MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER-15 EXAMINATION
Model Answer

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

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 judgment 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.

Marks 20

1.   Answer **any five** of the following:

a) Explain software engineering as a layered technology approach.

(Description - 3 Marks, Diagram - 1 Mark)

Ans: Software Engineering – A Layered Technology approach.
Software engineering is a layered technology. Referring to Figure any engineering approach (including software engineering) must rest on an organizational commitment to quality. Total quality management and similar philosophies foster a continuous process improvement culture, and this culture ultimately leads to the development of increasingly more mature approaches to software engineering. The bedrock that supports software engineering is a quality focus. The foundation for software engineering is the process layer. Software engineering process is the glue that holds the technology layers together and enables rational and timely development of computer software. Process defines a framework for a set of key process areas that must be established for effective delivery of software engineering technology. The key process areas form the basis for management control of software projects and establish the context in which technical methods are applied, work products (models, documents, data, reports, forms, etc.) are produced, milestones are established, quality is ensured, and change is properly managed.

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER-15 EXAMINATION
<u>**Model Answer**</u>

Subject Code: 17513   Subject Name: Software Engineering

Software engineering methods provide the technical how-to's for building software. Methods encompass a broad array of tasks that include requirements analysis, design, program construction, testing, and support. Software engineering methods rely on a set of basic principles that govern each area of the technology and include modeling activities and other descriptive techniques. Software engineering tools provide automated or semi-automated support for the process and the methods. When tools are integrated so that information created by one tool can be used by another, a system for the support of software development, called computer-aided software engineering, is established.

b) Enlist core principles of software engineering practice.

(For each principle -1 Mark; Any 4(four) principles)

Ans: **The First Principle: The Reason It All Exists**

The Second Principle: KISS (Keep It Simple, Stupid!)

The Third Principle: Maintain the Vision

The Fourth Principle: What You Produce, Others Will Consume.

The Fifth Principle: Be Open to the Future

The Sixth Principle: Plan Ahead for Reuse.

Seventh Principle: Think!

Subject Code: 17513   Subject Name: Software Engineering

c) Describe data objects and data attributes.

(Data object Definition - 1 Mark, Example - 1 Mark, Data attribute Definition - 1 Mark,

Example - 1 Mark)

Ans: **Data objects.**
A data object is a representation of almost any composite information that must be understood by software. By composite information, we mean something that has a number of different properties or attributes. Therefore, width (a single value) would not be a valid data object, but dimensions (incorporating height, width, and depth) could be defined as an object. A data object can be an external entity (e.g., anything that produces or consumes information), a thing (e.g., a report or a display), an occurrence (e.g., a telephone call) or event (e.g., an alarm), a role (e.g., salesperson), an organizational unit (e.g., accounting department), a place (e.g., a warehouse), or a structure (e.g., a file). For example, a person or a car can be viewed as a data object in the sense that either can be defined in terms of a set of attributes. The data object description incorporates the data object and all of its attributes. In this case, a car is defined in terms of make, model, ID number, body type, color and owner.

Data attribute
Attributes: Attributes define the properties of a data object and take on one of three different characteristics.
They can be used to (1) name an instance of the data object, (2) describe the instance, or (3) make reference to another instance in another table. In addition, one or more of the attributes must be defined as an identifier—that is, the identifier attribute becomes a "key" when we want to find an instance of the data object. In some cases, values for the identifier(s) are unique, although this is not a requirement. Referring to the data object car, a reasonable identifier might be the ID number. The set of attributes that is appropriate for a given data object is determined through an understanding of the problem context. The attributes for car might serve well for an application that would be used by a Department of Motor Vehicles, but these attributes would be useless for an automobile company that needs manufacturing control software.

SUMMER-15 EXAMINATION
Model Answer

d) List four objectives of testing.

(4 objectives - 4 Marks, 1Mark per objective)

Ans: **List of 4 objectives of testing**
1. Testing is a process of executing a program with the intent of finding an error.
2. A good test case is one that has a high probability of finding an as-yet undiscovered error.
3. A successful test is one that uncovers an as-yet-undiscovered error
4. All tests should be traceable to customer requirements.
5. A good test has a high probability of finding an error.
6. A good test is not redundant
7. A good test should be "best of breed"
8. A good test should be neither too simple nor too complex.

e) List four basic principles of project scheduling.

(4 principles - 4 Marks, 1Mark per principle)

Ans:

Basic principles software project scheduling:
Compartmentalization: The project must be compartmentalized into a number of manageable activities and tasks. To accomplish compartmentalization, both the product and the process are decomposed.
I**nterdependency:** The interdependency of each compartmentalized activity or task must be determined. Some tasks must occur in sequence while others can occur in parallel. Some activities cannot commence until the work product produced by another is available. Other activities can occur independently.
Time allocation: Each task to be scheduled must be allocated some number of work units (e.g., person-days of effort). In addition, each task must be assigned a start date and a completion date that are a function of the interdependencies and whether work will be conducted on a full-time or part-time basis.
Effort validation: Every project has a defined number of staff members. As time allocation occurs, the project manager must ensure that no more than the allocated number of people has been scheduled at any given time.
Defined responsibilities: Every task that is scheduled should be assigned to a specific team member. Defined outcomes: Every task that is scheduled should have a defined outcome.
Defined milestones: Every task or group of tasks should be associated with a project milestone. Program evaluation and review technique (PERT) and critical path method (CPM) are two project scheduling Methods that can be applied to software development.

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER-15 EXAMINATION
Model Answer

Subject Code: 17513   Subject Name: Software Engineering

f) What steps are required to perform statistical SQA?

(For each point-1 Mark)

Ans: Statistical quality assurance reflects a growing trend throughout industry to become more quantitative about quality. For software, statistical quality assurance implies the following steps:

1. Information about software defects is collected and categorized.
2. An attempt is made to trace each defect to its underlying cause (e.g., nonconformance to specifications, design error, violation of standards, and poor communication with the customer)
3. Using the Pareto principle (80 percent of the defects can be traced to 20 percent of all possible causes), isolate the 20 percent (the "vital few").
4. Once the vital few causes have been identified, move to correct the problems that have caused the defects.

g) State any four attributes of good software.

(For each attribute - 1 Mark)

Ans: 1. Functionality: It refers to the degree of performance of the software against its intended purpose.
2. Reliability: It refers to the ability of the software to provide desired functionality under the given conditions.
3. Usability: It refers to the extent to which the software can be used with ease and simple.
4. Maintainability: Software must evolve to meet changing needs.
5. Dependability: Software must be trustworthy.
6. Efficiency: Software should not make wasteful of system resources.
7. Acceptability: Software must accepted by the users for which it was designed.
8. Portability: It refers to the ease with which software developers can transfer software from one platform to another, without changes.
9. Integrity: It refers to the degree to which unauthorized access to the software can be prevented.
10. Robustness: It refers to the degree to which the software can keep on functioning in spite of being provided with invalid data.

SUMMER-15 EXAMINATION
Model Answer

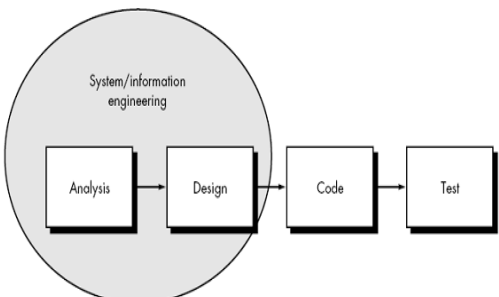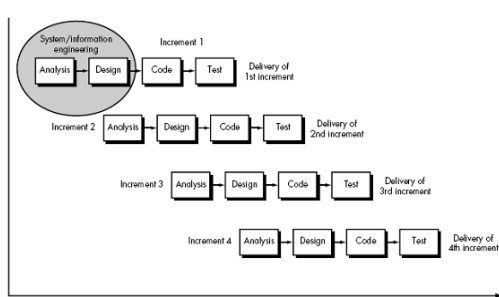**Subject Code: 17513**   **Subject Name: Software Engineering**

2. Answer **any four** of the following: **Marks 16**

a) Differentiate between waterfall model and incremental model.
   (Each Difference- 1Mark, any four difference)

Ans:

Waterfall model	Incremental model
The linear sequential model suggests a systematic, sequential approach to software development that begins at the system level and progresses through analysis, design, coding, testing, and support.	The incremental model combines elements of the linear sequential model (applied repetitively) with the iterative philosophy of prototyping
All phases are clearly defined.	Few of the steps are defined
One of the most systematic methods for software development.	It is iterative in nature.
Being oldest, this is one of the time tested methods	New model for development
Real projects rarely follow sequential model.	It provides on the rigid nature of sequential approach
It is often difficult for the customer to state all requirements explicitly	This method is of great help when organization is low on staffing.
The working model is available only in the latter part of the development.	This model could be time consuming.

SUMMER-15 EXAMINATION
Model Answer

b) What is SRS?

(SRS: Definition - 1 Mark, Explanation - 3 Marks)

Ans: 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. In addition to use cases, the SRS contains functional requirements and nonfunctional requirements. Functional requirements define the internal workings of the software: that is, the calculations, technical details, data manipulation and processing, and other specific functionality that shows how the use cases are to be satisfied. Non-functional requirements impose constraints on the design or implementation (such as performance requirements, quality standards, or design constraints).

The purpose of SRS document is providing a detailed overview of software product, its parameters and goals. SRS document describes the project's target audience and its user interface, hardware and software requirements. It defines how client, team and audience see the product and its functionality.

The importance of standard template for SRS documents:
Establish the basis for agreement between the customers and the suppliers on what the software product is to do. 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. 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. Careful review of the requirements in the SRS can reveal omissions, misunderstandings, and inconsistencies early ip the development cycle when these problems are easier to correct.

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. .

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.

Facilitate transfer. The SRS makes it easier to transfer the software product to new users or new machines. 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.

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

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER-15 EXAMINATION
<u>**Model Answer**</u>

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

product. The SRS may need to be altered, but it does provide a foundation for continued production evaluation.

c) Write importance of analysis modeling.

(For each importance – 1 Mark; any four relevant importance shall be consider)

Ans: Importance of analysis modeling:

1. Designing gets easier to the designer
2. Better understanding of system can be accomplished
3. System feasibility can be determined
4. Defines data objects
5. Describes data attributes
6. Establishes data relationships
7. Identifies functions that transform data objects
8. Indicates different states of the system
9. Specifies events that cause the system to change state
10. Refines each model to represent lower levels of abstraction
11. Refines data objects
12. Creates a functional hierarchy represent behavior at different levels of detail

d) State eight characteristics of software bugs.

(For each point- ½ Mark)

Ans:

1. The symptom and the cause may be geographically remote. That is, the symptom may appear in one part of a program, while the cause may actually be located at a site that is far removed. Highly coupled program structures exacerbate this situation.
2. The symptom may disappear (temporarily) when another error is corrected.
3. The symptom may actually be caused by non-errors (e.g., round-off inaccuracies).
4. The symptom may be caused by human error that is not easily traced.
5. The symptom may be a result of timing problems, rather than processing problems.
6. It may be difficult to accurately reproduce input conditions (e.g., a real-time application in which input ordering is indeterminate).
7. The symptom may be intermittent. This is particularly common in embedded systems that couple hardware and software inextricably.
8. The symptom may be due to causes that are distributed across a number of tasks running on different processors

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER-15 EXAMINATION
Model Answer

Subject Code: 17513   Subject Name: Software Engineering

e) Enlist the features of SCM.

(4 features - 4 Marks; 1 Mark per feature)

Ans: SCM Features:
To support SCM, the repository must have a tool set that provides support for the following features:
Versioning. As a project progresses, many versions of individual work products will be created.
The repository must be able to save all of these versions to enable effective management of product releases and to permit developers to go back to previous versions during testing and debugging. The repository must be able to control a wide variety of object types, including text, graphics, bit maps, complex documents, and unique objects like screen and report definitions, object files, test data, and results.
Dependency tracking and change management. The repository manages a wide variety of relationships among the data elements stored in it. These include relationships between enterprise entities and processes, among the parts of an application design, between design components and the enterprise information architecture, between design elements and deliverables, and so on. Some of these relationships are merely associations, and some are dependencies or mandatory relationships.
Requirements tracing. This special function depends on link management and provides the ability to track all the design and construction components and deliverables that result from a specific requirements specification (forward tracing). In addition, it provides the ability to identify which requirement generated any given work product (backward tracing).
Configuration management. A configuration management facility keeps track of a series of configurations representing specific project milestones or production releases.
Audit trails. An audit trail establishes additional information about when, why, and by whom changes are made. Information about the source of changes can be entered as attributes of specific
objects in the repository. A repository trigger mechanism is helpful for prompting the developer or the tool that is being used to initiate entry of audit information (such as the reason for a change) whenever a design element is modified.

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER-15 EXAMINATION
Model Answer

Subject Code: 17513   Subject Name: Software Engineering

f) Describe six sigma for software engineering.

(For DMA - 2 Marks, For DV or IC - 2 Marks)

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, thus yielding an RDMAIC methodology.

OR

DMADV or DFSS
The DMADV project methodology, known as DFSS ("Design For Six Sigma"), features five phases:
Define design goals that are consistent with customer demands and the enterprise strategy.
Measure and identify CTQs (characteristics that are Critical To Quality), product capabilities, production process capability, and risks.
Analyze to develop and design alternatives
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).

SUMMER-15 EXAMINATION
__Model Answer__

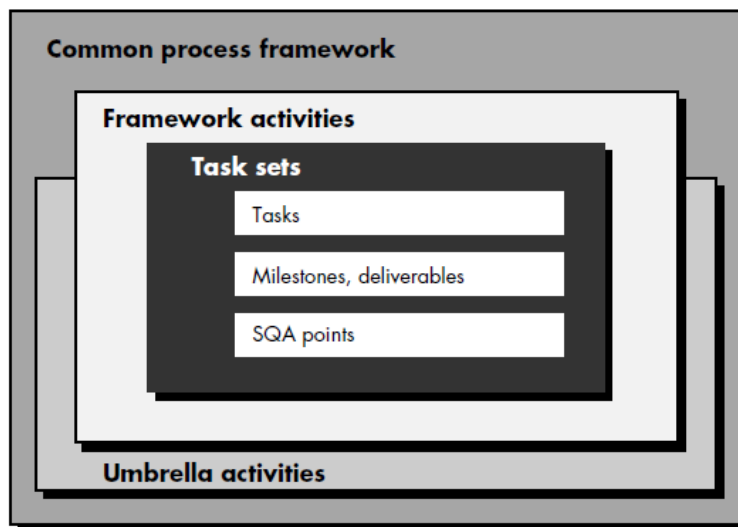**Subject Code: 17513**   **Subject Name: Software Engineering**

3.   Answer **any four** of the following: **Marks 16**

    a)   What do you mean by process framework? Explain with suitable diagram.

      (Process framework diagram - 2 Marks, Explanation – 2 Marks)

Ans: **Diagram:**

1. A process framework is a collection of software activities which forms a complete foundation or base for software development process or software engineering process.
2. It is clear that framework activity is set of software engineering actions. The software engineering activities could be design, program, test, etc.
3. When software product is to be developed, it is very much necessary to follow series of predictable steps. These steps are kind of a road map that will assist to develop the high quality product. This road map used to develop quality product is called as software process.
   These processes are chosen by the software engineers and their team heads to accomplish the task of software product development. The customer or users also play very important role in software processes.
4. Software product is computer software that is designed and built by software engineers. Software product is collection of programs that runs on computer.

Subject Code: 17513 — **Subject Name: Software Engineering**

5. Software product also contains documents in the form of hard copy and soft copy also data in the form of text, numbers, pictures, audio and video etc.
6. Software work products are collection of the programs, documents, and the resultant data produced by the software processes.
7. The framework activities included in the generic model also known as the framework model used for major and vast projects are:
1. Communication.
2. Planning.
3. Modeling.
4. Construction.
5. Deployment.

b) Write four drawback of RAD model.

   (Four drawbacks - 4 Marks, Each drawback- 1 Mark)

Ans:

1. RAD needs enough human resources to create the required number of RAD teams.
2. If developers and customers are not committed to the rapid model, the RAD project fails.
3. Rapid-fire activities need to be completed in very short or small time frame. Time is the major constraint in RAD.
4. RAD has to be modularized in a proper way otherwise creates a lots of confusions and problems.
5. In case of high performance requirement, RAD cannot be ideal model. Still if RAD model has to be used then it can be done by making tune between interface and system components.

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER-15 EXAMINATION
<u>**Model Answer**</u>

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

c) Explain deployment principle.

(Any 4 deployment principles with explanation -4 Marks)

Ans: **Deployment principle:**
1. **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.
2. **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.
3. **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.
4. **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.
5. **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.

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

---

d) What are the characteristics of good design?

(Any 4 characteristics of good design - 4 Marks)

Ans:

1. A design should show architecture that is developed with the help of understandable patterns, architectural styles and different components that are having characteristics of a good design.
2. Also it should be evolving over the period of time that will make testing and implementations easy.
3. Software design needs to be modular. Because of modular design, software can be partitioned logically into elements.
4. A design should depict different representations of components, interfaces, architecture and data.
5. A design has to have appropriate classes and data structures to be implemented. It should be sourced from recognizable data patterns.
6. A design should lead to components that exhibit independent functional characteristics.
7. A design should be done in iterating manner on the basis of information received during software requirement analysis.
8. A design should be depicted using meaningful and efficient notation for the purpose of knowing and communicating its meaning easily.

e) Differentiate between validation and verification.

(Any 4 differences - 4 Marks, per difference - 1 Mark)

Ans:

1. Verification:
1) Verification is a set of activities which ensures that software correctly implements a specific function.
2) Verification evaluates plans, documents, code, requirements and specifications etc.
3) Verification takes place before validation.
4) The input of verification could be check list, review of meetings, plans, walkthroughs, etc.

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER-15 EXAMINATION
Model Answer

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

2. Validation:
1) Validation is different set of activities which ensures that the software has been built is traceable to customer.
2) Validation evaluates product itself.
3) Validation is next step to verification.
4) An input of validation is actual testing of product.

f) What is risk projection? Enlist steps of risk projection.

(Risk projection explanation -1 Mark, Steps – 3Marks)

Ans:

1. Risk projection is called as risk estimation. Risk can be related in two ways as follows:
    1) The possibility or probability that the risk is practical or real.
    2) The results of the problems linked with the risk.
2. The project planners, other managers and technical staff follow four risk projection steps or activities:
    1) Establishment of a scale that follows the supposed possibility of a risk.
    2) Define the results of the risk.
    3) Find out or estimate the impact of the risk on the software project and the software product.
    4) Record the overall correctness of the risk projection so that there will be no confusions or misunderstandings.
3. The objectives of these four steps are to understand risks in manner that show the way to prioritization.
4. No software team has all the resources to address every possible risks with the same degree of severity.
5. After prioritizing risks, the team can allocate resources where they will be having more importance.
6. The impact of the risk refers to the consequences that takes place if the risk occurs.
7. Following are few important points that assess the risk impacts:
    1) Find out the average probability of occurrence value for every component of risk.
    2) Find out impact of every component of the risk on the basis of its criteria.
    3) Enter full data in risk table and carry out the analysis for the results.

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

---

4.   Answer **any four** of the following: **Marks 16**

   a) Explain different decomposition techniques.

    (4 decomposition techniques – any two point -2 Marks each)

Ans: **Software Sizing:**
The accuracy of a software project estimate is predicated on a number of things: (1) the degree to which the planner has properly estimated the size of the product to be built; (2) the ability to translate the size estimate into human effort, calendar time, and dollars (a function of the availability of reliable software metrics from past projects); (3) the degree to which the project plan reflects the abilities of the software team; and (4) the stability of product requirements and the environment that supports the software engineering effort. Because a project estimate is only as good as the estimate of the size of the work to be accomplished, sizing represents the project planner's first major challenge. In the context of project planning, size refers to a quantifiable outcome of the software project. If a direct approach is taken, size can be measured in LOC. If an indirect approach is chosen, size is represented as FP.

Problem-Based Estimation:
lines of code and function points were described as measures from which productivity metrics can be computed. LOC and FP data are used in two ways during software project estimation: (1) as an estimation variable to "size" each element of the software and (2) as baseline metrics collected from past projects and used in conjunction with estimation variables to develop cost and effort projections. LOC and FP estimation are distinct estimation techniques. Yet both have a number of characteristics in common. The project planner begins with a bounded statement of software scope and from this statement attempts to decompose software into problem functions that can each be estimated individually. LOC or FP (the estimation variable) is then estimated for each function. Alternatively, the planner may choose another component for sizing such as classes or objects, changes, or business processes affected.

LOC-Based Estimation:
As an example of LOC and FP problem-based estimation techniques, let us consider a software package to be developed for a computer-aided design application for mechanical components. A review of the System Specification indicates that the software is to execute on an engineering workstation and must interface with various computer graphics peripherals including a mouse, digitizer, high resolution color display and laser printer. Using the *System Specification* as a guide, a preliminary statement of software scope can be developed:

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER-15 EXAMINATION
Model Answer

Subject Code: 17513   Subject Name: Software Engineering

Function	Estimated LOC
User interface and control facilities (UICF)	2,300
Two-dimensional geometric analysis (2DGA)	5,300
Three-dimensional geometric analysis (3DGA)	6,800
Database management (DBM)	3,350
Computer graphics display facilities (CGDF)	4,950
Peripheral control function (PCF)	2,100
Design analysis modules (DAM)	8,400
Estimated lines of code	33,200

FP-Based Estimation:
Decomposition for FP-based estimation focuses on information domain values rather than software functions. Referring to the function point calculation table presented the project planner estimates inputs, outputs, inquiries, files, and external interfaces for the CAD software. For the purposes of this estimate, the complexity weighting factor is assumed to be average. Presents the results of this estimate.

Information domain value	Opt.	Likely	Pess.	Est. count	Weight	FP count
Number of inputs	20	24	30	24	4	97
Number of outputs	12	15	22	16	5	78
Number of inquiries	16	22	28	22	5	88
Number of files	4	4	5	4	10	42
Number of external interfaces	2	2	3	2	7	15
Count total						320

Process-Based Estimation:
The most common technique for estimating a project is to base the estimate on the process that will be used. That is, the process is decomposed into a relatively small set of tasks and the effort required to accomplish each task is estimated. Like the problem-based techniques, process-based estimation begins with a delineation of software functions obtained from the project scope. A series of software process activities must be performed for each function. Functions and related software process activities may be represented as part of a table similar to the one presented. Once problem functions and process activities are melded, the planner estimates the effort (e.g., person-months) that will be required to accomplish each software process activity for each software function. These data

SUMMER-15 EXAMINATION
Model Answer

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

constitute the central matrix of the table. Average labor rates (i.e., cost/unit effort) are then applied to the effort estimated for each process activity. It is very likely the labor rate will vary for each task. Senior staff heavily involved in early activities are generally more expensive than junior staff involved in later design tasks, code generation, and early testing.

Activity →	CC	Planning	Risk analysis	Engineering		Construction release		CE	Totals
Task →				Analysis	Design	Code	Test		
Function ↓									
UICF				0.50	2.50	0.40	5.00	n/a	8.40
2DGA				0.75	4.00	0.60	2.00	n/a	7.35
3DGA				0.50	4.00	1.00	3.00	n/a	8.50
CGDF				0.50	3.00	1.00	1.50	n/a	6.00
DBM				0.50	3.00	0.75	1.50	n/a	5.75
PCF				0.25	2.00	0.50	1.50	n/a	4.25
DAM				0.50	2.00	0.50	2.00	n/a	5.00
Totals	0.25	0.25	0.25	3.50	20.50	4.50	16.50		46.00
% effort	1%	1%	1%	8%	45%	10%	36%		

CC = customer communication   CE = customer evaluation

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER-15 EXAMINATION
Model Answer

Subject Code: 17513   Subject Name: Software Engineering

Process-Based Estimation:
To illustrate the use of process-based estimation, we again consider the CAD software
Introduced in Section 5.6.3. The system configuration and all software functions remain
unchanged and are indicated by project scope. Referring to the completed process-based
table shown in Figure 5.5, estimates of effort (in person-months) for each software
engineering activity are provided for each CAD software function (abbreviated for
brevity). The engineering and construction release activities are subdivided into the major
software engineering tasks shown. Gross estimates of effort are provided for customer
communication, planning, and risk analysis. These are noted in the total row at the bottom
of the table. Horizontal and vertical totals provide an indication of estimated effort
required for analysis, design, code, and test. It should be noted that 53 percent of all effort
is expended on front-end engineering tasks (requirements analysis and design), indicating
the relative importance of this work.

b) Describe integration testing.

(Integration testing description – 4Marks)

Ans:
1) Integration testing can be referred as a systematic technique for the purpose of
   software architecture construction and at the same time carrying out test to find out
   errors related to interfacing.
2) The objective is to take unit tested components and build a program structure that has
   been dictated by design.
3) In most of the cases non-incremental approach is used. There will be a combination of
   all components done in advance. Full program will be tested as a whole.
4) The number of errors can be found out in integration testing but correction of these
   errors is very difficult because separation of causes is complicated as the program
   might be very big.
5) Once the errors are removed, new errors may appear and this process will be continued
   like endless loop.
6) Small incremental are used to construct and test the program in incremental
   integration. Finding out errors and correcting them is easy here. Interfaces are tested
   completely with the systematic test approach.

SUMMER-15 EXAMINATION
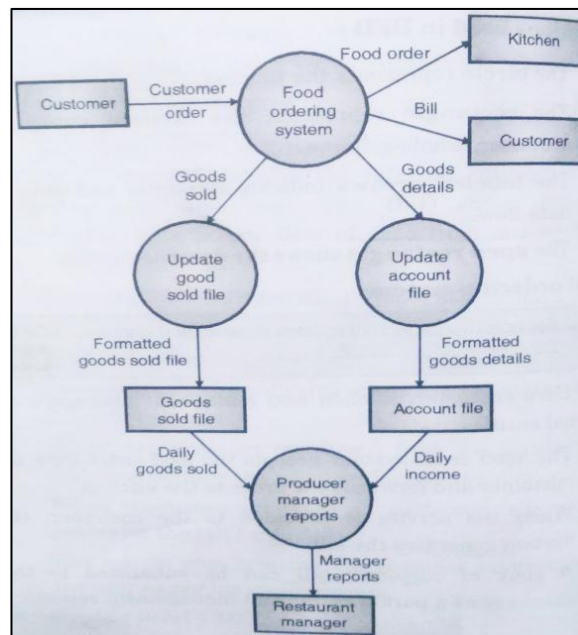**Model Answer**

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

c) What is DFD? Explain level 1 DFD with example.

(Definition of DFD -1 Mark, Example - 2 Marks, Diagram – 1Mark)

Ans: **Diagram:**

1) Data Flow Diagram (DFD) is also called as 'Bubble chart'. This is a graphical technique that represents information flow, and transformer those are applied when data moves from input to output.
2) DFD represents system requirements those becomes program in design.
3) DFD may be further partitioned into different levels to show detailed information flow e.g. level 0, level 1, level 2 etc.
4) DFD focuses on the fact 'what data flow' rather 'how data is processed'.
   Developing level 1 DFD:
5) DFD is used to represent information flow, and the transformers those are applied when data moves from input to output.
6) To show the data flow with more details the DFD is further extended to level 1, level 2, level 3 etc. as per the requirement.
7) The typical value for the DFD is seven. Any system can be well represented with details up to seventh levels.
8) Example: 'Food Ordering System' in the restaurant. (Figure shows the Level 1 DFD for Food Ordering System.

SUMMER-15 EXAMINATION
<u>**Model Answer**</u>

d) Explain cardinality and modality with example.

(Cardinality - 2 Marks, Modality - 2 Marks)

Ans: **Cardinality**:
1) Cardinality specifies number of occurrences of one object related to number of occurrences of another object.
2) That is the maximum number of object relationship is represented by cardinality.
3) Shows different relationship like one to one, one to many and many to many.
4) E.g. many employees occupy one room.
Modality:
1) A modality of relationship is zero if occurrence of relationship is optional and modality of relationship is 1 if occurrence of relationship is mandatory (i.e. compulsory).
2) The modality specifies the minimum number of relationship.
3) Shows maximum 1 to minimum or compulsory 1.
4) E.g. exactly one (maximum 1 and minimum 1) room is occupied by zero or many (maximum many and minimum 0) employees.

e) Explain spiral model with neat diagram.

(Spiral model diagram -2 Marks, description -2 Marks)

Ans: **Diagram:**
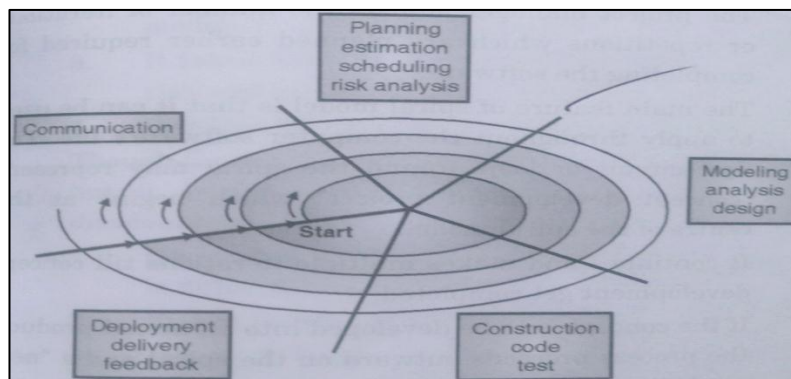
1) Barry, Boeham had proposed the spiral model. Spirals model's software process is combination of an iterative nature of prototyping and controlled and systematic aspects of traditional waterfall model.

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER-15 EXAMINATION
<u>**Model Answer**</u>

Subject Code: 17513   Subject Name: Software Engineering

2) This model is able to produce rapid development in more complicated versions of software.

3) In spiral model, software development takes place in a series of developed releases. In initial stage iterations, the release or model might be present on paper or may be a prototype model.

4) These are kind of designing stages. In later stages, when completing stages come, the software system is actually produced.

5) Every framework activity represents one section of the spiral path. As the development process start, the software team performs activities that are indirect by a path around the spiral model in a clockwise direction. It begins at the centre of spiral model and that is Risk.

6) The project manager adjusts the number of iterations or repetitions which are planned earlier required for completing the software.

7) The main feature of spiral model is that it can be to apply throughout the computer software's life. The first circuit or loop around the spiral may represent "concept development project" which begins at the center of the spiral model.

8) It continues and makes multiple integrations till concept development get completed.

f) Describe Agile process models in detail.

   (Agile model : diagram -2 Marks, Description - 2 Marks)

Ans: **Diagram:**

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

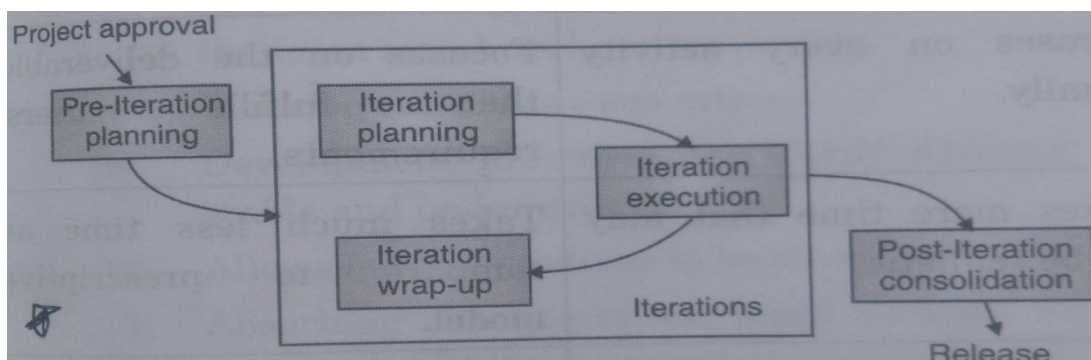SUMMER-15 EXAMINATION
Model Answer

Subject Code: 17513   Subject Name: Software Engineering

1) Agile software development is not only a process but is referred as philosophy.

2) The agile software development focuses on the rapid development of the software product by considering the current market requirements and time limits.

3) Today's market is rapidly changing and unpredictable too. Root of agile software development is in the reality of today's markets.

4) If we compare prescriptive model waterfall with agile software development, we find agile software development is very much useful and practically applicable.

5) Agile software development solves the problems of long time and heavy documentation software development processes.

6) Agile development focuses on face to face or interactive processes them documentation. It helps in quickly transferring the ideas.

7) It does not believe in more and more documentation because it makes difficult to find out the required information. It supports teams to work together with management for supporting technical decision making. This method focuses mainly on the coding because directly deliverable to the users.

8) Agile Software developments saves man power , cost, documentation and most importantly time.

5.   Answer **any two** of the following: **Marks 16**

a) Describe eight principles of good planning.

(Any Eight Principles, For Each Principle-1 Mark)

Ans: **Eight Principles of good planning are:**
Principle 1. Understand the scope of the project. It's impossible to use a road map if you don't know where you're going. Scope provides the software team with a destination.
Principle 2. Involve stakeholders in the planning activity. Stakeholders define priorities and establish project constraints. To accommodate these realities, software engineers must often negotiate order of delivery, time lines, and other project-related issues.
Principle 3. Recognize that planning is iterative. A project plan is never engraved in stone. As work begins, it is very likely that things will change. As a consequence, the plan must be adjusted to accommodate these changes. In addition, iterative, incremental process models dictate re-planning after the delivery of each software increment based on feedback received from users.

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER-15 EXAMINATION
<u>**Model Answer**</u>

Subject Code: 17513   Subject Name: Software Engineering

Principle 4. Estimate based on what you know. The intent of estimation is to provide an indication of effort, cost, and task duration, based on the team's current understanding of the work to be done. If information is vague or unreliable, estimates will be equally unreliable.

Principle 5.Consider risk as you defines the plan. If you have identified risks that have high impact and high probability, contingency planning is necessary. In addition, the project plan (including the schedule) should be adjusted to accommodate the likelihood that one or more of these risks will occur.

Principle 6. Be realistic. People don't work 100 percent of every day. Noise always enters into any human communication. Omissions and ambiguity are facts of life. Change will occur. Even the best software engineers make mistakes. These and other realities should be considered as a project plan is established.

Principle 7.Adjust granularity as you defines the plan. Granularity refers to the level of detail that is introduced as a project plan is developed. A —high-granularity‖ plan provides significant work task detail that is planned over relatively short time increments (so that tracking and control occur frequently). A —low-granularity‖ plan provides broader work tasks that are planned over longer time periods. In general, granularity moves from high to low as the project time line moves away from the current date. Over the next few weeks or months, the project can be planned in significant detail. Activities that won't occur for many months do not require high granularity (too much can change).

Principle 8. Define how you intend to ensure quality. The plan should identify how the software team intends to ensure quality. If technical reviews are to be conducted, they should be scheduled. If pair programming is to be used during construction, it should be explicitly defined within the plan.

Principle 9. Describe how you intend to accommodate change. Even the best planning can be obviated by uncontrolled change. You should identify how changes are to be accommodated as software engineering work proceeds. For example, can the customer request a change at any time? If a change is requested, is the team obliged to implement it immediately? How is the impact and cost of the change assessed?

Principle 10.Track the plan frequently and make adjustments as required. Software projects fall behind schedule one day at a time. Therefore, it makes sense to track progress on a daily basis, looking for problem areas and situations in which scheduled work does not conform to actual work conducted. When slippage is encountered, the plan is adjusted accordingly.

SUMMER-15 EXAMINATION
<u>**Model Answer**</u>

b) With neat diagram explain translation of analysis model into design model.

(For figure-3 Marks, For explanation- 5 Marks)

Ans: 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 analysis model   The design model
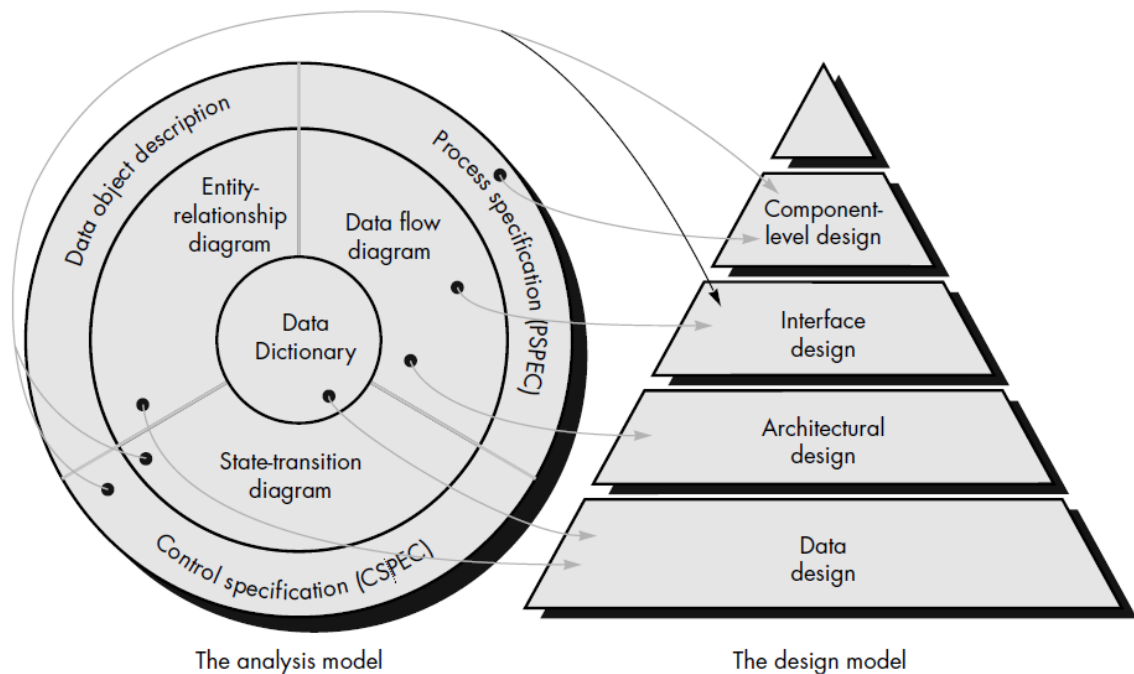
The flow of information during software design is illustrated in above figure. Software requirements, manifested by the data, functional, and behavioral models, feed the design task. Using one of a number of design methods, the design task produces a data design, an architectural design, an interface 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

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

*SUMMER-15 EXAMINATION*
<u>**Model Answer**</u>

Subject Code: 17513   Subject Name: Software Engineering

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.
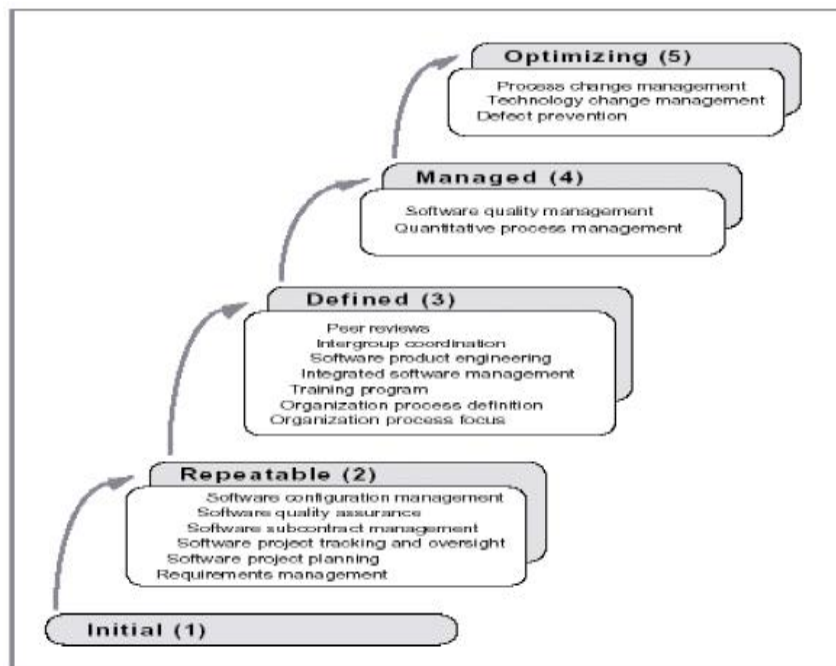
SUMMER-15 EXAMINATION
Model Answer

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

c) Explain CMMI model with neat diagram.
(2-Marks for CMMI definition, 2-Marks for diagram, 4-Marks for explanation)

Ans: **Definition**- Capability Maturity Model Integration (CMMI) is a process improvement approach that helps organizations improve their performance.

Level 1: Initial. The software process is characterized as ad hoc and occasionally even chaotic. Few processes are defined, and success depends on individual effort.

Level 2: Repeatable. Basic project management processes are established to track cost, schedule, and functionality. The necessary process discipline is in place to repeat earlier successes on projects with similar applications.

Level 3: Defined. The software process for both management and engineering activities is documented, standardized, and integrated into an organization wide software process. All projects use a documented and approved version of the organization's process for developing and supporting software. This level includes all characteristics defined for level

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER-15 EXAMINATION
<u>**Model Answer**</u>

Subject Code: 17513   Subject Name: Software Engineering

Level 4: Managed. Detailed measures of the software process and product quality are collected. Both the software process and products are quantitatively understood and controlled using detailed measures. This level includes all characteristics defined for level

Level 5: Optimizing. Continuous process improvement is enabled by quantitative feedback from the process and from testing innovative ideas and technologies. This level includes all characteristics defined for level 4.

6. Answer **any four** of the following: **Marks 16**

a) Compare PSP and TSP.

(Any 4 Points – 1 Mark Each)

Ans:

PSP	TSP
PSP(Personal Software Process)provide a standard personal process structure for software developers	**TSP** (Team Software Process) is a guideline for software product development teams.
PSP consists of a set of methods, tables, scripts etc. that serve as a guideline for software developers to plan, measure and manage their work, including how to define their processes and measure quality and productivity.	TSP focuses on helping development teams to improve their quality and productivity to better meet goals of cost and progress.
The PSP was created by Watts Humphrey to apply the underlying principles of the Software Engineering Institute's (SEI) Capability Maturity Model (CMM) to the software development practices of a single developer.	TSP is intended for use in academic settings, that focuses on the process of building a software production team, establishing team goals, distributing team roles, and other teamwork-related activities.
PSP skills are used in a TSP team environment.	TSP teams consist of PSP-trained developers who volunteer for areas

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER-15 EXAMINATION
Model Answer

Subject Code: 17513   Subject Name: Software Engineering

	of project responsibility, so the project is managed by the team itself.
The PSP is a personal process that can be adapted to suit the needs of the individual developer.	TSP uses team based planning sessions
The Personal Software Process (PSP) is an SEI technology that brings discipline to the practices of individual software engineers, dramatically improving product quality, increasing cost and schedule predictability, and reducing development cycle time for software.	The Team Software Process (TSP) is a complementary SEI technology that enables teams to develop software-intensive products more effectively. TSP shows a team of engineers how to produce quality products for planned costs and on aggressive schedules.

b) List seven task of requirement engineering.

(All 7 Tasks 4 Marks)

Ans: **1. Inception**
2. Elicitation
3. Elaboration
4. Negotiation
5. Specification
6. Validation
7. Requirements management

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER-15 EXAMINATION
<u>**Model Answer**</u>

Subject Code: 17513   Subject Name: Software Engineering

c) Differentiate between alpha and beta testing.

(Any 4 Points Each – 1 Mark, any other relevant points should be given Marks)

Ans:

ALPHA TESTING	BETA TESTING
Alpha Testing Conducted at Developer Site by End user.	Beta Testing is conducted at User site by End user.
Alpha Testing is Conducted in Control Environment as Developer is present	Beta Testing is conducted in Un-control Environment as Developer is Absent
Less Chances of Finding an error as Developer usually guides user.	More Chances of Finding an error as Developer can use system in any way
It is kind of mock up testing	The system is tested as Real application
Error/Problem may be solved in quick time if possible.	The user has to send difficulties to the developer who then corrects it.
Short process.	Lengthy Process

d) Compare white box and black box testing.

(Any 4 Points Each – 1 Mark)

Ans:

Criteria	Black Box Testing	White Box Testing
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 To	Mainly applicable to higher levels of testing: Acceptance Testing	Mainly applicable to lower levels of testing: Unit Testing Integration

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER-15 EXAMINATION
Model Answer

Subject Code: 17513   Subject Name: Software Engineering

	System Testing	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 RMMM strategy in detail.

(RMMM Strategy Description -4 Marks)

Ans: 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.

In many cases, the problems that occur during a project can be traced to more than one risk. Another job of risk monitoring is to attempt to allocate origin (what risk(s) caused which problems throughout the project).

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.

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER-15 EXAMINATION
Model Answer

Subject Code: 17513   Subject Name: Software Engineering

• 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.
• 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.
In addition to monitoring these factors, the project manager should monitor the effectiveness of risk mitigation steps. RMMM steps incur additional project cost. Part of risk management, therefore, is to evaluate when the benefits accrued by the RMMM steps are outweighed by the costs associated with implementing them. In essence, the project planner performs a classic cost/benefit analysis.

f) Differentiate between PERT and CPM.

(Any 4 Points Each – 1 Mark)

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

SUMMER-15 EXAMINATION
Model Answer

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

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 projects where time and cost estimates can be accurately calculated are uncertain.	CPM is best suited for routine and those projects where the project is performed for the first time and the estimate of duration are uncertain.
PERT uses event oriented Network.	CPM uses activity oriented network.