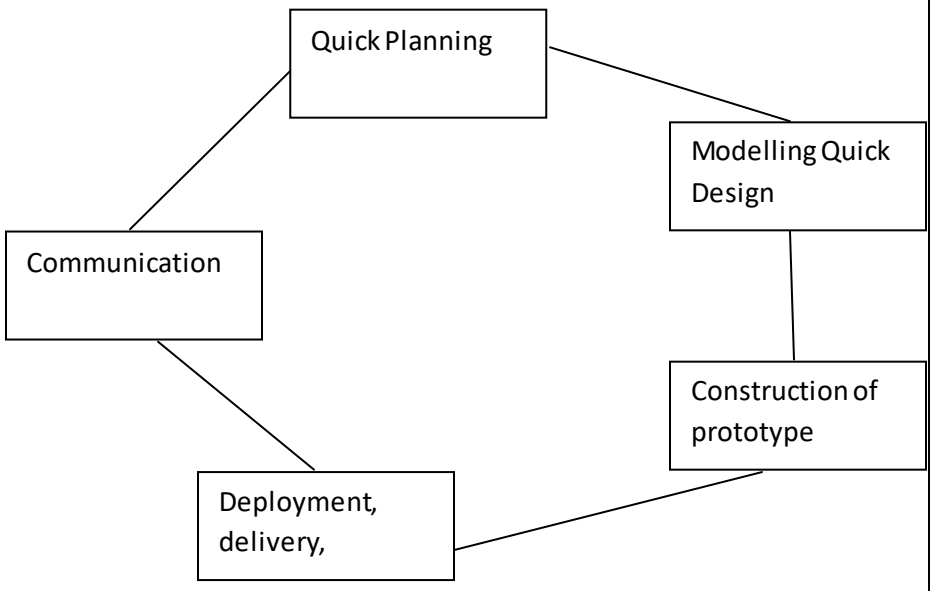
WINTER– 18 EXAMINATION

Subject Name: Software Engineering **Model Answer** Subject Code: 17513

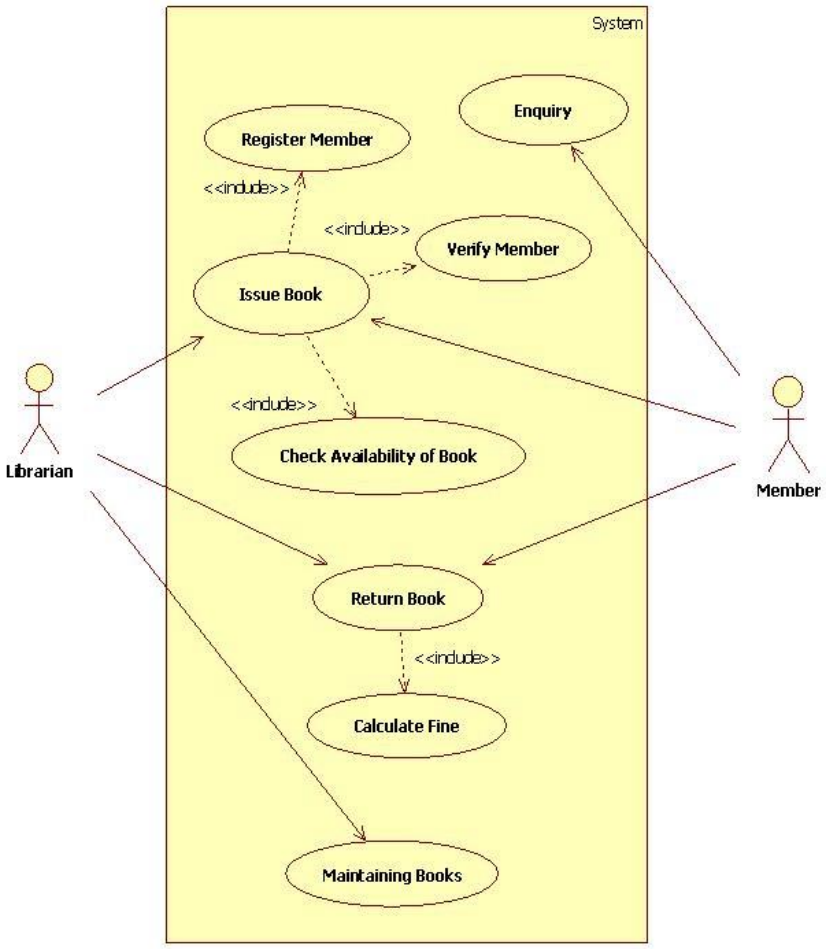
Important Instructions to examiners:

1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
3) The language errors such as grammatical, spelling errors should not be given more Importance (Not applicable for subject English and Communication Skills.
4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by candidate and model answer may vary. The examiner may give credit for any equivalent figure drawn.
5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
6) In case of some questions credit may be given by judgement on part of examiner of relevant answer based on candidate's understanding.
7) For programming language papers, credit may be given to any other program based on equivalent concept.

Q. No.	Sub Q. N.	Answer	Marking Scheme
1.		**Attempt any FIVE of the following:**	**20 M**
	a	**Explain Prototyping process model using suitable diagram.**	4M
	Ans	 **Fig Protyping Model** 1. In Prototyping model initailly requirment gathering is done.	diagram 2 marks, explanation 2 marks

		2. Developer and customer define overall objective and identify areas needing more requirement gathering. 3. Then a quick design is prepared. 4. This design represents what will be visible to user in input and output format. 5. From the quick design prototype is prepared. 6. Customer evaluates the prototype in order to refine the requirement intial prototype satisfy customer requirement. 7. When a working prototype is build the developer uses existing program to through away the prototype and rebuild the system to high quality. Advantages : • This model is cost effective. • Developer can use known programing language and OS. Dis- Advantages : • Sometimes developer may make implementation compromises to get proto-type working quickly. • There is no Guarantee that developer can use better programming language and OS.	
	b	**What is SRS? State importance of SRS.**	4 M
	Ans	Definition: A software requirements specification (SRS) is a complete description of the behavior of the system to be developed. It includes a set of use cases describe all of the interactions that the users will have with the software. **The importance SRS:** 1. Establish the basis for agreement between the customers and the suppliers on what the software product is to do. 2. The complete description of the functions to be performed by the software specified in the SRS will assist the potential users to determine if the software specified meets their needs or how the software must be modified to meet their needs. 3. Reduce the development effort. The preparation of the SRS forces the various concerned groups in the customer's organization to consider rigorously all of the requirements before design begins and reduces later redesign, recoding, and retesting.	SRS - 2 marks; importance of SRS - 2 marks

		4. Careful review of the requirements in the SRS can reveal omissions, misunderstandings, and inconsistencies early in the development cycle when these problems are easier to correct. 5. Provide a basis for estimating costs and schedules. The description of the product to be developed as given in the SRS is a realistic basis for estimating project costs and can be used to obtain approval for bids or price estimates. 6. Provide a baseline for validation and verification. Organizations can develop their validation and Verification plans much more productively from a good SRS. As a part of the development contract, the SRS provides a baseline against which compliance can be measured. 7. Facilitate transfer. The SRS makes it easier to transfer the software product to new users or new machines. 8. Customers thus find it easier to transfer the software to other parts of their organization, and suppliers find it easier to transfer it to new customers. 9. Serve as a basis for enhancement. Because the-SRS discusses the product but not the project that developed it, the SRS serves as a basis for later enhancement of the finished.	
	c	**Draw use case diagram of library system.**	4 M

	Ans		diagram 4 marks
	d	**Compare white box and black box testing.**	4 M

	Ans	**Criteria**	Black box testing	White box testing	Any 4 Points Each – 1 Mark
		Definition	Black Box Testing is a software testing method in which the internal structure/design/implementation of the item being tested is NOT known to the tester	White Box Testing is a software testing method in which the internal structure/ design/ implementation of the item being tested is known to the tester
		Levels Applicable	Mainly applicable to	Mainly applicable to

		To	higher levels of testing: Acceptance Testing System Testing	lower levels of testing: Unit Testing Integration Testing	
		Responsibility	Generally independent Software Testers	Generally Software Developers	
		Programming Knowledge	Not Required	Required	
		Implementation Knowledge	Not Required	Required	
		Basis for Test Cases	Requirement Specifications	Detail Design	
	e	**Describe ways of tracking project schedule.**			4 M
	Ans	Tracking of project can be accomplished in different ways: 1) This is a way in which periodic project status meetings are conducted where each and every team member reports progress & problems about their tasks. 2) In the second way the result of all reviews is evaluated by conducting s/w engineering process throughout the project. 3) In other way the formal project milestones are determined which have been completed by the scheduled data. 4) There could be a comparison of actual start date with the planned start date for each & every project task included in the list from the resource table. 5) Using informal meeting with the practitioner's to get their subjective evaluation of development to date & problems on the prospect. 6) Another method which can be used called earned value analysis to assess progress or development quantitatively.			Any 4 ways of Project tracking - 4 marks
	f	**Define SQA. List SQA Activities.**			4 M
	Ans	Software quality assurance is composed of a variety of tasks associated with two different constituencies - the software engineers who do technical work and an SQA group that has responsibility for quality assurance planning, oversight, record keeping, analysis, and reporting. Software engineers address quality (and perform quality assurance and quality control activities) by applying solid technical methods and measures, conducting formal technical reviews, and performing well-planned software testing. **Activities of SQA**: **1) Prepare an SQA plan for a project.** The plan is developed during project planning and is reviewed by all interested parties. Quality assurance			Definition of SQA – 1 marks; any three Activities of SQA – 3 marks

activities performed by the software engineering team and the SQA group are governed by the plan.

The plan identifies

- evaluations to be performed
- audits and reviews to be performed
- standards that are applicable to the project
- procedures for error reporting and tracking
- documents to be produced by the SQA group
- amount of feedback provided to the software project team

2) Participate in the development of the project's software process description. The software team selects a process for the work to be performed. The SQA group reviews the process description for compliance with organizational policy, internal software standards, externally imposed standards (e.g., ISO-9001), and other parts of the software project plan.

3) Review software engineering activities to verify compliance with the defined software process. The SQA group identifies, documents, and tracks deviations from the process and verifies that corrections have been made.

4) Audits designated software work products to verify compliance with those defined as part of the software process. The SQA group reviews selected work products; identifies, documents, and tracks deviations; verifies that corrections have been made; and periodically reports the results of its work to the project manager.

5) Ensure that deviations in software work and work products are documented and handled according to a documented procedure. Deviations may be encountered in the project plan, process description, applicable standards, or technical work products.

6) Records any noncompliance and reports to senior management.
Noncompliance items are tracked until they are resolved.

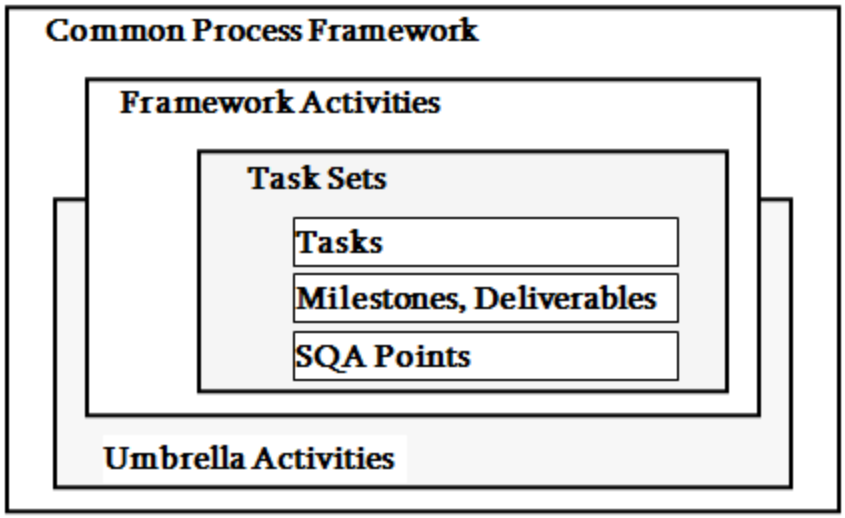
Or

Software quality assurance is composed of a variety of tasks associated with two different constituencies - the software engineers who do technical work and an SQA group that has responsibility for quality assurance planning, oversight, record keeping, analysis, and reporting. Software engineers address quality (and perform quality assurance and quality control activities) by applying solid technical methods and measures, conducting formal technical reviews, and performing well-planned software testing.

Definition of SQA – 1 marks; list of Activities of SQA – 3 marks

Activities of SQA:

		1. Evaluations to be performed 2. Prepares an SQA plan for a project. 3. Participates in the development of the project's software process description. 4. Reviews software engineering activities to verify compliance with the defined software process. 5. Audits designated software work products to verify compliance with those defined as part of the software process. 6. Ensures that deviations in software work and work products are documented and handled according to a documented procedure. 7. Records any noncompliance and reports to senior management. 8. Audits and reviews to be performed 9. Standards those are applicable to the project 10. Procedures for error reporting and tracking 11. Documents to be produced by the SQA group 12. Amount of feedback provided to the software project team.	
	g	**Explain any four software categories.**	4 M
	Ans	**1. System Software:** System Software is a collection of programs written to serve other programs. Some system software (e.g.- compliers, editors, and file management utilities) processes complex, but determinate information structures. Other system applications (e.g. - operating system components, drivers, networking software, telecommunications processors) process largely indeterminate data. **2. Application Software:** Application Software consists of standalone programs that solve a specific business need. Application in this area process business or technical data in a way that facilities business operations or management / technical decision making. **3. Engineering / Scientific Software:** Formerly characterized by —number crunching algorithms, engineering and scientific software applications range from astronomy to volcano logy. **4. Embedded Software:** Embedded Software resides within a product or system and is used to implement and control features and functions for the end-user and for the system itself. Embedded software can perform limited and esoteric functions (e.g. keypad control for a microwave oven, keypad of washing machine) **5. Web – applications:** —Web Apps‖, span a wide array of applications. Web	Any 4 Categories: 1 mark per category

		apps are evolving into sophisticated computing environments that not only provide standalone features, computing functions, and content to the end user, but also are integrated with corporate databases and business applications. **6. Artificial Intelligence Software:** AI Software makes use of non–numerical algorithms to solve complex problems that are not amenable to computation or straightforward analysis. Applications within this area include robotics, expert systems, pattern recognition (image and voice), artificial neural networks, theorem proving, and game playing.	
2		**Attempt any FOUR of the following:**	**16 M**
	a	**Describe basic framework and umbrella activities of generic process framework.**	4 M
	Ans	A process framework establishes the foundation for a complete software process by identifying a small number of framework activities that are applicable to all software projects; In addition, the process framework encompasses a set of umbrella activities that are applicable across the entire software process.  **Figure: Chart of Process Framework** **Basic framework activities:** 1. **Communication:** This framework activity involves heavy communication & collaboration with the customer (and the stakeholders) and encompasses requirements gathering and other related activities. 2. **Planning:** This activity establishes a plan for the software engineering work that follows. It describes the technical tasks to be conducted; the risks are analyzed. Project tracking should be done. Deadline is fixed. 3. **Modeling:** This activity encompasses the creation of models that allow the developer & the customer to better understand software requirements & the	Basic framework activities Description-: 2 marks, Umbrella activities Description-: 2marks

		design that will achieve those requirements. 4. **Construction:** This activity combines code generation and the testing that is required uncovering errors in the code. 5. **Deployment:** The software is delivered to the customer who evaluates the delivered product and provides feedback based on the evaluation. **Umbrella Activities:** Generic views of SE are complemented by a set of umbrella activities. **1. Software Project tracking and control:** It allows the software team to access progress against the project plan and takes necessary action to maintain schedule. Umbrella activities occur throughout the software process and focus primarily on project management, tracking and control. **2. Risk Management**: Assess risks that are likely to affect performance and quality of project. **3. Software quality assurance**: Define and conduct activities to ensure software quality. **4. Formal Technical Review**: Assess Software Engineering Work products to uncover and remove errors before they are shifted to next level of activity. **5. Measurement:** Defines and collects process, project and product measures to assist the team in delivering the software that meets customer needs can be used in conjunction with all framework and umbrella activities. **6. Software configuration Management (SCM):** Manages and effects the changes throughout the software process. **7. Reusability management:** Defines criteria for work product reuse (including software components) and establishes the mechanism to achieve reusable components. **8. Work product preparation and production:** Includes activities for creating work product such as models, documents, forms, reports etc.	
	b	**List seven tasks of requirement engineering.**	4 M
	Ans	Requirement Engineering:- Software process perspective, requirements engineering is a major software engineering action that begins during the communication activity and continues into the modeling activity. It must be adapted to the needs of the process, the project, the product, and the people doing the work. **Tasks included Requirement Engineering:-**	Requirement Engineering-def-optional requirement engineering task - 4 Marks

		i. Inception ii. Elicitation iii. Elaboration iv. Negotiation v. Specification vi. Validation vii. Requirements management	
	c	**Explain following design concepts:** **(i) Abstraction** **(ii) Pattern** **(iii) Refinement** **(iv) Refactoring**	4 M
	Ans	i) **Abstraction:** When we use modular approach to find the solution to the problem, there are chances of having different levels of abstraction at the topmost level of abstraction; a solution is present in broad terms with the help of language of the problem environment. At the lower levels of abstraction a more detailed description of the solution is provided. Data abstraction is named collection of data that describes a data object. ii**) Patterns:** A pattern can be referred as a thing that gives the essence of a proven solution to recurring problem in a particular context. Stated in another way, a design pattern describes a design structure that solves a particular design problem within specific context. Iii) **Refinement:** Top-down design strategy is stepwise refinement .A program is developed by consecutively refining levels of procedural detail. iv**) Refactoring**: It is a reorganization technique that simplifies the design of a component without changing its function or behaviour. When software is re-factored, the existing design is examined for redundancy, unused design elements, inefficient or unnecessary algorithms, poorly constructed data structures, or any other design failures that can be corrected to yield a better design.	each design concept explanation 1 marks each
	d	**What is need of debugging? List characteristics of bugs.**	4 M
	Ans	**Need of Debugging** • Debugging is an important Aspect in any Programming. • This is where debugging Terminology figures out. • Debugging is the stem by stem process of finding errors or bugs in a	need of debugging 2 marks, characteristics of bugs 2 marks

Program so that the bugs can be removed to make the program function in the way it was desired.

characteristics of bugs:

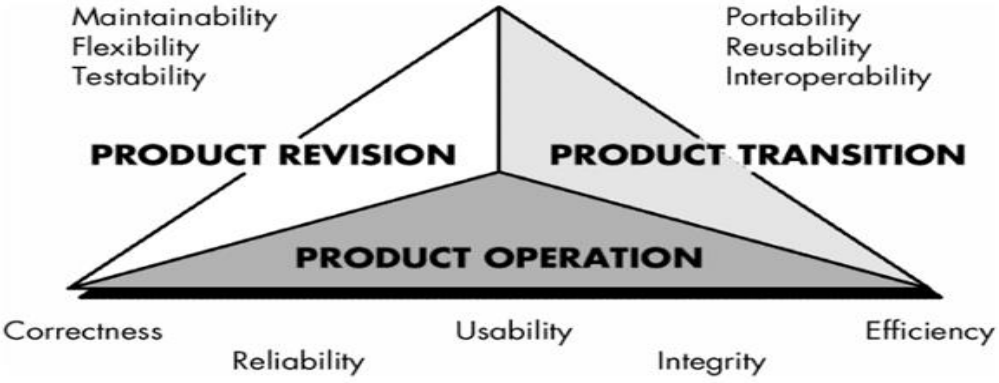
1. While correcting another error symptom may temporally disappear.
2. The sysmtom and cause may be gegraphtycally remote.
3. Non errors may cause the sysmtom for Example round up in Accuracy
4. The reason of sysmtom could be human error and are not easily traceable.
5. The symtom may result of timing problem rather than processing problem.
6. Reproducing input condition in accurate manner might be difficult.
7. The Cause of symtom could be no of Task running on different processor at the same time.

e **Differentiate between PERT and CPM.** 4 M

Ans

PERT	CPM
The Program Evaluation and Review Technique (PERT) is a project management technique or tool which is suitable for projects that have unpredictable activities.	Critical Path Method (CPM) is a project management tool which is suitable for projects that have predictable activities
PERT uses three estimates for the time that it can be completed.	CPM uses a single estimate for the time that a project can be completed.
PERT is a probabilistic project management tool.	CPM is a deterministic project management tool.
PERT does not allows project management planners to determine which aspect of the project to sacrifice when a trade-off is needed in order to complete the project.	CPM allows project management planners to determine which aspect of the project to sacrifice when a trade-off is needed in order to complete the project.
PERT tool is basically a tool for planning	CPM also allows an explicit estimate of control of time, costs in addition to time, therefore CPM can control both time and cost.
PERT is more suitable for R&D related	CPM is best suited for routine and those projects where the project is performed for projects where time and cost estimates can the first time and the estimate of duration be accurately calculated are uncertain.

Any 4 Points Each – 1 Mark

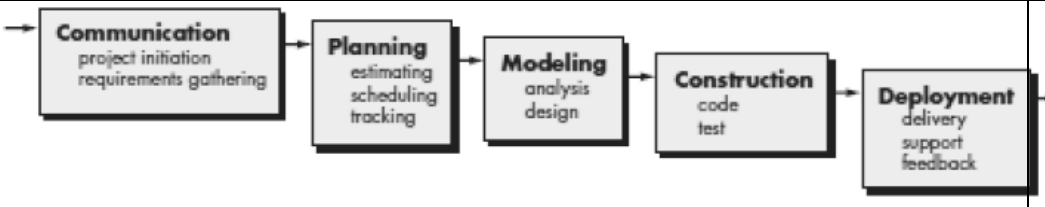
		PERT uses event oriented Network. | CPM uses activity oriented network.	
	f	**Explain McCall's quality factor.**	4 M
	Ans	The factors that affect S/W quality can be categorized in two broad groups: 1. Factors that can be directly measured (defects uncovered during testing) 2. Factors that can be measured only indirectly (Usability and maintainability)  The S/W quality factors shown above focus on three important aspects of a S/W product: i. Its operational characteristics ii. ii. Its ability to undergo change iii. iii. Its adaptability to new environments The various factors of quality are: **(a) Correctness**: The extent to which a program satisfies its specifications and fulfills the customer's mission objectives. **(b) Reliability:** The extent to which a program can be expected to perform its intended function with required precision. **(c) Efficiency:** The amount of computing resources and code required to perform is function. **(d) Integrity:** The extent to which access to S/W or data by unauthorized persons can be controlled. **(e) Usability:** The effort required to learn, operate, prepare input for, and interpret output of a program. **(f) Maintainability:** The effort required to locate and fix errors in a program.	Diagram – optional , Any Four quality factors - 1 Mark each

		(g) Flexibility: The effort required to modify an operational program. **(h) Testability:** The effort required to test a program to ensure that it performs its intended function. **(i) Portability:** The effort required to transfer the program from one hardware and/or software system environment to another. **(j) Reusability:** The extent to which a program can be reused in other applications related to the packaging and scope f the functions that the program performs. **(k) Interoperability**: The effort required to couple one system to another.	
3		. **Attempt any FOUR of the following:**	**16 M**
	a	**What are goals of PSP and TSP? Explain their framework activities.**	4 M
	Ans	**Goals of PSP** The Personal Software Process (PSP) emphasizes personal measurement of both the work product that is produced and the resultant quality of the work product. **The goal of TSP** is to build a "self-directed" project team that organizes itself to produce high-quality software. PSP model defines following five frame work activities: **Planning**- isolates requirements, develops size and resource estimates. Tests are identified and project schedule is created. **High level design**: External specification for each component to be constructed is developed and a component design is created. **High level design review**: formal verification methods are applied to uncover errors in the design. **Development:** component level design is refined and reviewed. Code is generated, reviewed, compiled and tested. **Post-mortem:** Using measures and matrix collected, the effectiveness of the process is determined. They provide guidance for improvement. TSP defines the following framework activities: **project launch, high-level design, implementation, integration and test,** and **postmortem.** **Project Launch:** It reviews core objective and describes the TSP structure and content. It assigns terms and roles to developers and describes the customer needs statement. It also establishes team and individual goals. **High Level Design:** It creates high-level design, specifies the design, and inspects the design develop an integration test plan. **Implementation:** Implementation uses the TSP to implement modules/unit, creates a detailed design of modules/units, reviews the design, translates the design to code, review the code, compile and test the modules/units and analyze the quality of the modules/units. **Integration and Test:** Testing builds and integrates these builds into a system. It conducts a system test and produce user documentation. **Postmortem:** It conducts a postmortem analysis, writes a cycle report and produce peer and team evaluations.	PSP Goal 1 Mark; TSP Goal 1 Mark; PSP Framework Activities 1 Mark; TSP Frame Work Activities 1 Mark

	b	**State software deployment principles.**	4 M
	Ans	**Principle 1: Manage customer's expectations.** It always happens that customer wants more than he has started earlier as his requirements. It may be the case that customer gets disappointed, even after getting all his requirements satisfied. Hence at time of delivery developer must have skills to manage customer's expectations. **Principle 2: Assembly and test complete delivery package.** It is not the case that the deliverable package is _only software'. The customer must get all supporting and essential help from developer's side. **Principle 3: Record-keeping mechanism must be established for customer support.** Customer support is important factor in deployment phase. If proper support is not provided, customer will not be satisfied. Hence support should be well planned and with record-keeping mechanism. **Principle 4: Provide essential instructions, documentations and manual.** Many times, developer thinks —when project is successful deliverable part is only working program‖. But realty is that working program is just part of software product. Actual project delivery includes all documentations, help files and guidance for handling the software by user. **Principle 5: Don't deliver any defective or buggy software to the customer.** In incremental type of software, software organizations may deliver some defective software to the customer by giving assurance that the defects will be removed in next increment.	Any 4 Principles: 1 Mark Each
	c	**State objectives of analysis modelling.**	4 M
	Ans	Describe Customer needs Establish a basis for software design Define a set of requirements that can be validated once the software is built.	All 3 Objectives 4 Marks
	d	**List and explain types of system testing.**	4 M
	Ans	**Types of System testing** **Recovery Testing** **Security Testing** **Stress Testing** **Performance Testing** **a. Recovery Testing** Many computer-based systems recover from the fault and resume processing within a pre-specified time for the system should be "fault" tolerant. The recovery testing for the software to fail in a variety of ways and verifies that the recovery is properly performed. If the recovery is automatic, re-initialization, check pointing mechanism, data recovery and restart are evaluated for correctness. If the recovery needs human intervention, mean-time-to- repairs (MTTR) is evaluated to find whether it is within acceptable limits. **b. Security Testing** Security testing attempts to verify that protection mechanisms built into a system protect it from improper penetration. During security testing, the tester plays the role of the individual who desires to penetrate the system. The tester may attempt to acquire passwords through external clerical means, may attack	Each Testing Type 1 Mark; 1 Mark shall be awarded if only list is given

		the system with custom software designed to break down any defenses that have been constructed, may overwhelm the system, thereby denying service to others, may purposely cause system errors, hoping to penetrate during recovery, may browse through insecure data, hoping to find the key to system entry. **c. Stress Testing** Stress tests are designed to confront programs with abnormal situations. Stress testing executes a system in a manner that demands resources in abnormal quantity, frequency, or volume. For example 1) special tests may be designed that generate ten interrupts per second, when one or two is the average rate, 2) input data rates may be increased by an order of magnitude to determine how input functions will respond, 3) test cases that require maximum memory or other resources are executed, 4) test cases that may cause thrashing in a virtual operating system are designed, 5) test cases that may cause excessive hunting for disk-resident data are created. Essentially, the tester attempts to break the program.	
	e	**Explain people and project factor in software management spectrum.**	4 M
	Ans	**The People** • People factor is very much important in the process of software development. • There are following areas for software people like, recruiting, selection performance management, training, compensation, career development, organization and work design, and team/culture development. • Organizations achieve high levels of maturity in the people management area. **The Project** • We conduct planned and controlled software projects for one primary reason it is the only known way to manage complexity. • 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.	People factor – 2 Marks; Project Factor 2 Marks
	f	**Compare CMMI and ISO.**	4 M
	Ans		Any 4 Points of differentiation; 1 Mark each

CMMI	ISO
CMMI is a set of related "best practices" derived from industry leaders and relates to product engineering and software development.	ISO is a certification tool that certifies businesses whose processes conform to the laid down standards.
CMMI is a process model.	ISO is an audit standard.

		CMMI is rigid and extends only to businesses developing software intensive systems.	ISO is flexible and applicable to all manufacturing industries.	
		CMMI compares the existing processes to industry best practices	ISO requires adjustment of existing processes to confirm to the specific ISO requirements.	
		CMMI is more focused, complex, and aligned with business objectives	ISO is flexible, wider in scope and not directly linked to business objectives.	
		It provides grade for process maturity.	ISO provides pass or fail criteria.	
		It reconnects the mechanism for step by step progress through its successive maturity levels.	ISO does not specify sequence of steps required to establish the quality system.	
		CMM is specially developed for software industry	It applies to any type of industry.	
		CMM has 5 levels: Initial Repeatable Defined Managed Optimization	ISO 9000 has no levels.	
4		**Attempt any TWO of the following:**		**16 M**
	a	**Explain waterfall model with neat diagram and its advantages and disadvantages.**		8 M
	Ans	 The waterfall model is a traditional method, sometimes called the classic life cycle. This is one of the initial models. As the figure implies stages are cascaded and shall be developed one after the other. It suggests a systematic, sequential approach to software development that begins with customer specification of requirements and progresses through, *communication, planning, modeling construction and deployment*. In other words one stage should be completed before the other begins. Hence, when all the requirements are elicited by the customer, analyzed for		Description 3 Marks; Diagram: 1 Mark; Any two Advantages 2 Marks; Any two Disadvantages 2 Marks

		completeness and consistency, documented as per requirements, the development and design activities commence. One of the main needs of this model is the user's explicit prescription of complete requirements at the start of development. For developers it is useful to layout what they need to do at the initial stages. Its simplicity makes it easy to explain to customers who may not be aware of software development process. It makes explicit with intermediate products to begin at every stage of development. One of the biggest limitation is it does not reflect the way code is really developed. Problem is well understood but software is developed with great deal of iteration. Often this is a solution to a problem which was not solved earlier and hence software developers shall have extensive experience to develop such application; as neither the user nor the developers are aware of the key factors affecting the desired outcome and the time needed. Hence at times the software development process may remain uncontrolled. Today software work is fast paced and subject to a never-ending stream of changes in features, functions and information content. Waterfall model is inappropriate for such work. This model is useful in situation where the requirements are fixed and work proceeds to completion in a linear manner. **Advantages of waterfall model:** 1. This model is simple and easy to understand and use. 2. It is easy to manage due to the rigidity of the model – each phase has specific deliverables and a review process. 3. In this model phases are processed and completed one at a time. Phases do not overlap. 4. Waterfall model works well for smaller projects where requirements are very well understood. Disadvantages of waterfall model: 1. Once an application is in the testing stage, it is very difficult to go back and change something that was not well-thought out in the concept stage. 2. No working software is produced until late during the life cycle. 3. High amounts of risk and uncertainty. 4. Not a good model for complex and object-oriented projects. 5. Poor model for long and ongoing projects. 6. Not suitable for the projects where requirements are at a moderate to high risk of changing.	
	b	**With neat diagram explain the translation of analysis model into design model.**	8 M

Ans

Diagram 3 Marks;
Description 5 Marks

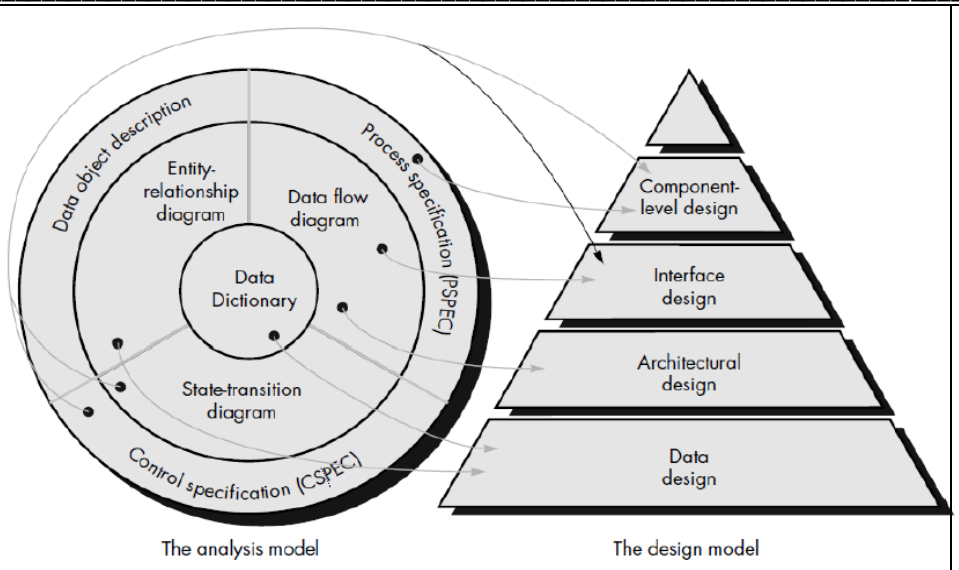


FIGURE 13.1 Translating the analysis model into a software design

Software design sits at the technical kernel of software engineering and is applied regardless of the software process model that is used. Beginning once software requirements have been analyzed and specified, software design is the first of three technical activities design, code generation, and test—that are required to build and verify the software. Each activity transforms information in a manner that ultimately results in validated computer software.

Each of the elements of the analysis model provides information that is necessary to create the four design models required for a complete specification of design. The flow of information during software design is illustrated in above figure.

Software requirements, manifested by the data, functional, and behavioral models, feed the design task. Using one of a number of design methods (discussed in later chapters), the design task produces a data design, an architectural design, an interface design, and a component design.

The data design transforms the information domain model created during analysis into the data structures that will be required to implement the software. The data objects and relationships defined in the entity relationship diagram and the detailed data content depicted in the data dictionary provide the basis for the data design activity.

Part of data design may occur in conjunction with the design of software architecture. More detailed data design occurs as each software component is designed. The architectural design defines the relationship between major structural elements of the software, the ―design patterns‖ that can be used to achieve the requirements that have been defined for the system, and the constraints that affect the way in which architectural design patterns

can be applied.

The architectural design representation the framework of a computer-based system—can be derived from the system specification, the analysis model, and the interaction of subsystems defined within the analysis model. The interface design describes how the software communicates within itself, with systems that interoperate with it, and with humans who use it. An interface implies a flow of information (e.g., data and/or control) and a specific type of behavior. Therefore, data and control flow diagrams provide much of the information required for interface design.

The component-level design transforms structural elements of the software architecture into a procedural description of software components. Information obtained from the PSPEC, CSPEC, and STD serve as the basis for component design.

c **Describe DMAIC and DMDAV approach of six sigma.** — 8 M

Ans

DMAIC

The DMAIC project methodology has five phases:

Define the system, the voice of the customer and their requirements, and the project goals, specifically.

Measure key aspects of the current process and collect relevant data.

Analyze the data to investigate and verify cause-and-effect relationships. Determine what the relationships are, and attempt to ensure that all factors have been considered. Seek out root cause of the defect under investigation.

Improve or optimize the current process based upon data analysis using techniques such as design of experiments or mistake proofing, and standard work to create a new, future state process. Set up pilot runs to establish process capability.

Control the future state process to ensure that any deviations from target are corrected before they result in defects. Implement control systems such as statistical process control, production boards, visual workplaces, and continuously monitor the process. Some organizations add a Recognize step at the beginning, which is to recognize the right problem to work on.

DMA Activities : 4 Marks; IC : 2 Marks; DV 2 Marks

DMADV

The DMADV project methodology, known as DFSS ("Design For Six Sigma"), features five phases it has first three phases same as DMAIC :

Design an improved alternative, best suited per analysis in the previous step

Verify the design, set up pilot runs, implement the production process and hand it over to the process owner(s).

5 **Attempt any FOUR of the following:** — **16 M**

a **Differentiate between prescriptive and agile process model.** — 4 M

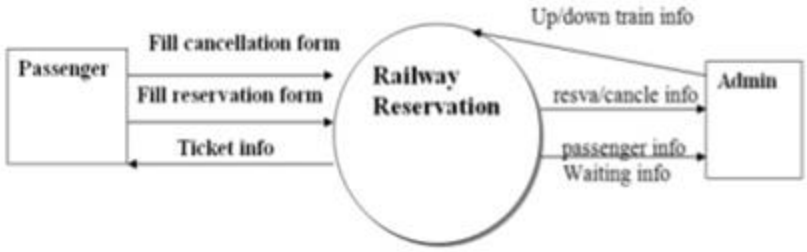
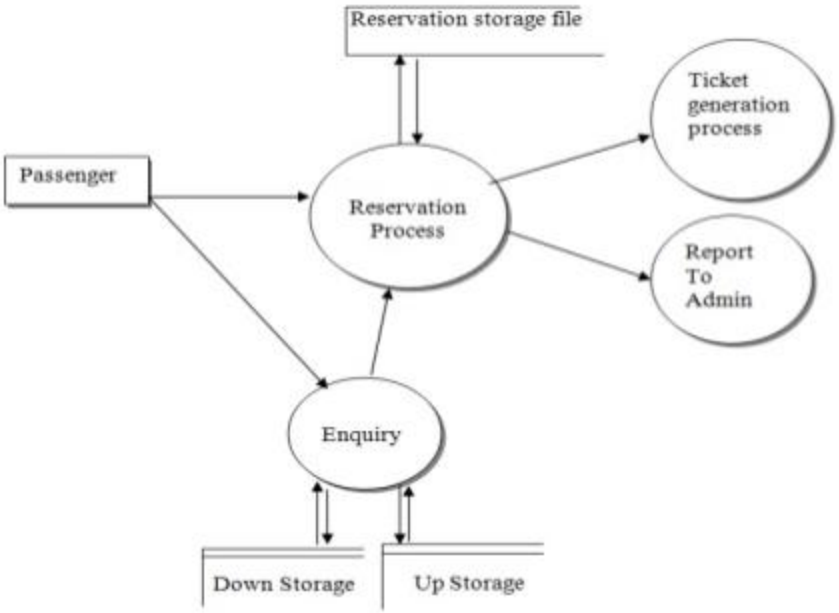
Ans

Sr N	Parameters	Prescriptive process model	Agile process

Any four points each point 1 mark

			o			model	
			1	Quality	It changes from analysis>Design>Code>Test	It focus on all aspects of SDLC at any given time	
			2	Quality control	Detection & Fixing during system and regression testing at the last phase of project	Early detection and fixing in each sprint followed by stabilization	
			3	Continual improvement	Lesson learned from previous release implemented in next release.	Lesson learned from Previous sprint will be implemented in the next sprint.	
			4	Risk	No risk identification. firefighting during testing phase	Early identification and mitigation in every sprints	
			5	Post mortem/retrospection	After Every place	After Every sprint in Retrospection meeting	
			6	Customer Feedback	At the end of project	At the end of every sprint	
	b	**List and explain software engineering core principles.**					4 M
	Ans	**1. Reason it all exists. Provide value to the user** i. The software system exists to provide value for the user. ii. Before specifying the problem the requirement and the specifications have					List- 1 mark, explain any three principles – 3 marks

		to be laid down. iii. The hardware and the software platform to be decided for implementation. **2.Keep it simple stupid** i. The terms and the design used for development of the project should be kept simple and easily understandable. ii. All the terms used should be easy to facilitate the basic concept of the project. **3.Maintain the vision** i. A clear vision is important for the development of a software. ii. Compromising the architectural vision of the project weakens the development of the software. ii. The developer should hold the vision and ensure the successful development and deployment of the software. **4. What you reproduce, someone else will have to consume. (implement knowing someone else will have to understand what you are doing)** i. Always specify, design and implement knowing that someone else is going to understand what is being developed. ii. Customers for the product development is very large. iii. Design the data structure and the implementation keeping implementation in mind and the end user. iv. Code with the concern that the product has to be implemented and maintained by the end user. **5.Be open to the future** i. The system designed today should be adaptable to the development and changes in the future at a low cost. ii. There should not be much changes to the software to adopt to the new changes in the future development. **6.Plan ahead for reuse:** i. The design and specifications should be developed in such a way that they can be reused for other implementations. ii. The code and the design should be well documented for the use in future. **7. Think!** . Before designing and implementation a proper thought should be to the end result. . Proper data structure and the design and implementation strategy should be developed if the software needs modification in the future.	
	c	**Draw data flow diagram for railway reservation system.**	4 M

	Ans	Context level or 0-level DFD:  1-level DFD:  	level 0 DFD- 2marks, level 1 DFD- 2marks
	d	**Explain the following terms:** **i) Software testing ii) Test case iii) Test plan iv) Bug**	4 M
	Ans	**1) Software testing:** Testing is a process of executing a program with the intent of finding an error. Testing is a set of activities that can be planned in advance and conducted systematically. For this reason, a template for software testing—a set of steps into which you can place specific test case design techniques and testing methods. **2) Test case:** A test case is a specification of the inputs, execution conditions, testing procedure, and expected results that define a single test to be executed to achieve a particular software testing objective, such as to exercise a	Each point 1mark

		particular program path or to verify compliance with a specific requirement. **3) Test plan:** A Test plan is a document describing software testing scope and activities. It is the basis for formally testing any software/product in a project. test plan: A document describing the scope, approach, resources and schedule of intended test activities. It is a record of the test planning process. **4) Bug:** A software bug is an error, flaw, failure or fault in a computer program or system that causes it to produce an incorrect or unexpected result, or to behave in unintended ways.	
	e	**What is need of SCM? List functions of SCM repository**.	4 M
	Ans	**Need of SCM** i. Also called software configuration management (SCM) ii. It is an umbrella activity that is applied throughout the software process iii. It's goal is to maximize productivity by minimizing mistakes caused by confusion when coordinating software development iv. SCM identifies, organizes, and controls modifications to the software being built by a software development team v. SCM activities are formulated to identify change, control change, ensure that change is being properly implemented, and report changes to others who may have an interest **Functions of SCM repository** 1.**Data integrity: Validates** entries, ensures consistency, cascades modifications 2.**Information sharing: Shares** information among developers and tools, manages and controls multi-user access 3.**Tool integration:** Establishes a data model that can be accessed by many software engineering tools, controls access to the data 4.**Data integration: Allows** various SCM tasks to be performed on one or more CSCIs 5.**Methodology enforcement: Defines** an entity-relationship model for the repository that implies a specific process model for software engineering 6. **Document standardization: Defines** objects in the repository to guarantee a standard approach for creation of software engineering documents.	Need of SCM- 2 marks, functions of SCM repository- 2 marks
	f	**Describe levels of CMMI technique.**	4 M
	Ans	1.CMMI (Capability Maturity Model Integration) is a proven industry framework to improve product quality and development efficiency for both	complete explanation of all levels 4 marks

hardware and software

i. Sponsored by US Department of Defence in cooperation with Carnegie Mellon University and the Software Engineering Institute (SEI)

ii. Many companies have been involved in CMMI definition such as Motorola and Ericsson.

Level 1 Initial	• At level 1, the process is usually chaotic and ad hoc • A capability is characterized on the basis of the individuals and not of the organization • Progress not measured • Products developed are often schedule and over budget • Wide variations in the schedule, cost, functionality, and quality targets
Level 2 Managed	• Requirement Management • Estimate project parameters like cost, schedule, and functionality • Measure actual progress • Develop plans and process • Software project standards are defined • Identify and control products, problem reports changes, etc. • Processes may differ between projects
Level-3 Defined	• Clarify customer requirements • Solve design requirements, develop an implementation process • Makes sure that product meets the requirements and intended use • Analyze decisions systematically • Rectify and control potential problems
Level-4 Quantitatively Managed	• Manages the project's processes and sub-processes statistically • Understand process performance, quantitatively manage the organization's project
Level-5	• Detect and remove the cause of defects early • Identify and deploy new tools and process

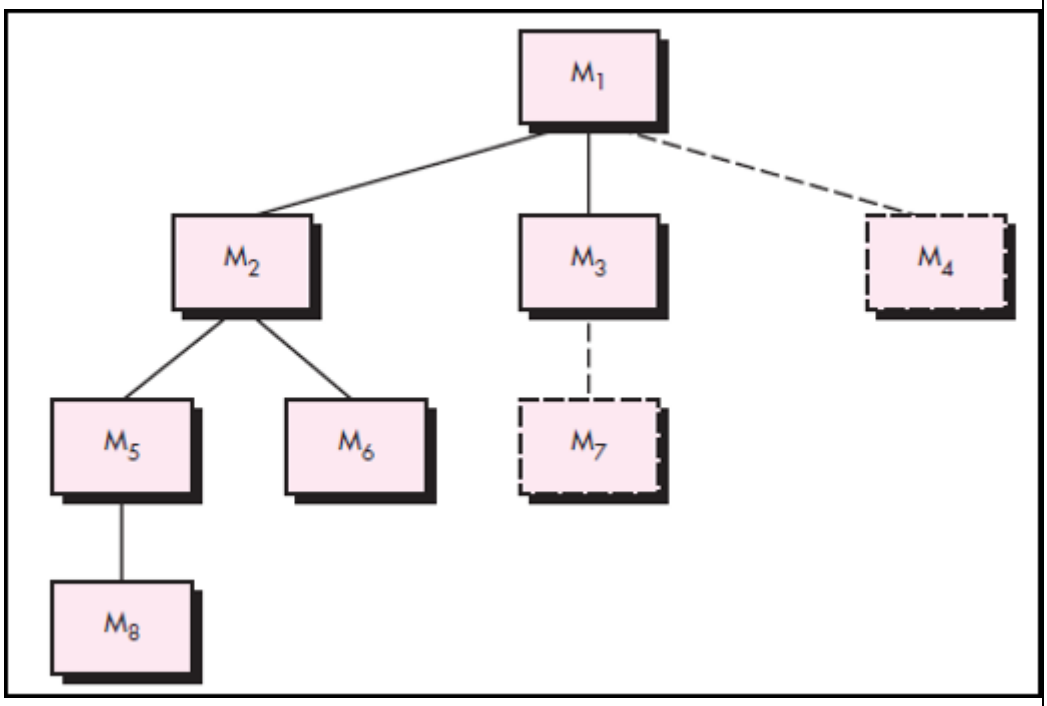
		Optimizing improvements to meet needs and business objectives	
6		**Attempt any TWO of the following:**	**16 M**
	a	**Explain design modelling practices in software engineering.**	8 M
	Ans	***i. Design should be traceable to the analysis model*** ➢The analysis model provides the information domain of the problem. ➢The design should be able to translate the information about the design, sub problems and the component level design. ***ii. Always consider the architecture of the system to be built*** ➢The software development is the skeleton to the product development. ➢It effects the implementation with respect to design, data structure, and the manner in which testing can be performed. ***iii.Data design is as important as algorithm design*** ➢Data design is an important architectural strategy. ➢The data should be designed as the flow of algorithm is designed for the implementation and deployment of the software. ***iv.Internal and external interfaces must be designed with care*** ➢Data flows between the modules should be designed with simplicity and processing efficiency. ➢A well designed interface makes integration easier. ***v.User interface design should be tuned to the needs of the end-user and must focus on use of user*** ➢The user interface is the visible implementation of the software. ➢The poor interface always leads to the perception that the software is bad. ***vi.Component-level design should be functionally independent*** ➢Each sub function or the module developed should focus on the perfect working functionality of that module. ***vii.Components should be loosely coupled to one another and to the external environment*** ➢Coupling is accomplished through many ways like message passing, component interface and global data. ➢When the level of coupling increases, error propagation increases with overall maintainability of software decreases. ➢Component coupling should be kept as low as possible.	Any 8 principles 1 mark each

viii.Design models should be easy to understand

- The design models are created for easy communication of information to the users about the code
- The design model should be easily understandable by the tester.
- If the software design model is difficult to understand and communicate than it is not the effective model.

ix.Design should be developed iteratively

- The design work should be open for improvement at the first level.
- At the final stages it should be easy and creative.

	b	**Explain integration testing with suitable example.**	8 M
	Ans	**Top-down integration testing** Top-down integration testing is an incremental approach to construction of the software architecture. Modules are integrated by moving downward through the control hierarchy, beginning with the main control module (main program). Modules subordinate (and ultimately subordinate) to the main control module are incorporated into the structure in either a depth-first or breadth-first manner.  The integration process is performed in a series of five steps: 1. The main control module is used as a test driver and stubs are substituted for all components directly subordinate to the main control module.	diagram- 2 marks, explanation- 2marks

2. Depending on the integration approach selected (i.e., depth or breadth first), subordinate stubs are replaced one at a time with actual components.

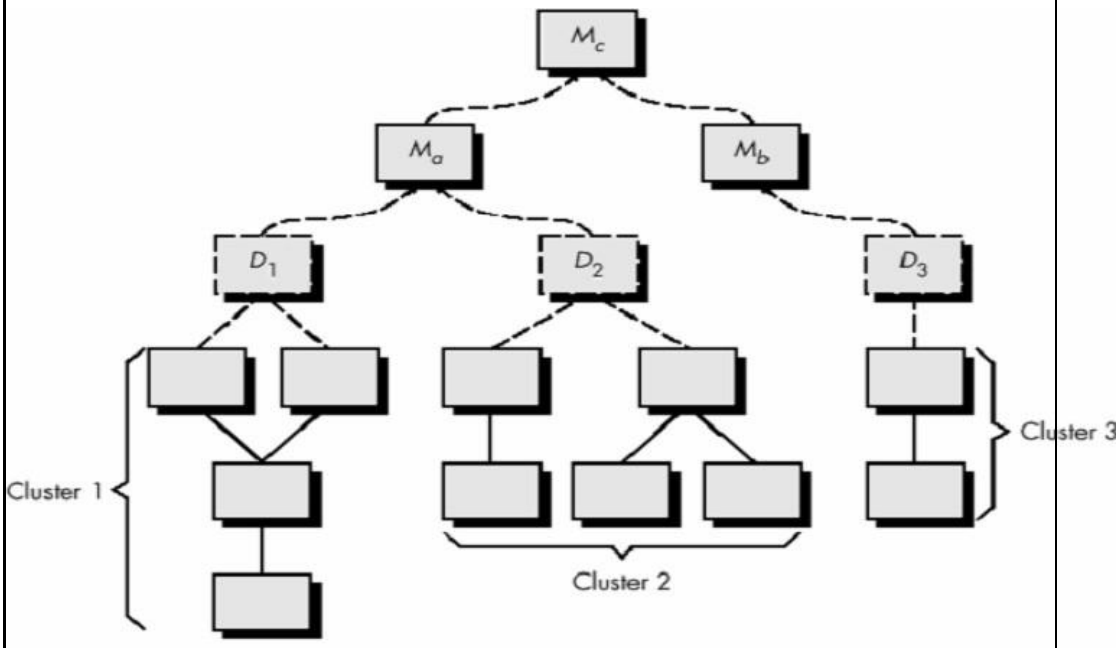
3. Tests are conducted as each component is integrated.

4. On completion of each set of tests, another stub is replaced with the real component.

Regression testing (discussed later in this section) may be conducted to ensure that new errors have not been introduced.

Bottom-up integration testing

- *Bottom-up integration testing*, as its name implies, begins construction and testing with *atomic modules* (i.e., components at the lowest levels in the program structure).
- Because components are integrated from the bottom up, the functionality provided by components subordinate to a given level is always available and the need for stubs is eliminated.



A bottom-up integration strategy may be implemented with the following steps:

1. Low-level components are combined into clusters (sometimes called *builds*) that perform a specific software sub function.

2. A *driver* (a control program for testing) is written to coordinate test case input and output.

3. The cluster is tested.

		4. Drivers are removed and clusters are combined moving upward in the program structure.	
	c	**What is software risk? Describe working of RMMM plan.**	8 M
	Ans :	Software risk always involves two characteristics: 1. *Uncertainty*—the risk may or may not happen; that is, there are no 100 per cent probable risks. 2. *Loss*—if the risk becomes a reality, unwanted consequences or losses will occur. When risks are analysed, it is important to quantify the level of uncertainty and the degree of loss associated with each risk. **RMMM Plan** Risk mitigation, monitoring, and management (RMMM) plan. ·A risk management strategy can be included in the software project plan or the risk management steps can be organized into a separate Risk Mitigation, Monitoring and Management Plan. The RMMM plan documents all work performed as part of risk analysis and is used by the project manager as part of the overall project plan. Once RMMM has been documented and the project has begun, risk mitigation and monitoring steps commence. ·Risk mitigation is a problem avoidance activity. ·Risk monitoring is a project tracking activity with three primary objectives: (1) To assess whether predicted risks do, in fact, occur; (2) To ensure that risk aversion steps defined for the risk are being properly applied; and (3) To collect information that can be used for future risk analysis. ·Another job of risk monitoring is to attempt to allocate origin (what risk(s) caused which problems throughout the project). ·An effective strategy must consider three issues: • Risk avoidance • Risk monitoring • Risk management and contingency planning ·If a software team adopts a proactive approach to risk; avoidance is always the best strategy. This is achieved by developing a plan for **risk mitigation**. To mitigate this risk, project management must develop a strategy for reducing turnover. ·Among the possible steps to be taken are • Meet with current staff to determine causes for turnover (e.g., poor working conditions, low pay, and competitive job market). • Mitigate those causes that are under our control before the project starts. • Once the project commences, assume turnover will occur and develop techniques to ensure continuity when people leave. • Organize project teams so that information about each development activity is widely dispersed. • Define documentation standards and establish mechanisms to be sure that documents are developed in a timely manner.	Definition- 2 marks, RMMM plan – 6marks

		• Conduct peer reviews of all work (so that more than one person is "up to speed). • Assign a backup staff member for every critical technologist. As the project proceeds, risk monitoring activities commence. The project manager monitors factors that may provide an indication of whether the risk is becoming more or less likely. ·In the case of high staff turnover, the following factors can be monitored: • General attitude of team members based on project pressures. • The degree to which the team has jelled. • Interpersonal relationships among team members. • Potential problems with compensation and benefits. • The availability of jobs within the company and outside it.	