=> Software Requirements:- Any function, constraint or other property that must be provided, met, or satisfied to fulfill the needs of system's intended user.

=> categories/Types of S.R

① General Requirements:- The very general requirements which set out in broad terms what the system should do.

② Functional Requirements:- It defines the part of the systems functionality and features that the system must pergain

③ Non-functional Requirements:- constraints and qualities that put limits on the systems development or describe how well it should perform

④ Pergormance Requirements:- minimum acceptable pergamance ~~standards~~ that the system must meet.

⑤ Implementation Requirements:- specification on how the system must be implemented or the technologies to be used.

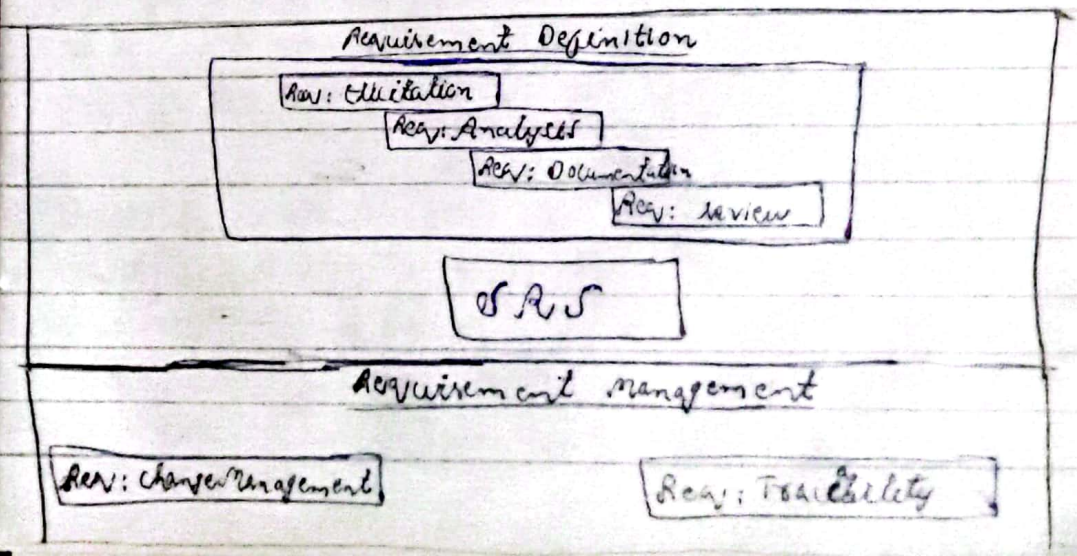
⑥ usability Requirements:- It specify the max acceptable time to demonstrate the use of the system or criteria related to the ease of use and user experience of the system.

=> Requirement Engineering:- It can be defined as the systematic process of developing requirements through an iterative-cooperative process of analyzing the problem, documenting the resulting observations in a variety of representation formats, and checking the accuracy of the understanding gained.

-> RE can be divided into 2 main sets of activities:

(i) Requirement Defination

(ii) Requirement Management



1) Requirements Definition

- Req: Elicitation:- simply gathering of reqs.
- Req: Analysis & modeling:- utilizing tools like dfd, erd, use cases, flowcharts and object-oriented modeling for modeling and analysis.
- Req: documentation:- compiling the gathered and modeled reqs into SRS.
- Req: Review:- stakeholders review and approval of the SRS, forming the foundation for subsequent software life cycle phases.

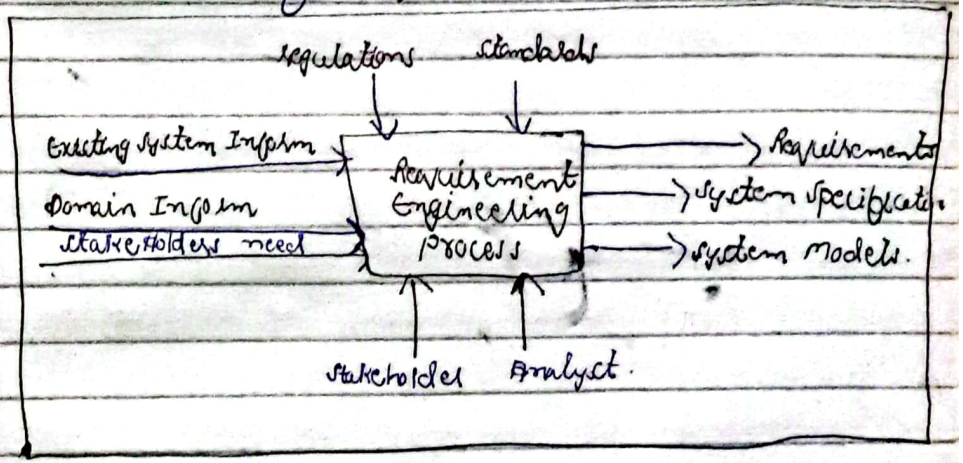
2) Requirements Management:-

- Req: change management:- systematic handling of changes to agreed-upon requirements (SRS) during the project.
- Req: Traceability: maintaining traceability b/w the SRS and downstream work products (e.g. design documents, test plans) created in the software life cycle.

The systematic approach ensures a comprehensive understanding of customer needs, feasibility assessments, and effective handling of changes throughout the SDC.

Requirement Engineering process

Traditionally, RE initiates at the beginning of the system development life cycle. In the context of large and complex systems, the challenge lies in crafting a stable set of reqs throughout development. RE adopts an incremental and iterative approach performed parallel with other system development activities. RE activities covers the entire system development life cycle.



Inputs.

- Existing system Infos:- Information about system that either will be replaced by the proposed system or which the system must interact with.
- stakeholder needs:- Detailed description of needs of stakeholders

- Domain Info's:- General information about the domains nature & covered activities.
  Controls
- organizational standards: standards used in the organization to coordinate system development or maintain quality.
- Regulations:- External regulations influencing problem consideration and solution evaluation

Mechanism (People who Involved)

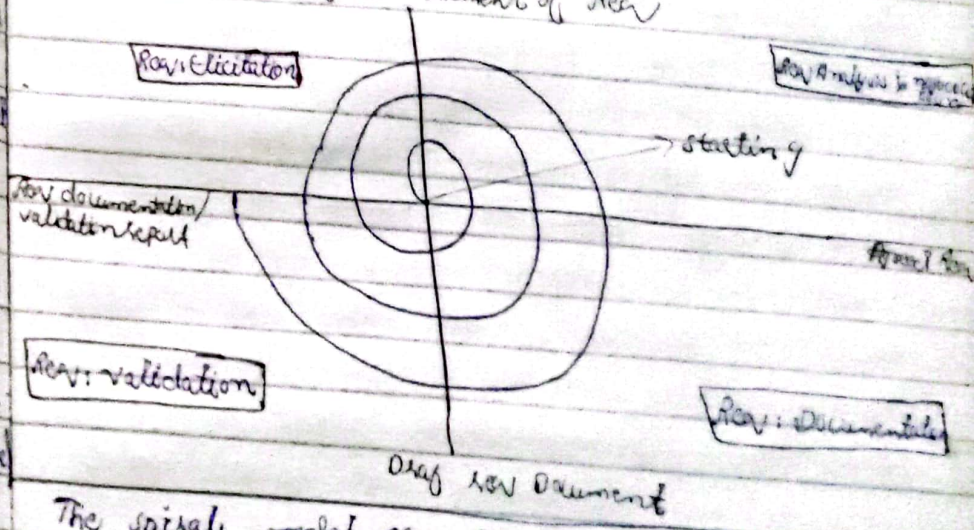
- Stake Holders: Individuals or organization affected by the system with direct influence on reqs.
- Challenges: Difficulty articulating needs, unrealistic demands, expressing reqs in subjective terms
- Analysts: Project team member responsible for managing the reqs process (requirements manager)

Outputs ▼

- Requirements:- Agreed upon functional & non-functional ~~reqs~~ system reqs
- System specification:- A more detailed version of the system, if needed.
- System models :- Diagrammatic representation describing the system from necessary perspectives

=> Spiral model of Req: Engineering Process

The spiral model of req: Engineering process emphasis on continuous reassessment of the elicitation and analysis and combines the iterative and sequential approach. Here a number of task regions are defined and these task regions are traversed in each iteration.



The spiral model of req: Engineering is devided into 4 number of tasks regions.

1. Req: Elicitation:- Task required to gather the reqs and after the completion of this task we get the informal statement of the requirements
2. Analysis & Negotiation:- Task

=> Software Requirements :- Any function, constraint or other property that must be provided, met, or satisfied to fulfill the need of system's intended user.

=> categories / Types of S.R

① General Requirements :- The very general requirements which set out in broad terms what the system should do.

② Functional Requirements :- It defines the part of the systems functionality and features that the system must pergorm.

③ Non-Functional Requirements :- constraints and qualities that put limits on the systems development or describe how well it should perform

④ Pergormance Requirements :- Minimum acceptable pergormance ~~standards~~ that the system must meet.

⑤ Implementation Requirements :- specification on how the system must be implemented or the technologies to be used.

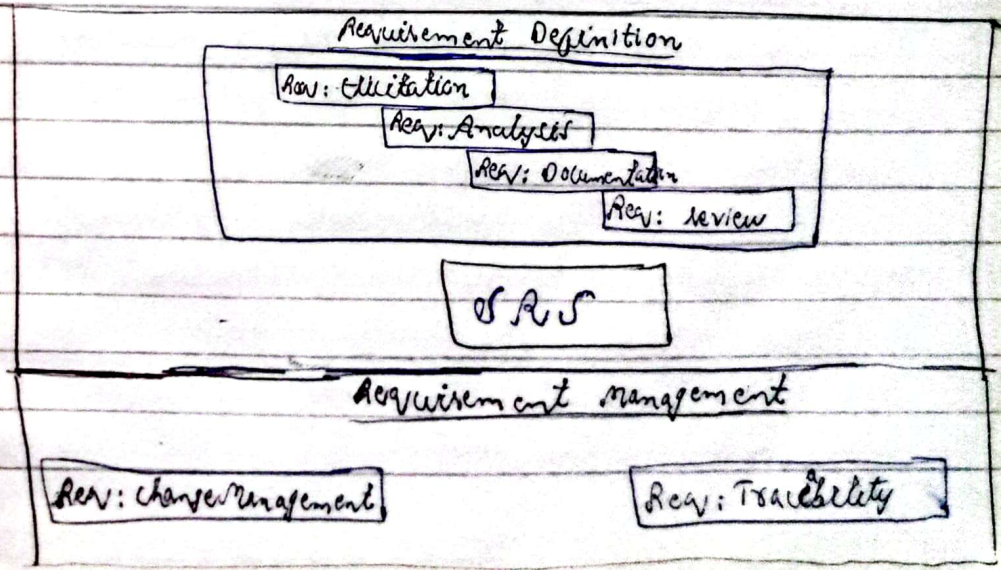
⑥ Usability Requirements :- It specify the max acceptable time to demonstrate the use of the system or criteria related to the ease of use and user experience of the system.

=> Requirement Engineering :- It can be defined as the systematic process of developing requirements through an iterative-cooperative process of analyzing the problem, documenting the resulting observations in a variety of representation formats, and checking the accuracy of the understanding gained.

-> RE can be divided into 2 main sets of activities :

(i) Requirement Defination

(ii) Requirement Management



1) Requirements Definition

- Req: Elicitation:- simply gathering of reqs.
- Req: Analysis & modeling:- utilizing tools like dfd, erd, use-cases, flowcharts and object-oriented modeling for modeling and analysis.
- Req: documentation:- compiling the gathered and modeled reqs into SRS.
- Req: Review:- stakeholders review and approval of the SRS, forming the foundation for subsequent software life cycle phases.

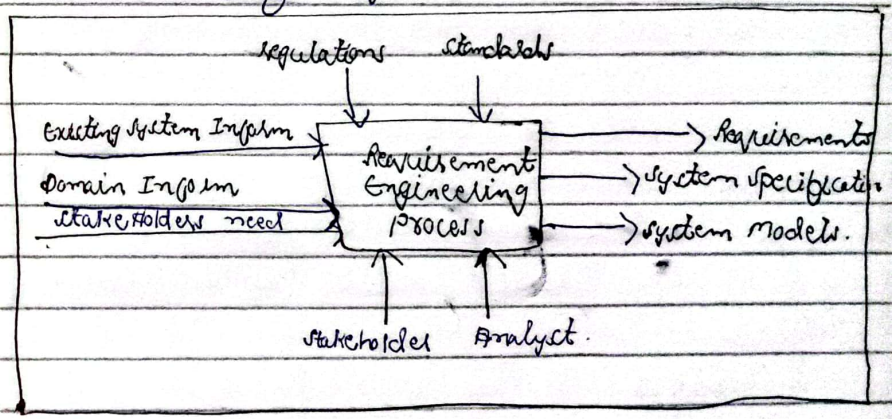
2) Requirements Management:-

- Req: change management:- systematic handling of changes to agreed-upon requirements (SRS) during the project.
- Req: Traceability:- maintaining traceability b/w the SRS and downstream work products (e.g. design documents, test plans) created in the software life cycle.

The systematic approach ensures a comprehensive understanding of customer needs, feasibility assessments, and effective handling of changes throughout the SDLC.

Requirement Engineering process

Traditionally, RE initiates at the beginning of the system development life cycle. In the context of large and complex systems, the challenge lies in crafting a stable set of reqs throughout development. RE adopts an incremental and iterative approach performed parallel with other system development activities. RE activities covers the entire system development life cycle.



Inputs.

- Existing system Infos:- Information about system that either will be replaced by the proposed system or which the system must interact with.
- stakeholder needs:- Detailed description of needs of stakeholders

- Domain Info's:- General information about the domain's nature & covered activities.
  Controls
- organizational standards: standards used in the organization to coordinate system development or maintain quality.
- Regulations:- External regulations influencing problem consideration and solution evaluation

Mechanisms (People who Involved)

- Stake Holders: Individuals or organization affected by the system with direct influence on reqs.

Challenges: Difficulty articulating needs, unrealistic demands, expressing reqs in subjective terms.

Analysts: Project team member responsible for managing the reqs process (requirements manager)

Outputs ▾

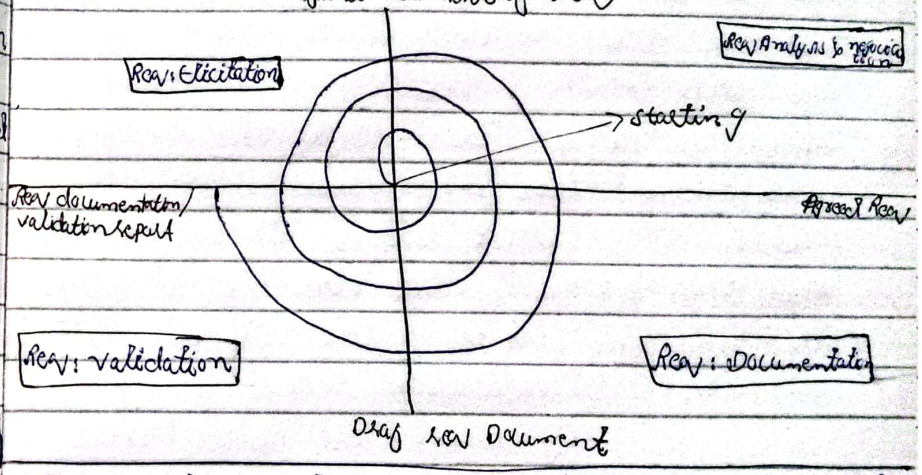
Requirements:- Agreed upon functional & non-functional ~~reqs~~ system reqs.

System specification:- A more detailed version of the system, if needed.

System models:- Diagrammatic representation describing the system from necessary perspectives

=> Spiral model of Req: Engineering Process

The spiral model of req: Engineering process emphasis on continuous reassessment of the elicitation and analysis and combines the iterative and sequential approach. Here a number of task regions are defined and these task regions are traversed in each iteration.



The spiral model of req: Engineering is divided into 4 number of tasks regions;

(1) Req: Elicitation:- Task required to gather the reqs and after the completion of this task we get the informal statements of the requirements.

(2) Analysis & Negotiation:- Task required to

analyze and model the gathered reqs. Analysts categorizes reqs and organizes them into related subsets, and ranks the reqs based on the needs of the customer/users.

③ Req Documentation :- Task required to document the gathered and modeled reqs at the end of this activity/task region we get the (SRS).

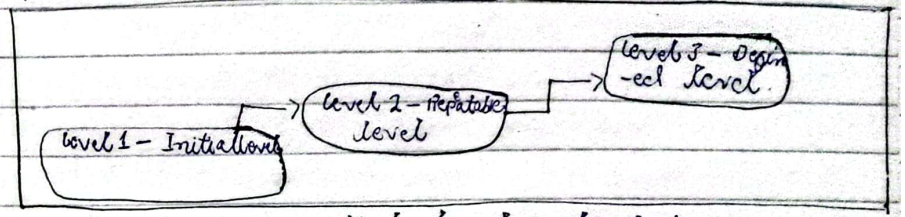
④ Req validation :- Task required to examine the specification to ensure that all system reqs have been stated completely and unambiguously. At the end of this task region we get the final reqs document & validation report.

By using the spiral model software is developed in a series of incremental releases.

⇒ RECMM

→ It stands for requirement engineering capability maturity model. It defines the standards for req documents and req descriptions. It uses automated tools to support process activities. It also used for management of policies and procedures.

→ RECMM has 3 levels:



① level-1 - Initial level :- level-1 organizations do not has well defined set of reqs engineering process. They don't use advance methods to support their reqs engineering process. They often fail to produce good quality reqs document on time and within budget. They are dependent on the skills & experience of individual engineers for reqs elicitation, analysis & validation.

② Level-2 - Repeatable level :- level-2 organizations have defined standards for requirement documents and descriptions and they introduce policies & procedures for req management. They may use advance tools & techniques in their RE process. Their req documents are more likely to be consistently high quality and

to be produced on schedule,

③ Level-3 - Defined-level:- Level-3 organizations have a defined req engineering process model based on good practices and techniques. They have an active process improvement program in place and can make objective assessments of the value of new methods and techniques.

⇒ Req: Engineering life cycle

→ The reqs engineering life cycle basically consists of 3 main activities, which are :-

① Feasibility study :- Determine if the proposed system is worthwhile or not.

Key Question: • Is the overall objective satisfied?
• Can the system be implemented using current technology and cost factors?
• Can the system be integrated with existing systems?

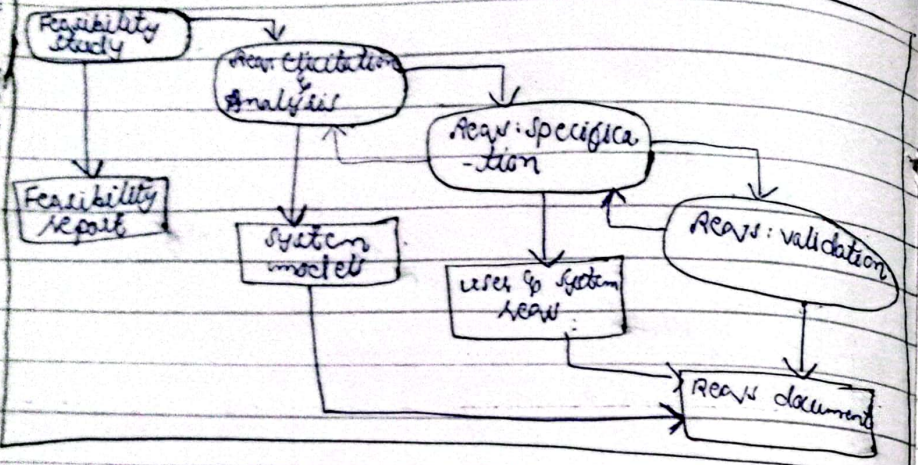
② Elicitation and Analysis :- understand the application domain, required services, system performance, and hardware constraints

Process Activities:

• Domain understanding :- workflow study, Identifying stakeholders and defining solution boundaries and constraints.
• Req collection :- understanding Request for proposal (RFP), conducting interviews and questionnaires, workshops / Brainstorming sessions, Analysis.
• Classification :- using use-case examples to categorize reqs.
• Prioritization :- discovering important reqs through stakeholder interaction & prioritizing them.
• Reqs checking :- Ensuring completeness and consistency.

③ validation & Refinement :- Ensure that reqs accurately define the desired system

• checks :- validity checks, consistency checks, completeness checks, Realism checks, verifiability
• Activities :- perform simulations, modeling, & Component decomposition.
Resolve conflicting reqs, specify baselines reqs.
• Output :- Refined, verified, and validated specification of Reqs.



=> Req Elicitation & it's types.

Req Elicitation:- It might be described as eliciting a specification of what is required by allowing experts in the problem domain to describe the goals to be reached during the problem resolution.

Types are listed below:-

1) Greenfield Engin:- Development starts from scratch, no prior system exists, the reqs are extracted from the end users and the client.
Triggered by user needs.

RE-Engineering:- Re-design/re-implementation of an existing system using newer technology.
Triggered by technology enablers.

3) Interface Engineering:- Provide the services of an existing system in a new environment.
- Triggered by technology enablers or new market needs.

=> Traditional Techniques of Req: Elicitation

1) Questionnaires:- To conduct the questionnaires session, the interviewer has to prepare the list of questions which should be ask via email, telephonic call, video conference or interview that's why this technique is said to be the questionnaires as the content free questions.

→ main strength of Questionnaires:- is that large amount of data can be gathered or collected from many users quickly. In addition, data can be collected over a wide geographical area without incurring travel expenses.

→ In expensive:- The questionnaires are relatively in expensive technique used for eliciting the reqs.

→ Rapidely Analyzed:- If the questionnaires can be are skillfully crafted, responses can be analyzed rapidely by computer.

→ Questions which must be asked

1) The Stacked question:- Should you continue to

be provided some sort of high level of services which you have already received in the past?

(ii) The leading question:- we should not be providing this type of service to you, should we?

(iii) The self answering ques:- how much time does it take to do this job?

(iv) The ambiguous quest:- Is the documentation nice.

→ The analyst starts the communication with the customer by asking the "context free question". For example analyst might ask:

- who is behind the request for this work?
- who will use the sol?
- what will be the economic benefit of a successful sol?
- Are you the right person to answer these ques?
- Are my questions relevant to your problem?
- Am I asking to many questions?
- Can anyone else provide additional information?

→ And the list will continue.

(2) Interviewing:- Interviews are designed to capture the same type of data from users as questionnaires, but they are conducted in more depth bc conducting interviews is relatively expensive, a small number of managers and users are going to be conducting the interviews. The data gathered by using the interviewing technique often provide a complete picture of the problem & opportunities to the developers. It also give analyst the opportunity to note user reaction & get further information. we take interview to make repository of the reqs for further work and explaing the solutions to the undiscovered reqs. Interview coordination with managers minimizes user work disruption, with varying success in eliciting system information, Interviews serve as a crucial bridges b/w users and developers.

Group Elicitation

(1) ⇒ JAD :- It stands for Joint Application Development It is a management process that is used to involve both users and technical staff on a project using intense concentrated and focused meetings for ensuring their collaboration. It is based on the recognition that user involvement is critical to project success and is required throughout the life cycle.

JAD sessions can be used throughout the systems development process but are particularly helpful during system planning, requirements analysis & system design.

→ There are some important roles in JAD which are discussed below:

(1) Sponsor:- is the most important role in JAD. Sponsor is the top management person whose sponsorship shows management commitment.

(2) Facilitator:- is the key person who is most directly responsible for the functioning of the JAD team, Facilitator is also called the project leader. He/she is responsible for planning, executing & managing the project. A facilitator must have leadership qualities, & must respect his team.

(3) Scribe:- also called the record keeper who are responsible to document the JAD session. The scribe has to capture all the important decisions and clearly record all identified action points, persons responsible, dates etc.

(4) Developers:- also called system/information specialist have to design the system all to the user's ways & need to see how technology can be best used for this purpose.

⇒ RAD:- stands for Rapid Application Development, is a software development process that integrates various cutting-edge approaches to accelerate system development & expedite the delivery of new systems. RAD allows usable systems to be build in as little as 60-90 days, often with some compromises. RAD is used mostly for the following purposes;

(1) To come together early towards a design, acceptable to the customer & feasible for the developers

(2) To limit a projects exposure to the forces of change

(3) To save development time, possibly at the expense of economy or product quality.

Strengths:- RAD addresses issues such as scope and requirements creep by minimizing the projects exposure to change, shortening the development cycle, and mitigating the cost of change by incorporating it upfront & prior to substantial investments in the development & testing.

Historically, RAD systems focused on speed over code efficiency, but modern RAD

systems produce gost code. Traditional environments now offer visual tools, blurring the distinction between RAD & other methods.

⇒ Brainstorming:- It involves both idea generation & idea reduction, often resulting in the most creative & innovative solutions by combining seemingly unrelated ideas. It can be conducted individually or in groups, with the latter often being more productive. The term "tink-tank" refers to the group engaged in brainstorming

→ Procedures to do it:-

① In a small or large group first select a leader and a recorder.

② Define the problem or idea to be brainstormed.

③ Make sure everyone is clear on the topic being explored.

④ Set up the rules for the session. They should include:
- Let the leader have control.
- Allowing everyone to contribute.
- Stating that no answer is wrong.
- Recording each answer unless it is a repeated.
- Setting the time limit.

⑤ Start the brainstorming sessions, participants share their ideas while the recorder documents them. No evaluation or criticism is allowed until the brainstorming is complete.

⑥ After brainstorming, evaluate the ideas by eliminating duplicates and those that don't fit. The remaining ideas can then be further refined or implemented.

⇒ Story Boarding:- serves as an invaluable tool in software development for gathering the early feedback from users on proposed application concepts. It's a cost effective, user-friendly, and interactive approach that allows developers to test ideas before investing in code or detailed specifications. There are 3 main types of storyboards.

① Passive storyboards:- Passive storyboards use sketches, pictures, or presentation to tell a story to users.
- The analyst acts as the system, guiding users through the story boards with "when you do this this happen" explanation.

② Active storyboards:- Active storyboards aim to give users a glimpse of "a movie that hasn't been produced yet".

- They are animated or automated, often using slide presentations, animation tools, or movies.
- Active storyboards offer an automated depiction of typical system behaviours during usage or operational scenarios.

(3) Interactive storyboards :- They allow users to experience the system realistically.
- users actively participate, often through simulations, mock-ups, or throwaway code
- Advanced interactive storyboards closely resemble with throwaway prototypes and require user engagement for execution.

=> Prototyping :- is a crucial phase in software development, allowing for the exploration and communication of ideas b/w users & analysts. with the help of prototypes stakeholders can visualize & better understand the proposed system. Prototypes serve as partial representations of the final product & facilitate early feedback from users and developers, improving requirement communication and refining system understanding.

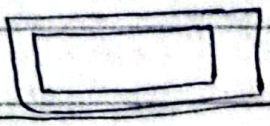
-> To maximize the benefit of Prototyping, it should be conducted systematically following these steps:

(1) Establish Goals for Prototyping :- clearly define the objectives of the prototypes, such as discussing the user interface, establishing technical feasibility, or demonstrating functionality.
- Lack of clarity on objectives may lead to misunderstanding during the prototype development

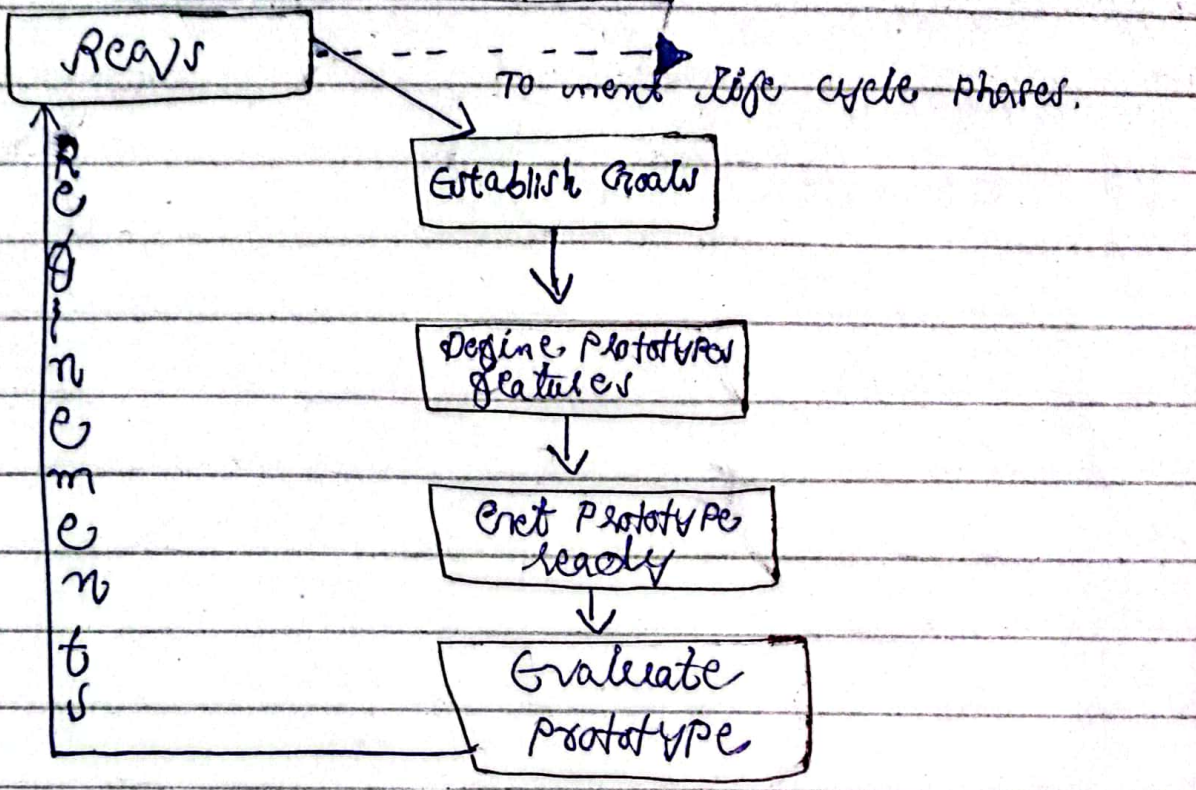
(2) Define Prototype features & functionality:- Document and discuss a detailed list of features and functions to be demonstrated in the prototype.
- clearly state what will be included and what will not included in the prototype to manage expectations.

(3) Develop the Prototype :- Create the prototype acc to the defined features & functionality
- Review the prototype to ensure that all required features & functionality are developed.

(4) Evaluate the Prototype :- Execute the prototype and collect feedback from users & developers.
-> Feedback may include new ideas, changes to the user interface, or dropping of ideas.
-> use the evaluation report to elicit & analyse new ideas.



Prototype diagram.



2 main approaches of prototyping.

1. Evolutionary Approach:- Also known as open ended approach.
- The prototype is built to be used in the construction phases.
- Allows for the development of a considerable part of the system during the reqs & design stages.

② Throw away Approach:- Also known as closed-end-ed approach.

- Prototyping ends once all reqs and clearly documented.
- The prototype is not used for the actual construction of the system and serves only as a demonstration of reqs.