Software Engineering

① Confidence
② Creativity
③ Character
Soft skills (Life skills)
i) Communication skills
↳ Kings skill (in form of umbrella)
⊙ Analytical thinking and innovation
(Edward De Gono → skills you need (website))
⊙ Active learning and learning strategies
⊙ Complex problem solving
→ multiple options done (don't think during runtime)
⊙ Critical thinking and Analysis
⊙ Creativity, originality and initiative
⊙ Leadership and social influence (team management)
→ key skill
→ makes impossible → possible
⊙ Technology use monitoring and control
⊙ Technology design and programming
⊙ Resilience, stress tolerance, anger management, time management and flexibility
⊙ Reasoning, ideation.

Learning:

Data → Information → Knowledge → Learning → Skill

- Data → Raw facts → Scattered form
- Information → Procesed data
- Knowledge — علم نافع
  - "وہ علم جس سے آپکو اور آپ سے منسلک لوگوں کو فائدہ پہنچے" (حضرت علیؓ)
  - علم بغیر عمل جہالت ہے (واصنف علی واصنف)
- Learning → specific
  → independent of source / teacher.
- Skill → application of learning.

To gain knowledge, learning, skill
→ A foundation is needed.

① ادب → اگر آپ سیکھے اور سیکھیں تو استاد کا معیار معنی نہیں رکھتا
② خلوص

Software:

→ Quality of a software is determined by these → ① size and ② complexity of instructions level will change.

- Triple constraints (For professional software development)
  1. Cost
  2. Time
  3. Requirements (Scope)

  } % depend upon nature of software.

Professionally Software → Enterprise Software (3 points make its defination)

↳ It has

1. Instructions
2. Data Structures (Data Base)
   - ↳ generate during runtime, manipulate data and then are deleted. e.g stack, trees, arrays, etc.
   - ↳ Permanent storage and use when needed e.g in Hard disk
   - ↳ Storage of data for future addition, use and deletion.
3. Documentation

- DBMS (Data Base Management System)
  ↳ Data Base used as software.

- Simple solution of problem of complexity is documentation.
- Two types of documentation
  1. Technical
     → From idea initialization to its use all its ~~utilize~~ documentation is technical.
  2. User

1. Analysis → we try to get answer of WHAT?
2. Design → On paper or in mind → a solution is made (e.g flowchart, algorithms, etc)
3. Code → Given syntax of programming language.
4. Testing

Parts of Course
① Fundamentals
② Structured analysis and design
③ Object Oriented approach
④ Testing

IEEE → Institute of Electrical and electronic engineers

Software Engineering:

Its an Approach : ① Systematic → Arrangment of things according to your goals. → Its static
② Disciplined → Its not static during execution → Constraints or limitations under which system is executed → executed at lowest level. e.g if symbol not used correctly → voilation of discipline → Saves cost and time
③ Quantifiable

Static in terms of content

For ① Development
② Operation
③ Maintainance of Software

③ Quantifiable:
Working with numerics, data → for clearity → needed for effective decision.

Maintainance

Types:

① **Enhancement**

After launching addition of features on customer's demand.
E.g updation of softwares

② **Coorection**

Same nature as testing
↳ during development after coding some errors are left.

Done during use (errors found during use)

③ **Adaptation**

Large scale (complete)
E.g softwares for Windows won't work on LINUX an overall change will have to be made.

Smaller scale → Flexibility
↳ should remain aware
↳ adjust acc to environment

E.g: Nokia (e.g of non-flexibility)
Blackberry

④ **Prevention**

- Not a typical type
- It an approach, mindset.
- If properly followed first 3 things won't be needed.
- Management perspective

- Reactive Approach
- Proactive Approach

} Part of prevention

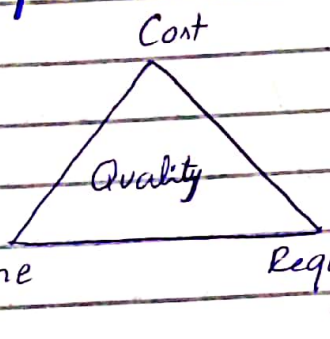
Proactive approach: پیش عملی
- Be ready before time for problems.
E.g. having an extra pair of glasses.
- It saves time, energy, resources providing you useful things
- Needed for success

Reactive approach
- On time approach. (instantaneous) (without proactive)
- More time, energy, resources needed.

If predetermined ⟶ Responsive approach
↳ If proactive

- Reactive and proactive relationship play key role in quality.

Triple Constraints



Quality: trade up of
- Determined on basis of tiple constraints for a particular customer.
⇒ called Customer Satisfaction
- Quality can't be the excellence everytime

⇒ صومن وہ ہے جس کو آنے والا دن پچھلے دن سے بہتر ہے (نبی پاک ﷺ کا فرمان)
⇒ Continuous Improvement
(ISO, CMM, 6 Sigma, Tic IT)
⇒ Fitness for use
Enterpise software becomes "Quality Enterpise Software".

Software → Enterprise software → Quality Enterprise Software
(has size and complexity)

Characteristics of Software

① Software is engineered, not constructed.

Engineering → Software
Construction → Physical (Hardware)

→ Difference of Hardware and Software.

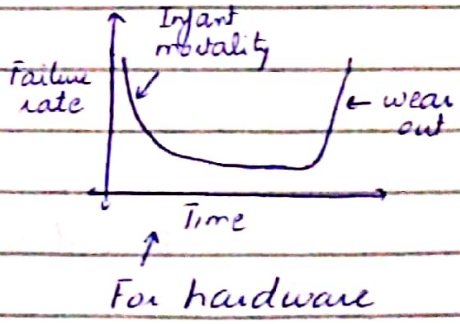
Work at top → Engineering (More mental process involved)
Implimentation physically → Construction

Dealing with software (intangible activity) → Software Engineering.

① Knowledge
② Skill
③ Abilities
} Combination Needed for development

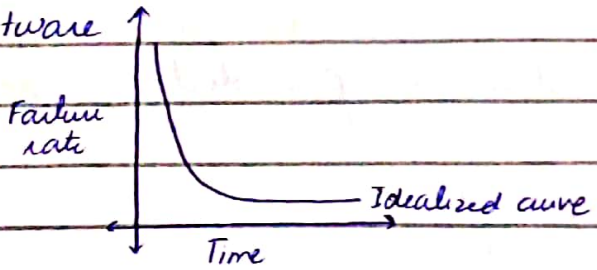
For it (i) analyze yourself

② Software doesn't wear out but it deteriorates.



For hardware

For Software



- Triple constraints → same → software stat stays
  ↳ if requirements changed → software deteriorates.

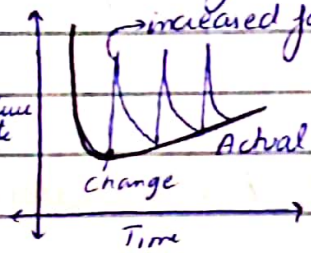
Requirements:

Types
① Functional
② Non-functional

① Functional Requirements:
→ Business logic of software
→ Generic
E.g accounting system
calculator (addition, subtraction, multiplication, division)
Functional Requirements.

② Non-functional Requirements.
They are constraints or limitations which are applied on functional requirements
E.g. cash is constraint on payment
↳ Non-function req. ↳ Functional requirement
E.g. Security, consistency, maintainability, performance, etc.

Software curve.



due to changes in requirements.
To manage deterioration → models are provided by Software Eng.
⇒ Change is inevitable
(Software Configuration Management)
Flexibility, adaptability is needed.
Agility → Flexible to deal with situations.
↳ minimizes gap between curves of failure.

③ Although industry is moving toward component based development most software continue to be custom built.
↳ From scratch

(Component based Software Engineering / Component based developme

Component as has:

① Independent functionality

② Interface (to interact)

④ Software is complex

Software is an abstraction of real life.

If you want to flourish → be ready to develop everything.

Software Eng. provides necessary knowledge, skill and abilities.

(Component based Software Engineering / Component based development)

Component or has:

① Independent functionality

② Interface (to interact)

④ Software is complex

Software is an abstraction of real life.

If you want to flourish → be ready to develop everything.

Software Eng. provides necessary knowledge, skill and abilities.

Why do we study S.E?

To develop quality software for the need of humanity.

Process

- Steps involved to achieve a substantial goal.

=> Sequential steps that are followed to achieve a substantial goal.

Software Process (also called Software Development Lifecycle or SDLC)

Sequential steps that are followed to achieve an enterprise quality software (follows triple constraints)

Steps:

① Initiation
② Planning
③ Analysis
④ Design
⑤ Coding
⑥ Deployment
⑦ Maintainance

- System Development Lifecycle (SDLC) (check dy)

① Initiation

- 2 cases
  - → Involves market survey
  - → Finalising one software house (e.g Netsol, Techlogin, Systems)
    Largest S.H of Pak 5th level
- (First step) Proposal needed → detail of triple constraints
  - → Provided by customer
  - → Functional requirements stated
- Last step of initiation is signing contact
- Small but most important phase
- Kicks off or provides starting authority to a software project.
  (Project Manager is assigned the project)
  - ↳ takes detail.

② Planning

- It will be done by Project Manager (PM)
- PM makes a team according to ~~need~~ need (Planners needed in this phase)

⓪ Scheduling

① Task (What?)

Requirements are refined using tools so that they have a connection.

- → Most important
- → Influences other steps
- → Most imp tool used → WBS
  - ↳ Work Breakdown System

② Time.

- Time duration is mentioned with each task.

③ Dependency

- → Sequence of tasks

→ Important when multiple people involved.
→ Tasks when done parallel → Less time required.
↳ not always possible.

④ Assignment of resources
Two types of resources
→ Human
→ Material
Both are to be assigned but more critical are human resources.
⇒ Principle:
"Right person for right job at the right time."
→ Most common mistake is voilating this step.
(Human Resource Management HRM)

⑤ Cost

Provides plan of obj
∴ MS Project → tool used to manage Softwares.

⇒ "If you are failing to plan, you have planned to fail"

→ Plan made is then circulated
→ Planning is not execution

③ Analysis (what?)
• Size and complexity were problems in structured programming which lead to use of OOP.
⇒ Simultaneous control of data and function
① Decision of approach structured or object oriented.
3 domains of Analysis
① Data ② Function
③ Behaviour

① Data
→ Characteristics
→ Their relationships
⇒ called DATA MODELING
↳ must be a diagram
↳ for structured approach is called ERD
Entity Relationship Diagram.

② Function
⇒ Function Modelling
→ Methods are identified / understood
→ Diaagram → DFD ↳ Data Fo Diagram

③ Behaviour
⇒ Behavioral Modelling → STD → State Transition Diagram
→ Behaviour of Software doesn't change during runtime
→ Behaviour → Changes
→ Attitude → constant.

④ Design:
① Data Design → Identifying data st. acc to design.
② Architectural Design
③ Component Level Design
④ User Interface Design
- Assigned to programmer
- Logic designer is not the task of programmer.

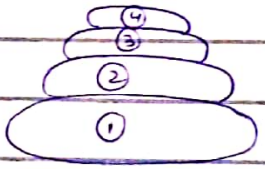
⑤ Code → Providing syntax of programming language.
⇒ Chief Programmer Team Structure

③ **Deployment** → Installing it in customer's requirement area and monitoring it for some time.

- Sequential activities → one after the other and interconnected.
- Project Management (done continuously) is done on top of A-D-C-E-T
  - ↳ Umbrella activity e.g SQA → Software Quality Assurance.
  - ↳ Measurements (done when needed)
- Umbrella activities → continuous → Project Managment
  - ↳ Need based → Measurements.

Software Engineering - A layered Technology

- 4 layers
  1. A quality focus
  2. Process
  3. Methods
  4. Tools



1. A quality focus

- Surrounds other layers
- Initialization actually / Initiation
- Constraints are told
- A proposal / content formed acc. to such constaints forms a quality focus
- Needed to be done first in order to move on.
- Clarity needed for which take as much time as needed.
- Litmus test on goal SMART

  S → Specific → Edison → 1249 inventions

  → Self discovery

M → Measurable → write it down in figures of Triple contraints
A → Achievable → doable → acc. to circumstances, resources
R → ~~Retainable~~ Realistic
T → Time bound → Start and end date decided
→ can change acc. to circumstances.

- This layer is called **Bed Rock**

2. Process

- Changes made acc. to situation
- Customization, adjustment.
- After these changes in tiple containts it becomes **Process Model**

Some sample cases
① Requirements are crystal clear
② Requirements are ambiguous (LSM)
③ Time is short
④ When you are developing a commercial software
⑤ When risks are there (uncertainities)

~~Above 5 are process models named as~~ Above 5 situations are dealt by following 5 process models

① Linear Sequential Model (LSM)
② Prototyping Model
③ Rapid Application Development Model
④ Incremental Model (all versions of Windows are made by this)
⑤ Spiral Model

- Often combination of ~~sequ~~ situations occur and new models are formed.
- This layer is called **Glue**
- This layer joins other layers

③ Methods

- Methods are present in process
- The 3 diagrams in analysis are example of methods.
- These are **How To's**
- SMART works saves us triple contraints and provides better quality.
- Methods need to be analyzed, there is a need to look for better methods
- Mindset matters more than resources

• خوش اسلوبی ، سلیقہ بنانا
adds value to your work

④ Tools

E.g to make ERD we use Ms Visio, Accers, Smart Draw.

- Using software to apply methods
- Support is taken from technology
- There are tools for various purpose
- **CASE** → Computer Aided Software Engineering

Quality Focus
Process Model
SDLC

Process Models

① Linear Sequential Model

- Basic and Simplest
- Evolved automatically
- Two main condition
  - → Problem statement is clear / Requirement
  - → No change will occur in requirement.

E.g:

Analyze → design → code → Test

Youtube Lecture:

- Small Projects
- Waterfall model

Disadvantages:

- Rigid / No flexibility
- No Feedback
- No Experiment
- No Parallelism
- High risk
- 60% efforts maintainance
- Major problems found in maintainance stage

② Prototyping Model

Disadvantages of LSM:

① Can't accomadate changes
→ doesn't support iterations
→ works straight-forward

⇒ • Core of LSM ~~with~~ is part of each model.

② Designer is blocked until analysis is completed.
→ overcame by assignment of resources in such a way that multiple tasks are done efficiently.

② Prototyping model →

- Throwaway model
- good for technical & requir. risks
- increase in cost of development.

- A sample / look alike is shown
- Feedback obtained can be positive or negative.
- In this way requirements start becoming clear.
- Prototype → look alike interface of functional software
- Requirements refined on basis of feedback → again prototype is made.
- Cycle continues until customer satisfaction.

⇒ It is not only a process model but also a technique to gather the requirements.

- Iteration
→ Evolution not necessary
→ all process models except LSM.

E.g: → Practice

- Evolution
→ iteration + improvement
⇒ all process models except LSM
→ Best case.

E.g: → Learning

- Prototypes don't consume alot of time.

Disadvantages:

⊙ A lot of overall time passes during process of clearing the requirements.
→ Inefficient algorithms are then used.
→ Students are appreciated bcz less time left.
→ Quality of Software can be compromised then.

⇒ This process model is not recommended for mature softwares.

⇒ This process model is very efficient as a requirement gathering technique.

⇒ Preferable to use it as a technique rather than a model.

Time Managment:

Types of everyday tasks

① Important ② Urgent ③ Imp and urgent
④ Imp ~~not~~ but not urgent
⑤ Not imp but urgent
⑥ Neither imp nor urgent.

Hours : 5

Yearly : 5 x 365

: 1825/12

= 152.08

: 2.5 months

① • Urgent + Important ⟹ Necesity (Duty, Responsibility)
→ No compromise
→ Have to manage

② • Important ✓ Urgent X ⇒ Quality and Personal Leadership → FOCUS
• Book Reading (min 30 min)
• Exercise
• Good Movies

→ 90% activities should be in first 2 quadrants

③ Urgent X Important ✓ ⇒ Deception → Use & caution or avoid
→ Delegate
• Personal Managment skills needed
⇒ Test → if its benefitting you or the people around you.

④ Urgent X Imp X ⇒ Waste → Avoid
• Looking busy doing nothing

③ Rapid Application Development Model

- 2 levels of time management
  - ① General time management
  - ②

- **Preconditions of RAD**
  - Requirements must be clear
  - Task should be doable
    - → Constrained scope

Yt lecture:

- Quick prototyping
- User involvement
  - ↳ Max.
- Iterative development.
- Incremental releases
- Time boxing
- Parallel developme

① General time Management

- Component based construction approach is used
  - → Useable components are already made.
  - → Saves time

  - → If useable components not available you will make them by dividing scope into fractions (components)
  - → Separate teams specified for each component.

=> Component development also called PARALLEL DEVELOPMENT

- → Problem → Resources → Team making
  - ↳ Capable, relevent, Experienced
  - ↳ Management skills
    (Project Manag., Team Manag., Proactive approa flexibility among all team members)
- → Specialized resources are needed

=> Customization of SDLC acc. to circumstances.

RAD Model:

- Busines Mod. + Data Model + Proces Modeling → Analysis and Design
  → Modeling → Pictorial Representation (Diagrams)
- Application Generation → Coding → 4th Generation P.Ls (.Net, Java, Python)
- Testing and Turnover → automated testing
  → tools available
  → errors identified and some errors fixed

=> Cost is to be monitored

Disadvantages:

- Cost
- Sufficient and appropriate ~~not~~ resources

Increment Model

Youtube lecture

- Flexible to changes
- Module by module working
- For large projects
- Max customer interaction
- Cost managed
- Early release product demand

especially for web, mobile apps

→ When developing commercial softwares.
↳ idea of organization

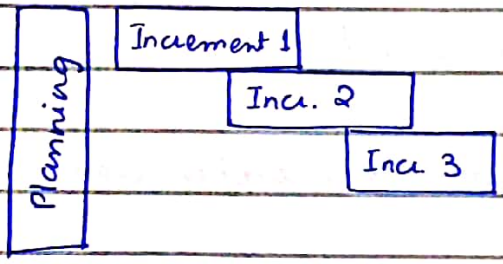
- Selling software and earning money

① Check demand of software (Market Analysis / Feasibility of idea)
or Marketting Research
→ Nature of customer is most important asped.
→ Then consider types, etc
→ Idea is matured
→ Financial Analysis

→ Not a one time release → increments are released
↳ small first e.g Starting version of Ms (DOS → Disc Operat. System) size in KBs

② First version should limited, constrained, smaller but outstanding.
E.g Ms word's first version was just a text editor.
improvements are made by increments.

③ Requirements are understandable
Proactively increments are planned



Overlapping

Top (Fortune Company)

o Resources, infrastructure, capable people are needed.
⇒ Business oriented model.

④ Previous ~~to~~ version's use stops on release of new ~~in~~ versions.
Windows 3.1 1985
995) Windows 95 → First version in which Internet, etc was introduced. } Big success
Windows 98 → Most secure and best time ~~rela~~ reliability
Previous versions → Stip down
Windows Me → Disaster → A lot of bugs
" 2000 → Huge Succes

Haram
Coca Cola

Windows XP → Great for graphics
↳ Most profit
" Vista → Medium

Youtube lecture:
- For large projects
- Radius of spiral = cost
- Angular dimension = Progress
- also called Meta model (Mixture of various models)

The Spiral Model →

→ Risk management.
→ Two things make risk
① Uncertainty
Probability (0.1 – 0.9) Range of Probability
evaluate percentage
in percent (- 99%)
② Loss associated with uncertainty
→ can be any kind of loss
→ other losses are converted to financial loss. (figures)

→ Intensity of Risk is determined by figures of above stated 2 points.
e.g uncertainty 50%
Loss → 1,00,000
0.5 x 1,00,000
Intensity of risk = 50,000

- To manage risk
  - Proactive approach needed (by well organized, well disciplined, systematic people)
  - You need to be well documented

"Document what is done and do what is documented".

- 4 Quadrants
  1. Determine Objectives
  2. Identify and resolve risks
  3. Development and test → SDLC
  4. Plan next iteration

=> Risk Management:

(i) Risk identification and specification

(ii) Risk Analysis → Probability, Loss identification

(iii) To identify / finalize risk strategy.

(iv) Risk monitoring and ~~con~~ controlling

Advantages: • Risk handling • large projects • Flexible • Customer Satisfaction.

Disadvantages: • Complex • Expensive • Too much risk analysis • Time.

- Book Models → result of starting iteration not software
→ Documentation is highly appreciated
  ↳ Max in start
→ Risk resolved first → so no working software formed first.

Conclusion:

- Situations are practically mixed so combination of ch. of process models used.
- Hybrid Nature of Process Models → use as needed

Project Management (PM)

- PMP → Project Management Profession → Certification
  Organization → PMI

① Project:

- Particular goal (Substantial goal)
- Non-repitative → are UNIQUE
  → Triple contraints will change.
  even if they apparently look same
- must be Time bound.
  ↳ evolved domain of PM.

→ Specialized skills, resources will be needed.

⇒ Non-Project is technically called Operation
→ repitative activities.
→ smaller goal
→ can be time bound.

Making → Project → use → operation
↳ Relationship of Project and operation.

=> Projects used after making transform to operation.

Skills needed: (Knowledge areas of Project Management)

Project ④ Cost Management
" ② Time Management
" ③ Scope Management
" ④ Communication Management.
⑤ Project Quality Management

⑥ Project Human Resource Managment
⑦ Project Risk Managment
⑧ Project ~~Pur~~ Procurement Management
→ Purchase Management actually
→ Least involvement in Software PM
Softwares are usually provided by company and some hardware may be needed
⑨ Project State Holders Management.
⑩ Project Integration Management.

⇒ Learning and application of above 10 areas is called Project Management.

Lifecycle of Project Management:

- What is connection b/w PM Lifecycle and SDLC life cycle?
- 5 Phases of PM Lifecycle:
  1. Initiation
  2. Planning
  3. Executing
  4. Monitoring and controlling
  5. Closing

① Initiation:

- Done by higher management (acquirement, etc)
- Common step of SDLC and PM
- Eg CMS self initiation

< RFP → Request For Proposal or RFTP → Tchn. Prop. Tender Notice >

- Most important phase
- Shortest phase
- Proposal submission and acceptance.
- Project Manager is decided after acceptance of proposal by higher management.
- Done by Higher Management.

② Planning

- Done by Project Manager.
- Will chiefly needed team member (2-3 max)
  - ① Task
  - ② Time
  - ③ Dependency
  - ④ Assignment
  - ⑤ Cost
- Max time served here.
- Failed to planning, planned to fail.
- Proactive approach.

③ Executing

(SDLC executed here)

→ Resources needed both material and human.

→ SDLC people will work under PM members.

⇒ Most bulky phase of PM

Mostly work done is physical all others are mostly conceptual

→ Proje ct starts to come to life.

④ Monitoring and controlling

=> All phases of PM are not done one after another instead overlapping is done.

→ Supervising phase

→ Problems will be there which will be avoided and controlled by this phase.

→ Controlling → action done if work is not being done properly.

→ It is the main skill checked in a Project Manager.

< Set of Tools → 7 Quality Control Tools (7QC) many needed in this phase >

→ Continue until Project Completion.

⑤ Closing:

- Irrespective of status → Proper closing is always done.
- Formal Closing
  → After delivery proper documently closed.
- Done in case of both success and failure.
- Done by Project Manager
- Reported to Higher Manag., customer, etc

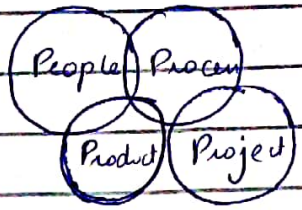
- Official doc. through which a project is authorized to be built
  ↳ Chartered document.

Project Management Spectrum

4 Ps

① People
→ PM and team members

→ Practitioners
→ End users
→ Senior Managers
→ Customers



② Procen
→ Procen Models
③ Project
→ Project Management discussion
④ Product

→ Product → target → achieved through people → by Procen models under PM.

Management	Leadership
→ Systematic, disciplined, quantifiable approach → First step towards leadership → Working with planning, properly, in a proper fashion. → Arrangment. "Right thing at right place in the right way." → Habit → Essential	→ Second step after management. → Management is foundation → Leader should have ① → Problem solving ② → Vision (Belief) ③ → Lead by example behaviour. • accepting failure • not blaming others • succes credit will be given to team • reward → team خلوص اور قربانی

(Leadership)

Purity and sacrifice

④ → Creativity and innovation

- gives confidence
- makes people follow.
- think out of the box

- "If your action inspires others to ① dream more, ② learn more, ③ do more and ④ become more, you are a leader."

① Clear vision

② Lead by example (Burns itself to make way for others)

③ Doesn't discriminate

→ Sacrific → Purity

④ Well managed, systematic.

< Tony (Anthony) Robbins
Les Brown
Freeman
Sonu Sharma
Sandeep (Samajeet) Singh
Qasim Ali Shah
Qaisar Abbas
Umair Jaliawala >

Motivation:

- To persuade someone
- one who has it can give it
- breaking of ice

Motivated person:

① Positive thinking

② Have sources

→ good books

→ good company

→ good teachers.

③ Good character

"عبادت سے پارسائی ملتی ہے نیکی سے رب ملتا ہے"

Team vs Group

- Team → single goal → not permanent.
- Group → common interest.

- Under the leader, opinion of team member doesn't matter.
- Tuckman's Team Model.
  1. Forming
  2. Storming
  3. Norming
  4. Performing
  5. Adjourning

Team Structures:

Decided upon
- Communication
- Decision making
- Leadership

① Communication
- Vertical or horizontal
- High to low → Vertical
- Same Level → horizontal
- Both types done in team extent is dif which changes model.

Work	Type of communic.
• Simple → order and obey	Vertical
• Complex	Horizontal
• Average	Vertical ≅ Horizontal

② Decision Making

Ways

① → Order and obey Leader. (Dictatorship)

- Group decision making

② → Democracy → voting
↳ decision by majority

③ → Consensus → dif of opinion arises on leader's idea. Discussion conducted and dif of opinion is removed. Skill needed.
↳ Time taking
↳ Personality and communication skills of leader matter.

④ → Plurality
↳ noone holds majority e.g 40 → 15, 17, 8
↳ decision on opinion of biggest group

③ Leadership

- Leader can be permanent or can be changed
- Simple → Permanent leader
- Complex → Temporary leaders

⇒ Project Manager will be one and permanent. He will change team leaders as per need.

① Control Centralized Team Structure
② Democratic Decentralized
③ Controlled Decentralized

① Control Centralized (CC) (Chief Programmer)
Communication : V > H
Decision Making : Order and Obey
Leadership : Permanent.

② Democratic Decentralized (DD) (Innovative Anarchy)
Comm. : H > V
Decision M. : With consensus
Leadership : Rotation based

③ Controlled Decentralized (CD) (Agile Team)
Comm. : V ≥ H
Decision M. : As per situation
Leadership : Mix.

(Structual) **Analysis Models**

Data Model:

Entity Relationship Diagram (ERD)
→ to understand data domain of current system.
→ In database → to design it
Here we are studying to understand data domain of system.

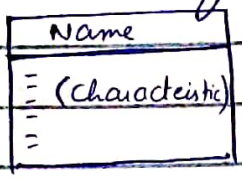
- 4 elements
  ① Entity
  ② Relationship
  ③ Cardinality
  ④ Modality

- Classes → OOP
- Function → PF
- ① Size & ② Complexity increases to deal → OOP
  ③ Control
- Simultaneous control of data and function provided in OOP by ~~to solve this~~ ~~was use~~

1. Entity:

Anything in the universe which can be recognized by certain characteristics is known as an entity.

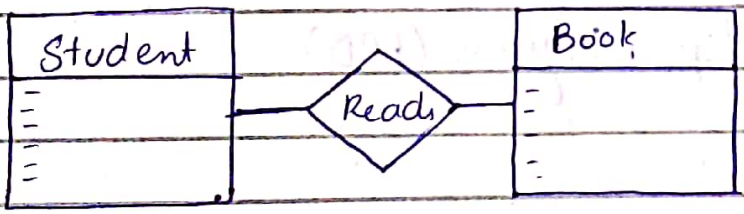
- In system we have to recognize only related entities.
- Denoted by:



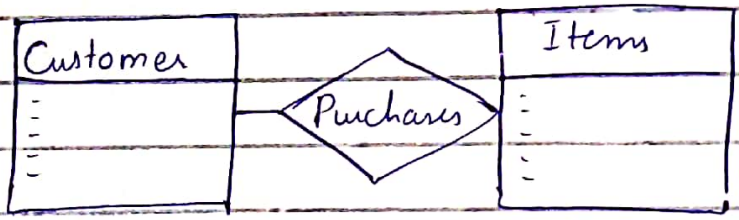
- Class has object similar way entity has instance.
- Physical representation of the entity → instance.
- Class or entity represents all of them, while object / instance represents just one.

2. Relationship:

- Reason of connectedness / relatedness b/w two entities
- Indicated by ◇ diamond.
  ↳ relationship is written in verbal phrase.



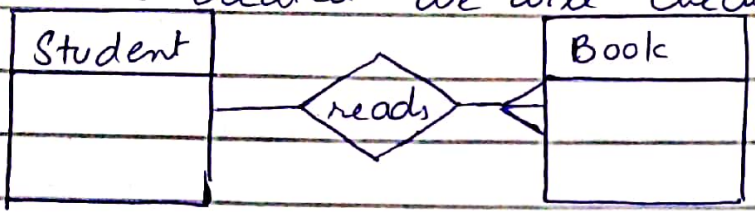
- An entity can relate with more than one entities



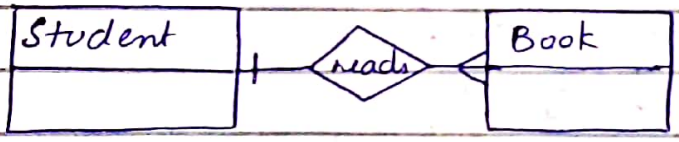
3. Cardinality

- one to one
- one to many
- many to many (exceptional)

E.g to measure cardinality of ~~student~~ book we ~~say~~ will check student suppose one instance of student is created we will check instances of book



- Cardinality of Book against student is many.
- many ⟵ ⟶ crow foot symbol.
- one +



one to many.

4. Modality

- If it is mandatory to create an instance of book against a student use +(near book)
- If optional use —O—
- Modality is of two types
  1. Mandatory
  2. Optional

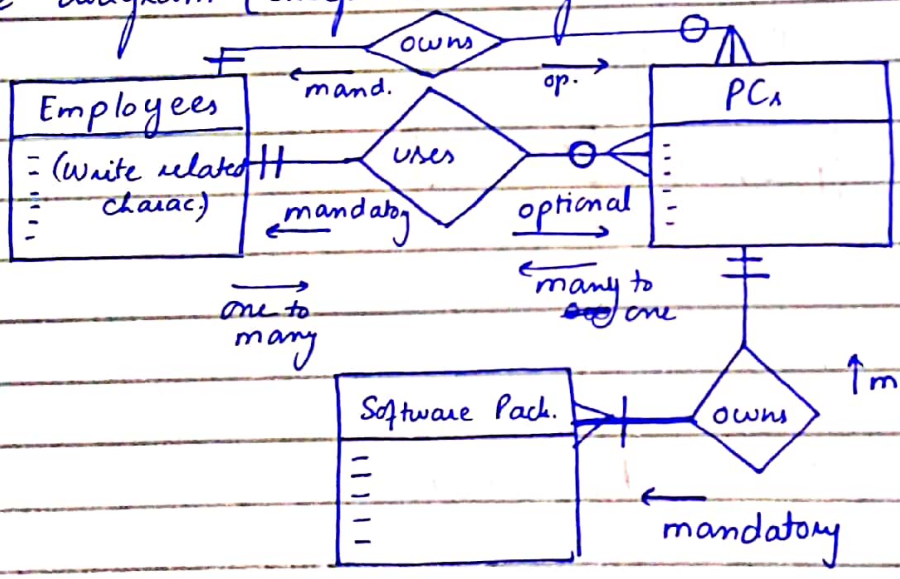
Step 1 : Read
Step 2: List entities (writing nouns) // May be entities or their characteristics
Step 3: Keep only those related

Example:
- PC Associates
- PCs
- Software Packages
- Employes

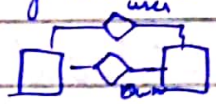
→ Four entities

Ch. of PCs
Ch. of Employe
Attributes of PCs
Ch. of Software Packages

⇒ Entity which represents whole system is not maintained in the diagram (exception in a few cases)



→ owns and uses so we use assumptions. if bothused



- Employee, Software Package → indirect relation

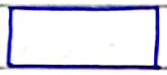
- Entity and relationship → conceptual level.
- Modality and cardinality → level of instance.

- Assumption taken and mark cardinality accordingly. Close an open situation using assumption.
- If multiple cardinalities exist "many cardinality is used" b/w two entities
- For Modality → again situation is open in this case so we take assumption.
- Human Being entities direct interaction is not shown. but ~~Example.~~ interaction through system exists. (System built to decrease human interaction)
- Indirect interaction → automation.

Data Flow Diagram (DFD)

- Part of Functional model
- Trying to understand function domain of system. (operations)

Notations, elements:

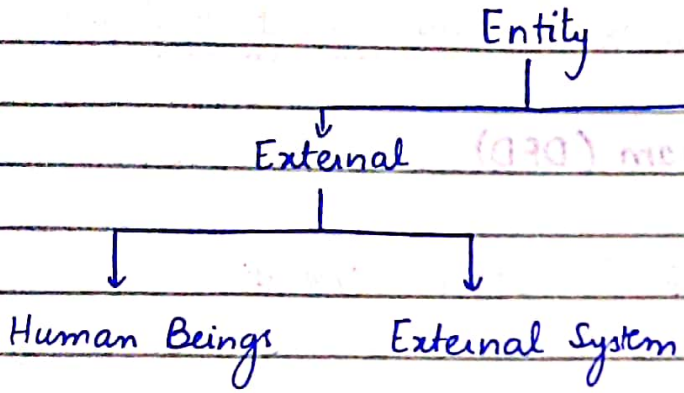
1. External entity 
2. Process 
3. Data Flow 
4. Data Store 

1. External Entity

- Two types of entities

• External	• Internal
→ reside outside boundary of system	→ Entities with in the system
→ Interacts with the system for producing or consuming data	
→ can or cannot be part	

of system.
→ Human entities are purely external.
→ Some entities are external not part of system just give and take information.



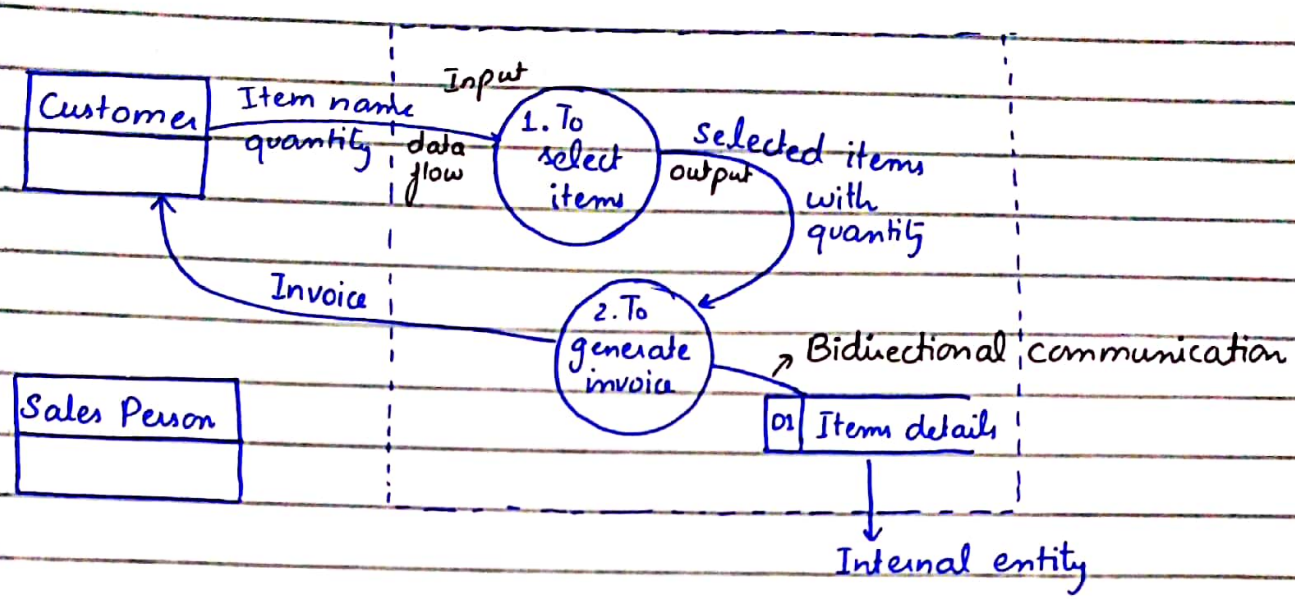
- External entities are intiators to start the functionality of the system.

2. Process

- Written in verbal phase.
- 3 Parts
  1. Input
  2. Process itself
  3. Output

- Name in form of data provided (Noun) is written on top of data flow. (Don't write process name).

- Data can be one or many



- Data Store interaction is always bidirectional.
- ~~Interact~~ External entities can't directly interact with internal entities

Example:

Case Study: Physician Billing System.

level 1: Context level or level Zero

- Whole system is represented as process.
- Called Black Box system.
- Write external entities.

- Interaction of external entities with system is shown.
- 4 inputs and 2 outputs with respect to system (Physician Billing System)
- These inputs and output should go to next level as it is this is called Balancing

- Level 1
  - Internal entities of ERD are shown as data store. in DFD.
  - Every system has a core process in financial stuff its usually invoice generation.
  - System is white box here processes and data stores are shown. This is called Level of decomposition.

- 1 input 2 output 1 data store → valid fee (e.g)

- Level # 2
  - Only those processes are taken to level 2 which need decomposition.

- Duplication of processes → not allowed similarly data store.
  Duplication of external entities → allowed in case of fewer external entities and they interact with all processes.
  - Top down approach → Decomposition approach.
  - System rules don't allow external entities to interact with internal entities.

Documentation

Documentation of DFD is of two types:

① Process Specification

② Mini Specification.

- Leaves of diagram (leaf processes) ⟶ Mini Specification
  ↳ no further levels
- The ones who have further levels ⟶ Process Specification.

Process Specification.

Physician Billing System: (Context level)

- System name
- External Entities
- Inputs and outputs of external entities.
  (If no input or output write none)

Level 1:

- Write process name with
  - input - output - data store.

Mini Specification:

- No headings
- Logics in st sentences