TCS-042

B.TECH.

EIGHTH SEMESTER EXAMINATION 2009-10

SOFTWARE PROJECT MANAGEMENT

Time : 3 Hours *Total Marks : 100*

Note: Attempt all questions:

Q.1. Attempt any TWO: [2X10=20]

(a) Explain software project planning, giving its various objective. Also discuss the structure of a Software Project management Plan(SPMP) in detail.

Ans. Software Project Planning: The purpose of Software Project Planning is to establish reasonable plans for performing the software engineering and for managing the software project.

Software Project Planning involves developing estimates for the work to be performed, establishing the necessary commitments, and defining the plan to perform the work.

The software planning process includes steps to estimate the size of the software work products and the resources needed, produce a schedule, identify and assess software risks, and negotiate commitments. Iterating through these steps may be necessary to establish the plan for the software project (*i.e., the software development plan*).

This plan provides the basis for performing and managing the software project's activities and addresses the commitments to the software project's customer according to the resources, constraints, and capabilities of the software project.

Goals:

1. Software estimates are documented for use in planning and tracking the software project.
2. Software project activities and commitments are planned and documented.
3. Affected groups and individuals agree to their commitments related to the software project

Objective of software Project management:

Project management is the discipline of organizing and managing resources (e.g. people) in a way that the project is completed within defined scope, quality, time and cost constraints. A project is a temporary and one-time endeavor undertaken to create a unique product or service, which brings about beneficial change or added value. This property of being a temporary and one-time undertaking contrasts with processes, or operations, which are permanent or semi-permanent ongoing functional work to create the same product or service over and over again. The management of these two systems is often very different and requires varying technical skills and philosophy, hence requiring the development of project managements.

The first challenge of project management is to make sure that a project is delivered within defined constraints. The second, more ambitious challenge is the optimized allocation and integration of inputs needed to meet pre-defined objectives. A project is a carefully defined set of activities that use resources (money, people, materials, energy, space, provisions, communication, etc.) to meet the pre-defined objectives.

SPMP - Software Project Management Plan:

A software process that includes all aspects of software including planning, leading, organizing, estimating, directing, monitoring and controlling software projects and their teams. SPMP also includes quantitative progress measures,

software lifecycles, estimating and risk management. SPMP may be used to manage outsouring software projects or insource development projects. See also Software Development Life Cycle (SDLC).

1. Overview

1.1 Project Summary

1.1.1 **Purpose, scope, and objectives.**

1.1.2 **Assumptions and constraints.** Constraints include the following:

- The deadline must be met.
- The budget constraint must be met.
- The product must be reliable.

1.1.3 **Project deliverables.** The complete product, including user manual, will be delivered 10 weeks after the project commences.

1.1.4 **Schedule and budget summary.**

1.2 Evolution of the project management plan. All changes to the project management plan must be agreed .

2 **Reference materials.** All artifacts will conform to the company's programming, documentation, and testing standards.

3 Definitions and acronyms.

4 Project organization

4.1 External interfaces.

4.2 Internal structure.

4.3 Roles and responsibilities.

5 Managerial process plans:

5.1 Start up plan

5.1.1 Estimation plan.

5.1.2 Staffing plan.

5.1.3 Resource acquisition plan.

5.1.4 Project staff training plan.

5.2 Work plan:

5.2.1 Work activities and Schedule allocation

5.2.2 Resource allocation.

5.2.3 Budget allocation.

5.3 Control plan

5.4 Risk management plan

6. Supporting process plans:

6.1 Configuration management plan.

6.2 Testing plan –

6.3 Documentation plan –

6.4 Quality assurance plan and

6.5 Reviews and audits plan.

6.6 **Problem resolution plan –**

6.7 **Subcontractor management plan.**

6.8 **Process improvement plan.**

(b) (i) Explain why the intangibility of software system possess special problems for software project management?

(ii) Discuss the various responsibilities of a software project manager.

Ans. (i) Software under development is considered and Intangible Asset and requires special accounting procedures. Since you can not see it, touch it, stand on it or eat it, you have to rely on your team to produce timely and accurate documentation needed to track progress and productivity. The intangible nature of software causes problems for management in planning, estimating, scheduling and budgeting for accounting purposes. Ensuring that the software is delivered on schedule and in accordance with the the projects requirements. Project management is needed because software development is subject to problems in development, flexible budgets and schedule constraints. The software project manager's job is to ensure that the software project meets these constraints on time and within projected investment calculations. From an accounting view which I see as the biggest obstacle, intangible assets have special guidelines for capitalization. Computer software is the most common type of an internally generated intangible asset. It can be purchased or licensed from a 3rd party and then modified it is still considered internally generated and re-classified as an intangible asset. The cost of internally generated intangibles, such as computer software, must be capitalized or expensed depending on three project stages:

1. Preliminary project stage -formulation and evaluation of alternatives, determining the existence of the needed technology.

2. Application development design includes coding, installation to hardware, testing, and parallel processing.

3. Post-implementation stage includes application training and software maintenance.

Stages 1 and 3 are expensed, while stage 2 is capitalized. Generally, you have not reached stage 2 until you have executive management support for the project along with a commitment of funding and personnel. Determining what stage you are in can be difficult for large projects. Large projects might need to be broken down into multiple modules which are capitalized separately. The people involved in designing a software project are its greatest assets. They represent intellectual capital and its up to the software project manager to get best out of their investment in people. This is achieved when people are respected by their company and given a level of responsibility and reward that matches their skills. Software Project management is about regulating and managing phases of development done by others on a product that you can't see, smell, touch or eat and you can't even demonstrate it until it's almost finished. A close working relationship and constant communication with the people you are relying on to get the job completed are your greatest tools.

(ii) Responsibility of a project Manager:

Those are the following responsibilities which project manager should take part:

1. Project /Practice Management

- Creating and executing project work plans and revise as appropriate to meet changing needs and requirements.
- Identifing resources needed and assigns individual responsibilities.
- Managing day-to-day operational aspects of a project and scope.
- Reviewing deliverables prepared by team before passing to client.
- Effectively appling methodology and enforcing project standards.

2. Project Accounting

- Tracking and reporting team hours and expenses on a weekly basis.
- Managing project budget.
- Determining appropriate revenue recognition, ensuring timely and accurate invoicing, and monitoring receivables for project.
- Analyzing project profitability, revenue, margins, bill rates and utilization.

3. Financial Management

- Understanding basic revenue models, Profit/Loss, and cost-to-completion projections and making decisions accordingly.
- Understanding pricing model and billing procedures.
- Accurately forecasting revenue, profitability, margins, bill rates and utilization.
- Assuring project and legal documents are completed and signed.

4. Business Development

- Identifing business development and "add-on" sales opportunities as they relate to a specific project.
- Effectively conveying company's message in both written and verbal in business development discussions.

5. Communication

- Manager has to facilitate team and client meetings effectively.
- Effectively communicating relevant project information to superiors.
- Delivering engaging, informative, well-organized presentations.
- Resolving and/or escalates issues in a timely fashion.

6. Technical Understanding

- Understanding Internet, Intranet, Extranet and client/server architectures.

- Possessing a thorough understanding of company's capabilities.
- Maintains awareness of new and emerging technologies and the potential application on client engagements.

(c) What do you mean by the software project estimation? Give various estimation models . Describe any one of the estimation model using suitable examples.

Ans. Software Project Estimation: The ability to accurately estimate the time and/or cost taken for a project to come in to its successful conclusion is a serious problem for software engineers. The use of a repeatable, clearly defined and well understood software development processes has, in recent years, shown itself to be the most effective method of gaining useful historical data that can be used for statistical estimation. In particular, the act of sampling more frequently, coupled with the loosening of constraints between parts of a project, has allowed more accurate estimation and more rapid development times.

The following are the cost models:

1. Algorithmic cost model
2. Expert judgment
3. Delphi cost estimation
4. COCOMO

The majority of software cost estimates are based on one of the following method:

ALGORITHMIC COST MODELS: In this process costs are analysed using mathematical formulae linking costs or inputs with metrics to produce an estimated output. The formulae used in a formal model arise from the analysis of historical data.

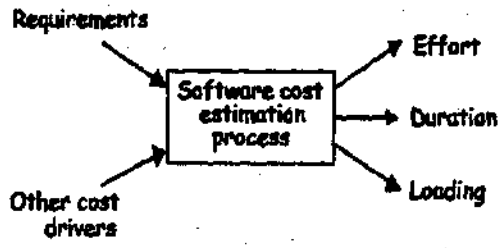


Fig. 1. *Classical view of the algorithmic cost estimation process*

An input requirement of an algorithmic model is to provide a metric to measure the size of the finished system. Typically lines of source code are used, this is obviously not known at the start of the project. SLOC is also very dependant on the programming language and programming environment, this is difficult to determine at an early stage in the problem especially as requirements are likely to be sketchy. Despite this SLOC has been the most widely used size metric in the past, but current trends indicate that it is fast becoming less stable. This is probably due to the changes in software development process in recent years highlighted with a tendency to use prototyping, case tools and so forth. An alternative is to use function points proposed by which are related to the functionality of the software rather than its size. A more recent approach is to use object points. This is in comparison a new methodology and has not been publicised in the same depth as function points and SLOC.

Algorithmic models generally provide direct estimates of effort or duration. As shown in figure the main input is usually a prediction of software size. Effort prediction models take the general form :

effort = p*S (1/productivity rate)

where **p** is a productivity constant and **S** is the size of the system.

Once the value for p is known. E.g. productivity = **450** source lines of code per month, making **p = 0.0022** and the size of the system has been estimated at **8500** KLOC.

effort = 0.0022 * 8500

effort = 18.7 person months

The example above assumes that the relationship between effort and size is a linear one. Most models allow for non-linear relationships by introducing economies or dis-economies of scale. The general formula being:

effort = p * Se

effort = 5.2 KLOC 0.91

Q.2. Attempt any TWO **[2×10=20]**

Q.2.(a) What do you understand by the Activity network and Gantt chart? Draw the activity network and gantt chart representations for the

following table that indicates the various task involved in completing a software project, the corresponding activities, and the estimated effort for each task in person-month:

Tasks	Activity	Efforts in person-months
T1	Requirements specification	1
T2	Design	2
T3	Code actuator interface module	2
T4	Code sensor interface module	5
T5	Code user interface part	3
T6	Code control processing part	1
T7	Integrate and test	6
T8	Write user manual	1

Ans. Gantt chart: A Gantt chart is a graphical representation of the duration of tasks against the progression of time. A Gantt chart is a useful tool for planning and scheduling projects. A Gantt chart is helpful when monitoring a project's progress

Gantt Chart Format

Task	Duration	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
1	2 mo.												
2	2 mo.												
3	2 mo.												
4	2 mo.												
5	2 mo.												
6	2 mo.												

The horizontal axis of the Gantt chart is a time scale, expressed either in absolute time or in relative time referenced to the beginning of the project. The time resolution depends on the project - the time unit typically is in weeks or months. Rows of bars in the chart show the beginning and ending dates of the individual tasks in the project.

In the above example, each task is shown to begin when the task above it completes. However, the bars may overlap in cases where a task can begin before the completion of another, and there may be several tasks performed in parallel. For such cases, the Gantt chart is quite useful for communicating the timing of the various tasks.

For larger projects, the tasks can be broken into subtasks having their own Gantt charts to maintain readability.

Gantt Chart Role *in* Project Planning: For larger projects, a **work breakdown structure** would be developed to identify the tasks before constructing a Gantt chart. For smaller projects, the Gantt chart itself may used to identify the tasks.

The strength of the Gantt chart is its ability to display the status of each activity at a glance. While often generated using project management software, it is easy to construct using a spreadsheet, and often appears in simple ascii formatting in e-mails among managers.

Activity Network Diagram (chart): An Activity Network Diagram (AND) is also called an Arrow Diagram (because the pictorial display has arrows in it) or a PERT (Program Evaluatioin Review Technique) Diagram, and it is used for identifying time sequences of events which are pivotal to

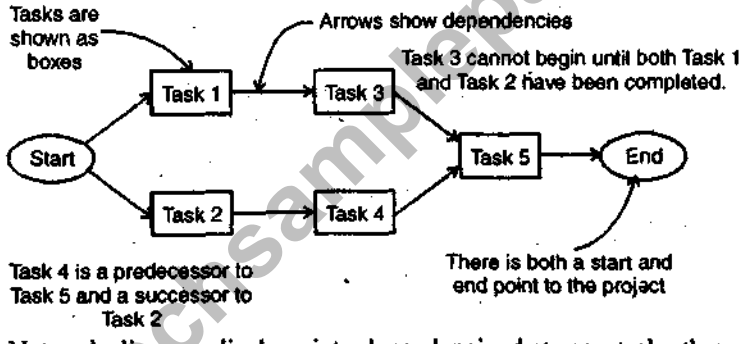
objectives. In critical Path Analysis this helps the teams to comprehend specific event sequences driving time requirements for objective achievement. Activity Network Diagrams are also very useful when a project has multiple activities which need simultaneous management.

Activity Network Diagrams started out as an engineering and construction project management tool. Critical Path Analysis draws on this methodology to identify and standardize medical management activities.

An Activity Network Diagram helps to find out the most efficient sequence of events needed to complete any project. It enables you to create a realistic project schedule by graphically showing:

- the total amount of time needed to complete the poject.
- the sequence in which tasks must be carried out
- which tasks can be carried out at the same time

Example of Activity Network Diagram: A project is composed of a set of actions or tasks which usually have some kind of interdependency. For example, before an axle can be turned, it must first be designed, the metal must be purchased, etc. This type of complex system is much easier to understand through the use of diagrams than through textual description, as actual interconnections between tasks can be shown. You can draw the activity network diagram easily with Edraw software.

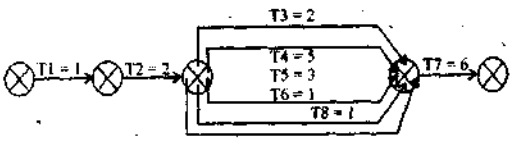


The Activity Network diagram displays interdependencies between tasks through the use of boxes and arrows. Arrows pointing into a task box come from its predecessor tasks, which must be completed before the task can start. Arrows pointing out of a task box go to its successor tasks, which cannot start until at least this task is complete.

Gantt chart:

MonthsTask	1	2	3	4	5	6	7	8	9	10	11	12	13	14
T1														
T2														
T3														
T4														
T5														
T6														
T7														
T8														

Activity Network chart:



Q.2. (b) What is the difference between a macroscopic schedule and a detailed schedule ? Is it possible to manage a project if only a macroscopic scheduled is developed? Discuss with suitable example.

Ans. Macroscopic Schedule: In order to get a general overview of the entire project the first thing that we did was the macroscopic schedule:

Detailed Schedule: Scheduling of software project does not differ greatly from scheduling of any multitask engineering effort. Therefore, generalized project scheduling tools and techniques can be applied with little modification to software projects.

Difference between a macroscopic schedule and a detailed schedule: Software project scheduling is an activity that distributes estimated efforts across the planned project duration by allocating the effort to specific software engineering tasks. It is important to note, however, that the schedule evolves over time. During early stages of project planning, a macroscopic schedule is developed. This type of schedule identifies all major software engineering activities and the project functions to which they are applied. As the project get under way, each entry on the macroscopic schedule is refined into detailed schedule. Here, specific software tasks are identified and scheduled.

Scheduling for software engineering projects can be viewed from rather different perspectives. In the first, an end-date for release of a computer-based system has already been established. The software organization is constrained to distribute effort within the prescribed time frame. The second view of software scheduling assumes that rough chronological organization. Effort is distributed to make best use of resources and an end-date is defined after careful analysis of the software. Unfortunately, the first situation is encountered far more frequently than the second.

Thus it is not possible to manage a project if only a macroscopic schedule is developed

Example : Scheduling of a software project does not differ greatly from scheduling of any multitask engineering effort. Therefore , generalized project scheduling tools and techniques can be applied with little modification to software projects.

Program evalutation and review technique (PERT) and critical path scheduling method (CPM) are two project scheduling methods that can be applied to software development. Both techniques are driven by information already developed in earlier project planning activities.

Q.2.(c) Write short notes on :

(i) Work Breakdown structure (WBS)

(ii) CPM

Ans. (i) Work Breakdown structure (WBS): A Work Breakdown Structure (WBS) describes the project activities in a top down hierarchical manner. It is a structured format where groups of related tasks are broken down into levels of increasing detail, each with unique identifiers. Each task identified in the WBS should have a defined output or "Milestone" which is related to a project deliverable. These "Milestones" allow progress on the project to be monitored against clearly defined goals.

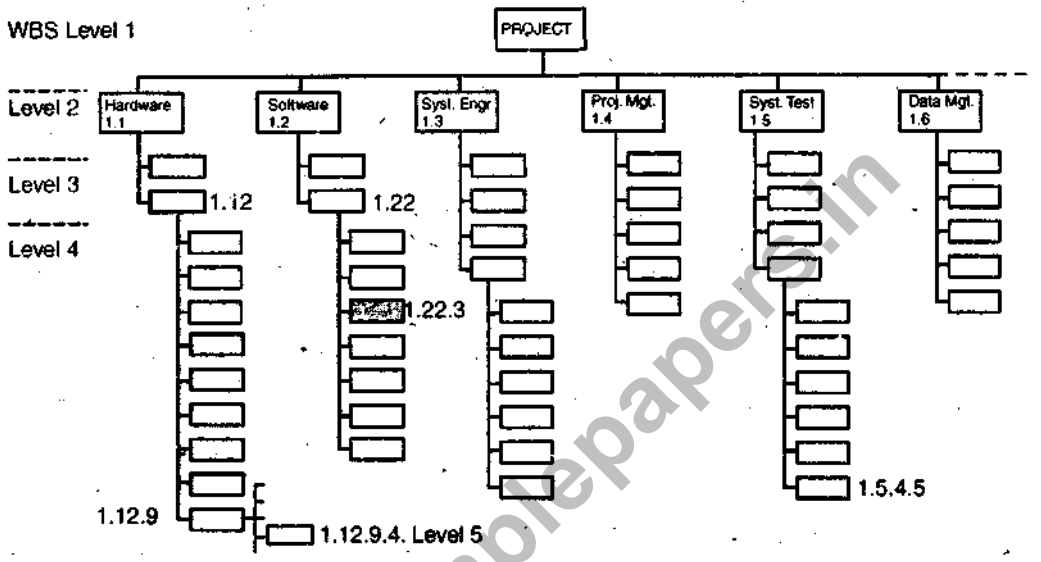
Major tasks, leading to a Milestone, must be fully defined so that all the activities can be identified and the resources required planned for in the project schedule. Sufficient information should be contained in each description so that the Project Manager can write an appropriate instruction for completion of that task. Each task should be capable of being subjected to the following analysis:

1. The entry conditions are fully definable
2. The activities required are fully definable / defined
3. The validation necessary to confirm that the task has been completed satisfactorily
4. The deliverable from the task is clearly identifiable

A well-designed WBS makes it easy to assign any project activity to one and only one terminal element of the WBS.

A sample WBS is shown in the figure below:

WBS Format for System Development Projects



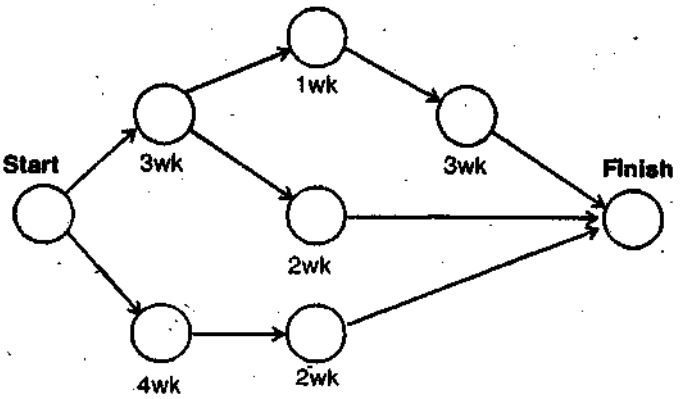
(ii) CPM - Critical Path Method: In 1957, DuPont developed a project management method designed to address the challenge of shutting down chemical plants for maintenance and then restarting the plants once the maintenance had been completed. Given the complexity of the process, they developed the **Critical Path Method** (CPM) for managing such projects.

CPM provides the following benefits:

- Provides a graphical view of the project.
- Predicts the time required to complete the project.
- Shows which activities are critical to maintaining the schedule and which are not.

CPM models the activities and events of a project as a network. Activities are depicted as nodes on the network and events that signify the beginning or ending of activities are depicted as arcs or lines between the nodes. The following is an example of a CPM network diagram:

CPM Diagram



Steps in CPM Project Planning:

1. Specify the individual activities.

2. Determine the sequence of those activities.
3. Draw a network diagram.
4. Estimate the completion time for each activity.
5. Identify the critical path (longest path through the network)
6. Update the CPM diagram as the project progresses.

1. **Specify the Individual Activities:** From the work breakdown structure, a listing can be made of all the activities in the project. This listing can be used as the basis for adding sequence and duration information in later steps.
2. **Determine the Sequence of the Activities:** Some activities are dependent on the completion of others. A listing of the immediate predecessors of each activity is useful for constructing the CPM network diagram.
3. **Draw the Network Diagram:** Once the activities and their sequencing have been defined, the CPM diagram can be drawn. CPM originally was developed as an *activity on node* (AON) network, but some project planners prefer to specify the activities on the arcs.
4. **Estimate Activity Completion Time:** The time required to complete each activity can be estimated using past experience or the estimates of knowledgeable persons. CPM is a deterministic model that does not take into account variation in the completion time, so only one number is used for an activity's time estimate.
5. **Identify the Critical Path:** The critical path is the longest-duration path through the network. The significance of the critical path is that the activities that lie on it cannot be delayed without delaying the project. Because of its impact on the entire project, critical path analysis is an important aspect of project planning.
6. **Update CPM Diagram**
   As the project progresses, the actual task completion times will be known and the network diagram can be updated to include this information. A new critical path may emerge, and structural changes may be made in the network if project requirements change.

Q.3. Attempt any *TWO* ***[2×10=20]***

(a) What do you mean by earned value analysis and earned value indicators? Discuss various earned value indicators with examples.

Ans. • EARNED VALUE ANALYSIS:

Earned Value Analysis: it can be defined as - "A method for measuring project performance. It compares the amount of work that was planned with what was actually accomplished to determine if cost and schedule performance is as planned."

- **SV:** Schedule Variance = BCWP - BCWS (negative means behind schedule)
  The Question: Are we ahead or behind production schedule?
  SV = BCWP - BCWS = -$25 (negative, so behind schedule)
- **SPI:** Schedule Performance Index (the RATE you are ahead of or behind schedule) = BCWP/BCWS (Less than 1 means behind schedule).
  The Question: How far ahead or behind schedule are we?
  SPI = BCWC/BCWS = 0.75 (LT 1, so behind schedule)
- **CV:** Cost Variance is measured as follows:
  If BCWP > ACWP, the project is UNDER budget! If BCWP < ACWP, the project is OVER budget!

The Question: Are we on or off budget?
CV = BCWP - ACWP = $75 - $90 = -$15 (negative means over budget)
CPI: Cost Performance Index (the rate you are over or under cost) = BCWP / ACWP (less than 1 means over budget)

The Question: How far on or off budget are we?

CPI = BCWP / ACWP = 0.8333 (less than 1 means over budget)

- **PROJECT FORECAST:**
  - **EAC:** Estimate at Completion =
    - Method 1: BCWP at a point + Estimates to completion.
    - Method 2: The ratio of BAC / CPI.
    - Method 3: (Wilkens) [(BAC - BCWP)/CPI] + ACWP

The Question: At the rate going, how much will all of this cost?

This is a SIMPLIFIED estimate of EAC

EAC = BAC / CPI = $500 / 0.8333 = $600

- **VAC:** Variance at Completion = | BCWP - ACWP | = (forecast) of final cost variance

The Question: How much over or under budget will we be?

VAC = BAC - EAC = $500 - $600 = -$100

- How long will the project really take?

Schedule at Completion = 50 / SPI = 50 / 0.75 = 66 2/3 days

Earned Value Management (EVM) is a project management technique for measuring project progress in an objective manner. EVM has the ability to combine measurements of scope, schedule, and cost in a single integrated system. When properly applied, EVM provides an early warning of performance problems. Additiionally, EVM promises to improve the definition of project scope, prevent scope creep, communicate objective progress to stakeholders, and keep the project team focused on achieving progress.

Indicatores From Earned Value: The indicators from Earned Value (EV) Management, which directly relate to efficiency of project execution, are:

1. Cost Performance Index (CPI) and

2. Schedule Performance Index (SPI). Their definitions are:

Cost Performance Index (CPI index): Represents the amount of work is being completed on a project for every unit of cost spent. CPI is computed by EV/AC. A value of above 1 means that the project is doing well against the budget. Cost Performance Index (CPI) measures the value of the work completed compared to the actual cost of the work completed. CPI is calculated as EV/AC. If CPI is equal to 1 the project is perfectly on budget. If the CPI is greater than 1 the project is under budget, if it's less than 1 the project is over budget.

CPI = BCWP/ACWP (where BCWP is the budgeted cost of work performed, and CWP is the actual cost of work performed).

Schedule Performance Index (SPI index): Represents how close actual work is being completed compared to the schedule. SPI is computed by EV/PV. A value of above one means that the project is doing well against the schedule. Schedule Performance Index (SPI) measures the progress achieved against that which was planned. SPI is calculated as EV/PV. If EV is equal to PV the value of the SP is 1. If EV is less than the PV then the value is less than 1, which means the project is behind schedule. If EV is greater than the PV the value of the SPI is greater than one, which means the project is ahead of schedule. A well performing project should have its SPI as close to 1 as possible, or may be even a little under 1.

SPI = BCWP / BCWS (where BCWS is the budgeted cost of work scheduled).

Schedule VAriance (SV) is a measurement of the schedule performance for a project. It's calculated by taking the Earned Value (EV) and subtracting the Planned Value (PV). Since EV is the actual value earned in the project and the PV is the value our project plan says we should have earned at this point, when we subtract what we planned from the actual we have a good measurement which tells us if we are ahead or behind the baseline schedule according to our project plan. If SV is zero, then

the project is perfectly on schedule. If SV is greater than zero, the project is earning more value than planned thus it's ahead of schedule. If SV is less than zero, the project is earning less value than planned thus it's behind schedule.

Cost Variance (CV) is a measurement of the budget performance for a project. CV is calculated by subtracting Actual Costs (AC) from Earned Value (EV). As we already know, EV is the actual value earned in the project. AC is the actual costs incurred to date, thus when we subtract what our actual costs from the EV we have a good measurement which tells us if we are above or below budget. If CV is zero, then the project is perfectly on budget. If CV is greater than zero, the project is earning more value than planned thus it's under budget. If CV is less than zero, the project is earning less value than planned thus it's over budget.

Q.3.(b) (i) Discuss error tracking with example. Does it affect the SPM schedule?
(ii) Discuss the significance of software project reviews.

Ans. (i) Error tracking or Bug Tracking System: A **bug tracking system** is a software application that is designed to help quality assurance and programmers keep track of reported software bugs in their work. It may be regarded as a sort of issue tracking system.

Many bug-tracking systems, such as those used by most open source software projects, allow users to enter bug reports directly. Other systems are used only internally in a company or organization doing software development. Typically bug tracking systems are integrated with other software project management applications.

Having a bug tracking system is extremely valuable in software development, and they are used extensively by companies developing software products.

The main benefit of a bug-tracking system is to provide a clear centralized overview of development requests (including both bugs and improvements, the boundary is often fuzzy), and their state. The prioritized list of pending items (often called backlog) provides valuable input when defining the product roadmap, or maybe just "the next release".

In a corporate environment, a bug-tracking system may be used to generate reports on the productivity of programmers at fixing bugs. However, this may sometimes yield inaccurate results because different bugs may have different levels of severity and complexity. The severity of a bug may not be directly related to the complexity of fixing the bug. There may be different opinions among the managers and architects.

Yes, it will effect the SPM schedule, because whenever any error comes in to your system , so for that you need some error tracking mechanism. It will take some time to resolve the errors, so it will effect the schedule of your project.

(ii) Significance of software project review: The most obvious value of software reviews (especially formal reviews) is that they can identify issues earlier and more cheaply than they would be identified by testing or by field use (the **defect detection process**). The cost to find and fix a defect by a well-conducted review may be one or two orders of magnitude less than when the same defect is found by test execution or in the field.

A second, but ultimately more important, value of software reviews is that they can be used to train technical authors in the development of extremely low-defect documents, and also to identify and remove process inadequacies that encourage defects (the **defect prevention process**).

This is particularly the case for peer reviews if they are conducted early and often, on samples of work, rather than waiting until the work has been completed. Early and frequent reviews of small work samples can identify systematic errors in the Author's work processes, which can be corrected before further faulty work is done. This improvement in Author skills can dramatically reduce the time it takes to develop a high-quality technical document, and dramatically decrease the error-rate in using the document in downstream processes.

As a general principle, the earlier a technical document is produced, the greater will be the impact of its defects on any downstream activities and their work products. Accordingly, greatest value will accrue from early reviews of

documents such as marketing plans, contracts, project plans and schedules, and requirements specifications. Researchers and practitioners have shown the effectiveness of reviewing process in finding bugs and security issues

(c) Explain the following :

(i) Code review

(ii) Project monitoring and control

Ans. (i) Code Reviews: Code review is systematic examination (often as peer review) of computer source code intended to find and fix mistakes overlooked in the initial development phase, improving both the overall quality of software and the developers' skills.

Code reviews can often find and remove common vulnerabilities such as format string exploits, race conditions, memory leaks and buffer overflows, thereby improving software security. Online software repositories, like anonymous CVS, allow groups of individuals to collaboratively review code. Newer tools designed specifically for collaborative code review, can also facilitate the code review process.

Automated code reviewing software lessens the task of reviewing large chunks of code on the developer by systematically checking source code for known vulnerabilities. Flawfinder and Rough Auditing Tool for Security (RATS) are two well-known examples of code reviewing software.

(ii) Project Monitoring and Control (PMC): PMC is performed to provide understanding and insight into the project's progress so that appropriate corrective actions can be taken when the project's performance deviates significantly from the plan. Aspects of a project's progress include interfaces to other organizations, deliverables, schedules, cost, effort, risk, reviews, verification, validation, and amount of supporting services. Planned management of these aspects is captured in one or more software and/or system plans.

The important modules of the project are:

1. Monitor project activities and resources:

(a) Monitor progress against the schedule by periodically measuring actual completion of activities and milestones.

- Compare this progress against the planned documented schedule.
- Identify significant deviations and trends.

(b) Monitor the project's cost and effort by periodically measuring actual cost and effort expended by project staff.

- Compare the cost and effort to the planned documented estimates.
- Identify significant deviations and trends.

(c) Monitor resources provided and used by the project.

- Compare the resources to the planned documented estimates.
- Identify significant deviations and trends.

(d) Monitor documented risks in the context of the project's current status and circumstances.

2. Monitor work products and project data

(a) Monitor the project's work products and tasks by periodically measuring the actual characteristics of the work products and tasks.

3. Monitor software acquisition:

(a) Reviewing the needs of the project for new acquisitions of software.

(b) Initiate Requests-for-Proposal (RFPs) to satisfy identified needs.

(c) Prepare or update contracts to acquire software.

(d) Monitor suppliers for compliance with contract provisions, on-time software delivery, and quality of software to be delivered.

(e) Monitor acceptance of software that is delivered to assure full compliance with acceptance processes and quality requirements.

4. Monitor commitments:

(a) Monitor internal and external commitments against the plan.

(b) Monitor the status of stakeholder involvement against the plan.

(c) Identify those commitments that have not been satisfied or those that are at significant risk of not being satisfied.

Q.4. Attempt any TWO **[2×10=20]**

(a) Describe the difference between verification and validation. Do both make use of test case design methods and testing strategies?

Ans.

Validation	Verification
Am I building the right product.	Am I building the product right
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.
Am I accessing the right data (in terms of the data required to satisfy the requirement)	Am I accessing the data right (in the right place in the right way).
High level activity	Low level activity
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
Determination of correctness of the final software product by a development project with respect to the user needs and requirements.	Demonstration of consistency, completeness, and correctness of the software at each stage and between each stage of the development life cycle.

Yes we can use both verification and validation for the test case design and testing strategy , but verification is highly used in both of the above.

Q.4. (b) (i) Why is a highly coupled module difficult to unit test? Explain.

(ii) How can project scheduling affect integration testing? Discuss.

Ans. (i) The goal of unit testing is to isolate each part of the program and show that the individual parts are correct. A unit test provides a strict, written contract that the piece of code must satisfy. As a result, it affords several benefits. Unit tests find problems early in the development cycle. So if your module is highly coupled that means the functionalities are depended on others, or highly related with each others , then it is not easy to isolate each part of the module, as they are properly working.

(ii) Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing.

The purpose of integration testing is to verify functional, performance, and reliability requirements placed on major design items. These "design items", i.e. assemblages (or groups of units), are exercised through their interfaces using Black box testing, success and error cases being simulated via appropriate parameter and data inputs. Simulated usage of shared data areas and inter-process communication is tested and individual subsystems are exercised through their input interface. Test cases are constructed to test that all components within assemblages interact correctly, for example across procedure calls or process activations, and this is done after testing individual modules, i.e. unit testing. The overall idea is a "building block" approach, in which verified assemblages are added to a verified base which is then used to support the integration testing of further assemblages.

So it is clear from the above that integration testing is depend on that how many modules are constructed as we are going to integrate the modules. So ,this thing is totally depend on the project schedule.

Q.4.(c) Explain the following:

(i) Software quality attributes

(ii) SEI-CMM

Ans. (i) Software Quality Attributes:

The following are the software quality attributes:

Availability: Is it available when and where I need to use it?

Efficiency: How few system resources does it consume?

Flexibility: How easy is it to add new capabilities?

Installability: How easy is it to correctly install the products ?

Integrity: Does it protect against unauthorized access, data loss?

Interoperability: How easily does it interconnect with other systems?

Maintainability: How easy it is to correct defects or make changes?

Portability: Can it be made to work on other platforms?

Reliability: How long does it run before experiencing a failure?

Reusability: How easily can we use components in other systems?

Robustness: How well does it respond to unanticipated conditions?

Safety: How well does it protect against injury or damage?

Testability: Can I verify that it was implemented correctly?

Usability: How easy is it for people to learn or to use?

(ii) SEI-CMM :

- In 1991 the Software Engineering Institute (SEI) at Carnegie Mellon University introduced the CMM. CMM is a common-sense application of software or Business Process Management and quality improvement concepts to software development and maintenance. Its a community-developed guide for evolving towards a culture of engineering excellence, model for organizational improvement. The underlying structure for reliable and consistent software process assessments and software capabilities evaluations.
- The Capability Maturity Model for Software (CMM) is a framework that describes the key elements of an effective software process. The CMM describes an evolutionary improvement path from an ad hoc, immature process to a mature, disciplined process. The CMM covers practices for planning, engineering and managing software development and maintenance. When followed, these key practices improve the ability of organizations to meet goals for cost, schedule, functionality and product quality.

Q.5. Attempt any TWO **[2×10=20]**

(a) What is the difference between a software configuration management audit and a formal technical review? Can their function be folded into one review? what are the pros and cons?

Ans. The formal technical review focuses on the technical correctness of the configuration object that has been modified. The reviewers assess the SCI (Software Configuratioin Items) to determine consistency with other SCIs, omissions, or potential side effects. A formal technical review should be conducted for all but the most trivial changes.

Objective of Formal Technical Reviews:

- Uncover errors in function, logic and implementation for representation of Software.
- Software represented according to predefined standard.
- Verify software under review meets requirements
- Achieve software developed in Uniform Manner
- Make projects more manageable

A software configuration audit complements the formal technical review by assessing a configuration object for characteristics that are generally not considered during review. The audit asks and answers the following questions:

1. Has the change specified in the ECO been made? Have any additional modifications been incorporated?

2. Has a formal technical review been conducted to assess technical correctness?

3. Has the software process been followed and have software engineering standards been properly applied?

4. Has the change been "highlighted" in the SCI? Have the change date and change author been specified? Do the attributes of the configuratioin object reflect the change?

5. Have SCM procedures for noting the change, recording it, and reporting it been followed?

In some cases, the audit questions are asked as part of a formal technical review. However, when SCM is a formal activity, the SCM audit is conducted separately by the quality assurance group.

An audit may be part of the formal tech review (FTR). Example, at the FTR you would probably want to make sure that all the items are in the package to be reviewed and that all the functionality has been tested and passed. So, the testing would not be part of the review, but the records showing that the product had passed are in the package (including copies of testing problem reports and corrective actions taken to get to the "Pass" conditioin.)

On the other hand, depending upon factors such the size, complexity, length-of-schedule for development and delivery, or management mandates, audits might also be called for outside the FTR.

(b) Differentiate between the following with examples:

(i) Known risks and predictable risks

(ii) Change control and version control

Ans. (i) Known Risks :

- That can be uncovered after careful evaluation of the project plan, the business, and technical environment in which the product is being developed
- Example : Unrealistic delivery rate

Predictable Risks :

- Extrapolated from past project experience
- Example : Staff turnover

(ii) Version control: A *version control system* (also known as a Revision Control System) is a repository of files, often the files for the source code of computer programs, with monitored access. Every change made to the source is tracked, along with who made the change, why they made it, and references to problems fixed, or enhancements introduced, by the change.

Version control systems are essential for any form of distributed, collaborative development. Whether it is the *history* of a wiki page or massive software development project, the ability to track each change as it was made, and to reverse changes when necessary can make all the difference between a well managed and controlled process and an uncontrolled "first come, first serve" system. It can also serve as a mechanism for due diligence for software projects.

***(ii)* Change Control:** Change control within Quality management system (QMS) and Information Technology (IT) systems is a formal process used to ensure that changes to a product or system are introduced in a controlled and coordinated manner. It reduces the possibility that unnecessary changes will be introduced to a system without forethought, introducing faults into the system or undoing changes made by other users of software. The goals of a change control procedure usually include minimal disruption to services, reduction in back-up activities, and cost-effective utilization of resources involved in implementing change.

A process for managing changes to an established system baseline:

— Assign change roles and responsibilities
— Process change requests
— Specify required change activities
— Track change history and approvals
— Establish new system baselines and continue version control.

(c) Write short notes on the following:

(i) CASE Tools

(ii) Risk Monitoring

Ans. CASE Tools: Computer-aided software engineering (CASE) is the scientific application of a set of tools and methods to a software system which is meant to result in high-quality, defect-free, and maintainable software products. It also refers to methods for the development of information systems together with automated tools that can be used in the software development process

Tools: CASE tools are a class of software that automates many of the activities involved in various life cycle phases. For example, when establishing the functional requirements of a

proposed application, prototyping tools can be used to develop graphic models of application screens to assist end users to visualize how an application will look after development. Subsequently, system designers can use automated design tools to transform the prototyped functional requirements into detailed design documents. Programmers can then use automated code generators to convert the design documents into code. Automated tools can be used collectively, as mentioned, or individually. For example, prototyping tools could be used to define application requirements that get passed to design technicians who convert the requirements into detailed designs in a traditional manner using flowcharts and narrative documents, without the assistance of automated design software

Existing CASE Environments can be classified along 4 different dimensions :

1. Life-Cycle Support
2. Integration Dimension
3. Construction Dimension
4. Knowledge Based CASE dimension

(ii) Risk Monitoring: Risk monitoring is the last major element of risk management - *but certainly not the least important!*. Risk management is a process of organizing and planning - just as important as strategic, financial, marketing and human resources planning. Like any good planning, the process should be continual or on-going.

Once your basic risk management plan is in place, monitoring risk means to review it and update it continuously.

- Identify new risks as soon as possible
- Decide where and how to handle that risk
- Look for other risks that might be reduced or eliminated and no longer need coverage
- Check operating volumes - they change so that coverage levels need to change...

Risk management really is an ongoing process.

Tools Available for Risk Control: Risk Metrics are appropriate metrics that will aid in monitoring risks on the project. These include risk events, probability, value and impact. Timeframe, type, priority and status are also part of this list.

Monitoring and control tools include:

- Work Breakdown Structure;
- Project budget both estimates and actuals;
- Project schedule;
- Earned value of project activities whether it be actual point-in-time or forecasting;
- Project resource list and plan; and
- Change control log and forms.

Risk triggers are actions, events or circumstances that, if ignored, will cause the occurrence of a risk event. The project manager needs to identify potential triggers that would indicate that a risk event will occur and to ensure that these triggers are visible to the project team. He/ she will need to monitor these triggers frequently.

Earned Value for Risk Analysis: Earned value provides several analytical opportunities. This analysis can be based on work packages or activities or on the entire project. It can be done in two (2) ways: point in time measurement or forecasting.

Incorporating Risk Reviews into the Management Plan: The project manager must establish a window of time to monitor the project. He should also review the schedule, budget and change control log for potential risks. The risk plan is reviewed after each occurrence of the risk and the probability and impact of each risk even is reassessed. It is important that risk management be part of the project's status meetings.

Results of Risk Control:

- The project status is known;
- Corrective action has been taken;
- The risk management plan has been updated;
- The budget (including contingency and reserve) has been updated;
- The schedule has been updated; and
- Lessons learned on the project with respect to the particular risk(s) have need prepared and disseminated.