CMR
INSTITUTE OF
TECHNOLOGY

										USN

CMR

Internal Assessment Test 1 – Sep 2016 - SCHEME OF EVALUATION

Sub:	Software Engineering			**Code:**	10IS51
Date: 6/09/2016	Duration: 90 mins	Max Marks: 50	**Sem:** V		**Branch:** ISE/CSE

Note: : Answer any 5 questions Total marks: 50

1. (a) Define Software, Software Engineering, Software Process, Software Process models. (4M)

Software:

It is collection of computer programs, configuration files (used to set up these programs), system documentation (describes the structure of the system), and user documentation (explains how to use the system) and web sites for users to download recent product information. **(1M)**

Software Engineering:

It is an engineering discipline that is concerned with all aspects of software production from early stages of system specification to maintaining the system after it has gone into use. Aim is cost effective development of high quality software systems to be developed by applying engineering principles. **(1M)**

Software Process:

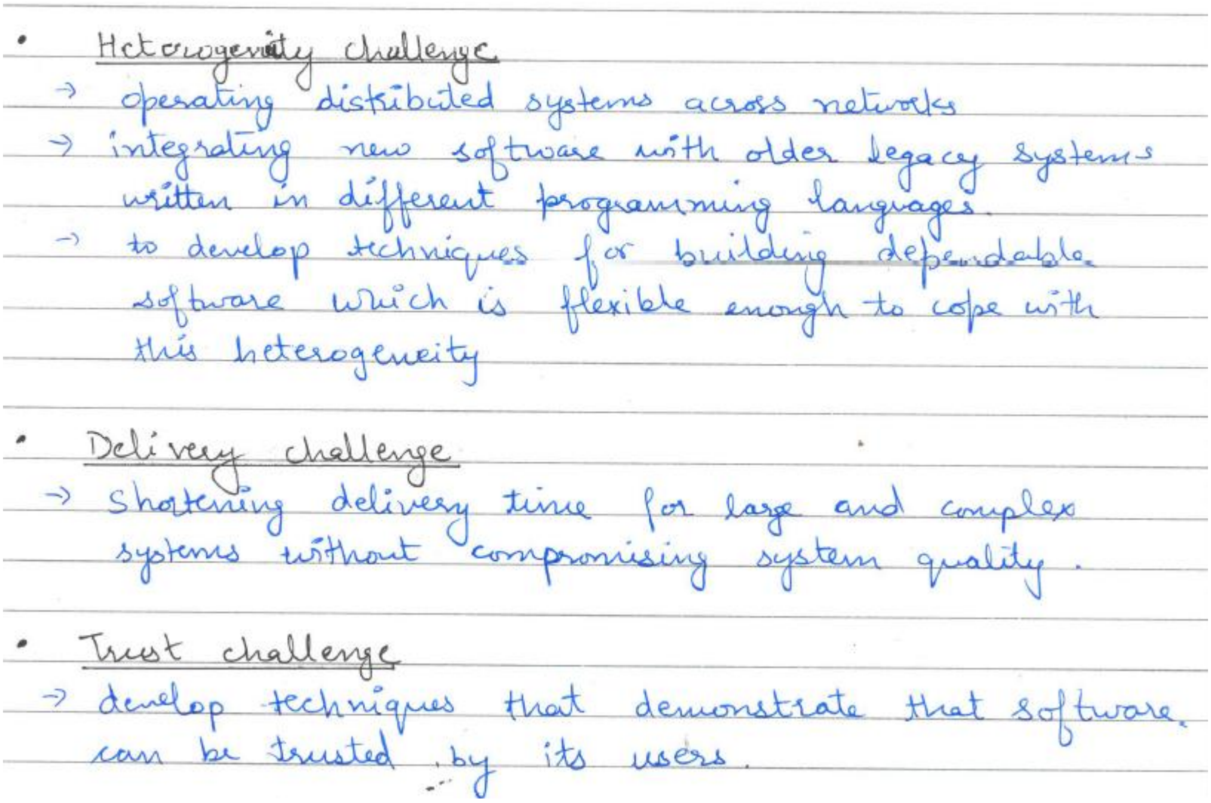
A software process is the set of activities and associated results that produce a software product. **(1M)**

Software Process model:

It is a simplified representation of a software process, presented from a specific perspective. **(1M)**

(b) Discuss the key challenges of software engineering.

(6M)



- Heterogeneity challenge

→ operating distributed systems across networks

→ integrating new software with older legacy systems written in different programming languages.

→ to develop techniques for building dependable software which is flexible enough to cope with this heterogeneity

- Delivery challenge

→ Shortening delivery time for large and complex systems without compromising system quality.

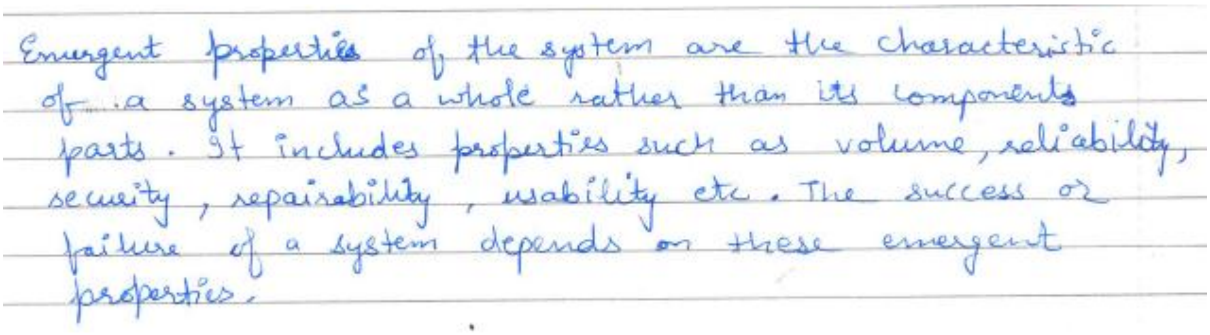
- Trust challenge

→ develop techniques that demonstrate that software can be trusted by its users.

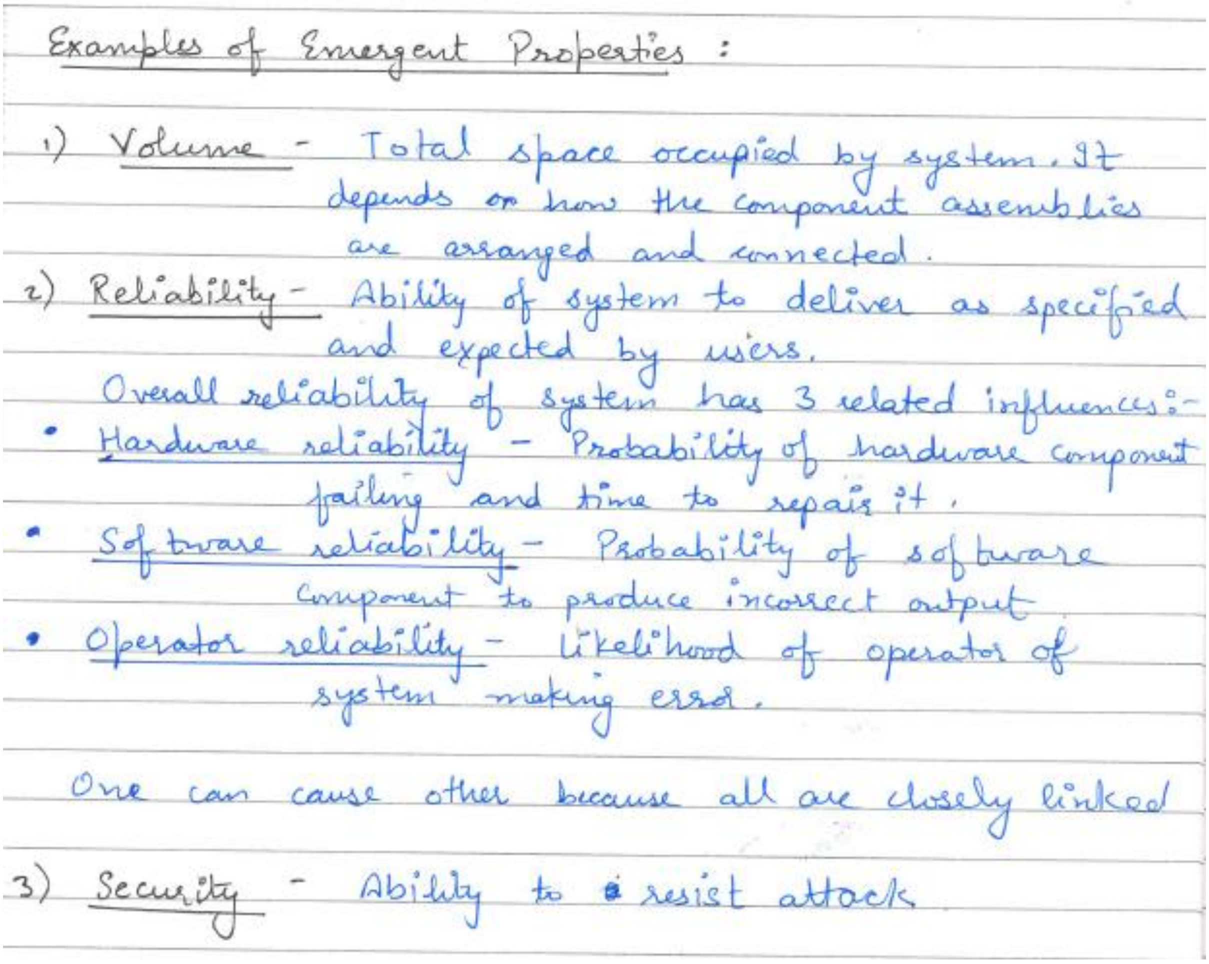
(6M)

2. Explain the emergent system properties with examples.

(10M)



Emergent properties of the system are the characteristic of a system as a whole rather than its components parts. It includes properties such as volume, reliability, security, repairability, usability etc. The success or failure of a system depends on these emergent properties.



Examples of Emergent Properties :

1) Volume - Total space occupied by system. It depends on how the component assemblies are arranged and connected.

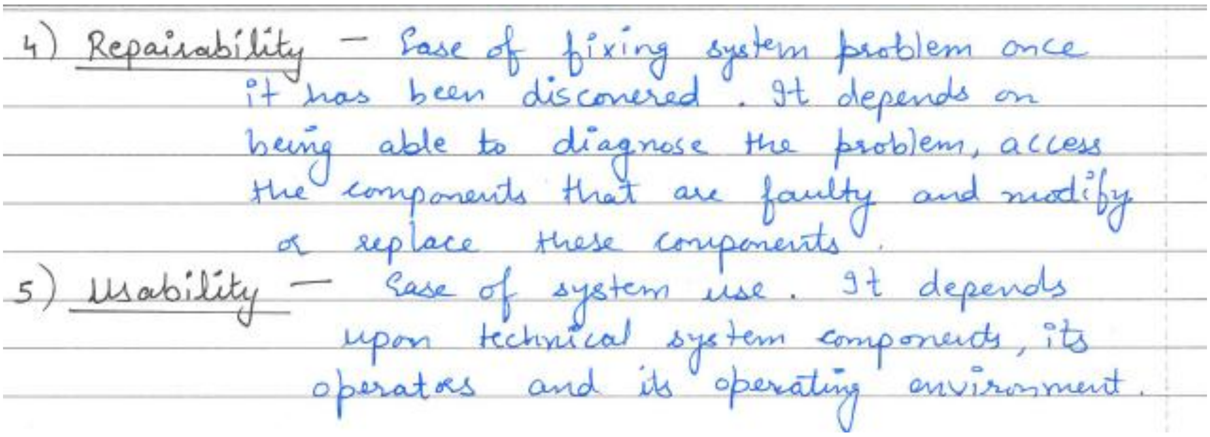
2) Reliability - Ability of system to deliver as specified and expected by users.

Overall reliability of system has 3 related influences:-

- Hardware reliability - Probability of hardware component failing and time to repair it.
- Software reliability - Probability of software component to produce incorrect output
- Operator reliability - Likelihood of operator of system making error.

One can cause other because all are closely linked

3) Security - Ability to resist attack



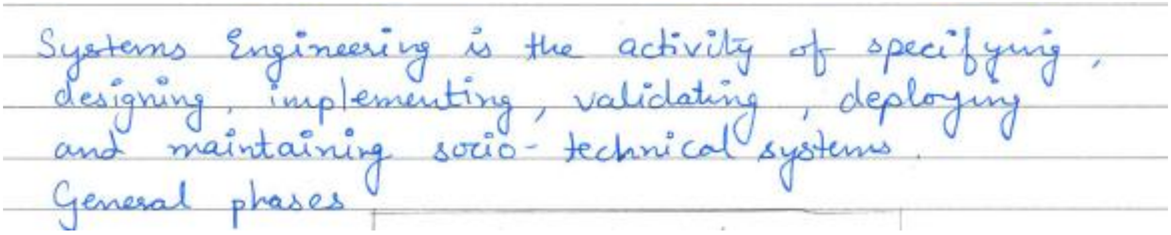
4) Repairability - Ease of fixing system problem once it has been discovered. It depends on being able to diagnose the problem, access the components that are faulty and modify or replace these components.

5) Usability - Ease of system use. It depends upon technical system components, its operators and its operating environment.

(10M)

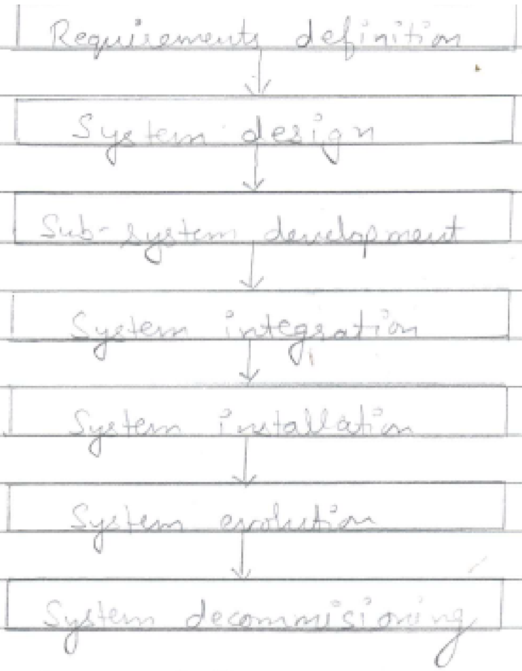
3. Explain system engineering process with a block diagram.

(10M)



Systems Engineering is the activity of specifying, designing, implementing, validating, deploying and maintaining socio-technical systems.

General phases



Systems Engineering process

1) System Requirements definition

- It specifies functions of the system, its essential and desirable system properties.
- It involves consulations with system customers and end-users.

2) System Design

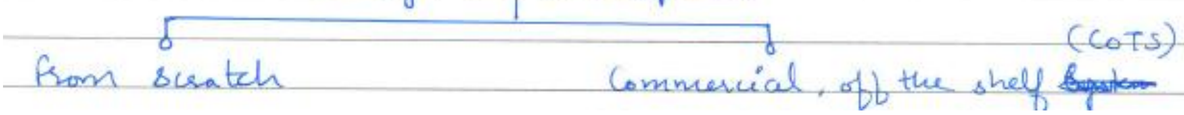
It specifies how the system functionality is to be provided by the components of the system.

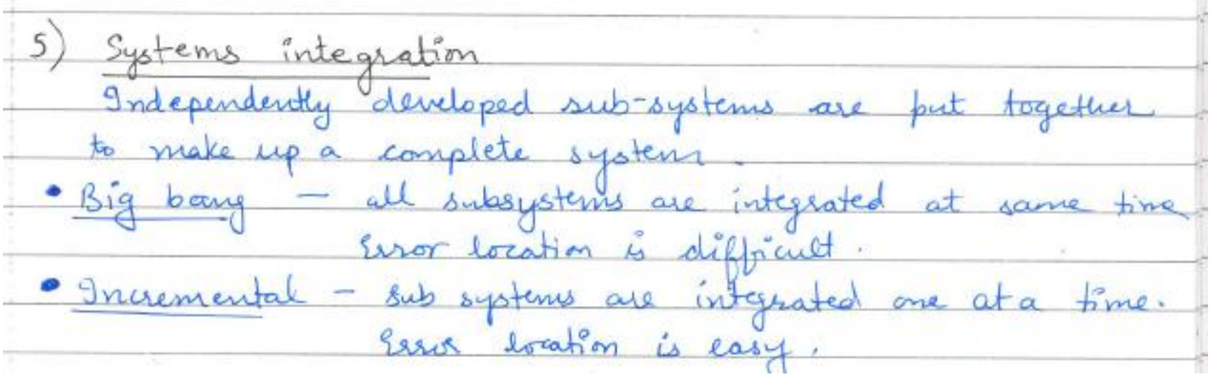
3) System modelling

Modelling system as a set of components and relationships between these components

4) Sub-system development

Sub-systems identified during system design are implemented

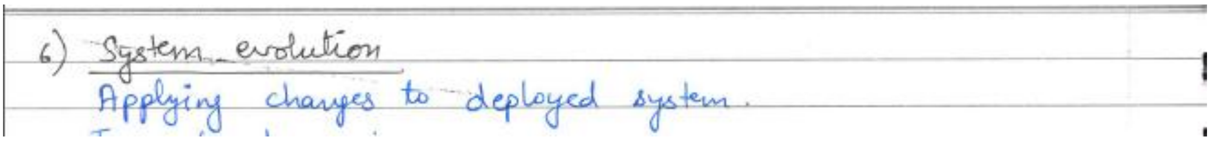




5) Systems integration

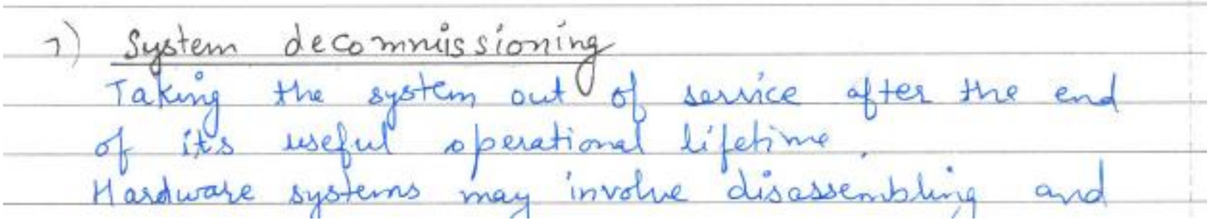
Independently developed sub-systems are put together to make up a complete system.

- Big bang — all subsystems are integrated at same time. Error location is difficult.
- Incremental — sub systems are integrated one at a time. Error location is easy.



6) System evolution

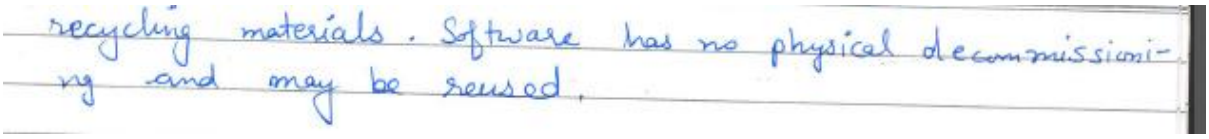
Applying changes to deployed system.



7) System decommissioning

Taking the system out of service after the end of it's useful operational lifetime.

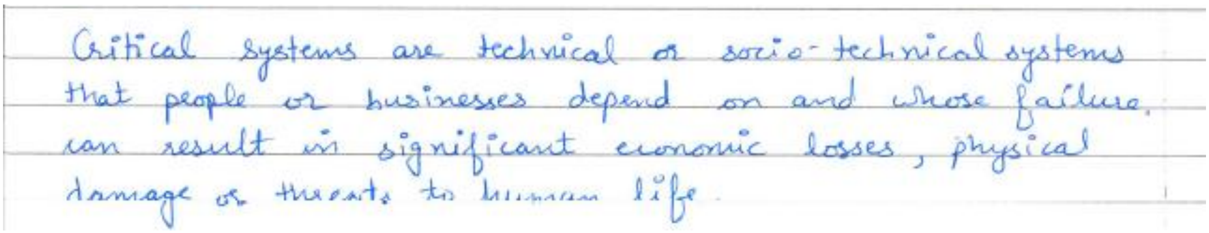
Hardware systems may involve disassembling and



recycling materials. Software has no physical decommissioning and may be reused.

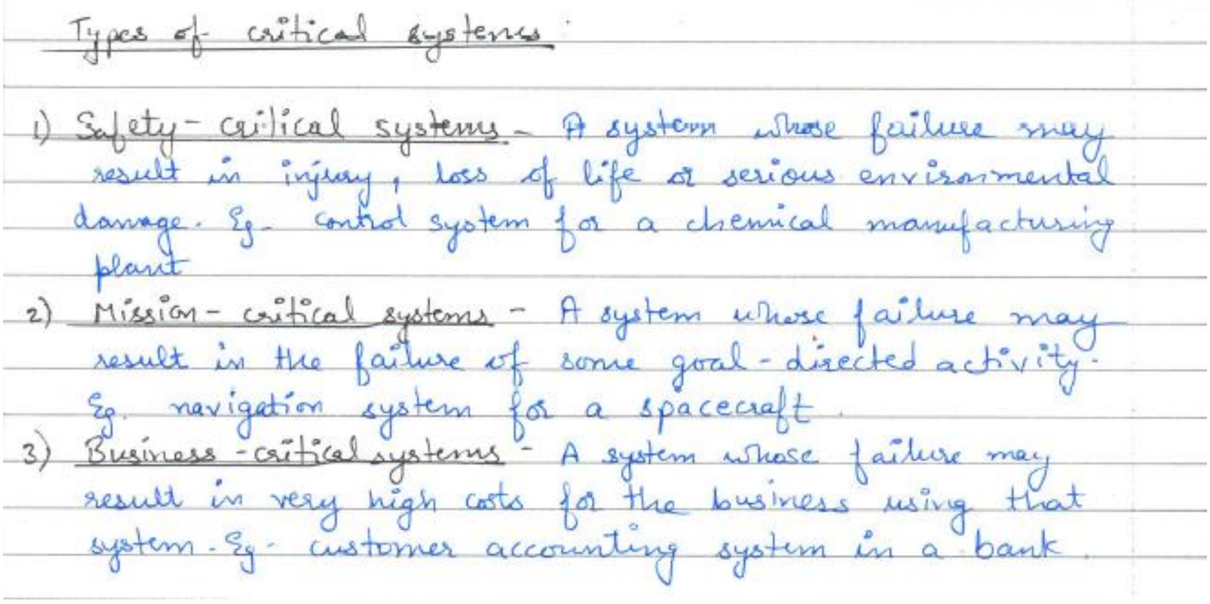
(10M)

4. (a) Define critical system and list out its types giving suitable example and justification. (6M)



Critical systems are technical or socio-technical systems that people or businesses depend on and whose failure can result in significant economic losses, physical damage or threats to human life.

(1.5M)



Types of critical systems:

1) Safety-critical systems - A system whose failure may result in injury, loss of life or serious environmental damage. Eg. control system for a chemical manufacturing plant

2) Mission-critical systems - A system whose failure may result in the failure of some goal-directed activity. Eg. navigation system for a spacecraft.

3) Business-critical systems - A system whose failure may result in very high costs for the business using that system. Eg. customer accounting system in a bank.

(1.5x3=4.5M)

(One mark for type definition and half mark for example)

(b) Explain why the perception of system reliability may vary between users.

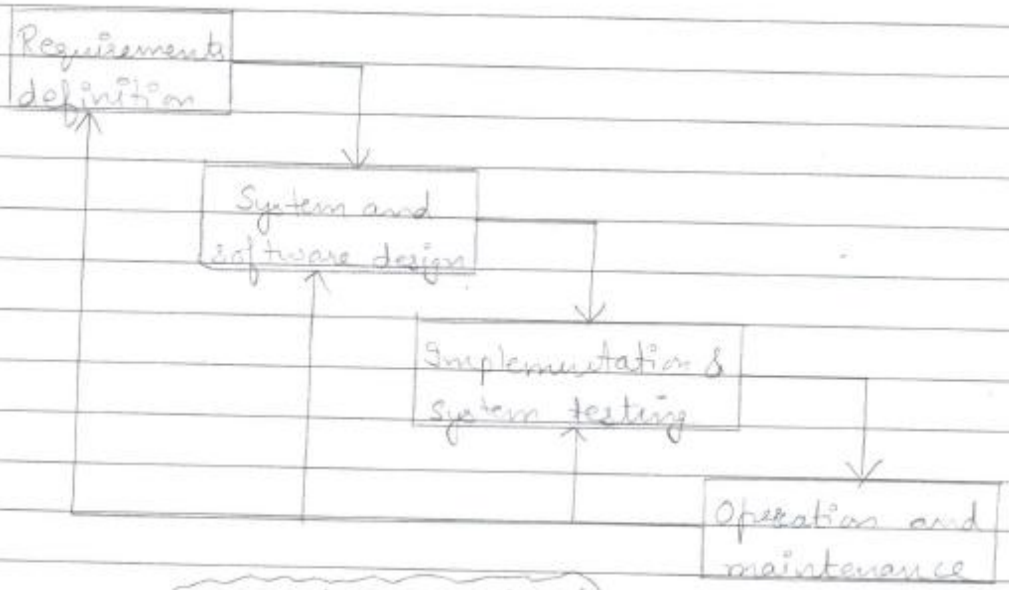
(4M)

The perception of system reliability may vary between users because perception of system reliability is influenced by the use of system and the consequences of the failure. For example, say a car has a fault in its windscreen wiper system that results in intermittent failures of the wipers to operate correctly in heavy rain. The reliability of that system as perceived by a driver depends on where they live and use the car. A driver in wet climate will probably be more affected by this failure than a driver in dry climate. The driver in wet climate may perceive the system to be unreliable whereas the driver in dry climate may never notice the problem and perceive the system to be reliable. Or, consider a fault with the engine management software which causes a car engine to cut out immediately after starting but operates correctly after a restart that corrects the initialization problem. This does not affect normal operation of the car, and many drivers would not think that the repair was needed. By contrast, most drivers would think that an engine that cuts out while they driving at high speed once per month (say) is both unreliable and unsafe and must be repaired. Hence, we can conclude that perception of reliability is subjective to the use of the system. **(4M)**

5. Explain the steps in waterfall model with a neat diagram. Discuss its advantages and disadvantages.

(10M)

1. The waterfall model


Requirements definition
System and software design
Implementation & system testing
Operation and maintenance

The waterfall model / Software life cycle

Because of the cascade from one phase to another, this model is known as the waterfall model of software life cycle.

(4M)

(i) Requirements analysis and definition – The system's services, constraints and goals are established by consulation with system users. They are then defined in detail and serve as a system specification.

(ii) System and software design – The system design process partitions the requirements to either hardware or software systems. It establishes an overall system architecture. Software design involves identifying and describing the fundamental software system abstractions and their relationships.

(iii) Implementation and unit testing – Software design is realised as a set of programs or program units. Unit testing involves verifying that each unit meets its specification.

(iv) 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.

(v) Operation and maintenance – The system is installed and put into practical use. Maintenance involves correcting errors which were not discovered in earlier stages of life cycle, improving the implementation of system units and enhancing the system's services as new requirements are discovered.

(3M)

Advantages

- Documentation.
- Sequential flow easy to understand.

Disadvantages

- Costs of producing and approving documents is high so iterations are costly and involve significant rework.

Therefore after a small number of iterations, it is normal to freeze parts of development.

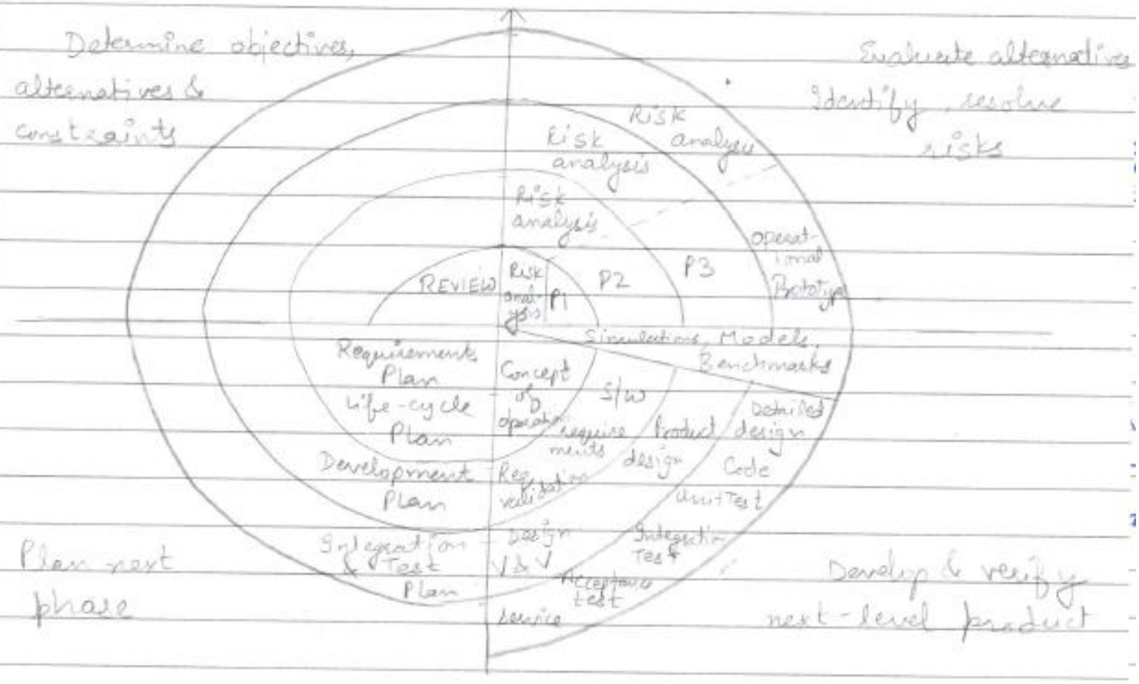
- Premature freezing may lead to badly structured systems.
- Partitioning of project into distinct stages is inflexible.
- Commitments must be made at an early stage in the process, which makes it difficult to respond to changing customer requirements.

(3M)

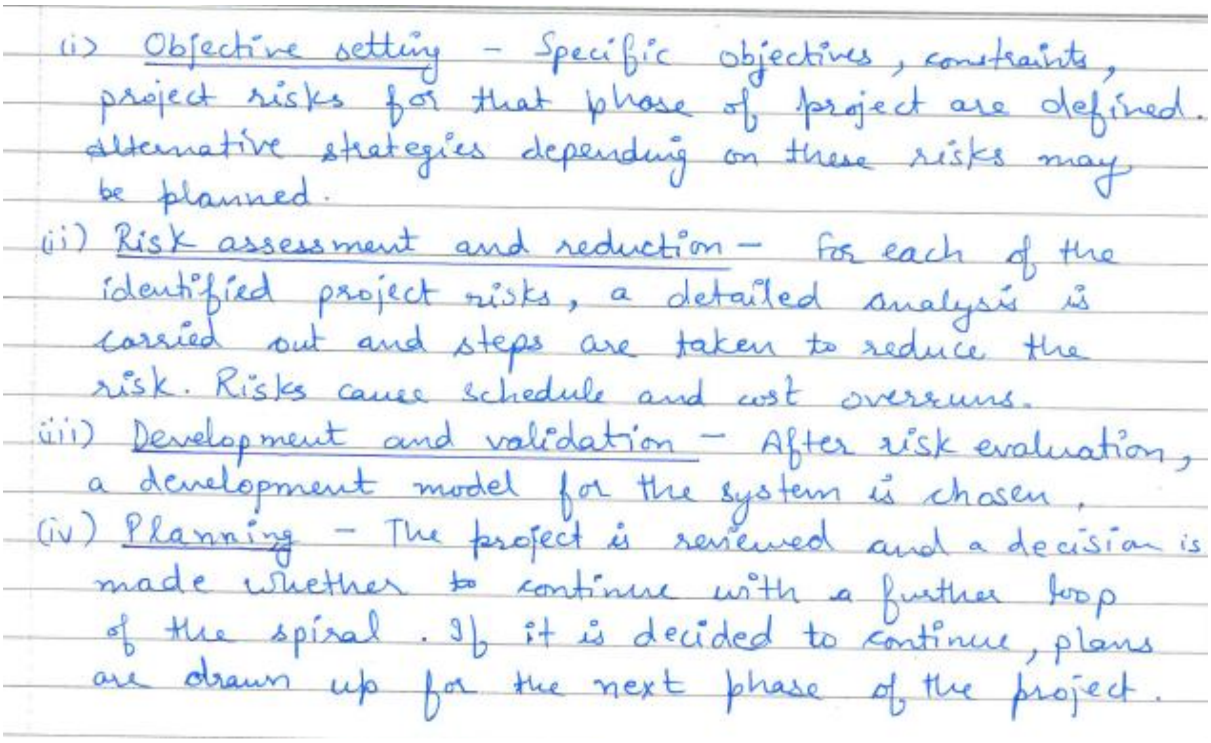
6. With a neat sketch to explain Boehm's spiral model of software development. Give its advantages. (10 M)

2. Spiral development

It was proposed by Boehm in 1988. In this, the process is represented as a spiral. Each loop in the spiral represents a phase of the software process.
Each loop in the spiral is split into four sectors:



Spiral model



(i) Objective setting - Specific objectives, constraints, project risks for that phase of project are defined. alternative strategies depending on these risks may be planned.

(ii) Risk assessment and reduction - For each of the identified project risks, a detailed analysis is carried out and steps are taken to reduce the risk. Risks cause schedule and cost overruns.

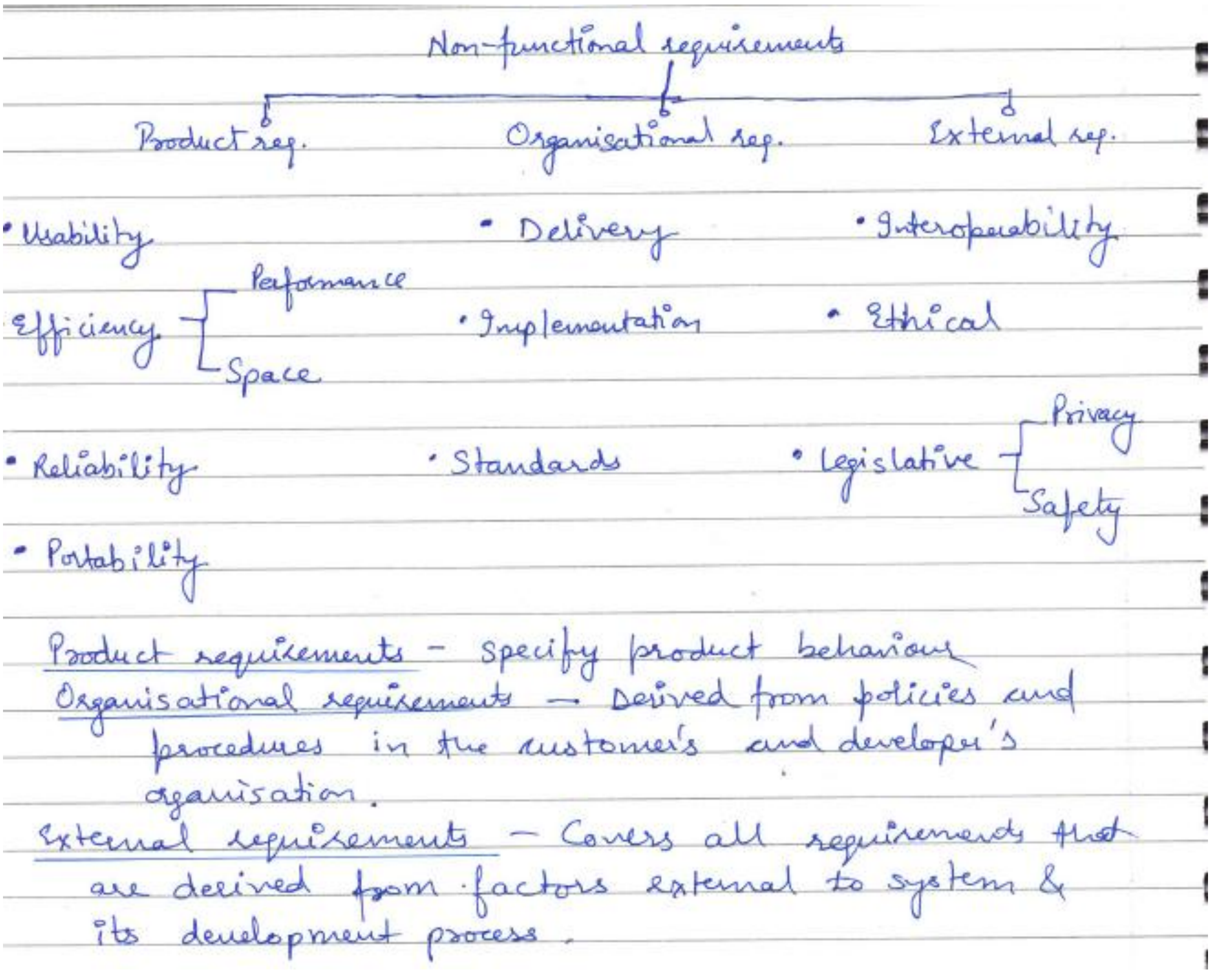
(iii) Development and validation - After risk evaluation, a development model for the system is chosen.

(iv) Planning - The project is reviewed and a decision is 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.

(10M)

7. Draw the hierarchy showing types of non-functional requirements. Explain giving appropriate examples.

(10M)



Non-functional requirements

Product req.
- Usability
- Efficiency – Performance, Space
- Reliability
- Portability

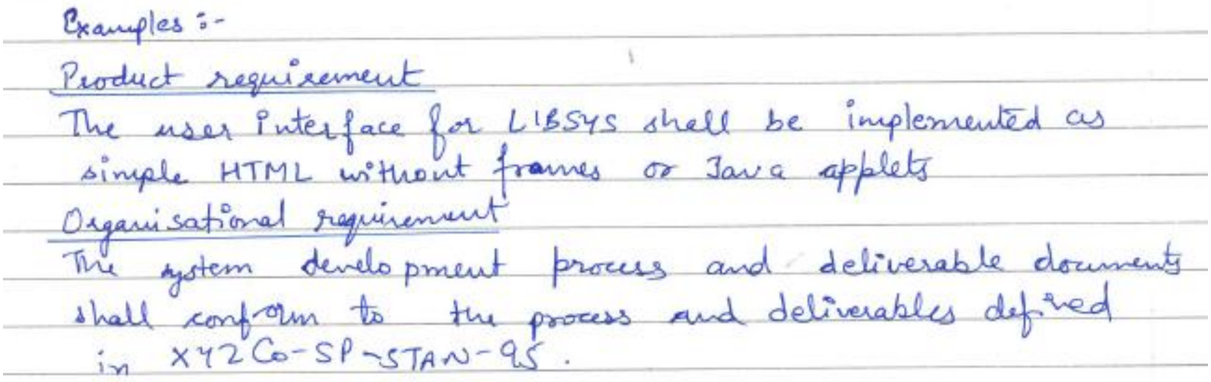
Organisational req.
- Delivery
- Implementation
- Standards

External req.
- Interoperability
- Ethical
- Legislative – Privacy, Safety

Product requirements - specify product behaviour

Organisational requirements - Derived from policies and procedures in the customer's and developer's organisation.

External requirements - Covers all requirements that are derived from factors external to system & its development process.



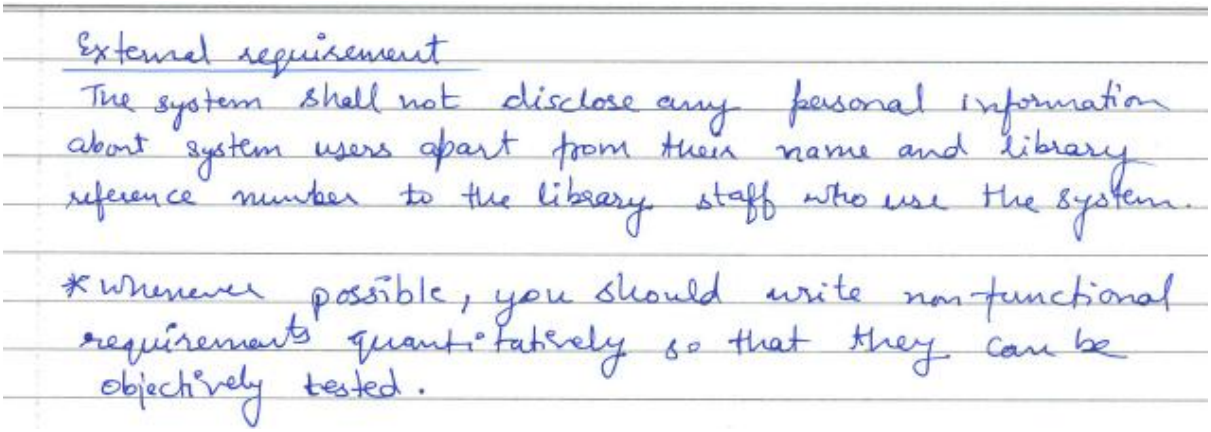
Examples :-

Product requirement

The user interface for LIBSYS shall be implemented as simple HTML without frames or Java applets

Organisational requirement

The system development process and deliverable documents shall conform to the process and deliverables defined in XYZCo-SP-STAN-95.



External requirement

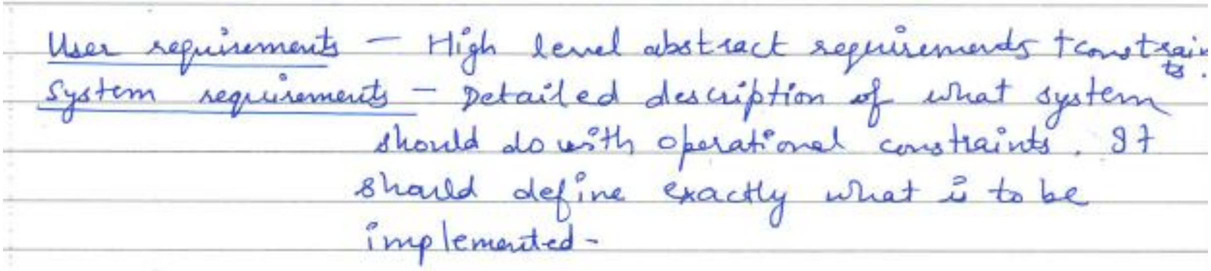
The system shall not disclose any personal information about system users apart from their name and library reference number to the library staff who use the system.

* Whenever possible, you should write non-functional requirements quantitatively so that they can be objectively tested.

(10M)

8. (a) Differentiate between **(2M+2M)**

i) System requirements and User requirements

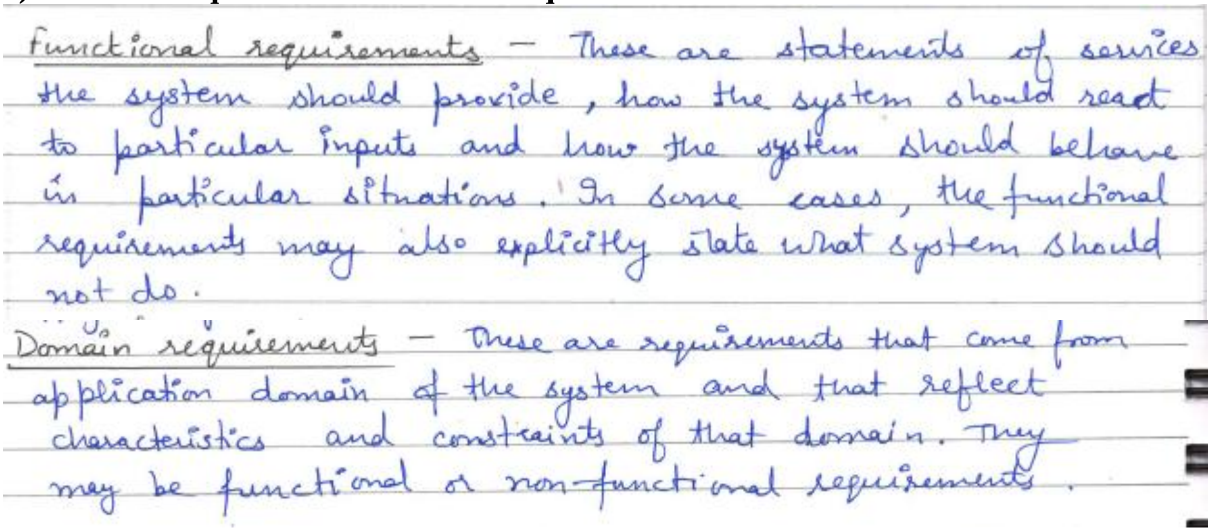


User requirements – High level abstract requirements + constraints.

System requirements – Detailed description of what system should do with operational constraints. It should define exactly what is to be implemented-

(2M)

ii) Functional requirements and Domain requirements.

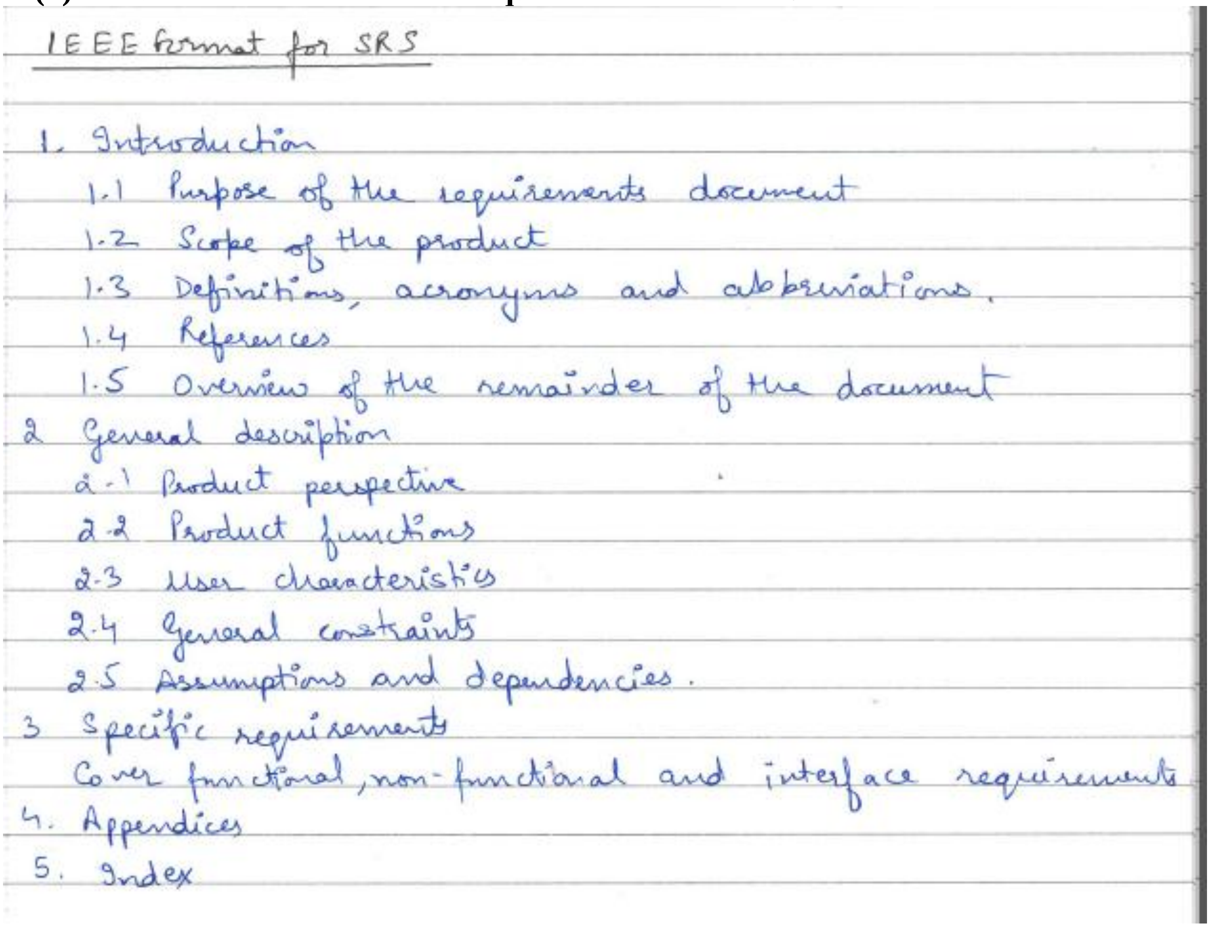


Functional requirements – These are statements of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situations. In some cases, the functional requirements may also explicitly state what system should not do.

Domain requirements – These are requirements that come from application domain of the system and that reflect characteristics and constraints of that domain. They may be functional or non-functional requirements.

(2M)

(b) Discuss the IEEE format for requirements document. **(6M)**



IEEE format for SRS

1. Introduction
   1.1 Purpose of the requirements document
   1.2 Scope of the product
   1.3 Definitions, acronyms and abbreviations.
   1.4 References
   1.5 Overview of the remainder of the document
2 General description
   2.1 Product perspective
   2.2 Product functions
   2.3 User characteristics
   2.4 General constraints
   2.5 Assumptions and dependencies.
3 Specific requirements
   Cover functional, non-functional and interface requirements
4. Appendices
5. Index

(6M)