UNIT-II

# Requirements Engineering & modelling

main Requirements

**Requirement (Software) Engineering :-**

Defining, Documenting & maintaining the requirements.

Requirement → various proposals
From high level - mathematical method function.

Function :-
Don't go to Design/development side.
must be specify,
No 2 opinions.

Requirement Abstraction :- what it is to do (Requirement)
For Bids.

**Types :-**

**Users :-**
* Statements, * services by Diagrams which given by System,
* operational constraints.

**System :-**
→ Contract signed b/w client & Contractor.
→ Detailed description of System's functions services.
→ Analyst lay down rules for building System.
→ operational Constraints
→ what to be followed.

Requirements:-

User, User needs, activities to perform with System

Contract signed

User Requirement Document (URD)

System, To Identify the functionality

SRS document => Black-box View

Concises:-

SRS report should be clean, Decrease Errors.

Structured:-

A structured document is easy to simplify. SRs document undergo several Revisions to Cope up with Requirements, user needs evolve over a period of time. To make modifications for SRS document

Black-box view:-

It defines, to system, what to do & refrain from stating how to do these,

Conceptual In

Reader, It

response t

Verifiable

=> SRS sho

Soft

Conceptual Integrity:-

To merely understand by the Readers. It should characterize acceptable responses to unwanted events. (Response to undesired events)

Verifiable:-

Is all correct in SRS document.

It should check whether Requirements met (or) not.

=) SRS shouldn't include:-

Project development plans

Ex:- Cost, staff, methods etc.

Product assurance plans

Ex:- verification and validation, Configuration management etc.

Designs

Ex:- Analysis, method of making etc.

Software Requirement Specification:-

- purpose (Software developing).
- Description (Software).
- Functionality (Software) (need).
- Performance (Software) (in a particular Situation).

- Non-functional requirements (quality) etc.
- External interfaces (or) Software will interact with hardware/other software it will Gather
- Limitations of Software run in. Environment that the

Four parts:-

a) Functional requirements
b) External interfaces
c) Non-functional requirements
d) Constraints.

a) Functional requirements:-

what is a bahavior, behavi

Defines a function / it's module.
of a software system
Functionality measured with inputs & outputs from the System.

Ex:-

How the designed
Rear-camera in car, web page for login, All bank transfer functions.

It talks about particular system outcome when a task is performed on them by the us

Gives the direction of implementation of a Sy system.

gathered/created by Developers.

main goal; To satisfy the customer.

overview:- function needs & uses

I/o :- Normal Input
Valid Inputs / Destinated outputs
timing -

a) Interoperability:- It describes whether commo's is possible b/w 2 different devices/not.

b) Security:- It describes the security aspect of Software requirements.

c) Accuracy:- The data is correctly calculated (or) not & used by the System, and output is correct.

d) Compliance:- It tells that developed System is compliant to Industrial Standards.

Ex:- online ticket reservation should follow Cyber security guidelines.

c) non-functional requirements:-

The system to be constructed.

performance, portability etc.

Implemented Incrementally in any system.

Ex:- Alerting threat's thoroughfare

Derived from functional requirements.

a) performance:- measures system performance

Time ↓.

b) usability:- measures usability of Sw system being developed.

c) maintainability:- ease with which the system can be maintained

If, meantime b/w failures MTBF ↓ / MTTR ↑ mean time to Repair

maintainability is low.

d) Realibility:- It Emphasizes the availability of a system under certain conditions.

e) portability:- Ability of a software system to work in a different Environment, if the underlying dependent framework stays

the same.

1) Serviceability:- To install Software & clean the technical issues.

2) Adaptability:- Adapting to the changing Environment.

Requirement Engineering processes:-

The processes mainly depends on Domain, set.

Inception, Elicitation
:- set of questions to Establish a software process.

⇒ methods of Elicitation:-

- Interviews, closed-Ended questions
  open-Ended questions
- Brain storming Session, group of People create solution for the problem. free from criticism, Documented
- Facilitated Application Specification Technique (FAST)

Bridge(Expectation) b/w developers to Lilt & customers to get. All inputs taken & draft generated.

• Quality function deployment

Customer satisfaction is of main prior

Requirements:-

Normal, Excepted, Exciting

| to develop Software | to customer wish | to develop feature beyond customer Expectation |
|---|---|---|

• Use case approach

Combo of text & pictures

Identifies Interactions b/w System & user

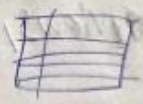
main things

Actor, Interaction (use case), connection

| Interacts with system | name of Interaction verb | Links b/w actors & Interact |
|---|---|---|

Functional view of a System.

• Scenario



name, actors, description, pre-condition, post-condition, Actions (main Senario), Exceptions

Analysis & Elaboration

Info taken during Inception & Elicita & predefined in Elaboration.

features/ constraints.

negotiation

In limited cost development, rough access on,

Specification

Final work product

In text & pictures (or) graphs.

Requirements verification & validation

Verification;- Software correctly implements a specific fn.

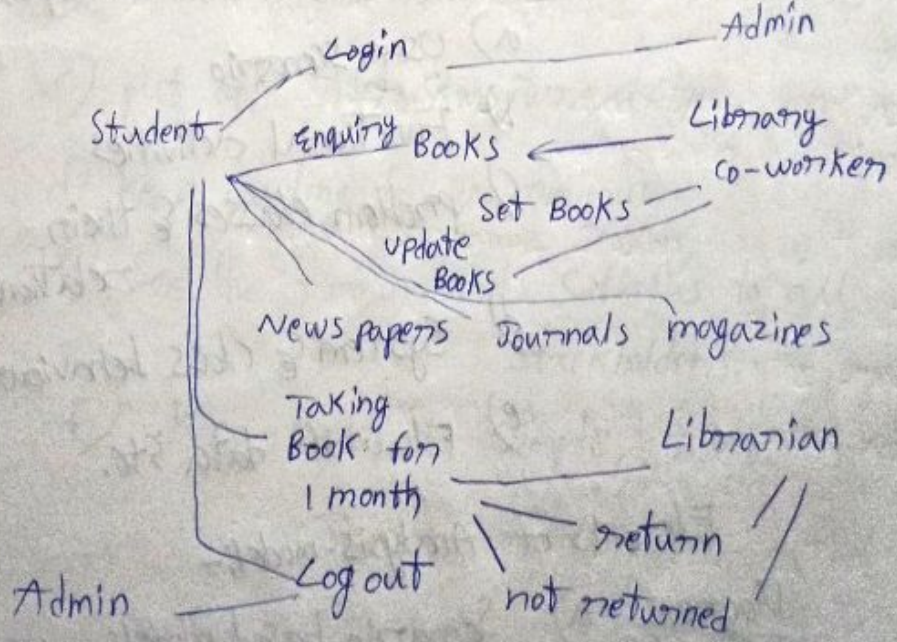
Validation;- Builded software is traceable to customer requirements.

Requirements management;- Tracks the requirements

Traceability table is developed.

features/cost &c.

=> Library - use case



## Requirements modelling:-

planning stage of a software
will give a deeper understanding
for development.
changes from Exact requirements
& specifications.

## Requirement Analysis:-

- operational characteristics - use, efficiency, correctness etc.
- Software interfaces with other Elements
- gives Constraints (Limitations).

It Allows modeler/Analyst to:-

- To Elaborate on Basic requirements
- Build models from different perspectives like,
  a) User scenario
  b) Functional activities
  c) problem classes & their relationships
  d) System & class behaviour
  e) Flow of data etc.

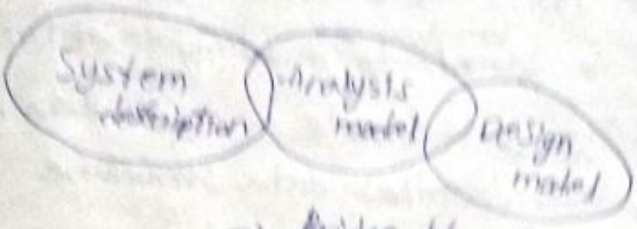
Elements of Analysis-model:-

Diagram
Use-case
Collaboration
Sequence

a) Scenario based models - User's point of view
b) Class-oriented models - similar objects, attributes
c) Behavioural & pattern-based models - Software behaviour
d) Data models - About Info

data flow-oriented model:- Functionality of a system

=>




=) Bridge b/w System-level & overall System
=) Software's Application architecture, User interface,
=) Achieved by humans, System all

Analysis of Thumb rules:-

a) requirements used for the system.
b) Broaden understanding of requirements for system.
c) put off Infrastructure, design
d) Keep coupling to a minimum.
e) See the requirements satisfies to all Stackholders.
f) Keep model as simple, short.

Data modelling:-

It is used to structure, how data is used & stored, to modelling relationships.

Goal is to create a visual data map that describes data structure.

objects, Attributes, Relationships

(E-R) Entity Relationship model

(UML) unified model language model

3 types of data models:-

Conceptual data:- what about

logical data:- How System implemented & regardless of DBMS

physical data:- How System implemented using DBMS,

To establish attributes, relationships, Entities.

To add Exact data

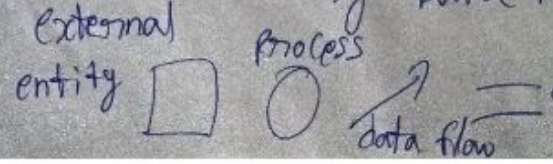
Data base Specific implementation of data model.

Primary, Foreign key, Views Etc.
Index, Authorization

Flow oriented modelling:-

object's transformation through the System,

By data flow diagrams.

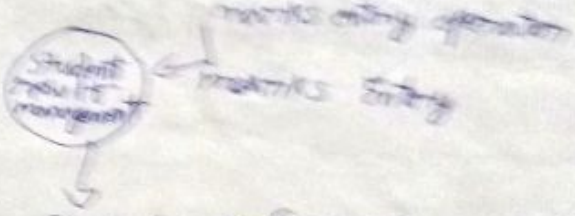


Levels:-

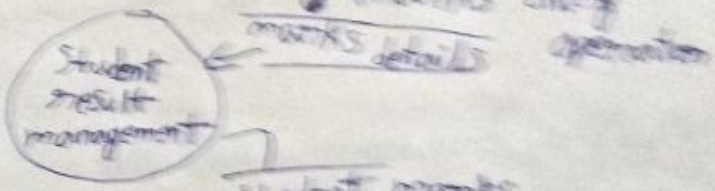
DFD numbered as 0, 1st, 2nd, 3rd ... depth

0 → Context diagram

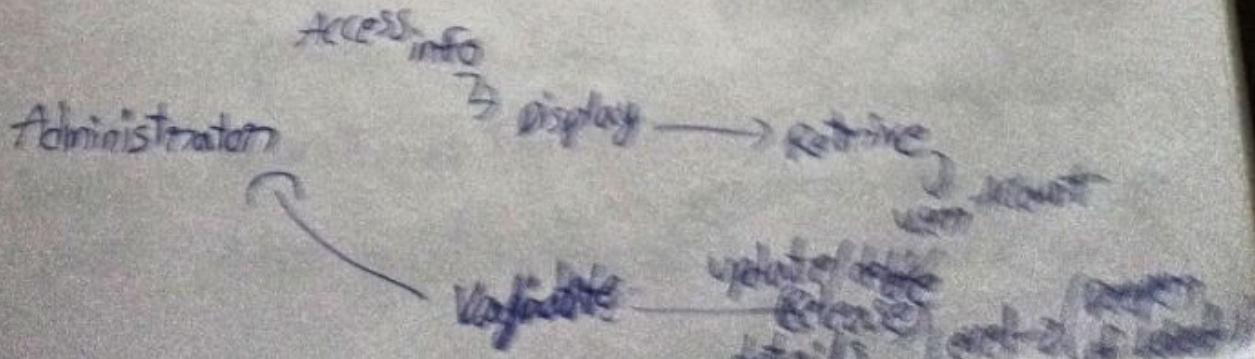
Basic overview of system/process being analyzed/modelled




Level -0




Level-1 (Deeper of Level-0)



DFD for SafeHome System:-

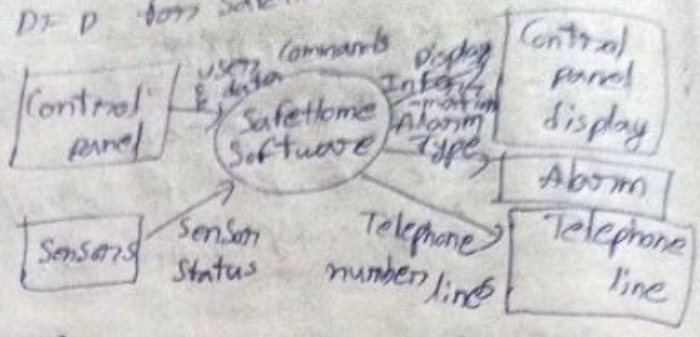



Diagram
⇒ Use case of Safe Home System