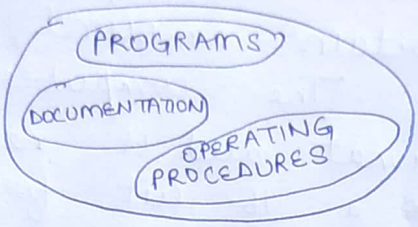
COLLATE

Software Engineering Notes UNIT - 1

SOURCE : NOTESHUB

CH1- INTRO TO S/W ENGINEERING

SOFTWARE:- It is more than just programs. It consists of programs, documentation and the operating procedures to set up the software system.



SOFTWARE

So program is a subset of software. Program is a combination of source code and object code.

- **SOFTWARE ENGINEERING:-** A discipline whose aim is the production of quality software, software that is delivered on time, within budget and that satisfies its requirements.

- **SOFTWARE FAILURES:-** The drawback of software industry is its inability to develop bug free software. So this is also known as software crisis.

Some of the major failures are:-

(i) **Y2K PROBLEM:-** The four digit date format like 1964 ↓ shortened to 2-digit format like 64. But developers could not visualize the problem of year 2000.

(ii) ONE LITTLE BUG, ONE BIG CRASH :-

The ARIANE-5 space rocket was destroyed after 39 seconds of its launch. The reason was simple. When the system's own computer tried to convert one piece of data - i.e. the altitude of certain amount, its velocity changed. The computer had a 64-bit format and it needed to convert it into a 16-bit format. The number was quite big and resulted in an overflow error. So, the unit failed.

(iii) STAR WARS PROGRAM :- USA launched a 'Patriot missile' under the Star wars Program - funding organization. It was used as a defence for Iraqi missiles. But due to a small timing error in the system's clock controlled by a software, it reversed back and killed U.S soldiers.

(iv) WINDOWS XP FAILURE :- It was released on October 25, 2001 and on the same day, some bugs were detected due to which the company posted the four PATCHES (source codes with modifications) to fix them. Two worked but the other two did not. So microsoft advised users to back up critical files before installing patches.

SOFTWARE PROCESS:-

The way in which we produce software is called as SOFTWARE PROCESS.
It encounters certain problems.

* **NOT ENOUGH TIME:-** unrealistic schedules leave insufficient time to do the essential project work. Ex:- Release 1.0, Release 2.0

* **LACK OF KNOWLEDGE:-**

S/w developers do not spend much time reading the literature to find out about the best known ways of software development.

* **WRONG MOTIVATIONS:-**

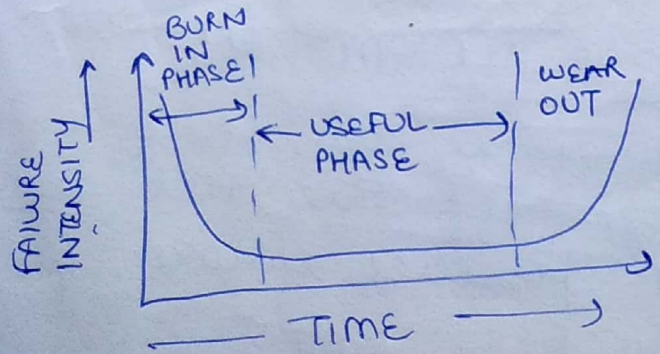
Sometimes, just to get a certified level from an organization, they launch the initiatives and hence do not think about improving the s/w process.

* **SOFTWARE CHARACTERISTICS:-**

(1) DOES NOT WEAR OUT:-

→ Infact, over time it becomes reliable.

SOFTWARE WRVE



HARDWARE PRODUCTS- BATH TUB WRVE

(2) IS NOT MANUFACTURED :- In case of hardware product, every product costs due to raw material. We do not have assembly line in software development.

(3) REUSABILITY OF COMPONENTS :- If we have to manufacture a TV, we may purchase picture tube from one vendor, cabinet from another. and other electronic components from fourth vendor. We assemble all and produce a good quality TV.

In S/w, every project is a new project. We start from the scratch and design every unit of the software produce.

(4) SOFTWARE IS FLEXIBLE :-

A program can be developed to do almost anything.

—

- SOFTWARE MYTHS :-

(1) S/w IS EASY TO CHANGE :-

Every change requires the system to be completely re-verified. Though the source code is easy to edit, but change at one place affects the functionality at other places.

(2) TESTING S/w CAN REMOVE ALL THE ERRORS :-

Testing shows the presence of errors. Our aim is to design effective test cases in order to find maximum possible errors.

(3) S/w CAN WORK RIGHT THE FIRST TIME :-

If an aeronautical engineering is asked to build a

a jet fighter craft and he is asked to put in production, he may refuse the job.

(4) SOFTWARE WITH MORE FEATURES IS BETTER

SOFTWARE:-

(5) ADDITION OF MORE SOFTWARE ENGINEERS WILL MAKE UP THE DELAY:-

By adding more software engineers, we are further delaying the project. Since on adding more, they first need to understand the flow of the project which will further cause the delay.

- SOME BASIC TERMS:-

DELIVERABLES:- Are generated during software development.

Ex:- Source code, user manuals.

MILESTONES:- Are the events that are used to ascertain the status of the project.

Ex:- Completion of design document

PRODUCT- what is delivered to the customer.

Ex- Source code, manuals

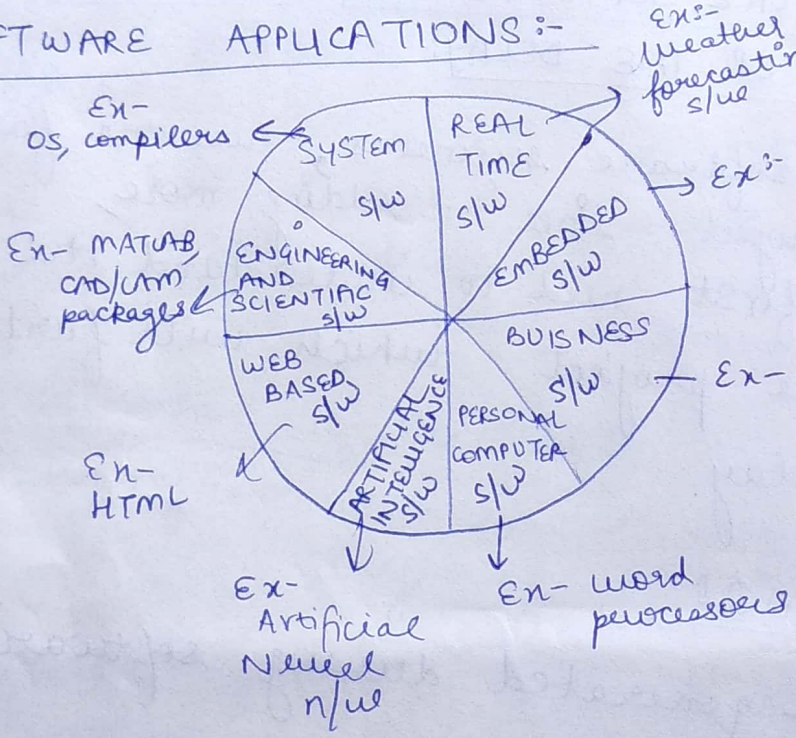
PROCESS- Way in which we produce a software.

(2) IS NOT MANU...

PRODUCT METRICS :-
↓
Measures for the software product
Ex- size, reliability

PROCESS METRICS :-
↓
Quantify the attributes of software development process
Ex:- productivity, quality.

- SOFTWARE APPLICATIONS :-



Ex- OS, compilers

Ex:- weather forecasting s/w

Ex:- This s/w is placed in ROM & controls various functions of product. Ex- Aircraft

Ex- MATLAB, CAD/CAM packages

Ex- Account mgmt

Ex- HTML

Ex- Artificial Neural n/w

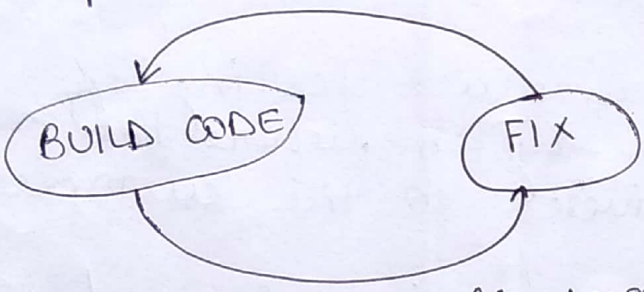
Ex- word processors

CH - 2 - SOFTWARE LIFE CYCLE MODEL

- Software life cycle / Software Development life cycle :- The period of time that starts when a software product is conceived and ends when the product is no longer available for use.

(1) BUILD AND FIX MODEL :- This is the most basic approach. It is an adhoc approach and not well defined.

Has 2 phases
- BUILD AND FIX



APPLICABLE :- For small programs

DRAWBACKS :- COST IS HIGH
NO DESIGNING IS INVOLVED
MAINTENANCE IS DIFFICULT

(2) WATERFALL MODEL :-

Has five phases

Requirement analysis and specification :- The goal of this phase is to understand the exact requirements of the customer and to document them properly. It describes "what of a system and not "how". This phase produces a large document containing a description of what the system will do without describing how it will be done. The resultant document is known as SRS (Software Requirement Specification).

It is a contract between the developer and the customer.

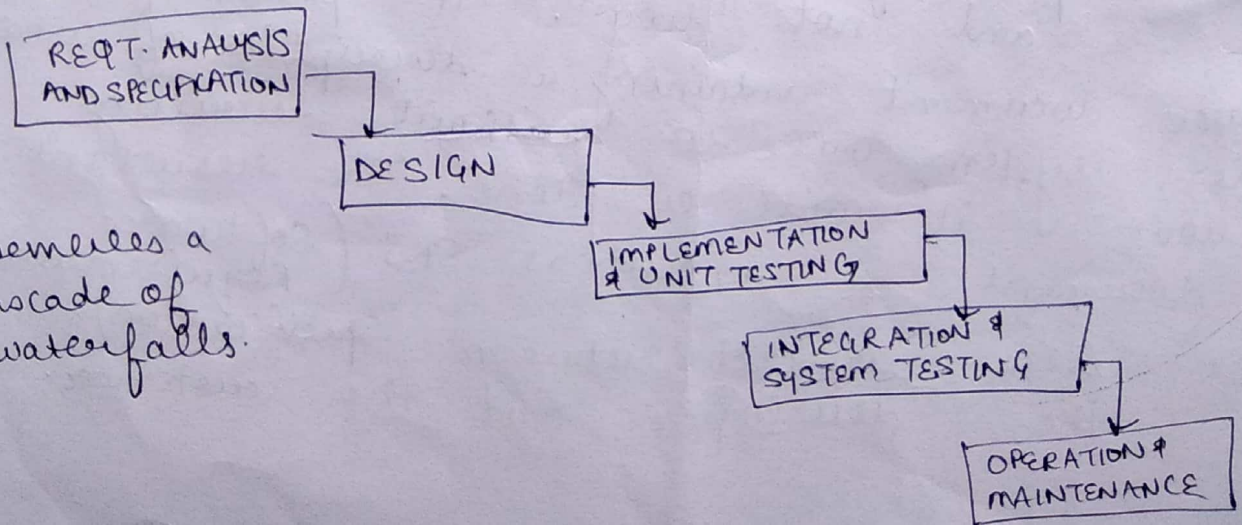
DESIGN PHASE :- The goal of this phase is to transform the requirements into a structure suitable for implementation in some programming language. This work is documented in SDD (Software Design Document)

IMPLEMENTATION AND UNIT TESTING PHASE :- Implement or coding phase proceeds. Here after that individual modules are tested in isolation.

INTEGRATION AND SYSTEM TESTING :- It involves testing of entire system before the software is delivered to the customer.

OPERATION AND MAINTENANCE PHASE :- Here s/w is delivered to the customers site, installed and is operational. maintenance includes error correction, optimization, enhancement of capabilities.

DRAWBACKS :-
- Difficult to define all requirements at the beginning
- model is not suitable to accomodate any change
- working version is not seen until the phases are completed
- Large & real time projects not suitable.



Resembles a cascade of waterfalls.

Diff. b/w evolutionary & iterative model is
- Here a usable product is not available at the end of each cycle.
- Here Requirements are implemented by category than by priority.

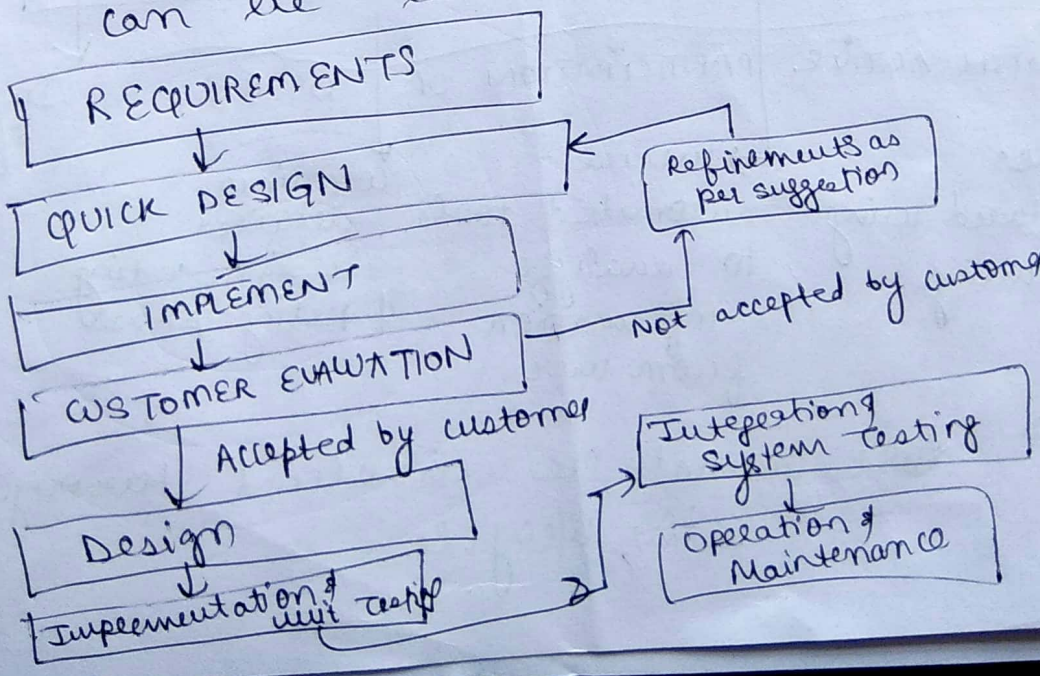
a) PROTOTYPING MODEL :- Here we develop a prototype of s/w instead of developing the actual software so that the serious errors can be avoided
(a dummy/ rough/mode)

Ex:- An Online Exam mgmt system is made :-
U have made a form in which a student has to fill the particulars like the subjects, role no.

NAME - [ ] Tanvi
ROLL NO. - [ ] → 021
SUBJECTS - [ ] ↓ he/she can select code

Monica
021 → X - ALREADY EXISTS - VALIDATIONS HAVE BEEN PUT i.e PRIMARY KEY IS PUT AS ROLL NO.

So the form can be filled but the final working version with all the security parameters put into it, can be launched.

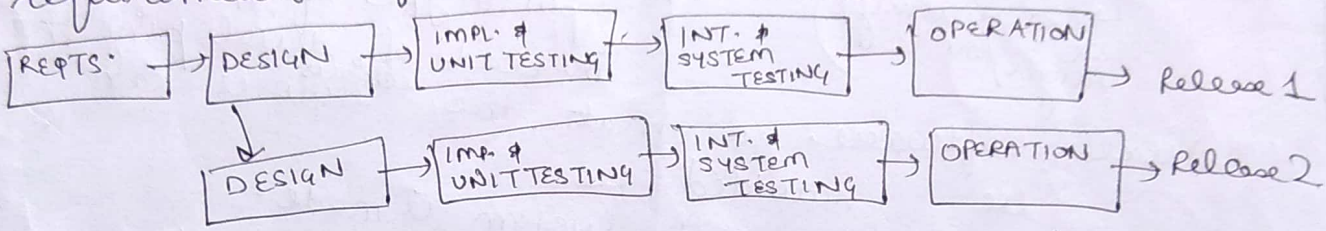


• INCREMENT PROCESS MODELS:-

It works in different cycles and after every cycle, an additional functionality is added. A usable product is given to the customer.

a) ITERATIVE ENHANCEMENT MODEL:

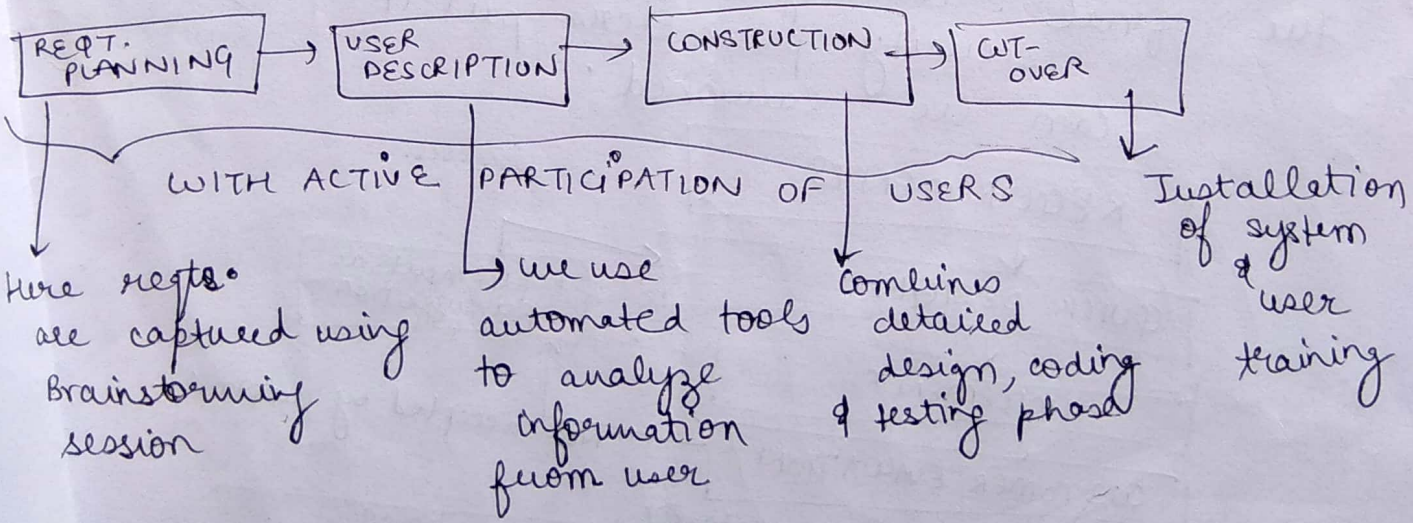
Conducted in several cycles.
The complete product is divided into releases and satisfies only a subset of customers requirements after each release.



The first release is available within few weeks as compared to waterfall model where user waits for months.

b) RAPID APPLICATION DEVELOPMENT (RAD) MODEL:-

- This model is an incremental model.
- Here user involvement is essential from requirement phase to delivery of the product.



DRAWBACK:- User cannot be involved throughout the lifecycle.

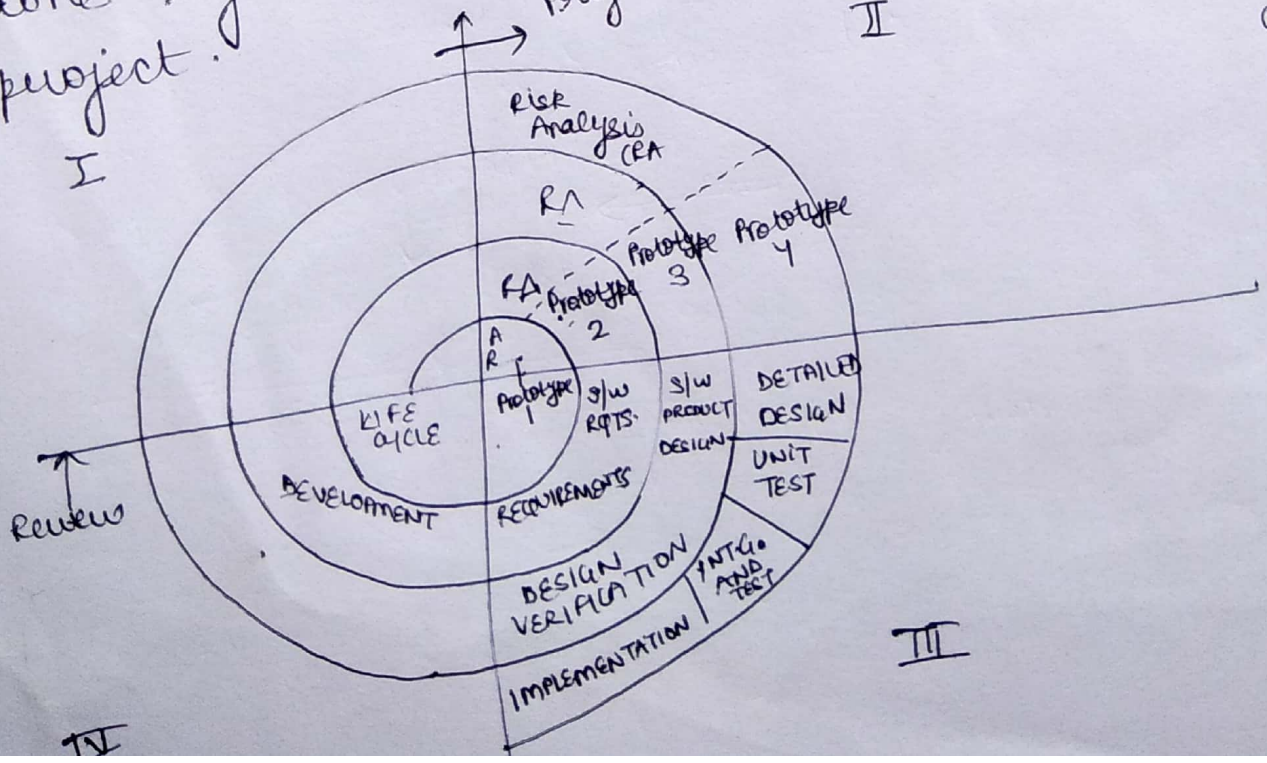
- Though the cost is high, but the experience gathered from developing the prototype helps in developing the actual system.

b) SPIRAL MODEL :-

The problem with traditional software process models is that they do not deal with uncertainity / risk.
Spiral Model was developed by Barry Boehm and it incorporated the 'risk factor'.
It is divided into four phases :-

* PLANNING :- To determine what objectives are to be met
* RISK ANALYSIS :- Attempt to identify and resolve the risks involved.
* DEVELOPMENT :- Product development & testing
* ASSESSMENT :- Customer Evaluation.

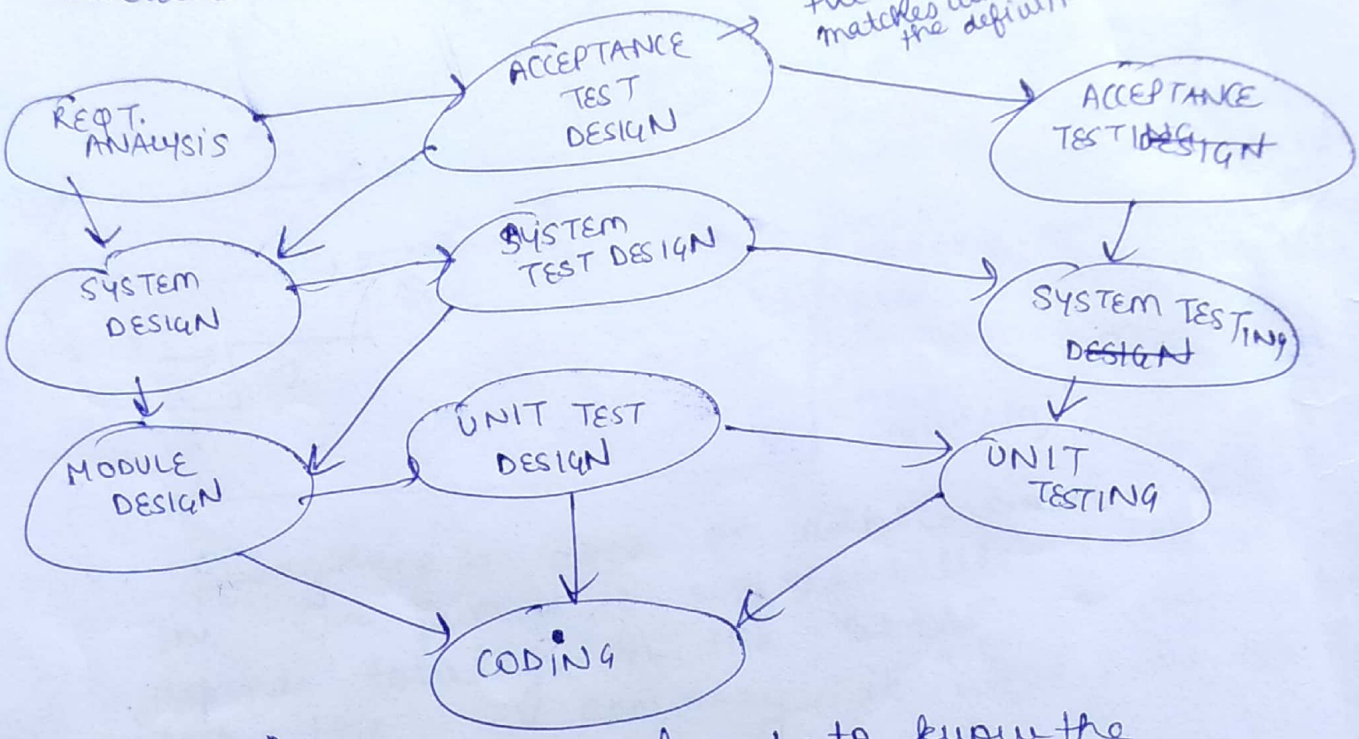
Here after each phase is completed, a review is done by the people concerned with the project.



Advantage :-
It accomodate the good feature of oth life c model

6. V- MODEL :-

It is an SDLC model where execution of processes happens in a sequential manner. Also known as VERIFICATION and VALIDATION MODEL. Here after every single phase in the development cycle, there is a testing phase associated with it.



REQT. ANALYSIS- gathering of reqts. to know the customer's expectations

Here individual modules is designed & then the system as a whole.

It is matched with the expectations stated by the user at the time of meeting with the developer.

TEST DESIGN means TEST CASE- a set of conditions under which a tester will determine whether a system under test satisfies the reqts. correctly

S NOT MANUFACTURER :-

TEST CASE — TO WITHDRAW MONEY FROM ATM

EX:- Generate Coin Button

↳ on clicking it showed generate coins

PREREQUISITES:-

User is authorized

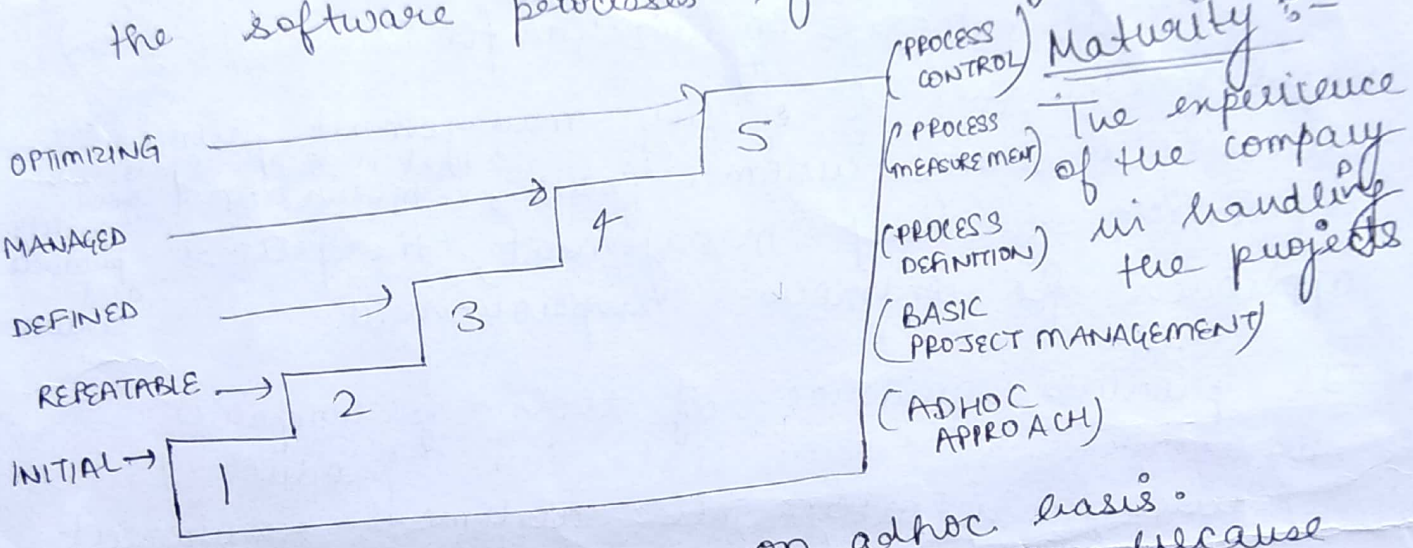
Coin Balance is available

QUALITY STANDARDS :-

(1) CMM (Capability maturity model) -

Is not a software life model but it is a strategy for improvised software process. Developed by software Engineering Institute (SEI).

- It is used to judge the maturity of the software processes of an organization.



Maturity :- The experience of the company in handling the projects

INITIAL :- Everything is done on adhoc basis. The s/w process is unpredictable because it depends totally on the staff, so difficult to predict the accuracy it will take to develop a product.

REPEATABLE :- Certain policies and procedures to implement those policies are established. A protocol is fixed. So the objective is to use these policies to repeatedly be successful in their projects.

DEFINED :- Here organization organizes training program to ensure staff and managers have knowledge and skills to train them. It is to make software managers & technical staff to perform more effectively.

(2) IS NOT MANUFACTURED :

- MANAGED :- Here productivity and quality are measured and maintained, evaluated for the project's software processes.
- OPTIMIZING :- Here entire focus is on continous process improvement.

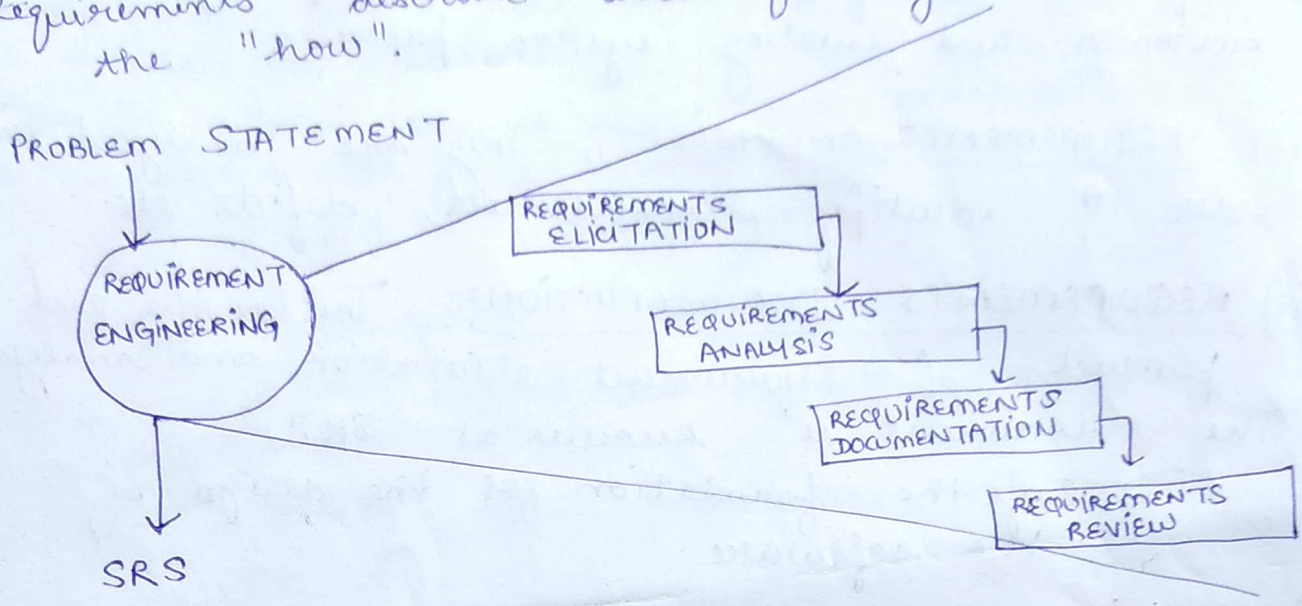
* ISO-9001 :- QUALITY MGMT. STANDARDS
  (International Organization for Standardization)

- Based on numbers of quality management principles including a strong customer focus, motivation & good implication of top management, the process approach for continous improvement. so that they are consistent, quality products and services
- It provides guidance of tools for companies who want to ensure that their products & services consistently meet customer's requirement
- Applicable for any organization, large or small, regardless of its field of activity.

2. RETURN OF BOOKS :-

CH 3- SOFTWARE REQUIREMENTS : ANALYSIS AND SPECIFICATIONS

- Requirements describe "what" of a system and not the "how".



REQUIREMENT ENGINEERING CRUCIAL STEPS

Requirement Engineering is a discipline to describe a proposed system's behaviour and its associated constraints. It produces one large document containing a description of what the system will do without describing how it will do.

The input - PROBLEM STATEMENT
gives an overview of the existing system along with broad expectation from new system.

The output - SRS

(i) REQUIREMENTS ELICITATION:- This is also known as gathering of requirements. They are identified with the help of customer and existing system processes.

(ii) REQUIREMENTS ANALYSIS:- They are analysed in order to identify inconsistencies, defects etc.

(iii) REQUIREMENTS DOCUMENTATION:- This is the end product of Requirement elicitation and analysis. The document is known as SRS.
It lays the foundation for the design of the software.

(iv) REQUIREMENT REVIEW:- It is carried out to improve the quality of the software Requirement Specification (SRS). Also called as REQUIREMENT VERIFICATION

- PROBLEM STATEMENT DESIGN :-

LIBRARY MANAGEMENT SYSTEM

A software has to be developed for automating manual library system of a university. It should be designed to provide functionalities as explained below :-

1. ISSUE OF BOOKS:-

a) A student should be allee to get books issued.
b) BOOKS from GENERAL SECTION are issued to all but BOOK BANK books are issued only for respective courses
c) A limitation is imposed on numeer of books.

2. RETURN OF BOOKS :-

a) Any person can return the issued books
b) The student info. is displayed using bar code detector.
c) System displays student details on whose name the books were issued.
d) The information is saved and corresponding updations take place.
e) A fine of Re 1 is charged per day.

3. QUERY PROCESSING :-

The system should be able to provide information like :-

(i) Availability of a particular book.
(ii) Number of copies available of the desired book.

TYPES OF REQUIREMENTS :-

FUNCTIONAL / PRODUCT FEATURES

- Describes what the software. Related to the expectations from the intended software.

NON-FUNCTIONAL / QUALITY REQUIREMENTS

- Tells how well the software does what it has to do.

Ex- maintainability, portability, testability.

• FEASIBILITY STUDIES:-

It is defined as an evaluation/ analysis of potential impact of a proposed project/program. It is conducted to assist decision makers in determining whether or not to implement a project. Based on intensive research on both the current practises and proposed project and its impact on the system as a whole.

Types

- TECHNICAL
- ~~ETHICAL~~ ECONOMIC
- LEGAL
- CULTURAL
- OPERATIONAL — solution for possible errors has been defined.

CANDIDATE KEYS:-

REQUIREMENT ELICITATION:-

It means gathering of requirements.
The requirements actually reside in user's mind. Hence, the most important thing is to find out what users really need.
It is done by asking questions, writing down the answers etc.

① INTERVIEWS:-

It is an one-on-one interaction between the specialised developers and customers. The objective is to understand the customer's expectations from the software.
Both parties have different goals, opinions but both want the project to be a success.

Interviews can be :-

OPEN-ENDED
- No pre-set agenda
- Questions are asked just to have an overview of the system.

Ex:- In Result-mgmt system, the developer may ask :-
- WHO WILL USE THE SYSTEM
- WHO IS CONTROLLER OF EXAMINATION

STRUCTURED
- Agenda is prepared.
- A questionnaire is designed for the interview

• FEASIBILITY STUDIES:-

TYPES OF USERS / STAKEHOLDERS:-

(i) ENTRY LEVEL PERSONNEL :- They might not have sufficient domain knowledge but may be useful for fresh ideas & different views.

(ii) MID-LEVEL STAKEHOLDERS :- Have better domain knowledge and know the complex areas of the project.

(iii) MANAGER:- Higher level management officers also provide a broader outlook towards the software development.

• **BRAINSTORMING:-** Is a group technique used to understand the requirements.

It leads to new ideas quickly and helps to promote creative thinking.

All participants are encouraged to say whatever ideas comes in their mind; whether they are relevant or not. No one is criticized for speaking up.

A facilitator is available to conduct the smooth session so that no clashes occur or no individual ego takes place.

White boards or computer projections are used.

At the end of the session, the trained facilitator prepares a detailed report.

uniquely identify an

- QFD (QUALITY FUNCTION DEPLOYMENT) :-

It is a quality management technique that incorporates the voice of the customer. The voice is then translated into technical requirements. → further into design requirements.

Three types of requirements :-

NORMAL
- The objective is discussed with customer and presented.

Ex- In result management system :- entry of marks

EXPECTED
- If such requirements are not present, customer will be dissatisfied.

Ex- protection from unauthorized access.

EXCITING
- Some features go beyond the customer's expectation

Ex- when unauthorized access occurs, system will shutdown all processes & e-mail is generated to system admin

STEPS :-

1. Identify all stakeholders & identify initial constraints
2. list out requirements.
3. A value indicating a degree of importance is assigned to each requirement
   - 5 : Very imp
   - 4 : imp
   - 3 : not imp., but nice to have
   - 2 : not imp
   - 1 : unrealistic.

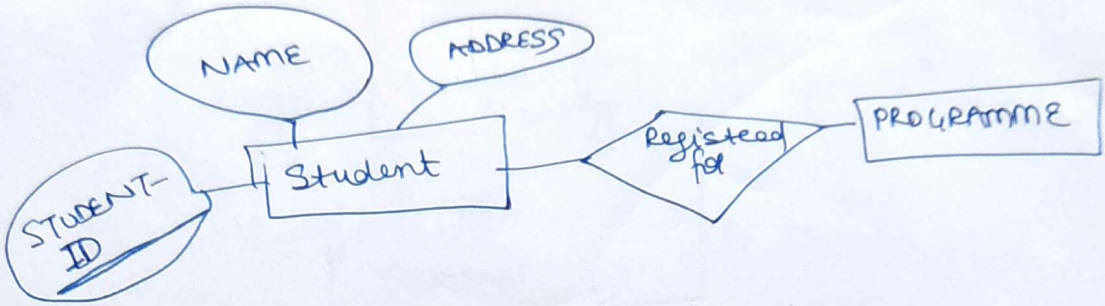
- FAST (FACILITATED APPLICATION SPECIFICATION TECHNIQUE)
  - Is a team oriented approach to bridge the expectation gap difference between what developers think they are supposed to build and what customers think they are going to get.
    - Arrange a meeting for developers and customers
    - Appoint a facilitator
    - Prepare a definition mechanism
    - Participants should not criticize or debate.

ACTIVITIES:-

- Each participant presents his or her list of objects, services. List is displayed in meeting by using large sheets of paper. so that visible to all participants.
- The combined lists for each topic are prepared by eliminating redundant entries
- The participants who have written out same views are combined in subteams. to allow more views.
- All the issues raised during the discussion that cannot be resolved are put in an issue list.

CANDIDATE KEYS:-

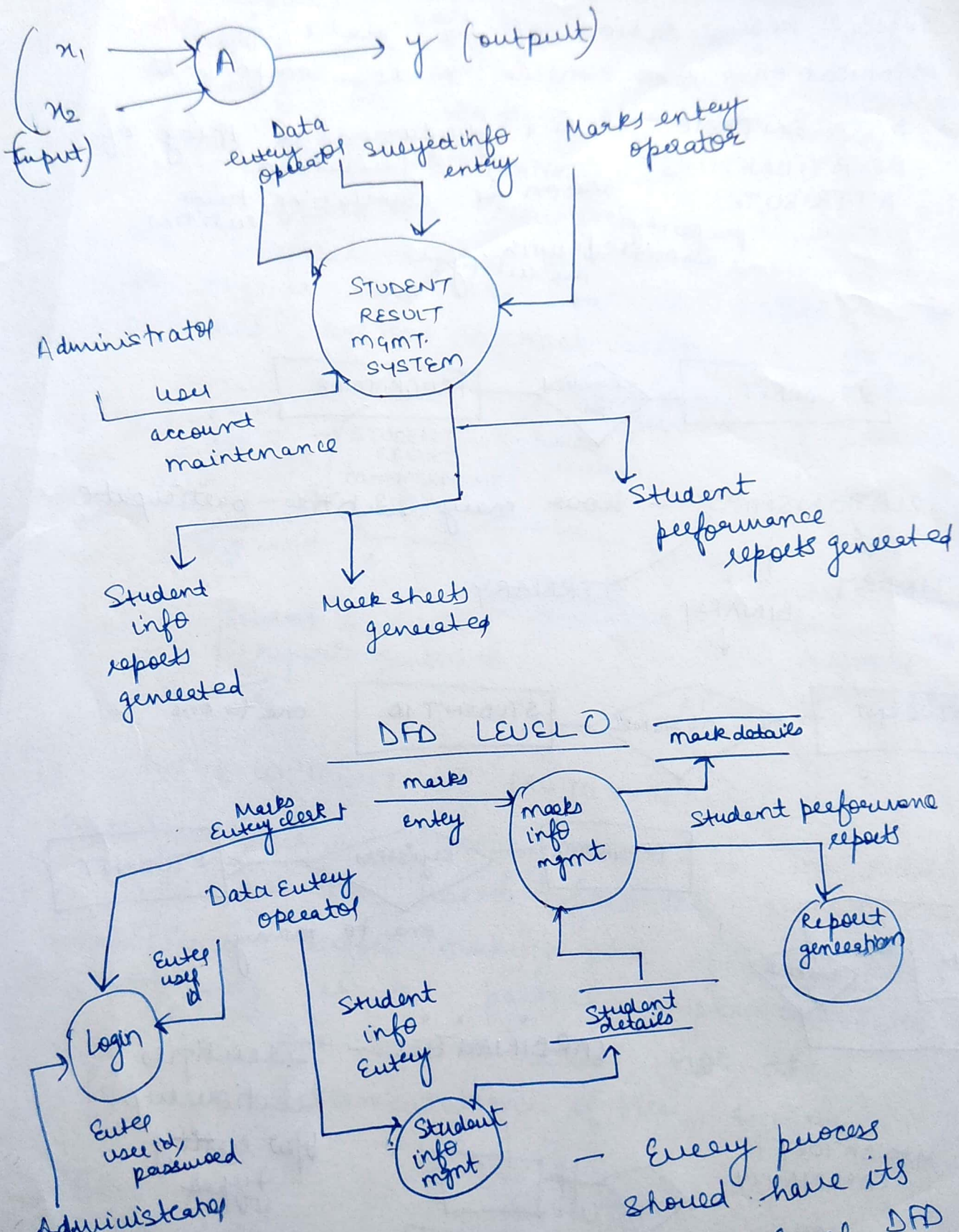
↳ to uniquely identify an attribute



- REQUIREMENT DOCUMENTATION:-
  - ↳ Characteristics of SRS
  - ↳ Organization of SRS

REQUIREMENT ANA...

LEVEL 0 - / CONTEXT DIAGRAM :-



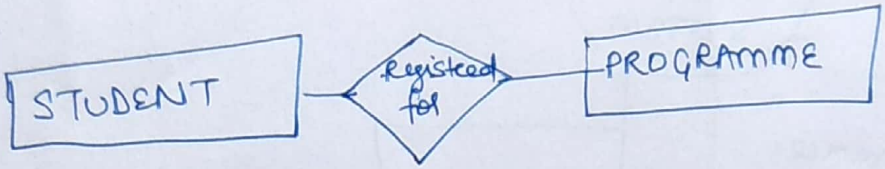
DFD LEVEL-1

- Every process should have its zero-level DFD as well as data store.

(b) ENTITY-RELATIONSHIP DIAGRAMS:- It is a detailed logical representation of the data for an organisation uses three main components

- DATA ENTITIES – is a fundamental thing of an organization.
- RELATIONSHIPS – reason for associating two entities
- ATTRIBUTES – property associated with an entity

Ex:-

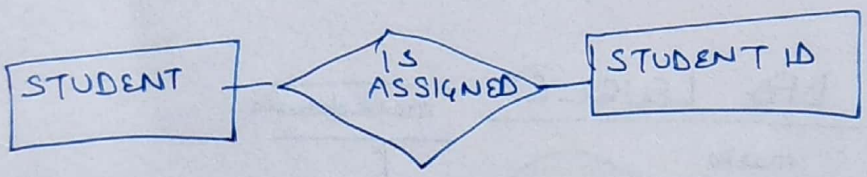


RELATIONSHIPS – how many entities participate

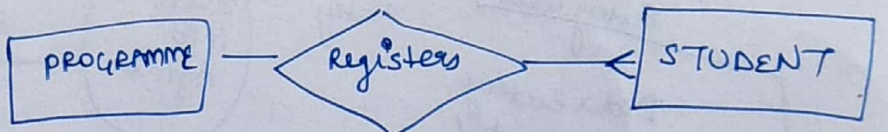
UNARY

BINARY

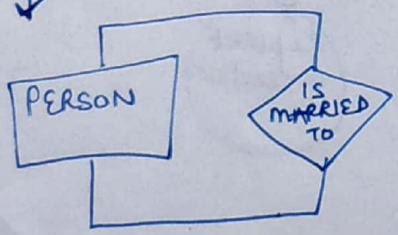
TERNARY



one to one



one to many

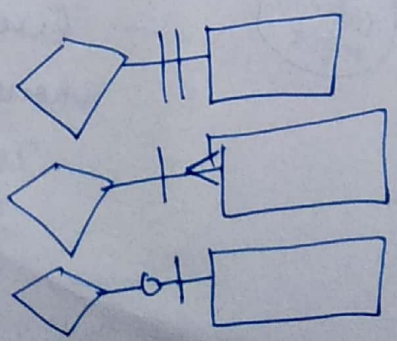


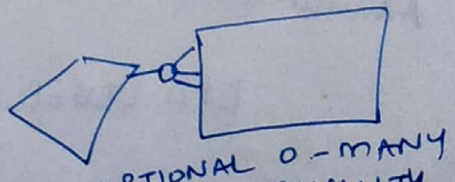
CARDINALITIES – Identifies relationship b/w entity types

MANDATORY 1 CARDINALITY

MANDATORY MANY

OPTIONAL 0/1





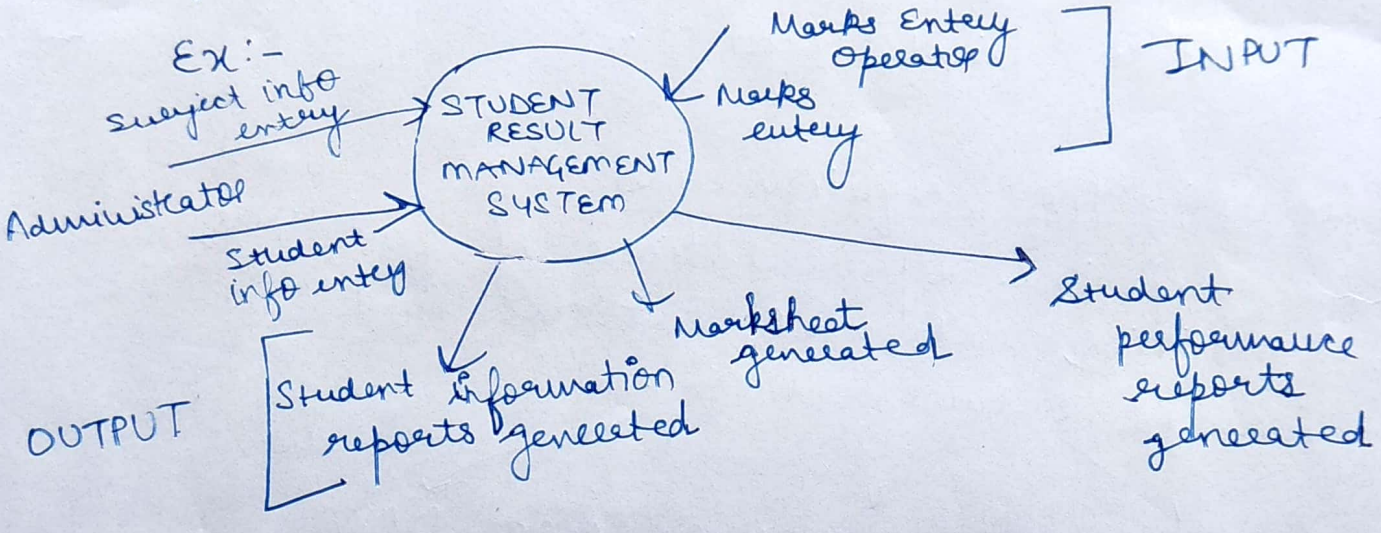
OPTIONAL 0-MANY CARDINALITY

REQUIREMENT ANALYSIS :-

It is a team effort that demands a combination of hardware, software and the skills in dealing with people so as to improve our system. The various steps to achieve are :-

(i) DRAW CONTEXT DIAGRAM :-

It defines the boundaries and interfaces of the proposed system with the external world. It defines how the entities outside the proposed system interact with the system



(ii) DEVELOPMENT OF PROTOTYPE :-

Prototype is a dummy model. The developer asks for the user's feedback and continously modifies it before making the final product. Prototyping is a partial implementation of the system and not the full implementation of the system.

- THROW-AWAY PROTOTYPING
- EVOLUTIONARY PROTOTYPING

(b) THROW-AWAY PROTOTYPING :- Prototype is constructed with the idea that it will be discarded after the analysis is complete and the final system is built from the scratch.

THRO EVOWTIONARY PROTOTYPING :- In this approach, the prototype is built with the idea that it will be eventually be converted into final system.

(iii) Modeling the requirements: It is used for graphical representations of functions, data entities, external entities and relationship between them.

- DFD (Data-Flow Diagrams)
- ER-DIAGRAMS (Entity-Relationship)
- DATA DICTIONARIES

a) DFD/ BUBBLE CHART/ DATA FLOW GRAPH Shows the flow of data through a system.

Symbols :-

- Data flow - to connect process
- Process - Input/output data
- Source/sink - System input/output
- Data store

OPTIONAL 0/1