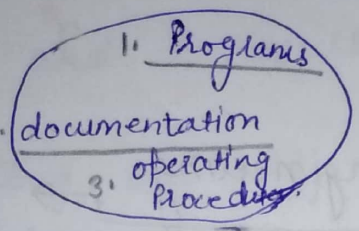
UNIT - 1

Software :→ Software is a set of programs to accomplish a particular task. It consists of programs, documentation of any facet of the program and the procedures used to setup and operate the software system.



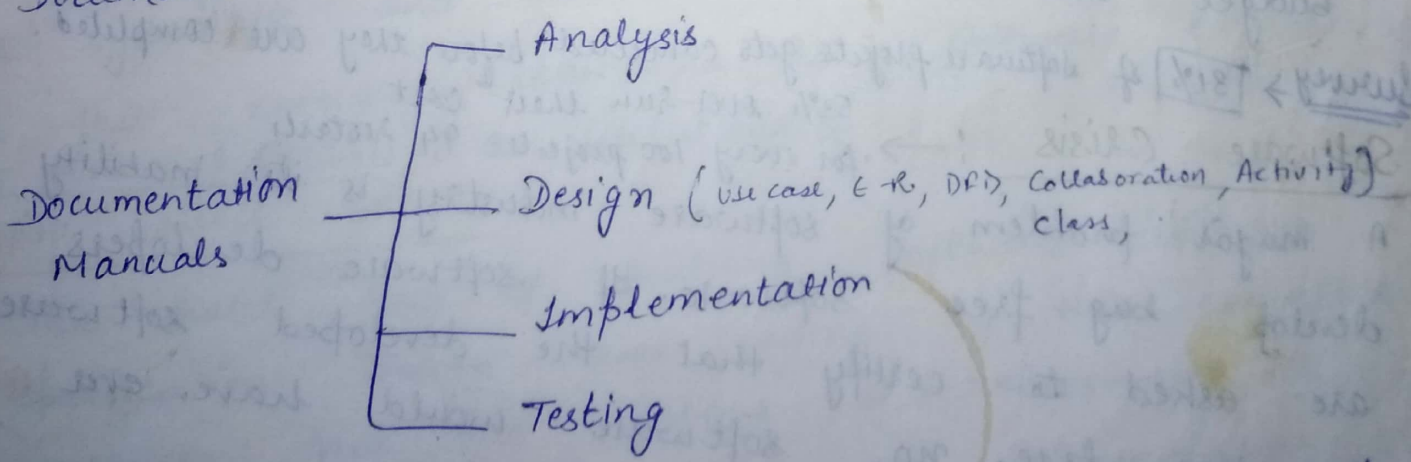
Software = Program + Documentation + operating procedures

Program :- Program is a subset of software and it becomes software only if documentation and operating procedures manuals are prepared.

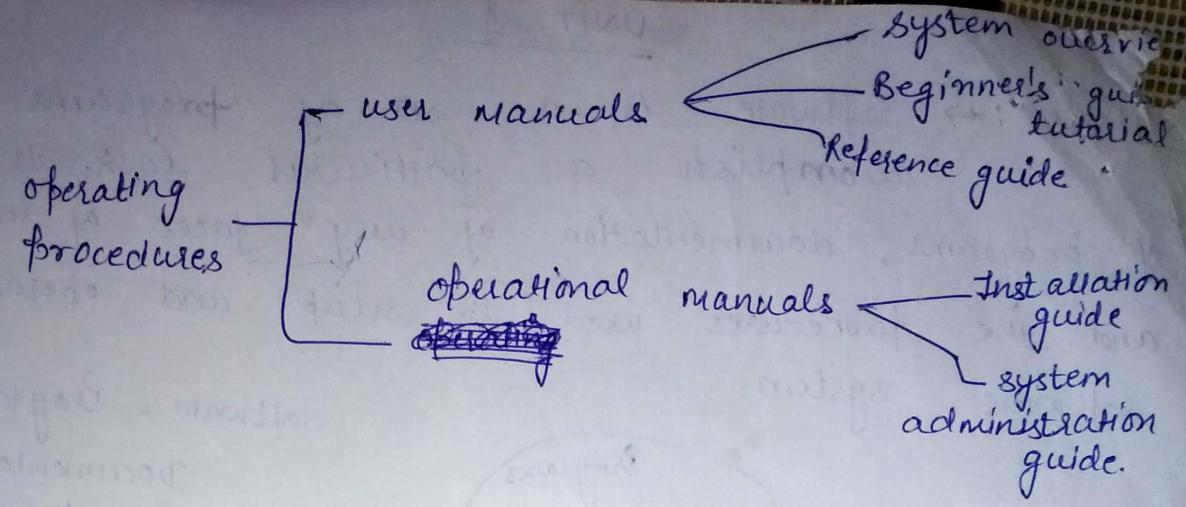
Program is a combination of source code and object code.

Program = source code + object code.

Documentation consists of different types of manuals



operating procedures consist of instructions to setup and use the software system and instructions on how to react to system failure.



Definition of Software Engineering :→

* Software Engineering is a discipline that integrates methods, tools and procedures for the development of computer software.
* The systematic approach to the development, operation maintenance and requirement of software.
* A discipline whose aim is the production of quality software, that is delivered on time, within budget and that satisfy its requirements.

Survey → 31% of software projects gets cancelled before they are completed.
53% over-run their cost
for every 100 projects - 94 restarts

Software Crisis :→

A major problem of software industry is its inability to develop bug free software. If software developers are asked to certify that the developed software is bug free, no software would have, ever been released.

Software failures :→

(i) The Y2k problem was the most crucial problem of last century.

2.) The star wars program of USA produced "patriot missile" and was used first time gulf war. The patriot missiles failed several times to hit scud missiles, including one that killed 28 U.S soldiers. A review was constituted to find the reason and result was software bug.

3). "one little bug, one big crash", of Ariane-5 space rocket, developed at a cost of $ 7000M over 10 years period. The space rocket was destroyed after 39 seconds of its launch, at an altitude of two and a half miles alongwith its payload of four expensive and uninsured scientific satellites.

Hence, in order to handle such unfortunate events, a systematic and scientific discipline is required & this emerging discipline is software engineering.

④ Financial software is an essential part of S.T.

No silver bullet →

The hard part of building software is the specification, design and testing of this conceptual construct, not the labour of representing it and testing the correctness of representation. We still make syntax errors, ~~to be sure~~ That is why, building software is always hard and there is inherently no silver bullet.

2012

11/11/2012

④.

We cannot imagine a day without using cellphones, logging on internet, sending e-mails, watching television and so on. All these activities are dependent on software and software bugs exists nearly everywhere.

→ case tools (computer Aided software engineering)

→ Software Process :→

Software process is the way in which we produce software. This differs from organization to organization. Many software organizations are looking at software process improvement as a way to improve the quality, productivity, predictability of their software development, and maintenance efforts.

→ few reasons why is it difficult to improve software process.

↳ Not enough time :→

(i) unrealistic schedules leave insufficient time to do essential project work.

(ii) No one have time to explore what is wrong with current development process and what they should be doing differently.

(iii) Shortage of time. Customer and senior managers are demanding s/w of high quality.

(iv) ~~Reale~~. Release 1.0, then Release 1.01

↳ Lack of knowledge →

(i) s/w developers are not familiar with

also correct
ed before.

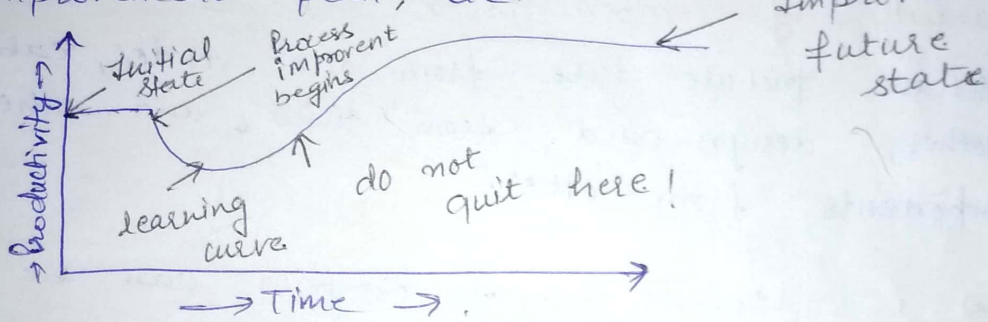
(ii) S/w developers do not spend much time reading the literature to find out.

(iii) more focus on java, visual basic, oracle, rather than books on process, testing, quality.

↳ Wrong motivation → External entity, such as a contractor, demanded that the development organization should achieve CMM (Capability Maturity Model) level x by date Y.

Motivation should be of remove the difficulties we experience on our projects

↳ Insufficient Commitment :→ lack of resources, write no improvement plan, etc.



Process Improvement learning curve.

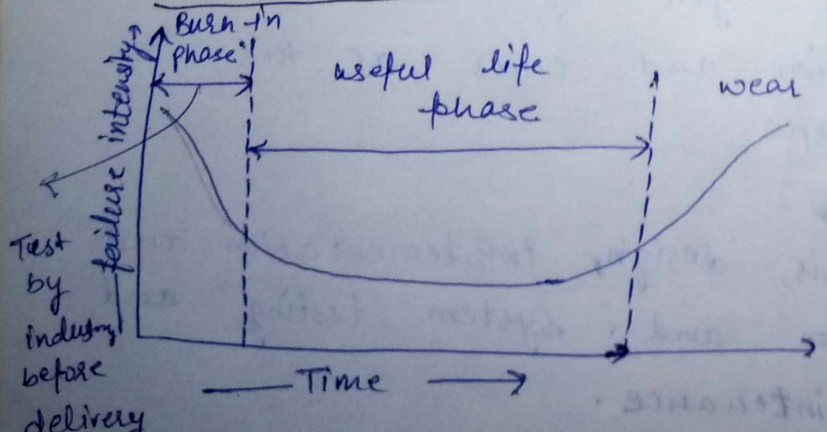
→ Software characteristics →

S/w is one time development product

S/w's behaviours and nature is different than other products of human life.

Some of the important characteristics are:

(1) S/w does not wear out →

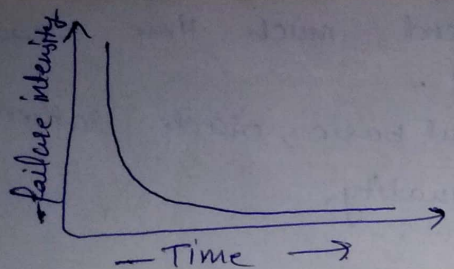


three phases of h/w product:-

(i) Burn-in phase

(ii) useful life phase

(iii) wear-out phase.



As s/w does not wear out, ∴ it does not have wear out phase. ⑥ ⑤

It becomes reliable overtime instead of wearing out.

environment does not affect.

(ii) s/w is not manufactured → (windows)

s/w is one time development effort and continuous maintenance effort in order to keep it operational. Making 1000 copies is not an issue.

(iii) Reusability of components:→ Eg. Tv. (windows, files)

(iv) purchase picture tube from one vendor, cabinet from another, design card from third, and other electronic components from other.

(iv) s/w is flexible → a program can be developed (mind change easily) to do almost anything.

Build & fix model

↳ s/w life cycle Models :→ (SDLC)

s/w development organisations follow some process when developing a s/w product. Process is written and actively managed. A key component of a s/w development process is the life cycle model on which the process is based. Life cycle of the s/w starts from concept exploration and ends at the retirement of the s/w.

→ water fall Model :→

↳ Requirement analysis, design, implementation and unit testing, integration and system testing and operation and maintenance.

system.
ation of new ...

(IEEE standard) fixed formal.

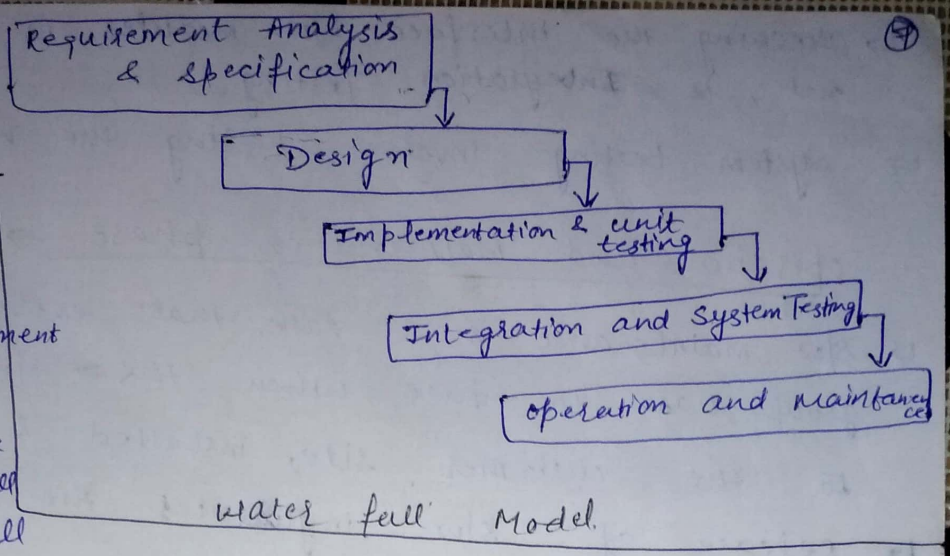
1. Requirement Analysis & specification :

↳ understand exact requirement of the customer and document them properly.

↳ Requirements describe what the system will do, not how it will be done.

↳ SRS document prepared.

↳ SRS is contract b/w the developer & the customer.



2. Design →

↳ Transform the requirements specification into a structure that is suitable for implementation in some programming language.

↳ Work is documented on SDD (software design description) document.

3. Implementation & unit testing :→

↳ Design is implemented.

↳ If SDD is complete, Implementation or coding phase proceeds smoothly.

↳ Small Modules are tested in isolation.

↳ Problem associated with testing the module in isolation.

↳ Problems are solved by writing some overhead code.

4. Integration and System Testing :→

↳ Effective testing will result in a good quality s/w product, more satisfied customer, lower maintainance cost, and more reliable result.

↳ Very expensive activity.

↳ Check the interface b/w Modules is also correct or not.

↳ Checking the interface b/w modules is correct or not, is Integration Testing.

↳ System testing involves testing the whole system.

5. operation and maintenance phase →

↳ s/w maintenance is a task, that every s/w development group has to face, when the s/w is delivered to the customer site, installed & operational.

↳ Release of s/w inguarantes the operation & maintenance phase of the life cycle.

↳ s/w maintenance includes error correction, enhacement of capabilities.

↳ Purpose of this phase is to preserve the value of the s/w over time.

→ Problem of waterfall model :→

(i) It is very difficult to define all the requirements at the begning of a project

(ii) Not suitable for accomodating any change.

(iii) Does not scale up well for large projects

(iv) Real projects are rarely sequential.

Note: Making new projects based on existing designs is very easy with waterfall Model.

→ Increment Process models →

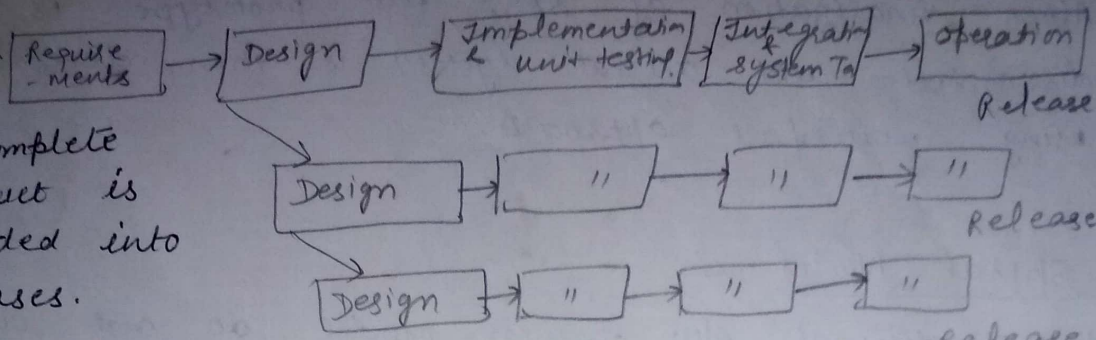
(i) Iterative Enhancement model →

→ After every cycle, a usable product is given to the customer.

→ Every new cycle will have an additional functionality.

→ Developers and customers prioritize the requirements.

of new
Requirements & their priority (5)



A complete Product is divided into Releases.

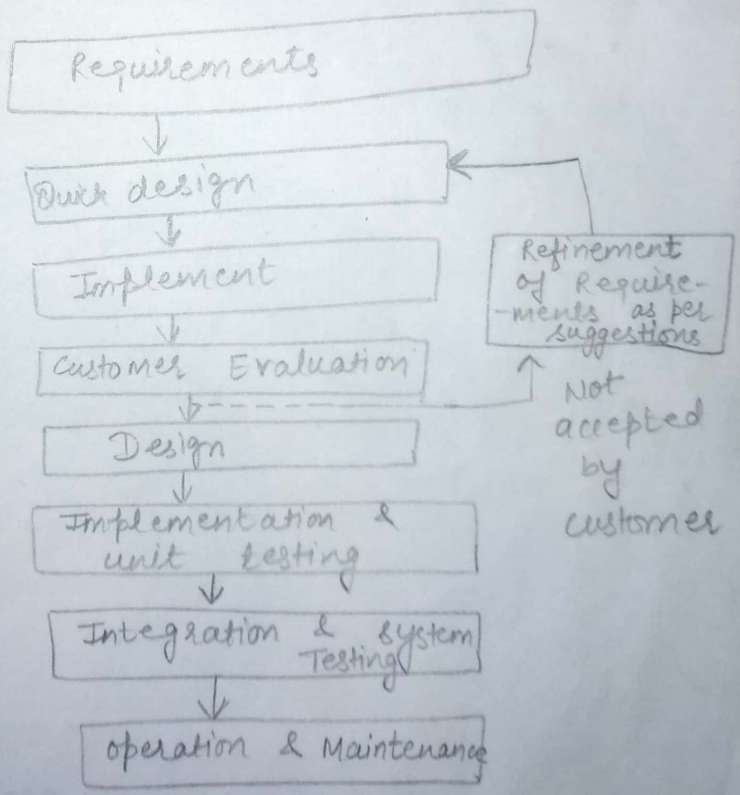
⇒ Evolutionary Process Models →

→ In Evolutionary Process Models, requirements are implemented by category rather than priority.

→ These models are useful for projects using new technology that is not well understood.

→ used for complex projects where all functionality must be delivered at one time, but the requirements are unstable, or not well understood at begning.

(1) Prototype Model →



1. develop a working Prototype of the s/w, instead of developing the actual s/w.
2. Working prototype is developed as per current avilable requirements.
3. Resulting specification document will be correct.
4. Prototype is reviewed by the customer and feedback is given to the developer.
5. Experience gathered from developing the prototype helps in

After finalization of SRS, the prototype is discarded and actual system is then developed using waterfall approach.

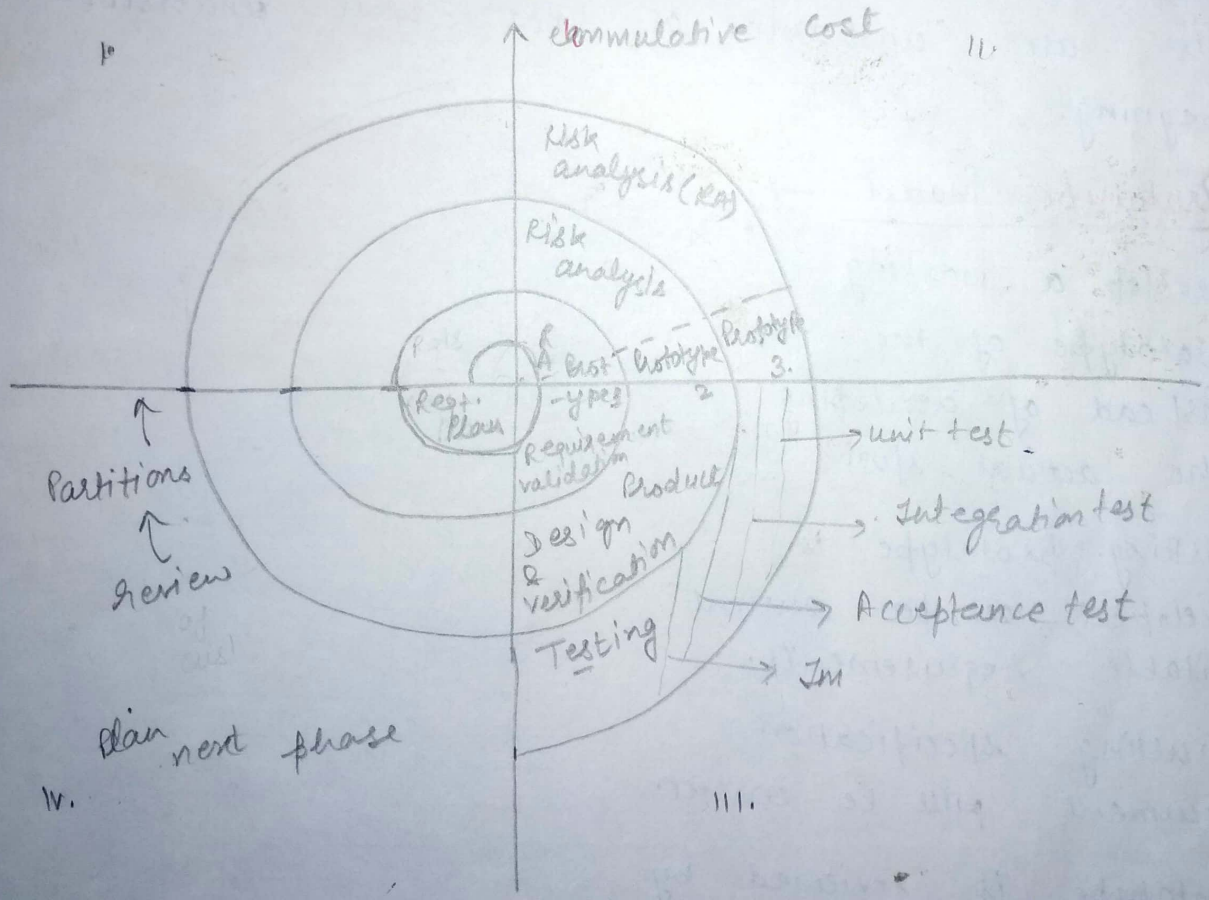
Spiral Model →

(a) Traditional s/w process model do not deal sufficiently with the uncertainity.

(b) Project risks were neglected, and nobody was prepared, when something unforeseen happened.

(c) Barry Bohm recognized this and tried to incorporate the "project risk" factor into a life cycle model.

(d) Result is spiral Model, developed in 1986.



(e) Radial dimension of the model represents the cummulative cost.

(f) Path around the spiral is indicative of increased cost.

→ Angular Dimension represents the progress made in completing each cycle.

→ Each loop of the spiral from x-axis clockwise through 360° represents one phase.

Phase is split roughly into four sectors :→

1. Planning : Determination of objectives, alternatives & constraints.
2. Risk analysis ; Analyze alternatives and attempts to identify, and resolve the risk involved.
3. Development : Product development and testing
4. Assessment : Customer Evaluation & plan for next phase.

1st phase → Planning is performed, risk are analyzed, prototype are built, customess evaluate the prototype.

IInd phase → More refined prototype is built, requirements are documented & validated.

IIIrd phase → By the third phase, risk are known. then design the product.

Aim → Eliminate high-risk problems before they threaten the s/w operation.

Importance →

→ Each phase is completed with a review by the people by the people concerned with the project (developer, programmer).

→ Review consists of review upto that point and includes the plans for next cycle.

→ Partition

→ If plan fails spiral terminated, otherwise it terminates with the initiation of new s/w.

Unit 2 – Part 2

⇒ S/w Requirement Analysis & Specification →

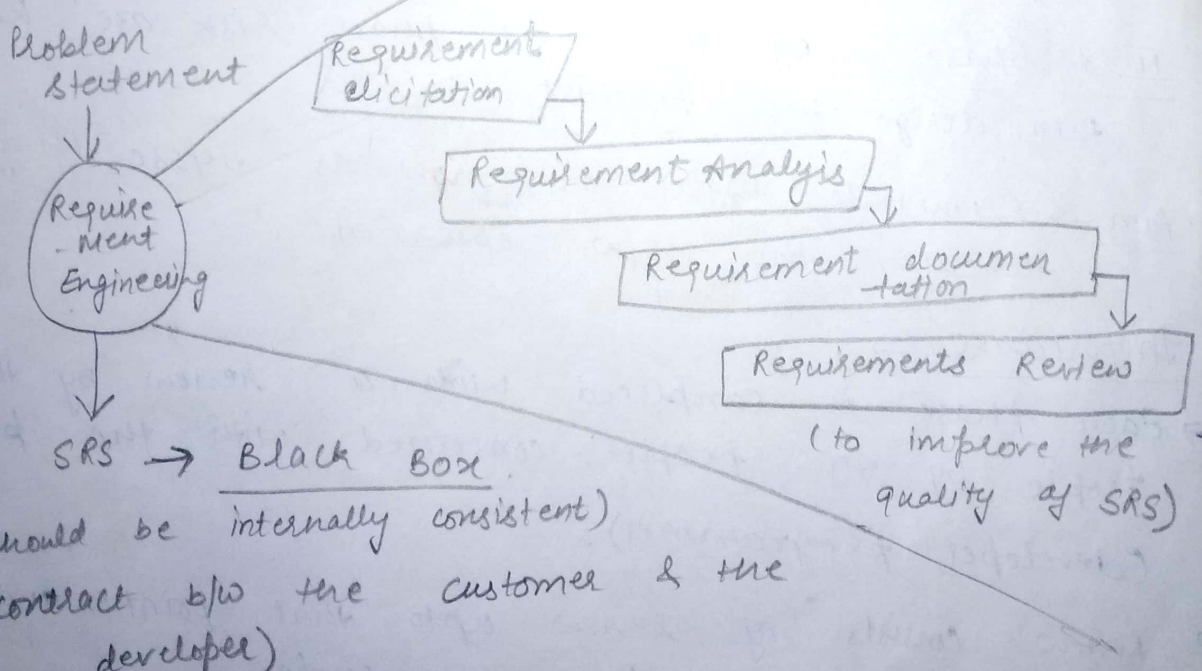
↳ Requirement Engineering :→

* It produce one large document, written in natural language, contains a description of what the system will do, not how, it will do.
* Input to RE is problem statement prepared by the customer.

(i) Crucial process steps :

quality of the s/w is only as good as the process that creates it.
Requirement engineering is the most crucial ~~process~~ activity in this creation process.

without this customer donot know what to expect → developer do not know what to build.



Primary output → Requirement specification

(ii) Present state of Practice →

Most s/w development organizations agree to the fact that there should be a set of activities called requirement engineering and these success is vital to the success of the entire project.

Why the state of Practice no better than it is :→

(*) Requirements are difficult to uncovers.

(*) Requirements change :-

(*) over Reliance on CASE Tools

We must not rely on requirement engineering tools, without first understanding and establishing requirements engineering principles, techniques and processes.

Must have realistic expectations from the tools.

(*) Tight project schedule

(*) Communication barriars; RE is communication intensive activity. (b/w user & developers)

(*) Lack of Resources :→ Rank Requirements (most imp. Can be implemented first)

→ Requirement Elicitation :→

→ Requirement Elicitation is perhaps the most difficult most critical, most error-prone, and most communi-cation intensive aspect of s/w development.

→ Elicitation can succeed only through an effective customer-developer relationship.

→ Real requirements resides in customer's mind. Hence, most important goal of requirement engineering is to find out what users really

→ Requirements are gathered by asking questions, writing down the answers, asking other questions etc.

→ Requirement Gathering is the most intesive activity of s/w development.

Requirement Elicitation Methods :→

(A) Interviews :

→ After receiving the problem statement from the customer, the first step is to arrange a meeting with the customer.

→ objective of conducting an interview is to understand the customer's expectations from the s/w.

→ Both parties have different feelings, goals, opinions, vocabularies, understandings, but one thing is common, both want the project to be a success.

→ Interviews may be open-~~minded~~ ended or structured.
* In open ended, there is no pre-set agenda.
* context-free questions may be asked to understand the problem and to have an over-view of the situation.

→ For a Result Management System :→

↳ Who is controller of Examination
↳ Who has requested for such a s/w.
↳ How many officiers are placed in the examination divisons
↳ Who will use the s/w.
↳ Who will explain the manual system.
↳ Is there any opposition for this project
↳ How many stake-holders are computer friendly.

→ Type of stakeholders →

(i) Entry-level personnel :- They may not have sufficient knowledge and experience, but can give fresh ideas & useful views.

(ii) Mid-level stakeholder :- have better domain knowledge and experience of the project. They know the sensitive, complex, critical areas of the project.

(iii) Managers → their knowledge may provide different but rich information for s/w development.

(iv) User of the s/w → spend more time interacting with the s/w than any one else.

⇒(B) Brainstorming Sessions :→

→ It is a group technique that may be used during requirements elicitation to understand the requirements.

↳ It promotes creative thinking, generates new ideas, and provides platform to share views.

↳ All participants are encouraged to say whatever ideas come to mind, whether they seem relevent or not.

↳ This group technique may be carried out with specialised group like actual users, middle-level managers etc.

↳ The facilitator is responsible for smoth conduct of brainstorming sessions.

→ Every idea will be documented in such a way that everyone can see it.

→ A detailed report is prepared and facilitator review this report.

→ Incomplete ideas may be listed separately and should be discussed at length to make them complete ideas. Finally, a document is prepared, will

customer.

(c) FAST → Facilitated Application Specification Technique :→

↳ Its objective is to bridge the expectation gap - a difference b/w what developer think they are supposed to build and what customer think they are going to get.

→ Encourages the creation of a joint team of customers and developers.

guidelines for FAST →

↳ Arranage a meeting at a neutral site.
↳ Establishment of Rules for participation
↳ Prepare an informal agenda
↳ Appoint a facilitator to control the meeting
↳ Prepare worksheets, flip charts etc.
↳ Participants should not criticize.

Preparations :→

↳ Each FAST attendee make a list of objects that are
: part of the environment
: Produced by the system
: used by the system.

Activities :→

↳ Each participant present his/her list of objects, services, constraints, List may be displayed on white board, projector.
↳ Combined list for each topic is prepared.
↳ Again discuss combined list and finalized by facilitator.
↳ Team is divided into smaller sub-teams, each then presents mini specifications for list entries.

really wants. It looks
stem.

↳ An issue list is prepared, which will be discussed later.

↳ A subteam may be asked to write the complete draft specification using all inputs from the FAST meeting.

(D) QFD → Quality function Deployment :→

→ Quality Management technique that helps to incorporate the voice of the customer.

→ Voice is then translated into technical requirements. Result is SRS document.

→ QFD emphasizes on understanding of what is valuable to the customer and then delivers these values throughout the s/w engineering process.

Types of Requirements :

(a) Normal Requirements : like entry of marks, calculat°, display the result etc.

(b) Expected Requirements : These are implicit to the s/w product but customer does not explicitly state them.
-eg→ Protection from unauthorized access, some warning system for wrong entry of data, feasibility for modification of data.

(c) Exciting Requirements → Some features go beyond the customer's expectation and prove to be very satisfying when present.
-eg→ If an unauthorized access is noticed by the s/w, it should immidiately shutdown all the processes, and an email is generated to the system admin, additional copy of important files.

Steps of QFD :→

(i) identify all stakeholders, & initial constraints. (cost & size)

(ii) List out requirements from customer.

really wants.
stem.

→ A degree of importance is assigned to the ~~importance~~ requirement, on a scale 1 to 5.

5: very imp
4: important
3: not imp, but nice to have
2. not imp.
1. unrealistic, requires further exploration.

It may be based on cost/benefit.

(E) Use-Case Approach (are graphical representation)

This approach use a combination of text & pictures in order to improve the understanding of Requirements.

(b) These are structured outline or template for the description of user requirement. } use cases

→ Actors :- • external agent, lies outside the system but interacts with it in some way.
• can be an ~~actor~~ person, machine or an info system that is external to the system model.
• not part of the system itself.

• uses devices to communicate with the system.

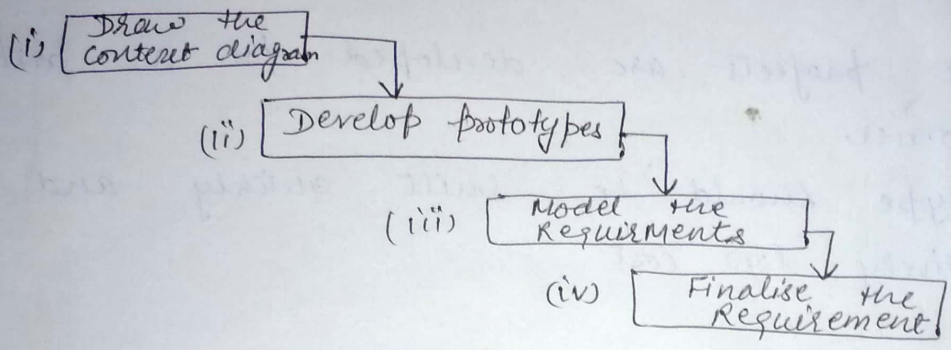
Requirement Analysis :→

↳ After elicitation it is very important and essential activity.

↳ Analyze and refine the gathered requirements in order to make consistent and unambiguous requirements.

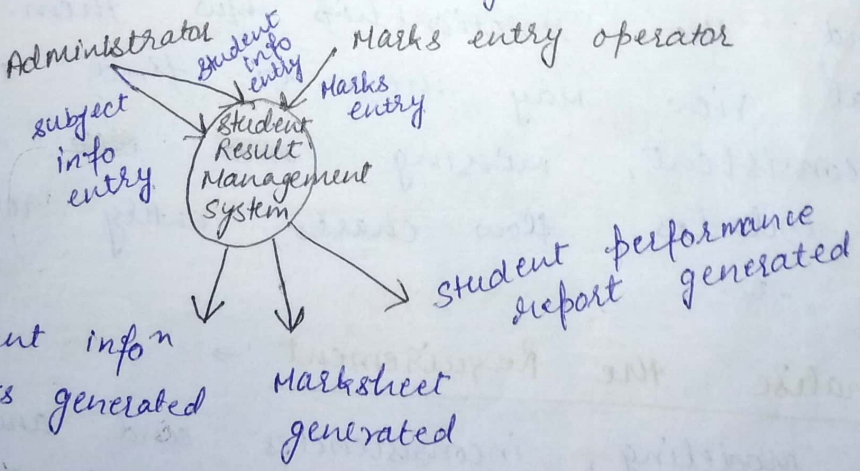
↳ It review all requirements and may provide a graphical view of the entire system.

↳ It improves the understandability of the project.



(i) Draw the context diagram :→

→ identifies the entities outside the proposed system that interact with the system.



context diagram shows the interface of the proposed system with External world.

context diagram of student Result Management system.

(ii) Development of Prototype :→ construct a prototype to find out, what the customer really wants. It looks

its

Completes

→ continuously Modify the prototype until the customer is satisfied.

→ When developers and users are not certain about some of the requirements, a prototype may help both the parties to take a final decision.

→ Some projects are developed for general Market. A Sample is shown to the representative population. Their feedback may allow us to make the product more attractive to others.

→ Some projects are developed for a particular customer.

→ Prototype should be built quickly and at a relatively low cost.

(iii) Model the Requirements :→

It usually consist of various graphical representations of the functions, data entities, external entities and the relationship b/w them. The graphical view may help to find incorrect, inconsistent, missing requirement. It includes flow charts, entity relationship diagrams etc.

(iv) Finalise the Requirement :→

→ In Modeling, inconsistencies and ambiguities have been identified and corrected. Flow of data amongst various modules has been analysed. Now finalise these requirements and next step is to document these requirements.

E-R Diagrams → Entity Relationship diagrams :-

* It is a tool for Requirement Analysis.
* Detailed logical representation of data for an organisation and uses three main constructs i.e data entities, relationships, and their associated attributes.

1. Entities → An entity has its own identity, which distinguishes it from each other entity. An entity is a fundamental thing of an organisation about which data may be maintained.

eg- [student] [Programme]

An entity type is the description of all entities to which a common definition and common relationships and attributes apply.

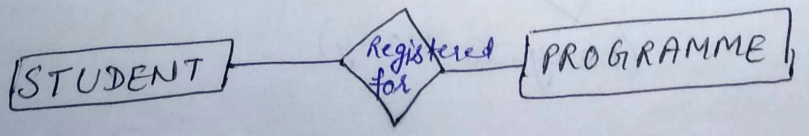
→ eg: consider a university offers both regular & distance Education programmes, To national & international students.

Entity types < student, Programme | National & International are entities of Student

NOTE: ENTITY TYPE always written in caps.

[STUDENT] [PROGRAMME]

2. Relationships → A reason for associating two entity types.

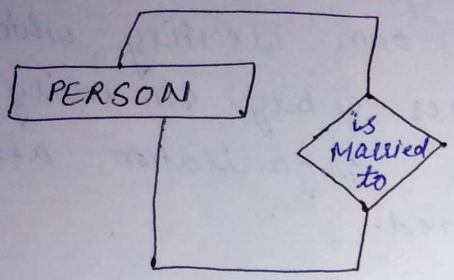


(ii) eg. A university is interested in tracking which subjects each of its students has completed.

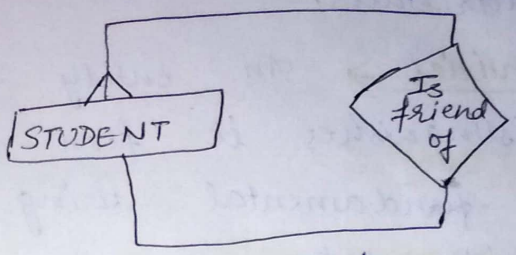
Degree of Relationship → Number of Entity types that participate in that relationship.

1. Unary
2. Binary (degree 2)
3. ternary (degree 3)

1. **Unary Relationship** :→ Also called Recursive Relationship. Relationship b/w the instances of one entity type.

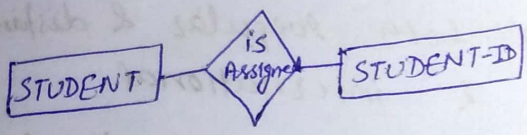


one-to-one



one to Many

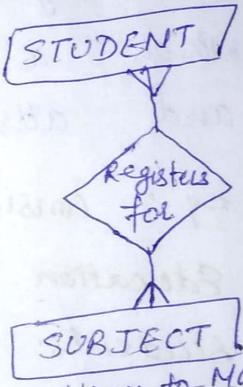
2. **Binary Relationship** :→ Relationship b/w instances of two entity types.



One-to-one

case 1

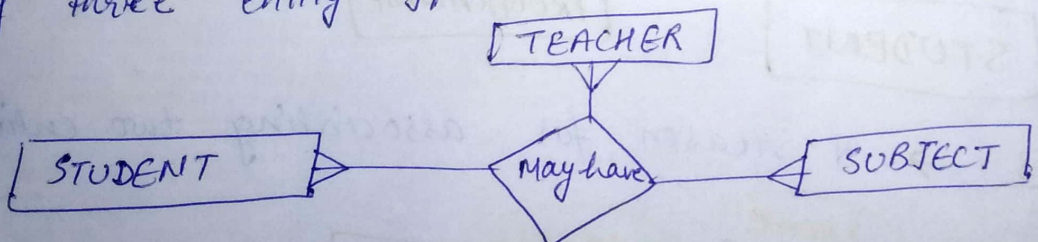
Case II
One-to-Many



Many-to-Many

Case III

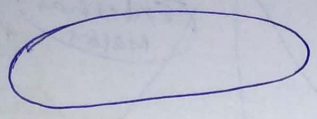
3. **Ternary Relationship** :→ Relationship amongst instances of three entity types.



(Many-to Many)

* Attributes → Each entity type has a set of attributes associated with it. An attribute is a property or characteristic of an entity that is of interest to organisation.

Representation → Use initial capital letter, followed by lower-case letters.

Notation → 

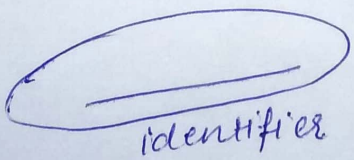
STUDENT ; Student_ID, Student_Name, Address, Phone_Number

* CANDIDATE KEY :→ A candidate key is an attribute that uniquely identifies each instance of an entity type.

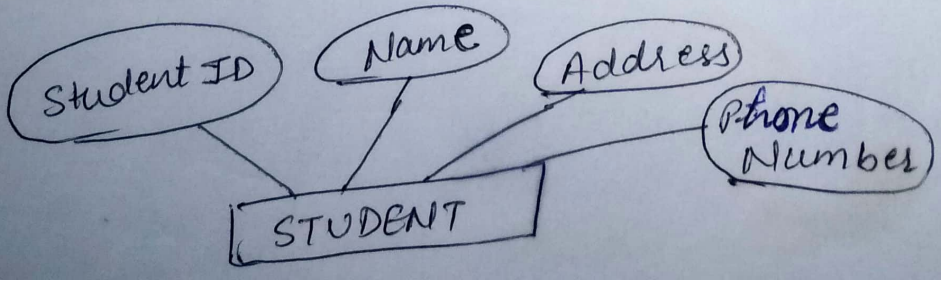
A candidate key for a STUDENT entity type might be student_ID.

* Some entities may have more than one candidate key. Like one candidate key for EMPLOYEE is Employee_ID & second is combination of Employee_Name & Address.

* Identifier :→ There is more than one candidate, the designer must choose one of the candidate key as the identifier → (unique characteristic for an entity type).



→ Representation



* Entity Relationship diagram for student - ~~Result~~ COURSE Management System.