Subject – Software Engineering (SEN– 22413)

(IMP Questions with Answers)

SY Diploma- Semester 4

I Scheme – MSBTE

Branches: CO/CM/CW/IF

Chapter 01 – Software Development Process

(Weightage – 12 Marks)

2 Marks Questions

1. Enlist and explain software characteristics (any two).

OR

State two characteristics of Software.

Answer:

Software characteristics:

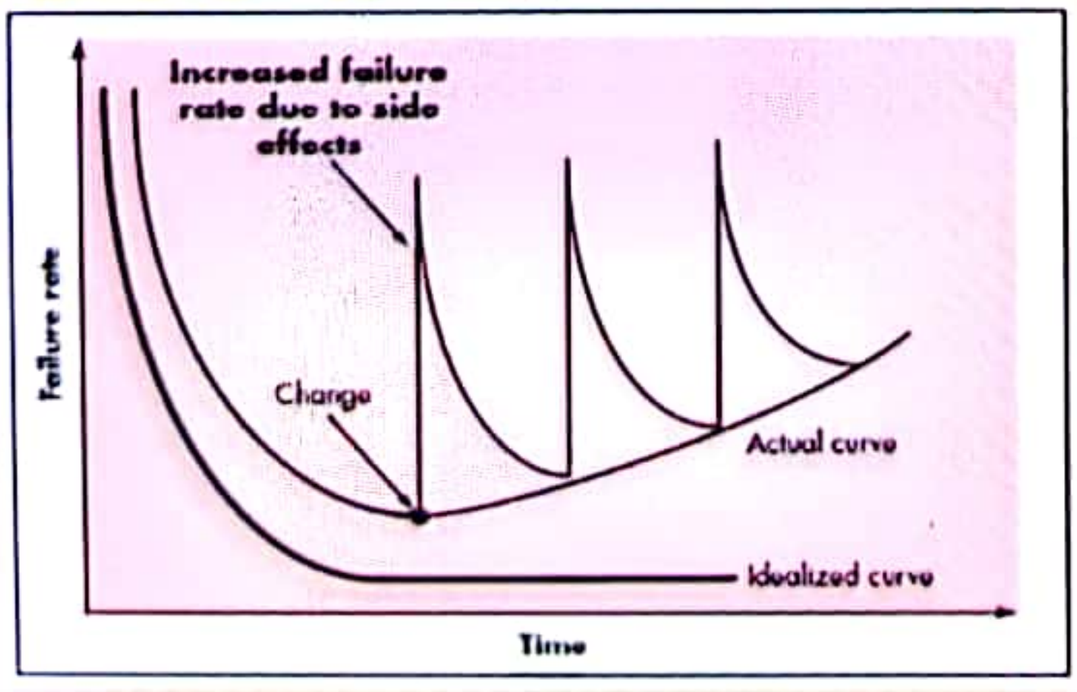
1) Software is developed or engineered; it is not manufactured in the classical sense.
2) Software doesn't "wear out."
3) Although the industry is moving toward component-based construction, most software continues to be custom built.

2. Define software. Draw the failure curve for software.

Answer:

Definition of Software Software is:

1) Instructions (computer programs) that when executed provide desired features, function, and performance;
2) Data structures that enable the programs to adequately manipulate information, and
3) Descriptive information (documents) in both hard copy and virtual forms that describes the operation and use of the programs.



Scanned with OKEN Scanner

3. Define software engineering.

Answer:

Software engineering is the establishment and use of sound engineering principles in order to obtain economically software that is reliable and works efficiently on real machines

---

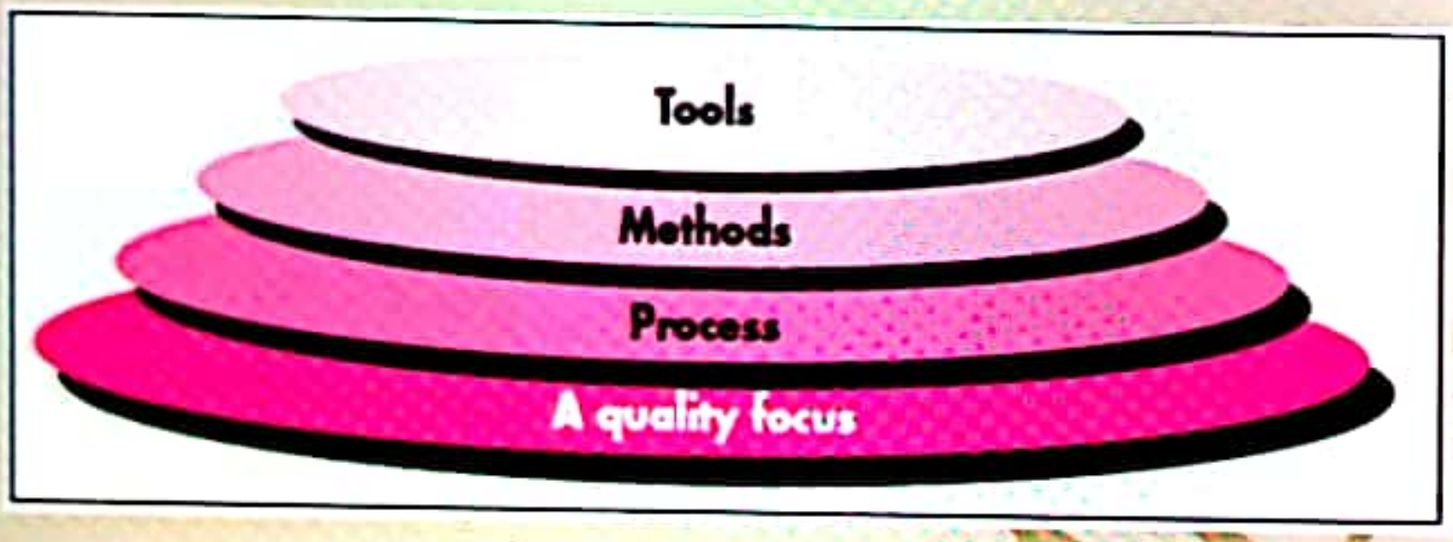
4 Marks Questions

---

4. Explain Software Engineering as layered technology approach.

Answer:

Software engineering is a layered technology. The layers of software engineering as shown in the above diagram are:-



1) **A Quality Focus:** Any engineering approach (including software engineering) must rest on an organizational commitment to quality. Total quality management, six sigma and similar philosophies foster a continuous process improvement culture, and it is this culture that ultimately leads to the development of increasingly more effective approaches to software engineering. The bedrock that supports software engineering is a quality focus.
2) **Process Layer:** 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 that must be established for effective delivery of software engineering technology. The software process forms the basis for management control of software projects and establishes the context in which technical methods are applied, works products (models, documents, data, reports, forms etc.) are produced, milestones are established, quantity is ensured and change is properly managed.
3) **Methods:** Software Engineering methods provide the technical –how to building software. Methods encompass a broad array of tasks that include communication, requirements analysis, design modeling, program construction, testing and support.
4) **Tools:** 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

Scanned with OKEN Scanner

of software development, called computer-aided software engineering is established.

5. State and explain with examples four categories of software.

OR

List any four types of software.

Answer:

Types / Categories of Software

(i) System Software:

1) System software is a collection of programs written to service other programs.
2) Few examples of system software are compilers, editors, and file management utilities, process complex, but determinate, information structures.
3) Other systems applications are operating system components, drivers, and telecommunications.
   Example: DOS, WINDOWS

(ii) Real-time Software:

1) Software that monitors or analyses or controls real-world events as they occur is called real time.
2) Elements of real-time software include a data gathering component that collects and formats information from an external environment, an analysis component that transforms information as required by the application.
3) A control/output component that responds to the external environment and a monitoring component that coordinates all other components so that real-time response can be maintained.
   Example: airline reservation system, railway reservation system

(iii) Business Software:

1) Business information processing is the largest single software application area. Discrete "systems".
2) For example: payroll, accounts receivable/payable, inventory have evolved into management information system (MIS) software that accesses one or more large databases containing business information.
3) Applications in this area restructure existing data in a way that facilitates business operations or management decision making.
4) In addition to conventional data processing application, business software applications also encompass interactive computing.
   Example: Tally

(iv) Engineering and Scientific Software:

1) Engineering and scientific software have been characterized by "number crunching" algorithms.
2) Applications range from astronomy to volcanology, from automotive stress analysis to space shuttle orbital dynamics, and from molecular biology to automated manufacturing.

Scanned with OKEN Scanner

3) However, modern applications within the engineering/scientific area are moving away from conventional numerical algorithms. 4. Computer-aided design, system simulation, and other interactive applications have begun to take on real-time and even system software characteristics.
Example: CAD / CAM software

6. Explain waterfall process model. State its advantages and disadvantages.

Answer:

1) 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.
2) 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.
3) In other words, one stage should be completed before the other begins. Hence, when all the requirements are elicited by the customer, analyzed for completeness and consistency, documented as per requirements, the development and design activities commence.
4) 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.
5) 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 limitations is it does not reflect the way code is really developed. Problem is well understood but software is developed with great deal of iteration.

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.

Scanned with OKEN Scanner

7. Prescriptive process model and agile process model.

Answer:

Sr. No.	Prescriptive process model	Agile process mode
1	Prescriptive process models stress detailed definition, identification, and application of process activates and tasks	Agile process models emphasize project "agility" and follow a set of principles that lead to a more informal approach to software process.
2	A prescriptive model also describes how each of these elements are related to one another	Agile methods note that not only do the software requirements change, but so do team members, the technology being used.
3	It is Process oriented.	It is people oriented.
4	It follows Life cycle model (waterfall, spiral) development model.	It follows Iterative and Incremental development model.
5	Documentation required is to be comprehensive and constant.	Documentation required is to be minimal and evolving.

8. Explain Process framework with a suitable diagram.

Answer:

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.

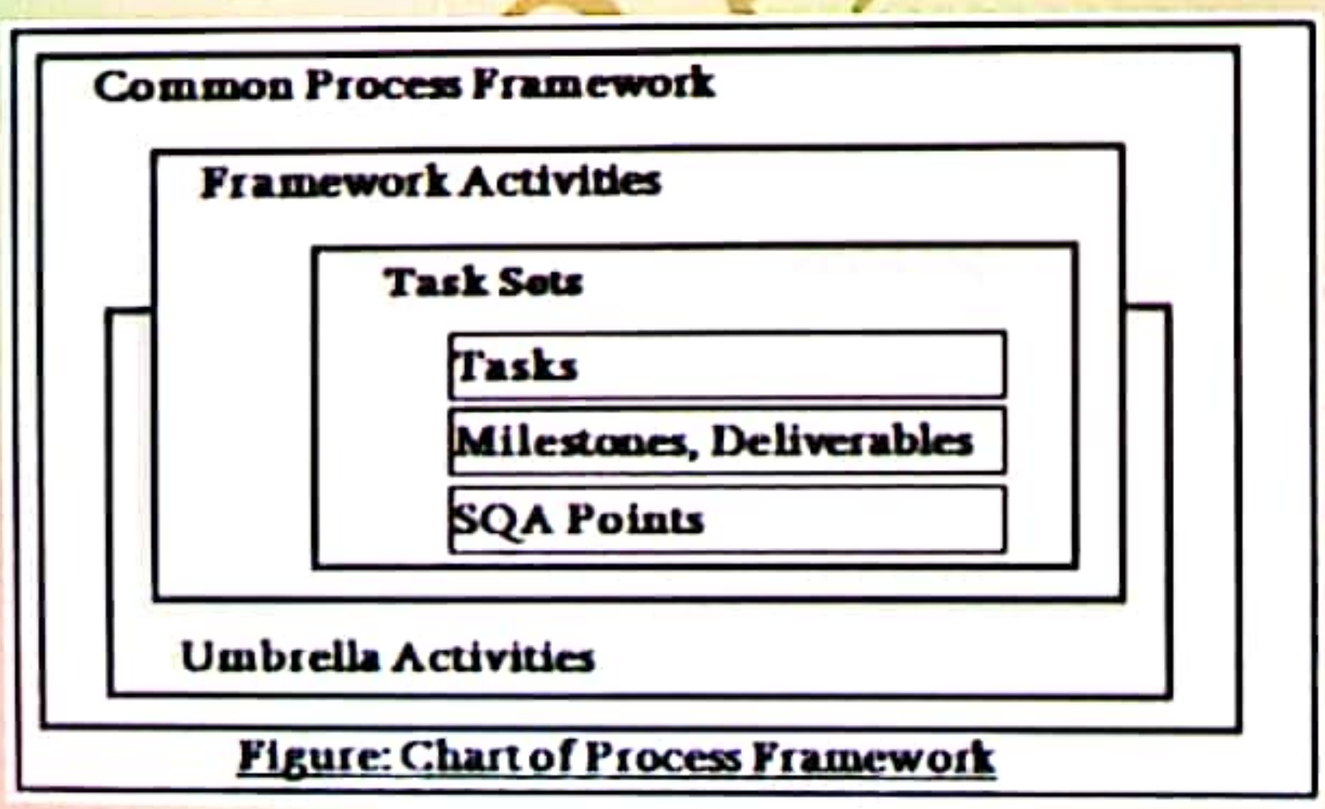


Figure: Chart of Process Framework

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

Scanned with OKEN Scanner

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 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.

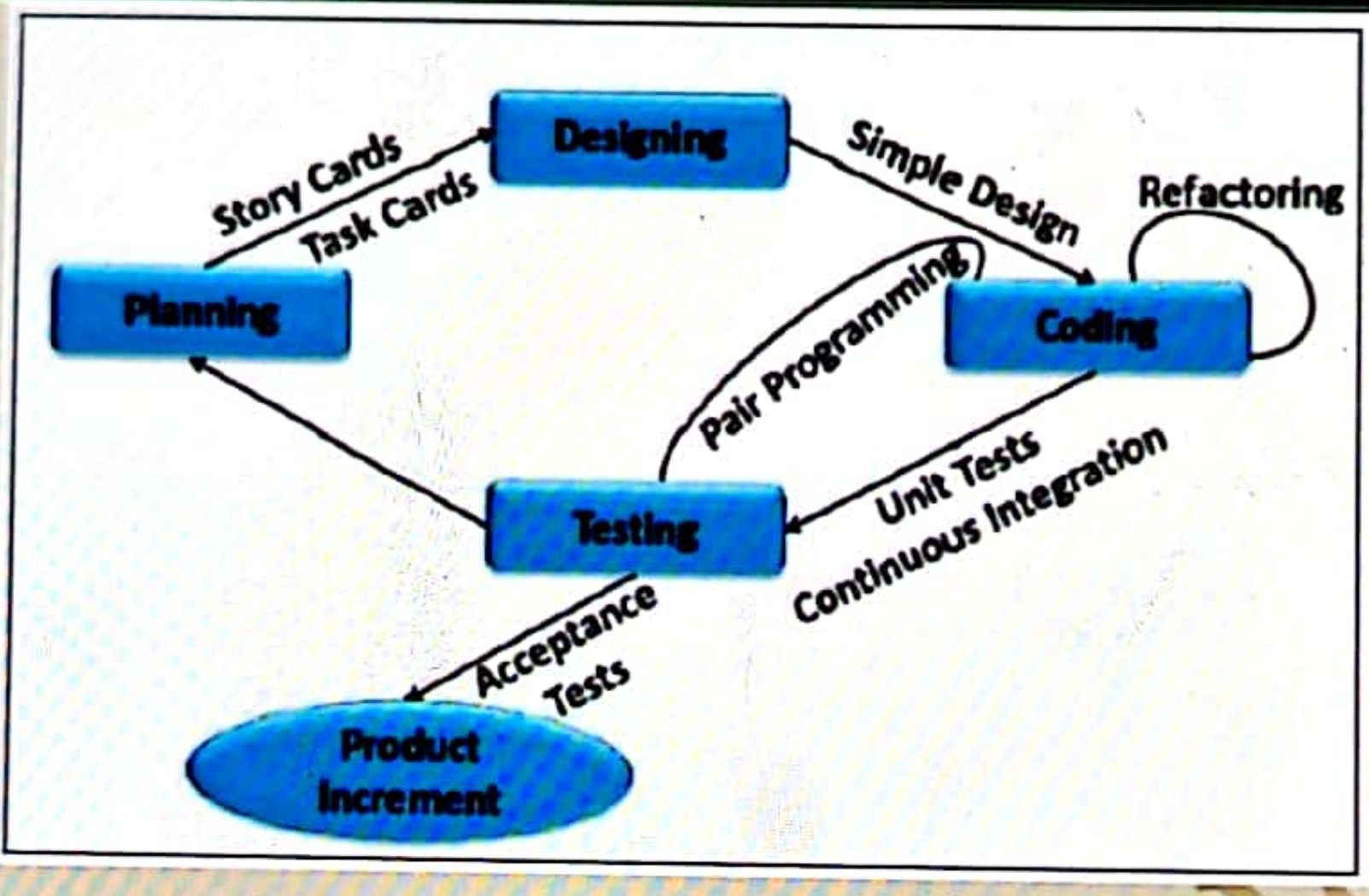
9. Describe Extreme programming with proper diagram.

Answer:

1) Extreme programming is a lightweight, efficient, low-risk, flexible, predictable, scientific, and fun way to develop a software.
2) eXtreme Programming (XP) was conceived and developed to address the specific needs of software development by small teams in the face of vague and changing requirements. Extreme Programming is one of the Agile software development methodologies. It provides values and principles to guide the team behavior. The team is expected to selforganize.
3) Extreme Programming provides specific core practices where-
   - Each practice is simple and self-complete.
   - Combination of practices produces more complex and emergent behavior.
4) Extreme Programming is based on the following values-
   - Communication
   - Simplicity
   - Feedback
   - Courage
   - Respect
5) Extreme Programming involves-
   - Writing unit tests before programming and keeping all of the tests running at all times. The unit tests are automated and eliminates defects early, thus reducing the costs.
   - Starting with a simple design just enough to code the features at hand and redesigning when required.
   - Programming in pairs (called pair programming), with two programmers at one screen, taking turns to use the keyboard. While one of them is at the keyboard, the other constantly reviews and provides inputs.
   - Integrating and testing the whole system several times a day.
   - Putting a minimal working system into the production quickly and upgrading it whenever required.
   - Keeping the customer involved all the time and obtaining constant feedback.

   Iterating facilitates the accommodating changes as the software evolves with the changing requirements

Scanned with OKEN Scanner



Scanned with OKEN Scanner

Subject – Software Engineering (SEN– 22413)

(IMP Questions with Answers)

SY Diploma– Semester 4

I Scheme – MSBTE

Branches: CO/CM/CW/IF

Chapter 02 – Software Requirement Engineering

(Weightage – 14 Marks)

2 Marks Questions

1. Define software requirement specification.

Answer:

Concept: A software requirements specification (SRS) is a document that is created when a detailed description of all aspects of the software to be built that must be specified before the project is to commence. It is a primary document for development of software. It is written by Business Analysts who interact with client and gather the requirements to build the software

2. State need of software requirement specification (SRS).

Answer:

The need of SRS document is to provide:

1) A detailed overview of software product, its parameters and goals.
2) The description regarding the project's target audience and its user interface hardware and software requirements.
3) How client, team and audience see the product and its functionality

3. Enlist the characteristics of SRS.

Answer:

1) Correctness
2) Completeness
3) Consistency
4) Unambiguousness
5) Modifiability
6) Traceability
7) Testability
8) Understandable by stakeholder

4. List any four selection criteria for Software Process Model.

Answer:

1) Requirements Characteristics

Scanned with OKEN Scanner

2) Development team
3) User involvement in the project
4) Project type and associated risk

---

4 Marks Questions

---

5. Describe any four principles of communication for software engineering.

Answer:

(i) Principle – 1: Listen
1) Try to focus on the speaker's words, rather than formulating your response to those words.
2) Ask for clarification if something is unclear, but avoid constant interruptions.
3) Never become contentious in your words or actions (e.g., rolling your eyes or shaking your head) as a person is talking.

(ii) Principle – 2: Prepare before you communicate
1) Spend the time to understand the problem before you meet with others. If necessary, perform some research to understand business domain.
2) If you have responsibility for conducting a meeting, prepare an agenda in advance of the meeting.

(iii) Principle – 3: someone should facilitate the activity
1) Every communication meeting should have a leader (a facilitator)
2) To keep the conversation moving in a productive direction, • To mediate any conflict that does occur, and
3) To ensure that other principles are followed.

(iv) Principle – 4: Face-to-face communication is best
1) It usually works better when some other representation of the relevant information is present.
2) For example, a participant may create a drawing /document that serve as a focus for discussion.

(v) Principle – 5: Strive for collaboration
1) Collaboration in terms of teamwork is required for the successful completion of the software.
2) The collective knowledge of the team members should be implemented in the development.

(vi) Principle – 6: Stay focused and modularize your discussion
1) As the development is the working of many team members, so the possibility of the discussion going from one topic to the other topic is quite possible.
2) As a good software developer it is required that the discussion remains focused on the specified area

Scanned with OKEN Scanner

6. List and explain any four principles of "Core Principles" of Software Engineering.

OR

Enlist core principles of software engineering practice.

OR

State and describe any four core principles.

Answer:

(i) The First Principle: **The Reason It All Exists**

1) A software system exists for one reason: to provide value to its users. All decisions should be made with this in mind.
2) Before specifying a system requirement, system functionality, before determining the hardware platforms, first determine, whether it adds value to the system.

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

1) All design should be as simple as possible, but no simpler. This facilitates having a more easily understood and easily maintained system. •
2) It doesn't mean that features should be discarded in the name of simplicity.
3) Simple also does not mean "quick and dirty." In fact, it often takes a lot of thought and work over multiple iterations to simplify.

(iii) The Third Principle: **Maintain the Vision**

1) A clear vision is essential to the success of a software project.
2) If you make compromise in the architectural vision of a software system, it will weaken and will eventually break even the well-designed systems.
3) Having a powerful architect who can hold the vision helps to ensure a very successful software project.

(iv) The Fourth Principle: **What You Produce, Others Will Consume**

1) Always specify, design, and implement by keeping in mind that someone else will have to understand what you are doing.
2) The audience for any product of software development is potentially large.
3) Design (make design), keeping the implementers (programmers) in mind. Code (program) with concern for those who will maintain and extend the system.
4) Someone may have to debug the code you write, and that makes them a user of your code.

(v) The Fifth Principle: Be Open to the Future

1) A system with a long lifetime has more value.
2) True "industrial-strength" software systems must last for longer. To do this successfully, these systems must be ready to adapt changes.
3) Always ask "what if," and prepare for all possible answers by creating systems that solve the general problem.

Scanned with OKEN Scanner

7. **Describe four principles of good planning.**

OR

List any four planning principles

Answer:

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 replanning after the delivery of each software increment based on feedback received from users.

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

8. **State and describe any four deployment principles.**

Answer:

Principle 1: Manage customer's expectations.

It always happens that customer wants more than he has stated 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.

---

6 Marks Questions

Scanned with OKEN Scanner

Actual project delivery includes all documentations, help files and guidance for handling the software by user.

---

6 Marks Questions

---

9. Enlist requirement Gathering and Analysis for web based project for registering candidates for contest.

Answer:

Requirement gathering includes suggestions and ideas for ways to best capture the different types of requirement (functional, system, technical, etc.) during the gathering process.

(i) Functional requirements

The functional requirements are the requirements that will enable solving the real world problem. The web based project must be able to register the candidates for contest.

(ii) Non-functional requirements

These requirements aim at providing support, security and facilitate user interaction segment of the website. • The project must enable the candidates to safely enter their passwords and other biometric information. • There must be no repetition in registration of candidates i.e the candidates must be registered only once.

(iii) Business requirements:

They are high-level requirements that are taken from the business case from the projects.

For eg:-

Qualifying criteria	Allowed/Disallowed
Indian Nationality Registration	Allowed
Age>18	Allowed
No criminal record	Allowed

(iv) Architectural and Design requirements:

These requirements are more detailed than business requirements. It determines the overall design required to implement the business requirement.

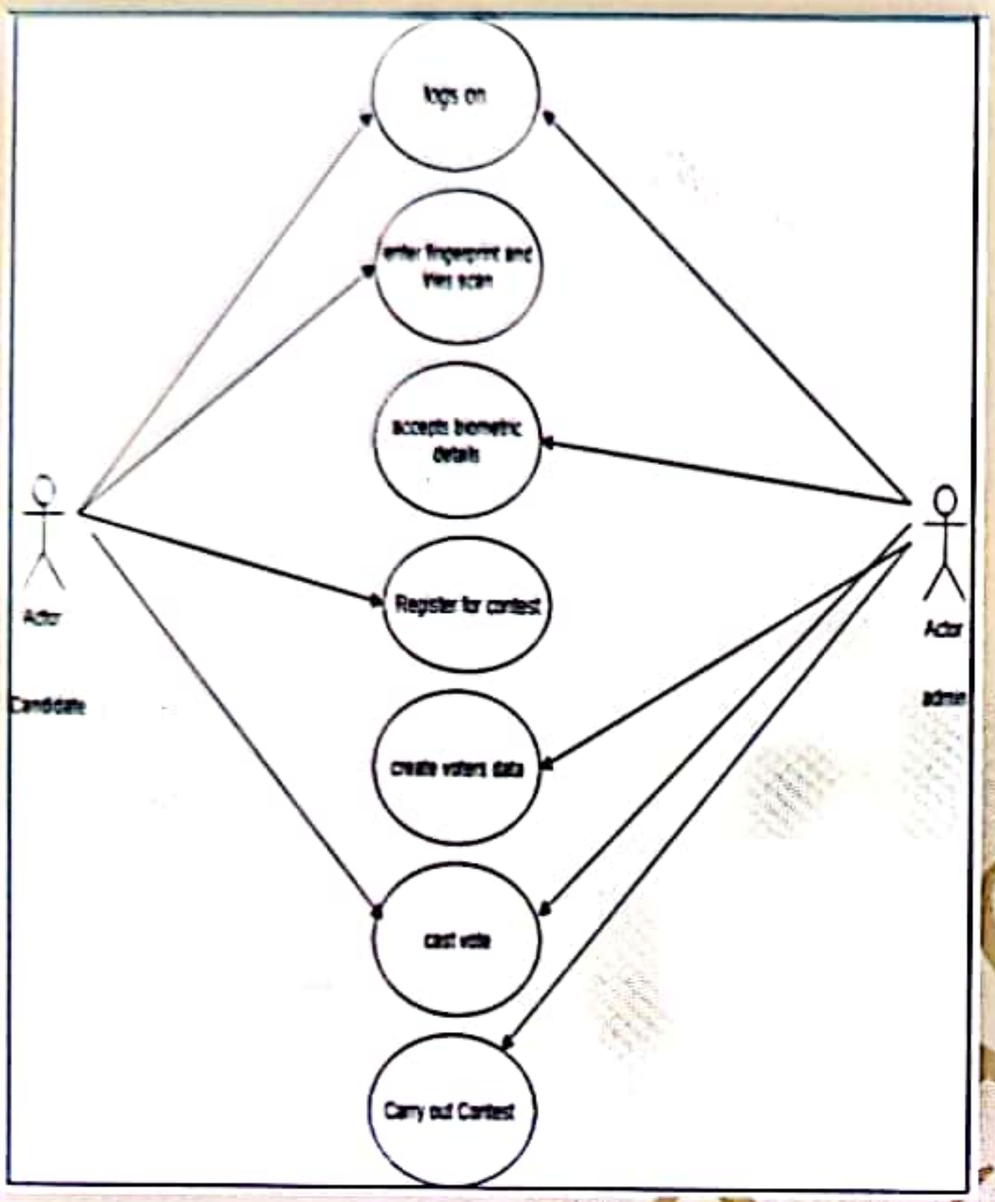
• The web based project must be supported by different operating systems , PC and mobile compatibility etc.

• The hardware must be integrated so as to accept the fingerprint details of a candidate and register him in the system.

• The database of the project must be updated.

(v) System and Integration requirements:

At the lowest level, we have system and integration requirements. It is detailed description of each and every requirement. It can be in form of user stories which is really describing everyday business language. The requirements are in abundant details so that developers can begin coding.



Documenting the requirement using traceability matrix A Traceability Matrix is a document that co-relates any two baseline documents that require a many-to-many relationship to check the completeness of the relationship. It is used to track the requirements and to check the current project requirements are met.

10. Explain six function of requirement engineering process.

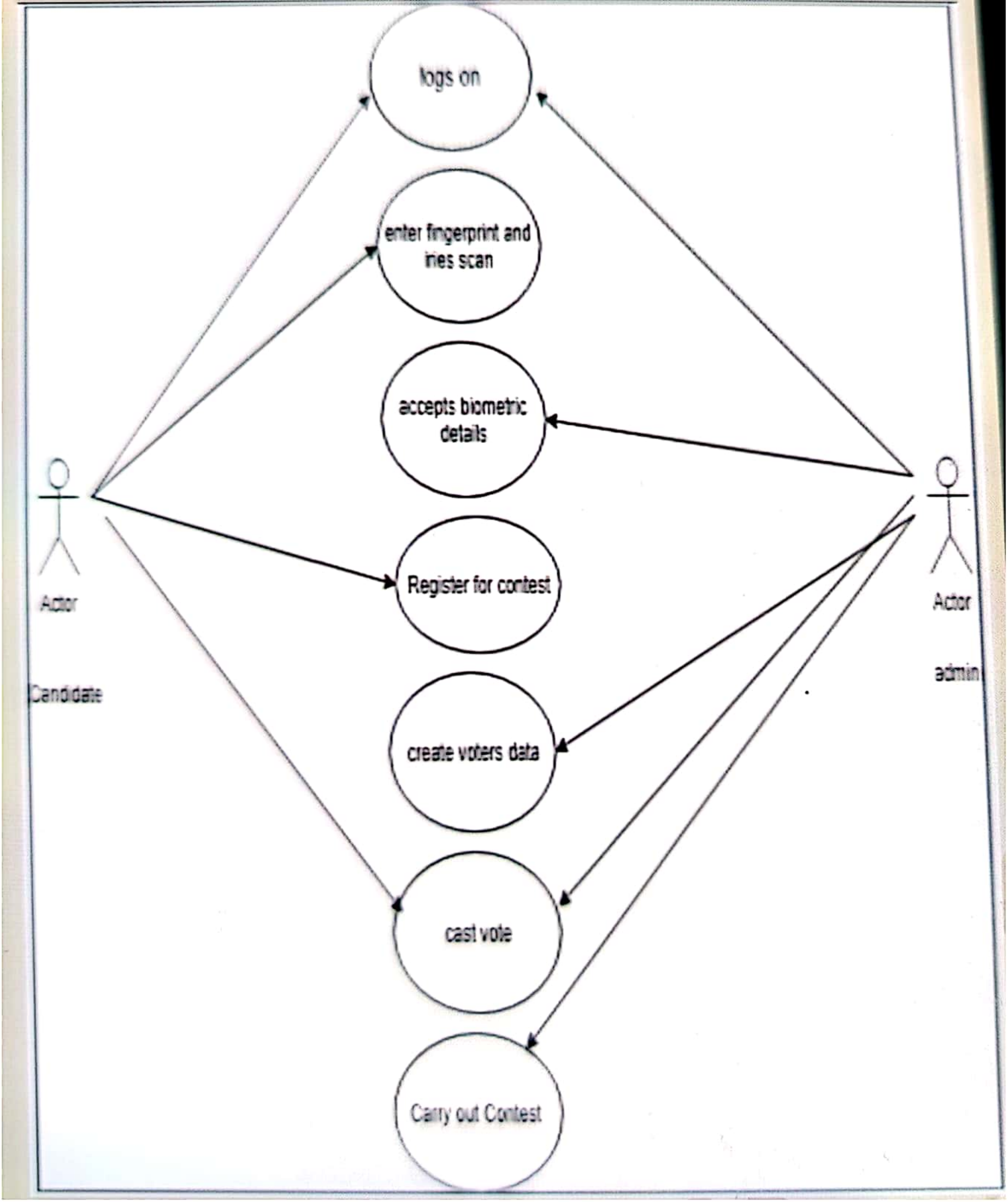
Answer:

Requirement Engineering:

1) The broad spectrum of tasks and techniques that lead to an understanding of requirements is called requirements engineering. It starts during the communication activity and continues into the modeling activity.
2) Requirements engineering provides the appropriate mechanism for understanding what the customer wants by analyzing need, assessing feasibility negotiating a reasonable solution, specifying the solution ambiguously, validating the specification, and managing the requirements as they are transformed into an operational system.
3) It encompasses seven distinct tasks: inception, elicitation, elaboration, negotiation, specification, validation, and management.

(i) Inception:

The question why you want to do this will be answered and analyses to identify business need, a potential new market with breadth and depth and services to be provided. The above points establish a basic understanding of the problem, the people who want a solution, the nature of the solution



Scanned with OKEN Scanner

The question why you want to do this will be answered and analyses to identify business need, a potential new market with breadth and depth and services to be provided. The above points establish a basic understanding of the problem, the people who want a solution, the nature of the solution that is desired to understand the scope of the project.

(ii) Elicitation:

This answers for what are things need to do by asking the customer, the users, and others what the objectives for the system or product are, what is to be accomplished, how the system or product fits into the needs of the business, and finally, how the system or product is to be used on a day-to-day basis

(iii) Elaboration:

The information obtained from the customer during inception and elicitation is expanded and refined during elaboration. This task focuses on developing a refined requirements model that identifies requirements for three domains, information, functional and behavioral domain. It • Describe how the end user (and other actors) will interact with the system. • Business domain entities that is visible to the end user. • The attributes of each analysis class are defined, and the services that are required by each class are identified. • The relationships and collaboration between classes are identified, and a variety of supplementary diagrams are produced.

(iv) Negotiation:

It answers for is it actually required? Through which Customers, users, and other stakeholders are asked to rank requirements and prioritized the same. Using an iterative approach that prioritizes requirements, assesses their cost and risk, and addresses internal conflicts, requirements are eliminated, combined, and/or modified so that each party achieves some measure of satisfaction.

(v) Specification:

A specification can be a written document, a set of graphical models, a formal mathematical model, a collection of usage scenarios, a prototype, or any combination of these to present gathered requirements. The formality and format of a specification varies with the size and the complexity of the software to be built.

(vi) Validation:

As a part of this task documented software requirement specification will be examining by conducting technical reviews in order to examine errors in content or interpretation, areas where clarification may be required, missing information, inconsistencies (a major problem when large products or systems are engineered), conflicting requirements, or unrealistic (unachievable) requirements.

(vii) Requirements management:

Requirements management is a set of activities that help the project team identify, control, and track requirements and changes to requirements at any time as the project proceeds.

Scanned with OKEN Scanner

11. Identify and enlist requirement for given modules of employee management software.

Answer:

i. Employee detail
ii. Employee salary
iii. Employee performance

This is with perspective of employee management software. Requirements for following modules will be as

i. Employee details

a. Getting information about the customer

b. Updation of employee details (department, change of address, emp_code etc)

c. Assignment of tasks , duties and responsibilities.

d. Recording of employee attendance.

ii. Employee salary

a. Salary calculation

b. Allowances, special bonus calculation and approval

c. Tax statement/certificate d. Apply loan/approvals

iii. Performance

a. Recording annual performance

b. Details about parameters for performance appraisal

c. Analysis performance and determining hike in payment

Scanned with OKEN Scanner

Subject – Software Engineering (SEN– 22413)

(IMP Questions with Answers)

SY Diploma– Semester 4

I Scheme – MSBTE

Branches: CO/CM/CW/IF

Chapter 03 – Software Modelling and Design

(Weightage – 14 Marks)

---

2 Marks Questions

---

1. Describe following design concepts i) Abstraction ii) Information hiding.

Answer:

Abstraction - Abstraction is hiding the internal implementation and highlight the set of services. It is achieved by using the abstract class and interfaces and further implementing the same.

Information Hiding - It is the principle of segregation of the design decisions in a computer program that are most likely to change, thus protecting other parts of the program from extensive modification if the design decision is changed.

---

4 Marks Questions

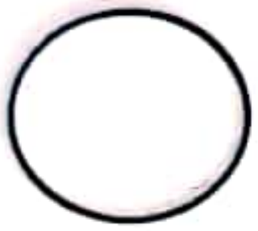
---

2. Explain the notations used for preparing a Data Flow diagram.

Answer:

Circle: A circle (bubble) shows a process that transforms data inputs into data outputs.
Data Flow: A curved line shows the flow of data into or out of a process or data store.
Data Store: A set of parallel lines shows a place for the collection of data items. A data store indicates that the data is stored which can be used at a later stage or by the other processes in a different order. The data store can have an element or group of elements.
Source or Sink: Source or Sink is an external entity and acts as a source of system inputs or sink of system outputs.

Scanned with OKEN Scanner

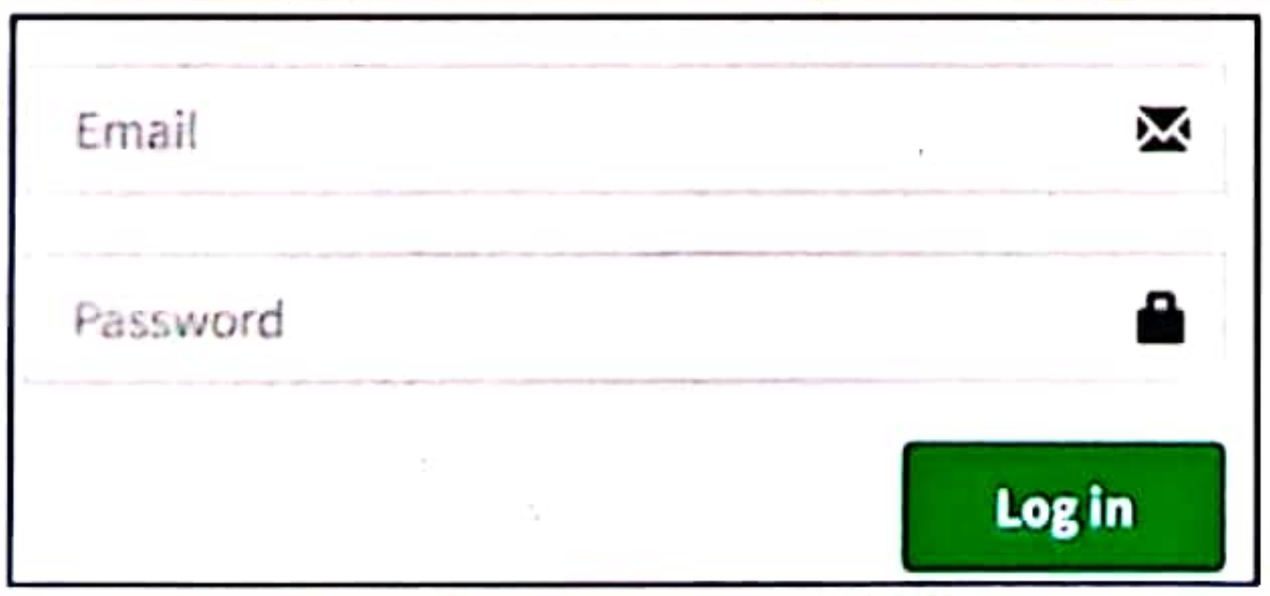
Symbol	Name	Function
	Data flow	Used to Connect Processes to each , other , to sources or Sinks; te arrow head indicates direction of data flow.
	Process	Perfroms Some transformation of input data to yield output data.
	Source of Sink (External Entity)	A Source of System inputs or Sink of System outputs.
	Data Store	A repository of data; the arrow heads indicate net inputs and net outputs to store.

3. Explain with example Decision table.

Answer:

1) Decision table is a software testing technique used to test system behaviour for different input combinations.
2) This is a systematic approach where the different input combinations and their corresponding system behaviour (Output) are captured in a tabular form.
3) That is why it is also called as a Cause-Effect table where Cause and effects are captured for better test coverage

Example 1: Decision Base Table for Login Screen



4) **The condition is simple if the user provides correct username and password the user will be redirected to the homepage. If any of the input is wrong, an error message will be displayed.**

Decision Table:

Conditions	Rule 1	Rule2	Rule3	Rule 4
Username(T/F)	F	T	F	T
Password(T/F)	F	F	T	T
Output(E/H)	E	E	E	H

Legend:

T – Correct username/password

F – Wrong username/password

E – Error message is displayed

H – Home screen is displayed

Interpretation:

Case 1 – Username and password both were wrong. The user is shown an error message.

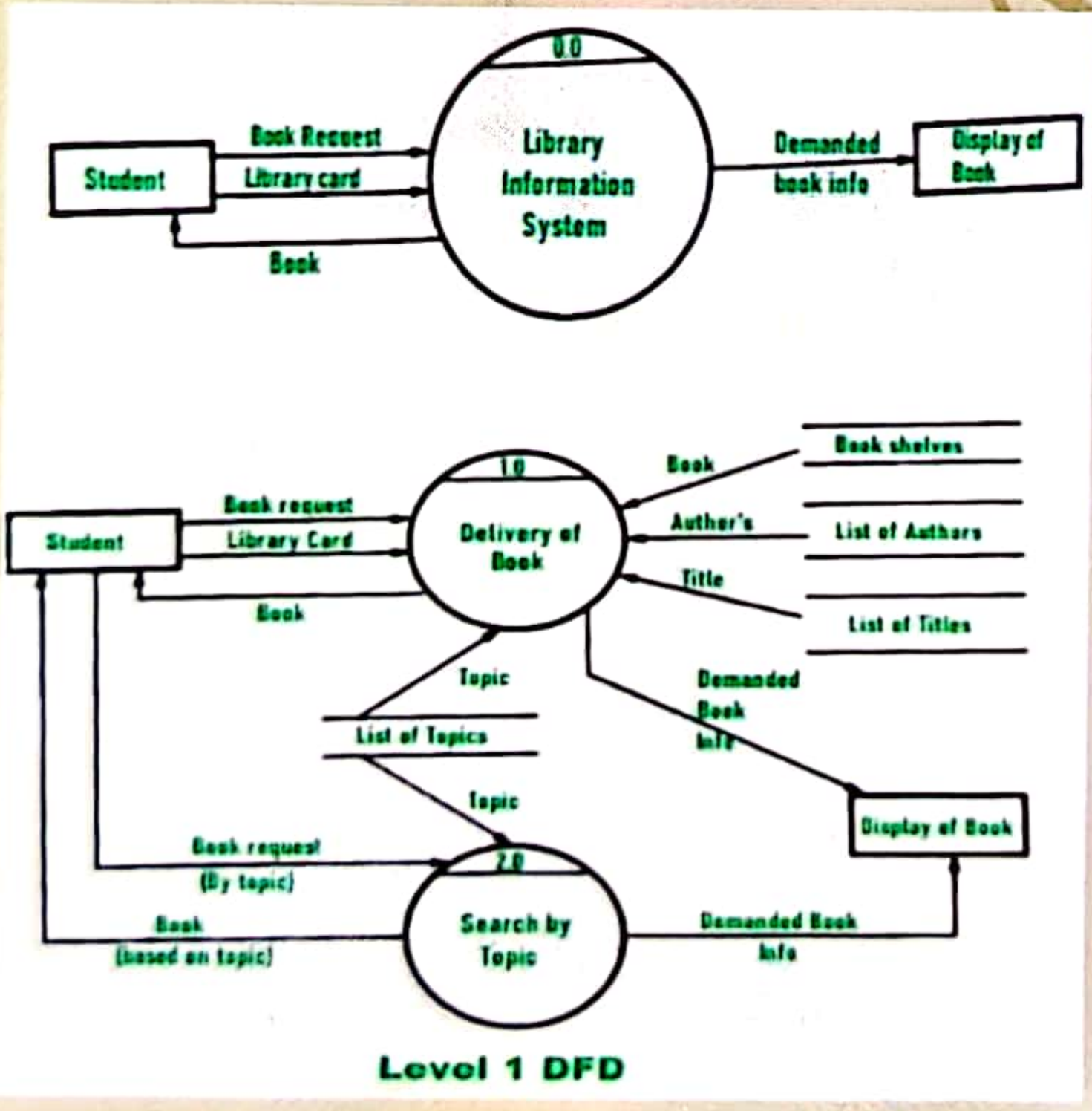
Case 2 – Username was correct, but the password was wrong. The user is shown an error message.

Case 3 – Username was wrong, but the password was correct. The user is shown an error message.

Case 4 – Username and password both were correct, and the user navigated to homepage.

4. Draw DFD 0 and DFD 1 diagram for Library Management System.

Answer:

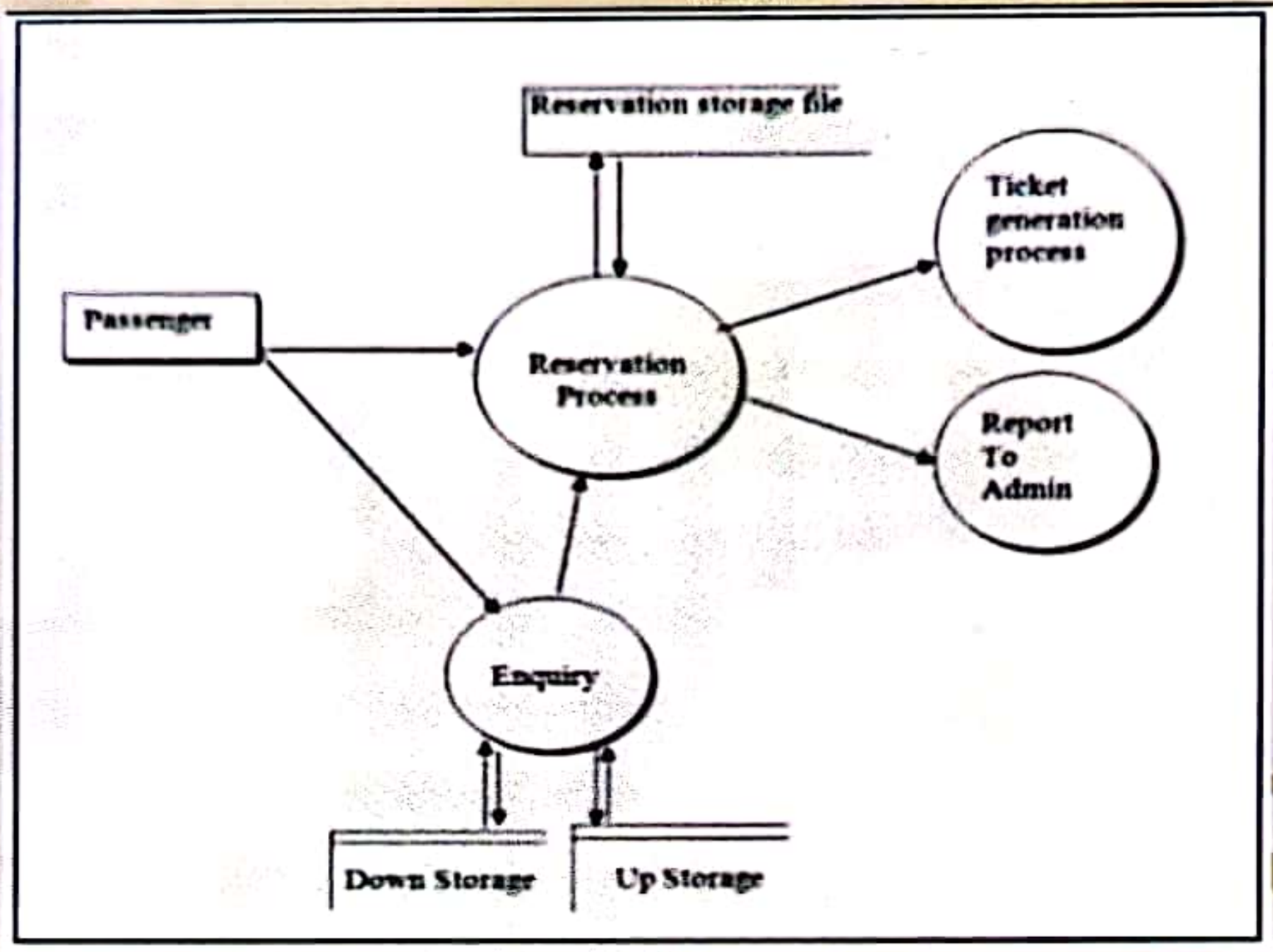


Level 1 DFD

Scanned with OKEN Scanner

5. Draw and explain Level 1 DFD for railway reservation system.

Answer:

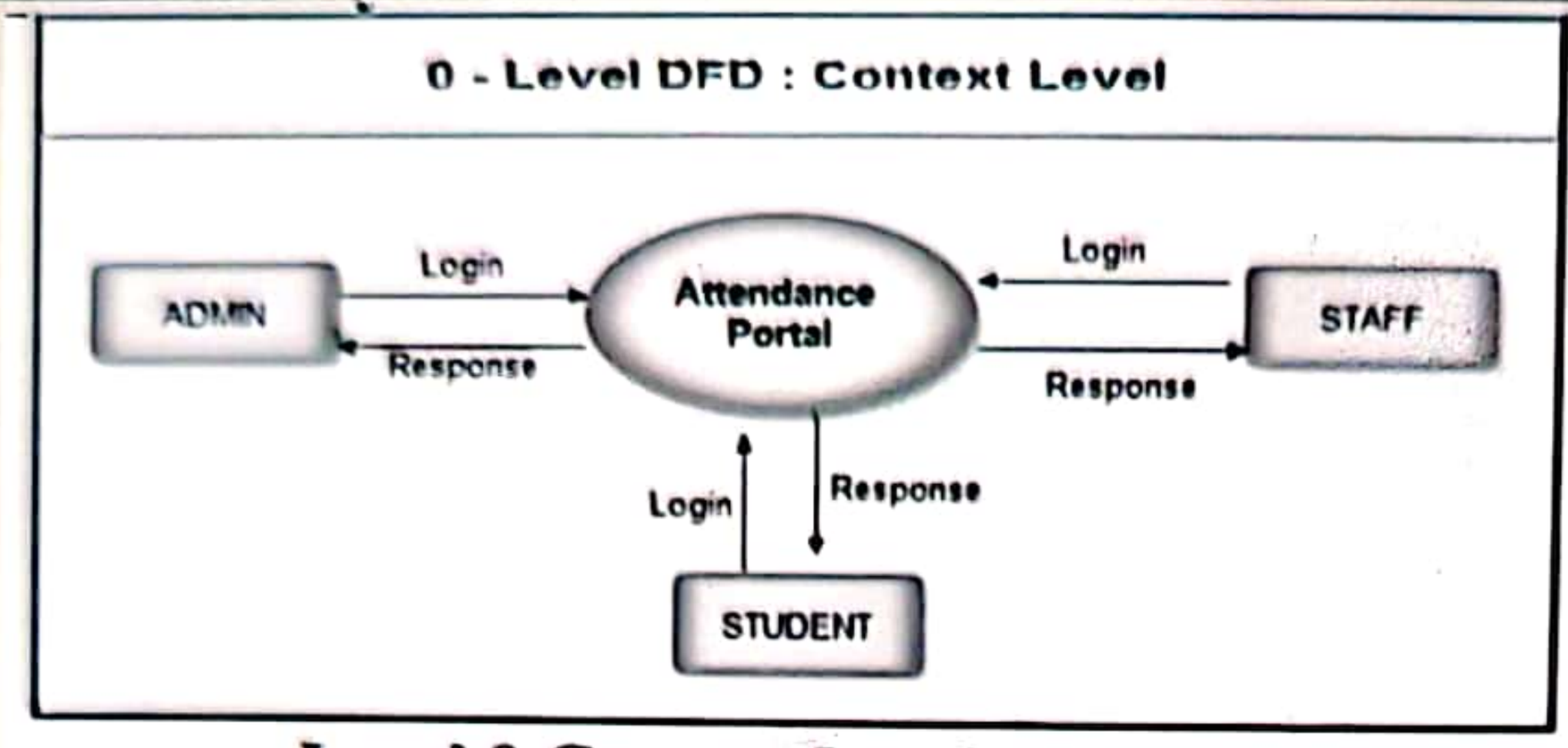


The passenger can initiate either Reservation process or Enquiry process; If a user opts for Reservation process then the system shall proceed with ticket generation process and same needs to be notified to the Admin. If user opts for enquiry module then appropriate request shall be entertain and result to be displayed to the user.

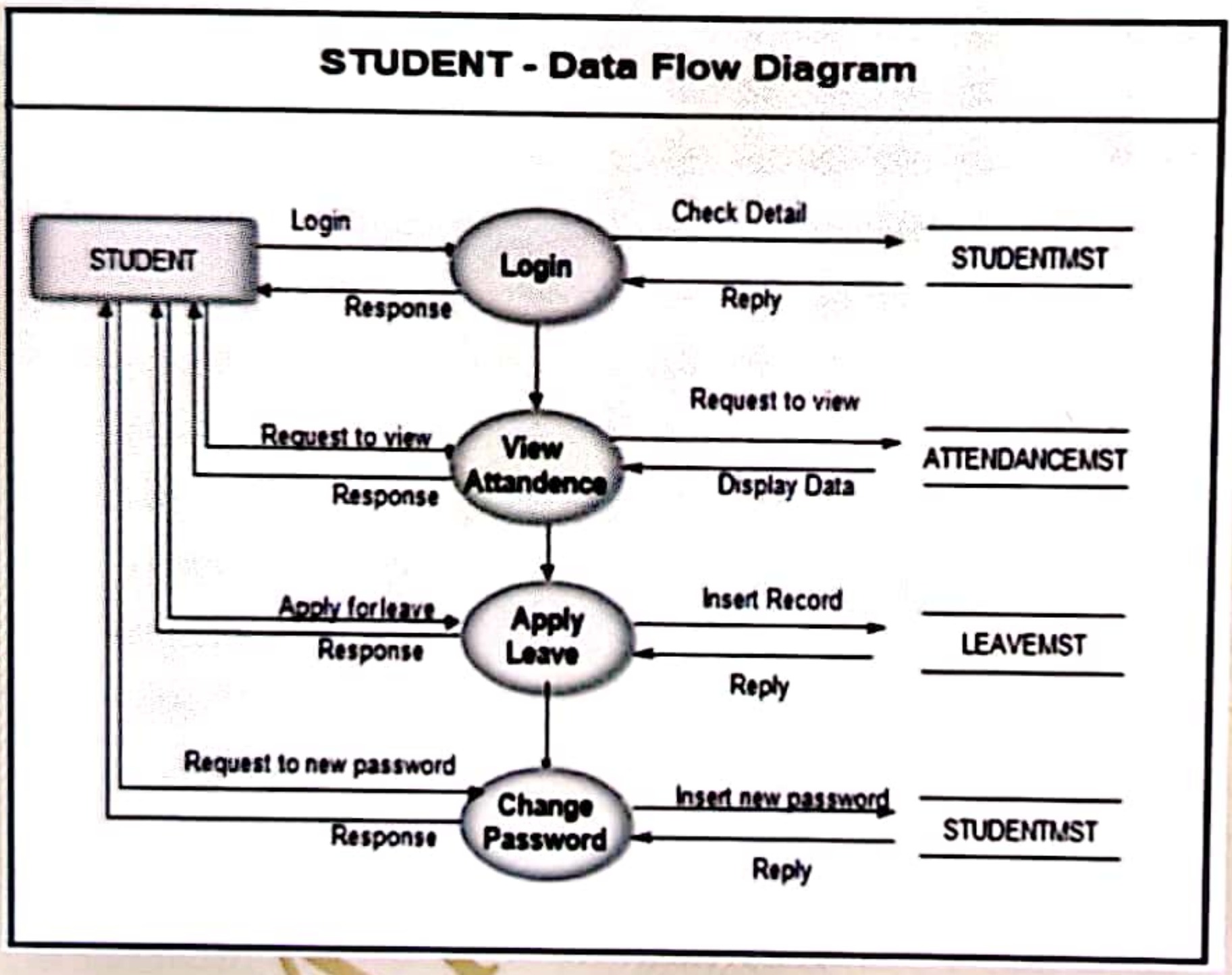
6. Draw proper labelled "LEVEL 1 Data Flow Diagram" (DFD) for student attendance system.

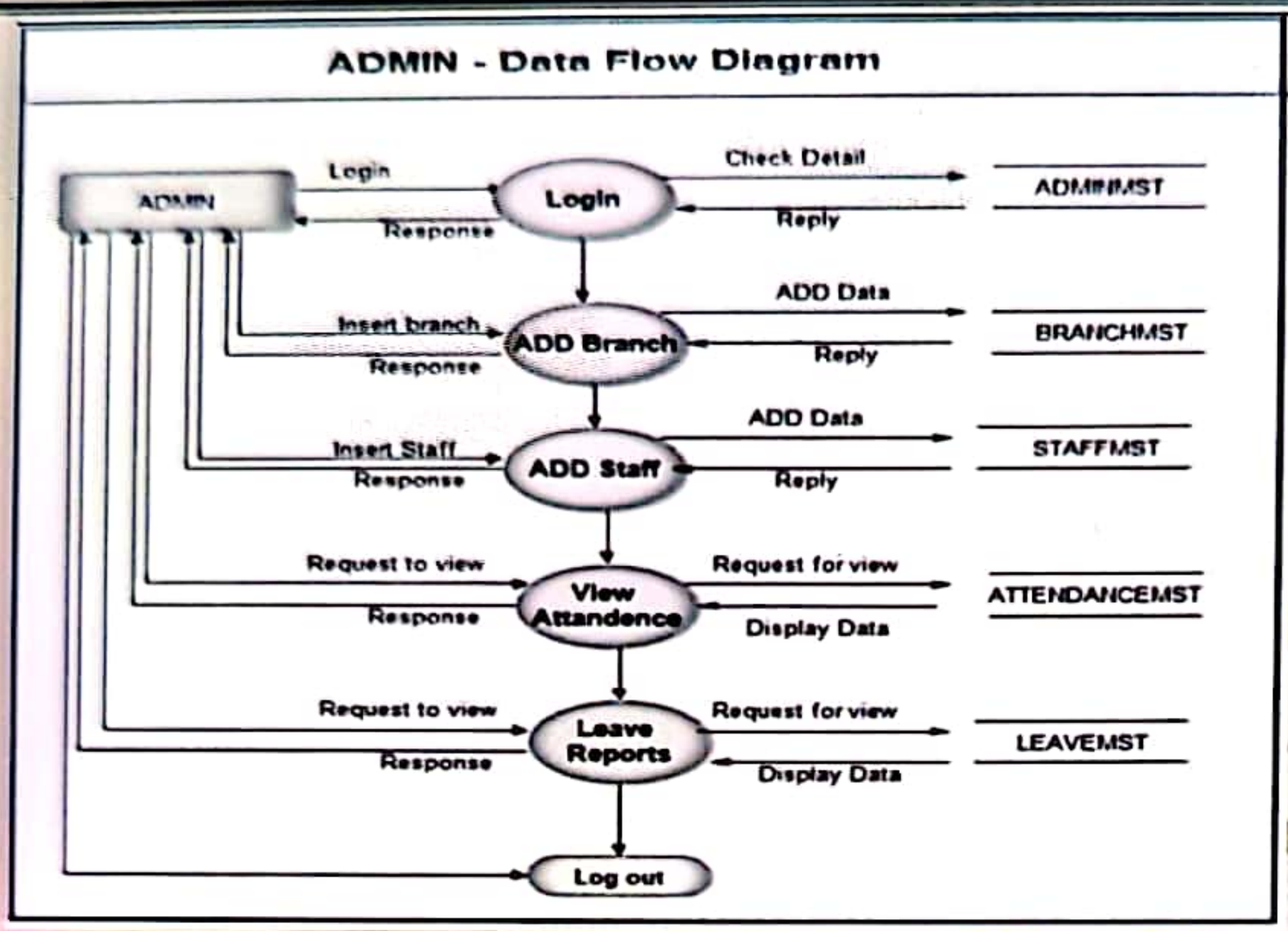
Answer:

Scanned with OKEN Scanner



Level 0 Context Level





7. Explain Test Documentation with the help of following terms i) Test Case ii) Test Data iii) Test Plan

Answer:

Test Documentation

Test documentation is documentation of artifacts created before or during the testing of software. It helps the testing team to estimate testing effort needed, test coverage, resource tracking, execution progress, etc. It is a complete suite of documents that allows you to describe and document test planning, test design, test execution, test results that are drawn from the testing activity

1) **Test Case:** It is a detailed document that describes step by step procedure to test an application. It consists of the complete navigation steps and inputs and all the scenarios that need to be tested for the application. We will write the test case to maintain the consistency, or every tester will follow the same approach for organizing the test document. It is a document that is prepared by the managers or test lead.
2) **Test Data:** Data created or selected to satisfy the execution preconditions and inputs to execute one or more test cases
3) **Test Plan:** It consists of all information about the testing activities. The test plan consists of multiple components such as Objectives, Scope, Approach, Test Environments, Test methodology, Template, Role & Responsibility, Effort estimation, Entry and Exit criteria, Schedule, Tools, Defect tracking, Test Deliverable, Assumption, Risk, and Mitigation Plan or Contingency Plan.

Scanned with OKEN Scanner

6 Marks Questions

8. Define data objects, attributes, relationship, and cardinality, with example of each.

Answer:

Data Object: A data object is an entity/object in the real world with an independent existence that can be differentiated from other objects.

Example: An entity might be

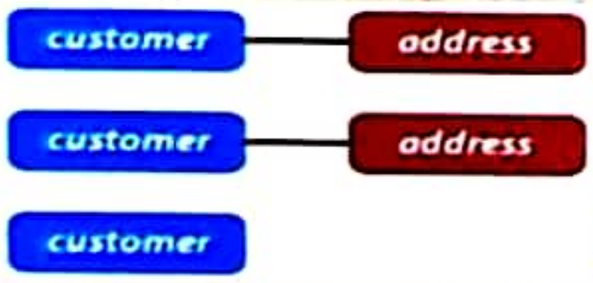
1) An object with physical existence (e.g., a lecturer, a student, a car)
2) An object with conceptual existence (e.g., a course, a job, a position)

Attributes: Each data object/ entity is described by a set of attributes (e.g., Employee = (Name, Address, Birthdate (Age), Salary).

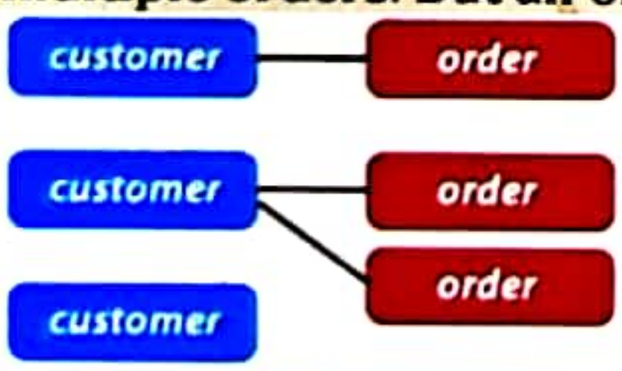
Each attribute has a name, and is associated with an entity and a domain of legal values. Example: Employee = (Name, Address, Birthdate (Age), Salary).

Relationship: A relationship identifies names and defines an association between two entity types.

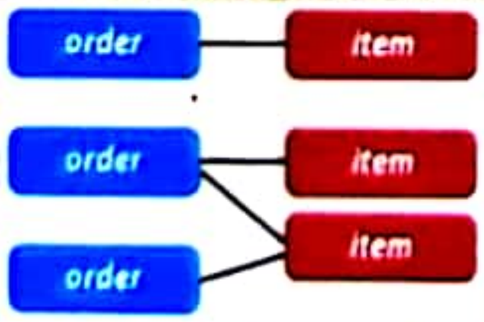
1) One-to-one relationship: Example: We have a relationship between the Customers table and the Addresses table. If each address can belong to only one customer, this relationship is "One to One".



2) One –to – many relationship: Example: Each customer may have zero, one or multiple orders. But an order can belong to only one customer.



3) Many- to – many Relationship: Example: In some cases, you may need multiple instances on both sides of the relationship.



For example, each order can contain multiple items. And each item can also be in multiple orders.

Cardinality: In the case of Data Modeling, Cardinality defines the number of attributes in one entity set, which can be associated with the number of attributes of other set via relationship set.

Example: One-to-one, One-to-many, Many-to-one, Many-to-many.

9. Differentiate between White box and Black Box Testing.

OR

Explain the concept of black box testing and white box testing.

Answer:

Sr. No.	White box testing	Black Box Testing
1	The tester needs to have the knowledge of internal code or program	This technique is used to test the software without the knowledge of internal code or program.
2	It aims at testing the structure of the item being tested.	It aims at testing the functionality of the software.
3	It is also called structural testing, clear box testing, code-based testing, or glass box testing.	It also knowns as data driven, box testing, data-, and functional testing.
4	Testing is best suited for a lower level of testing like Unit Testing, Integration testing.	This type of testing is ideal for higher levels of testing like System Testing, Acceptance testing.
5	Statement Coverage, Branch coverage, and Path coverage are White Box testing technique.	Equivalence partitioning, Boundary value analysis are Black Box testing technique
6	Can be based on detailed design documents.	Can be based on Requirement specification document

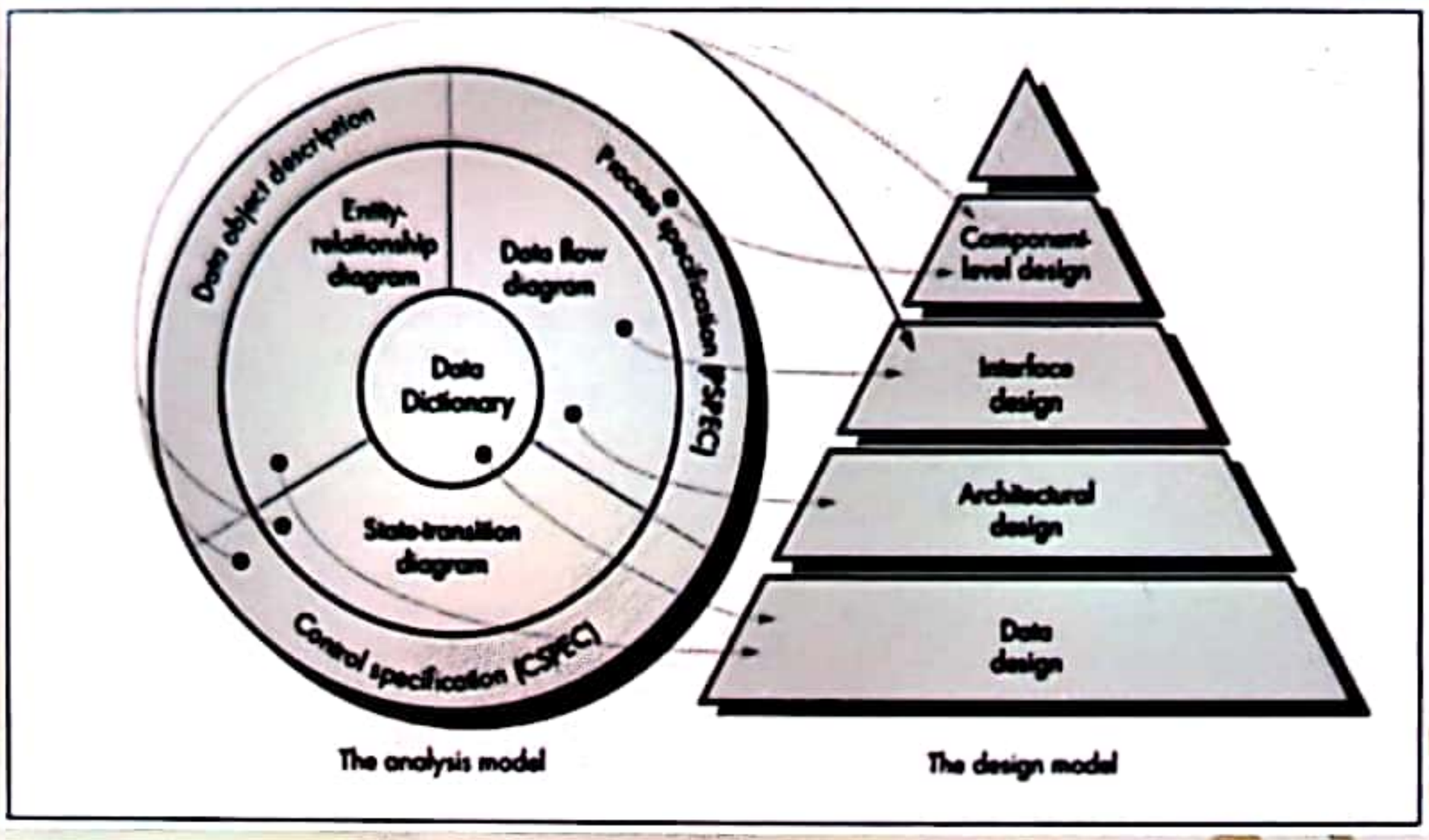
10. Draw and explain Transition diagram from requirement model to design model.

OR

Draw and explain translating requirement model into design model.

Answer:

Scanned with OKEN Scanner



1) Software requirements, manifested by the data, functional, and behavioural 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.
2) 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. Design is a meaningful engineering representation of something that is to be built.
3) It can be traced to a customer's requirements and at the same time assessed for quality against a set of predefined criteria for —good|| design. In the software engineering context, design focuses on four major areas of concern: data, architecture, interfaces, and components
4) Design begins with the requirements model. The data design transforms the information domain model created during analysis into the data structures that will be required to implement the software.
5) 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.
6) The architectural design defines the relationship between major structural elements of the software, the design pattern 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
7) The architectural design representation the framework of a computerbased system can be derived from the system specification, the analysis model, and the interaction of subsystems defined within the analysis model.
8) 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.

Scanned with OKEN Scanner

9) 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

Scanned with OKEN Scanner

Subject – Software Engineering (SEN– 22413)

(IMP Questions with Answers)

SY Diploma– Semester 4

I Scheme – MSBTE

Branches: CO/CM/CW/IF

Chapter 04 – Software Project Estimation

(Weightage – 16 Marks)

2 Marks Questions

1. Define Reactive Risk strategies.

Answer:

1) A reactive risk strategy monitors the project for likely risks. Resources are set aside to deal with them, should they become actual problems.
2) More commonly, the software team does nothing about risks until something goes wrong. Then, the team flies into action in an attempt to correct the problem rapidly.
3) This is often called a fire-fighting mode. When this fails, "crisis management" takes over and the project is in real jeopardy.

2. Specify following cost directives of cocomo:

- **Product attributes (any two)**
- **Hardware attributes (any two).**

Answer:

Product attributes –

1) Required software reliability extent
2) Size of the application database
3) The complexity of the product

Hardware attributes –

1) Run-time performance constraints
2) Memory constraints
3) The volatility of the virtual machine environment
4) Required turnabout time

3. Define Project Cost Estimation.

Answer:

Software cost estimation is the process of predicting the effort required to develop a software system. Project cost estimating is the process of predicting the total cost of the tasks, time, and resources required to deliver a project's scope of work.

Scanned with OKEN Scanner

4. Name two cost estimation approaches.

Answer:

1) Heuristic Estimation Approach
2) Analytical Estimation Approach
3) Empirical Estimation Approach

---

4 Marks Questions

---

5. Explain following elements of management spectrum: i. People ii. Process iii. Product iv. Project

OR

Describe 4 P's of management spectrum giving their significance.

OR

List 4P's of Management spectrum.

Answer:

The management Spectrum: 4p's Effective software project management focuses on the four P's: people, product, process, and project.

(i) The People:

1) The "people factor" is so important that the Software Engineering Institute has developed a People Capability Maturity Model (PeopleCMM) to continually improve its ability to attract, develop, motivate, organize, and retain the workforce needed to accomplish its strategic business objectives.
2) The people capability maturity model defines the following key practice areas for software people: a. Staffing b. communication and coordination c. work environment d. performance management e. Training, compensation, competency analysis and development, career development, workgroup development, team/culture development and others.
3) Organizations that achieve high levels of People-CMM maturity have higher likelihood of implementing effective software project management practices.

(ii) The Product:

1) Before a project can be planned, product objectives and scope should be established, alternative solutions should be considered and technical and management constraints should be identified.
2) Without this information, it is impossible to define reasonable (and accurate) estimates of the cost, an effective assessment of risk, a realistic breakdown of project tasks, or a manageable project schedule that provides a meaningful indication of progress.
3) Objectives identify the overall goals for the product (from the stakeholders' points of view) without considering how these goals will be achieved.
4) Scope identifies the primary data, functions, and behaviors that characterize the product

Scanned with OKEN Scanner

5) The alternatives enable managers and practitioners to select a "best" approach, given the constraints imposed by delivery deadlines, budgetary restrictions, personnel availability, technical interfaces, and other factors.

(iii) The Process:

1) A software process provides the framework from which a comprehensive plan for software development can be established.
2) A small number of framework activities are applicable to all software projects, regardless of their size or complexity.
3) A number of different task sets—tasks, milestones, work products, and quality assurance points enable the framework activities to be adapted to the characteristics of the software project and the requirements of the project team.
4) Finally, umbrella activities—such as software quality assurance, software configuration management, and measurement occur throughout the process.

(iv) The Project:

1) To manage complexity, we conduct planned and controlled software projects.
2) The success rate for present-day software projects may have improved but our project failure rate remains much higher than it should be.
3) To avoid project failure, a software project manager and the software engineers who build the product must avoid a set of common warning signs, understand the critical success factors that lead to good project management, and develop a common-sense approach for planning, monitoring, and controlling the project.

6. With an example, explain Line of Code (LOC) based estimation.

Answer:

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:

Example:

Scanned with OKEN Scanner

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

7. State importance of "Function point" and "lines of code" in concerned with project estimation.

OR

State and describe two metrics of project size estimation

Answer:

Currently two metrics are popularly being used widely to estimate size: lines of code (LOC) and function point (FP).

Lines of Code (LOC):

1) LOC is the simplest among all metrics available to estimate project size. This metric is very popular because it is the simplest to use.
2) Using this metric, the project size is estimated by counting the number of source instructions in the developed program. Obviously, while counting the number of source instructions, lines used for commenting the code and the header lines should be ignored.

Function Point (FP):

1) The conceptual idea behind the function point metric is that the size of a software product is directly dependent on the number of different Functions or features it supports.
2) A software product supporting many features would certainly be of larger size than a product with less number of features. Each function when invoked reads some input data and transforms it to the corresponding output data.
3) For example, the issue book feature (as shown in figure) of a Library Automation Software takes the name of the book as input and displays its location and the number of copies available. Thus, a computation of the number of input and the output data values to a system gives some indication of the number of functions supported by the system.

Scanned with OKEN Scanner

4) Albrecht postulated that in addition to the number of basic functions that a software performs, the size is also dependent on the number of files and the number of interfaces.

8. Describe the Analytical method of project cost estimation with example.

Answer:

1) Analytical estimation techniques derive the required results starting with basic assumptions regarding the project. Thus, unlike empirical and heuristic techniques, analytical techniques do have scientific basis.
2) Halstead's software science is an example of an analytical technique. Halstead's software science can be used to derive some interesting results starting with a few simple assumptions.
3) Halstead's software science is especially useful for estimating software maintenance efforts. In fact, it outperforms both empirical and heuristic techniques when used for predicting software maintenance efforts.
4) Halstead's Software Science – An Analytical Technique Halstead's software science is an analytical technique to measure size, development effort, and development cost of software products.
5) Halstead used a few primitive program parameters to develop the expressions for overall program length, potential minimum value, actual volume, effort, and development time.
6) For a given program, let:
   - η1 be the number of unique operators used in the program,
   - η2 be the number of unique operands used in the program,
   - N1 be the total number of operators used in the program,
   - N2 be the total number of operands used in the program.

Example: Let us consider the following C program:

```
main( )
{ int a, b, c, avg;
     scanf("%d %d %d", &a, &b, &c);
     avg = (a+b+c)/3;
     printf("avg = %d", avg);
} The unique operators are: main, (), {}, int, scanf, &, ", ", =, +, /,
printf
The unique operands are: a, b, c, &a, &b, &c, a+b+c, avg, 3, "%d %d
%d", "avg = %d"
Therefore, η1 = 12, η2 = 11
Estimated Length = (12*log12 + 11*log11)
= (12*3.58 + 11*3.45)
= (43+38) = 81
Volume = Length*log(23)
= 81*4.52
= 366
```


Scanned with OKEN Scanner

9. Explain any one project cost estimation approach.

Answer:

Heuristic:

1) Heuristic techniques assume that the relationships among the different project parameters can be modeled using suitable mathematical expressions.
2) Once the basic (independent) parameters are known, the other (dependent) parameters can be easily determined by substituting the value of the basic parameters in the mathematical expression.
3) Different heuristic estimation models can be divided into the following two classes: single variable model and the multi variable model.
4) Single variable estimation models provide a means to estimate the desired characteristics of a problem, using some previously estimated basic (independent) characteristic of the software product such as its size.
5) A single variable estimation model takes the following form:
   Estimated Parameter = $c_1 * e_1 d_1$
6) In the above expression, e is the characteristic of the software which has already been estimated (independent variable). Estimated Parameter is the dependent parameter to be estimated. The dependent parameter to be estimated could be effort, project duration, staff size, etc. c1 and d1 are constants. The values of the constants c1 and d1 are usually determined using data collected from past projects (historical data). The basic COCOMO model is an example of single variable cost estimation model. A multivariable cost estimation model takes the following form: Estimated Resource = $c_1 * e_1 d_1 + c_2 * e_2 d_2 + ...$ Where $e_1, e_2, ...$ are the basic (independent) characteristics of the software already estimated, and $c_1, c_2, d_1, d_2, ...$ are constants.

10. Describe COCOMO II model for evaluating size of software project with any three parameters in detail.

Answer:

1) COCOMO-II is the revised version of the original Cocomo (Constructive Cost Model) and is developed at University of Southern California. It is the model that allows one to estimate the cost, effort and schedule when planning a new software development activity.
2) COCOMO II provides the following three-stage series of models for estimation of Application Generator, System Integration, and Infrastructure software projects:

Scanned with OKEN Scanner

End User Programming	Application Generators and composition aids	Infrastructure
	Application Composition	
	System Integration	

3) The Application Composition Model This model involves prototyping efforts to resolve potential highrisk issues such as user interfaces, software/system interaction, performance, or technology maturity. The costs of this type of effort are best estimated by the Applications Composition model. It is suitable for projects built with modern GUI-builder tools. It is based on new Object Points. •
4) The Early Design Model The Early Design model involves exploration of alternative software/system architectures and concepts of operation. It uses a small set of new Cost Drivers, and new estimating equations. Based on Unadjusted Function Points or KSLOC.
5) The Post-Architecture Model The Post-Architecture model involves the actual development and maintenance of a software product Estimates In COCOMO II effort is expressed as Person Months (PM). The inputs are the Size of software development, a constant, A, and a scale factor, B. The size is in units of thousands of source lines of code (KSLOC). The constant, A, is used to capture the multiplicative effects on effort with projects of increasing size.
6) The parameters used in COCOMO II are described below:- a. Person month- A person month is the amount of time one person spends working on the software development project for one month.
7) COCOMO II uses the reuse model for maintenance when the amount of added or changed base source code is less than or equal to 20% or the new code being developed. Base code is source code that already exists and is being changed for use in the current project. For maintenance projects that involve more than 20% change in the existing base code (relative to new code being developed) COCOMO II uses maintenance size.

---

6 Marks Questions

---

11. Describe following project cost estimation approaches i) Heuristic ii) Analytical iii) Empirical.

Answer:

Empirical cost estimation approach:

Empirical estimation techniques are based on making an educated guess of the project parameters. While using this technique, prior experience with development of similar

Scanned with OKEN Scanner

products is helpful. Although empirical estimation techniques are based on common sense, different activities involved in estimation have been formalized over the years. Two popular empirical estimation techniques are:

1. Expert judgment technique and

2. Delphi cost estimation.

1) **Expert Judgment Technique**

Expert judgment is one of the most widely used estimation techniques.

- In this approach, an expert makes an educated guess of the problem size after analyzing the problem thoroughly.

- Usually, the expert estimates the cost of the different components (i.e. modules or subsystems) of the system and then combines them to arrive at the overall estimate.

- However, this technique is subject to human errors and individual bias. Also, it is possible that the expert may overlook some factors inadvertently.

- Further, an expert making an estimate may not have experience and knowledge of all aspects of a project.

2) **Delphi cost estimation**

- Delphi cost estimation approach tries to overcome some of the short comings of the expert judgment approach.

- Delphi estimation is carried out by a team comprising of a group of experts and a coordinator.

- In this approach, the coordinator provides each estimator with a copy of the software requirements specification (SRS) document and a form for recording his cost estimate.

- Estimators complete their individual estimates anonymously and submit to the coordinator. In their estimates, the estimators mention any unusual characteristic of the product which has influenced his estimation.

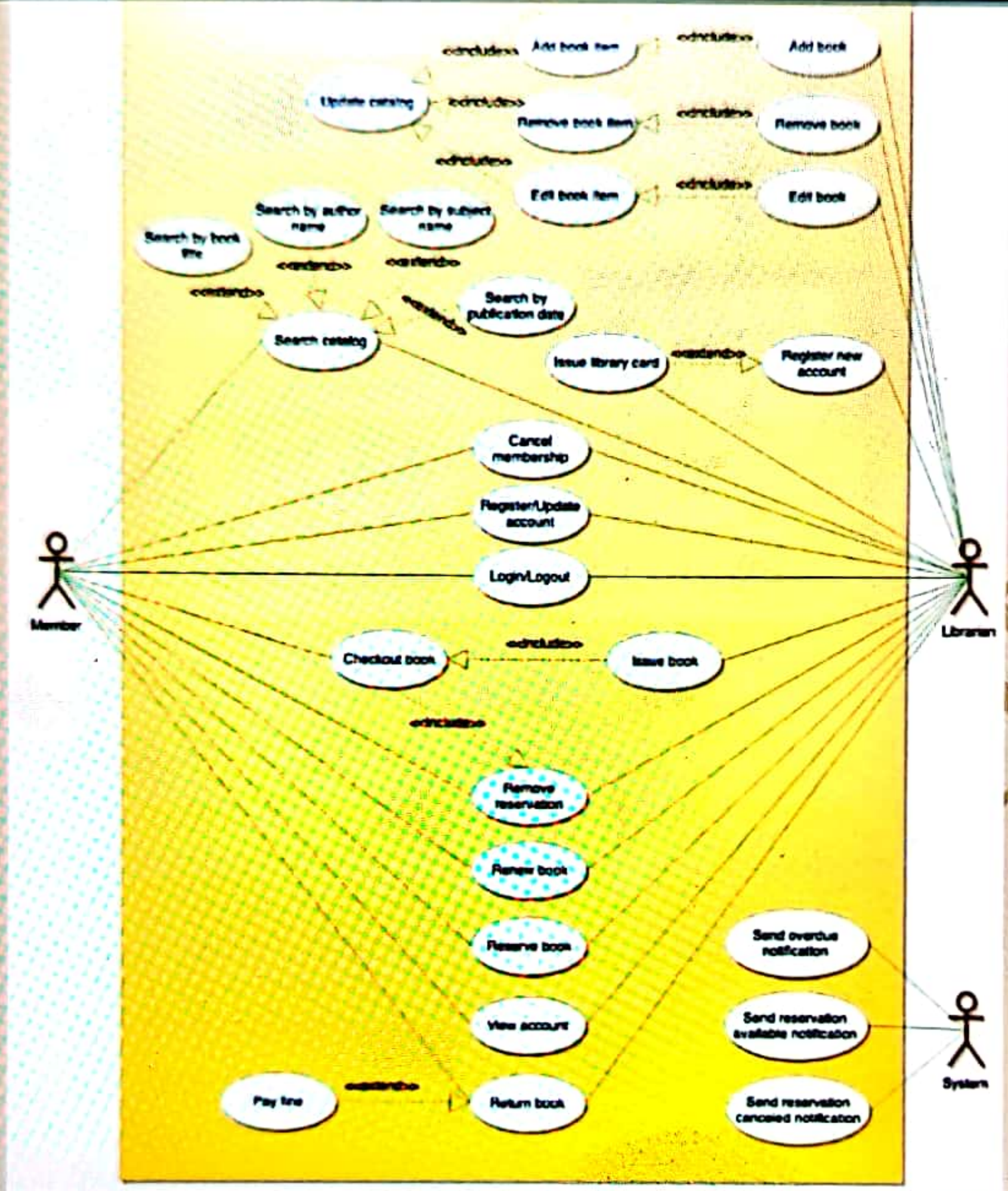
- The coordinator prepares and distributes the summary of the responses of all the estimators, and includes any unusual rationale noted by any of the estimators.

- Based on this summary, the estimators re-estimate.

12. Sketch a use case diagram for library management system with minimum four use cases and two actors.

Answer:

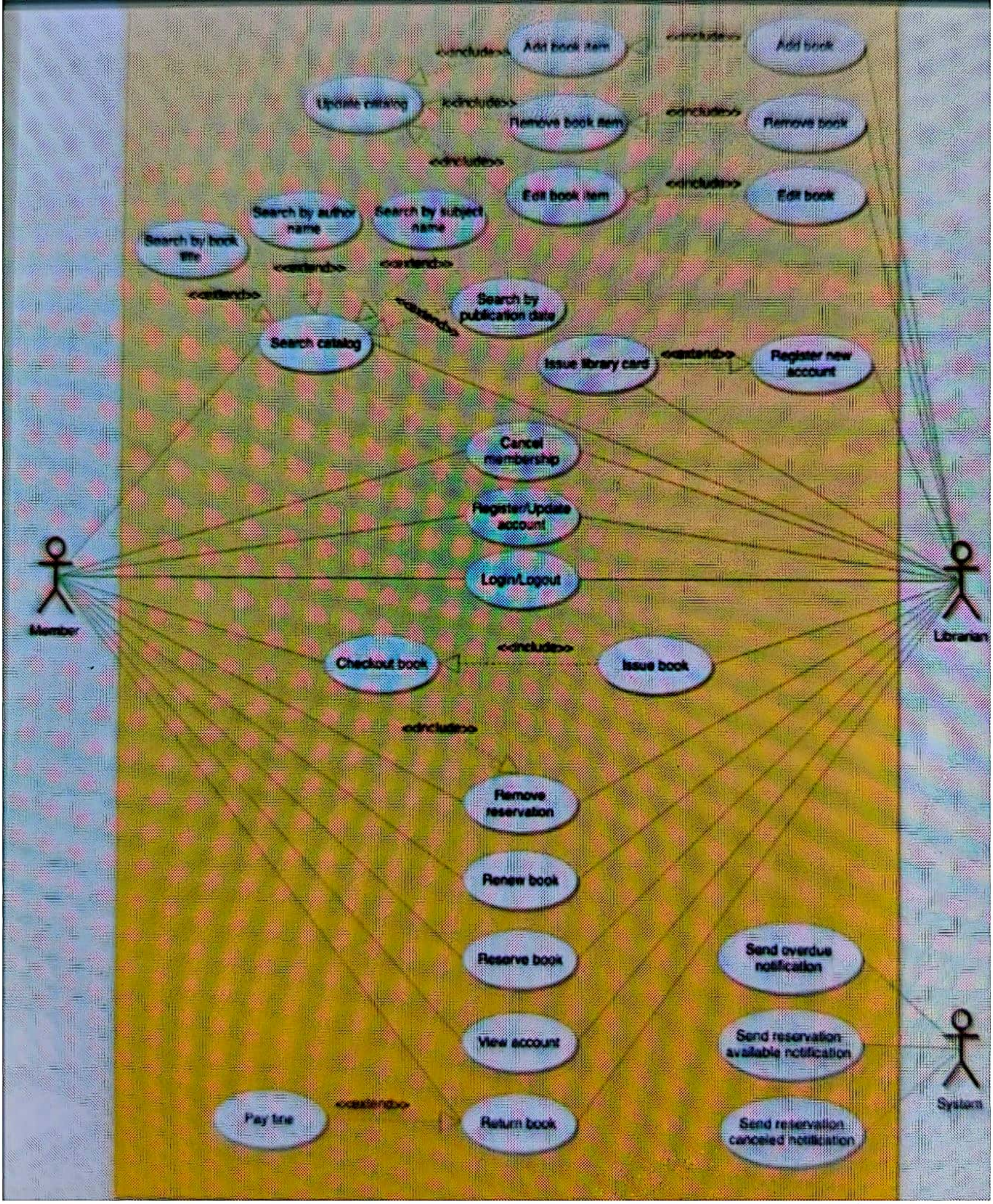
Scanned with OKEN Scanner



13. Calculate using COCOMO model i)Effort ii)Project duration iii)Average staff size If estimated size of project is 200 KLOC using organic mode.

Answer:

Scanned with OKEN Scanner



Scanned with OKEN Scanner

Given data: size=200 KLOC mode= organic

1. Effort:

   $E = a (KLOC)^{b}$

   For organic a=2.4 and b= 1.05

   $E= 2.4 (200)^{1.05}$

   = 626 staff members

2. Project duration:

   $TDEV= c (E)^{d}$

   Where TDEV= time for development

   c and d are constant to be determined

   E = effort

   For organic mode, c= 2.5 and d= 0.38

   $TDEV= 2.5 (626)^{0.38}$

   = 29 months

3. Average staff size:

   SS = E/TDEV

   SS = 626/29 = 22 staffs

14. Use COCOMO model to calculate i) Effort ii) Development Time iii) Average staff size iv) Productivity
If estimated size of project is 400 KLOC using embedded mode.

Answer:

Scanned with OKEN Scanner

Given size if project= 400 KLOC; mode = embedded
In embedded mode : a= 1.8 b=1.20 c=2.5 d=0.32
i) Effort
E = ai (KLOC) bi
E=1.8* (400) 1.20
=1.8 * 1325.78
= 2386.40 per month

ii)Development time
D= c * E d
= 2.5 * (2386.40) 0.32
=2.5 * 12.04
=75.25 months

iii) Average staff size
ss= E/ D
=2386.40 / 75.25
=31.71 persons

iv) Productivity
P=KLOC /E
=400/ 2386.40
=0.16

Subject – Software Engineering (SEN– 22413)

(IMP Questions with Answers)

SY Diploma- Semester 4

I Scheme – MSBTE

Branches: CO/CM/CW/IF

Chapter 05 – Software Quality Assurance and Security

(Weightage – 14 Marks)

2 Marks Questions

1. Define software quality.

Answer:

(i) Quality means that a product satisfies the demands of its specifications
(ii) It also means achieving a high level of customer satisfaction with the product
(iii) In software systems this is difficult

1) Customer quality requirements(e.g. efficiency or reliability) often conflict with developer quality requirements (e.g. maintainability or reusability)
2) Software specifications are often incomplete, inconsistent, or ambiguous

2. Define Quality control.

Answer:

Quality Control: Software quality control is the set of procedures used by organizations to ensure that a software product meets its quality goals at the best value to the customer, and to continually improve the organization's ability to produce software products in the future.

3. Define Software Quality Assurance.

Answer:

Quality assurance consists of the auditing and reporting functions of management.

The goal of quality assurance is to provide management with the data necessary to be informed about product quality, thereby gaining insight and confidence that product quality is meeting its goals.

4. Name four software quality assurance activities.

Answer:

These activities are performed (or facilitated) by an independent SQA group that:

1) Prepares an SQA plan for a project.
2) Participates in the development of the project's software process description.
3) Reviews software engineering activities to verify compliance with the defined software process.

Scanned with OKEN Scanner

4) Audits designated software work products to verify compliance with those defined as part of the software process.

5. Define proactive and reactive risk strategy.

Answer:

Reactive risk strategies:

1) Reactive risk strategy follows that the risks have to be tackled at the time of their occurrence.
2) No precautions are to be taken as per this strategy.
3) They are meant for risks with relatively smaller impact.
4) More commonly, the software team does nothing about risks until something goes wrong.

Proactive risk strategies:

1) It follows that the risks have to be identified before start of the project.
2) They have to be analyzed by assessing their probability of occurrence, their impact after occurrence, and steps to be followed for its precaution

6. Differentiate between Software Quality Management and Software Quality Assurance (any two points).

Answer:

Sr. No.	Software Quality Assurance (QA)	Software Quality Control (QC)
1	It is a procedure that focuses on providing assurance that quality requested will be achieved	It is a procedure that focuses on fulfilling the quality requested
2	QA aims to prevent the defect	QC aims to identify and fix defects
3	It is a method to manage the quality- Verification	It is a method to verify the quality-Validation
4	It does not involve executing the program	It always involves executing a program

4 Marks Questions

7. List and explain basic principles of project scheduling.

State and describe any four basic project scheduling principles.

Answer:

Basic Principles:

1) Compartmentalization: The project must be compartmentalized into a number of manageable activities and tasks.
2) Interdependency: The interdependency of each compartmentalized activity or task must be determined.

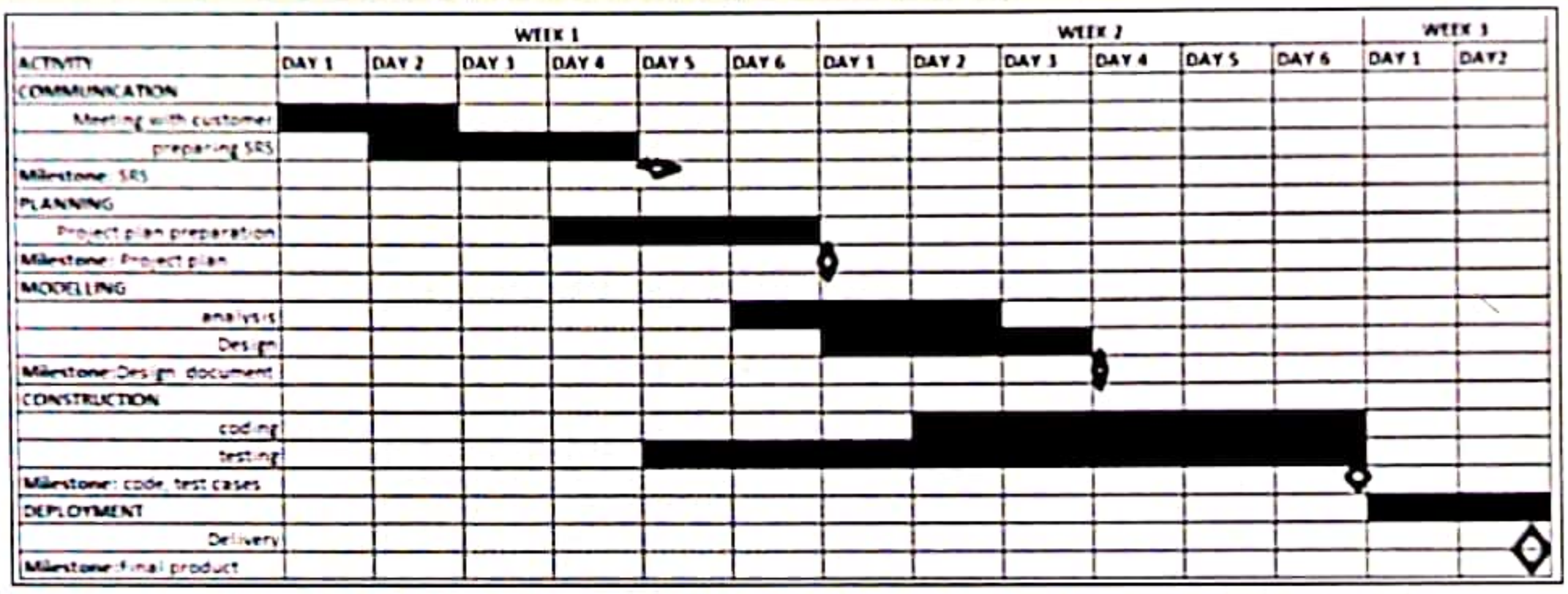
Scanned with OKEN Scanner

task must be determined.

3) Time allocation: Each task to be scheduled must be allocated some number of work units.
4) Effort validation: Every project has a defined number of staff members.
5) Defined responsibilities: Every task that is scheduled should be assigned to a specific team member.
6) Defined outcomes: Every task that is scheduled should have a defined outcome.

8. Prepare Macro Timeline chart for 20 days of Hotel Management system (6 days a week) consider broad phase of SDLC.

Answer:

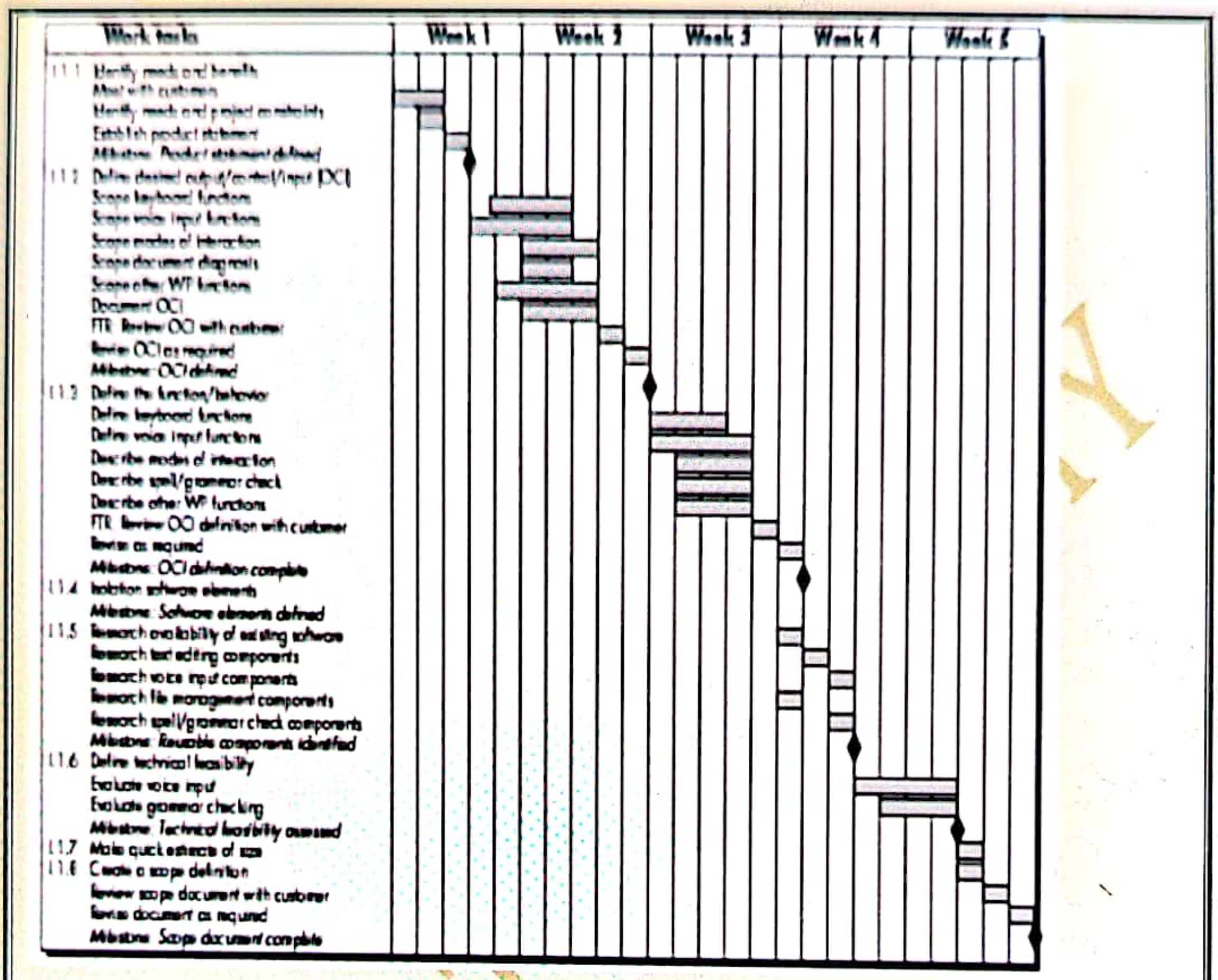


9. Explain GANTT chart and its application for project tracking with an example.

Answer:

1) When creating software project schedule, we begin with a set of tasks. If automated tools are used, the work breakdown is input as a task network or task outline.
2) Effort, duration and start date are then input for each task, In addition, tasks may be assigned to specific individuals. As a consequence of this input, a time-line chart, also called a Gantt chart is generated.
3) A time-line chart can be developed for the entire project. The figure below depicts a part of a software project schedule that emphasizes scoping task for a word-processing (WP) software product. All project tasks are listed in the left-hand column.
4) The horizontal bars indicate the duration of each task. When multiple bars occur at the same time on the calendar, task concurrency is implied. The diamond indicates milestones.
5) Once the information necessary for the generation of a time-line chart has been input, the majority of software project scheduling tools produce project tables – a tabular listing of all project tasks, their planned and actual start and end dates, and a variety of related information.
6) Used in conjunction with the time-line chart, project tables enable to track progress

Scanned with OKEN Scanner



Application of Gantt chart:

1) The sheer simplicity and ease-of-access of all relevant information make Gantt charts an ideal choice for teams to use them for organizing their schedules. Due to this, Gantt charts are widely used in project management, IT and development teams.
2) Apart from them, marketing, engineering, product launch, manufacturing teams can also use Gantt charts to get an overview of how things are rolling on the work front.

10. Explain RMMM plan with example.

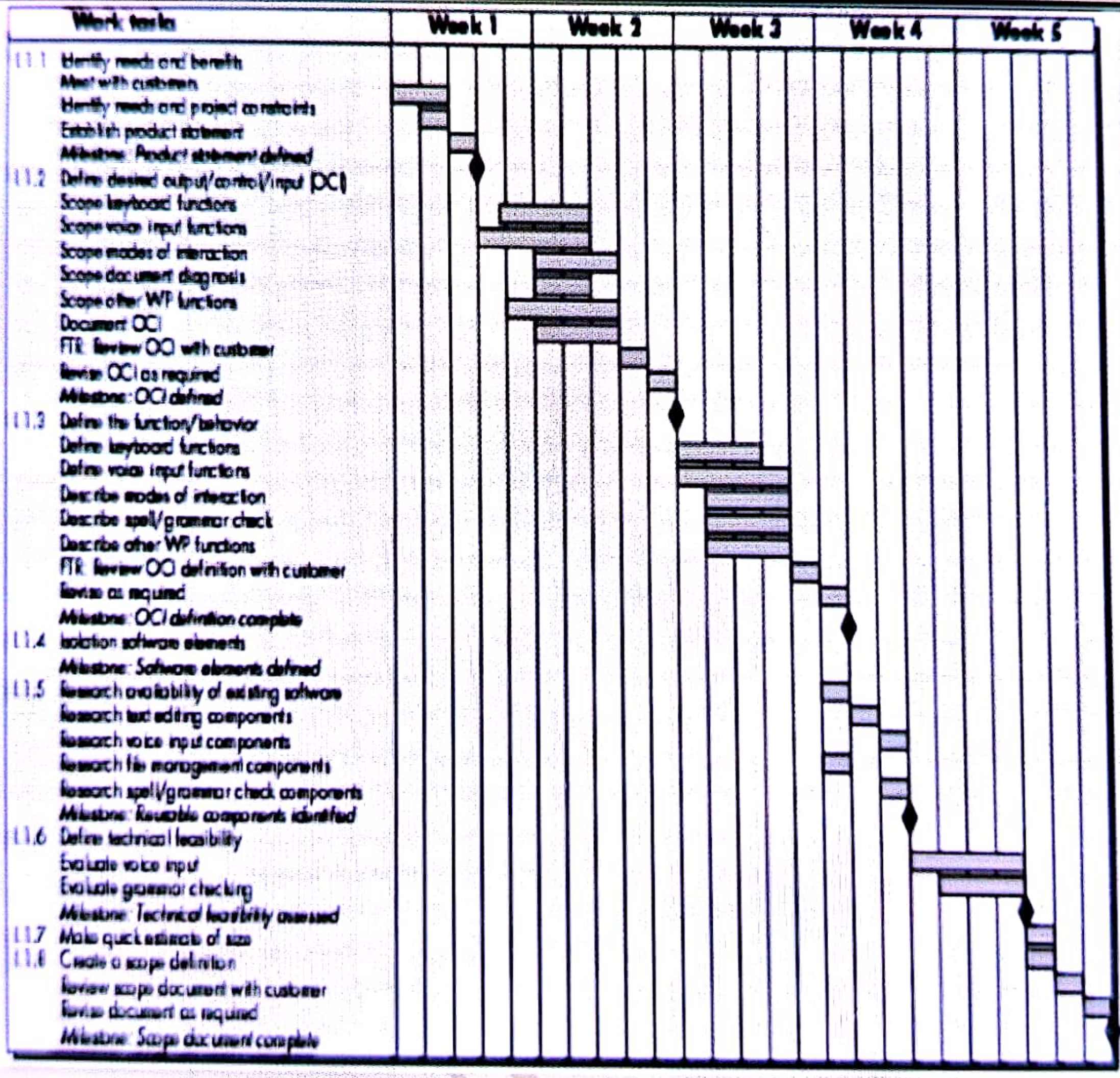
OR

Describe RMMM Strategy.

Answer:

1) A risk management plan or plan risk management is a document that a prepares to foresee risks, estimate impacts, and define responses to risks. It also contains a risk matrix.
2) A risk is "an uncertain event or condition that, if it occurs, has a positive or negative effect on a project's objectives." Risk is inherent with any and project manager should assess risks continually and develop plans to address them.

Scanned with OKEN Scanner



Application of Gantt chart:

1) The sheer simplicity and ease-of-access of all relevant information make Gantt charts an ideal choice for teams to use them for organizing their schedules. Due to this, Gantt charts are widely used in project management, IT and developmen teams.
2) Apart from them, marketing, engineering, product launch, manufacturing teams can also use Gantt charts to get an overview of how things are rolling on the wo

Scanned with OKEN Scanner

3) The risk management plan contains an analysis of likely risks with both high and low impact, as well as mitigation strategies to help the project avoid being derailed should common problems arise.
4) Risk management plans should be periodically reviewed by the project team to avoid having the analysis become stale and not reflective of actual potential project risks. Most critically, risk management plans include a risk strategy.
5) There are two characteristics of risk i.e. uncertainty and loss. Risk Mitigation, Monitoring and Management (RMMM) Risk analysis support the project team in constructing a strategy to deal with risks.
6) There are three important issues considered in developing an effective strategy: • Risk avoidance or mitigation - It is the primary strategy which is fulfilled through a plan. • Risk monitoring - The project manager monitors the factors and gives an indication whether the risk is becoming more or less. • Risk management and planning - It assumes that the mitigation effort failed and the risk is a reality.
7) RMMM PlanIt is a part of the software development plan or a separate document.
8) The RMMM plan documents all work executed as a part of risk analysis and used by the project manager as a part of the overall project plan. • The risk mitigation and monitoring starts after the project is started and the documentation of RMMM is completed.

Risk: Computer Crash

Mitigation: The cost associated with a computer crash resulting in a loss of data is crucial. A computer crash itself is not crucial, but rather the loss of data. A loss of data will result in not being able to deliver the product to the customer. This will result in a not receiving a letter of acceptance from the customer. Without the letter of acceptance, the group will receive a failing grade for the course. As a result the organization is taking steps to make multiple backup copies of the software in development and all documentation associated with it, in multiple locations. ·

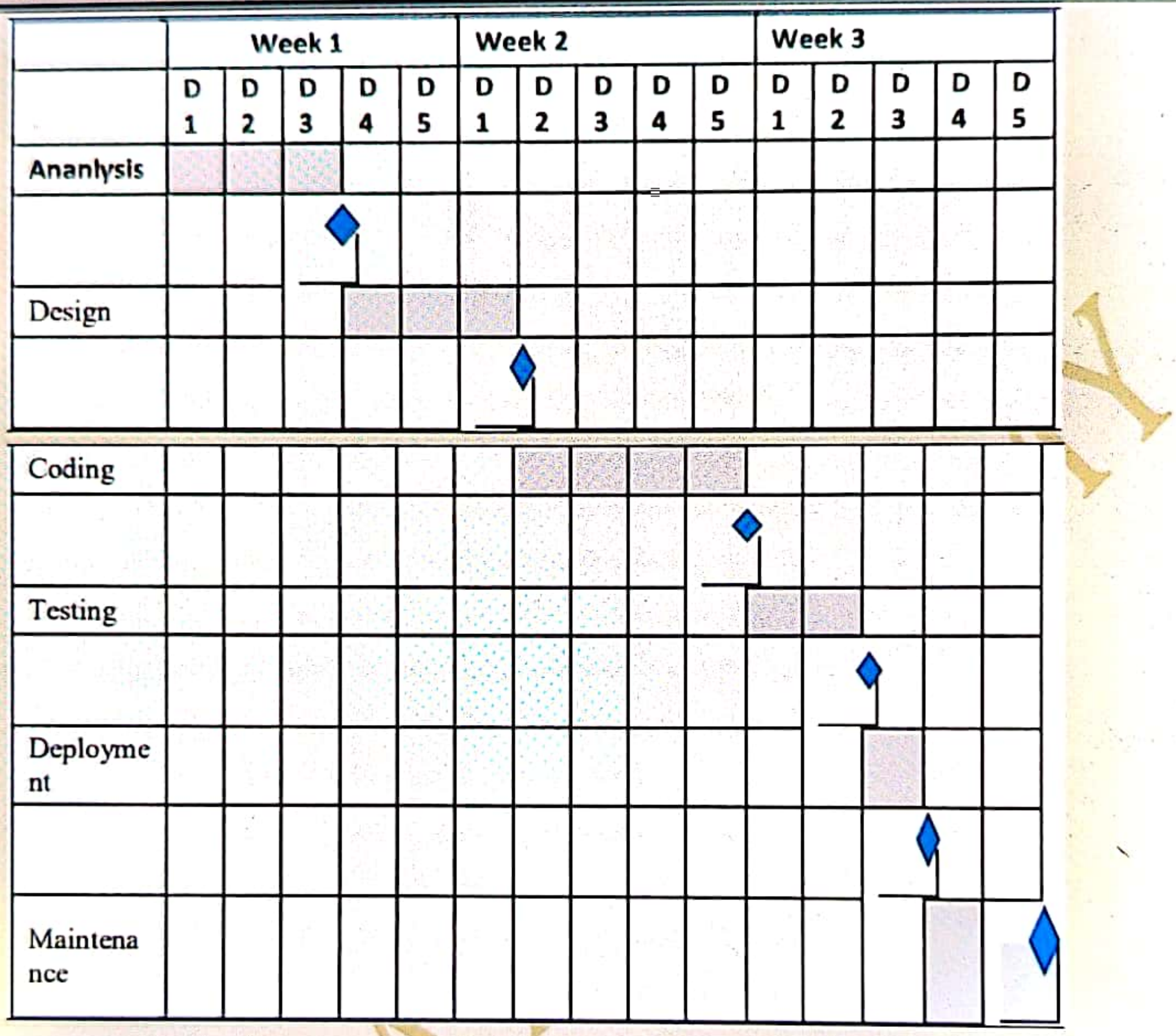
Monitoring: When working on the product or documentation, the staff member should always be aware of the stability of the computing environment they're working in. Any changes in the stability of the environment should be recognized and taken seriously.

Management: The lack of a stable-computing environment is extremely hazardous to a software development team. In the event that the computing environment is found unstable, the development team should cease work on that system until the environment is made stable again, or should move to a system that is stable and continue working there

11. Draw time chart for Library management system System (5 days a week). Consider broad phases of SDLC.

Answer:

Scanned with OKEN Scanner



6 Marks Questions

12. Describe CMMI. Give significance of each level.

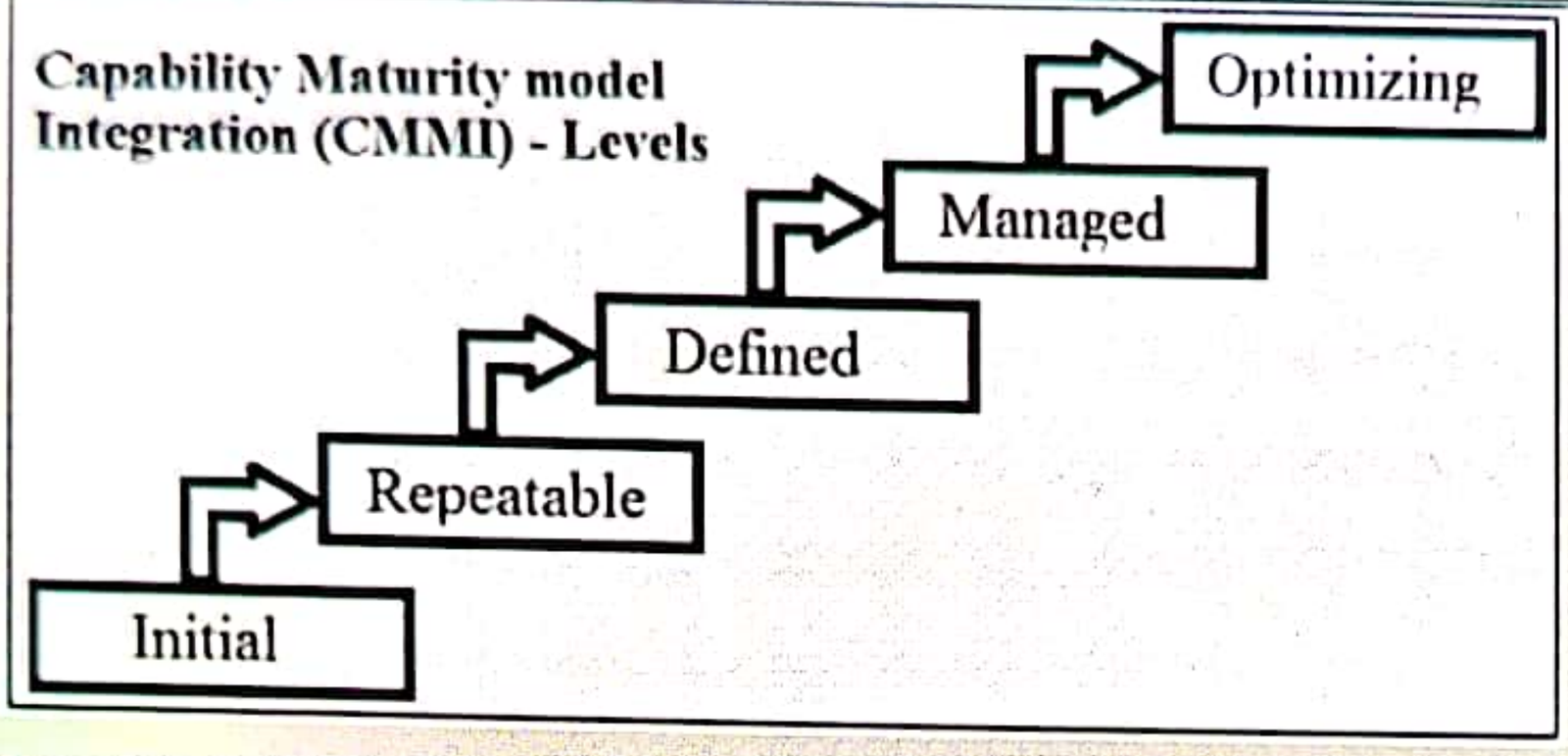
OR

Explain CMMI in detail with neat diagram.

Answer:

The Capability Maturity Model Integration (CMMI), a comprehensive process meta-model that is predicated on a set of system and software engineering capabilities that should be present as organizations reach different levels of process capability and maturity. The CMMI represents a process meta-model in two different ways: (1) Continuous model and (2) Staged model. The continuous CMMI meta-model describes a process in two dimensions. Each process area (e.g. project planning or requirements management) is formally assessed against specific goals and practices and is rated according to the following capability levels:

Scanned with OKEN Scanner



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 2

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 3

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.

13. Compare CMMI and ISO for software w.r.to i) scope ii)Approach Iii) Implementation.

Answer:

Difference between CMMI and ISO based on

(i) SCOPE:

1) CMMI is rigid and extends only to businesses developing software intensive systems. ISO is flexible and applicable to all manufacturing industries. CMMI focuses on engineering and project management processes whereas ISO's focus is generic in nature.
2) CMMI mandates generic and specific practices and businesses have a choice of selecting the model relevant to their business needs from 22 developed process areas. ISO requirements are same for all companies, industries, and disciplines.

(ii) APPROACH:

Scanned with OKEN Scanner

1) CMMI requires ingraining processes into business needs so that such processes become part of corporate culture and do not break down under the pressure of deadlines.
2) ISO specifies to conformance and remains oblivious as to whether such conformance is of strategic business value or not.
3) CMMI approaches risk management as an organized and technical discipline by identifying risk factors, quantifying such risk factors, and tracking them throughout the project life cycle. ISO was until recently neutral on risk management.
4) ISO 31000:2009 now provides generic guidelines for the design, implementation, and maintenance of risk management processes throughout an organization.
5) Although CMMI focuses on linkage of processes to business goals, customer satisfaction is not a factor in the ranking whereas customer satisfaction is an important part of ISO requirements.

(iii) IMPLEMENTATION:

1) Neither CMMI nor ISO requires the establishment of new processes. CMMI compares the existing processes to industry best practices whereas ISO requires adjustment of existing processes to confirm to the specific ISO requirements.
2) In practice, some organizations tend to rely on extensive documentation while implementing both CMMI and ISO. Most organizations tend to constitute in-house teams, or rely on external auditors to see through the implementation process.

Scanned with OKEN Scanner