Varun K A016 23-08-20

SE
Assignment-II

Q1. SOFTWARE

Software is a set of instructions, data or programs used to operate computers and execute specific tasks. Software is a generic term used to refer to application, scripts and programs that run on a device. This can be thought of a variable part of computer. In other words, its a computer program and associated documents. Software products may be developed for a particular customer or may be developed for a general market.

A good software should deliver the required functionality and performance to the user and should be maintainable dependable and usable.

- Software is developed or engineered, it is not manufactured in the classical sense.
- Software doesn't wear out.

SOFTWARE ENGINEERING

Software engineering is an engineering discipline that is concerned concerned with all aspects of software production. Its an application of principles used in the field of engineering, which usually deals with physical system, to design, development, testing, development and management of software system. Its fundamental activities are software specification, development, validation and evolution. Concerned with the practicalities of developing and delivering a useful software and is a part of a more general process. The key challenges faced by them are: Coping with increasing diversity, demands for reduced delivery times and

developing trustworthy software. In software engineering, roughly 60% of software cost are for development cost and the remaining 40% for testing. As of for a custom software, evolution cost often exceed development cost. Software engineering is typically used for large and intricate software system rather than single application or programs. Development is simply one phase of the process. The web has led to the availability of software services and the possibility of developing highly distributed service-based system which has helped a lot for software engineering.

SOFTWARE PROCESS

A software process is a structured set of activities that leads to the production of the software. Any software process must include the following 4 process:-

- Software Specification
- Design and Implementation
- Software Verification and Validation
- Evolution.

A software process model is an abstract representation of a process. It represents a description of a process from some particular perspective.

The various model are

- Waterfall Model
- Spiral Model
- Agile Model
- Iterative Model

These models are destraction of the process that can be used to explain different approaches to software development. They can be adapted and extended to create more specific process.

Q3 GENERIC SOFTWARE DEVELOPMENT PRODUCTS

The Generic software Development is a process executed by by the developers that develops the software product. Usually, this product is made for all types of business needs which has a positive demand in the market over a duration of time. Software development companies develop gendric software on their own and handled it to a group of customers having a similar need.

- Product developer own the specification.
- Developer can be able to change the specification due to some other external changes.
- Generic product user cannot control the evolution of the product.
- User can get application quickly.
- Generic products ~~user~~ are stand-alone system that are marketed and sold to any customer who wishes to buy them.
- Eg: MS Word, CAD software.

CUSTOM SOFTWARE DEVELOPMENT PRODUCTS

The Custom Software Development is a mechanism by which a company develops a product for a individual client. Individual client can be a company or a group of persons. This product mostly has a distant need in the market only for a limited time and is for the specialised business needs. Software development companies develop custom software at cost of particular customer.

- The customer own the specification and it also is controlled by him.
- Customer is also involved, therefore for changing the specification, both the customer and developer must take a decision before implementing it.
- The customer and developer will involve in the business process change and is implemented only if both are satisfied.
- Application is not available quickly and takes time for the development process.
- Eg: Traffic Monitoring System, ATC etc.

GENERIC	CUSTOMISED
- Done for developing a general purpose software.	Done to develop a software product as per the needs of particular customer.
- For development, the software developers has to depict the end user specification.	For development, the end user requirements can be aggregated by communicating them.
- Development team controls the process of generic software development.	Customer determines the process of software development.
- The software developed is economical.	Software developed is of high cost as the particular product is developed for the customer.

Q4. Some classes are 'critical system' where system failure may result in injury to people, damage of to the environment, or extensive economic losses. Examples of critical system include embedded system in medical devices, spacecraft navigation system and online money transfer system. Critical system are very expensive to develop. Not only must they be designed so that failures are very rare but they must also include recovery mechanism that are used if and when it fails.

SAFETY CRITICAL SYSTEM

Safety critical system deal with scenarios that may lead to loss of life, serious personal injury or damage to the natural environment. Eg: Safety Control system for chemical manufacturing plant, aircraft etc.

MISSION CRITICAL SYSTEM

Mission Critical System are made to avoid inability to complete the overall system, project objectives or one of the goals for which the system is designed. Eg: Navigation system for aircraft etc.

BUSINESS CRITICAL SYSTEM

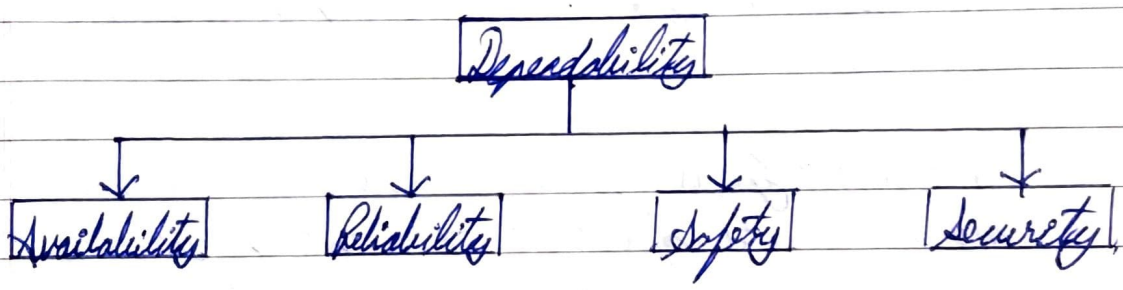
Business Critical System are programmed to avoid significant tangible or intangible economic cost. Eg: loss of business. This is often due to the interruption of service caused by the system being unusable. Eg: Stock trading system.

Q6. System Dependability

For many software intensive systems, the most important system property is the dependability of the system. The dependability of a system reflects the extent of the users confidence that it will not corrupt data or other system and will not fail in normal use. Users mostly trust a system that is dependable.

Importance of System Dependability

- System failures may have widespread effects with large number of people effected by failure.
- Systems that are not dependable and are unreliable, unsafe or insecure may be rejected by their users.
- The cost of system failure may be very high if the failure leads to economic losses or physical damage.
- Undependable systems may cause information loss with a high consequent recovery ~~loss~~ cost.



The 4 Principles Dimensions

- Availability: The availability of system is the probability that it will be up and running and to deliver useful services to user at any given time.
- Reliability: The reliability of system is the probability ~~that~~ it will over a given period of time, that the system will correctly deliver services as expected by the user.

- Safety: The safety of system is a judgement of how likely it is that the system will cause damage to people or its environment.
- Security: The security of system is a judgement of how likely the system can resist accidental or deliberate intrusions.

Dependability Properties

- Repairability: System failures are inevitable, but the disruption caused by failure can be minimized if the system can be repaired quickly. For that to happen, it must be possible to diagonize the problem, access the component that has failed, and make changes to fix that component. Repairability in software is enhanced when the organization using the system has access to source code and has skills to make change in it. Open source software makes this easier but the reuse of component make it more difficult.

- Maintainability: As systems are used, new requirements emerge and its important to maintain usefulness of system by changing it to accomodate those new requirements. Maintainable software is software that can be adapted economically to cope up with the new requirements, and where there is low probability that making changes will introduce new errors in the system.

- Survivability: A very important internet-based systems is survivability. Survivability is attribute of system to continue the deliver services while it is under the attack and potentially the part of system is disabled. Three strategies are used to enhance survivability:
  a. Resistance to Attack.
  b. Attack Recognition.
  c. Recovery of damage caused by an attack.

- Error Tolerance: This is a property considered as a part of usability and reflects the extent to which the system has been designed so that user input errors are avoided and tolerated.

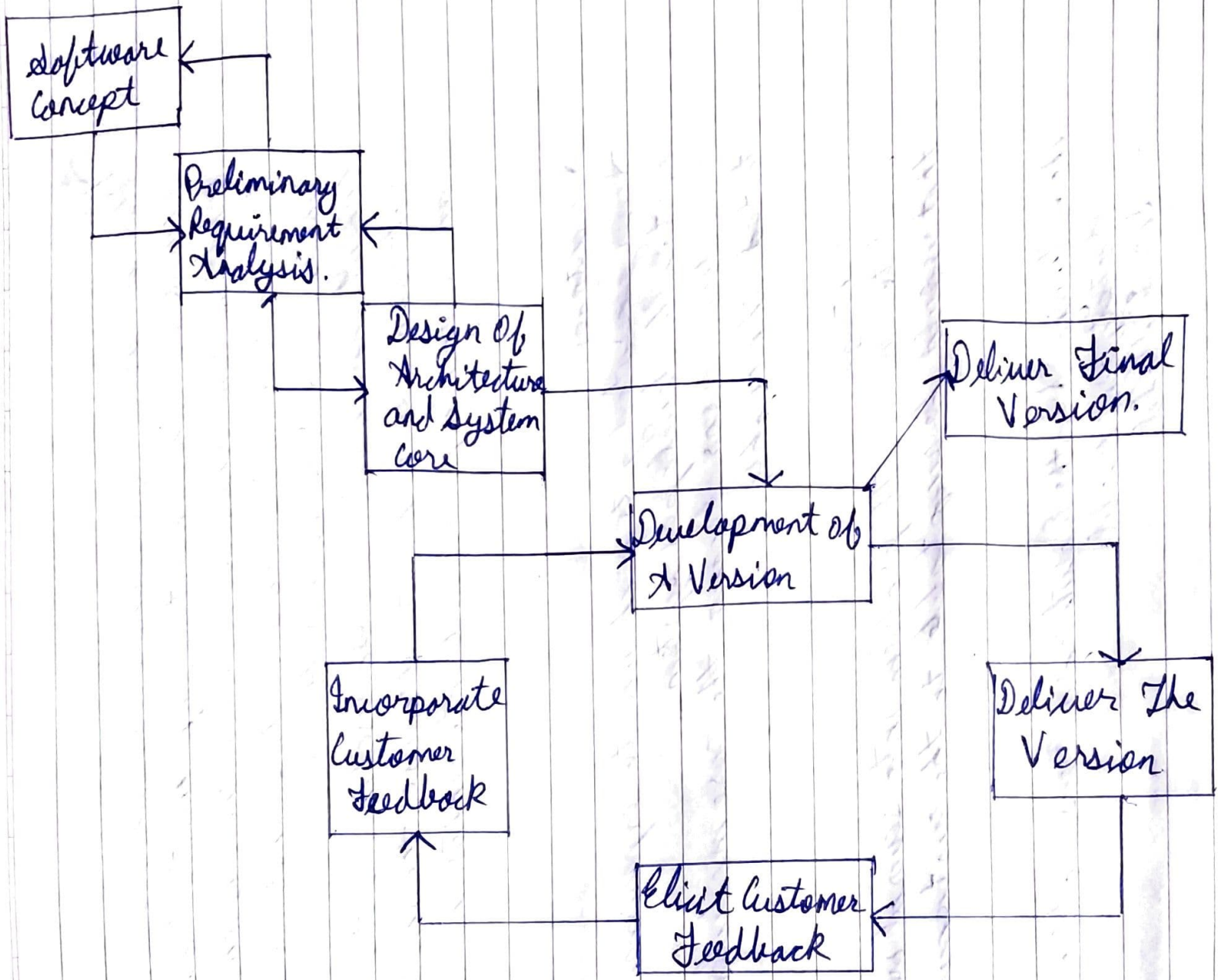
Hence, dependability is a composite system property that reflects the degree of trust users have in a system. It includes availability, reliability, safety and security the 4 properties.

Dependability is a subjective and depends on the expectation and judgement of system users

Q5. Software Process is a coherent set of activities for specifying, designing, implementing and testing software system. A software process model is an abstract representation of a process that represents a description of a process from a some particular perspective. There are many different software processes but all involve:
- Specification - defining what the system should do.
- Design and Implementation - Defining the organization of the system and implementing the system.
- Validation - Checking that it does what the customer wants.
- Evolution - Changing the system in response to changing customer needs.

EVOLUTIONARY MODEL

The evolutionary model is a combination of Iterative and Incremental model of software development life cycle. Delivering your system in a big bang release, delivering it in incremental process over time is the action done in this model. Some initial requirements and architecture envisioning need to be done.
Evolutionary model suggests breaking down of work into small chunks, prioritizing them and then delivering them to customer one by one. The number of chunks is large and is the number of ~~activities~~ deliveries made to the customer. The main advantage is that the customers confidence increases as he gets constantly quantifiable goods from the beginning of project to verify and validates its requirements. The model allows for changing requirement as well as all work broken down into maintainable chunks.


Software Concept
Preliminary Requirement Analysis.
Design Of Architecture and System core
Development of A Version
Deliver Final Version.
Deliver The Version
Elicit Customer Feedback
Incorporate Customer Feedback

Page No.
Date

Application

- Used in large projects where you can easily find modules for incremental implementation. This model is commonly used when the customer wants to start using the core features instead of waiting for the full software.
- Evolutionary model is also used in object oriented software development because the system can be easily partitioned into units in terms of objects.

Advantages

- A user ~~gets~~ gets a chance to experiment a partially developed system.
- It reduces error because the core modules get tested thoroughly.

Disadvantages

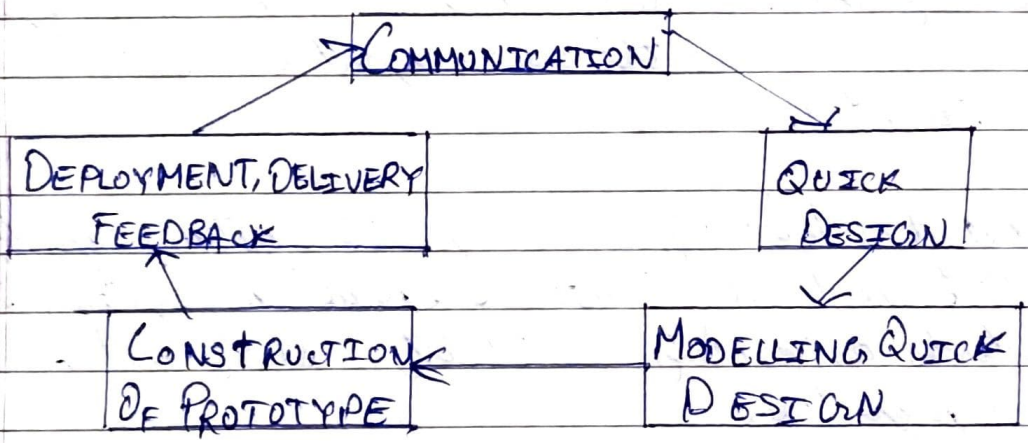
- Hard to divide the problem into several versions that would be acceptable to the customer.

The Following Are the Evolutionary Process Model:

- Prototyping Model
- Spiral Model

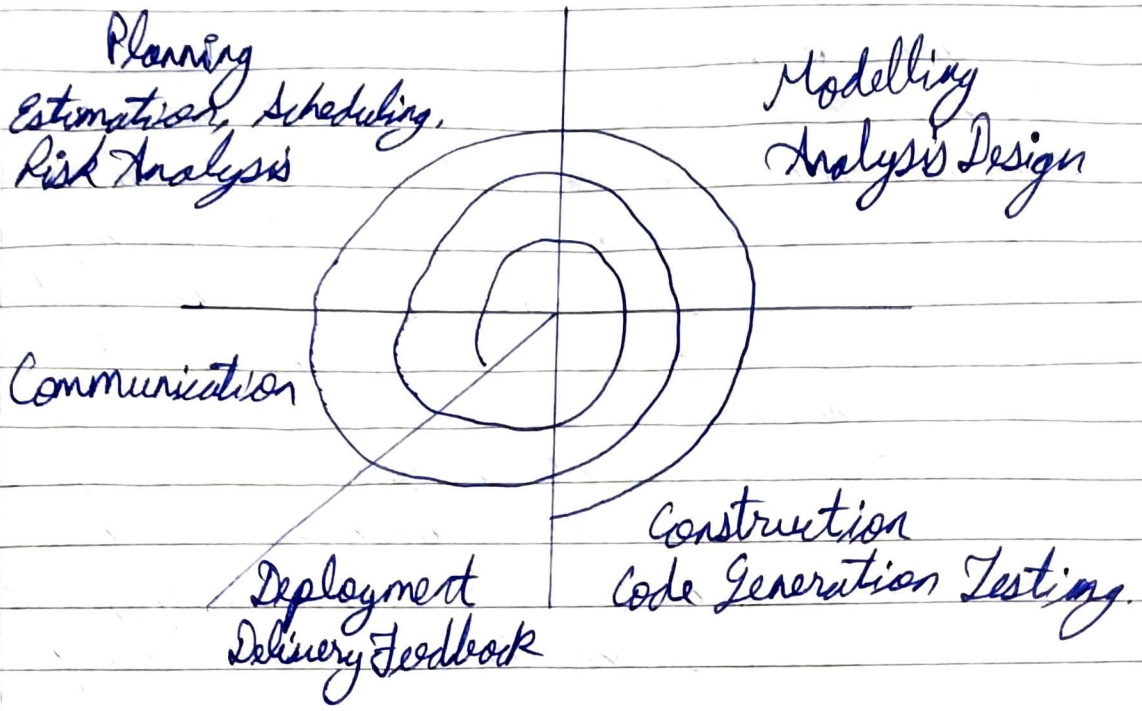
1. Prototyping Model

- Prototype is defined as first or preliminary form using which other forms are copied or derived.
- Prototype model is a set of general objectives for software.
- It doesn't identify the requirements like detailed input, output.
- It is software working model of limited functionality.
- Working programs are quickly produced.



2. Spiral Model

- Its a risk driven process model.
- Used for generating software projects.
- An alternate soln is provided if risk is found in risk analysis.
- Its a combination of prototype and waterfall model.
- In one iteration all activities are done. for large projects the output is small.



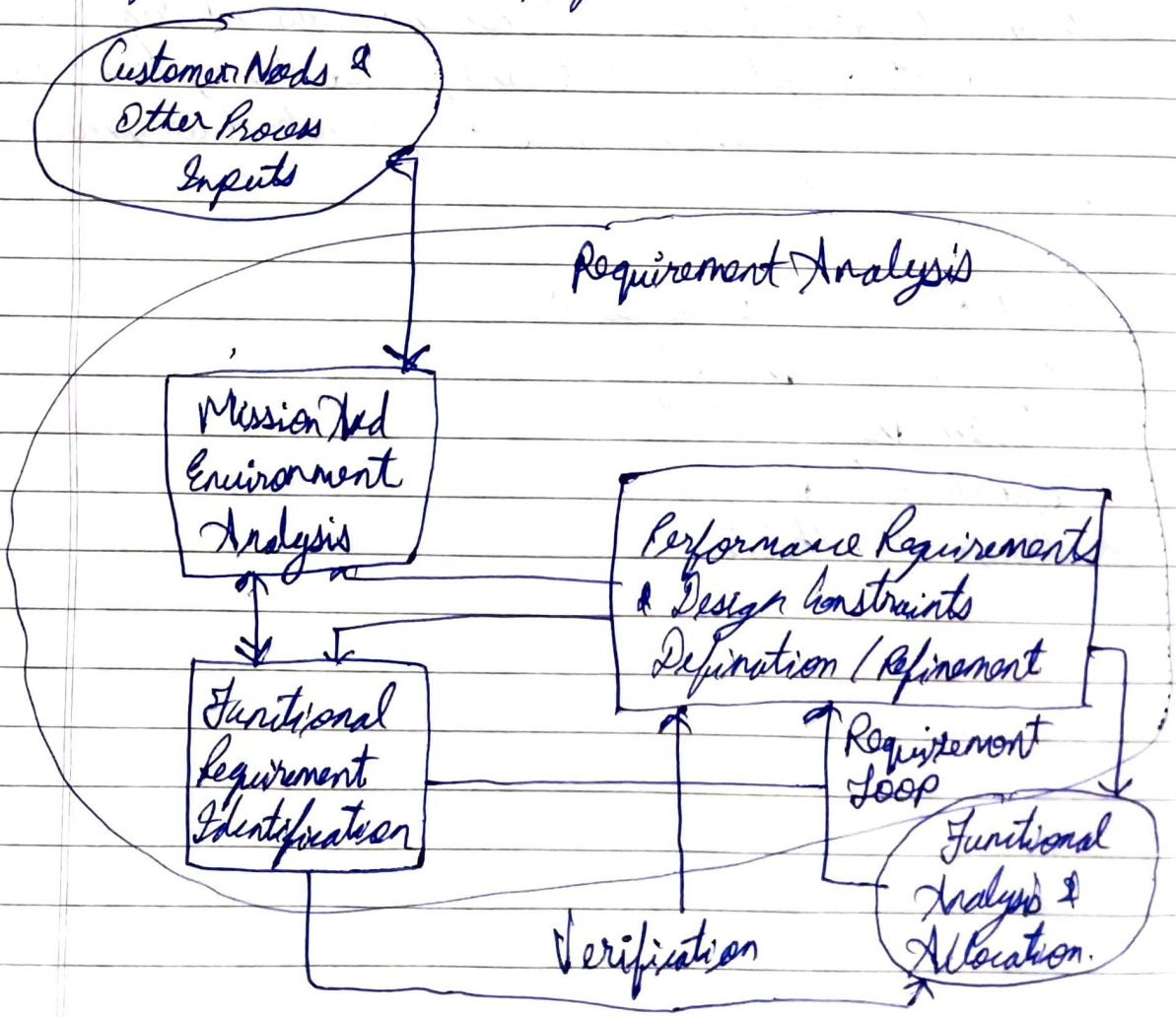
Q2. System Engineering Process is a comprehensive, iterative and recursive problem solving process, applied sequentially top-down integrated teams. It transforms needs and requirements into a set of system products and process description, generate information for decision makers and provide development for next gen. input.

The 4 Phases of SE Process are:
- Requirement Analysis
- System Analysis Control
- Functional Analysis
- Design Synthesis.

Requirement Analysis

This is one of the first activities of SEP and functions somewhat as an interface between the internal activities and the external sources providing input to system process. It examines, evaluates and translates the external input into a set of functional and performance requirements that are basis of Functional Analysis and Allocation.

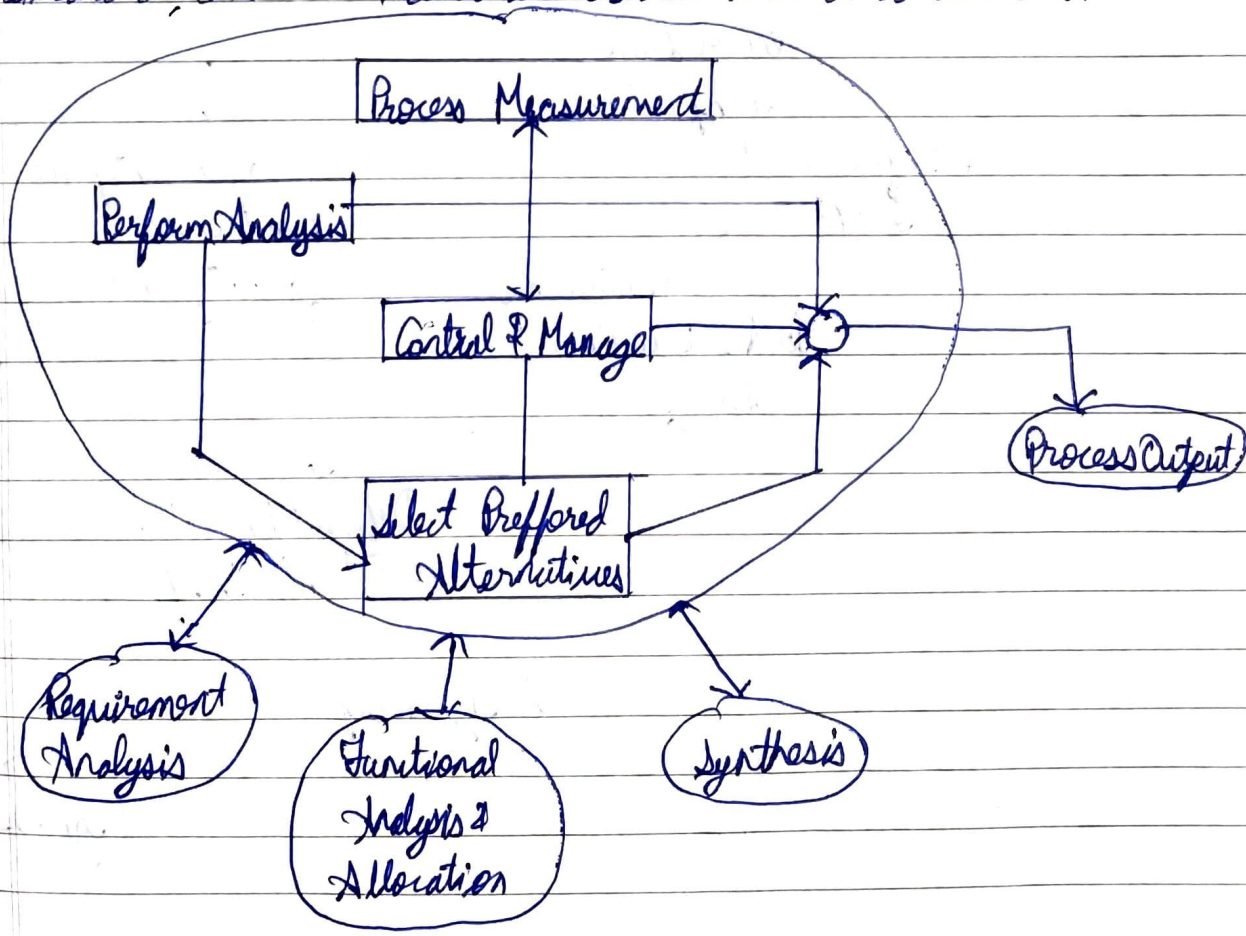
The output of Requirement Analysis is a set of top level functional definitions and accompanying performance and design requirements which become starting point of functional analysis and allocation. The requirements loop serve to refine the performance requirements and initiate requirement resolution to determine how firm the requirements are for items that prove to be major cost, schedule, performance and or risk drivers.



System Analysis And Control

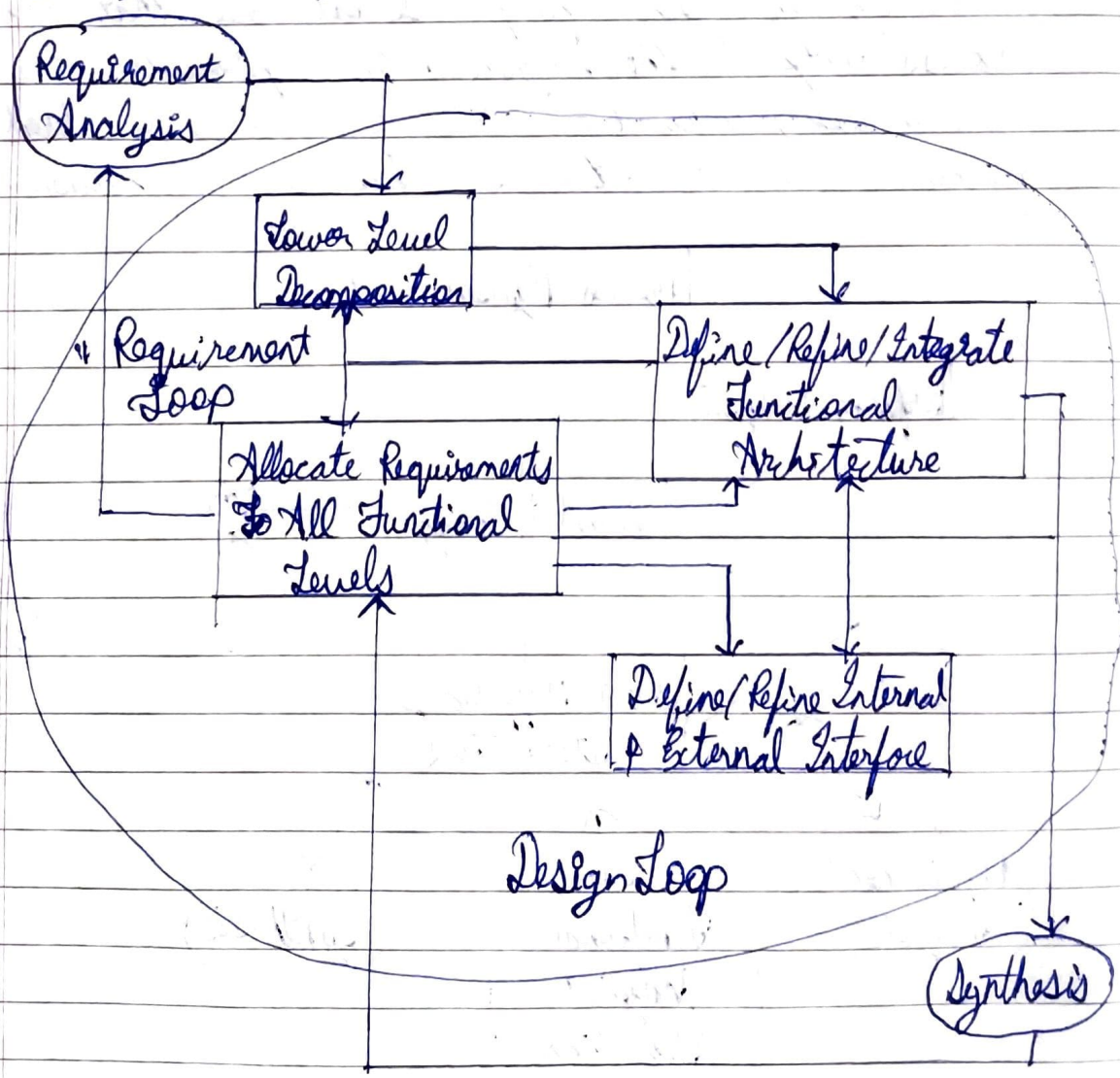
System analysis and control is controls the overall SEP. This activity identifies the work to be performed and develop the schedule and cost estimate for effort. Coordinates all activities and assures that all are operating from same set of requirement, agreement and design iteration. Its the center for configuration management throughout the SEP. It interacts with all the other activities of the SEP. It evaluates the output of the other activities and conducts independent studies to determine which of the alternate approaches is best suited for the application.

The information from System Analysis And Control is a major part of system engineering process database that forms the process output. The analysis activity provides the results of all analysis performed, identifies approaches, considered and discarded, and the rationale used to reach all conclusions.



Functional Analysis

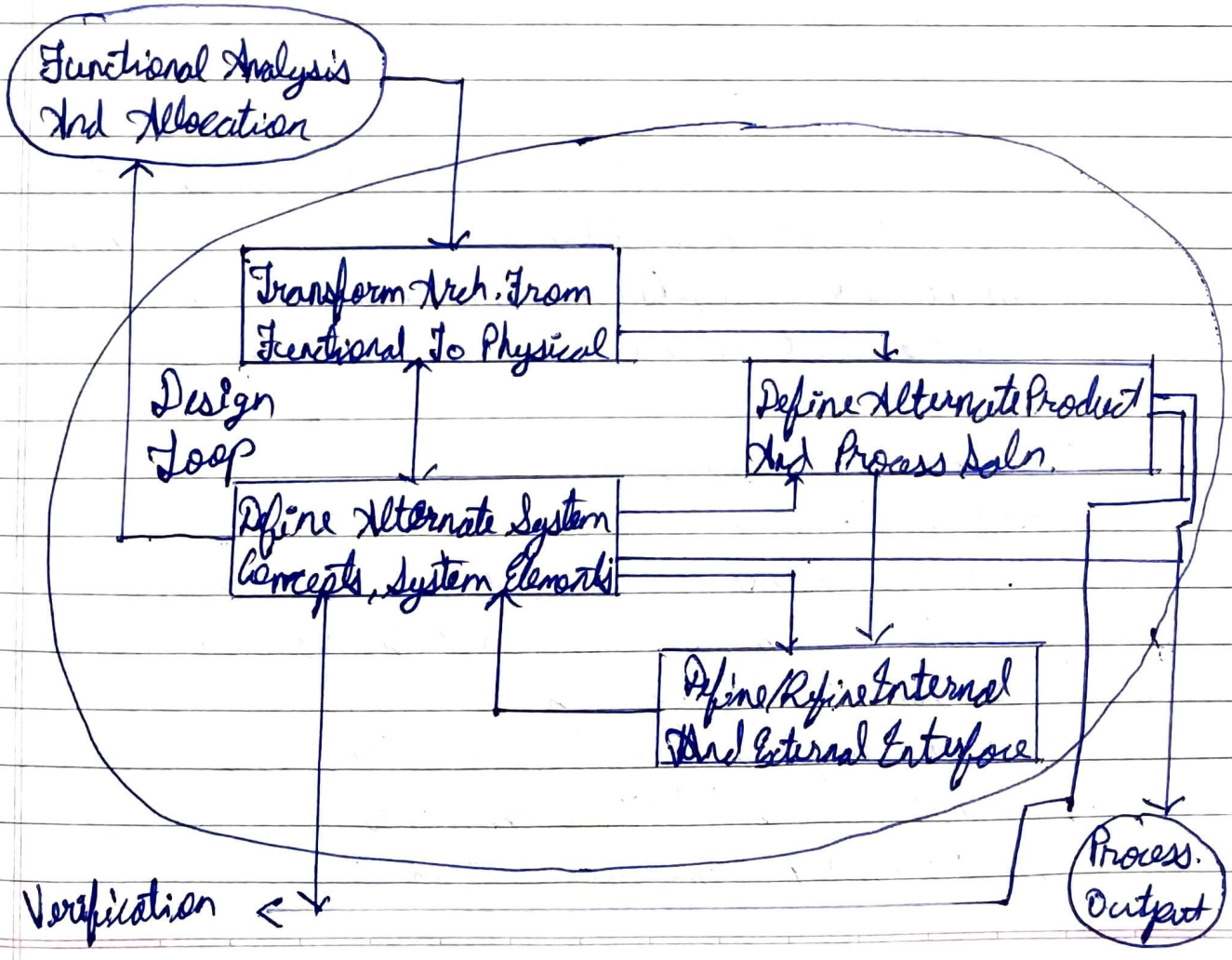
It is a top-down process of translating system level requirements into detailed functional and performance design criteria. The result of the process is defined Functional Architecture with allocated system requirements that are traceable to each system function.
This bridges a gap between the high level set of system requirements and constraints and the detailed set required to develop or purchase systems and implementing programs.
Its an integral part of Requirement & Design Loop. It is an iterative process, interacting and reacting to the ongoing activities in both the loops.



Design Synthesis

Design Synthesis is process of taken the funtional architecture developed in the Funtional Analysis and Allocation step and decomposing these fuction into a physical arch. that satisfy required functions.

Synthesis is the process whereby the functional architecture and their associated requirements are translated into physical arch. and one or more physical sets of hardware, software and personal solutions. It is the output end of the Design Loop. As the design are formulated, their characteristics are compared to the original requirements, developed at beginning of the process to verify the fit. The output of this activity is a set of analysis-verified specification which describe a balanced, integrated system meeting the requirements and a database which documents the process and rationale used to establish these spec.



Q7. Requirement Engineering Process is the process to gather the software requirements from client, analyze and document them. The goal of requirement engineering is to develop and maintain sophisticated and descriptive 'System Requirement Specification' document. This process may include 4 high level activities. These focus on assessing if the system is useful to the business, discovering requirements into some standard form and checking that the requirements actually define the system that customer wants.

- Feasibility Study
It is focused towards the goal of the organization. This study analyses whether software product can be practically materialized in terms of implementation, contribution of project to organization, cost constraints and as per core values and objectives of the organization.

- Requirement Gathering
If the feasibility report is positive towards undertaking the project, next phase starts with gathering requirements from the user. Analysts and engineering communicate with the client and end user gather information.

- Software Requirement Specification
SRS should come up with following features:
  - User requirements are expressed in natural language.
  - Technical requirements are expressed in structured language, used inside the organization.
  - Design description should be written in Pseudo Code
  - Format of forms & GUI screen prints
  - Conditional & Mathematical notation for DFD's

- Software Requirement Validation
Requirements can be checked against following conditions:
- If they can be practically implemented.
- If they are valid and as functionality and domain of software.
- If there are any ambiguities
- If they are complete
- If they can be demonstrated.

IEEE Standards for Requirement Documents
The most widely known requirement documents standard is IEEE (1998). This IEE standards suggest the following structure of requirement documents.

1. INTRODUCTION
   1.1 Purpose Of Requirement Document
   1.2 Scope Of Product
   1.3 Defination, acronyms & abbreviation
   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

4. APPENDICES

5. INDEX

Although IEEE standard is not ideal, it contains a great deal of good advice on how to write requirements and how to write requirements and how to avoid problems. It is too general to be an organizational standard in its own right. It is a general framework that can be tailored and adapted to define a standard geared to the needs of a particular organization