SOFTWARE ENGINEERING UNIT: 2

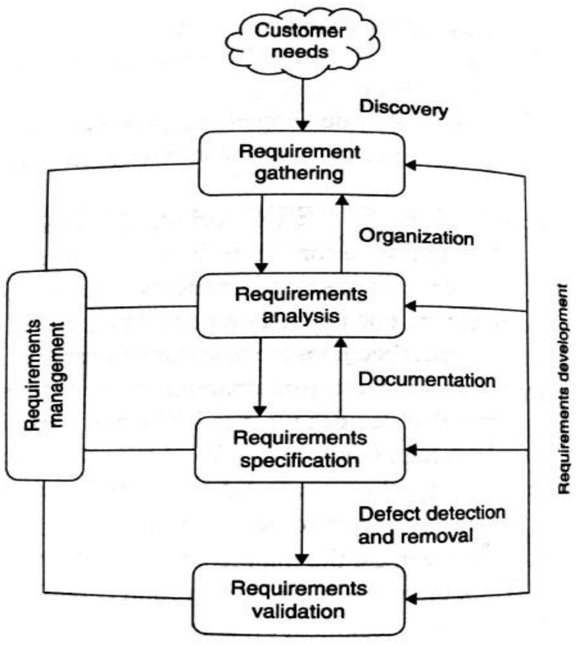
Requirements Analysis and Specification: Requirements Gathering and Analysis, Software Requirement Specification (SRS), Formal System Specification.

PART 1: REQUIREMENTS ANALYSIS AND SPECIFICATION

REQUIREMENTS GATHERING AND ANALYSIS

Requirement Engineering Process:

Requirement Engineering is the disciplined, process-oriented approach to the **development** and **management** of requirements. Development of requirements focuses on preparing validated requirement that are used throughout software development which includes activities such as elicitation, analysis, specification and validation of requirement. Requirement management is concerned with managing requirements that change dynamically, controlling the baseline requirements, monitoring the commitments and consistency of requirements throughout software developments.



In requirement engineering process, the **Requirement elicitation** phase aims to gather requirements from different perspectives to understand the customer needs in real world scenario. **Requirement analyst** concentrates on causes and effects. The **Requirement Specification** is used to document the requirements that can be communicated to stakeholders. Finally **Requirements Validation** activity ensures that all requirements are accurately specified, completed and satisfy the customer needs.

Requirement Gathering:

Requirement Gathering or Elicitation is an activity that helps us to understand what problem has to be solved and what customers expect from the software. As clients or customers may not able to clearly define their problems or requirements.

Detailed understanding of problem is possible by understanding the business area for which the solution is being developed. Here the system analyst plans an important role who involves the domain expert, software engineer, clients, sponsors, managers, vendors, suppliers, and other stakeholders and follows standards and guidelines to elicit requirements.

Some techniques that help in gathering information are:

- Interviews
- Questionnaires
- Studying existing system documents
- Analyzing business data.

The main challenges of requirement elicitation are identification of problem scope, identification of stakeholders, understanding of problem, and volatility of requirements.

Requirement Analysis:

Requirement Analysis includes various activities such as classification, organization, prioritization, negotiation and modeling requirements. During requirement analysis models are prepared to analyze the requirements the requirements.

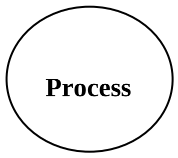
The following techniques are used to model the requirements -

❖ **Structured Analysis**

Dataflow diagram:

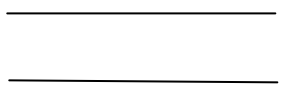
It is used to understand the working processor of system .It represents the behavior of the system

Process-It denotes transformations of the input data to produce the output data . Output is again used as input.



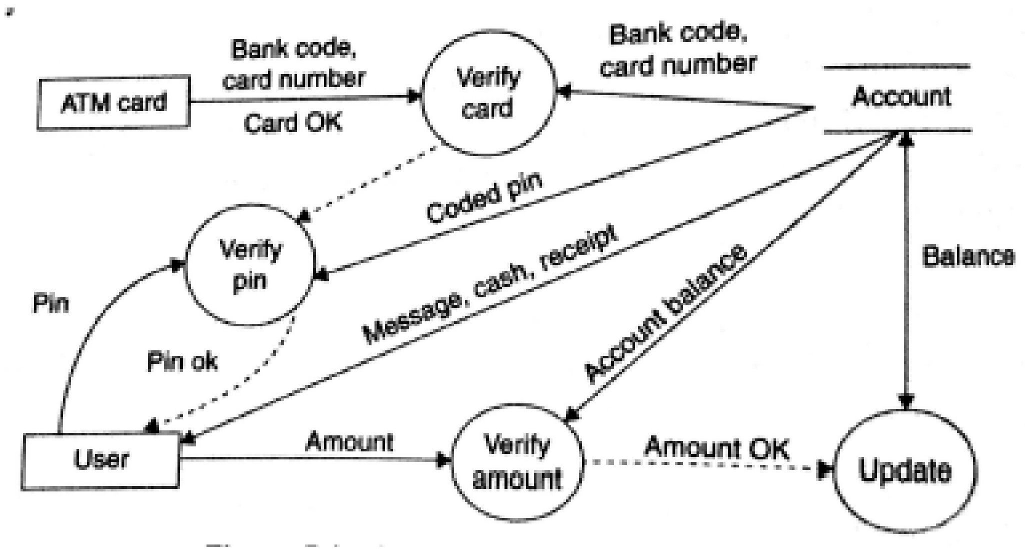
Dataflow –It represents movement of data.

Data Store – Data in rest where the incoming arrows represent updating data and outgoing arrow represents retrieval of data from database



Actor-It indicates people, organization or another system consumes data.

Finally dataflow diagram represents the dataflow between source, destination, process and data store .



<u>Data Dictionary:</u>

It is metadata which describes another data about another data. Construction of the DFD starts with the high level functionality of the system. Data Flow Diagram represents data processing in a system. The process at level 1 can be numbered as 1,2,3 …and so on. The sub process at level 2 is the decomposed sub processes of level 1.

Data flow diagrams are different from flow charts Data Dictionary is metadata which describes the composite data structures defined in DFD.

Data Dictionary is written using special symbols:

'+' for composition

'-' for repetition

'1' for selection

The combination of repeated data is written using '[ ]' symbol.

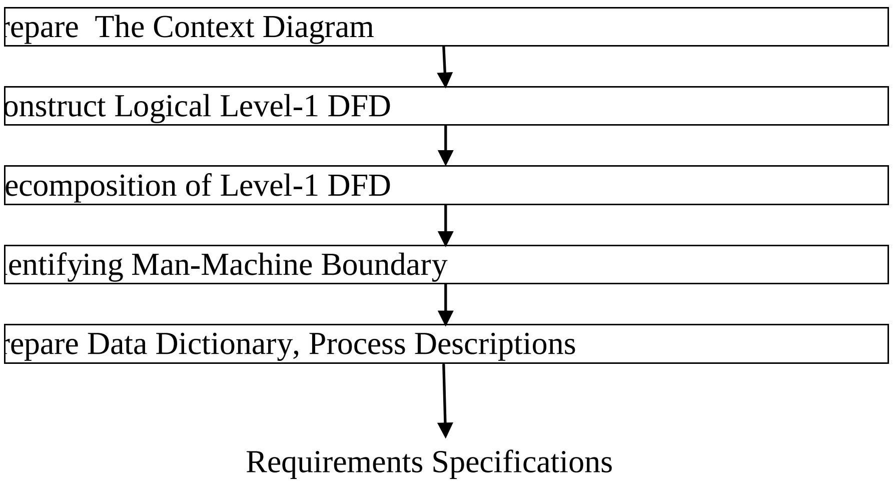
Eg: restaurant bill =dish price +sales tax +service charge + grand total.

Dish price =[dish name +quantity +price]

Seat charge =[reserved table /common table]

<u>Structured analysis method:</u>

It is a systematic approach for requirements analysis. It uses dataflow diagrams and data dictionary for requirement analysis. This process begins with studying the existing system to understand the working system to design the context diagram.


epare The Context Diagram
onstruct Logical Level-1 DFD
ecomposition of Level-1 DFD
entifying Man-Machine Boundary
epare Data Dictionary, Process Descriptions
Requirements Specifications

<u>Pros and cons of structured analysis:</u>

- Its easy to present the customer problems in pictorial form ,that can easily be converted into structured design.
- DFD based approach can be used to support other methodologies.
- It does not require much technical expertise.
- It provides proper communication.
- It is time consuming and tedious job to manage such voluminous data.

❖ **Data Oriented Analysis**

<u>Entity relationship model:</u>

A weak entity is an entity that cannot be identified uniquely by its attributes alone .

ER Diagram- Pictorial method of data representation. Entity Relationship Model is represented by the ER diagram that represents data and organizes.

✓ **Entities-**

An entity may represent a group of people, places, things, events or concepts. It is the data object about which information is stored. Ex: student ,course, university An entity instance is the single instance of an entity class. An entity is represented by a rectangle and the entity name with it.Weak entity set is represented by a doubly outlined box and its relationship is represented by a doubly outlined diamond.

✓ **Attributes-**

Each entity will have one or more attributes. For example entity 'course' contains course ID, name, credits, students and faculty. Attributes are represented by ellipses. Logically grouped attributes are compound attributes. The attribute which is used to uniquely identify an entity is called key attribute. An attribute can be single valued or multi valued.

✓ **Relationships-**

It represents the association between two or more entities. Relationships are classified in terms of degree, connectivity, cordiality, direction, and participation. Number of entities associated with a relationship is called the degree of relationship.

Cardinality defines the number of occurrences of entities which are related to each other.

It can be

- One to one
- One to many
- Many to many .

Aggregation-

It is part-of or part whole relationship between entities.

Specialization-

It represents the is-a relationship. It is an Top down process. There can be specialized subjects of theory and practical.

Ex: Student is a person. It has a specific feature.

Two wheeler is a vehicle.

Generalization-

It is also referred as a kind of relationship means child class is a kind of base class .Here child class can access all functionalities from parent class. Child substitutes in parent.

<u>DOA method:</u>

1. Identification of entity and relationships
2. Construct the basic ER diagram
3. Add key attributes to the basic ER model
4. Add non key attributes to the basic ER model
5. Apply hierarchical relation
6. Perform normalization
7. Adding integrity rules

- **Object Oriented analysis**

<u>Object modeling:</u>

-Identifying objects and classes

-Prepare data dictionary

-Identifying associations

-Identifying attributes

-Refining with inheritance

-Grouping classes into modules

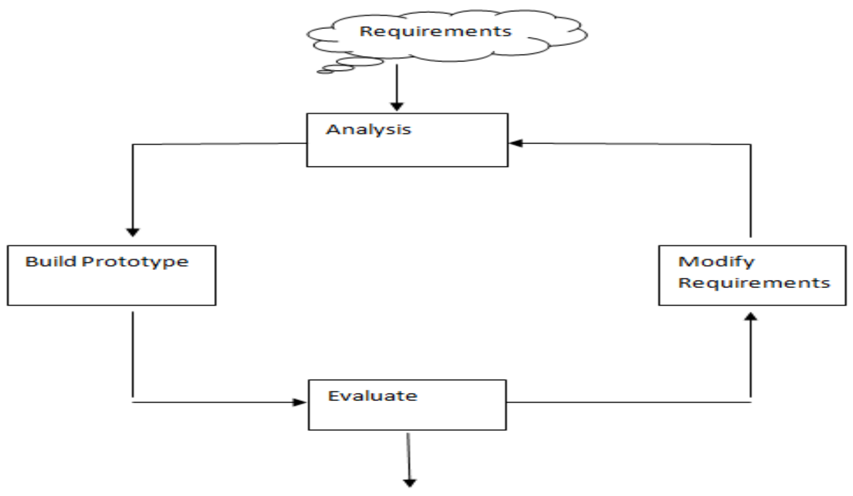
Function modeling: It describes computations within the system .It describes what actions performed are without specifying how (or) when they are performed. Functional model is represented through DFD.

- These three models object model, dynamic model, functional model are necessary for understanding the problem and system behavior.
-Functional model involves inputs, transformations and outcomes.
-Outputs are generated on some input after applying certain transformation to it.

❖ Prototyping

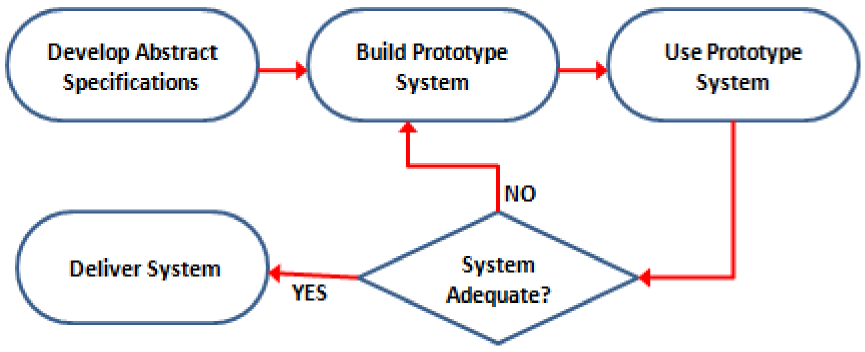
Throwaway Prototyping:

In this prototype is built as quickly as possible for the purpose of observing the product viability .If the product is not acceptable to the customer then it is totally discarded and project begins from scratch. Various prototypes are developed until the customer is satisfied with its operational features .It is useful in detecting uncovered errors.



Evolutionary prototyping:

Here is a prototype is build with the focus that the working prototype will be considered the final system. The process begins with the customer requirements prototypes are produced in several iterations .They are shown to the customer for their acceptance and customer suggestions until final product is constructed.



SOFTWARE REQUIREMENT SPECIFICATION (SRS)

SRS document is a formal document that provides the complete description of the proposed software. It is written in structured manner after gathering and analyzing the requirements .It is very important document for software development.

-It provides basis for later stages of SD
-It acts as software documentation for validation and verification
-It acts as an agreement on the features incorporated in the final system
-Good quality of SRS includes high quality software product
-High quality SRS document reduces development effort because unclear requirements always lead to unsuccessful projects

Characteristics of SRS

- ✓ **Correctness-**An SRS is correct if it possess all the required capabilities of the real world operational environment. That is it should meet all the needs of the customers.

- ✓ **Unambiguity**-An SRS is unambiguous if and only if it has only one semantic interpretation. Unambiguity is seen in the use of natural languages because different people use them in their own ways for representation.

- ✓ **Completeness**-An SRS is complete if and only if it includes all the important requirements; defines all the real world situations, both valid and invalid input data and covers all the references related to illustrations.

- ✓ **Consistency** –An SRS is consistent if there are no contradictory requirements in the different parts of the SRS . The SRS should be consistent within itself and consistence to the reference documents .The requirement becomes inconsistent if there is any mismatch in the facts, inputs, outputs, standards, etc.

- ✓ **Stability / Should Be Ranked For Importance** – An SRS is ranked for importance and /or stability if each requirement has an identifier to indicate either the importance or the stability of a particular requirement.

- ✓ **Verifiability**- An SRS is verifiable if and only if every requirement stated therein is verifiable .A requirement is verifiable if and only if every requirement stated therin is verifiable .

- ✓ **Modifiability-** An SRS is a modifiable if and only if its structure and style are such that any changes to the requirements can be made easily, completely, and consistently while retaining the structure and style . Redundant requirements can easily lead to errors.

- ✓ **Testability –**An SRS is testable if and only if all the requirements can be tested against pass/fail or some quantitative assessment criteria .Every requirement should be testable from the specification or its referenced documents.

- ✓ **Validity –** The validity of requirements comes from the facts that have been provided by the stakeholders. Valid requirements can be understood, analyzed, and accepted or approved by the stakeholders.

- ✓ **Traceability** –An SRS is traceable if the origin of each of its requirements is clear and if it facilitates the referencing of each requirement in future development or enhancement documentation. Requirements can be traced out in either forward or backward direction.

Components of SRS

Completeness of specifications is difficult to achieve and even more difficult to verify. Having guidelines about what different things an SRS should specify will help in completely specifying the requirements. The basic issues an SRS must address are:

- ➢ **FUNCTIONAL REQUIREMENTS-**

Functional requirements describe the behavior of the system that is supposed to have in software product. They specify the functions that will accept inputs, perform processing on these inputs and produce outputs .For each functional requirement, a description of inputs and sources, sequence of operations, measurement, effect of input, intermediate input and range of valid inputs must be specified .The input data play an important role in processing and producing correct outcomes. Thus validity and check checks and constraints are specified in the inputs of each functional requirement.

- ➢ **PERFORMANCE REQUIREMENTS –**

This component of SRS specifies the performance characteristics or non functional aspects of software system .performance requirements are classified into static and dynamic .Static requirements are fixed and they do not impose constraints on execution of the system .These are based on capacity of the system .Dynamic requirements specify constrains o the execution behavior of system .These include system efficiency ,response time, throughput system priorities, fault recovery ,integrity and availability constraints on system .

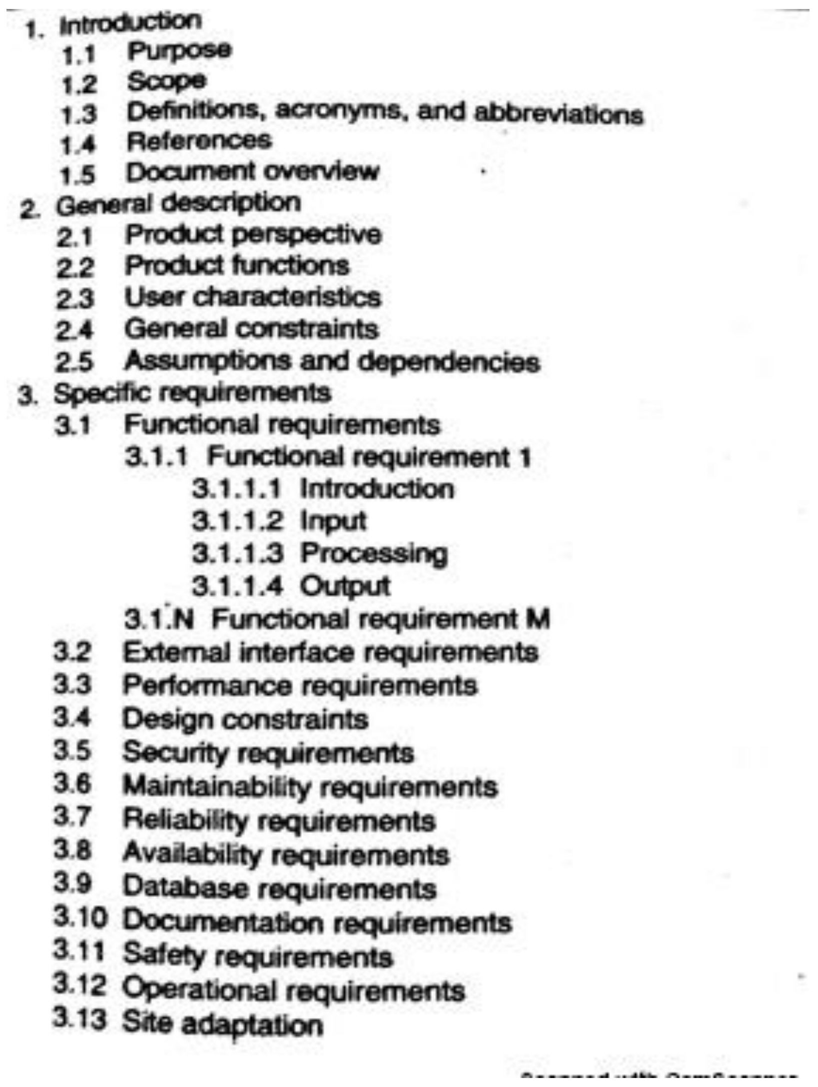
- ➢ **Design constraints**

There are certain design constraints that can be imposed by other standards hardware limitations, and software constraints at the clients environment .These constraints may be related to the standard compliances and policies, hardware and software constraints .Standards compliance specifies the requirements for the standards which are followed according to the regulations, procedures, and policies. For example it may include report formats, data naming, accounting procedures and audit tracing.

Structure of SRS

Structure describes the organization of the software requirement document. Here the concept of template allows the requirement alanyst to focus on organization of contents.

So, the IEEE has published a standard for writing an SRS.



1. Introduction
   - 1.1 Purpose
   - 1.2 Scope
   - 1.3 Definitions, acronyms, and abbreviations
   - 1.4 References
   - 1.5 Document overview
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
   - 3.1 Functional requirements
     - 3.1.1 Functional requirement 1
       - 3.1.1.1 Introduction
       - 3.1.1.2 Input
       - 3.1.1.3 Processing
       - 3.1.1.4 Output
     - 3.1.N Functional requirement M
   - 3.2 External interface requirements
   - 3.3 Performance requirements
   - 3.4 Design constraints
   - 3.5 Security requirements
   - 3.6 Maintainability requirements
   - 3.7 Reliability requirements
   - 3.8 Availability requirements
   - 3.9 Database requirements
   - 3.10 Documentation requirements
   - 3.11 Safety requirements
   - 3.12 Operational requirements
   - 3.13 Site adaptation

FORMAL SYSTEM SPECIFICATION

A formal specification language consists of two sets syn and sem, and a relation sat between them. The set syn is called the syntactic domain, the set sem is called the semantic domain, and the relation sat is called the satisfaction relation. For a given specification syn, and model of the system sem, if sat (syn, sem), then syn is said to be the specification of sem, and sem is said to be the specificand of syn.

- ✓ **Syntactic Domains**

The syntactic domain of a formal specification language consists of an alphabet of symbols and set of formation rules to construct well-formed formulas from the alphabet.

- ✓ **Semantic Domains**

Formal techniques can have considerably different semantic domains. Abstract data type specification languages are used to specify algebras, theories, and programs. Programming languages are used to specify functions from input to output values.

- ✓ **Satisfaction Relation**

Given the model of a system, it is important to determine whether an element of the semantic domain satisfies the specifications. This satisfaction is determined by using a homomorphism known as semantic abstraction function. The semantic abstraction function maps the elements of the semantic domain into equivalent classes.

- ✓ **Model-oriented vs. property-oriented approaches**

Formal methods are usually classified into two broad categories – model – oriented and property – oriented approaches. In a model-oriented style, one defines a system's behavior directly by constructing a model of the system in terms of mathematical structures such as tuples, relations, functions, sets, sequences, etc.

- ✓ **Operational Semantics**

Informally, the operational semantics of a formal method is the way computations are represented. There are different types of operational semantics according to what is meant by a single run of the system and how the runs are grouped together to describe the behavior of the system.

- ✓ **Linear Semantics:-**

In this approach, a run of a system is described by a sequence (possibly infinite) of events or states. The concurrent activities of the system are represented by non-deterministic interleaving of the automatic actions.

- ✓ **Branching Semantics:-**

In this approach, the behaviour of a system is represented by a directed graph. The nodes of the graph represent the possible states in the evolution of a system.