CMR INSTITUTE OF TECHNOLOGY

USN | | | | | | | | | | |

Internal Assesment Test – I

SCHEME AND SOLUTION

Sub:	SOFTWARE ENGINEERING					Code:	15CS42		
Date:	27 / 03 / 2017	Duration:	90 mins	Max Marks:	50	Sem:	4 -D	Branch:	CSE

Answer FIVE FULL questions selecting AT LEAST ONE question from each part

		Marks	OBE CO	OBE RBT
	PART A			
1 (a)	**Define Software, Software crisis and Software Engineering.**	[3]	CO1	L1

Scheme:

Software Definition- 1 mark

Software crisis definition – 1 mark

Software Engineering Definition- 1 mark

Solution:

Software:

- Computer programs and associated docum is called Software

Software Crisis:

- Definition:- The inability to develop software on time, within budget and within requirements which causes large scale financial loss, that affects economic, political and administrative systems of various countries in world is called Software crisis.

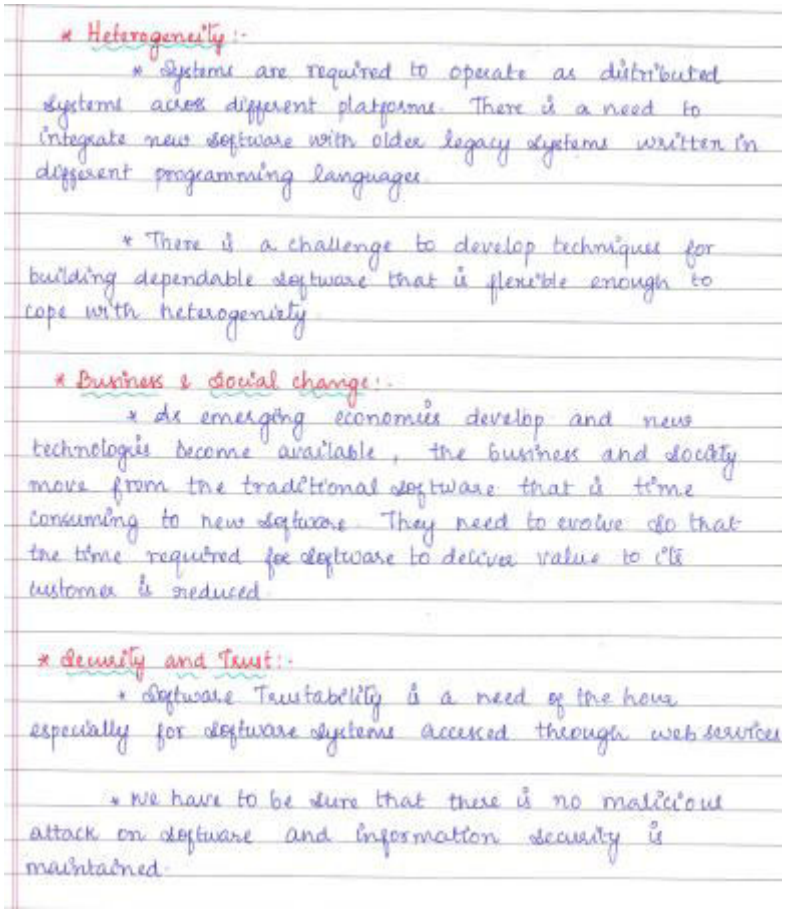
Software Engineering :

- Software Engineering is an engineering discipline that is concerned with all aspects of Software production from the early stages of system specification through to maintaining the system after it has gone into use.

(b)	**Discuss the key challenges of software engineering.**	[3]	CO1	L2

Scheme: **3 challenges x 1 mark = 3 mark**

SOLUTION:



* Heterogeneity :-

* Systems are required to operate as distributed systems across different platforms. There is a need to integrate new software with older legacy systems written in different programming languages.

* There is a challenge to develop techniques for building dependable software that is flexible enough to cope with heterogeneity.

* Business & Social change :-

* As emerging economies develop and new technologies become available, the business and society move from the traditional software that is time consuming to new software. They need to evolve so that the time required for software to deliver value to its customers is reduced.

* Security and Trust :-

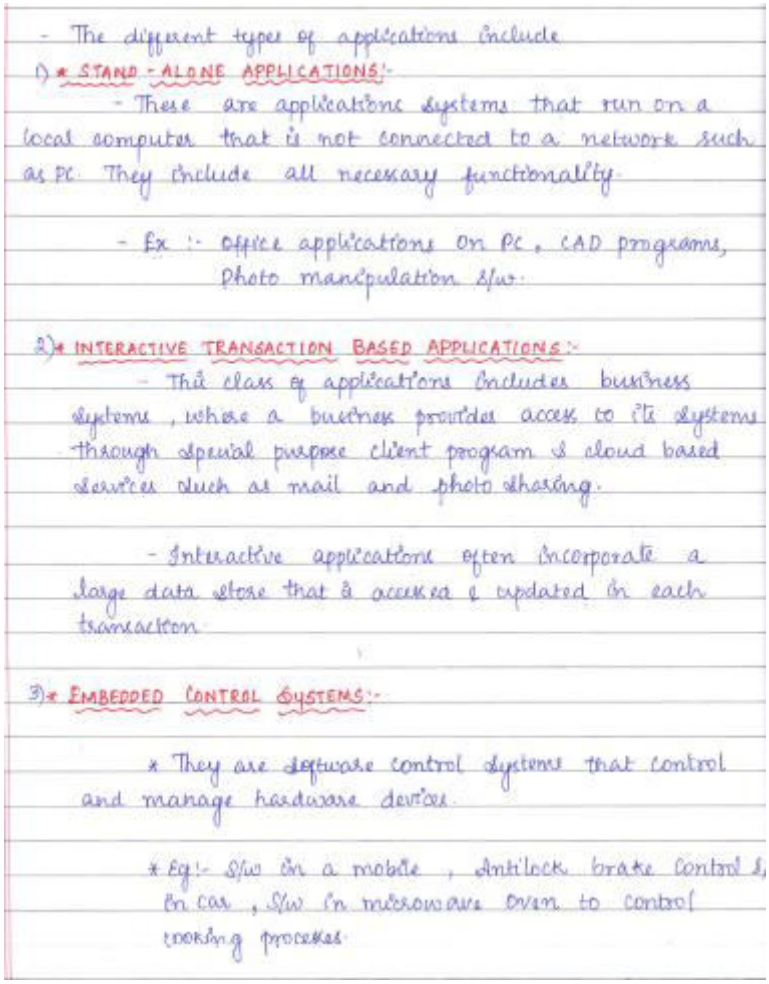
* Software Trustability is a need of the hour especially for software systems accessed through web services.

* We have to be sure that there is no malicious attack on software and information security is maintained.

(c) **Define any four types of applications with example.** [4] CO3 L1

SCHEME : FOUR APPLICATIONS x 1 mark = 4 marks

SOLUTION :



- The different types of applications include

1) * STAND - ALONE APPLICATIONS:-

- These are applications systems that run on a local computer that is not connected to a network such as PC. They include all necessary functionality.

- Ex :- office applications on PC, CAD programs, Photo manipulation s/w.

2) * INTERACTIVE TRANSACTION BASED APPLICATIONS :-

- This class of applications includes business systems, where a business provides access to its systems through special purpose client program & cloud based services such as mail and photo sharing.

- Interactive applications often incorporate a large data store that is accessed & updated in each transaction.

3) * EMBEDDED CONTROL SYSTEMS :-

* They are software control systems that control and manage hardware devices.

* Eg :- S/w in a mobile, Antilock brake control S/w in car, S/w in microwave oven to control cooking processes.

4) * BATCH PROCESSING SYSTEMS:-

- These are business systems designed to process data in large batches. They process large number of individual inputs to corresponding outputs.
Eg:- Periodic Billing Systems < Phone Billing System, Salary Payment System.

5) * ENTERTAINMENT SYSTEMS:-

- Entertainment systems are intended to entertain user. Most of these systems are game or the other media.

- The Quality of interaction offered is the most important characteristic of entertainment systems.

6) * SYSTEM FOR MODELING & SIMULATION:-

- These are systems developed to model physical process or simulation which include many individual interacting objects.

- They are computationally intensive and require high performance parallel systems for execution.

7) * Data Collection systems:-

- These are systems that collect data from their environment interacting with sensors and send that data to other systems for processing. The sensors are installed in a hostile environment such as inside an engine or in remote location.

8) * System of Systems:- Ex: U.S Airspace S/w Systems.

These systems are composed of a number of other software systems. Some software systems may be generic and some specific to the environment.

OR

2 (a) **State the two fundamental types of software products.** [1] CO1 L1

SCHEME: Each product— 0.5 mark x2

SOLUTION:

* Generic Product:-

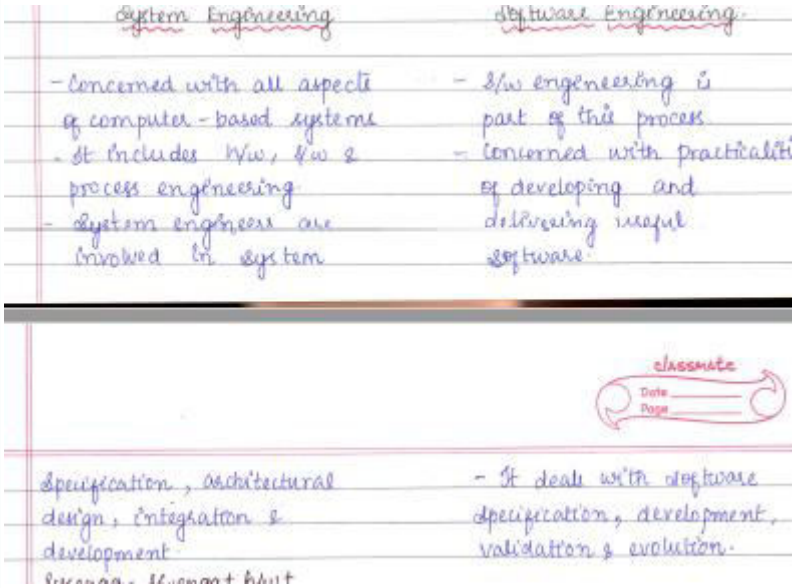
- These are stand alone systems that are produced by a development organization and sold on the open market to any customer who is able to buy them.

- Customized (or bespoke) products:-

- These are systems commissioned by a particular customer. A Software contractor develops software especially for that customer.

(b) **What is the difference between Software engineering and System Engineering** [2] CO1 L1

SCHEME: 2 differences X 1 mark
SOLUTION:



System Engineering

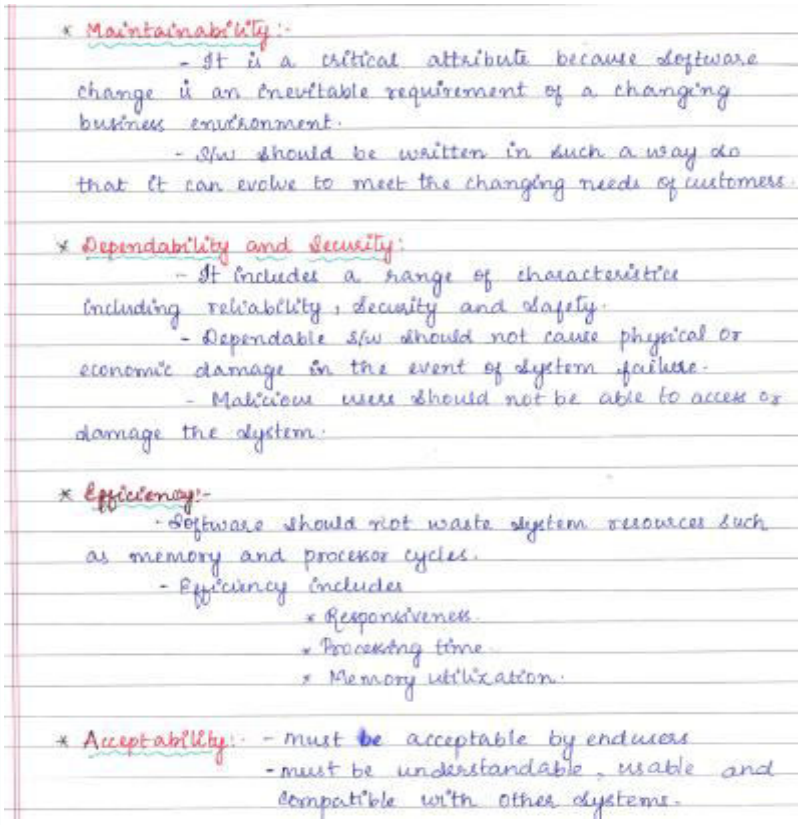
- Concerned with all aspects of computer-based systems
- It includes H/w, S/w & process engineering
- System engineers are involved in system Specification, Architectural design, Integration & development

Software Engineering

- S/w engineering is part of this process
- Concerned with practicalities of developing and delivering useful software.
- It deals with software specification, development, validation & evolution.

(c) **List and explain any three attributes of good software.** [3] CO1 L2

SCHEME: 3 attributes – Each 1 mark
SOLUTION:



* Maintainability:-
  - It is a critical attribute because software change is an inevitable requirement of a changing business environment.
  - S/w should be written in such a way so that it can evolve to meet the changing needs of customers.

* Dependability and Security:-
  - It includes a range of characteristics including reliability, security and safety.
  - Dependable S/w should not cause physical or economic damage in the event of system failure.
  - Malicious users should not be able to access or damage the system.

* Efficiency:-
  - Software should not waste system resources such as memory and processor cycles.
  - Efficiency includes
    * Responsiveness.
    * Processing time.
    * Memory utilization.

* Acceptability:- - must be acceptable by end users
  - must be understandable, usable and compatible with other systems.

(d) **Explain any four professional responsibilities.** [4] CO2 L1

SCHEME: 4 responsibilties – Each 1 mark

SOLUTION:

- Some of the professional responsibility are
* Confidentiality:-
  Respect the confidentiality of employees or clients irrespective of formal confidentiality agreement
* Competence:-
  - Donot misrepresent the level of competance
  - Donot accept the work outside competance
* Intellectual property rights:-
  - Be aware of local laws governing intellectual property such as patents and copyright
  - Be careful that IPR of employees and clients is protected
* Computer misuse:-
  Donot misuse computers, anything from trivial to serious misuses Eg:- Playing game (trivial), Hacking (serious)

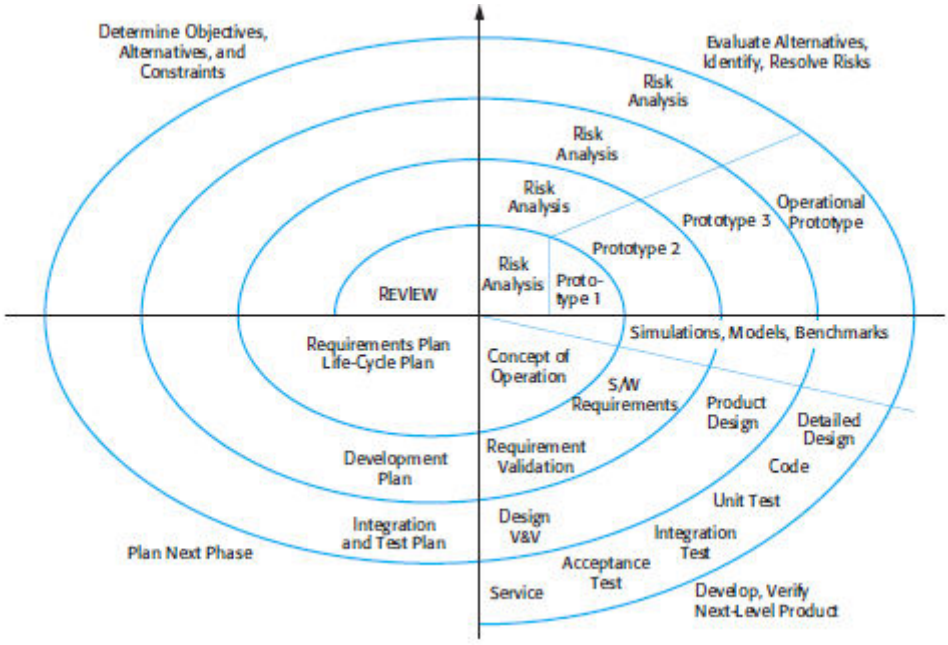
PART B

3 **Explain with a neat diagram the process model best suited for risk analysis.** [10] CO1 L2

SCHEME: For the identification (Boehm's spiral model) - 1 mark
Diagram – 4 marks
Spiral model introduction – 1 mark
Sectors – 4 mark

SOLUTION:

Boehm's spiral model:



- It is a risk-driven software process framework.

- Here the software process is spiral the software process is represented as a spiral, rather than a sequence of activities with some backtracking from one activity to another.

- Each loop in the spiral represents a phase of the software process.

- Each loop in the spiral is split into four sectors:

➢ *Objective setting*

- Specific objectives for that phase of the project are defined.
- Constraints on the process and the product are identified and a detailed management plan is drawn up.
- Project risks are identified.
- Alternative strategies may be planned.

➢ *Risk assessment and reduction*

- For each of the identified project risks, a detailed analysis is carried out.
- Steps are taken to reduce the risk.
- Ex: If there is a risk that the requirements are inappropriate, a prototype system may be developed.

➢ *Development and validation*

- After risk evaluation, a development model for the system is chosen.

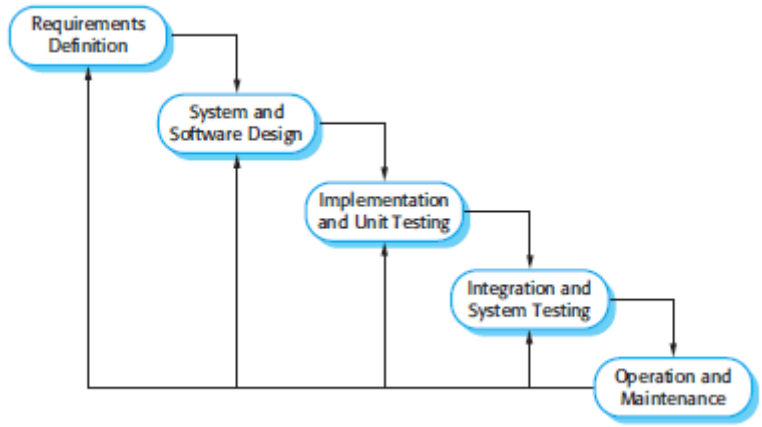
➢ *Planning*

- The project is reviewed and a decision made whether to continue with a further loop of the spiral.
- If it is decided to continue, plans are drawn up for the next phase of the project.

OR

4 (a)	**Name the process model that has reduced cost of accommodating change, involves feedback from customer and releases software in versions.** **SCHEME: For the identification (Incremental development) - 2 mark** **SOLUTION:** **Incremental development**	[2]	CO1	L2
(b)	Explain the steps in a waterfall model with a neat diagram. Discuss its advantages and disadvantages. **SCHEME:** **Diagram + explanation : 4 marks** **Advantages: 2 marks** **Disadvantages: 2 marks**		CO1	L2

SOLUTION:



- The waterfall model is an example of a plan-driven process.
- The principal stages of the waterfall model directly reflect the fundamental development activities:
  - *Requirements analysis and definition*
    The system's services, constraints, and goals are established by consultation with system users. They are then defined in detail and serve as a system specification.
  - *System and software design*
    The systems design process allocates the requirements to either hardware or software systems by establishing an overall system architecture.
  - *Implementation and unit testing*
    During this stage, the software design is realized as a set of programs or program units. Unit testing involves verifying that each unit meets its specification.
  - *Integration and system testing*
    The individual program units or programs are integrated and tested as a complete system to ensure that the software requirements have been met.
  - *Operation and maintenance*
    Maintenance involves correcting errors which were not discovered in earlier stages of the life cycle, enhancing the system's services as new requirements are discovered.
- Advantages:
  - It is consistent.
  - Managers can monitor progress against the development plan.

Disadvantages:

- Inflexible partitioning of the project into distinct stages.
- PreFreezing of requirements.

PART C

5 (a)	Differentiate between user and system requirements. Give example. SCHEME: difference: 1 mark Example: 1 mark	[2]	CO1	L2

SOLUTION:

User Requirement	System Requirement
User requirements are statements, in a natural language plus diagrams, of what services the system is expected to provide to system users and the constraints under which it must operate.	System requirements are more detailed descriptions of the software system's functions, services, and operational constraints.
Ex: The MHC-PMS shall generate monthly management reports showing the cost of drugs prescribed by each clinic during that month.	Ex: 1.1 On the last working day of each month, a summary of the drugs prescribed, their cost, and the prescribing clinics shall be generated. 1.2 The system shall automatically generate the report for printing after 17.30 on the last working day of the month. 1.3 A report shall be created for each clinic and shall list the individual drug names, the total number of prescriptions, the number of doses prescribed, and the total cost of the prescribed drugs. 1.4 If drugs are available in different dose units (e.g., 10 mg, 20 mg) separate reports shall be created for each dose unit. 1.5 Access to all cost reports shall be restricted to authorized users listed on a management access control list.

(b) Explain the structure of SRS. [8] CO1 L2

SCHEME: SRS TABLE: 8 marks

SOLUTION:

CHAPTER	DESCRIPTION
Preface	This should - define the expected readership of the document - describe its version history - a rationale for the creation of a new version - A summary of the changes made in each version.
Introduction	This should - describe the need for the system. - describe the system's functions and explain how it will work with other systems. - describe how the system fits into the overall business or strategic objectives of the organization commissioning the software
Glossary	This should define the technical terms used in the document
User Requirements Definition	- This should describe the services provided for the user. - The non-functional system requirements should also be described.
System Architecture	- This chapter should present a high-level overview of the anticipated system architecture, showing the distribution of functions across system modules. - Reuse of components should be specified.
System requirements specification	This should describe the functional and non-functional requirements in more detail.
System models	This might include graphical system models showing the relationships between the system components, the system, and its environment
System evolution	This should describe the fundamental assumptions on which the system is based, and any anticipated changes due to hardware evolution, changing user needs,
Appendices	These should provide detailed, specific information that is related to the application being developed; for example, hardware and database descriptions.
Index	Several indexes to the document may be included.

OR

6 (a) Differentiate between functional and non-functional requirements. Give example. [2] CO1 L2

SCHEME: difference :1 mark
Example: 2 marks

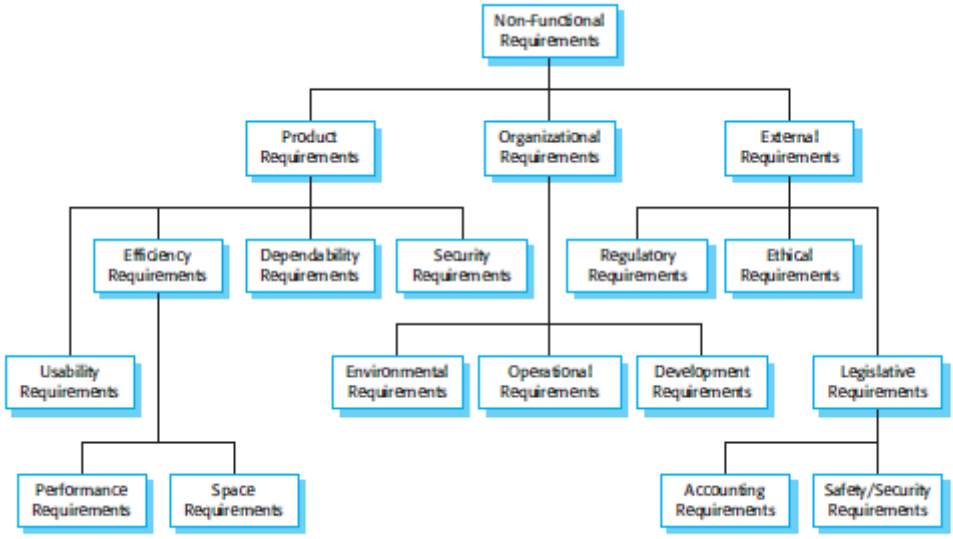
Solution:

Functional Requirement	Non functional Requirement
These are statements of services the system shouldprovide, how the system should react to particular inputs, and how the system should behave in particular situations	These are constraints on the services or functions offered by the system.
Ex: A user shall be able to search the appointments lists for all clinics.	Ex: The system downtime should not be more than 5 sec for a day

(b) Explain the types of non-functional requirements with examples. [8] CO1 L2

SCHEME: Diagram: 2 marks
3 categories: Each category: explanation:1 mark
Example: 1 mark (3 x2=6)

SOLUTION:



The non functional requirements are classified as

- *Product requirements* These requirements specify or constrain the behavior of the software.
- Examples include performance requirements on how fast the system must execute and how much memory it requires, reliability requirements that set out the acceptable failure rate, security requirements, and usability requirements.
- *Organizational requirements* These requirements are broad system requirement derived from policies and procedures in the customer's and developer's organization.
- Examples include operational process requirements that define how the system will be used, development process requirements that specify the programming language, the development environment or process standards to be used, and environmental requirements that specify the operating environment of the system.
- *External requirements* This broad heading covers all requirements that are derived from

factors external to the system and its development process.

- These may include regulatory requirements that set out what must be done for the system to be approved for use by a regulator, such as a central bank; legislative requirements that must be followed to ensure that the system operates within the law; and ethical requirements that ensure that the system will be acceptable to its users and the general public.

PART D

7 **List and Explain the Software process activities in detail.** [10] CO1 L2

SCHEME: list of four activities: (1)mark
Software specification: Diagram-1 , explanation-1(2)
Software Development: Diagram-1 , explanation-1(2)
Software Validation: idea: 1 mark, acceptance testing: 2 marks(3)
Software evolution: 2 marks

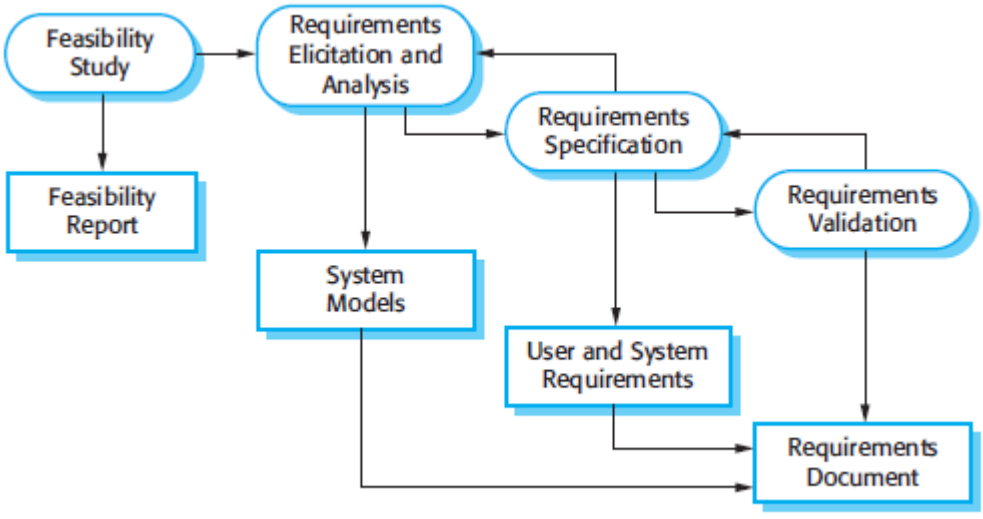
Solution:

- The four basic process activities are
  - Software specification
  - Software development
  - Software validation and
  - Software evolution

- **Software specification**

- Software specification or requirements engineering is the process of understanding and defining what services are required from the system and identifying the constraints on the system's operation and development.

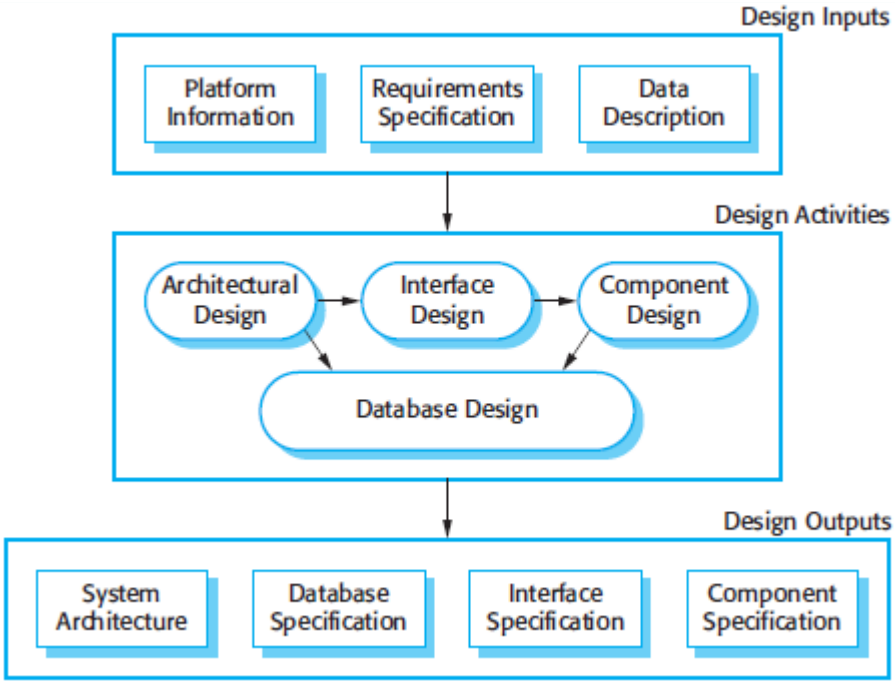
The requirements engineering process



- There are four main activities in the requirements engineering process:
  1. ***Feasibility study***
  2. ***Requirements elicitation and analysis***
  3. ***Requirements specification***
  4. ***Requirements validation***

- **Software design and implementation**
  - The implementation stage of software development is the process of converting a system specification into an executable system.

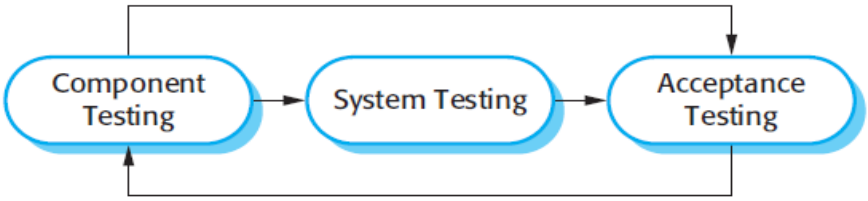
A general model of the design process



- There are four activities that are a part of the design process for information systems:
  1. ***Architectural design,*** where we identify the overall structure of the system,
  2. ***Interface design*** defines the interfaces between system components.
  3. ***Component design*** designs takes each system component and design how it will operate.
  4. ***Database design*** designs the system data structures and how these are to be represented in a database.

- **Software validation**
  - Verification – This is intended to show that a system both conforms to its specification
  - Validation – This is to confirm whether the system meets the expectations of the system customer.

  - There are three-stages in testing process



  - ***Development testing***
    - The components of the system are tested by the developers independently, without other system components

  - ***System testing***
    - This testing is done after integration of system components.
  - ***Acceptance testing***
    - In this testing, the system is tested with data supplied by the system customer rather than with simulated test data.
    - Acceptance testing reveals errors and omissions in the system requirements definition, because the real data executes in different ways from the test data.
    - **Alpha Testing**
      - Acceptance testing is sometimes called 'alpha testing'. Custom systems

are
developed for a single client.

- The alpha testing process continues until the system developer and the client agree that the delivered system is an acceptable implementation of the requirements.

- **Beta Testing**
  - When a system is to be marketed as a software product, a testing process called 'beta testing' is used.
  - Beta testing involves delivering a system to a number of potential customers who agree to use that system.
  - They report problems to the system developers.
  - After this feedback, the system is modified and released either for further beta testing or for general sale.

➢ **Software evolution**

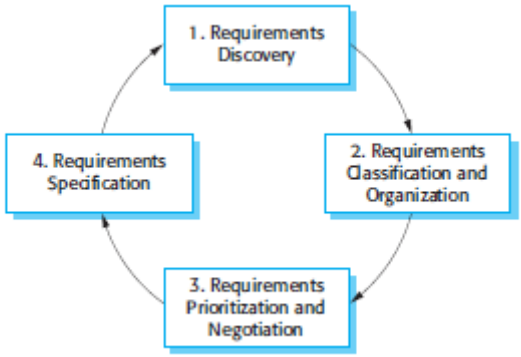
- The flexibility of software systems is one of the main reasons for the development of large, complex systems.
- Software changes are little cheaper than hardware changes.
- The costs of maintenance are often several times the initial development costs.
- Software engineering is an evolutionary process where software is continually changed over its lifetime in response to changing requirements and customer needs.

OR

8 (a) Draw the process model of the elicitation and analysis process. [2] CO1 L1

SCHEME: Diagram- 2marks

SOLUTION:



(b) Write a note on: [8] CO1 L2

- Scenarios
- Ethnography
- Use cases
- Interviews

SCHEME: Each topic 2 marks

SOLUTION:

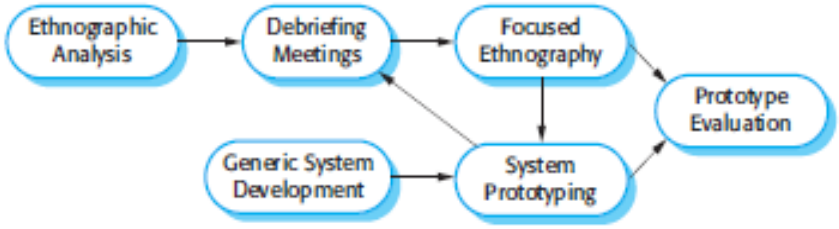
Scenarios:

A scenario starts with an outline of the interaction. During the elicitation process, details are added to this to create a complete description of that interaction. At its most general, a scenario may include:

1. A description of what the system and users expects when the scenario starts.

2. A description of the normal flow of events in the scenario.
3. A description of what can go wrong and how this is handled.
4. Information about other activities that might be going on at the same time.
5. A description of the system state when the scenario finishes.

Ethnography:

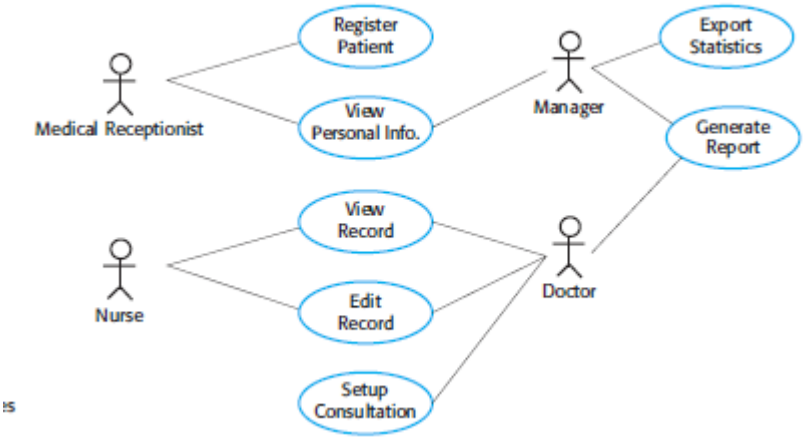


- Ethnography is an observational technique that can be used to understand operational processes and help derive support requirements for these processes.
- Ethnography is particularly effective for discovering two types of requirements:

  1. Requirements that are derived from the way in which people actually work, rather than the way in which process definitions say they ought to work.

  2. Requirements that are derived from cooperation and awareness of other people's activities.

Use cases



- Use cases are a requirements discovery technique that were first introduced in the Objectory method.
- Use cases are documented using a high-level use case diagram.
- The set of use cases represents all of the possible interactions that will be described in the system requirements.
- Actors in the process, who may be human or other systems, are represented as stick figures.
- Each class of interaction is represented as a named ellipse.
- Lines link the actors with the interaction. Optionally, arrowheads may be added to lines to show how the interaction is initiated.

Interviews:

- Interviews may be of two types:
- 1. Closed interviews, where the stakeholder answers a pre-defined set of questions.
- 2. Open interviews, in which there is no pre-defined agenda.
- It can be difficult to elicit domain knowledge through interviews for two reasons:
  - All application specialists use terminology and jargon that are specific to a domain.
  - Some domain knowledge is so familiar to stakeholders that they either find it difficult to explain or they think it is so fundamental that it isn't worth mentioning.
- Effective interviewers have two characteristics:
  - They are open-minded, avoid pre-conceived ideas about the requirements, and are willing to listen to stakeholders.
  - They prompt the interviewee to get discussions going using a springboard question, a requirements proposal, or by working together on a prototype system.

PART E

9	Define system modeling. Explain the Context model of MHC-PMS.	.	CO5	L3

SCHEME: Definition of system modeling : 2 marks
Context model idea: 1 mark
Diagram: 3 marks
Explanation : 4 marks

Solution: [2+8]

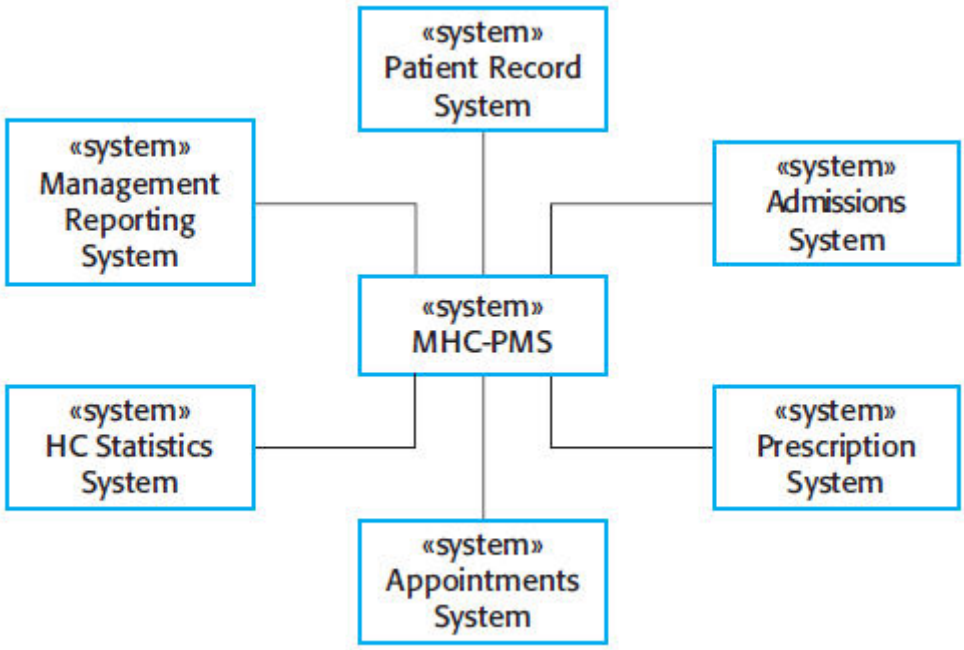
System Modeling:

- System modeling is the process of developing abstract models of a system with each model presenting a different view or perspective of that system.

Context model

It gives an external perspective and it used to model the context or environment of the system

The context model of the MHC-PMS



- The MHC-PMS is connected to
  - an appointments system
  - a more general patient record system with which it shares data.
  - Systems for management reporting and
  - System for admissions related to hospital bed allocation
  - a statistics system that collects information for research.
  - a prescription system to generate prescriptions for patients' medication.
- **Diasadvantages of context model**
  - Context models normally show that the environment includes several other automated systems. But they do not show the types of relationships between the systems in the environment and the system that is being specified
  - Therefore, simple context models are used along with other models, such as business process models.

OR

10	Draw and Explain the state diagram of microwave oven.	[10]	CO5	L3

SCHEME: State Diagram: 5 marks
Explanation : 5 marks

Solution:

State diagram of a microwave oven

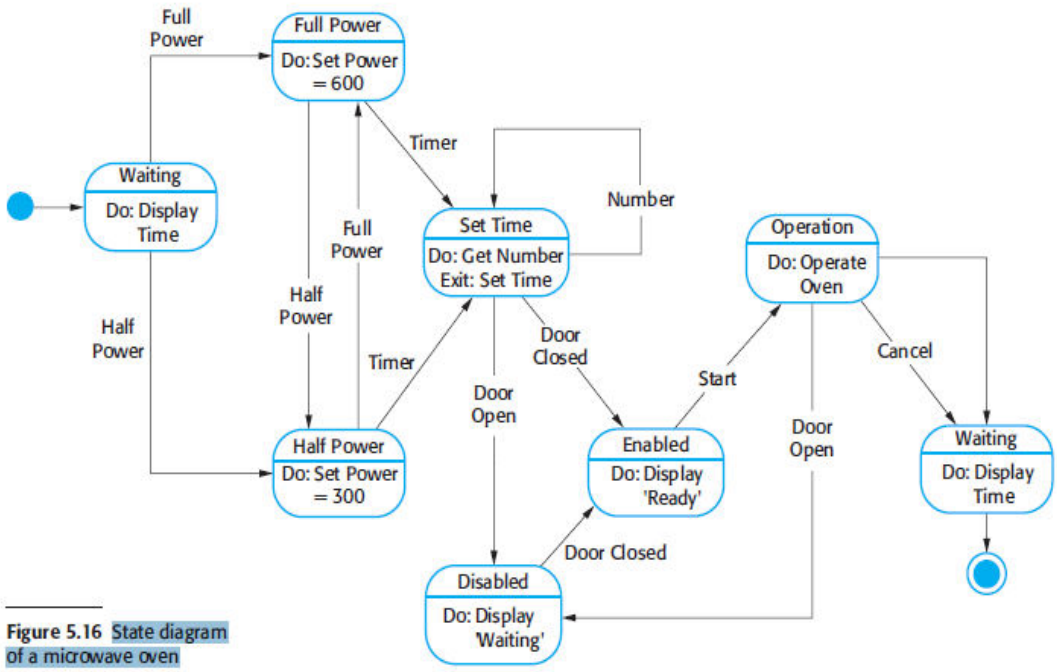


Figure 5.16 State diagram of a microwave oven

Explanation:

- The system starts in a waiting state and responds initially to either the full-power or the half-power button.
- Users can change their mind after selecting one of these and press the other button.
- The time is set and, if the door is closed, the Start button is enabled.
- Pushing this button starts the oven operation and cooking takes place for the specified time.
- This is the end of the cooking cycle and the system returns to the waiting state.

- In a detailed system specification ,a tabular description of each state and how the stimuli that force state transitions are specified.

States and stimuli for the microwave oven

State	Description
Waiting	The oven is waiting for input. The display shows the current time.
Half power	The oven power is set to 300 watts. The display shows 'Half power'.
Full power	The oven power is set to 600 watts. The display shows 'Full power'.
Set time	The cooking time is set to the user's input value. The display shows the cooking time selected and is updated as the time is set.
Disabled	Oven operation is disabled for safety. Interior oven light is on. Display shows 'Not ready'.
Enabled	Oven operation is enabled. Interior oven light is off. Display shows 'Ready to cook'.
Operation	Oven in operation. Interior oven light is on. Display shows the timer countdown. On completion of cooking, the buzzer is sounded for five seconds. Oven light is on. Display shows 'Cooking complete' while buzzer is sounding.
Stimulus	**Description**
Half power	The user has pressed the half-power button.
Full power	The user has pressed the full-power button.
Timer	The user has pressed one of the timer buttons.
Number	The user has pressed a numeric key.
Door open	The oven door switch is not closed.
Door closed	The oven door switch is closed.
Start	The user has pressed the Start button.
Cancel	The user has pressed the Cancel button.

- The problem with state-based modeling is that the number of possible states increases rapidly.
- For large system models, we use thenotion of a superstate that encapsulates a number of separate states.
- The superstate looks like a single state on a high-level model but is then expanded to show more detail on a separate diagram.

Course Outcomes		PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CO1:	Design a software system, component, or process to meet desired needs within realistic constraints	3	3	3	3	2	3	2	-	2	2	2	3
CO2:	Assess professional and ethical responsibility	2	-	-	-	-	3	2	3	3	2	-	3
CO3:	Function on multi-disciplinary teams	2	2	3	3	2	2	2	-	3	3	2	3
CO4:	Use the techniques, skills, and modern engineering tools necessary for engineering practice	3	3	3	3	3	2	2	-	2	2	2	3
CO5:	Analyze, design, implement, verify, validate, implement, apply, and maintain software systems or parts of software systems	3	3	3	3	2	2	2	-	2	2	2	3

Cognitive level	KEYWORDS
L1	List, define, tell, describe, identify, show, label, collect, examine, tabulate, quote, name, who, when, where, etc.
L2	summarize, describe, interpret, contrast, predict, associate, distinguish, estimate, differentiate, discuss, extend
L3	Apply, demonstrate, calculate, complete, illustrate, show, solve, examine, modify, relate, change, classify, experiment, discover.
L4	Analyze, separate, order, explain, connect, classify, arrange, divide, compare, select, explain, infer.
L5	Assess, decide, rank, grade, test, measure, recommend, convince, select, judge, explain, discriminate, support, conclude, compare, summarize.

PO1 - *Engineering knowledge*; PO2 - *Problem analysis*; PO3 - *Design/development of solutions*; PO4 - *Conduct investigations of complex problems*; PO5 - *Modern tool usage*; PO6 - *The Engineer and society*; PO7- *Environment and sustainability*; PO8 – *Ethics*; PO9 - *Individual and team work*; PO10 - *Communication*; PO11 - *Project management and finance*; PO12 - *Life-long learning*