create the products of the project.

3. The V-process model can be seen as expanding the activity testing in the waterfall model.
4. Each step has a matching validation process which on finding the defects causes a loop back to the corresponding development stage and a reworking of the following steps.
5. Ideally, this feedback should only occur where a discrepancy has been found between what was specified by a particular activity and what was actually implemented in the next lower activity on the descent of the V loop.
6. For example, the system designer might have written that a calculation be carried out in a certain way.
7. A second developer building code to meet this design might have misunderstood what was required.
8. At system testing stage, the original designer would be responsible for checking that the software is doing what was specified and this would discover the coder's misreading of that document.

Que 2.14. Write short note on spiral model.

Answer

1. In 1987, Boehm proposed a model for the development of software known as Boehm spiral life cycle model.
2. According to name, the activities of this model are organized like a spiral that has many circles whose number depends on software requirement.
3. The radial dimension of this model, the cumulative cost for accomplishing different stages (phases) and angular dimension show the progress in completing each cycle of the spiral.
4. The main objective of this model is to minimize the risk through the use of prototype. This model is mainly used for large projects.
5. The spiral model can said to be made up of waterfall model in which each stage is preceded by risk analysis.
6. Its main feature is risk avoidance rather than documentation or coding.
7. This model is more flexible than any other model as number of phases through which the product will be developed is not fixed, it depends on software requirement.
8. The two basic step of this model are :
   a. Identify the sub-problem which is having highest risk.
   b. Find solution for that particular problem (risk).
9. Generally, there are four spirals in Boehm spiral life cycle model.
10. The inner (first) spiral is concept development cycle, the second spiral indicates new product development cycle; the third spiral represents product enhancement cycle and fourth spiral is known as product maintenance cycle.

11. Each phase of this model is split into four quadrant (sections) having specific functions :

a. In the first quadrant, we do identification of objectives; find out different alternative for achieving the objective and present constraints.

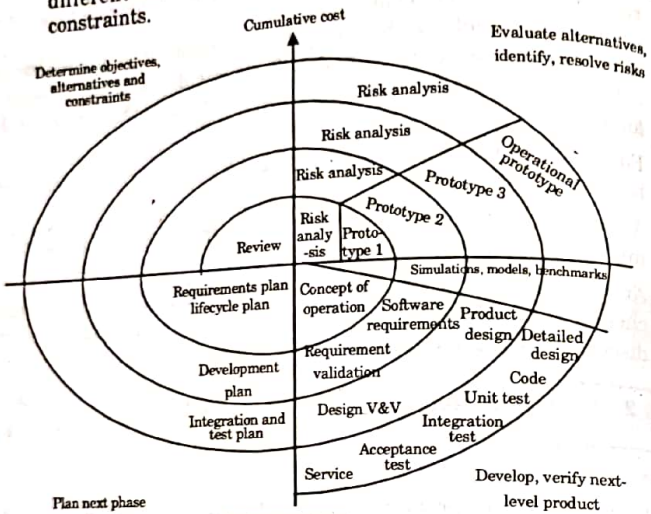


Fig. 2.14.1.

b. In the second quadrant, we evaluate these alternatives on the basis of objective and constraints. The main focus in this step is given on evolution of alternative on the basis of risk as risk causes the chances of unmet objectives.

c. In the third quadrant, project development and validation is carry out.

d. In the fourth quadrant, the project is reviewed and decision is made up whether to continue with a further loop of spiral or not. If it is decided to continue, the project plan is drawn up for the next phase of project.

Que 2.15. Discuss incremental life cycle model in brief.

Answer

1. The incremental life cycle model is similar to the waterfall in many respects, but it differs in that it produces some tangible results to the customer sooner.
2. The initial processes of system requirements and feasibility, software requirements and general design are done in sequence, once for the overall project.
3. A partitioning into increments then occurs, where a number of different development efforts, beginning with detailed design are identified.

4. These increments can be planned as sequential or parallel efforts, depending upon the project characteristics and project constraints.
5. For the same reason as the waterfall model, incremental life cycle model is suitable for large projects with requirements that are known, stable and understood.
6. Due to above characteristics, incremental life cycle model is the best choice when early release of some parts of the software is beneficial or when earlier release of the entire system can be accomplished through multiple teams working in parallel on different increments.
7. When requirements are known and understood but may not be stable, the incremental model is a logical choice because later releases can incorporate changes in that surface during the earlier development efforts.
8. Use of this model requires careful partitioning of the system/product and well defined interfaces between the increments, especially if they will be developed in parallel.
9. Project managers using the incremental model must be aware of the need for additional attention to the coordination of multiple efforts.
10. This includes additional effort that will likely be needed in the test process, where integration is addressed.

Que 2.16. Discuss evolutionary life cycle model in brief.

Answer

1. The evolutionary life cycle model applies in sequential aspects of the waterfall model, and partitioning of the project borrowed from the incremental model, but adds the evolution or the discovery of requirements.
2. Even though incremental development is planned when using the evolutionary model, the increments are not likely to be developed in parallel.
3. This is because the purpose of each increment is to deliver a portion of the system with known requirements for subsequent increments.
4. Unlike the waterfall and incremental models, the system requirements and feasibility and software requirements processes are not done once for the entire project, but these are revised at the start of each evolutionary development cycle to incorporate the latest requirements.
5. Evolutionary life cycle model is preferable life cycle model when requirements are not fully known, but a subset of the requirements is known, stable and understood.
6. Benefits include the early delivery of some functions and the early testing of some assumptions before the entire system is built around them.

7. The major weakness of this model is related to the inability to plan in detail at the outset of the project.
8. Because the requirements are not fully known, problems with the scope creep, inaccurate estimating and less than optimal architecture are possible.
9. The predominately sequential nature of this life cycle makes it not particularly rapid or cost efficient for complex systems.
10. Project managers using the evolutionary model must plan to revise the overall architecture as the system evolves.
11. Flexibility must also be built-in at the individual software module level.
12. Time that appears to be gained on the front end of a project because of the early release of the first increment may be spent on later increments on modification and integration efforts.

Que 2.17. Explain iterative enhancement model in brief.

Answer

1. The classical waterfall model work on the concept that once the requirements specified, no further change will require in any phase of life cycle of product.
2. Iterative model is developed to overcome this drawback of waterfall model.
3. It is a combination of benefits of waterfall and prototype model. It is very popular model used by industries.
4. In this model, software is developed in increments, each increment adds some functional capability to the system until full system is developed.
5. It provide better testing result as testing after each increment is easy as compare to entire model testing of waterfall model.
6. Prototyping used in this model help in identifying the system requirements.
7. In this model, a partial product is developed on few easily understandable requirements of overall requirements, and then a project control list is developed which contain the entire task which have to be performed in final implementation.
8. This helps in finding out how far the product is from final product.
9. This product is given to the user to work and slowly enhancement is done in this product which increases its functionality. That's why it is called incremental model.
10. The model prioritizes the system requirement and implements them in group.
11. With each step, next task is removed from control list. Designing, coding and implementation of selected task is done and analysis of partial

product is done for checking the performance after this increment and the list is updated.

12. The process is iterated until the control list is empty and final implementation of system is done.
13. In this model, developer themselves provide specification, so they have good control over system development.

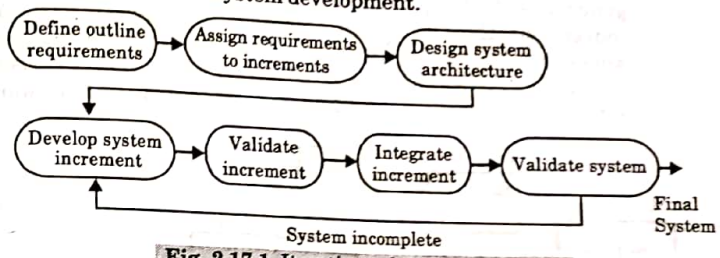


Fig. 2.17.1. Iterative enhancement model.

Que 2.18. Discuss Rapid Application Development (RAD) model.

Answer

1. Rapid application development (RAD) is an incremental software development process model that emphasizes an extremely short development cycle.
2. The RAD model is a "high-speed" adaptation of the linear sequential model in which rapid development is achieved by using component-based construction.
3. If requirements are well understood and project scope is constrained, the RAD process enables a development team to create a "fully functional system" within very short time periods (example, 60 to 90 days).
4. Used primarily for information systems applications, the RAD approach encompasses the following phases :
   a. **Business modeling :** The information flow among business functions is modeled in a way that answers the following questions :
      i. What information drives the business process ?
      ii. What information is generated ?
      iii. Who generates it ?
      iv. Where does the information go ?
      v. Who processes it ?
   b. **Data modeling :** The information flow defined as part of the business modeling phase is refined into a set of data objects that are needed to support the business. The characteristics (called attributes) of each object identified and the relationships between these objects are defined.

c. **Process modeling :** The data objects defined in the data modeling phase are transformed to achieve the information flow necessary to implement a business function. Processing descriptions are created for adding, modifying, deleting, or retrieving a data object.

d. **Application generation :** RAD assumes the use of fourth generation techniques rather than creating software using conventional third generation programming languages. The RAD works to reuse existing program components (when possible) or create reusable components (when necessary). In all cases, automated tools are used to facilitate construction of the software.

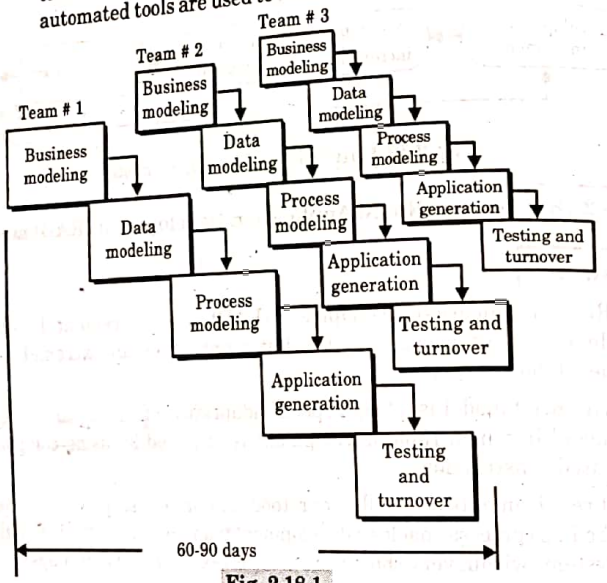


Fig. 2.18.1.

e. **Testing and turnover :** Since the RAD process emphasizes reuse, many of the program components have already been tested. This reduces overall testing time. However, new components must be tested and all interfaces must be fully exercised.

Que 2.19. What do you mean by project feasibility ? Discuss the project feasibility report and its format.

Answer

Project feasibility :

1. When we are developing a system (software), we must know whether proposed system will be feasible or not *i.e.*, practically implemented or not ?
2. It may be possible that the proposed (candidate) system may not be implemented due to many reasons like, it may take long time in development than the specified time limit, cost may increase than proposed one etc. Therefore we must analyze feasibility of system.
3. Feasibility is the analysis of risks, costs and benefits relating to economics, technology and user operations.
4. The output of feasibility study is a document known as feasibility study report.
5. The report must be answering the following key questions :
   a. Will the new system provide a better way to do jobs ?
   b. What the proposed system will do ?
   c. What will be estimated cost of proposed system ?
   d. What will be benefits from proposed system ?
6. There are several types of feasibility depending on the aspect they covers. All the feasibilities are equally important. These are described below :
   a. **Technical feasibility :** The technical feasibility study basically centers on alternatives for hardware, software and design approach to determine the functional aspect of system. The technical issues raised during the feasibility study are :
      i. Does the necessary technology exists (can it be acquired) to do suggested ?
      ii. Does the proposed equipment have technical capacity to hold the data required to use the new system ?
      iii. Will the proposed system and components provide adequate response to enquiries regardless of number or location of users ?
      iv. Can the system be expanded, if developed ?
      v. Are there technical guarantee of accuracy, reliability and security of data ?
   b. **Operational feasibility :** Operational feasibility is a measure of how people are able to work with system. This type of feasibility demands if the system will work when developed and installed. The following question may help to test the operational feasibility of a project :
      i. Is there a sufficient support for the project from management and user ?
      ii. Have the users been involved in the planning and development of project.
      iii. Are current business methods acceptable to user ?
   c. **Economical feasibility :**
      i. Economic analysis is most frequently used for evaluating the effectiveness of proposed (candidate) system, more commonly known as cost benefit analysis.

ii. The cost benefit analysis is used to determine benefits and savings that are expected from candidate system and compare them with cost.

iii. If the benefits are more than the cost then decision is made to design and implement the system.

iv. The cost and benefits may be direct or indirect and tangible or intangible.

Project feasibility report and its format :

1. The report is a formal document for management use, brief enough and sufficiently non-technical to be understandable, yet detailed enough to provide basis for system design.
2. There is no standard format for preparing feasibility reports. Analysts usually decide on format that suits the particular user and system. Generally a feasibility report contains following sections :
   a. The cover letter, which represent report formally and brief introduction to management nature, general finding and recommendation.
   b. Table of contents specifies the location of various parts of the reports. Management can quickly refer to the section that concerns them.
   c. Explanation of purpose and scope of project, reason for doing feasibility study, and name of departments involved in feasibility study.
   d. Methods used for detail finding of present system, checking of system efficiency and effectiveness of system, description of objectives of the system. A discussion of output, reports, costs, and benefits gives management a feel for the pros and cons for the candidate system.
   e. Economic justification details point-by-point cost comparison and preliminary cost estimated for the development and the operation of the candidate system. A return on investment analysis of the project is also included.
   f. Recommendation and suggestion suggest to management the most beneficial and cost effective system.
   g. Appendixes document all memos and data compiled during the investigation. They are presented at the end of the report for the reference.

Que 2.20. Discuss the economic feasibility on the basis of direct cost, indirect cost, tangible cost, and intangible cost.

Answer

1. The cost benefit analysis is necessary to determine the economic feasibility.
2. The basic objective of cost benefit analysis is to find out whether it is economically worthwhile to invest in the project.
3. If the return on the investment is good then the project is considered economically worthwhile.
4. Cost benefits analysis is performed by first listing all costs associated with the project.

i. **Cost :** The cost consists of direct cost/tangible cost and indirect costs/ intangible cost.

   a. **The direct cost/tangible cost :** The direct cost is directly associated with the project such as cost of buying the equipments (Hardware like printer, computers, disk drive etc.,) software, salary to staff, cost involved in preparation of physical location such as air conditioner, lights, wiring etc., operating cost such as use of paper, CD's etc.

   b. **The indirect cost/intangible cost :** The indirect cost is not directly associated with project, it is a result of operations that directly associated with a system (a given system/project). It often referred as over head. Indirect cost involves time spent by user in discussing problems with system analyst, gathering the data about the problems etc.

ii. **Benefits :** The benefits can also be broadly classified as tangible benefits and intangible benefits.

   a. **Tangible benefits :** The tangible benefits are directly measurable. They are :
      1. Decreasing salary cost by automating manual procedure.
      2. Preventing costly but frequent errors.
      3. Sending bills early in the month.
      4. Increasing control over inventory level.
      5. Increase in production.

b. **Intangible benefits :** Intangible benefits are :
   1. Better service to customer.
   2. Superior quality of product.
   3. Upgrading or creating new customer services.
   4. Developing a new image in market.
   5. Reducing repetitive or monotonous work for employees.

PART-2

Ways to Organize Personnel, Project Schedule, Scheduling Objectives, Building the Project Schedule, Scheduling Terminology and Techniques.

CONCEPT OUTLINE : PART-2

- Project scheduling is an inexact process in that it tries to predict the future.
- Some of the project scheduling are :
  a. It provides visibility.
  b. It provides basic structure for reporting information.
  c. It aids in scheduling risk analysis.
- Some of terminology and techniques are activity, level of effort, project network diagram etc.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 2.21. Write a short note on software project team organization.

Answer

1. Project management is not a one-person operation.
2. It requires a group of individuals dedicated to the achievement of a specific goal.
3. Project management includes :
   a. A project manager
   b. An assistant project manager
   c. A project (home) office
   d. A project team
4. Generally, project office personnel are assigned full-time to the project and work out of the project office, whereas the project team members work out of the functional units and may spend only a small percentage of their time on the project.
5. Normally, project office personnel report directly to the project manager.

6. A project office usually is not required on small projects, and sometimes the project can be accomplished by just one person who may fill all of the project office positions.
7. Before the staffing function begins, five basic questions are usually considered :
   i. What are the requirements for an individual to become a successful project manager ?
   ii. Who should be a member of the project team ?
   iii. Who should be a member of the project office ?
   iv. What problems can occur during recruiting activities ?
   v. What can happen downstream to cause the loss of key team members ?

Que 2.22. What consistency, respect, inclusion, and honesty are factors that contribute to effective people management ? Discuss.

UPTU 2012-13, Marks 10

Answer

1. The people working in a software organization are its greatest assets.
2. In successful companies and economies, this is achieved when people are respected by organization and are assigned responsibilities that reflect their skills and experience.
3. Team members should pay some attention to consistency, respect, inclusion and honesty.
4. These are the key factors contributed towards effective people management :
   a. **Consistency :** People in a project team should all be treated in a comparable way. No one expects all rewards to be identical but people should not feel that their contribution to the organization is undervalued.
   b. **Respect :**
      i. Respect is right evaluation or showing an appropriate behaviour to the object of respect.
      ii. It is a feeling of admiration to someone because of their qualities and achievements.
      iii. It is a key human value which is a genuine expression of regard to a person. It helps to succeed in life.
      iv. Different people have different skills and managers should respect these differences.

v. In some cases, people do not fit into a team and they cannot continue, it is important not to jump to conclusions about this at an early stage in the project.

c. **Inclusion :** People contribute effectively when they feel that others listen to them and take account of their proposals. It is important to develop a working environment where all views, even though of junior staff, are considered.

d. **Honesty :**
   i. A manager should always be honest about what is going well and what is going bad in the team.
   ii. Also should be honest about level of technical knowledge and willing to refer to staff with more knowledge when necessary.
   iii. If try to cover up ignorance or problems, we will eventually be found out and will lose the respect of the group.

Que 2.23. **What do you mean by project schedule ? Write down steps in building the project schedule.**

OR

Discuss various steps for building a project schedule with an example.

Answer

Project schedule :

1. Scheduling is an inexact process in that it tries to predict the future.
2. While it is not possible to know with certainty how long a project will take, there are techniques that can increase the likelihood of being close.
3. If we are close in our planning and estimating, we can manage the project to achieve the schedule by accelerating some efforts or modifying approaches to meet required deadlines.
4. Building the project schedule is a complex activity.
5. Basically there are five key processes for developing a project schedule.
6. They are as follows :
   a. **Define activities :**
      i. The goal of the activity definition step is to identify all the tasks required to accomplish the product.
      ii. This frequently results in identifying all the work products and deliverables that comprise the project.
      iii. These deliverables are found as the components of a Work Breakdown structure (WBS).
      iv. The project schedule further decomposes these deliverables into the actual activities required to complete the work.

   b. **Sequence activities :**
      i. In this step, design the sequence of activities with dependencies required to complete the project.
      ii. During this step, one will identify any dependencies of related tasks and document them in the project schedule.
      iii. One will need to analyze each of the tasks to understand which task has a dependency on additional tasks.
      iv. Dependencies relationships must include finish-to-start and start-to-finish dependencies.
   c. **Estimate activity resources :**
      i. The next step is to identify the resources and their availability to our project.
      ii. Remember that not all team members will be 100% available to our project as some team members will be working on multiple projects.
      iii. In this step, one will also assign resources to each of the tasks.
   d. **Estimate activity durations :**
      i. With resources assigned, the next step is to estimate each task's duration.
      ii. The activity's duration is the number of working periods required to complete the task.
      iii. Selecting the correct duration type impacts the resource availability and the forecasted task end date.
   e. **Develop schedule :**
      i. The last step is to analyze the project schedule and examine the sequences, durations, resources and inevitable scheduling constraints.
      ii. The goal of this step is to validate the project schedule which correctly models the planned work.
      iii. In this step one will not only validate the duration estimates are accurate, but validate the resource allocations are correct.

Que 2.24. **What are the basic objectives of scheduling ?**

Answer

The basic objectives of scheduling are as follows :

1. It is the basis for all planning and predicting and help management decide how to use its resources to achieve time and cost goals.
2. It provides visibility and enables management to control "one-of-a-kind" programs.
3. It helps management to evaluate alternatives by answering such questions as how time delays will influence project completion, where slack exists between elements, and what elements are crucial to meet the completion date.

4. It provides a basis for obtaining facts for decision-making.
5. It utilizes a so-called time network analysis as the basic method to determine manpower, material, and capital requirements, as well as to provide a means for checking progress.
6. It provides the basic structure for reporting information.
7. It reveals interdependencies of activities.
8. It facilitates "what if" exercises.
9. It identifies the longest path or critical paths.
10. It aids in scheduling risk analysis.

Que 2.25. Discuss the various terms used in scheduling and techniques.

Answer

Terminology used in scheduling and techniques are discussed as follows :

1. **Activity :** An element of work performed during the course of a project. (Normally have duration, cost, and resource requirements).
2. **Baseline :** The original plan plus or minus approved changes.
3. **Arrow Diagram Method (ADM) :**
   a. A network diagramming technique in which activities are represented by arrows.
   b. The tail of the arrow represents the start and the head of the arrow represents the end of the activity.
   c. Activities are connected at points called nodes to illustrate the sequence in which activities are expected to be performed.
   d. Also called Activity-On-Arrow (AOA).
4. **Backward pass :** The calculation of late finish and start dates for the uncompleted portions of all network activities determined by working backwards through the network logic from the project's end date.
5. **Critical activity :** An activity on a critical path.
6. **Critical path :** The series of activities which determines the earliest completion of the project. The critical path is usually defined as those activities with float less than or equal to a specified value (usually zero).
7. **Critical Path Method (CPM) :** A network analysis technique used to predict project duration by analyzing which path has the least amount of scheduling flexibility. Early dates are calculated using a forward pass; late dates are calculated using a backwards pass.
8. **Dummy activity :**
   a. An activity of zero duration used to show a logical relationship in the arrow diagramming method.

   b. Dummy activities are used when logical relationships cannot be completely or correctly described with regular activity arrows.
   c. Dummies are shown graphically as a dashed line headed by an arrow.
9. **Duration (DU) :** The number of work periods (not including holidays and other non-working periods) required to complete an activity or other project element.
10. **Early finish date (EF) :** In the critical path method, the earliest possible date in which the uncompleted portions of an activity or project can complete and can change as the project progresses.
11. **Early start date (ES) :** In the critical path method, the earliest possible date in which the uncompleted portions of an activity or project can start, can change as the project progresses.
12. **Effort :** The number of labour units required to complete an activity or other project element should not be confused with duration.
13. **Event-on-node :** A network diagramming technique in which events are represented by boxes (or nodes) connected by arrows to show the sequence in which the events are to occur.
14. **Float :** The amount of time that an activity may be delayed from its early start without delaying the project finish date. (Also called slack, total float, and path float).
15. **Forward pass :** The calculation of the early start and early finish dates for the uncompleted portions of all network activities.
16. **Free Float (FF) :** The amount of time an activity can be delayed without delaying the early start of any immediately succeeding activities.
17. **Gantt chart :** A graphic display of schedule-related information using bars.
18. **Hammock :** An aggregate or summary activity.
19. **Hanger :** An unintended break in a network path. Hangers are usually caused by missing activities or missing logical relationships.
20. **Lag :** A modification of a logical relationship which directs a delay in the successor task.
21. **Late finish date (LF) :** In the critical path method, the latest possible date that an activity may be completed without delaying a specified milestone (usually the project finish date).
22. **Late start date (SF) :** In the critical path method, the latest possible date that an activity may begin without delaying a specified milestone (usually the project finish date).
23. **Lead :** A modification of a logical relationship which allows an acceleration of the successor task. For example, in a FS relationship with a 10 day lead, the successor can start 10 days prior to the completion of the predecessor.

24. **Level of effort (LOE) :** Support type activity (e.g., vendor or customer liaison) that does not readily lend itself to measurement of discrete accomplishment. Generally characterized by a uniform rate of activity over a specific period of time.
25. **Logical relationship :** A dependency between two project activities or between an activity and a milestone. Four possible types : FS, FF, SS, and SF.
26. **Master schedule :** A summary level schedule which identifies the major activities and milestones.
27. **Milestone :** A significant event in the project, usually completion of a major deliverable.
28. **Milestone schedule :** A summary level schedule which identifies the major milestones.
29. **Path convergence :** In mathematical analysis, the tendency of parallel paths of approximately equal duration to delay the completion of the milestone where they meet.
30. **Precedence Diagram Method (PDM) :** A network diagramming technique in which activities are represented by nodes. Activities are linked by precedence relationships to show the sequence in which the activities are to be performed.
31. **Program Evaluation and Review Technique (PERT) :** An event-oriented network analysis technique used to estimate project duration when there is a high degree of uncertainty with the individual activity duration estimates.
32. **Project network diagram :** Any schematic display of the logical relationships of project activities.
33. **Remaining Duration (RDU) :** The time needed to complete an activity.
34. **Resource leveling :** Any form of network analysis in which start and finish dates are driven by resource management concerns.
35. **Resource-Limited schedule :** It is a project schedule whose start and finish dates reflect expected resource availability. The final project schedule should always be resource limited.
36. **Scheduled Finish date (SF) :** The point in time work was scheduled to finish on an activity. The scheduled finish date is normally within the range of dates delimited by the early finish date and the late finish date.
37. **Scheduled Start date (SS) :** The point in time work was scheduled to start on an activity. The scheduled start date is normally within the range of dates delimited by the early start and late start dates.
38. **Time-Scaled network diagram :** Any project network diagram drawn in such a way that the positioning and length of the activity represents its duration. Essentially, it is a bar chart that includes network logic.

Que 2.26. Describe project schedule with :

a. **Scheduling objectives**
b. **Scheduling terminology**

c. **Scheduling techniques**

UPTU 2013-14, Marks 10

Answer

a. **Scheduling objectives :** Refer Q. 2.24, Page 75C, Unit-2.
b. **Scheduling terminology :** Refer Q. 2.25, Page 76C, Unit-2.
c. **Scheduling techniques :** Refer Q. 2.25, Page 76C, Unit-2.

PART-3

Network Diagrams : PERT, CPM, Bar Charts : Milestone Charts and Gantt Charts.

CONCEPT OUTLINE : PART-3

- PERT chart is a project management tool used to schedule, organize and coordinate tasks within a project.
- CPM is similar to PERT chart and act as both for preparation of schedule and resource planning.
- A milestone chart shows a group of milestones in an organized way.
- Gantt charts are useful for monitoring the progress of a project which is underway.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 2.27. What do you mean by activity networks ?

Answer

1. The foundation of this approach originated from the Special Projects Office of the US Navy in 1958.
2. They developed this technique for evaluating the performance of large development projects.
3. The technique can be broken down into three stages :
   a. **Planning :** It identifies tasks and estimate duration of times.
   b. **Scheduling :** Establish time table of start and finish times.
   c. **Analysis :** Establish the float and evaluate and revise as necessary.
4. The activity network of tasks, needed to complete project, showing the order in which the tasks need to be completed and the dependencies between them.

5. This is represented graphically, which is shown in Fig. 2.27.1.

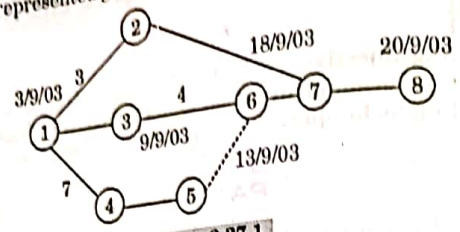


Fig. 2.27.1.

6. The diagram consists of a number of circles, representing events within the development life cycle, such as the start or completion of a task, and lines, which represent the task themselves.
7. Each task is additionally labeled by its time duration.
8. Thus the task between events 4 and 5 is planned to take 3 time units.
9. The primary benefit is the identification of the critical path.
10. In critical path, total time for activities on this path is greater than any other path through the network (delay in any task on the critical path leads to a delay in the project).
11. The degree of difficulty in developing a plan is usually a function of the number of activities or tasks, their sequence, their timing, and the complexity.
12. For small projects, the plan may consist of only a simple flow diagram, bar charts, or computer spreadsheets.
13. For larger projects, there are a number of tools that are used to develop plans.
14. There are two types of activity networks diagram. They are :
    a. Program Evaluation and Review Techniques (PERT)
    b. Critical Path Method (CPM)

Que 2.28. **An activity network for a software project consisting of 8 events, 12 activities, and 1 dummy activity is shown in the Fig. 2.28.1.**

UPTU 2012-13, Marks 10

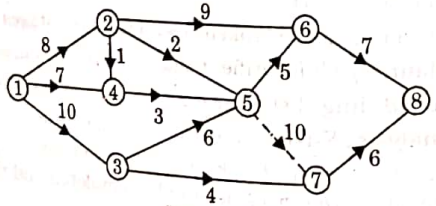


Fig. 2.28.1.

a. Find the critical path for the above.
b. Calculate slack for each activity.
c. Find earliest and latest start time.
d. Find the earliest and latest finish time.

Answer

Forward calculations :

E1 = 0
E2 = 8
E3 = 10
E4 = max (8 + 1, 0 + 7) = 9
E5 = max (8 + 2, 9 + 3, 10 + 6) = 16
E6 = max (16 + 5, 8 + 9) = 21
E7 = max (10 + 16, 10 + 4) = 26
E8 = max (6 + 26, 7 + 21) = 32

Backward calculations :

L8 = 32
L7 = 32 – 6 = 26
L6 = 32 – 7 = 25
L5 = min (25 – 5, 26 –10) = 16
L4 = 16 – 3 =13
L3 = min (16 – 6, 26 – 4) = 10
L2 = min (13 – 1, 16 – 2, 25 – 9) = 12
L1 = min (12 – 8, 13 – 7, 10 – 10) = 0

Task	Normal	Earliest Time		Latest Time		
	Time	Start (E_i)	Finish (E_j)	Start (L_i)	Finish (L_j)	Slack
(1,2)	8	0	8	4	12	4
(1,4)	7	0	7	6	13	6
(1,3)	10	0	10	0	10	0
(2,4)	1	8	9	12	13	4
(2,5)	2	8	10	14	16	6
(2,6)	9	8	17	16	25	8
(3,5)	6	10	16	10	16	0
(3,7)	4	10	14	22	26	12
(4,5)	3	9	12	13	16	4
(5,6)	5	16	21	20	25	4
(5,7)	10	16	26	16	26	0
(6,8)	7	21	28	25	32	4
(7,8)	6	26	32	26	32	0

Critical Path :
(1,3) ⇒ (3,5) ⇒ (5,7) ⇒ (7,8)

Que 2.29. **Write a short note on PERT.**

Answer

1. A Project (or program) Evaluation and Review Technique (PERT) chart is a project management tool used to schedule, organize, and coordinate tasks within a project.
2. PERT can be both a cost and a time management systems.
3. PERT is organized by events and activities or tasks.
4. PERT charts depict task, duration, and dependency information.
5. Each chart starts with an initiation node from which the first task, or tasks, originates.
6. If multiple tasks begin at the same time, they are all started from the node or branch, or fork out from the starting point.
7. Each task is represented by a line, which states its name or other identifier, its duration, the number of people assigned to it, and in some cases the initials of the personnel assigned.
8. The other end of the task line is terminated by another node, which identifies the start of another task, or the beginning of any slack time, that is, waiting time between tasks.

Steps in drawing a PERT chart :

1. Make a list of the project tasks.
2. Assign a task identification letter to each task.
3. Determine the time duration for each task.
4. Draw the PERT network, number each node, label each task with its task identification letter, connect each node from start to finish, and put each task's duration on the network.
5. Determine the need for any dummy tasks.
6. Determine the earliest completion time for each task node.
7. Determine the latest completion time for each task node and verify the PERT network for correctness.

The benefits of PERT are as follows :

1. The PERT network is continuously useful to project managers prior to and during a project.
2. The PERT network is straightforward in its concept and is supported by software.
3. The PERT network's graphical representation of the project's tasks help to show the task interrelationships.
4. The use of the PERT network is applicable in a wide variety of projects.
5. PERT is a scheduling tool that also shows graphically which tasks must be completed before other tasks begins.

6. By displaying the various task paths, PERT enables the calculation of a critical path.
7. PERT controls time and costs during the project and also facilitates finding the right balance between completing a project on time and completing it within the budget.
8. It exposes all possible parallelism in the activities and thus helps in allocating resources.
9. It allows scheduling and simulation of alternative schedules.

Limitations of PERT :

1. In order of the PERT network to be useful, projects tasks have to be clearly defined as well as their relationships to each other.
2. The PERT network does not deal very well with task overlap. PERT assumes the following tasks begin after their preceding tasks end.
3. The PERT network is only as good as the time estimates that entered by the project manager.
4. PERT does not help in deciding which activities are necessary or how long each will take.

Que 2.30. Write a short note on CPM.

OR

Discuss the concept of PERT / CPM schedule along with example.

UPTU 2014-15, Marks 05

Answer

PERT : Refer Q. 2.29, Page 81C, Unit-2.

CPM :

1. Critical Path Method (CPM) charts are similar to PERT charts and are sometimes known as PERT/CPM. CPM acts as the basis both for preparation of a schedule, and of resource planning.
2. They were developed in the 1950s to control large defense projects, and have been used routinely since then.
3. During management of a project, it allows to monitor the achievements of project goals. It also helps to see where remedial action needs to be taken to get a project back on course.
4. In a CPM chart, the critical path is indicated. Critical path is the path of longest duration as determined on a project network diagram.
5. The critical path determines the total duration of the project. If a task on the critical path is delayed, the final completion of the project will likely to be delayed.
6. The critical path is "critical" because tasks that follow a critical task cannot be started until all of the previous tasks on the critical path are completed.

7. Thus, if a task on the critical path is delayed, all tasks following the delayed critical task will be pushed out in time.
8. The critical tasks will have starting and finishing times, that is, fixed relative to the start of the project.
9. Tasks not on the critical path will usually have some flexibility relative to when they can start and finish.
10. This flexibility is called "float", or sometimes "slack". Float is the difference between the time available for performing a task and time required to complete a task.

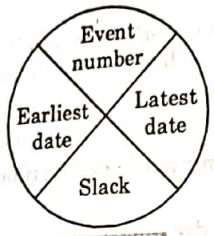


Fig. 2.30.1.

11. The most important aspect of it is the ability to identify those activities that can be done at the same time.
12. Critical path analysis is an effective and powerful method of assessing :
    a. What tasks must be carried out ?
    b. Where parallel activity can be performed ?
    c. The shortest time in which you can complete a project.
    d. Resources needed to execute a project.
    e. The sequence of activities, scheduling and timings involved.
    f. Task priorities.
    g. The most efficient way of shortening time on urgent projects.

Benefits of CPM :

a. It identifies the task that must be completed on time for the whole project to be completed on time.
b. It also identifies which tasks can be delayed for a while if resource needs to be reallocated to catch up on missed tasks.
c. CPM helps to minimize cost.

Limitations of CPM :

a. The relation of tasks to time is not as immediately obvious as with Gantt charts.
b. These are more difficult to understand.

Que 2.31. Write a note on milestone chart.

Answer

1. A milestone is used to represent groups of activities or significant events or commitments in the project.
2. A milestone chart shows a group of milestones in an organized way similar to a gantt chart with one milestone per line vertically with a description on the left and the milestone located horizontally along a time scale showing when it occurs.
3. Milestones differ from the bars in a gantt chart in that they show only a single date and are usually depicted as a triangle instead of a bar.
4. Milestones can be shown in various colors depicting the status of the milestone.
5. Milestones can also appear on gantt charts; project management software supports the placement of milestones on gantt charts and other project reports and displays generated by the software.
6. The milestone chart was devised to save space on the project manager supervisor's walls. Each project manager collected related groups of activities in the project and assigned a milestone to each group.
7. A milestone was placed on the project schedule representing the group. Another milestone was placed on the project manager supervisor's milestone chart as well.
8. If there were changes in the schedule that affected the completion date of the milestone, the project manager had to visit the supervisor's office and move the milestone.
9. Milestone schedules can be produced using today's project management software.
10. They are created simply by listing the milestones as activities and giving them duration of zero.
11. Since, they are being created on a gantt chart, the length of the gantt schedule bar for the milestone would have a zero length and could not be seen.
12. A triangle or another symbol is put on the chart instead. The symbol can be coloured to show various statuses and conditions as needed.

Advantages of milestone chart :

1. Controlling can be easily done and inter relationships between other similar activities can be easily established.
2. These points are those that can be easily identified over the main bar.
3. Milestones are key events of a main activity represented by a bar.
4. These specific points in time that mark the completion of certain portions of the main activity.

5. If the activity is broken or sub divided into a number of sub-activities, each one of which can be easily recognized during the progress of the project.
6. The beginning and end of these sub-divided activities or tasks are termed as milestones.

Disadvantages of milestone chart :

The milestone chart's disadvantages include the following :

1. When used separately from a detailed schedule with activity dependencies, it is difficult to understand how to reach a milestone. This is especially true when the chart includes many milestones.
2. As the number of milestones grows, the chart loses its appeal. By being overcrowded, it may become ineffective in managing the work, thereby defeating its own purpose.
3. Coupling it with a schedule with activity dependencies may be the best option to mitigate risks associated with detailed milestone charts.

Que 2.32. Explain Gantt chart.

Answer

1. A Gantt chart is a horizontal bar chart developed as a production control tool in 1917 by Henry L. Gantt, an American engineer and social scientist.
2. Frequently used in project management, a Gantt chart provides a graphical.
3. Gantt charts are useful tools for planning and scheduling projects.
4. They allow to assess how long a project should take, determine the resources needed, and lay out the order in which tasks need to be carried out.
5. They are useful in managing the dependencies between tasks.
6. When a project is under way, Gantt charts are useful for monitoring its progress.
7. We can immediately see what should have been achieved at a point in time, and can therefore take remedial action to bring the project back on course.
8. This can be essential for the successful and profitable implementation of the project.
9. To draw up a Gantt chart, it must follow these steps :

   Step 1 : List all activities in the plan. For each task, show the earliest start date, estimated length of time it will take, and whether it is parallel or sequential. If tasks are sequential, show which stages they depend on.

   Step 2 : Head up graph paper with the days or weeks through to task completion.

Step 3 : Plot the tasks onto the graph paper. Next draw up a rough draft of the Gantt chart. Plot each task on the graph paper, showing it starting on the earliest possible date. Draw it as a bar, with the length of the bar being the length of the task. Above the task bars, mark the time taken to complete them.

Step 4 : Schedule activities. Now take the draft Gantt chart, and use it to schedule actions. Schedule them in such a way that sequential actions are carried out in the required sequence. Ensure that dependent activities do not start until the activities they depend on have been completed. While scheduling, ensure that we make best use of the resources we have available, and do not over-commit resource.

Step 5 : Presenting the analysis. The final stage in this process is to prepare a final version of the Gantt chart. This should combine the draft analysis with our scheduling and analysis of resources. This chart will show when we anticipate that jobs should start and finish.

Advantages of gantt chart :

1. This is a simple and very inexpensive method and can be developed even by supervisory staff with some amount of training.
2. These charts clearly show the decided time and work schedules for every job.
3. Monitoring and control are easier and can be done within a minimum time frame and at the lowest cost.
4. These charts can be changed and updated quickly at a lower cost.
5. There is no need to develop the customized Gantt chart boards as the standard chart boards are available in the market.

Disadvantages of gantt chart :

In spite of the above-mentioned advantages, there are certain disadvantages.

1. They do not show job interrelationships and interdependence.
2. Cost implications cannot be shown.
3. With these charts, it is not possible to depict other alternatives for project completion.
4. The shape and form of Gantt charts can differ according to the nature of the requirement.

Que 2.33. Differentiate between :

a. PERT and CPM

b. Gantt chart and Milestone chart **UPTU 2013-14, Marks 10**

Answer

a. PERT and CPM :

S.No.	PERT	CPM
1.	PERT uses event oriented network.	CPM uses activity oriented network.
2.	Estimate of time for activities is not so accurate and definite.	Durations of activity may be estimated with a fair degree of accuracy.
3.	It is used mostly in research and development projects, particularly projects of non-repetitive nature.	It is used extensively in construction projects.
4.	Probabilistic model concept is used.	Deterministic concept is used.
5.	PERT is basically a tool for planning.	CPM can control both time and cost when planning.
6.	In PERT, it is assumed that cost varies directly with time. Attention is therefore given to minimize the time so that minimum cost results. Thus in PERT, time is the controlling factor.	In CPM, cost optimization is given prime importance. The time for the completion of the project depends upon cost optimization. The cost is not directly proportional to time. Thus, cost is the controlling factor.

b. Gantt chart and milestone chart :

S.No.	Gantt chart	Milestone
1.	It is used to depict key events along a timescale graphically.	It is used to represent the timing of various tasks that are required to complete a project.
2.	It focuses mainly on the end-dates by which something needs to be complete or by which certain objectives need to be achieved.	It focuses more on the activities to be carried out to complete the project.

Que 2.34. Explain various advantages of Milestone charts and Gantt charts.

UPTU 2014-15, Marks 05

Answer

Advantages of milestone charts : Refer Q. 2.31, Page 85C, Unit-2.
Advantages of gantt charts : Refer Q. 2.32, Page 86C, Unit-2.

VERY IMPORTANT QUESTIONS

Following questions are very important. These questions may be asked in your SESSIONALS as well as UNIVERSITY EXAMINATION.

Q. 1. What do you mean by work breakdown structure ? Also, discuss its types and features.

Ans. Refer Q. 2.2, Q. 2.3, and Q. 2.6.

Q. 2. Write short note on :
i. Project life cycle
ii. Product life cycle

Ans. Refer Q. 2.9.

Q. 3. Explain the following terms :
i. Project schedule
ii. Scheduling objectives
iii. Scheduling terminology and techniques.

Ans.
i. Refer Q. 2.23.
ii. Refer Q. 2.24.
iii. Refer Q. 2.25.

Q. 4. Discuss the activity network, PERT and CPM.

Ans. Refer Q. 2.27, Q. 2.29. and Q. 2.30.

Q. 5. Write short note on Gantt and Milestone charts.

Ans. Refer Q. 2.31 and Q. 2.32.

☺☺☺

3 UNIT

Project Monitor and Co

Part-1 ..... (9)

- *Dimensions of Project Monitoring and Control*
- *Earned Value Analysis*
- *Earned Value Indicators*
- *Budgeted Cost for Work Scheduled (BCWS)*
- *Cost Variance (CV)*
- *Schedule Variance (SV)*
- *Cost Performance Index (CPI)*
- *Schedule Performance Index (SPI)*
- *Interpretation of Earned Value Indicators*

Part-2 ..... (10)

- *Error Tracking*
- *Software Reviews*
- *Types of Reviews : Inspections*
- *Deskchecks*
- *Walkthroughs*
- *Code Reviews*
- *Pair Programming*

PART-1

Dimensions of Project Monitoring and Control, Earned Value Analysis, Earned Value Indicators : Budgeted Cost for Work Scheduled (BCWS), Cost Variance (CV), Schedule Variance (SV), Cost Performance Index (CPI), Schedule Performance Index (SPI) and Interpretation of Earned Value Indicators.

CONCEPT OUTLINE : PART-1

- Earned value analysis looks at three basic parameters :
  a. Planned value
  b. Earned value
  c. Actual cost
- BCWP is the budgeted cost of work that has actually been performed in carrying out a scheduled task during specific time period.
- BCWS is approved budget that has been allocated to complete a scheduled task during specific time period.
- ACWP is actual cost that has been spent, rather than budgeted cost.
- Cost variance represents the algebraic difference between earned value of project and actual cost of project.
- Schedule variance is used by project management personnel to determine schedule performance during or after the completion of project.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 3.1. What do you mean by project monitoring and control process ? Explain. How these process affect the project schedule ? Discuss. **UPTU 2012-13, Marks 10**

Answer

1. The purpose of project planning is to identify the scope of the project, estimate the work involved, and create a project schedule.
2. The main goal of monitoring for project managers is to get visibility into the project execution so that they can determine whether any action needs to be taken to ensure that the project goals are met.

3. Different types of monitoring might be done for a project.
4. The three main levels of monitoring are activity level, status reporting and milestone analysis.
5. Measurements taken on the project are employed for monitoring.
6. Activity-level monitoring ensures that each activity in the detailed schedule has been done properly and within time.
7. This type of monitoring may be done daily in project team meetings or by the project manager checking the status of all the tasks scheduled to be completed on that day.
8. A completed task is often marked as 100 % complete in detailed schedule this is used by tools like the Microsoft project to track the percentage completion of the overall project or a higher level task.
9. Status reports are often prepared weekly to take stock of what has happened and what needs to be done.
10. Status reports typically contain a summary of the activities successfully completed since the last status report, any activities that have been delayed, any issues in the project that need attention, and if everything is in place for the next week.
11. The milestone analysis is done at each milestone or every few weeks, if milestones are too far apart.
12. If there is some deviation from planned activity then task of control start.
13. So, the project must complete within time and cost limit.
14. The project control and monitoring activity can be represented as shown in Fig. 3.1.1.

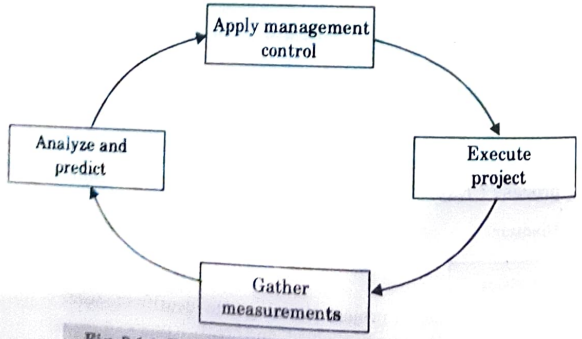


Fig. 3.1.1. The project monitoring and control cycle

Project monitoring and control process affect the project schedule :

15. Monitoring and control process are used to :
    a. Monitor the progress of projects.
    b. Assess the risk of slippage.
    c. Visualize and assess the state of project.
    d. Revise targets to correct or counter act drift.
    e. Control changes to a project's requirements.

Progress assessment will be made on the basis of information collected at regular intervals or when specific event occurs. This information will be objective and tangible. Progress assessment will have to rely on the judgment of the team members who are carrying out project activities.

Que 3.2. Why project monitoring and control is needed for successful implementation of a software project ?

Answer

1. Creating and maintaining the project plan is an important part of project leadership. However, constantly monitoring a project's "health" and regularly controlling a project's overall status is vital for ensuring the successful implementation of the project plan.
2. There are three major reasons for this :
    a. Many project managers are only trained in project planning and simply lack the skill or at least the "awareness" that monitoring and controlling is at least as important as planning.
    b. Project monitoring and controlling involve some additional effort not everyone is committed to invest.
    c. Most project management software packages only support the planning phase, but provide little additional value for project monitoring or controlling.
3. Nevertheless, constantly monitoring a project is important. And it is even more important to define a simple controlling cycle.
4. When the project's performance deviates significantly from the plan so that appropriate corrective actions and preventive actions will be taken.
5. Project activity monitoring is an aspect of project management that is performed throughout the project. Controlling is the aspect of the project in which corrective and preventive actions are taken.
6. It falls to the project manager to ensure that the combined monitor and control process is effectively executed. Effective execution of the project monitoring and controlling process leads to successful project delivery.

Que 3.3. What do you understand by earned value analysis and management ? Explain with example. Why many individuals and firms have failed to adapt the methods under earned value analysis and management ? Discuss. **UPTU 2012-13, Marks 10**

Answer

Earned value analysis :

1. Earned Value Analysis (EVA) was developed by the US Department of Defense to determine the performance of large military procurement contracts.
2. Its techniques can still be applied to the smaller projects currently in use today.
3. EVA looks at three basic parameters :
   a. How much work did you plan to complete ? (Planned Value)
   b. How much work did you actually complete ? (Earned Value)
   c. How much did it cost to complete the work ? (Actual Cost)
5. By comparing these parameters, an objective assessment of cost and schedule performance can be gained.
6. Instead of simply concentrating on how much time has been taken to achieve progress, earned value looks at how much value has been achieved so far.

Earned value management :

1. Earned Value Management (EVM) has proven itself to be one of the most effective performance measurement and feedback tools for managing projects.
2. It enables managers to close the loop in the plan-do-check-act management cycle.
3. If the application of EVM to a project reveals that the project is behind schedule or over budget, the project manager can use the EVM methodology to help identify :
   a. Where problems are occurring.
   b. Whether the problems are critical or not.
   c. What it will take to get the project back on track.
4. EVM provides organizations with the methodology needed to integrate the management of project scope, schedule, and cost.
5. EVM can play a crucial role in answering management questions that are critical to the success of every project, such as :
   a. Are we ahead of or behind schedule ?
   b. How efficiently are we using our time ?
   c. When is the project likely to be completed ?
   d. Are we currently under or over our budget ?
   e. How efficiently are we using our resources ?
   f. What is the remaining work likely to cost ?
   g. What is the entire project likely to cost ?
   h. How much will we be under or over budget at the end ?
6. EVM strategically augments good project management to facilitate the planning and control of cost and schedule performance.
7. The key practices of EVM include :
   a. Establish a performance measurement baseline (PMB)
      i. Decompose work scope to a manageable level.
      ii. Assign unambiguous management responsibility.
      iii. Develop a time-phased budget for each work task.
      iv. Select EV measurement techniques for all tasks.
      v. Maintain integrity of PMB throughout the project.
   b. Measure and analyze performance against the baseline
      i. Record resource usage during project execution.
      ii. Objectively measure the physical work progress.
      iii. Credit EV according to EV techniques.
      iv. Analyze and forecast cost/schedule performance.
      v. Report performance problems and/or take action.

Many individuals and firms have failed to adapt the methods under earned value analysis and management because :

a. The first stage in setting up an earned value analysis is to create the baseline budget.
b. The baseline budget is based on the project plan and shows the forecast growth in earned value through time.
c. Earned value may be measured in monetary values but, in case of staff-intensive projects such as software development, it is common to measure earned value in person-hours or work days.

Que 3.4. What are the essentials of EVM ?

Answer

The essentials of EVM are as follows :

1. A clear set of well defined, accomplishable and measurable deliverables.
2. A clear set of standards to judge whether a deliverable has been accomplished.
3. An ability to know when and how work can be divided among multiple persons.
4. A method of determining parallel and serial deliverable accomplishment efforts.
5. An ability to know all the different staff and skills necessary for deliverable accomplishment.

6. A clear set of well defined, accomplishable, and measurable tasks that result in accomplished deliverables.
7. A clear set of staff hour, unit-effort-based, allocations to accomplish deliverables via tasks.
8. A strategy to assess staff and quantify the velocity at which they work, in order to accomplish deliverables.
9. An enumeration of work environment factors that affect the rate of deliverable accomplishment.
10. A strategy to quantify and assign work environment factors to individual deliverable accomplishments.

Que 3.5. Show the relationship among earned value key parameters, earned value performance measures and earned value forecasting indicators.

Answer

Fig. 3.5.1 shows the relationship among earned value key parameters, earned value performance measures and earned value forecasting indicators.

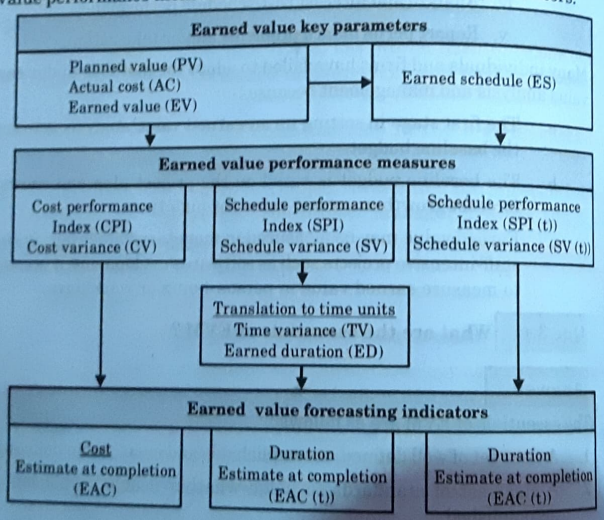


Fig. 3.5.1.

Que 3.6. What do you mean by earned value analysis and earned value indicators ? Discuss various earned value indicators with examples.

UPTU 2014-15, Marks 05

Answer

Earned value analysis : Refer Q. 3.3, Page 93C, Unit-3.

Earned value indicators : Earned value indicators are the indicators which defines something about the performance compared to the plan.

Various earned value indicators are :

1. **Budgeted Cost of Work Performed (BCWP) :** Budgeted cost of work performed (BCWP) or "Earned Value" (EV), in project management is the budgeted cost of work that has actually been performed in carrying out a scheduled task during a specific time period.
2. **Budgeted Cost of Work Scheduled (BCWS) :** It is the approved budget that has been allocated to complete a scheduled task during a specific time period.
3. **Actual Cost of Work Performed (ACWP) :** It is the actual cost that has been spent, rather than the budgeted cost.
4. **Cost Variance (CV) :**
   a. The term cost variance, also known by the abbreviation of CV, refers specifically to the true measurement of cost performance on a particular project.
   b. The cost variance represents the algebraic difference between the earned value of a project (also known by the abbreviation of EV), and the actual cost of the project (also known by the abbreviation AC).

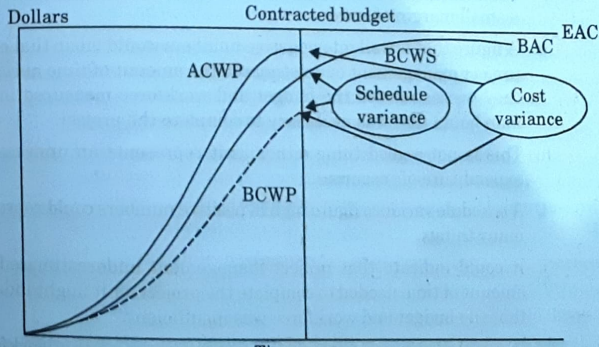


Fig. 3.6.1.

   c. The equation to determine the cost variance would be broken down as follows : CV = EV minus AC.
   d. If the resulting value for the cost variance is a number greater than zero (or "positive value"), then it is considered to be a favourable cost variance condition.

e. A value that is less than zero (or a resulting "negative" value) represents a cost variance that is considered less than favourable.

f. Because the cost variance is so dependent on the earned value of the actual cost, in order to maintain a favourable cost variance it is advantageous for the project to minimize actual costs to the extent possible.

CV = BCWP – ACWP and CV = EV – AC

5. **Schedule Variance (SV) :**

a. Schedule variance is a quantitative measure used by project management personnel to determine schedule performance during or after the completion of a project.

b. It is calculated using a simple algebraic equation where the earned value (EV) represents the actual amount of time taken to either complete the project or progress to the project's current stage.

c. The planned value (PV) represents the amount of time which reaching the project's current progress should have taken to achieve according to the project management's schedule.

d. Schedule variance (SV) is found by subtracting PV from EV. (EV – PV = SV).

e. Schedule variance and its exact number may indicate many possible things to project management.

f. A number approaching zero would indicate that the scheduling and timeframes generated by project management were accurate within a small margin of error.

g. A figure that is well into negative numbers would mean that either project management overestimated the amount of time needed or they overestimated the budget and workforce measured in raw man hours that was necessary to complete the project.

h. This is not a good thing either as it represents an unnecessary expenditure of resources.

i. A schedule variance figure high in positive numbers could represent many things.

j. It could indicate that project management underestimated the amount of time needed to complete the project, or it might indicate that the budget and workforce was insufficient.

k. It could also mean that project management or the workforce suffered setbacks, foreseen or otherwise, which may or may not have been avoidable. It is given by formula

SV= EV – PV

6. **Cost Performance Index :**

a. The cost performance index, also referred by the abbreviation (CPI), refers specifically to a method, chart, or other instrument that is

implemented for the purposes of determining/measuring the actual cost efficiency of a project.

b. The cost performance index is determined by measuring the ratio of earned value (also known by the abbreviation of EV) to actual costs (also known by the abbreviation of AC).

c. The equation to determine the cost performance index can be derived by the following equation. CPI = EV divided by AC.

d. If the resulting value is greater than one indicates that the conditions of cost efficiency for the project are considered to be favourable.

e. A resulting value that is less than one indicates that the conditions of cost efficiency for the project are considered to be less than favourable.

f. The cost performance index can change over the life of a project depending on the ways in which the earned values and actual cost have changed.

g. This can also be shown by a simple formula,

CPI = EV/ACWP

7. **Schedule Performance Index (SPI) :**

a. The schedule performance index is a measure of project efficiency given by project management to gauge the progress and efficiency.

b. A schedule performance index score of 1 or greater is an optimum goal since it shows the project management that the project is on track and has favourable conditions of meeting the required goals.

c. However, a schedule performance index less than 1 is to be avoided since that shows the project is not meeting goals and is showing unfavourable conditions that could lead to project failure if the current course of action is allowed to continue.

d. If the schedule performance index showing a trend that is at or approaching 1, the project management will reevaluate the current conditions of the project and begin an analysis of the current project trends and begin corrective actions.

e. If the schedule performance index trend is rising, the project management will analyze the goals and the current favourable conditions to possibly re-assess the project's short term goals.

f. The schedule performance index is a ratio of Earned Value (EV) to the Planned Value (PV).

g. Earned Value is the value of the project at its current timeframe. Planned Value is the overall projected value of the project at the same time as the Earned Value.

h. To determine the project's schedule performance index the project management divides the EV by the PV. This can also be shown by a simple formula :

SPI = EV/PV.

Que 3.7. **Suppose you are managing a software development project. The project is expected to be completed in 8 months at a cost of $10000 per month. After 2 months, you realize that the project is 30% completed at a cost of $40,000. You need to determine whether the project is on-time and on-budget after 2 months. Let's see how healthy the project is by calculating the cost variance and schedule variance.**

Answer

We can solve the problem in two steps as follows :

Step1 : Calculate the Planned Value and Earned Value

From the scenario :

1. Budget at Completion (BAC) = $ 10,000 * 8 = $ 80,000
2. Actual Cost (AC) = $ 40,000
3. Planned Completion = 2/8 = 25 %
4. Actual Completion = 30 %
5. Therefore,
   a. Planned Value = Planned Completion (%) * BAC = 25% * $ 80,000 = $ 20,000
   b. Earned Value = Actual Completion (%) * BAC = 30 % * $ 80,000 = $ 24,000

Step 2 : Compute the earned value management cost and schedule variances :

1. Cost Variance = EV – AC = $ 24,000 – $ 40,000 = – $ 16,000
2. Schedule Variance = EV – PV = $ 24,000 – $ 20,000 = $ 4,000

Que 3.8. **Differentiate between Cost Variance (CV) and Schedule Variance (SV).** **UPTU 2014-15, Marks 05**

Answer

Cost variance (CV) : Refer Q. 3.6, Page 96C, Unit-3.

Schedule variance (SV) : Refer Q. 3.6, Page 96C, Unit-3.

Que 3.9. **What do you understand by 'Earned Value Analysis' ? Discuss the following indicators :**

1. **Cost Variance (CV)**
2. **Schedule Performance Index (SPI)**
3. **Cost Performance Index (CPI)** **UPTU 2013-14, Marks 10**

Answer

Refer Q. 3.6, Page 96C, Unit-3.

PART-2

Error Tracking, Software Reviews, Types of Review : Inspections, Deskchecks, Walkthrough, Code Reviews and Pair Programming.

CONCEPT OUTLINE : PART-2

- Software review is an effective way of filtering error in a software product.
- Inspections improve the reliability, availability, and maintainability of a software product.
- Code inspection is a process of examining the code of program for identification of certain errors.
- Code review is a phase in the software development process in which the authors of code, peer reviewers, and perhaps quality assurance testers get together to review code.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 3.10. **What do you mean by software technical reviews ?**

Answer

1. Software review is an effective way of filtering errors in a software product.
2. Software review is used as a filter at various points of software development.
3. Reviews conducted at each of these phases, analysis, design, coding, and testing reveal areas of improvement in the product.
4. Reviews also indicate those areas that do not need any improvement.
5. We can use software reviews to achieve consistency and uniformity across products.
6. Reviews also make the task of product creation more manageable.
7. Some of the most common software review techniques, practiced across software organizations include :
   a. Inspection
   b. Walkthrough
   c. Formal technical reviews
   d. Code reviews
   e. Pair programming

Que 3.11. What is software review and formal technical review ?

OR

Explain formal technical review (Peer reviews).

Answer

Software review :

1. The software review are filter or software engineering process *i.e.*, review are applied at various points during software development and serve to uncover errors and defects that can then be removed.
2. Software review "Purifies" the software engineering activities that we have called analysis, design and coding.
3. Many different type of reviews can be conducted as a part of software engineering. Each has its place.
4. An informal meeting around the coffee machine is a form of review, if technical problems are discusses.
5. A formal presentation of software design to an audience of customer's managements and technical staff is also a form of review.

FTR :

a. A formal technical review is software quality assurance activity performed by software engineers (and others).

b. A Formal Technical Review (FTR) is known as walkthrough or an inspection.

c. The objectives of FTR are :

   1. To uncover errors in function, logic or implementation for representation of software.
   2. To verify that software under review meets its requirements.
   3. To ensure that the software has been represented according to predefined standards.
   4. To achieve software that is developed in a uniform manner.
   5. To make projects more manageable.

d. In addition, the FTR serves as a training ground, enabling junior engineers to observe different approach to software analysis, design and implementation.

e. The FTR also serves to promote backup and continuity because a number of people became familiar with parts of the software that they may not have otherwise seen.

Que 3.13. What do you mean by inspection ?

Answer

1. Inspections improve the reliability, availability, and maintainability of a software product.
2. Anything readable that is produced during software development can be inspected.
3. Inspections can be combined with structured, systematic testing to provide a powerful tool for creating defect-free programs.
4. The inspection activity follows a specified process and the participants play well-defined roles.
5. An inspection team consists of three to eight members who play the roles of moderator, author, reader, recorder, and inspector.
6. It also helps to have a client representative participate in requirements specification inspections.
7. Group inspections enable team members to exchange knowledge and ideas during an inspection session.
8. Moderator leads the inspection, schedules meetings, controls meetings, reports inspection results, and follows up on rework issues.
9. Author creates or maintains the work product being inspected.
10. Reader describes the sections of the work product to the team as they proceed through inspection.
11. Recorder classifies and records defects and issues raised during the inspection.
12. All participants play the role of inspectors. However, good inspectors are those who have created the specification for the work product being inspected.
13. For example, the designer can act as an inspector during code inspection while a quality assurance representative can act as standard enforcer.

An error checklist for inspections :

1. An important part of the inspection process is the use of a checklist to examine the program for common errors.
2. The checklist is largely language independent, meaning that most of the errors can occur with any programming language.
3. We may wish to supplement this list with errors peculiar to our programming language.
4. The errors may be
   a. Data Reference errors
   b. Data-Declaration errors
   c. Computation errors
   d. Comparison errors

e. Control Flow errors
f. Interface errors
g. Input-Output errors

Que 3.14. **What do you mean by code inspection ? Which types of errors can be removed by code inspection ?**

Answer

1. Code inspection is done to find out some common types of errors caused due to misunderstanding and improper programming.
2. In other words, code inspection is a process of examining the code of program for identification of certain errors which are not identifiable by code walkthrough.
3. In an inspection, in contrast to a walkthrough, the meeting and the procedure are much more formal.
4. The inspection of a work product is done by a group of peers, who first inspect the product privately and then get together in a formal meeting to discuss potential defects found by individuals and to detect more defects.
5. An inspection can be held for any technical product, which may include requirements specifications, system design document, detailed design, code, and test plan.
6. During identifying errors through code inspection, the standard of coding is also checked.

There are three reasons for having review or inspections :

a. Defect removal.
b. Productivity increase.
c. Provide information for project monitoring.

For code review, the inspection package can consist of the following :

a. Program source listing.
b. Pertinent portions of design or specification document.
c. Pertinent parts of common definitions (e.g., macros and data structures) that are used by the code.
d. Any system constraints.
e. Blank copies of all forms and reports.
f. Checklists to be used for review.

There is a list of some classical programming errors which can be checked during code inspection as given below :

a. Use of un-initialized variables.
b. Jumps within loop (use of transfer control statements (goto) to transfer execution control of program in between the loop i.e., for-loop, while-loop, do-while loop, etc).

c. Non-terminating loops (condition for completion of loop is not given).
d. Mismatched assignment.
e. Array indices out of bounds (size of array is not initialized).
f. Improper storage allocation and de-allocation.

Que 3.15. **What is desk checking ?**

Answer

1. A third human error-detection process is the older practice of desk checking.
2. A desk check can be viewed as a one-person inspection or walkthrough : a person reads a program, checks it with respect to an error list, and/or walks test data through it.
3. Desk checking is relatively unproductive. It is a completely undisciplined process.
4. It runs counter to a testing principle that people are generally ineffective in testing their own programs.
5. For this reason, you could deduce that desk checking is best performed by a person other than the author of the program (e.g., two programmers might swap programs rather than desk check their own programs), but even this is less effective than the walkthrough or inspection process.
6. Desk checking may be more valuable than doing nothing at all, but it is much less effective than the inspection or walkthrough.

Que 3.16. **Write a short note on walkthroughs.**

OR

What is walkthrough ? When we can perform the walkthrough and what are the objectives of walkthrough ?

Answer

1. In a walkthrough, author describes and explains the work product in an informal meeting to his peers or supervisor to get feedback.
2. Here validity of the proposed solution for work product is checked.
3. Structured walkthrough technique is very useful technique to analyze a product for its effectiveness.
4. In design phase of the product, the purpose of walkthrough is to find out as many possible problems in product design while the design is on paper.
5. It is cheaper to make changes when the design is on paper rather than at the time of conversion.
6. Generally walkthrough can be done at any stage of product development as given below :

a. At the time of deciding schedule for different phases
b. At the time of problem specification
c. Designing data structure
d. Program designing
e. Preparing documentation and user manual
f. Coding
g. Test plan, data and result
h. Maintenance changes

7. So, the walkthrough can start in early stage of software development as design and planning, long before the testing begins.
8. It is a static method of quality assurance. Walkthrough are informal meetings but with purpose.
9. **Objectives of walkthrough :** It is one of the methods of review whose purpose is to ensure high quality. Its main objectives are to find :
   a. Bugs, misinterpretation, errors, inconsistencies, and anything that is unclear.
   b. Anything that is complex and difficult to modify.
   c. Any deviation from standard.
10. The purpose of walkthrough is to only find out the problem not to correct them, the correction is the field of developer.
11. Different authorities give different recommendation for optimum number of participant in walkthrough.
12. Generally four participants are taken for walkthrough. This is because the more people there are the more passivity of waste of time due to difference of options.
13. Now who will attend the walkthrough ? Again there is difference in opinion but generally this is decided on the basis of work product being walkthrough.
14. There is general agreement that author of the product, a maintenance expert and a member of quality assurance group should attend the walkthrough.
15. Users can attend specification walkthrough, testing walkthrough, design walkthrough but not the code walkthrough.
16. User involvement in walkthrough may help system developer.

Que 3.17. Write short notes on code review.

UPTU 2012-13, Marks 05

Answer

a. Code review is a phase in the software development process in which the authors of code, peer reviewers, and perhaps quality assurance (QA) testers get together to review code.

b. Finding and correcting errors at this stage is relatively inexpensive and tends to reduce the more expensive process of handling, locating, and fixing bugs during later stages of development or after programs are delivered to users.

c. Reviewers read the code line by line to check for :
   1. Flaws or potential flaws.
   2. Consistency with the overall program design.
   3. The quality of comments.
   4. Adherence to coding standards.

d. Code review may be especially productive for identifying security vulnerabilities. Specialized application programs are available that can help with this process.

e. Automated code reviewing facilitates systematic testing of source code for potential trouble such as buffer overflows, race conditions, memory leakage, size violations, and duplicate statements.

f. Code review is also commonly done to test the quality of patches.

Steps in code review are as follows :

1. Obtain print-outs of the specification and design documents, and of the code. Write comments neatly on the print-outs, with name, date and other relevant details.
2. Read through the specification and design documents to get an understanding of the purpose of the code and how it achieves this purpose.
3. Compare the class hierarchy and/or function call-tree from the design document with the actual code. Note any discrepancies.
4. Identify the important data structures from the design document. Check this against the actual code. Note any discrepancies.
5. Check for adherence to the project's coding standard.
6. Check the style and correctness of each of the following :
   a. each file
   b. each class
   c. each function/method
7. Check the handling of exceptions and errors.
8. Check the user interaction.
9. Look for common errors.

10. Read the test plan. Verify that it checks the software limits.

Que 3.18. **What do you mean by 'Code Review' ? Also, discuss the difference between code inspection and code walkthrough.**

UPTU 2013-14, Marks 10

Answer

Code review : Refer Q. 3.17, Page 106C, Unit-3.

S.No.	Code Inspection	Code Walkthrough
1.	Formal.	Informal.
2.	Initiated by the project team.	Initiated by the author.
3.	Planned meeting with fixed roles assigned to all the members involved.	Unplanned.
4.	Reader reads the product code. Everyone inspects it and comes up with defects.	Author reads the product code and his team mate comes up with defects or suggestions.
5.	Recorder records the defects.	Author makes a note of defects and suggestions offered by team mate.
6.	Moderator has a role in making sure that the discussions proceed on the productive lines.	Informal, so there is no moderator.

Que 3.19. **Write short note on pair programming.**

UPTU 2012-13, Marks 05

Answer

1. All code to be sent into production is created by two people working together at a single computer. Pair programming increases software quality without impacting time to deliver.
2. It is natural, two people working at a single computer will add as much functionality as two working separately except that it will be much higher in quality.
3. With increased quality, comes big savings later in the project. The best way to pair program is to just sit side by side in front of the monitor.

4. Slide the keyboard and mouse back and forth. Both programmers concentrate on the code being written.
5. Pair programming is a social skill that takes time to learn. You are striving for a co-operative way to work that includes give and take from both partners regardless of corporate status.
6. The best pair programmers know when to say "let's try your idea first". Do not expect people to be good at it from the start.
7. It helps if you have someone on your team with experience to show everyone what it should feel like.

The advantages of pair programming are as follows :

1. **Flexibility :** If you pair on everything, then you can contribute to anything. This is a huge advantage.
2. **Code ownership :** If you worked on each piece of the application, then you bear some responsibility when it breaks. If you spread this across the team, problems get solved very quickly.
3. **Collaboration :** Nothing builds team unity like solving problems together. You also get the benefit of gaining your co-workers expertise when you code with them.

Disadvantages of pair programming are :

1. **It ties up two programmers :** It would appear that you are wasting at least one person's time. One of the programmer is not actually programming.
2. **It's inconvenient and reduces flexibility :** Sometimes it is inconvenient. I really enjoy working on something until it is completed. If I have to switch pairs and tasks, it really bugs me that the other task was not finished first. You also have fewer pairs than people, which mean fewer tasks being worked on at the same time.

VERY IMPORTANT QUESTIONS

Following questions are very important. These questions may be asked in your SESSIONALS as well as UNIVERSITY EXAMINATION.

Q. 1. What do you mean by project monitoring and control ?
Ans. Refer Q. 3.1

Q. 2. Discuss earned value analysis.
Ans. Refer Q. 3.3

4 UNIT

Software Quality Assurance and Testing

PART-1

Testing Objectives, Testing Principles, Test Plans, Test Cases, Types of Testing, Level of Testing, Testing Strategies.

CONCEPT OUTLINE : PART-1

- Software testing is the process of executing a program with the intention of finding errors in the code.
- Objectives of software testing are :
  a. Software quality improvement
  b. Verification and validation
  c. Software reliability estimation
- Some important principles of software testing are :
  a. All tests should be traceable to customer requirements.
  b. It is impossible to test everything.
  c. Use effective resources to test.
  d. Test should be planned long before testing begins.
- Test plan is a document describing the scope, approach, resources and schedule of intended testing activities.
- Levels of testing are :
  a. Unit testing
  b. Integration testing
  c. System testing
  d. Acceptance testing

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 4.1. What is software testing ? What are the objectives of software testing ?

Answer

1. Software testing is the process of executing a program with the intention of finding errors in the code.
2. It is a process of evolution of a system or its parts by manual or automatic means to verify that it is satisfying specified requirements or not.
3. Testing is the fundamental of software success.
4. Testing is not a distinct phase in system development life cycle but should be applicable throughout all phases *i.e.*, design development and maintenance phase.

Objectives of software testing : The software testing is usually performed for the following objectives :

1. **Software quality improvement :**
   a. Software quality means the conformance to the specified software design requirements.
   b. The minimum requirement of quality means performing as required under specified circumstances.
   c. Software testing is not only used to remove bugs, but also to find out design defect by the programmer.
2. **Verification and validation :**
   a. The second main objective of software testing is to verify the system and checking its validation.
   b. Verification means to test that we are building the product in right way *i.e.*, we are using the correct procedure for the development of software so that it can meet the user requirements.
   c. Whereas validation is the process which checks that whether we are building the right product or not.
3. **Software reliability estimation :**
   a. Software reliability has important relationship with many aspects of software development.
   b. Its objective is to discover the residual designing errors before delivery to the customer.
   c. The failure data during the testing process are taken down in order to estimate the software reliability.

Que 4.2. State the principles of software testing.

Answer

Principles of software testing :

1. All tests should be traceable to customer requirements.
2. Testing time and resources are limited *i.e.*, avoid redundant test.
3. It is impossible to test everything.
4. Use effective resources to test.
5. Test should be planned long before testing begins *i.e.*, after requirement phase.
6. Test for invalid and unexpected input conditions as well as valid conditions.
7. The probability of existence of more errors in a module or a group of modules is directly proportional to the number of errors already found.
8. Testing should begin "in the small" and progress towards testing "in the large".
9. For the most effective testing, testing must be conducted by an independent third party.

10. Testing should not be planned under the implicit assumption that no error will be found.

Que 4.3. Discuss the basic terms failure, test cases and test suite associated with testing.

Answer

Testing a program consists of providing the program with a set of test inputs (or test cases) and observing if the program behaves as expected. If the program fails to behave as expected, then the conditions under which failure occurs are noted for later debugging and correction.

1. **Test plan :** A test specification is called a test plan. A test plan is documented so that it can be used to verify and ensure that a product or system meets its design specification.
2. **Traceability matrix :** This is a table that correlates or design documents to test documents. This verifies that the test results are correct and is also used to change tests when the source documents are changed.
3. **Failure :** This is a manifestation of an error (or defect or bug). But, the mere presence of an error may not necessarily lead to a failure.
4. **Test case :** This is triplet [I, S, O], where I is the data input to the system, S is the state of the system at which the data is input, and O is the expected output of the system.
5. **Test data :** When multiple sets of values or data are used to test the same functionality of a particular feature in the test case, the test values and changeable environmental components are collected in separate files and stored as test data.
6. **Test scripts :** The test script is the combination of a test case, test procedure and test data.
7. **Test suite :** This is the set of all test cases with which a given software product is to be tested.

Que 4.4. What is the aim of software testing ?

Answer

1. The aim of the testing process is to identify all defects existing in a software product.
2. However, for most practical systems, even after satisfactorily carrying out the testing phase, it is not possible to guarantee that the software is error free.
3. This is because of the fact that the input data domain of most software products is very large.
4. It is not practical to test the software exhaustively with respect to each value that the input data may assume.

5. Even with this practical limitation of the testing process, the importance of testing should not be underestimated.
6. Testing provides a practical way of reducing defects in a system and increasing the user's confidence in a developed system.

Que 4.5. Define test plan and discuss various types of test plans.

Answer

Test plan is a document describing the scope, approach, resources, and schedule of intended testing activities. It identifies the test items, the features to be tested, the testing tasks, who will do each task, and any risks requiring contingency planning. Several different general types of test plans are :

1. A mission plan tells "why". Usually, only one mission plan appears per organization or group. Mission plans describe the reason that the organization exists, are typically very short (5–10 pages), and are the least detailed of all plans.
2. A strategic plan tells "what" and "when". Again, only one strategic plan is usually used per organization, although some organizations develop a strategic plan for each category of project. Strategic plans are more detailed and longer than mission plans, sometimes 20-50 pages or more. They are seldom detailed enough to be directly useful to practicing testers or developers.
3. A tactical plan tells "how" and "who". Most organizations use one overall tactical plan per product. Tactical plans are the most detailed of all plans, and are usually living documents. For example, a tactical test plan would specify how each individual unit will be tested.

Que 4.6. What is the purpose of test plan and what are its contents ?

Answer

1. The main purpose of preparing a test plan is that everyone concerned with the project are in sync with regards to the scope, responsibilities, deadlines and deliverables for the project.
2. Purpose of preparing a test plan :
   a. A test plan is a useful way to think through the efforts needed to validate the acceptability of a software product.
   b. The completed document will help people outside the test group to understand the 'why' and 'how' of product validation.
   c. It should be thorough enough to be useful but not so thorough that no one outside the test group will read it.

Contents of a test plan are as follows :

1. **Purpose :** This section should contain the purpose of preparing the test plan.
2. **Scope :** This section should talk about the areas of the application which are to be tested by the QA team and specify those areas which are definitely out of scope (screens, database, mainframe processes etc).
3. **Test approach :** This would contain details on how the testing is to be performed and whether any specific strategy is to be followed (including configuration management).
4. **Entry criteria :** This section explains the various steps to be performed before the start of a test *i.e.*, pre-requisites.
5. **Resources :** This section should list out the people who would be involved in the project and their designation etc.
6. **Tasks/Responsibilities :** This section talks about the tasks to be performed and the responsibilities assigned to the various members in the project.
7. **Exit criteria :** Contains tasks like bringing down the system/server, restoring system to pre-test environment, database refresh etc.
8. **Schedules/Milestones :** This section deals with the final delivery date and the various milestone dates to be met in the course of the project.
9. **Hardware/Software requirements :** This section would contain the details of PC's/servers required to install the application or perform the testing.
10. **Risks and mitigation plans :** This section should list out all the possible risks that can arise during the testing and the mitigation plans that the QA team plans to implement the risk actually turns into a reality.
11. **Tools to be used :** This would list out the testing tools or utilities (if any) that are to be used in the project, example, WinRunner, Test Director, PCOM, WinSQL.
12. **Deliverables :** This section contains the various deliverables that are due to the client at various points of time *i.e.*, daily, weekly, start of the project, end of the project etc.
13. **References :**
    a. Procedures
    b. Templates (Client specific or otherwise)
    c. Standards/Guidelines, example, QView
    d. Project related documents (RSD, ADD, FSD etc).
14. **Annexure :** This could contain embedded documents or links to documents which have been/will be used in the course of testing, for example, templates used for reports, test cases etc.

15. **Sign-off :** This should contain the mutual agreement between the client and the QA team with both leads/managers signing off their agreement on the test plan.

Que 4.7. What should be the criteria for designing test cases ? Derive a set of test cases for the following : A sort routine which sorts arrays of integers.

Answer

1. A test case is a detailed procedure that fully tests a feature or an aspect of a feature whereas the test plan describes what is to be tested.
2. A test case describes how to perform a particular test.
3. We need to develop a test case for each test listed in the test plan or test specification.
4. Test cases should be written by someone who understands the function or technology being tested and should go through peer review.
5. Test cases include information such as the following :
   a. Purpose of the test.
   b. Special hardware requirements, such as modem.
   c. Special software requirements, such as a tool.
   d. Specific setup or configuration requirements.
   e. Description of how to perform the test.
   f. Expected results or success criteria for the test.

Test case for insertion sort :

```
Void sort(int array [ ], int len) {
int tempElem ;                  /* The element we are going to insert in
                                   the right place */
int tempElemIndex;              /*The index to the element we are about
                                   to move to its correct place */
int i ;                         /*A counter used to shift elements in the
                                   array to make room for tempElem */
if (NULL != array) { /* The array cannot be NULL */}
/* We are going to go through the entire array and insert each element
in its proper place in the already sorted portion of the array */
for (tempElemIndex = 0 ; tempElemIndex < len; tempElemIndex++)
{tempElem = array[tempElemIndex] ;
                i = tempElemIndex – 1 ;
                /* Go through the sorted portion of the array and if the
                element is larger than the element we are trying to insert,
                shift the larger element down one step.
                Repeat this until we find the right spot for the element we
                are inserting */
                while (i >= 0 && array[i] > tempElem) {
```

```
                array[i + 1] = array[i]
                i = i - 1 ;
                }
                /* Insert the element in its place in the stored portion of
                the array */
                array[i + 1] = tempElem ;
                }
        }
        return ;
        }
```

Que 4.8. **What do you mean by testing? Describe type and levels of testing in brief. Also, discuss all testing strategies with suitable example and diagram.** **UPTU 2013-14, Marks 05**

OR

Discuss about types of testing and levels of software testing. **UPTU 2014-15, Marks 10**

Answer

Testing : Refer Q. 4.1, Page 112C, Unit-4.

Levels of testing : The relation of the faults introduced in different phases, and the different levels of testing is shown in Fig. 4.8.1.

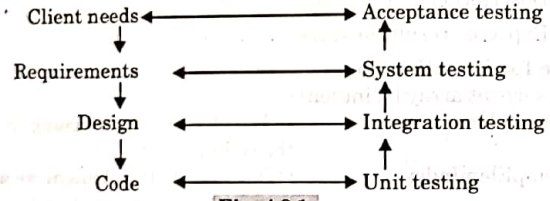


Fig. 4.8.1.

1. The basic levels are unit testing, integration testing, system testing and acceptance testing. These different levels of testing attempt to detect different types of faults.
2. The first level of testing is called unit testing. In this, different modules are tested against the specifications produced during design for the modules.

   This includes two types of testing :

   a. White box testing

   b. Black box testing
3. The next level of testing is often called integration testing. In this, many unit-tested modules are combined into subsystems, which are then tested. The goal here is to see if the modules can be integrated properly.

4. **Integration testing includes :**
   a. **Regression testing :** Change of behaviour due to modification or addition is called 'Regression'. It is used to bring changes from worst to least.
   b. **Incremental integration testing :** Checks out for bugs which encounter when a module has been integrated to the existing.
   c. **Smoke testing :** It is the battery of test which checks the basic functionality of program. If fails, then the program is not sent for further testing.
5. The next level of testing is system testing. Here, the entire software system is tested. The reference document for this process is the requirements document, and the goal is to see if the software meets its requirements.
6. **System testing includes :**
   a. **Recovery testing :** System is forced to fail and is checked out how well the system recovers the failure.
   b. **Security testing :** Checks the capability of system to defend itself from hostile attack on programs and data.
   c. **Performance testing :** Used to determine the processing speed.
   d. **Installation testing :** Installation and uninstallation is checked out in the target platform.
   e. **Load and stress testing :** The system is tested for maximum load and extreme stress points are figured out.
7. Acceptance testing is sometimes performed with realistic data of the client to demonstrate that the software is working satisfactorily.
8. **Acceptance testing includes :**
   a. **UAT (User acceptance testing) :** It ensures that the project satisfies the customer requirements.
   b. **Alpha testing :** It is the test done by the client at the developer's site.
   c. **Beta testing :** This is the test done by the end users at the client's site.
   d. **Long term testing :** Checks out for faults occurrence in a long term usage of the product.
   e. **Compatibility testing :** Determines how well the product is substantial to product transition.

Que 4.9. **What do you mean by unit/module testing?**

Answer

1. Unit testing focuses verification effort on the smallest unit of software design, the software component or module.

2. The relative complexity of tests and uncovered errors is limited by the constrained scope established for unit testing.
3. The unit testing is white-box oriented, and the step can be conducted in parallel for multiple components.
4. In unit testing, individual components are tested to ensure that they are working properly in the same manner as required.
5. In this process, a module (software component) is taken and executed in isolation from the rest of software product. That's why, it is also called "software component testing".
6. It is the lowest level of testing of an application, under test software meets the requirements expected from it.
7. Unit testing is typically conducted by development team and programmer who coded the unit.
8. The main reasons for doing unit testing are :
   a. The size of a single module is small enough that can locate an error very easily.
   b. Due to small size, it can be tested in some demonstrably exhaustive fashion.
   c. Confusing interactions of multiple errors which occurs when many parts tested together are eliminated.

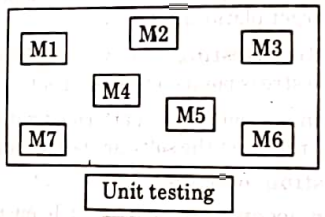


Fig. 4.9.1.

9. Unit testing becomes simple where modules are highly cohesive.
10. When module address less number of functions then testing is more easy and accurate as less number of test cases is required to test it.

Que 4.10. **What do you mean by software integration strategies ? Illustrate their importance.**

OR

Differentiate between top down, bottom up integration and big-bang strategies.

OR

Compare the relative merits and demerits of different integration testing strategies.

Answer

Integration testing :

1. Once individual program components have been tested, they must be integrated to create a partial or complete system.
2. Integration test should be developed during system specification, and this integration testing should begin as soon as usable versions of some of the system components are available.
3. The primary objective of integration testing is to check the modules interface *i.e.*, there are no errors in parameter passing, when one module invokes the functionality of another module.
4. After each integration step, the partially integrated system is tested.
5. This each step testing is done to avoid the errors causes in complex system during component connecting process.

The main integration techniques are as follows :

1. **Top-down integration testing :**
   a. In this approach, testing process starts with testing the top most module/component in the hierarchy and move downwards.
   b. This process continues until all components are integrated and then whole system has been completely tested.
   c. Top-down integrating testing approach requires the use of program stubs to simulate the effect of lower-level routines that are called by the routines under test.
   d. A pure top-down integration does not require any driver routines.

Advantages of top-down integration testing :

a. Design errors are detected as early as possible, saving development time and costs because corrections in the module design can be made before their implementation.
b. The characteristics of a software system are evident from the start, which enables a simple test of the development state and the acceptance by the user.

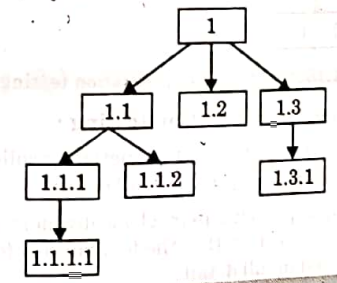


Fig. 1.10.1. Top-down integration.

c. The software system can be tested thoroughly from the start with test cases without providing (expensive) test environments.

Disadvantages of top-down integration testing :

a. Strict top-down testing proves extremely difficult because designing usable surrogate objects can prove very complicated, especially for complex operations.

b. Errors in lower hierarchy levels are hard to localize.

2. Bottom-up integration testing :

a. It is very popular approach of merging the components to test a larger system.

b. In this method, each subsystem (component) at the lower level of system hierarchy is tested individually first.

c. Then the next component who calls the previously tested ones to be tested.

d. This approach is followed repeatedly until all components are included in the testing.

e. The primary purpose to test each subsystem is to test interface among various models making up the subsystem.

f. In this, both control and data interface are tested.

g. In pure bottom-up testing, no stubs are required, and only test-drivers are required.

h. Bottom-up integration testing is mainly used when the performance and machine interface of the system is the main concern.

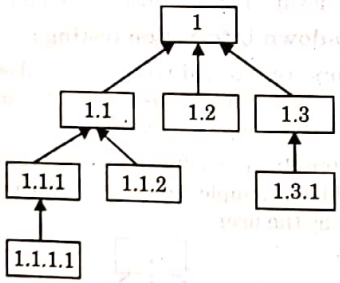


Fig. 1.10.2. Bottom-up integration testing.

Advantages of bottom-up integration testing :

a. The bottom-up integration testing method is solid and proven. The objects to be tested are known in full detail.

b. The bottom-up approach is psychologically more satisfying because the tester can be certain that the foundations for the test objects have been tested in full detail.

Disadvantages of bottom-up integration testing :

a. The characteristics of the finished product are only known after the completion of all implementation and testing, which means that design errors in the upper levels are detected very late.

b. Testing individual levels also inflicts high costs for providing a suitable test environment.

c. The complexity occurs when the system is made up of a large number of small subsystems that are at the same level.

3. Big-bang testing :

a. It is a non-incremental integration approach.

b. It is the simplest testing approach, where all modules making a complete system are integrated and tested as a whole and disorder usually gives unpredictable results.

c. The set of errors encountered here are very difficult to localize as it may potentially belong to any of the module being integrated.

d. And once these errors are corrected, new ones appear and the process continues in an endless loop.

e. Therefore, debugging errors reported during big-bang integration testing are very expensive to fix.

f. So, this technique can only be used for very small system because of difficulty in identifying the errors occurring in module interface for large system.

g. The main problem with big-bang integration is the difficulty of isolating the sources of error.

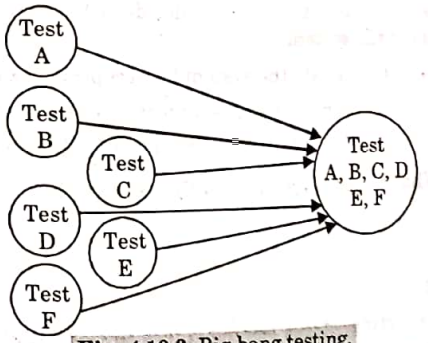


Fig. 4.10.3. Big-bang testing.

4. Sandwich integration testing :

a. The combination of top-down and bottom-up integration testing techniques is called sandwich integration.

b. This system is viewed as three layers just like a sandwich.

c. An upper layer of sandwich use top-down integration, the lower layer of sandwich integration use bottom-up integration, and the middle layer, that is, called target layer use the testing on the basis of system characteristics and structure of the component hierarchy.

d. It allows integration testing to begin early in the test process.

e. It contains advantage of top-down and bottom-up integration testing.

f. It is most commonly used integration testing approach.

Que 4.11. What is acceptance testing ? Why acceptance testing is needed ?

Answer

1. The purpose of acceptance testing is to confirm that the system meets its business requirements and to provide confidence that the system is working properly before it is "hand over" to the customer.
2. In this testing, customer is responsible for checking the performance of the system.
3. Acceptance testing is performed by nominated user representatives under guidance and supervision of testing team and developer.
4. The acceptance test is written, conducted and evaluated by customers, with assistance from the developers, only when the customer request an answer to a technical question.
5. During acceptance testing, it is very important that there should be an independent observer of testing process especially when the customers are not having too much IT (information technology) knowledge.
6. By doing acceptance testing the user decides whether to accept or reject the delivery of the system.
7. The main way to evaluate the system for acceptance is benchmark test.
8. In this, the customer prepares a set of test cases that represents typical condition under which the system will operate when actually installed.

Que 4.12. What is stress and security testing ?

Answer

Stress testing :

1. In software testing, stress testing refers to tests that determine the robustness of software by testing beyond the limits of normal operation.
2. Stress testing is particularly important for "mission critical" software, but is used for all types of software.
3. Stress tests commonly put a greater emphasis on robustness, availability, and error handling under a heavy load, than on what would be considered correct behaviour under normal circumstances.

4. Stress testing is the system testing of an integrated, black box application that attempts to cause failures involving how its performance varies under extreme but valid conditions (example, extreme utilization, insufficient memory, inadequate hardware, and dependency on over-utilized shared resources).
5. Stress testing typically involves the independent test team performing the following testing tasks using the following techniques :
   a. Test planning
   b. Test reuse
   c. Test design
      i. Use case based testing.
      ii. Workload analysis to determine the maximum production workloads.
   d. Test implementation
      i. Develop test scripts simulating extreme workloads.
   e. Test execution
      i. Regression testing
   f. Test reporting

Typical examples include stress testing as an application in :

1. Software only.
2. A system including software, hardware and data components.
3. Batch with no real time requirements.
4. Soft real time (*i.e.*, human reaction times).
5. Hard real time (for example, avionics, radar, automotive engine control).
6. Embedded within another system (for example, flight-control software, cruise-control software).

Security testing : Any computer based system that manages sensitive information or causes actions that can harm individuals is a target for improper or illegal penetration.

1. Security testing attempts to verify that protection mechanisms built into a system will protect it from improper penetration.
2. During security testing, the tester plays the role of the individual who desires to penetrate the system.
3. The tester may attempt to acquire passwords through external means, may attack the system with custom software designed to breakdown any defenses that have been constructed, may overwhelm the system, thereby denying services to others.
4. Good security testing will ultimately penetrate a system. The role of the system designer is to make penetration cost more than the value of information that will be obtained.

Que 4.13. **What is regression testing ? When we need regression testing ?**

Answer

1. Testing is used to find out errors in the software due to which it is not working properly. When these errors are found out then different methods are used to correct these errors.
2. Regression testing is used to identify these new errors or faults.
3. Regression testing identifies new faults that may have been introduced as correct ones.
4. A regression test is applied on new version or on release of software or its components to verify that it is still performing the same function in same manner.
5. Regression testing can be applied in development as well as maintenance phase of software life cycle. In development phase, regression testing is done after correcting the errors found during the testing of software.
6. While in maintenance phase of software life cycle, adaptive, corrective and preventive maintenance is done due to which some modification is done in the software and these modification may cause some new errors.
7. To find out these errors, regression testing is used in maintenance phase of software.
8. Regression testing may be conducted manually, by re-executing a subset of all test cases or using automated capture/playback tools.
9. Capture/playback tools enables software engineer to capture test cases and results for subsequent playback and comparison.
10. A regression test suite (the subset of test to be executed) contains three different classes of test cases :
    a. A representative sample of tests that will exercise all software functions.
    b. Additional tests that focus on software functions are likely to be affected by the change.
    c. Test that focuses on the software components that have been changed.

Que 4.14. **What do you understand by test driver and test stub ? Explain in detail.**

OR

What are drivers and stub modules in context of integration and unit testing of software product ? Why are stubs and drivers modules required ?

Answer

Test driver :

1. A test driver is a software module or application used to invoke a test and provide test data, control and monitor execution and reports test outcomes.
2. Test driver are used for testing of sub-module in the absence of main control module.
3. A component driver routine calls a particular component and passes test case to it.
4. The driver may be written for a unit (module) or for integration (for combining the module).
5. The test drivers are not difficult to code since it rarely requires complex processing.

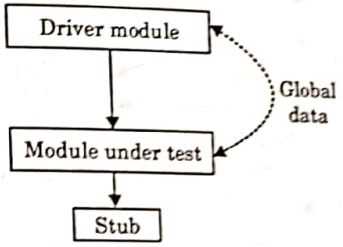


Fig. 4.14.1.

6. A driver module should contain the non-local data structures accessed by the module under test and should also have code to call the different functions of the module under test with appropriate parameter values for testing.
7. In case of bottom-up integration testing, the role of test driver can be easily understood by following example. Consider a component based hierarchy :

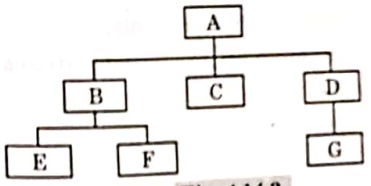


Fig. 4.14.2.

   a. To test the system bottom-up, we first test the lowest level '*E*', '*F*', '*G*' module as described above.
   b. Once these module is tested separately, we have to write a special code to aid integration *i.e.*, known as test driver.
   c. If each individual module '*E*', '*F*', and '*G*' are working correctly we move to next higher level.

d. Unlike the lowest level components, the next level components are not tested separately.

e. Instead they are combined with components they call (which have already tested).

f. In this case, we will test 'B', 'E' and 'F' together.

g. If problem occurs we know that its cause is either in 'B' or in the interface between 'B' and 'E' or 'B' and 'F'. Since 'E' and 'F' are functioning properly on their own.

h. If we try to test 'B', 'E' and 'F' as a whole then we may not be easily able to isolate the problems caused.

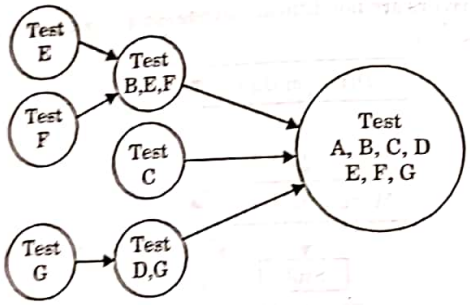


Fig. 4.14.3.

i. Similarly, we test 'D' with 'G'. Because 'C' calls no other components, we test it by itself.

j. Finally, we test all components together. Fig. 4.18.3 shows the sequence of tests and their dependencies.

Advantages of bottom-up (driver based) integration technique:

1. No test stub needed.
2. Errors in critical modules are found easily.

Disadvantages of bottom-up (driver based) integration technique:

1. More test driver needed.
2. Interface errors are discovered later.

Test stub:

1. Test stubs are specialized implementation of elements used for the testing purpose, which are dummy of a real component.
2. Test stubs are program or components that have deterministic behaviour and are used to interface with subsystem in order to take care of dependencies.
3. Basically, stubs are used in top-down approach.

4. In it, the main control module is tested in the absence of stub module.
5. For example, if we want to test a module 'A' but it needs module 'B', 'C' and 'D'. In this case, we create dummy modules for 'B', 'C' and 'D'. These all are called stub and used to run 'A'.

Advantages of top-down (stub based) integration technique:

1. No need of test driver.
2. Modular feature.

Disadvantage of top-down (stub based) integration technique:

1. It requires lots of stub writing.

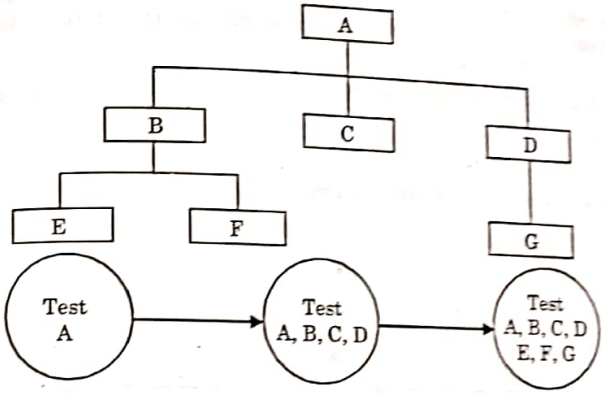


Fig. 4.14.4.

Que 4.15. What is white box testing? List out advantages and disadvantages of white box testing.

Answer

1. White box testing is a software testing approach that examines the program structure and derives test data from the program logic.
2. In this approach, test group must have complete knowledge about the internal structure of the software.
3. Structural testing is usually applied to relatively small program units such as subroutine or operations associated with objects.
4. As the name implies, the tester can analyze the code and use knowledge about the structure of component to derive test data.
5. The white box testing allows to peep (preview) inside the box, it basically focuses on internal knowledge of the software to guide the selection of test data.
6. The white box testing is also known by name of glass box testing, structural testing, clear box testing, open-box testing, logic driven testing and path-oriented testing.

7. White box testing is a test case design method that uses the control structure of the procedural design methods to derive test cases.
8. Test cases can be derived that :
   a. Guarantee that all independent paths within a module have been exercise at least once.
   b. Exercise all logical decisions on their true and false sides.
   c. Execute all loops at their boundaries and within their operational bounds.
   d. Exercise internal data structures to assure their validity.
9. In white box testing, the test cases are selected on the basis of examination of the code rather than specification. The white box testing is illustrated in Fig. 4.15.1.

Advantages of white box testing :

1. Helps to optimize the code.
2. Reveals errors in hidden code.

Disadvantages of white box testing :

1. It is very expensive.
2. It is time consuming.

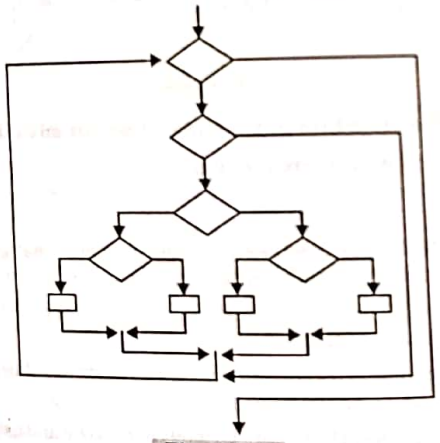


Fig. 4.15.1.

Que 4.16. Explain white box testing techniques.

OR

Explain basis path testing using flow graph analysis in detail.

Answer

The important white box techniques are as follows :

1. **Basis path testing :** This method enables the designer to derive a logical complexity measure of a procedural design and use it as a guide for defining a basis set of execution paths. Test cases that exercise the basis set are guaranteed to execute every statement in the program at least once during testing.

   a. Flow graph notation :

   i. A flow graph depicts logical flow control using the notation given in the Fig. 4.16.1.

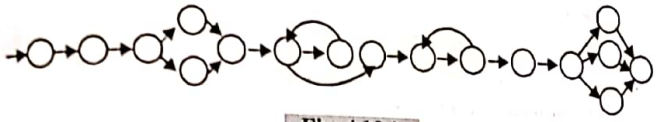


Fig. 4.16.1.

   ii. Each circle in a flow graph known as flow graph node represents one or more procedural statements.

   iii. The arrows on the flow graph, called edges or links, represent flow of control.

   b. Cyclomatic complexity :

   i. Cyclomatic complexity is software metric that provides a quantitative measure of the logical complexity of a program.

   ii. It defines the number of independent paths in the basis set and thus provides an upper bound for the number of tests that must be performed.

   iii. $V(G) = 3$ (predicate nodes) $+ 1 = 4$.

2. **Condition testing :**

   i. Condition testing is a test case design method that is used for checking up logical conditions contained in a program module.

   ii. A simple condition is a boolean variable or a relational expression, which includes NOT (¬), EQUALTO (=), NOT EQUALTO (!=), LESS-THAN (<), LESS-THAN EQUALTO (<=), GREATER-THAN (>), GREATER-THAN EQUALTO (>=).

   iii. The compound statements are constructed with the help of OR (|), AND (&&) and NOT (>).

   iv. If a condition is incorrect, then at least one component of condition is incorrect.

   v. The condition testing method focuses on testing each condition in the program.

3. **Data flow testing :**
   i. Data flow testing focus on the points at which variables receives the values and the points at which these values are used.
   ii. This method selects test paths of a program according to the locations of definitions and uses of variables in the program.

4. **Loop testing :** Loops are the foundation stone for vast majority of all algorithms implemented in software. Hence, proper attention must be paid to loop testing while performing white box testing.

   Loops are classified as :

   a. **Simple loops :** The following tests can be applied to simple loops where n is the maximum number of allowable passes through the loop :
      i. Skip the loop entirely.
      ii. Only one pass through the loop.
      iii. m passes through the loop where $m < n$.
      iv. $n-1, n, n+1$ passes through the loop.

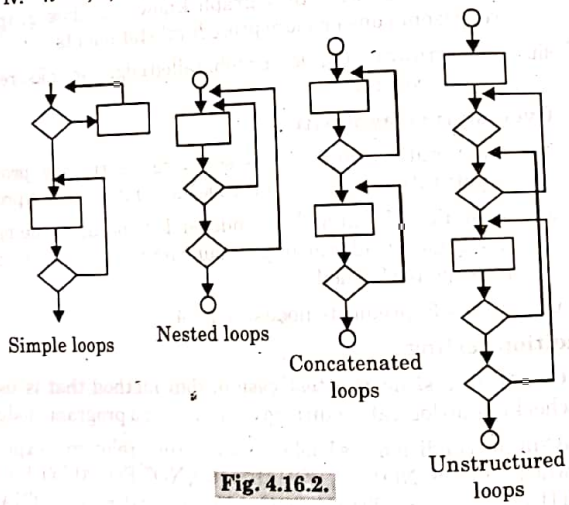


Fig. 4.16.2.

   b. **Nested loops :** The testing of nested loops cannot simply extend the technique of simple loops since this would results in a geometrically increasing number of test cases. The approach for nested loops that will help to reduce the number of test :
      i. Start at the innermost loop. Set all other loops to minimum values.
      ii. Conduct simple loop tests for the innermost loop while holding the outer loops at their minimum iteration parameter values. Add other tests for out-of-range or excluded values.

      iii. Work outward, conducting tests for the next loop while keeping all other outer loops at minimums and other nested loops to typical values.
      iv. Continue until all loops have been tested.

   c. **Concatenated loops :** Concatenated loops can be tested as simple loops, if each loop is independent of the other. If they are not independent (for example, the loop counter for one is the loop counter for the other), then the nested approach can be used.

   d. **Unstructured loops :** Whenever possible, this type of loops should be redesigned to reflect the use of the structured programming constructs.

Que 4.17. What is black box testing ? List out advantages and disadvantages of black box testing.

Answer

1. In black box testing, the tester don't know the internal structure of module, he tests only for input/output behaviour.
2. Black box testing focuses on the functional requirements of the software.
3. It enables the software engineer to derive sets of input conditions that will fully exercise all functional requirements for a program.
4. Black box tests serve to detect deviations of a test object from its specification.
5. The selection of test cases is based on the specification of the test object without knowledge of its inner structure.
6. It is also important to design test cases that demonstrate the behaviour of the test object on erroneous input data.
7. The black box testing is also known by the name of functional testing, exterior testing, specification testing, data-driven testing and input/output driven testing.

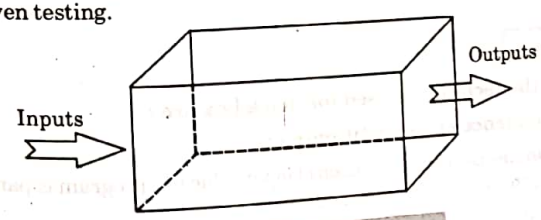


Fig. 4.17.1. Black box testing.

8. Test case should be derived which :
   a. Reduce the number of additional test cases that must be designed to achieve reasonable testing.
   b. Tell us something about the presence or absence of classes of errors, rather than an error associated only with the specific test at hand.

Tests are designed to answer the following questions :

1. How is the function's validity tested ?
2. What classes of input will make good test cases ?
3. Is the system particularly sensitive to certain input values ?
4. How are the boundaries of a data class isolated ?
5. What data rates and data volume can the system tolerate ?
6. What effect will specific combination of data have on system operation ?

Black box testing attempts to find errors in the following categories :

a. Incorrect functions
b. Missing functions
c. Interface errors
d. Data structure errors
e. External database access errors
f. Performance errors
g. Initialization and termination errors

Advantages of black box testing :

1. The test is unbiased because the designer and tester are independent of each other.
2. Test is done from the point of view of user not the designer.
3. Test cases can be designed as soon as specification is completed.
4. For tester, there is no need to know any programming language.

Disadvantages of black box testing :

1. The test can be redundant if software designer has already run a test case.
2. The test cases are difficult to design.

Que 4.18. Explain the different techniques used in black box testing.

Answer

Some of the techniques used for black box are :

1. **Equivalence class partitioning :**
   a. In this method, the domain of input value to a program is partitioned into a set of equivalence classes.
   b. Equivalence classes are usually formed for input domains and are classified into a valid equivalence class and one or more invalid equivalence classes.
   c. The idea is to partition the input domain of the system into a finite number of equivalence classes such that each member of the class would behaves in a similar fashion, *i.e.*, if a test case in one class results some errors then the other member of class would also results same error.
   d. The techniques increase the efficiency of software testing as the numbers of input states are drastically reduced.
   e. This techniques involves two steps :
      i. Identification of equivalence classes.
      ii. Generating the test cases.
2. **Boundary value analysis :**
   a. The boundary value analysis is refinement of equivalence class approach.
   b. In boundary value analysis, we choose an input for a test case from an equivalence class, such that the input lies at the edge of the equivalence classes.
   c. Boundary values for each equivalence class, including the equivalence classes of the output, should be covered.
   d. Boundary value test cases are also called "extreme cases".
   e. Hence, we can say that a boundary value test case is a set of input data that lies on the edge of boundary of a class of input data or that generates output that lies at the boundary of a class of output data.
   f. Guidelines for boundary value analysis are :
      i. If an input condition specifies a range bounded by values a and b, test cases should be designed with the values a and b and just above and just below a and b.
      ii. If an input condition specifies a number of input value, test cases should be developed that exercise the minimum and maximum number. Values just above and just below minimum and maximum are also tested.
      iii. Apply guidelines (ii) and (iii) for each output condition.

By applying above guidelines, the boundary testing will be more complete thereby having a higher likelihood for error detection.

3. **Cause-effect graphing :**
   a. One weakness with the equivalence class partitioning and boundary value methods is that they consider each input separately.
   b. That is, both concentrate on the conditions and classes of one input.
   c. They do not consider combinations of input circumstances that may form interesting situations that should be tested.
   d. One way to exercise combinations of different input conditions is to consider all valid combinations of the equivalence classes of input conditions.

e. This simple approach will result in an unusually large number of test cases, many of which will not be useful for revealing any new errors.

f. Cause-effect graphing is a technique that aids in selecting combinations of input conditions in a systematic way, such that the number of test cases does not become unmanageably large.

g. This technique starts with identifying causes and effects of the system under testing.

h. A cause is a distinct input condition, and an effect is a distinct output condition.

i. Each condition forms a node in the cause-effect graph.

j. The condition should be states such that they can be set to either true or false.

In steps, cause effect testing are as follows :

1. For a model identify input conditions (causes) and actions (effect).
2. Develop a cause-effect graph.
3. Transform cause-effect graph into a decision table.
4. Convert decision table rules to test cases. Each column of decision table represents a test case.

Que 4.19. Discuss the differences between black box and structural testing and suggest how they can be used together in the defect testing process ?

Answer

Refer Q. 4.15, Page 129C and Q. 4.17, Page 133C; Unit-4.

Que 4.20. Differentiate between alpha and beta testing.

Answer

1. When software is developed for a particular customer a series of acceptance test can be done to help the customer to validate his/her requirements but when the software is developed for multiple customers (example utility software) this series of testing is not feasible.
2. For this, two main types of testing are connected with acceptance testing *i.e.,* Alpha and Beta testing. These are referred as internal and external testing.

a. Alpha testing :

1. A lot of testing of software product is done to ensure about the system performance before delivering the software product to the customer but it is not possible to "predict" how the customer will really use program/module/system.

2. While operating manuals and user manual are explained and given to user, even then, there are possibility of misinterpretation of instruction, complex data combination that is not understandable to user, the output produced by system in user environment is not upto the mark though tester found it ok when he/she tested the system.
3. To solve this problem customer is called at developer site to test the system. This type of testing is called alpha testing.
4. In alpha testing, the software is used in its future environment on which it has to work on user site.
5. In alpha testing, the developer play a role of "invisible observer" he do not test anything, he/she only records the usage problems/errors that occurs while end user interact with system/software. It is done at developer's site so it is described as internal testing process.

b. Beta testing :

1. Beta testing of software is done at user site in "real-world environment".
2. This type of testing is done when developer is quite confident on the performance of their system and thinks it is ready for final delivery.
3. Its purpose is to validate that the software is ready to release to "real customers".
4. This testing is done out of developing environment of software so it is described as external testing.
5. Following points should be kept in mind while doing beta testing :
   i. "Know" the beta tester : It is very important to have knowledge about beta customer as whether they are experienced or novice (beginner) tester. As experienced user/ tester will not take interest in lower detail while it will be important for novice user.
   ii. Make sure that beta tester are testing the given system and providing appropriate report according to it.
   iii. It is very good way for checking the capability and configuration of the bugs especially for web based applications.
   iv. In beta testing only the superficial problems may get reported. A lot of efforts required to motivate the user to spend an adequate time on trying out the system.

Que 4.21. What is the difference between testing and debugging ?

Answer

The difference between testing and debugging are as follows :

S. No.	Testing	Debugging
1.	Starts with known conditions, user predefined procedures, predictable outcomes.	Unknown initial condition, end cannot be predicted.
2.	Should be planned, designed, and scheduled.	Cannot be constrained.
3.	Is a demonstration of errors/ apparent correctness.	Is a deductive process.
4.	Proves a programmer's "failure"	Vindicates a programmer.
5.	Should strive to be predictable, dull constrained, rigid, inhuman.	Demands intuitive leaps, conjectures, experimentations, freedom.
6.	Much can be done without design knowledge.	Impossible without design knowledge.
7.	Can be done by outsider.	Must be done by insider.
8.	Theory of testing is available.	Only recently have theorists started.
9.	Much of the test design and execution can be automated.	Automation is still a dream.

Que 4.22. Write short note on walkthrough.

Answer

Refer Q. 3.16, Page 105C, Unit-3.

Que 4.23. Write a short note on test strategies of software testing.

UPTU 2012-13, Marks 05

Answer

The purpose of test strategy is to clarify the major tasks and challenges of the test project. A good test strategy must be specific, practical and justified. The test strategy aims are as follows :

1. "Testing-in-the-small" and move toward "testing-in-the-large".
2. State testing objectives explicitly.
3. Understand the users of the software and develop a profile for each user category.

4. Develop a testing plan that emphasizes "rapid cycle testing".
5. Build "robust" software, that is, designed to test itself.
6. Use effective formal technical reviews as a filter prior to testing.
7. Conduct formal technical reviews to assess the test strategy and test cases themselves.
8. Develop a continuous improvement approach for the testing process.

PART-2

Program Correctness, Program Verification and Validation, Testing Automation and Testing Tools, Concept of Software Quality, Software Quality Attributes, Software Quality Metrics and Indicators.

CONCEPT OUTLINE : PART-2

- Software validation refers to a set of activities that ensure that the software, that is, going to be built is traceable to customer requirements.
- Software verification refers to a set of activities that ensures that software correctly implements a specific function.
- Software test automation refers to the activities and efforts that intend to automate engineering tasks, and operations in a software test process.
- Software quality refers to achieving high levels of user satisfaction, portability, maintainability, robustness and fitness for use.
- Software metrics are all about measurements which in turn, guide us how to make things better.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 4.24. What is software validation and verification ? Explain its objectives.

OR

Define verification and validation.

Answer

1. Requirements validation is concerned with showing that the requirements actually define the system that the customer wants.

2. It refers to a set of activities that ensure that the software, that is, going to be build is traceable to customer requirements, *i.e.*,
"Are we building the right product".

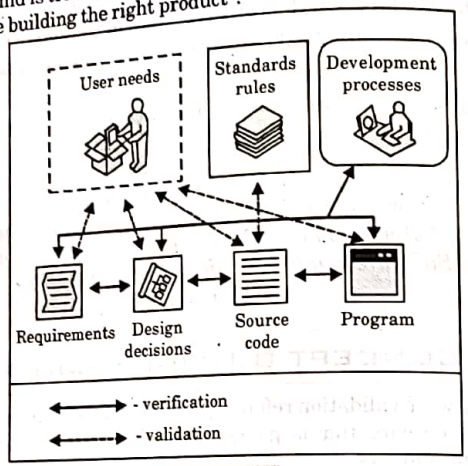


Fig. 4.24.1.

3. Whereas software verification refers to a set of activities that ensure that software correctly implements a specific function *i.e.*,
"Are we building the product right".
4. Traditionally, software verification and validation is defined as system engineering methodology to ensure that quality is built into software during development.
5. The analysis and test activities are performed by V & V, evaluate and assess the software products and the development processes during each software life cycle phase in parallel with, not after the completion of, the development effort.
6. This provides early identification of errors, improving software performance, identification of program risks. V & V attacks on two of major contributor of software failure :
   a. incorrect or missing requirements, and
   b. poor organization in software architecture and failure to plan effectively.
7. It also acts as powerful risk management tool. In many cases, V & V is governed by standards establishing software development project management, and the software quality assurance requirements.

Objectives of verification and validation : Software V & V comprehensively analyzes and tests software during all stages of its development and maintenance in parallel with development process to :

1. Determine that it performs its intended functions correctly.
2. Ensure that it performs no unplanned functions.
3. Measure and assess the quality and reliability of software.

4. Hardware including all hardware directly or indirectly influenced by the software.
5. External software linked to the system.
6. Find errors in the software process and product as early as possible.
7. Assist in determining good requirements and design.
8. Ensure that quality is built into the software and that the software will satisfies the software requirements.
9. Satisfy the user that system is being developed according to standards and specifications.
10. Predict how well the interim products will result in a final product that satisfies the software requirement.

Que 4.25. Differentiate between validation and verification.

UPTU 2004-05, Marks 05

Answer

S.No.	Validation	Verification
1.	Am I building the right product ?	Am I building the product right ?
2.	Determining if the system complies with the requirements and performs functions for which it is intended and meets the organization's goals and user needs. It is traditional and is performed at the end of the project.	The review of interim work steps and interim deliverables during a project to ensure they are acceptable. To determine if the system is consistent, adheres to standards, uses reliable techniques and prudent practices, and performs the selected functions in the correct manner.
3.	Am I accessing the right data (in terms of the data required to satisfy the requirements).	Am I accessing the data right (in the right place; in the right way).
4.	High level activity.	Low level activity.
5.	Performed after a work product is produced against established criteria ensuring that the product integrates correctly into the environment.	Performed during development on key artifacts, like walkthroughs, reviews and inspections, mentor feedback, training, checklists and standards.

6.	Determination of correctness of the final software product by a development project with respect to the user needs and require-ments.	Demonstration of consistency, completeness, and correctness of the software at each stage and between each stage of the development life cycle.

Que 4.26. Define the term "software quality". Why we need to manage the quality ?

OR

What is software quality ?

Answer

a. Now a days, quality is critical for survival and success. The quality of any product describes as "achieving high levels of user satisfaction, portability, maintainability, robustness and fitness for use".

b. Maintaining the quality of a product is not an easy task.

c. Software is now a global business and organizations will not succeed in the global market unless they produce quality products and services.

d. There are several reasons why business should be concerned with quality :

1. Quality is competitive issue now.
2. Quality is must for survival.
3. Quality gives you global reach.
4. Quality is cost effective.
5. Quality helps in retaining customers and increasing the profits.
6. Quality is a hallmark of world class business.

e. In software development, quality of designs encompasses requirements, specifications and the design of system.

f. 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.

g. The quality plays important role for user satisfaction. It can be observed as,

User satisfaction = Complaint product + Good quality + Delivery within budget and schedule

h. The quality control or quality management involves a series of inspections, reviews and test used throughout the software process to ensure that each work product meets the requirements placed upon it.

i. Quality control includes a feedback loop to the process that created the work product.

j. The combination of measurement and feedback allows us to tune the process when the work product created fails to meet their specifications.

k. This approach views quality control as part of the manufacturing process.

l. A key concept of quality control is that all work product have defined, measurable specifications to which we may compare the output of each process.

m. Thus, at last we can describe software quality as "Conformance to explicitly stated functional and performance requirements, explicitly documented development standard and implicit characteristics that are expected from all professionally developed software".

Que 4.27. What are software quality assurance attributes ?

Answer

The software quality assurance attributes are as follows :

1. **Runtime system qualities :** It can be measured as the system executes :
   - **a. Functionality :** The ability of the system to do the work for which it was intended.
   - **b. Performance :** The response time, utilization, and throughput behaviour of the system. Not to be confused with human performance or system delivery time.
   - **c. Security :** A measure of system's ability to resist unauthorized attempts at usage or behaviour modification, while still providing service to legitimate users.
   - **d. Availability (reliability quality attributes falls under this category) :** The measure of time that the system is up and running correctly; the length of time between failures and the length of time needed to resume operation after a failure.
   - **e. Usability :** The ability of two or more systems to cooperate at runtime.
2. **Non-runtime system qualities :** It cannot be measured as the system executes :
   - **a. Modifiability :** The ease with which a software system can accommodate changes to its software.
   - **b. Portability :** The ability of a system to run under different computing environments. The environment types can be either hardware or software, but is usually a combination of the two.
   - **c. Integrability :** The ability to make the separately developed components of the system to work together in a correct way.
   - **d. Testability :** The ease with which software can be made to demonstrate its faults.

3. **Business qualities :** Non-software system qualities that influence other quality attributes :
   a. **Cost and schedule :** The cost of the system with respect to time to market, expected project lifetime, and utilization of legacy.
   b. **Marketability :** The use of the system with respect to market competition.
   c. **Appropriateness for organization :** Availability of the human input, allocation of expertise, and alignment of team and software structure.
4. **Architecture qualities :** Quality attributes specific to the architecture itself.
   a. **Conceptual integrity :** The integrity of the overall structure that is composed from a number of small architectural structures.
   b. **Correctness :** Accountability for satisfying all requirements of the system.

Que 4.28. **What do you mean by software metrics ? Also, explain its application area.**

OR

Describe the characteristics of good software metric. Also, discuss various software metrics for project monitoring.

UPTU 2013-14, Marks 10

Answer

Software metrics :

1. Software metrics can be defined as "It is a continuous application of measurement based techniques to the software development process and its products to supply meaningful and timely management information, together with the use of those techniques to improve that process and its products".
2. Software metrics are all about measurements which, in turn, guides us how to make things better, to improve the process of developing software and to improve all aspects of the management of that process.
3. Software metrics are applicable to the whole development life cycle from initiation, when costs must be estimated, to monitoring the reliability of the end product in the field, and the way that product changes over time with enhancement.
4. It covers the techniques for monitoring and controlling the process of the software development, such that the fact that it is going to be six months late, is recognized as early as possible rather than the day before delivery is due.

5. Some metrics belong to multiple categories like quality metric. It may belong to all three categories. It focuses on the quality aspects of the product process, and the project.
6. The prediction of quality levels for software, often in terms of reliability, is another area where software metrics have an important role to play.
7. The use of software metrics to provide quantitative checks on software design is also a well established area.
8. Much research has been carried out, and some organizations have used such techniques to very good effect.
9. This area of software metrics is also being used to control software products which are in place and are subject to enhancement.
10. Software metrics are also used to provide management information. This includes information about productivity, quality and process effectiveness.
11. It is important to realize that this should be seen as an ongoing activity.
12. Snapshots of the current situation have their place, but the most valuable information comes when we can see trends in data. Is productivity or quality getting better or worse overtime ? Why is this happening ? What can management do to improve things ?
13. Statistical analysis is a part of it, but the information must be presented in a way that managers find it useful, at the right time and for the right reasons.
14. All this shows that software metrics is a vast field and have wide variety of applications throughout the software life cycle.

Categories of metrics : There are three categories of software metrics which are given below :

1. **Product metrics :** It describes the characteristics of the product such as size, complexity, design features, performance, efficiency, reliability, portability etc.
2. **Process metrics :** It describes the effectiveness and quality of the processes that produce the software product. Examples are :
   a. Effort required in the process.
   b. Time to produce the product.
   c. Effectiveness of defect removal during development.
   d. Number of defects found during testing.
   e. Maturity of the process.
3. **Project metrics :** It describes the project characteristics and execution. Examples are :
   a. Number of software developers.
   b. Staffing pattern over the life cycle of the software.
   c. Cost and schedule.
   d. Productivity.

Areas of applications : The most established area of software metrics is cost and size estimation techniques. These are :

1. There are many proprietary packages in the market that provide estimates of software system size, cost to develop a system, and the duration of the development or enhancement of the project.
2. These packages are based on estimation models, like COCOMO-I, COCOMO-II, developed by Berry Boehm. Various techniques that do not require the use of readymade tools are also available.
3. There has been a great deal of research carried out in this area, and this research continues in all important software industries and other organizations.
4. One thing that does come across strongly from the results of this research work is that organizations cannot rely, solely, on the use of proprietary packages.

Que 4.29. Describe lines of code (LOC), the size estimation method.

Answer

1. Lines of code (LOC) was the first measurement attempted. It has the advantage of being easily recognizable, seen and therefore counted.
2. Although this may seem to be a simple metric that can be counted algorithmically, there is no general agreement about what constitutes a line of code.
3. Early users of lines of code did not include data declarations, comments, or any other lines that did not result in object-code.
4. Later users decided to include declarations and other unexecutable statements but still excluded comments and blank lines.
5. The reason for this shift is the recognition that modern code can have 50 % or more data statements and that bugs occur as often in such statements as in real code.
6. For example, in the function, if LOC is simply a count of the number of lines then it contains 18 LOC.

```
1.  int sort (int x[ ], int n)
2.  {
3.  int i, j, save, im ;
4.  /* This function sorts array x in ascending order*/
5.  if (n < 2) return 1 ;
6.  for (i = 2; i < = n; i++)
7.  {
8.  im = i - 1 ;
9.  for (i = 1; i < = im; i++)
```



```
10.  if (x[i] < x[j])
11.  {
12.  save = x[i] ;
13.  x[i] = x[j] ;
14.  x[j] = save ;
15.  }
16.  }
17.  return 0 ;
18.  }
```

7. But most researchers agree that the LOC metric should not include comments or blank lines. Since these are really internal documentation and their presence or absence does not affect the functions of the program.
8. Moreover, comments and blank lines are not as difficult to construct as program lines. The inclusion of comments and blank lines in the count may encourage developers to introduce artificially many such lines in project development in order to create the illusion of high productivity, which is normally measured in LOC/PM (lines of code/person-month).
9. When comments and blank lines are ignored, the above program contains 17 LOC.
10. Line of code is defined as "A line of code is any line of program text that is not a comment or blank line, regardless of the number of statements or fragments of statements on the line".
11. This specially includes all lines containing program header, declarations, and executable and non-executable statements.
12 LOC is language dependent. A line of assembler is not the same as a line of COBOL. They also reflect what the system is rather than what it does.

Advantage of LOC : Simple to measure.

Drawbacks of LOC :

1. It is defined on code. For example, it cannot measure the size of specification.
2. It characterize only one specific view of size, namely length, it takes no account of functionality or complexity.
3. Bad software design may cause excessive line of code.
4. It is language dependent.
5. Users cannot easily understand it.

Que 4.30. Give Halstead's software science measures for :

1. **Program length**

2. **Program volume**
3. **Program level**
4. **Efforts**
5. **Language level**

OR

Describe two metrics that have been used to measure programmer productivity. Comment briefly on the advantages and disadvantages of each of these metrics.

Answer

Halstead's software science measure :

Halstead's software science is an analytical technique to measure size, development effort and development cost of software products. Halstead uses a few primitive programs parameters to develop the expressive for the overall program length, potential minimum volume, actual volume, language level, effort and development time.

1. **Program length :** The length of the program is defined by Halstead's quantifies the total usage of all operators and operands in the program so length,

$$N = N_1 + N_2$$

where

N_1 = total usage of all of the operators appearing in that implementation

N_2 = total usage of all of the operands appearing in that implementation.

2. **Program volume :** This is for the size of any implementation of any algorithm. The number of bits required to encode the program measured is known as volume and given by,

$$V = N \log_2 n$$

where n represents different identifiers uniquely

and $n = n_1 + n_2$

then $V = N \log_2 (n_1 + n_2)$

3. **Program level :** It measures the level of abstraction provided by the programming language.

The program level is given by,

$$L = \frac{V^*}{L}$$

Here, V^* = Potential minimum volume

4. **Efforts or programming effort :** The effort required to implement the program and can be obtained as,

$$\text{Efforts } (E) = \frac{V}{L} = \frac{\text{Program volume}}{\text{Program level}}$$

But $$L = \frac{V^*}{V}$$

Then $$E = \frac{V.V}{V^*} = \frac{V^2}{V^*}$$

5. **Language level :** Language level implies that as higher the level of a language, the less effort it takes to develop a program using that language, *i.e.*, language level is inversely proportional to the effort to develop a program.

$$L \alpha \frac{1}{E}$$

6. **Potential volume :** It is a metric for denoting the corresponding parameters in an algorithm's shortest possible form. Neither operators nor operands can require repetition.

$$V^* = (n_1^* + n_2^*) \log_2 (n_1^* + n_2^*)$$

Advantages of Halstead metrics :

1. Do not require in-depth analysis of programming structure.
2. Predicts rate of error.
3. Predicts maintenance effort.
4. Useful in scheduling and reporting projects.
5. Measure overall quality of programs.
6. Simple to calculate.
7. Can be used for any programming language.
8. Numerous industry studies support the use of Halstead in predicting programming effort and mean number of programming bugs.

Disadvantages of Halstead metrics :

1. It depends on the completed code.
2. It has little or no use of predictive estimating model. But McCabe's model is more signed to application at the design level.

Que 4.31. Calculate Halstead's measure on factorial code given below :

```
int fact (int n) {
if (n == 0)
{
(return 1 ;)
}
else
{return n * fact (n – 1) ; } }
```

Answer

The table factorial of operand and operators is as follows :

1. Program length $N = N_1 + N_2 = 18 + 8 = 26$
2. Vocabulary of program $n = n_1 + n_2 = 10 + 3 = 13$

Operator	Occurrence	Operand	Occurrence
int	2	n	4
( )	4	fact	2
if	1	1	2
;	2		
= =	1		
–	1		
return	2		
else	1		
*	1		
{ }	3		
$n_1 = 10$	$N_1 = 18$	$n_2 = 3$	$N_2 = 8$

3. Volume $V = N \log_2 n = 26 \log_2 13 = 96.211$

4. Estimated length $N' = n_1 \log_2 n_1 + n_2 \log_2 n_2 = 10 \log_2 10 + 3 \log_2 3$
$= 33.219 + 4.754 = 37.973$

5. Effort $E = \frac{n_1 N_2}{2n_2}(N \log_2 n) = \frac{10 \times 8}{2 \times 3}(26 \log_2 12)$
$= \frac{10 \times 8 \times 93.209}{6}$
$= 1242.787$

Que 4.32. Describe the function point based measures and metrics.

Answer

1. The collection of function point data has two primary motivations. One is the desire by managers to monitor levels of productivity, for example, number of function points achieved per working hour expended.
2. Another use of function points is in the estimation of software development cost which is the most important potential use of function point data.
3. Function point may be used to compute the following important metrics:

Productivity = FP / persons-months
Quality = Defects / FP
Cost = Rupees / FP
Documentation = Pages of documentation per FP

Thus, metrics are controversial and are not universally acceptable.

4. The five functional units are ranked according to their complexity, *i.e.*, Low, Average, or High, using a set of prescriptive standards.

5. Organizations that use FP methods develop criteria for determining whether a particular entry is Low, Average or High.
6. Nonetheless, the determination of complexity is somewhat subjective.
7. After classifying each of the five function types, the unadjusted function points (UFP) are calculated using predefined weights for each function type as given in Table 3.32.1.

Table 3.32.1.

Functional Units	Weighting factors		
	Low	Average	High
External Inputs (EI)	3	4	6
External Outputs (EO)	4	5	7
External Inquiries (EQ)	3	4	6
Internal Logical Files (ILF)	7	10	15
External Interface Files (EIF)	5	7	10

Table 3.32.2.

Functional Units	Count	Complexity	Complexity Total	Functional Unit Totals
External Inputs (EIs)		Low × 3 = Average × 4 = High × 6 =		
External Outputs (EOs)		Low × 4 = Average × 5 = High × 7 =		
External Inquiries (EQs)		Low × 3 = Average × 4 = High × 6 =		
Internal Logical Files (ILFs)		Low × 7 = Average × 10 = High × 15 =		
External Interface Files (EIFs)		Low × 5 = Average × 7 = High × 10 =		
Total unadjusted function point count =				

8. The weighting factors are identified as per Table 3.32.1 for all functional units and multiplied with the functional units accordingly.

9. The procedure for the calculation of UFP in mathematical form is given below :

$$UFP = \sum_{i=1}^{n} \sum_{j=1}^{3} Z_{ij} * W_{ij}$$

where i indicates the row and j indicates the column of Table 3.32.1.
W_{ij} : It is the entry of the i^{th} row and j^{th} column of the Table 3.32.1.
Z_{ij} : It is the count of the number of functional units of type i that have been classified as having the complexity corresponding to column j.

10. The final number of function points is arrived at by multiplying the UFP by an adjustment factor that is determined by considering 14 aspects of processing complexity.

11. In this, adjustment factor allows the UFP count to be modified by at most 35%. The final adjusted FP count is obtained by using the following relationship,

$$FP = UFP * CAF$$

where CAF is complexity adjustment factor and is equal to $[0.65 + 0.01 \times \Sigma F_i)$.

12. The F_i (i = 1 to 14) are the degrees of influence and can be calculated according to following sequence :
Rate each factor on a scale of 0 to 5.

1. Does the system require reliable backup and recovery ?
2. Is data communication required ?
3. Are there distributed processing functions ?
4. Is performance critical ?
5. Will the system run in an existing heavily utilized operational environment ?
6. Does the system require online data entry ?
7. Does the online data entry require the input transaction to be built over multiple screens or operations ?
8. Are the master files updated online ?
9. Is the inputs, outputs, files or inquiries complex ?
10. Is the internal processing complex ?
11. Is the code designed to be reusable ?
12. Are conversion and installation included in the design ?
13. Is the system designed for multiple installations in different organizations ?
14. Is the application designed to facilitate change and ease of use by user ?

Que 4.33. **Consider a project with the following parameters :**

1. **External inputs :**
   a. **10 with low complexity**
   b. **15 with average complexity**
   c. **17 with high complexity**
2. **External outputs :**
   a. **6 with low complexity**
   b. **13 with high complexity**
3. **External inquiries :**
   a. **3 with low complexity**
   b. **4 with average complexity**
   c. **2 with high complexity**
4. **Internal logical files :**
   a. **2 with average complexity**
   b. **1 with high complexity**
5. **External interface files :**
   a. **9 with low complexity**

In addition to above, system requires

1. **Significant data, communication**
2. **Performance is very critical**
3. **Designed code may be moderately reusable**
4. **System is not designed for multiple installations in different organizations.**

Other complexity adjustment factors are treated as average. Compute the function points for the project.

Answer

To calculate the function point firstly, we have to calculate the unadjusted functional point counts as follows :

Functional Units	Count	Complexity		Complexity Total	Functional Unit totals
External inputs (ELs)	10	Low × 3	=	30	192
	15	Average × 4	=	60	
	17	High × 6	=	102	
External outputs (EOs)	6	Low × 4	=	24	115
	0	Average × 5	=	0	
	13	High × 7	=	91	

External inquiries (EQs)	3	Low × 3	=	9	37
	4	Average × 4	=	16	
	2	High × 6	=	12	
Internal logical files (ILFs)	0	Low × 7	=	0	35
	2	Average × 10	=	20	
	1	High × 15	=	15	
External interface files (EIFs)	9	Low × 5	=	45	45
	0	Average × 7	=	0	
	0	High × 10	=	0	
	Total unadjusted function point count =				424

The factors given previously can be calculated as :

$$\sum_{i=1}^{14} F_i = 3+4+3+5+3+3+3+3+3+3+2+3+0+3 = 41$$

$$\begin{aligned} CAF &= (0.65 + 0.01 * \Sigma F_i) \\ &= (0.65 + 0.01 * 41) \\ &= 1.06 \\ FP &= UFP * CAF \\ &= 424 * 1.06 \\ &= 449.44 \end{aligned}$$

Que 4.34. What do you mean by cyclomatic complexity ? Explain.

OR

Describe the cyclomatic complexity measure and metrics.

Answer

1. The cyclomatic complexity is also known as structural complexity because it gives internal view of the code.
2. This approach is used to find the number of independent paths through a program. This provides us the upper bound for the number of tests that must be conducted to ensure that all statements have been executed at least once and every condition has been executed on its true and false side.
3. If a program has backward branch then it may have infinite number of paths. Although it is possible to define a set of algebraic expressions that gives the total number of possible paths through a program, however, using total number of paths has been found to be impractical.
4. Because of this, the complexity measure is defined in terms of independent paths that when taken in combination will generate every possible path.

5. An independent path is any path through the program that introduces at least one new set of processing statements or a new condition.
6. McCabe's cyclomatic metric, $V(G)$ of a graph G with n vertices, e edges, and P connected components is $V(G) = e - n + 2P$.
7. Given a program, we will associate with it a directed graph that has unique entry and exit nodes. Each node in the graph corresponds to a block of code in the program where the flow is sequential and the arcs correspond to branches taken in the program.
8. This graph is classically known as flow graph and it is assumed that each node can be reached by the entry node and each node can reach the exit node.
9. For example, a flow graph shown in Fig. 4.34.1 represents entry node '*a*' and exit node '*f*'.

The value of cyclomatic complexity can be calculated as :

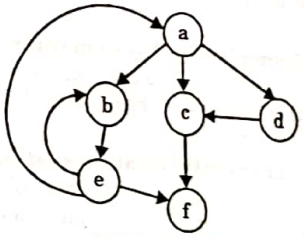


Fig. 4.34.1.

$$V(G) = 9 - 6 + 2 = 5$$

Here, $e = 9$, $n = 6$ and $P = 1$

There will be five independent paths for the flow graph illustrated in Fig. 4.34.1.

path 1 : a c f

path 2 : a b e f

path 3 : a d c f

path 4 : a b e a c f or a b e a b e f

path 5 : a b e b e f

Properties of cyclomatic complexity :

1. $V(G) >= 1$.
2. $V(G)$ is the maximum number of independent paths in graph G.
3. Inserting and deleting functional statements to G does not affect $V(G)$.
4. G has only one path if and only if $V(G) = 1$.
5. Inserting a new row in G increases $V(G)$ by unity.
6. $V(G)$ depends only on the decision structure of G.

Que 4.35. **Why it is difficult to validate the relationships between internal product attributes such as cyclomatic complexity and external attributes such as maintainability ? Discuss.**

UPTU 2012-13, Marks 10

Answer

1. Predicting the number of change requests for a system requires an understanding of the relationship between the system and its external environment.
2. Some systems have a very complex relationship with their external environment and changes to that environment inevitably result in changes to system.
3. To evaluate the relationships between system and its environment, we assess :
   a. **Number and complexity of system interfaces :** The larger the number of interfaces and more complex these interfaces, the more likely it is that interface changes will be required as new requirements are proposed.
   b. **Number of inherently volatile system requirements :** Requirement that reflect organizational policies and procedures are likely to be more volatile than requirements that are based on stable domain characteristics.
4. The relationship between program complexity, as measured by metrics such as cyclomatic complexity and maintainability is that the more complex a system or component, the more expensive it is to maintain.
5. Complexity measurements are particularly useful in identifying program components that are likely to be expensive to maintain.
6. To reduce maintenance costs, therefore, we should try to replace complex system components with simpler alternatives.
7. After a system has been put into service, we are able to use process data to predict maintainability.

PART-3

The SEI Capability Maturity Model (CMM), SQA Activities, Formal SQA Approaches : Proof of Correctness, Statistical Quality Assurance, Cleanroom Process.

CONCEPT OUTLINE : PART-3

- The Capability Maturity Model (CMM) is a strategy for improving the software process, irrespective of actual life cycle model used.

- Software quality assurance is defined as a planned and systematic approach to the evaluation of quality.
- SQA plan is a standard activity which ensures the quality goals of the organization.
- The cleanroom strategy is a software development process intended to produce software with a certifiable level of reliability.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 4.36. **What do you understand by ISO 9000 certificate ?**

Answer

ISO 9000 certification :

1. ISO 9000 certification serves as reference for contact between independent parties.
2. The ISO 9000 standard specifies the guidelines for maintaining a quality system.
3. The ISO standard mainly addresses operational aspects and organizational aspects such as responsibility, reporting etc.
4. In a nutshell, ISO 9000 specifies a set of guidelines for repeatable and high quality development.
5. It is important to realize that ISO 9000 standard is a set of guidelines for the production process and is not directly concerned with product itself.
6. **ISO 9000 is a series of three standards :** ISO 9001, ISO 9002 and ISO 9003. The different types of software industries to which the different ISO standard apply are as follows :
   a. **ISO 9001 :** This standard applies to the organizations engaged in design, development, production and servicing of goods. This is the standard that is applicable to most software development organizations.
   b. **ISO 9002 :** This standard applies to those organizations which do not design products but are only involved in production. Like category of industries include steel and car manufacturing industries who buy the products and plant design from external sources and only involved in manufacturing those products. Therefore, ISO 9002 is not applicable to software development organizations.
   c. **ISO 9003 :** This standard applies to organizations involved only in installation and testing of the product.

Benefits of ISO 9000 certification: Some basic benefits of ISO 9000 certification are as follows :

1. Continuous improvement : Each procedure and work instruction must be documented to become spring board for continuous improvement.
2. Eliminate verification.
3. Higher real and perceived quality.
4. Boost employee morale.
5. Improved customer satisfaction.
6. Increased employee participation.
7. Better product and services.
8. Greater quality assurance.
9. Improved profit levels.
10. Improved communication.
11. Reduced cost.
12. Competitive edge.

Limitations of ISO 9000 certification : Though ISO 9000 has proved to be very effective for software development organizations, however, it also suffers from several limitations.

1. ISO 9000 does not automatically lead to total quality management (TQM) *i.e.*, continuous improvement.
2. ISO 9000 does not provide any guideline for defining an appropriate process.
3. ISO 9000 certification process is not full proof and thus variation in the certification norms may exist.

Que 4.37. Explain SEI capability maturity model.

UPTU 2012-13, Marks 05

OR

Describe the term : SEI-CMM. **UPTU 2013-14, Marks 05**

Answer

1. The capability maturity model (CMM) is not a software life cycle model. Instead, it is a strategy for improving the software process, irrespective of actual life cycle model used.
2. The CMM has been developed by Software Engineering Institute (SEI) at Carnegie Mellon University around 1987 and is still undergoing the process of refinement.
3. CMM is used to judge the maturity of the software processes of an organization and to identify the key practices that are required to increase the maturity of these processes.

4. The model is based on best practices followed in the organizations world wide. The five CMM level of maturity are :

i. **Level 1: Initial :**

a. At the level 1, the processes followed by the organizations are not well defined.

b. They are immature. As a result, a stable environment is not available for software development.

c. Further, success and failure of any project depends on the team member's competence.

d. There is no basis for predicting product quality, time for completion etc. No KPA's are defined at level 1.

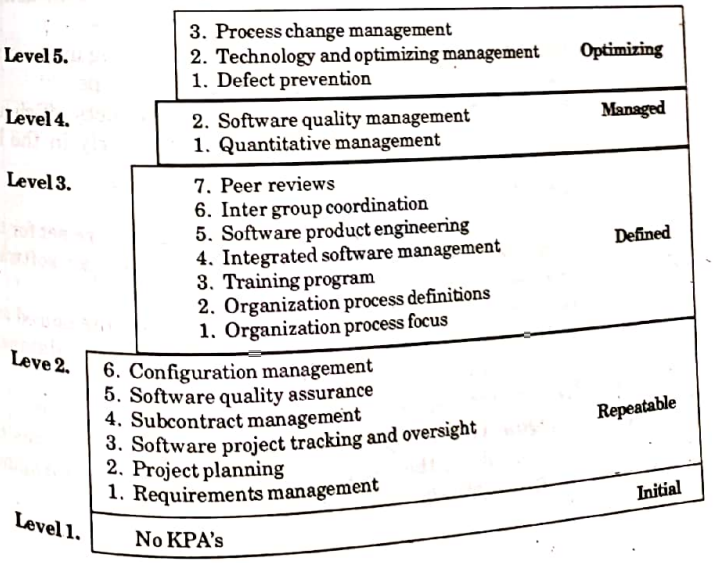
ii. **Level 2 : Repeatable :**

a. The level 2 focuses on establishing basic project management policies. In this, experience with earlier projects is used for managing new projects of similar nature.

b. The managers are able to predict the cost and schedule of the project based on earlier experience. At this level, organization can be called a disciplined and organized organization.

c. The key process areas to achieve this level of maturity are :

1. **Requirement management :** Establishes common understanding of the project established between the developer and customer by documenting the requirements.
2. **Project planning :** Establishes plans for performing the software engineering and for managing the software project, by defining resources, goals, constraints etc.



3. **Software project tracking and oversight :** Establishes people's visibility into actual progress of software projects.
4. **Subcontract management :** Focuses on selection of qualified software contractors and their effective management.
5. **Software quality assurance :** Provides adequate visibility into software project process, and the product being built by the management by following suitable quality standards.
6. **Configuration management :** Focuses on maintaining the integrity of all the products of the software project throughout the life cycle of the project.

iii. **Level 3 : defined :** At this level standard process to be used across the organization are defined and documented. Standardization of organization wide processes helps the manager and other team members to perform efficiently. Compared to level 2 defects are down 20 %, project cycle wise is down 10 %, the main features at level 3 are :

1. **Organization process focus :** Focuses on improvement of organization's overall software process capabilities.
2. **Training program :** Focuses on training for improving knowledge, skills and efficiency of individuals.
3. **Organization process definition :** Focuses on development and maintenance of organization standard process.
4. **Integrated software management :** Focuses on integration of software engineering and management activities into well defined coherent software process.
5. **Software product engineering :** Focuses on well defined engineering process and integrating all activities.
6. **Inter group coordination :** Focuses on planning interactions and technical interfaces between different groups.
7. **Peer reviews :** Focuses on removing the defects efficiently from the different software work products early in the life cycle.

iv. **Level 4 : Managed :**

a. At this level of CMM, quantitative quality goals are set for the organization for the software product as well as software processes.

b. Here quality and productivity of the process are measured and maintained by organization, wide software process database and feedback is given to management.

c. The key processes associated at management level are :

1. **Quantitative process management :** Focuses on controlling the software project process performance quantitatively.

2. **Software quality management :** Establishing strategies and plans for quantitative understanding of software project.

v. **Level 5 : Optimizing :** It focuses on continuous process improvement in organization using quantitative feedback. The main features at level 5 are :

1. **Defect prevention :** Focuses on identification of causes of defects and to prevent them from recurring in future projects by improving project defined process.
2. **Technology change management :** Focuses on identification of new technology and transferring them into an organization in systematic way to improve quality of product.
3. **Process change management :** Focuses on continuous improvement of software processes to improve productivity, quality and cycle time.

Que 4.38. Write a short note on SEI Capability Maturity Model (CMM). How does it differ from ISO 9000 ? **UPTU 2014-15, Marks 10**

Answer

SEI-CMM : Refer Q. 4.37, Page 158C, Unit-4.

ISO9000 v/s CMM :

S. No.	ISO	CMM
1.	ISO talks about minimum requirement for acceptable quality system.	CMM focuses on continuous software process improvement.
2.	ISO is a way to communicate the process.	CMM is a way to communicate capabilities.
3.	ISO9000 procedures describe a (possibly) definite development process but gives no indication of the likely quality of the designs or whether multiple software efforts are likely to produce software of similar quality.	The CMM is a very specific way of classifying an organizations software development methods. In a certain way, it tells how the quality of its software design is likely to be repeated.
4.	ISO basically used in manufacturing and production.	SEI CMM was developed specifically for software industry.

Que 4.39. **How would you define 'software quality' ? What do you mean by software quality assurance ? Also, discuss the various software quality activities.** **UPTU 2013-14, Marks 10**

Answer

Software quality : Refer Q. 4.26, Page 142C, Unit-4.

Software quality assurance :

1. To ensure that final product produced of high quality, some quality control activities must be performed throughout the development process.
2. If it is not done, correcting errors in the final stage can be very expensive, especially if they are originated in early phases.
3. The purpose of software quality assurance plan (SQAP) is to specify all the work products that are needed to be produced during the project, activities that need to be performed for checking the quality of each of work products and the tools and methods that may be used for the SQA activities.
4. Note that SQAP takes a broad view of quality. It is interested in the quality of not only final product but also the intermediate products ; even though in a project we are ultimately interested in the quality of the delivered product.
5. To ensure that the delivered software is of good quality, it is essential to make sure that the requirements and design are also of good quality.
6. For this reason, an SQAP will contain QA activities carried out throughout the project.
7. The SQAP specifies the tasks that need to be undertaken at different times in the life cycle to improve software quality and how they are to be managed.
8. These tasks generally include reviews and audits.
9. Each task should be defined with an entry and exit criterion, that is, the criterion that should be satisfied to initiate the task and the criterion that should be satisfied to terminate the task.
10. Both criteria should be stated so that they can be evaluated objectively. The responsibilities for different tasks should be identified.
11. The software quality assurance plan is developed during project planning and is reviewed by all interested parties.
12. Quality assurance activities performed by the software engineering team and the SQA group are governed by the plan.
13. The plan identifies :
    a. Evaluation to be performed.

    b. Audits and reviews to be performed.
    c. Standards those are applicable to the project.
    d. Procedures for error reporting and tracking.
    e. Documents to be produced by the SQA group.
    f. Amount of feedback provided to the software project team.

Major activities of SQA are :

1. **Application of technical methods :** Software quality is designed into the software but not added afterwards. It must have set of technical methods and tools to help analyst to generate high quality specifications and designs.
2. **Formal technical reviews :** A formal technical review is a meeting conducted by technical staff to uncover quality problems.
3. **Software testing :** Appropriate strategies for software testing should be applied.
4. **Enforcement of standards :** There are several quality standards such as ISO 9000, SEI CMMs, etc. meant for ensuring software quality. If these standards are used, they can be applied by developers as part of a formal review.
5. **Control of change :** Changes can introduce errors. Change control includes :
    a. Formalised requests for change.
    b. Evaluates the nature of the change.
    c. Controls the impact of the change.
6. **Measurement :** Software metrics are needed to track quality and assess impact of methodological procedural changes.
7. **Record keeping and reporting :** Collection and dissemination of software quality assurance information is required. Results of audits, reviews, change control, testing and other SQA activities are part of the historical record of the project.

Que 4.40. **Develop your own metrics for correctness, maintainability, integrity and usability of software. What is Statistical Quality Assurance (SQA) ?** **UPTU 2014-15, Marks 10**

Answer

Correctness : The extent to which a program satisfies its specification and fulfills the customer are the main objectives of correctness.

Maintainability : The effort required to learn, operate, prepare input and interpret output of a program.

Integrity : The extent to which access to software or data by unauthorized persons can be controlled.

Usability : The extent to which a program can be used in other applications also.

Statistical quality assurance :

Statistical quality assurance reflects a growing trend throughout industry to become more quantitative about quality. For software, statistical quality assurance implies the following steps :

1. Information about software errors and defects is collected and categorized.
2. An attempt is made to trace each error and defect to its underlying cause (e.g., nonconformance to specifications, design error, violation of standards, poor communication with the customer).
3. Using the Pareto principle (80 percent of the defects can be traced to 20 percent of all possible causes), isolate the 20 percent (the vital few).
4. Once the vital few causes have been identified, move to correct the problems that have caused the errors and defects.

For example :

1. To illustrate the use of statistical methods for software engineering work, assume that a software engineering organization collects information on errors and defects for a period of one year.
2. Some of the errors are uncovered as software is being developed. Others (defects) are encountered after the software has been released to its end users.
3. Although hundreds of different problems are uncovered, all can be tracked to one (or more) of the following causes :
   a. Incomplete or erroneous specifications (IES)
   b. Misinterpretation of customer communication (MCC)
   c. Intentional deviation from specifications (IDS)
   d. Violation of programming standards (VPS)
   e. Error in data representation (EDR)
   f. Inconsistent component interface (ICI)
   g. Error in design logic (EDL)
   h. Incomplete or erroneous testing (IET)
   i. Inaccurate or incomplete documentation (IID)
   j. Error in programming language translation of design (PLT)
   k. Ambiguous or inconsistent human/computer interface (HCI)
   l. Miscellaneous (MIS)

Que 4.41. What do you understand by the term SQA plans ?

Answer

SQA plan is a standard activity which ensures the quality goals of the organization. It gives details and templates on all activities, which have become part of the standard implementation.

The SQA plan include the following :

1. Project planning
2. Models of data, classes and objects, processes, design architecture
3. SRS
4. Test plans for testing SRS
5. User manuals online help etc.
6. Audit and review

In SQA plan, mainly three kinds of activities are performed :

1. Organization specific
2. Software specific
3. Customer specific

Que 4.42. Explain the following formal SQA approaches :

1. **Proof of correctness**
2. **Statistical quality assurance**
3. **Cleanroom process**

OR

Describe the term cleanroom process. **UPTU 2013-14, Marks 05**

Answer

1. Proof of correctness :

A proof of correctness is a mathematical proof that a computer program or a part thereof will, when executed, yield correct results *i.e.*, results fulfilling specific requirements. Before proving a program correct, the theorem to be proved must, of course, be formulated.

1. Treat a program as a mathematical object.
2. Developed with a language with a rigorous syntax.
3. Are attempts at developing a rigorous approach to specification of software requirements.
4. With both can attempt to develop a mathematical proof that a program conforms exactly to its specification.
5. In the code, can at selected statements formulate assertions on the set of correct values for program variables.
6. Can then show that the statements between these assertions do the correct transformation of the values in the assertions.

2. **Statistical quality assurance :** Refer Q. 4.40, Page 163C, Unit-4.

3. **Cleanroom process :**

a. The cleanroom software engineering process is a software development process intended to produce software with a certifiable level of reliability.

b. The cleanroom process was originally developed by Harlan Mills and several of his colleagues including Alan Henver at IBM.

c. The focus of the cleanroom process is on defect prevention, rather than defect removal.

d. The basic principles of the cleanroom process are :

1. **Software development based on formal methods :** Cleanroom development makes use of the box structure method to specify and design a software product. Verification that the design correctly implements the specification is performed through team review.

2. **Incremental implementation under statistical quality control :**

a. Cleanroom development uses an iterative approach, in which the product is developed in increments that gradually increase the implemented functionality.

b. The quality of each increment is measured against pre-established standards to verify that the development process is proceeding acceptably.

c. A failure to meet quality standards results in the cessation of testing for the current increment, and a return to the design phase.

3. **Statistically sound testing :**

i. Software testing in the cleanroom process is carried out as a statistical experiment.

ii. Based on the formal specification, a representative subset of software input/output trajectories is selected and tested.

iii. This sample is then statistically analyzed to produce an estimate of the reliability of the software, and a level of confidence in that estimate.

VERY IMPORTANT QUESTIONS

Following questions are very important. These questions may be asked in your SESSIONALS as well as UNIVERSITY EXAMINATION.

Q. 1. What is software testing ? Explain testing objectives and principle.
Ans. Refer Q. 4.1 and Q. 4.2.

Q. 2. Discuss test plans and test cases.
Ans. Refer Q.4.5 and 4.7.

Q. 3. What are various levels of testing and types of testing ?
Ans. Refer Q.4.8.

Q. 4. Write a short note on test strategies.
Ans. Refer Q. 4.23.

Q. 5. Explain the terms "software verification" and "software validation".
Ans. Refer Q. 4.24.

Q. 6. Define the term :
a. Software quality
b. Software quality assurance
c. Software quality metrics
Ans.
a. Refer Q. 4.26.
b. Refer Q. 4.40.
c. Refer Q. 4.28.

Q. 7. Write a note on CMM model.
Ans. Refer Q. 4.37.

Q. 8. Explain cleanroom process.
Ans. Refer Q. 4.42

☺☺☺

5 UNIT

Project Management

PART-1

Software Configuration Management : Software Configuration Items and Tasks, Baselines, Plan for Change, Change Control, Change Request Management and Version Control.

CONCEPT OUTLINE : PART-1

- Software Configuration Management ensures that all people involved in the software process know what is being designed, developed, built, tested and delivered.
- Four major SCM activities are :
  a. Configuration identification
  b. Change control
  c. Software configuration status reporting
  d. Audits and reviews
- A configuration item is any component of an information technology infrastructure, that is, under the control of configuration management.
- The change request management is the process of requesting, determining attainability, planning, implementing and evaluation of change to a system.
- Version control combines procedure and tool that manages different versions of configuration items.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 5.1. What do you understand by Software Configuration Management (SCM) ? Explain the goal and objectives of SCM.

UPTU 2013-14, Marks 10

Answer

1. Software Configuration Management (SCM) is one of the foundations of software engineering. It is used to track and manage the emerging product and its versions.
2. This is to assure quality of the product during development, and operational maintenance of the product.
3. SCM ensures that all people involved in the software process know what is being designed, developed, built, tested, and delivered.

4. Through SCM, the design requirements can be traced to the final software product.
5. Thus, the Software Configuration Management (SCM) can be defined as a process of defining and implementing a standard configuration, which results into the primary benefits such as easier setup and maintenance, less down-time, better integration with enterprise management, and more efficient and reliable backups and also maximize productivity by minimizing mistakes.

Goals of software configuration management :

a. Software configuration management activities are planned.
b. Selected software work products are identified, controlled, and available.
c. Changes to identified software work products are controlled.
d. Affected groups and individuals are informed of the status and content software baselines.

Objectives of software configuration standards :

i. Remote system administration :

a. The configuration standard should include necessary software and/or privileges for remote system administration tools.
b. A remote administration client, that is, correctly installed and configured on the client side is the cornerstone of the remotely administered network.
c. These remote tools can be used to check the version of virus protection, check machine configuration, or offer remote help-desk functionality.

ii. Reduced user down-time :

a. A great advantage of using a standard configuration is that system becomes completely interchangeable resulting in reduced user down-time.
b. If a given system experiences an unrecoverable error, an identical new system can be dropped into place.
c. User data can be transferred if the non-functional machine is still accessible, or the most recent copy can be pulled off the backup tape with the ultimate goal being that the user experiences little change in the system interface.

iii. Reliable data backups :

a. Using a standard directory for user data allows backup systems to selectively backup a small portion of a machine, greatly 'reducing the network traffic and tape usage for backup systems.
b. Also, should a catastrophic failure occur, the data directory could be restored to a new machine with little time and effort.

c. A divided directory structure, between system and user data, is one of the main goals of the configuration standard.

iv. Easy workstation setup :

a. Any sort of standardized configuration streamlines the process of setting up the system and insures that vital components are available.
b. If multiple machines are being setup according to a standard setup, most of the setup and configuration can be automated.

v. Multi-user support :

a. Although, it is common for users to share a workstation, the system configuration is designed to allow multiple users to use the same workstation without interfering each other's work.
b. Some software packages do not support completely independent settings for all users, however, users can have independent data areas.
c. The directory structure implemented does not impose limits on the number of independent users a system, can have.

vi. Remote software installation :

a. Most modern software packages are installed in factory pre-defined directories. While software installed in the manner will function correctly for a single user, it will lead to non-uniform configuration among a collection of machines.
b. A good configuration standard will have software installed in specified directory areas to logically divide software on the disk.
c. This will lead to easier identification of installed components and the possibility of automating installation procedures through the use of universal scripts.
d. With software installed into specific directories, maintenance and upgraded running software becomes less complex.

Que 5.2. Give the format of Software Configuration Management (SCM) plan.

Answer

The procedures for software configuration management are laid down in the form of a SCM plan document. Sample structure of a configuration management plan (source IEEE standard for software configuration management plans) is given here :

i. Introduction :

a. Purpose
b. Scope
c. Definitions and acronyms
d. References

ii. **Management :**
   a. Organizations
   b. Configuration management responsibilities
   c. Interface control
   d. Implementation of software
   e. Configuration of software
   f. Applicable policies, directives and procedures

iii. **Configuration management activities :**
   a. Configuration identification
   b. Configuration control
   c. Configuration status accounting
   d. Audits and reviews

iv. Tools, techniques and methodologies

v. Supplier control

vi. Records collection and retention

Que 5.3. Illustrate the various SCM activities.

Answer

Following are the four major SCM activities :

1. **Configuration identification :**
   a. SCM process manages all the software entities and their related representations in documentation.
   b. Basically, SCM should manage all software related components that are used during development, testing, and production.
   c. The identification and structuring of the software configuration management plan is done when the software process is being defined.
   d. Software configuration items (SCIs) added to the SCM are things such as tools, design documents, requirements documents etc.
   e. The SCI that are identified, determine the baseline(s), that is, associated with the project.
   f. The number and types of baselines will depend upon the project. Baselines are the core of SCM, they provide a stable platform for work to continue from.
   g. Only authorized changes can be made to the baseline, and all changes are recorded as deltas until such time as a new baseline is generated.

2. **Change control :**
   a. Change control involves procedures and tools to bring in order to change process.
   b. Larger projects have a formal change control board (CCB), whose responsibility is to review and approve and disapprove changes.
   c. It is the CCB responsibility to provide the mechanism to maintain orderly change process.
   d. The CCB includes the following individuals :
      i. Executive sponsor
      ii. Customer representative
      iii. Marketing representative
      iv. Program manager
      v. Project manager
      vi. Quality assurance manager
      vii. Documentation leader

3. **Software configuration status reporting :**
   a. Software configuration status reporting (or accounting) is a record keeping activity of software configuration management.
   b. From the time the first SCI's were identified, all changes and the current status of changes and documents are recorded in a status accounting database.
   c. The record in the database provides management with reports as to the current state of the project.
   d. The configuration status accounting database holds the records showing the products history, how the product has evolved, and where the system is at anytime in the relation to the current baseline.
   e. Administrator uses status accounting to track and report on all SCI's formally identified and controlled. The status accounting database also maintains records to supports SCM auditing.

4. **Audit and reviews :**
   a. Audits and reviews are used to ensure that changes have been properly implemented. The formal review looks at the technical correctness of any SCI that has been modified.
   b. An SCI is looked as to determine any omissions, potential side effects, and its consistency with other SCIs.
   c. Formal reviews are conducted for all but the most trivial changes.
   d. Auditing gives us a picture of how close the current software system mirrors the software systems pictured in the baseline and the requirements documents.

e. Auditing provides the mechanism for establishing a new baseline. The final stages of an audit are used to sanction the new baseline.

f. Auditing increases the software visibility and established traceability. It helps to avoid costly re-design and refits.

Que 5.4. Explain software configuration items, tasks and baselines. **UPTU 2014-15, Marks 10**

Answer

Configuration item :

1. A configuration item (CI) is any component of an information technology infrastructure, that is, under the control of configuration management.
2. Configuration items (CIs) can be individually managed and versioned, and they are usually treated as self-contained units for the purposes of identification and change control.
3. All configuration items (CIs) are uniquely identified by names, version numbers, and other attributes. The lowest level CI is usually the smallest unit that will be changed independently of other components.
4. CIs vary in complexity, size, and type. They can range from an entire service which may consist of hardware, software and documentation to a single program module or a minor hardware component.

Configuration tasks : Refer Q. 5.3, Page172C, Unit-5.

Configuration baselines :

1. Configuration Management (CM) plans, establish and document the requirements, standards, practices, and procedures for configuration management.
2. This includes defining baselines and establishing the labeling scheme for configuration items.
3. A baseline is a group of configuration items (products, deliverables) developed during a specific phase of the development process that has been formally accepted.
4. Once the baseline is established, changes to the items can only be done through a formal change process.
5. In other words we can say, a software baselines library is established containing the software baselines as they developed.
6. Changes to baselines and the release of software products built from the software baseline library are systematically controlled via the change control and configuration auditing functions of software configuration management.

Que 5.5. What is change and change control process ?

Answer

Change : Change is defined as anything hardware, software, system components, services, documents, or processes that is deliberately introduced into the production environment and which may affect a Service Level Agreement (SLA) or otherwise affect the functioning of the environment or one of its components.

Change control process :

1. Change control procedure ensures that the changes to the system are controlled and that their effect on the system can be predicted.
2. It will be appropriate if changes to software can be predicted. Change control process comes into effect when the software and associated documentation are delivered to configuration management change request form (also known as change control form or software problem report), which should record the recommendations regarding the change.
3. The recommendations may include assessment of the proposed change, the estimated costs and how the change should be implemented.
4. This form is submitted to a Change Control Authority (CCA), which decides whether or not the change is to be accepted. If change is approved by the CCA, it is applied to the software.
5. The revised software is revalidated by the Software Quality Assurance (SQA) team to ensure that the change has not adversely affected other parts of the software.
6. The changed software is handed over to the software configuration team and is incorporated in a new version of the system.

Que 5.6. What is change management ?

Answer

1. Change management is a structured approach to shifting/transitioning individuals, teams, and organizations from a current state to a desired future state.
2. Successful change management is more likely to occur if the following are included :
   a. Benefits management and realization to define measurable stakeholder aims, create a business case for their achievement (which should be continuously updated), and monitor assumptions, risks, dependencies, costs, return on investment, dis-benefits and cultural issues affecting the progress of the associated work.
   b. Effective communications that informs various stakeholders of the reasons for the change (why ?), the benefits of successful implementation (what is in it for us) as well as the details of the change (when ? where ? who is involved ? how much will it cost ? etc.).

c. Devise an effective education, training and/or skills upgrading scheme for the organization.

d. Counter resistance from the employees of companies and align them to overall strategic direction of the organization.

e. Provide personal counseling (if required) to alleviate any change related fears.

f. Monitoring of the implementation and fine-tuning as required.

Que 5.7. Write a short note on change request management.

Answer

1. A change request is considered to be addition, deletion, or modification to the scope as was defined in the contract.
2. A change request usually results in change in baselined plan.
3. A baselined plan consists of :
   a. negotiated scope,
   b. quality of work products,
   c. effort and cost for the project, and
   d. timeline for delivery.
4. When customer sends a change request, asking for additional modified features for the application, or asking in schedule changes, a change request is logged.
5. Usually, a unique number is given to this request and logged by the project manager.
6. An impact analysis is done to determine the impact of change request on configuration items.
7. A list of impacted configuration items is prepared. These configuration items include programs as well as documents that need to be modified to take care of change request. Along with these, cost effort and schedule impact are also analyzed.
8. Change request management has two important processes :
   a. negotiation with the customer after the impact analysis and
   b. executing change request process.
9. Once impact analysis has been done, customer has to be shared with information related to extra cost, schedule, and work products that need to be changed (some of these work products could have been delivered earlier).
10. These changes are given in the form of a proposal for getting customer's approval and also project management plan is reflected with these changes.

11. All these are usually authorized by the project manager as he/she is answerable to the top management for meeting project objectives (such as schedule, effort, quality, etc.).
12. The change request is recorded so that in future reference can be given for the same.
13. The steps involved in change request logging are as follows : log the change request in a standard template, carry out impact analysis, find impact on effort and schedule, get approval from customer for extra cost and modified schedule of deliveries, and then start working on the change request.
14. A change request log is maintained to keep track of each change request. The tracker uses unique change request number to track change requests.
15. Once the change request has been accepted and negotiation with customer for effort and schedule is completed, execution starts.

Que 5.8. Explain initiate change and authorizing change request.

Answer

Initiate change :

1. The objective of initiate change is to formally initiate a change through the submission of a Change Request (CR).
2. Change requests may be initiated :
   a. Before a project is completed *i.e.*, a project change request made after a project has started but before it has completed and transitioned to production.
   b. After a project has moved to production *i.e.*, a production change request.
   c. By any member of the project team or any other stakeholder with a vested interest in the project.
3. The change request management process ensures that change requests are created with consistent quality and completeness and discards irrelevant requests.
4. Although a project change request can be requested by any stakeholder, it is typically initiated and submitted by one of the project team members who submit requests for changes relating to its area of responsibility.
5. For example, the data movement team may initiate database change requests required to satisfy new or changed data sources.

Authorizing change request :

1. The objective of authorize change requests is to establish a formal process for approving change.

2. A Change Management Team (CMT) should be established and responsible for reviewing change requests and voting on the changes according to predefined voting logic.
3. The following change request management process is required to authorize a change request :
   a. **Authorize standard change request :** A standard changes do not require CMT approval since they are automatically approved and move directly to change deployment.
   b. **Authorize minor change request :** The change manager is responsible for determining if the change is minor and, if so, may approve it or submit it to the CMT for review/approval.
   c. **Authorize significant or major change request :** The change manager is responsible for submitting all significant and/or major changes to the CMT. For approval, the CMT shall vote to :
      i. Accept the change.
      ii. Reject the change.
      iii. Escalate the change to senior management.
   d. **Log authorization :** The change manager shall update change log with the CMT decision (status) and status date.

Que 5.9. What do you mean by version control and explain the activities participating in version control ?

Answer

1. Version control combines procedure and a tool that manages different versions of configuration items and that are created during the software engineering process.
2. A version control tool is the first stage towards being able to manage multiple versions. Once it is in place, a detailed record of every version of the software must be kept.
3. This comprises the name of each source code component, including the variations and revisions. Each version of software is a collection of software configuration items (source code, documentation and data) and each version may be composed of different variants.
4. Version control activity is split into mainly four sub-activities :
   a. **Identifying new version :**
      i. A software configuration items will receives a new version number when there has been a change to its established baseline.
      ii. Each previous version will be stored in a corresponding directory such as version0, version1, version2 etc.

   b. **Numbering scheme :**
      i. The numbering scheme will have the following format : Version X.Y.Z
      ii. The first letter (X) represents the entire SCI. Therefore changes made to the entire configuration items or changes large enough to warranty a completely new release of items will causes the first digit to increase.
      iii. The second letter (Y) represents a component of SCI. The digit will sequentially increase if a change is made to a component, or small changes to multiple components.
      iv. The third letter (Z) represents a section of components of SCI. This number will only be possible if component of an SCI can be broken down into individual sections.
   c. **Visibility :**
      i. The version number will be visible either in a frame or below the title.
      ii. The decision for this depends upon the group decision to code all the documents for a frame capable browser or allow for non frame capable browser.
      iii. In either case, number will always be made available.
   d. **Tracking :** The best way to keep track of the different versions is with a version evolution graph which shown in Fig. 5.9.1.

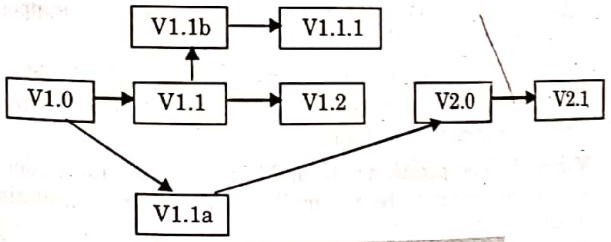


Fig. 5.9.1. Version evolution graph.

Que 5.10. Describe two ways in which system building tools can optimize the process of building a version of a system from its components. **UPTU 2012-13, Marks 10**

Answer

1. System building is a process of assembling program components, data and libraries and then compiling and linking these to create an executable system.

2. System building tools and version management tools must communicate as the build process involves checking out component versions from repository managed by the version management system.
3. The configuration description used to identify a baseline is also used by the system building tool.
4. System building is a complex process, which is potentially error-prone, as there may be different platforms involved such as :
   a. The development system, which includes development tools such as compilers, source code editors etc. Developers check out code from the version management system into a private workspace before making changes to the system. They may wish to build a version of a system for testing in their development environment before committing changes that they have made to the version management system.
   b. The build server, which is used to build definitive, executable versions of the system. This interacts closely with the version management system. Developers check in code to the version management system before it is built. The system built may rely on external libraries.
5. There are many build tools available and build system may provide following features :
   a. **Build script generation :** If necessary, the build system should analyze the program that is being built, identify dependent components, and automatically generate a build script (sometimes called a configuration file). The system should also support the manual creation and editing of build scripts.
   b. **Version management system integration :** The build system should check out the required versions of components from the version management system.
   c. **Minimal recompilation :** The build system should work out what source code needs to be recompiled and set up compilations if required.
   d. **Executable system creation :** The build system should link the compiled object code files with each other and with other required files, such as libraries and configuration files, to create an executable system.
   e. **Test automation :** Some build systems can automatically run automated tests using test automation tools such as JUnit. These check that the build has not been 'broken' by changes.
   f. **Reporting :** The build system should provide reports about the success or failure of the build and the test that have been run.
   g. **Documentation generation :** The build system may be able to generate release notes about the build and system help pages.

PART-2

Risk Management : Risks and Risk Types, Risk Breakdown Structure (RBS), Risk Management Process : Risk Identification, Risk Analysis, Risk Planning, Risk Monitoring and Cost Benefits Analysis.

CONCEPT OUTLINE : PART-2

- Software risk is a problem that could cause some loss or threaten the success of a software project, but which has not happened yet.
- Types of risks :
  a. Estimation errors
  b. Planning assumptions
  c. Eventualities
- Risk management involves basically two steps :
  a. Risk assessment
  b. Risk control
- Risk planning is to identify strategies to deal with risk.
- Risk monitoring is the continually reassessing of risk as the project proceeds and conditions change.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 5.11. Define risk. What are various categories of risks ?

Answer

1. Software risk is a problem that could cause some loss or threaten the success of a software project, but which has not happened yet.
2. These potential problems might have an adverse impact on the cost, schedule, or technical success of the software project, the quality of software products, or project team morale.
3. Software developers are extremely optimists. When planning software projects, we often assume that everything will go exactly as planned.
4. Alternatively, we take the other extreme position. The creative nature of software development means we can never accurately predict what is going to happen, so what is the point of making detailed plans ?

5. Both these perspectives can lead to software surprises, when unexpected things happen that throw the project completely out of track. Software surprises are never good news.
6. Risk management is becoming recognized as an important area in the software industry to reduce this surprise factor.
7. Risk management means dealing with a concern before it becomes crisis. Therefore, most of the software development activities include risk management as a key part of the planning process and expect the plan to highlight the specific risk areas.
8. The object planning is expected to quantify both probability of failure and consequences of failure and to describe what will be done to reduce the risk.

Risks can be categorized as follows :

a. **Project risks :** Risks that threaten the project (or the project schedule).
b. **Product risks :** Risks that threaten the quality of the software developed.
c. **Business risks :** Risks that threaten the development (or client) organization.

Que 5.12. List and discuss some of the points specific for identifying the risks during software development. Also, give some of the category of risks that are to be identified.

UPTU 2014-15, Marks 10

Answer

For the purpose of identifying and managing those risks that may cause a project to overrun its time scale or budget, it is convenient to identify three types of risk :

1. Those caused by the inherent difficulties of estimation.
2. Those due to assumptions made during the planning process.
3. Those of unforeseen (or at least unplanned) events or hazards occurring.

1. Estimation errors :

a. Some tasks are harder to estimate than others because of the lack of experience of similar tasks or because of the inherent nature of the task.
b. Producing a set of user manuals is reasonably straightforward and, given that we have carried out similar tasks previously, we should be able to estimate with some degree of accuracy how long the task will take and how much it will cost.
c. On the other hand, the time required for program testing and debugging might be difficult to predict with a similar degree of accuracy, even if we have written similar programs in the past.

d. Estimation can be improved by analyzing historic data for similar activities and similar systems. Keeping records comparing our original estimates with the final outcome will reveal the type of tasks that are difficult to estimate correctly.

2. Planning assumptions :

a. At every stage during planning, assumptions are made which (if not valid) may put the plan at risk.
b. Our activity network is likely to be built on the assumption of using a particular design methodology which may be subsequently changed.
c. We generally assume that, following coding, a module will be tested and then integrated with others.
d. We might not plan for module testing showing up the need for changes in the original design but, in the event, it might happen.
e. At each stage in the planning process, it is important to list explicitly all of the assumptions that have been made and identify what effects they might have on the plan if they are inappropriate.

3. Eventualities :

a. Some eventualities might never be foreseen and we can only resign ourselves to the fact that unimaginable things do, sometimes, happen. They are, however, very rare.
b. The majority of unexpected events can, in fact, be identified.
c. The requirements specification might be altered after some of the modules have been coded, the senior programmer might take maternity leave, the required hardware might not be delivered on time.
d. Such events do happen from time to time and, although the likelihood of anyone of them happening during a particular project may be relatively low, they must be considered and planned for.

Que 5.13. Write a short note on Risk Breakdown Structure (RBS).

Answer

1. When planning a project to meet targets for cost, schedule, or quality, it is useful to identify likely risks to the success of the project. A risk is any possible situation that is not planned for, but that, if it occurs, is likely to divert the project from its planned result.
2. For example, an established project team plans for the work to be done by its staff, but there is the risk that an employee may unexpectedly leave the team.
3. In project management, the risk management process has the objectives of identifying, assessing, and managing risks, both positive and negative.

4. All too often, project manager's focus only on negative risk, however, good things can happen in a project, "things" that were foreseen, but not expressly planned.
5. The objective of risk management is to predict risks, assess their likelihood and impact, and to actively plan what should be done ahead of time to best deal with situations when they occur.
6. The risk management process usually occurs in five distinct steps : plan risk management, risk identification, qualitative and quantitative risk analysis, risk response planning, and risk monitoring and control.
7. The central point of risk identification and assessment in risk management is understanding the risk. However, this is also where project managers and risk subject matter experts (SMEs) get the least help from recognized references, best practices, or work standards.
8. Currently, the Project Management Institute (PMIr) has a team of SMEs working on a Practice Standard for Risk Management. This team has identified one very good tool: theRisk Breakdown Structure (RBS).
9. The RBS helps the project manager, the risk manager, and almost any stakeholder to understand, and therefore be able to identify and assess risk.
10. The RBS will prove extremely valuable to better grasp when a project needs to receive special scrutiny, in other words, when risk might happen.
11. The RBS can also help the project manager and the risk manager to better understand recurring risks and concentrations of risk that could lead to issues that affect the status of the project.
12. Following the concept of the Work Breakdown Structure (WBS), the Risk Breakdown Structure provides a means for the project manager and risk manager to structure the risks being addressed or tracked.
13. Just as PMI defines the Work Breakdown Structure as a "deliverable-oriented grouping of project elements that organizes and defines the total work scope of the project", the RBS could be considered as a "hierarchically organized depiction of the identified project risks arranged by risk category."
14. In project management language, risks include anything unplanned and unforeseen that can have a negative impact on the project's costs, timing or quality. A good project manager should be able to manage the risks effectively and get the project on track.
15. Many project managers and risk managers currently use "home-grown" methods for listing, identifying, assessing, and tracking risks in their projects.
16. These methods include:spreadsheets, listing, generic risk taxonomy, based somewhat loosely on various standards and guidelines.

17. An approach that simply places the risks in a list, a simple table, or even in a database does not provide the strength of using a structured, organized method similar to a Work Breakdown Structure.
18. To fully understand the risks and better identify and assess the risk, a "deep-dive" into each risk, recording as many levels of identification as necessary, may be required.

Que 5.14. Write a short note on risk management process.

UPTU 2013-14, Marks 10

OR

What do you mean by risk ? Explain. How risk planning and monitoring is carried out ? Discuss.

UPTU 2012-13, Marks 10

OR

Discuss risk monitoring.

UPTU 2014-15, Marks 05

OR

Write a short note on risk analysis.

Answer

Risk management is a very tedious task. It involves basically two steps :

1. Risk assessment
2. Risk control

1. **Risk assessment :** It is the process of examining a project and identifying areas of potential risk. The risk assessment consists of three activities :
   a. Risks Identifying.
   b. Risk Analyzing.
   c. Risk Prioritization.

   a. **Risk identification :**
      i. Risk identification is a systematic attempt to specify threats to the project plan. The purpose of risk identification is to develop a list of risk items called risk statement.
      ii. Risk identification can be facilitated with the help of a checklist of common risk areas for software projects or by examining the contents of an organizational database of previously identified risks and mitigation strategies (both successful and unsuccessful).
      iii. Risk identification is carried out as a team process using brainstorming. To assist the process a list of risk types can be used.
      iv. The end product of this step of the process is a list of risks that could occur and affect the product, the process or the business.

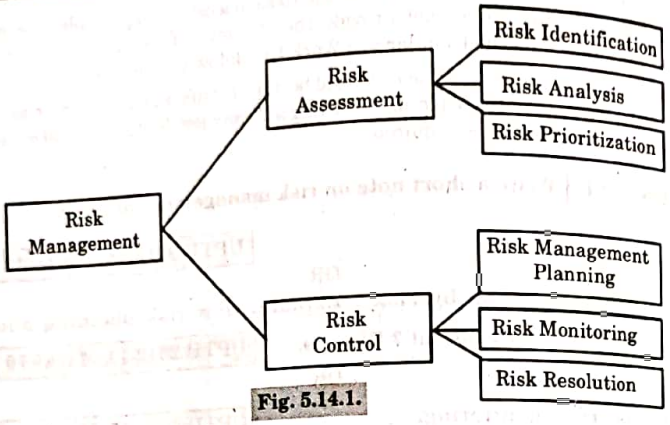


Fig. 5.14.1.

v. Within the identification phase, several activities occur. The main activities are :

1. **Identify risks :** There are many techniques to be used to identify risk. Some of these are checklists, interviews, brainstorm meetings, reviews and surveys. A checklist to be used as a tool for identification of risks is provided.
2. **Define risk attributes :** After the risks are identified, they are evaluated with the criteria : likelihood of occurrence (probability), consequence and time frame for action. These values are initial estimations which are analyzed more in the next phase.
3. **Document :** The risks are then documented. Together with the name of the risks, a risk statement and context are to be specified. In this initial phase, the description of the risk issue, the probability and the consequence are specified in subjective terms.
4. **Communicate :**
   a. Spreading the knowledge to the project members. Inputs to the identification phase are uncertainties, knowledge, concerns and issues.
   b. If projects have been conducted before, the resolution of these inputs may be stored in a database that helps the project managers to detect and find appropriate risk items.
   c. The output of the identification phase is the risk statement that contains identified risks that may affect the project.

   d. Furthermore, together with the statements, risk context is produced.
   e. The purpose of the context is to describe the risk items events, conditions, constraints, assumptions, circumstances, contributing factors and related issues by answering the question what, when, where, how and why of each risk identified.
   f. One risk may have several answers to each of these questions.
   g. The risk identification should include the project team and the risk management team. In the second iteration, the entire project team and primary stakeholders should participate.
   h. If an unbiased analysis is wanted, a person who is not involved in the project may perform a third iteration of the risk identification.

b. **Risk analysis :**

i. When the risks have been identified, all items are analyzed using different criteria. The purpose of the risk analysis is to assess the loss probability and magnitude of each risk item.

ii. The input is the risk statement and context developed in the identification phase.

iii. The output of this phase is a risk list containing relative ranking of the risks and a further analysis of the description, probability, consequence and context.

iv. The main activities in this phase are :

1. **Group similar risks :** Detect duplicates and find new risk items by grouping the identified risks into categories.
2. **Determine risk drivers :** The risk drivers are parameters that affect the identified risk. For example, schedule drivers are included in the critical path model. Determining these properties help to assess and prioritize the risks.
3. **Determine source of risks :** The sources of risks are the root causes of the risks. These are determined by asking the question why ? and trying to figure out what may have caused the risk. Several root causes may lead to the same risk.
4. **Estimate risk exposure :** The risk exposure is measure of the probability and the consequence of a risk item. The consequence can also be stated in terms of loss (for example, life, money, property, reputation).

5. **Evaluate against criteria :**
   a. Each risk item is evaluated using the predefined criteria, which are important for the specific project.
   b. Criteria may be stated in terms of the probability of occurrence, the consequence and the time frame.
   c. This information is used to prioritize the risks. Once this is done, risks can be prioritized and the most serious risks can be identified for monitoring.

c. **Risk prioritization :**
   i. Risk prioritization helps the project focus on its most severe risks by assessing the risk exposure.
   ii. Exposure is the product of the probability of incurring a loss due to the risk and the potential magnitude of that loss.
   iii. This prioritization can be done in a quantitative way, by estimating the probability (0.1–1.0) and relative loss, on a scale of 1 to 10.
   iv. Multiplying these factors together provide an estimation of risk exposure due to each risk item, which can run from 0.1 to 10.
   v. The higher the exposure, the more aggressively the risk should be tackled. It may be easier to simply estimate both probability and impact as High, Medium or Low.
   vi. Those items having at least one dimension rated as high are the ones to worry about first.

2. **Risk control :** Risk control is the process of managing risks to achieve the desired outcomes. Risk control process involves the following activities :
   a. Risk planning
   b. Risk mitigation
   c. Risk resolution
   d. Risk monitoring

a. **Risk planning :**
   i. Risk management planning produces a plan for dealing with each significant risk, including mitigation approaches, owners, and time lines.
   ii. Risk resolution is execution of the plans for dealing with each risk. Finally, risk monitoring involves tracking your progress toward resolving each risk item.
   iii. Risk planning is to identify strategies to deal with risk. These strategies fall into three categories :
      1. Risk avoidance
      2. Risk minimization
      3. Risk contingency plans.

b. **Risk mitigation :**
   i. The risk mitigation is plan that would reduce or eliminate the highest risks. The key question is : what should be done and who is responsible to eliminate or minimize the risk ?
   ii. The mitigation plan includes a description of the actions that can be taken to mitigate the red rated risk and assigns a primary handler for the action.

c. **Risk resolution :**
   i. When a risk has occurred, it has to be solved. Risk resolution is the execution of the plans for dealing with each risk.
   ii. If the risk is at the watch list, a plan of how to resolve the risk already had taken place. The project manager has to respond to the trigger and execute the action plan.
   iii. The project manager also needs to report progress against the plan and correct for deviation.
   iv. The input to this phase is the lists of risk and its implementation. The output is risk action plan.

d. **Risk monitoring :**
   i. Risk monitoring is the continually reassessing of risks as the project proceeds and conditions change.
   ii. For example, successful completion of beta testing means that the risk of the client organization rejecting the system is minimal, while large turnover in development staff usually increases project and product risks.

Que 5.15. What are the factors which affects the risk identification procedure of any software project ?

Answer

The categories of factors that will need to be considered include the following :

1. **Application factors :**
   a. The nature of the application, whether it is a simple data processing application, a safety-critical system or a large distributed system with real-time elements is likely to be a critical factor.
   b. The expected size of the application is also important because the larger the system, the greater is the likelihood of errors and communication and management problems.

2. **Staff factors :**
   a. The experience and skills of the staff involved are clearly major factors an experienced programmer is, one would hope, less likely to make errors than one with little experience.

b. We must, however, also consider the appropriateness of the experience - experience in coding small data processing modules in COBOL may be little value if we are developing a complex real-time control system using C++.

c. Such factors as the level of staff satisfaction and the staff turn-over rates are also important to the success of any project - demotivated staff or key personnel leaving unexpectedly have caused many a project to fail.

3. **Project factors :**

a. It is important that the project and its objectives are well defined and that they are absolutely clear to all members of the project team and all key stakeholders.

b. Any possibility that this is not the case will pose a risk to the success of the project.

c. Similarly, an agreed and formal quality plan must be in place and adhered to by all participants and possibility that quality plan is inadequate or not adhered to will jeopardize the project.

4. **Project methods :**

a. Using well-specified and structured methods (such a PRINCE 2 and SSADM) for project management and system development will decrease the risk of delivering a system that is unsatisfactory or late.

b. Using such methods for the first time, though, may cause problems and delays it is only with experience that the benefits accrue.

5. **Hardware/Software factors :**

a. A project that requires new hardware for development is likely to pose a higher risk than one where the software can be developed on existing (and familiar) hardware.

b. Where a system is developed on type of hardware or software platform to be used on another there might be additional (and high) risks at installation.

6. **Changeover factors :** The need for an 'all-in-one' changeover to the new system poses particular risks. Incremental or gradual changeover minimizes the risks involved but is not always practical. Parallel running can provide a safety net but might be impossible or too costly.

7. **Supplier factors :**

a. The extent to which a project relies on external organizations that cannot be directly controlled often influences the project's success.

b. For example, delays in the installation of telephone lines or delivery of equipment may be difficult to avoid, particularly if the project is of little consequence to the external supplier.

8. **Environment factors :** Changes in the environment can affect a project's success. For example, a significant change in the taxation regulations could have serious consequences for the development of a payroll application.

9. **Health and safety factors :** While not generally a major issue for software projects, the possible effects of project activities on the health and safety of the participants and the environment should be considered.

Que 5.16. How to prioritize the monitoring ? Explain.

Answer

So far we have assumed that all aspects of a project will receive equal treatment in terms of the degree of monitoring applied. We must not forget, however, that monitoring takes time and uses resources that might sometimes be put to better use.

Following are the priorities applied in deciding levels of monitoring :

1. **Critical path activities :**

a. Any delay in an activity in the critical path will cause a delay in the completion date for the project.

b. Critical path activities are therefore likely to have-a very high priority for close monitoring.

2. **Activities with no free float :**

a. A delay in any activity with no free float will delay at least some subsequent activities. It might not delay the project completion date.

b. These subsequent delays can have serious effects on our resource schedule as a delay in a subsequent activity could mean that the resources for that activity will become unavailable before that activity is completed because they are committed elsewhere.

3. **Activities with less than a specified float :**

a. If any activity has very little float it might use up this float before the regular activity monitoring brings the problem to the project manager's attention.

b. It is common practice to monitor closely those activities with less than, say one week free float.

4. **High risk activities :**

a. A set of high risk activities should have been identified as part of the initial risk profiling exercise.

b. If we are using the PERT three-estimate approach we will designate as high risk those activities that have a high estimated duration variance.

c. These activities will be given close attention because they are most likely to overrun or overspend.

5. **Activities using critical resources :**
   a. Activities can be critical because they are very expensive (as in the case of specialized contract programmers).
   b. Staff or other resources might be available only for a limited period, especially if they are controlled outside the project team.
   c. In any event, an activity that demands a critical resource requires a high level of monitoring.

Que 5.17. **Explain cost benefit analysis.**

UPTU 2014-15, Marks 05

Answer

1. The most common way of carrying out an economic assessment of a proposed information system or other development, is by comparing the expected costs of development and operation of the system with the benefits of having it in place.
2. Assessment focuses on whether the estimated costs are exceeded by the estimated income and benefits.
3. Additionally, it is usually necessary to ask whether or not the project under consideration is the best of a number of options.
4. There might be more candidate projects that can be undertaken at any one time and, in any case, projects will need to be prioritized so that resources are allocated effectively.
5. The standard way of evaluating the economic benefits of any project is to carry out a cost benefit analysis, which consists of two steps :
   a. **Identifying and estimating all of the costs and benefits of carrying out the project and operating the delivered application :**
      i. These include the development costs of the system, the operating costs and the benefits that are expected to accrue from the operation of the new system.
      ii. Where the proposed system is replacing an existing one, these estimates should reflect the change in costs and benefits due to the new system.
      iii. For example, a new sales order processing system could not claim to benefit an organization by the total value of sales, only by the increase due to the use of the new system.
   b. **Expressing these costs and benefits in common units :**
      i. We need to evaluate the net benefit, that is, the difference between the total benefit accruing from the system and the total cost of creating and operating it.

ii. To do this, we must express each cost and each benefit in some common unit. The fundamental common unit of measurement is money.

iii. We therefore need to express each of the expected benefits and costs in monetary terms.

6. Most direct costs are relatively easy to identify and quantify in approximate monetary terms. It is helpful to categorize costs according to where they originate in the life of the project.
   a. **Development costs :** Include the salaries and other employment costs of the staff involved in the development project and all associated costs.
   b. **Setup cost :** Include the costs of putting the system into place. These consist mainly of the costs of any new hardware and ancillary equipment, but will also include costs of file conversion, recruitment and staff training.
   c. **Operational costs :** Consist of the costs of operating the system once it has been installed.
7. Benefits, on the other hand, are often quite difficult to quantify in monetary terms even once they have been identified. Benefits may be categorized as follows :
   a. **Direct benefits :** These accrue directly from the operation of the proposed system. These could, for example, include the reduction in salary bills through the introduction of a new, computerized system.
   b. **Assessable indirect benefits :** These are generally secondary benefits, such as increased accuracy through the introduction of a more user-friendly screen design where we might be able to estimate the reduction in errors, and hence costs, of the proposed system.
   c. **Intangible benefits :** These are generally longer term or benefits that are considered very difficult to quantify. Enhanced job interest can lead to reduced staff turnover and, hence, lower recruitment costs.

Que 5.18. **What are the cost benefit evaluation techniques ? Explain each.**

OR

Write short notes on :

1. **Net profit**
2. **Payback period**
3. **Return on investment**
4. **Net present value**

Answer

We would consider proceeding with a project only when the benefits outweigh the costs. However, in order to choose among projects, we need to take into account the timing of the costs and benefits as well as the benefits relative to the size of the investment.

1. **Net profit :** The net profit of a project is the difference between the total costs and the total income over the life of the project.
2. **Payback period :**
   a. The payback period is the time taken to break even or payback the initial investment.
   b. Normally, the project with the shortest payback period will be chosen on the basis that an organization will wish to minimize the time that a project is 'in debt'.
   c. The advantage of the payback period is that it is simple to calculate and is not particularly sensitive to small forecasting errors.
   d. Its disadvantage as a selection technique is that it ignores the overall profitability of the project - in fact, it totally ignores any income (or expenditure) once the project has broken even.
3. **Return on investment :**
   a. The return on investment (ROI), also known as the accounting rate of return (ARR), provides a way of comparing the net profitability to the investment required.
   b. There are some variations on the formula used to calculate the return on investment, but a straightforward common version is :

$$ROI = \frac{\text{average annual profit}}{\text{total investment}} \times 100$$

4. **Net present value :**
   a. The calculation of net present value (NPV) is a project evaluation technique that takes into account the profitability of a project and the timing of the cash flows that are produced.
   b. It does so by discounting future cash flows by a percentage known as the discount rate.
   c. The present value of any future cash flow may be obtained by applying the following formula :

$$\text{Present value} = \frac{\text{Value in year } t}{(1+r)^t}$$

where r is the discount rate, and t is the number of years into the future that the cash flow occurs.

Que 5.19. What do you mean by cash flow forecasting ? Explain.

Answer

1. As important as estimating the overall costs and benefits of a project is the forecasting of the cash flows that will take place and their timing.
2. A cash flow forecast will indicate when expenditure and income will take place.
3. We need to spend money, such as staff wages, during the development stages of a project.
4. Such expenditure cannot be deferred until income is received (either from using the software if it is being developed for in-house or from selling it).
5. It is important that we know that we can fund the development expenditure either from the company's own resources or by borrowing from the bank.
6. In any event, it is vital to have some forecast of when expenditure such as the payment of salaries and bank interest will take place and when any income is to be expected, such as payment on completion or, possibly, stage payments.

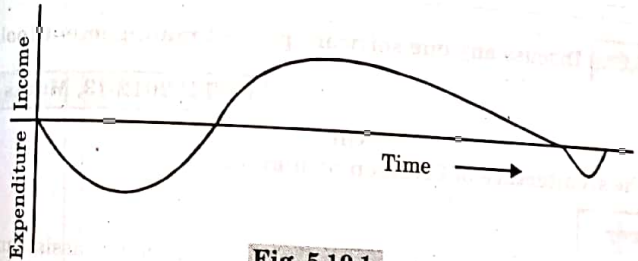


Fig. 5.19.1.

7. Accurate cash flow forecasting is not easy, as it generally needs to be done early in the project's life cycle (at least before any significant expenditure is committed) and many items to be estimated (particularly the benefits of using software or decommissioning costs) might be some years in the future.
8. When estimating future cash flows, it is usual to ignore the effects of inflation.
9. Trying to forecast the effects of inflation increases the uncertainty of the forecasts.
10. Moreover, if expenditure is increased due to inflation it is likely that income will increase proportionately.
11. However, measures to deal with increases in costs where work is being done for an external customer must be in place, for example, index, linked prices where work involves use of raw materials.

PART-3

Software Project Management Tools : CASE Tools, Planning and Scheduling Tools, MS-Project.

CONCEPT OUTLINE : PART-3

- CASE tools are software programs that are designed to assist human programmers with the complexity of the processes and the artifacts of software engineering.
- Microsoft Project helps you achieve your project goal on time and on budget.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 5.20. Discuss any one software project management tool.

UPTU 2012-13, Marks 10

OR

Give the architecture of CASE environment.

Answer

1. CASE tools are software programs that are designed to assist human programmers with the complexity of the processes and the artifacts of software engineering.
2. They constitute the laws of and the automated tools that aid in the synthesis, analysis, modeling, or documentation of software.
3. CASE stands for a large number of applications reaching from simple editing tools to environments supporting the whole life cycle.
4. CASE attacks software productivity problems at both ends of the life cycle by automating many analysis and design tasks, as well as program implementation and maintenance tasks.
5. The major task of a CASE tool is to accept different kinds of specifications, analyze the specifications, transform specifications and maintain a large, ever-growing set of interrelated specifications possibly in several versions.

CASE architecture :

1. Fig. 5.20.1 shows a general architecture of a CASE environment. Central to this architecture is repository, the most critical and important feature of a CASE environment.

2. The complete information about meta-models, methods, project artifacts, baseline, process models etc., is available in the repository.
3. This information can be shared by different team members efficiently thereby improving the project management as a whole.
4. While implementing, the repository can be either centralized or distributed. Normally this is implemented using a database or a file system.

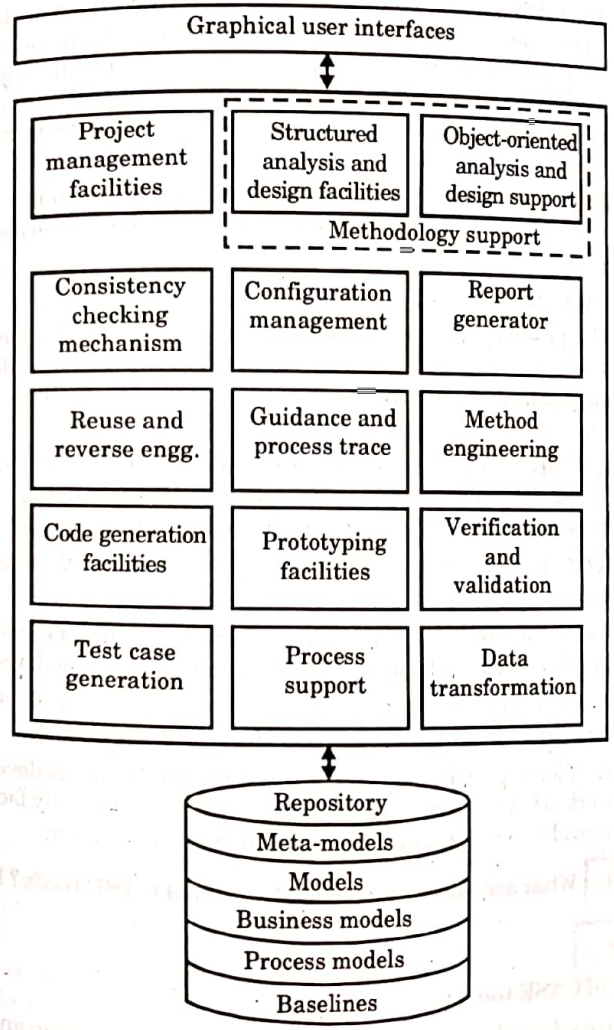


Fig. 5.20.1.

5. As repository holds different types of data from different phases of software life cycle, it must make the data available to all tools that require them

6. Some of the important properties of a CASE environment are discussed below :
   a. A CASE environment must have facility to update and retrieve information at any point of development.
   b. It must support project management and planning facilities in order to ensure timely completion of a project and to track development in the project.
   c. The CASE environment must also have support for methodology (structured, object-oriented etc.) used by the organization. As there is no universal method which is applicable to all projects, the environment should support building of new methods from existing method components (method engineering) using concept of meta-model.
   d. The CASE environment must have proper consistency checking mechanisms in order to maintain integrity of the system description.
   e. The CASE environment must also have necessary support for tool integration.
   f. The prototyping is an important activity during development in order to understand requirements. As far as possible, CASE environment therefore must have prototyping facility including report generation, animation etc.
   g. A CASE environment must also be able to store data in multiple forms *i.e.*, tables, diagrams, matrix etc. All these forms represent same semantic information but from different viewpoints.
   h. A CASE environment must support integration with other tools in the organization also.
   i. A CASE environment must have some kind of guidance mechanism in order to guide the application engineering during development.
   j. A CASE environment should be able to trace product/process components, store them in repository for future reuse.
   k. In order to provide easy access to user to facilitate his development work, a CASE environment must provide user friendly facilities by providing proper graphics and windows environment.

Que 5.21. What are benefits and limitations of CASE tools? Explain.

Answer

Benefits of CASE tools :

1. **Improved productivity :** CASE tools provide automation and reduce the time to complete many tasks, especially those involving diagramming and associated specifications. Estimates of improvements in productivity after application range from 355 to more than 200%.

2. **Better documentation :** By using CASE tools, vast amount of documentation are produce along the way. Most tools have revisions for comments and notes on systems development and maintenance.
3. **Reduced lifetime maintenance :**
   a. As a result of better design, better analysis and automatic code generation, automatic testing and debugging, overall systems quality improves.
   b. CASE provide re-engineering tools that can be very important because they make this process more efficient, less time consuming and less expensive by discovering older parts of the system that can be reused.
4. **Improved accuracy :**
   a. CASE tools can provide ongoing debugging and error checking which is very vital for early defect removal, which actually played a major role in shaping modern software.
   b. The importance of early defect removal is very important. Less effort and time are consumed if corrections are made at an early stage, such as the design stage.
   c. As the system grows larger, it becomes difficult to modify. Error identification becomes harder.
5. **Opportunity to non-programmers :**
   a. With the increased movement toward object-oriented technology and client-server based, programming can also be done by people who do not have a complete programming background.
   b. By using the lower CASE tools, it could be possible to develop software from the initial design and analysis phase.
6. **Intangible benefits :** CASE tools can be used to allow for greater user participation, which can lead to better acceptance of the new system. This can reduce the initial learning curve.

Limitations of CASE tools :

1. **Cost :**
   a. Using CASE tools is a very costly affair. In fact, most firms engaged in software development on a small scale do not invest in CASE tools because they think that the benefits of CASE are justifiable only in the development of large systems.
   b. The cost of outfitting every systems developer with a preferred CASE tool kit can be quite high.
   c. Hardware and systems software, training and consulting are all factors in the total cost equation of using CASE tools.
2. **Learning curve :**
   a. In most cases, programmer productivity may fall in the initial phase of implementation, because users need time to learn the technology.

b. In fact, a CASE consulting industry has evolved to support users of CASE tools.

c. The consultants offer training and on-site services that can be crucial to accelerate the learning curve and to the development and use of the tools.

3. **Tool mix :**
   a. It is important to make an appropriate selection of tool mix to get cost advantage. CASE integration and data integration across all platforms is also very important.
   b. The ability to share the results of work done on one CASE tool with another CASE tool is perhaps the most important type of CASE integration.

Que 5.22. **Give some examples of CASE tools which support the different stage of software life cycle.**

OR

List out three most used CASE tools.

UPTU 2012-13, Marks 05

Answer

Some popular CASE tools supporting different stages of software life cycle are :

1. **Software requirements tools :** A number of tools are proposed for modeling, tracing, and analyzing requirements. CASE diagramming or upper CASE tools represent the system requirements visually using structured or object-oriented methodologies. Examples are :
   a. Turbo-analyst
   b. Oracle's Designer/2000, Argo/UML Rational ROSE
   c. MS Access
   d. Dia
   e. Excelerator, Stalemate
2. **Software design tools :** These tools are used to support the system design stage of SDLC and in most of the cases are extensions of CASE tools used to requirements analysis stage. They can be used to support design, verification and optimization.
3. **Software construction tools :** Software construction tools are the tools which are used to code and implement the software and hence transform the software requirements into working product. These tools are required even after the product is installed so as to remove defects and maintain it. Broadly they can be classified as :
   a. Program editors
   b. Compilers

   c. Interpreters
   d. Debuggers
4. **Software testing tools :** These are the automated tools that support various activities of a software testing stage of SDLC. They can be classified as :
   a. **Test generators :** These are tools which assist the developers in test case design and their documentation. Example, MS Access, Quick List.
   b. **Test execution and evaluation tools :** These tools support the process of test case execution and also evaluate the results of execution. The various tools under this category are :
      i. Capture/Playback tools to automate execution of test cases. Example SQA Robot, QA Playback, Java-Star, Ferret, Auto-Tester, Web etc.
      ii. Coverage analysis tools to ensure all parts of code i.e. statement, decision, conditions, paths, loops etc. are executed. Examples are Visual Testing, Java-Scope, Aqua-Prova etc.
      iii. Memory testing tools to test memory related problems example, Bounds-Checker, Heap-Agent etc.
      iv. Simulators replace the actual hardware/software which interacts with the software to be tested and also evaluate system. Examples are Performance Studio, Load-Runner etc.
   c. **Test management tools :** These tools support management of different test artifacts. Examples are Visual Source Safe, CVS etc. There are automated tools to support reviews and inspections also.
5. **Software maintenance tools :** Tools under this category are :
   a. **Comprehensive tools :** These tools are used to assist in human comprehension and visualization of the programs. Examples are Visual Studio, exref etc.
   b. **Re-engineering tools :** These are tools which allow the change of existing format of a program to a new format *i.e.*, a new language or new database or new technology in general.
6. **Software quality tools :** These tools are used to automate techniques like static analysis, reviews, inspection etc., to ensure software quality.
7. **Configuration management tools :** These tools support version control and other activities related to configuration management *i.e.*, management and control of changes made to the documents. Examples are Clear-Case, Change-man etc.
8. **Project management tools :** Tools under this category automate size estimations, cost estimation, schedule estimation, risk management activities etc. Some of the tools are MS Project, Excel, COCOMO, FPA etc.

Que 5.23. **Write a short note on planning and scheduling tools.**

Answer

Planning and scheduling tools are :

1. **WBS :** Refer Q. 2.2, Page 45C, Unit-2.
2. **PERT chart :** Refer Q. 2.29, Page 81C, Unit-2.
3. **Gantt chart :** Refer Q. 2.32, Page 86C, Unit-2.

Que 5.24. **Explain MS-Project.** **UPTU 2013-14, Marks 10**

Answer

1. Project management software such as Microsoft Project helps us to achieve our project goal on time and on budget.
2. Computer software can significantly aid in project management as a tool for recording, calculating, analysing, consolidating and presenting project details.
3. However, it is important to note that the software cannot produce or even guarantee a successful project plan.
4. Despite this, Microsoft Project assists you to develop a better plan. It does so in the following ways :
   a. MS Project requires you to specifically define the tasks in the project, making you think more carefully about project details.
   b. MS Project makes projections easier to calculate and more reliable. Based on the data you enter, the software will calculate a schedule that will show the various dates and the resources required to perform specific tasks.
   c. MS Project helps you detect inconsistencies and problems in the plan. It will detect when resources are scheduled for more hours than are available or when deadlines cannot be met.
   d. MS Project helps you communicate the plan to others as you can generate printed reports that make the "selling" of the plan to upper-level management, who must approve the plan, an easier task.
   e. Likewise, it is easier to communicate the plan to supervisors and workers, which simplifies securing their approval and co-operation.
   f. MS Project helps you track progress and detect potential difficulties once the project is underway. We can replace projected dates for the scheduled tasks with actual dates, as tasks are being performed.
   g. The software revises the schedule and the new projection will provide you with advance warning of potential delays (if any) so that you can take any required corrective measures.

5. Project management involves more than just opening a blank document and typing a list of tasks. There are "housekeeping" chores to be done and choices to be made about how to calculate the project schedule.
6. However, Microsoft Project does not have rigid requirements about the order in which you deal with these preliminaries.
7. We can begin by jotting down some ideas about tasks that you think might be required, and you can later adjust the scheduling calendar, enter the basic project information, revise the calculation and display options, and define the resources. In fact, you can execute all of the previous steps in any order.

VERY IMPORTANT QUESTIONS

Following questions are very important. These questions may be asked in your SESSIONALS as well as UNIVERSITY EXAMINATION.

Q. 1. What do you mean by software configuration management ?
Ans. Refer Q. 5.1.

Q. 2. Explain change request management.
Ans. Refer Q. 5.7.

Q. 3. Write a short note on risk management.
Ans. Refer Q. 5.14.

Q. 4. Discuss cost benefit analysis.
Ans. Refer Q. 5.18.

Q. 5. Discuss any software project management tool in detail.
Ans. Refer Q. 5.22.

☺☺☺