Name :
Date 19/01/2023

* Documentations consists of diff. of manuals ac-

Operating procedures :-
- user manuals
  - sys. overview
  - refer. guide
  - Beginers guide
  - Tutorial.
- operational manuals

* Software product - i) Generic
  ii) customize. (Bespoke)
source, code, obj. code, data; prototypes, report, documents, plans, test results test suits.

S/w engg. - Def^n. is an engineering discipline which is connected with all aspects of S/w production.

SE- should ~~have~~ -
- adopt a systematic & organised approch to their work.
- use appropriate tools and techniques depending on
  - problem to be solved
  - the development constraints &
- use resources available.

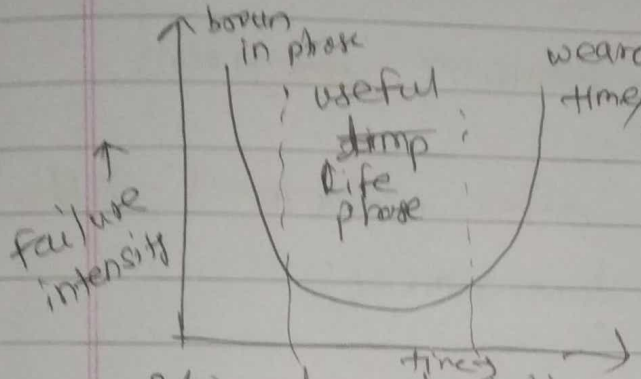
First conference on SE, 1968 - Friz
The establishment and use of sound engg principles in order to obtain economical development S/w that is reliable & works efficiently on real machines.

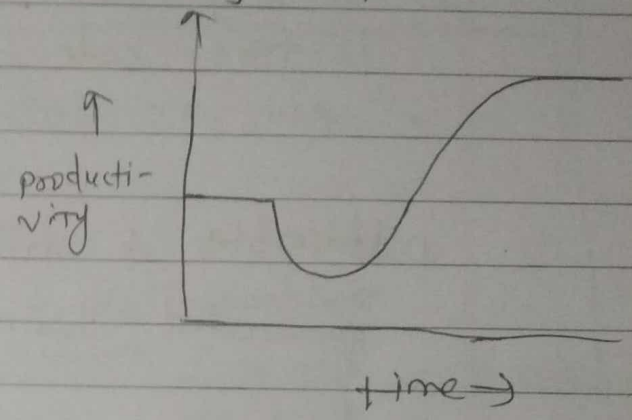
stephen Schach -

S/w Process - in which ae way to ^ produce s/w

Why it difficult improve s/w process ?

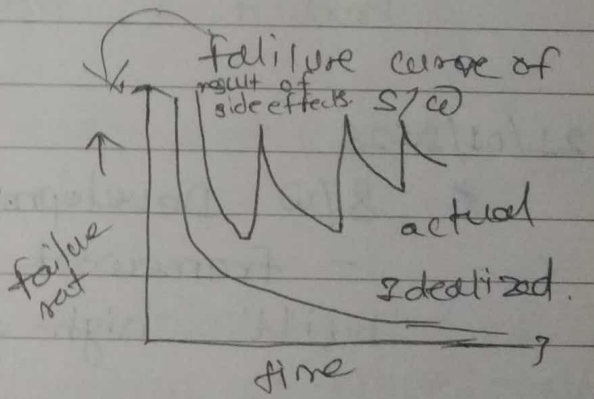
- Not enough time
- Lack of knowledge.
- insufficient commitment
- wrong motivation.





S/w characteristics -

- s/w does not wear.
- s/w is not manufactu.
- realiabity of components.
- s/w is feasible.



Changing nature of s/w.

Sy s/w myths - (management perspectives)

- Management can be confident about good standards.
- Company has latest computers & state-of-the-art s/w tools, so we wouldn't worry about quality of product.
- s/w easy to change.
- Addition of new s/w specialt

computer provide greater reliability the device they use.

* S/w myths - (customer perspective)
  - s/w with more feature is better s/w
  - s/w can work right for first time.

* S/w myths (Developer perspective)
  - Once s/w is demonstrated then job is done.
  - S/w quality can not be assessed before testing.
  - The only deliverable for s/w deplopment project is test code.
  - Aim is to develop working program.

Deliverables & milestone - after requirment engg.
↑ generated during s/w development.

Product & process

21/01/2023

* S/w Development Process & process models.
  - framework for task that are required to built high quality s/w.

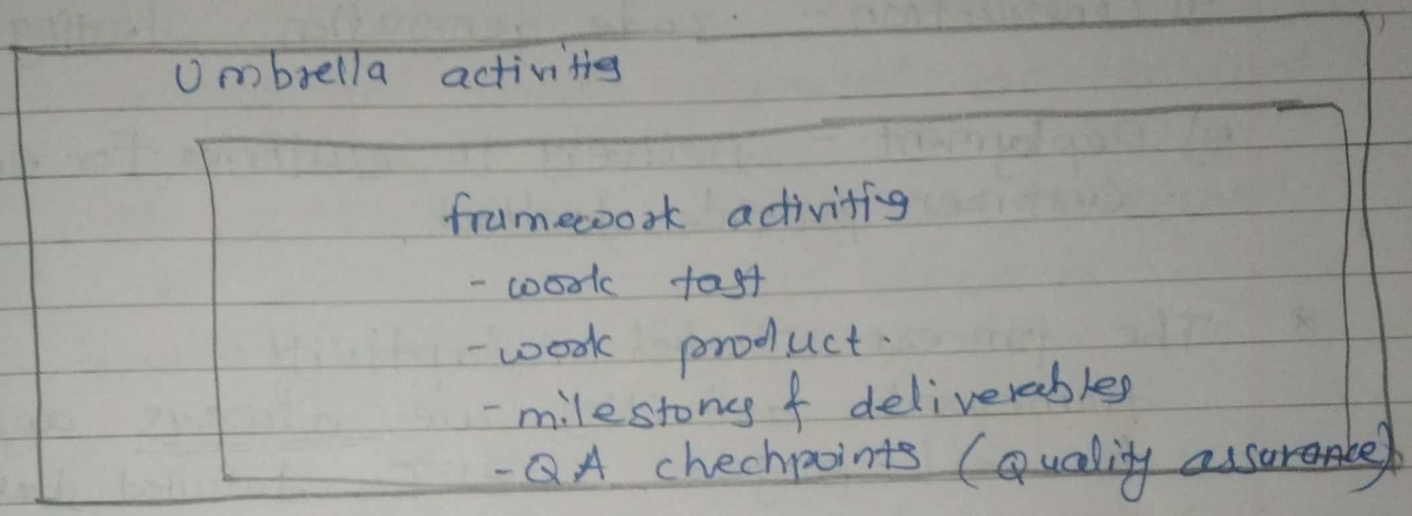
Who? - who are involved in this process
stake holder - who is direclty / indirectly benefit by product.
manger / group leaders, s/w engineers, & customer

Why? - provide stability control & organization to an chaotic activity.
Steps? - handful of activities are common for all s/w, ti small variation.
work product - programs, docu, etc.

5 levels.

* Process framework:-
Umbrella activities are all same for all, but certain activities may change for particular project.

Process framwork.



Why s/w process -
a process defines who is doing what, when & how to reach a certain goal.

each framework acti. is populated by set for s/w engi. action - a collection of related tasks
each action has individual work task. (same as milestone)
related to process doing during SE., necessary
work product - delievrable is that we are giving after SE.

* Generic process framwork activities
(CPMCD)
i) communication - heavy communication with customer, stakholders, team, encompasses requirements gathering & related activities.

2) Planning - workflow that is to follow.
- Describle technical task, risk, resources, work product

3) Modeling - help to develope & customer to understand requirements & Design s/w. (analysis)

4) Construction - code generation, testing.-

5) Deployment - Delivery to customer for evaluation
- Customer provide feedback.

* The process Model : Adaptivity-
framework activities are always applicable for all project .. BUT detailed task may differ

* Umbrella activity - 1) s/w project tracking & control assessing project against proj. plan
2) formal tech. review - assessing & remove error before next step/task.
3) s/w quality assurance - define & conduct act. to ensure s/w qual.
4) s/w configuration managment - manages changes. (git)
5) Document preperation - help to create models, documents, logs forms & list.
6) Reusability managment - criteriak to work product reuse
7) Measurement (person-months - efforts are calculated)
8) Risk mangment

* CMMI (Capability Maturity Model Integration)
SE institute (SEI) has developed proces meta model

CMMI - developed by SEI
- defines each process area in terms of specific goals & specific practices.

Specific practices refine a goal into a set of process-related activities.
Specific goals - establish characteristics that must exist if activities implied by a process area are to be effective.

1) Level 1 (Initial)
- process are poorly managed or controlled
- unpredictable outcomes
- No KPAs (Key Process Area) defined
- ad hoc and chaotic approch
- lowest quality and highes risk.

2) Level 2 (managed) -
- process are planed and controlled
- requirements are manged.
- project are manged & implenanted acco.. to their documented plans.
- The risk involved is lower & quality is better than initial level.

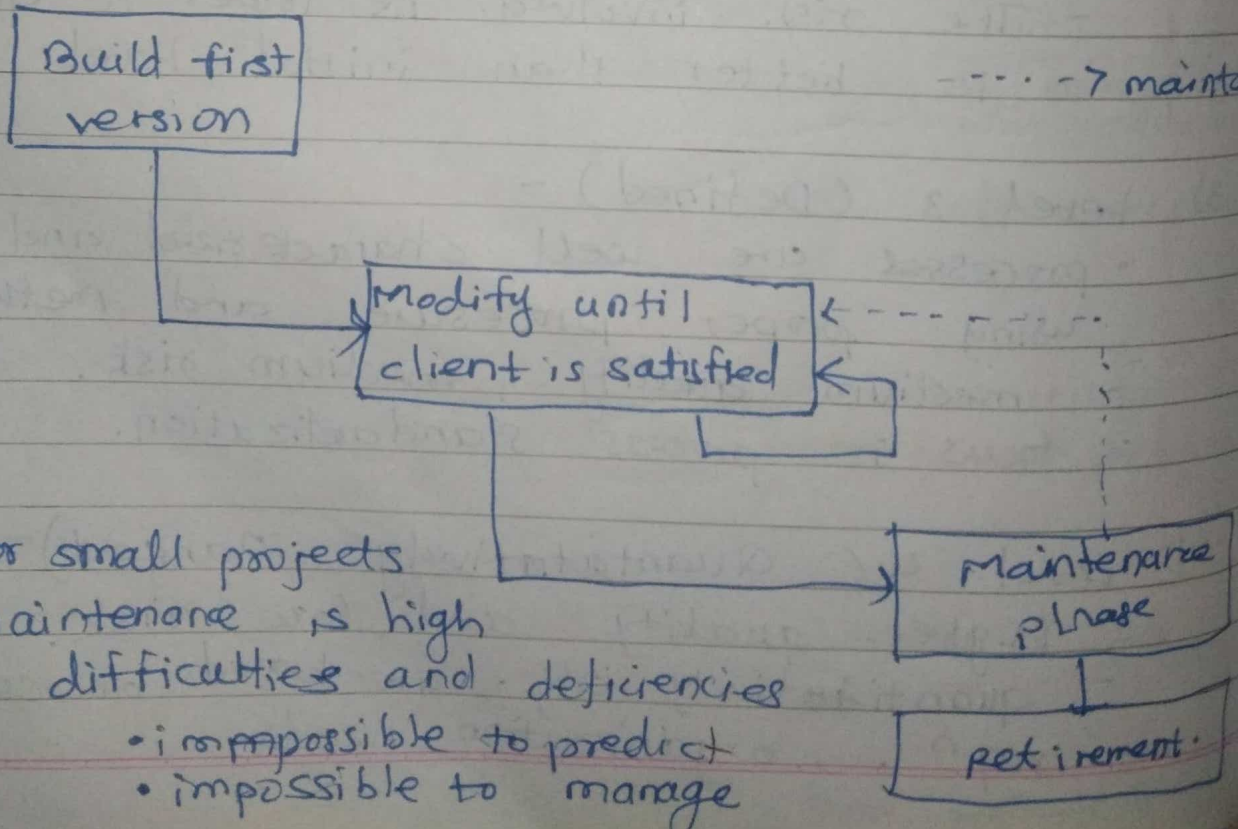
3) Level 3 (Defined) -
- processes are well characterized and described using proper procesure and method, tools.
- medium quality, medium risk.
- focus is process standardization.

4) Level 4 (Quantitatively manged) - analysis involved
- higher quality and low risk.
- quantitative obj. are based on customer req^., organization needs.

5) Level 5 (optimized)
- continuous improvement in process and their performance.
- high quality of process.

* Software Process Models - Roadmap for SE work
Why models are needed ?
processes has → black-box - problems
→ white-box - advantages.
- Build and fix.
- Prescriptive model
  - waterfall or linear sequential
- Incremental process model.
  - RAD or incremental
- Evolutionary
  - prototyping and, spiral

- Build and fix -



* Prescriptive model -
orderly approach

Waterfal Model or classic ~~life~~ life cycle or linear sequential.

Communication → project initiation & req^n. gathering

planning → estimating, scheduling tracting

modeling → anyalysis design.

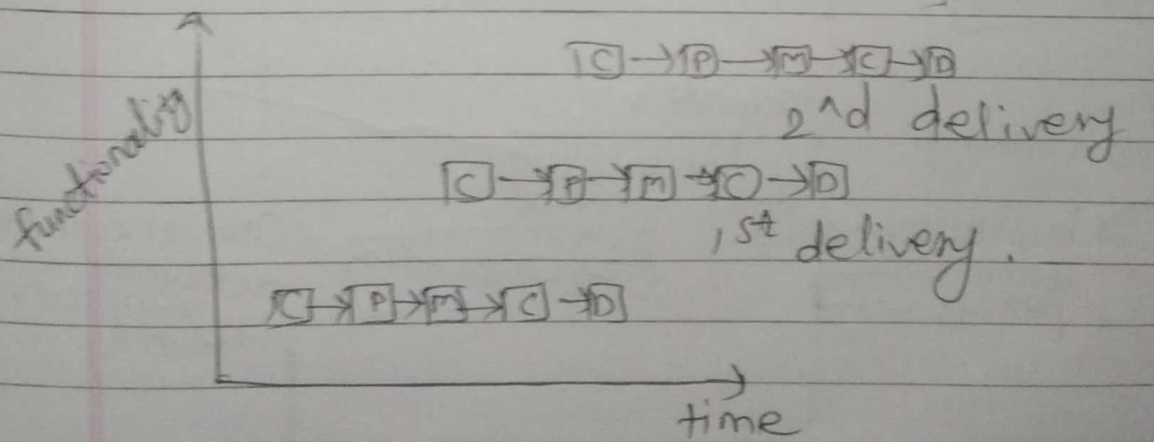
construction → code, test.

deployment → delivery, support feedback.

Requirements are 100% fixed, not changing.

02/02/2023

Incremental Process Model -



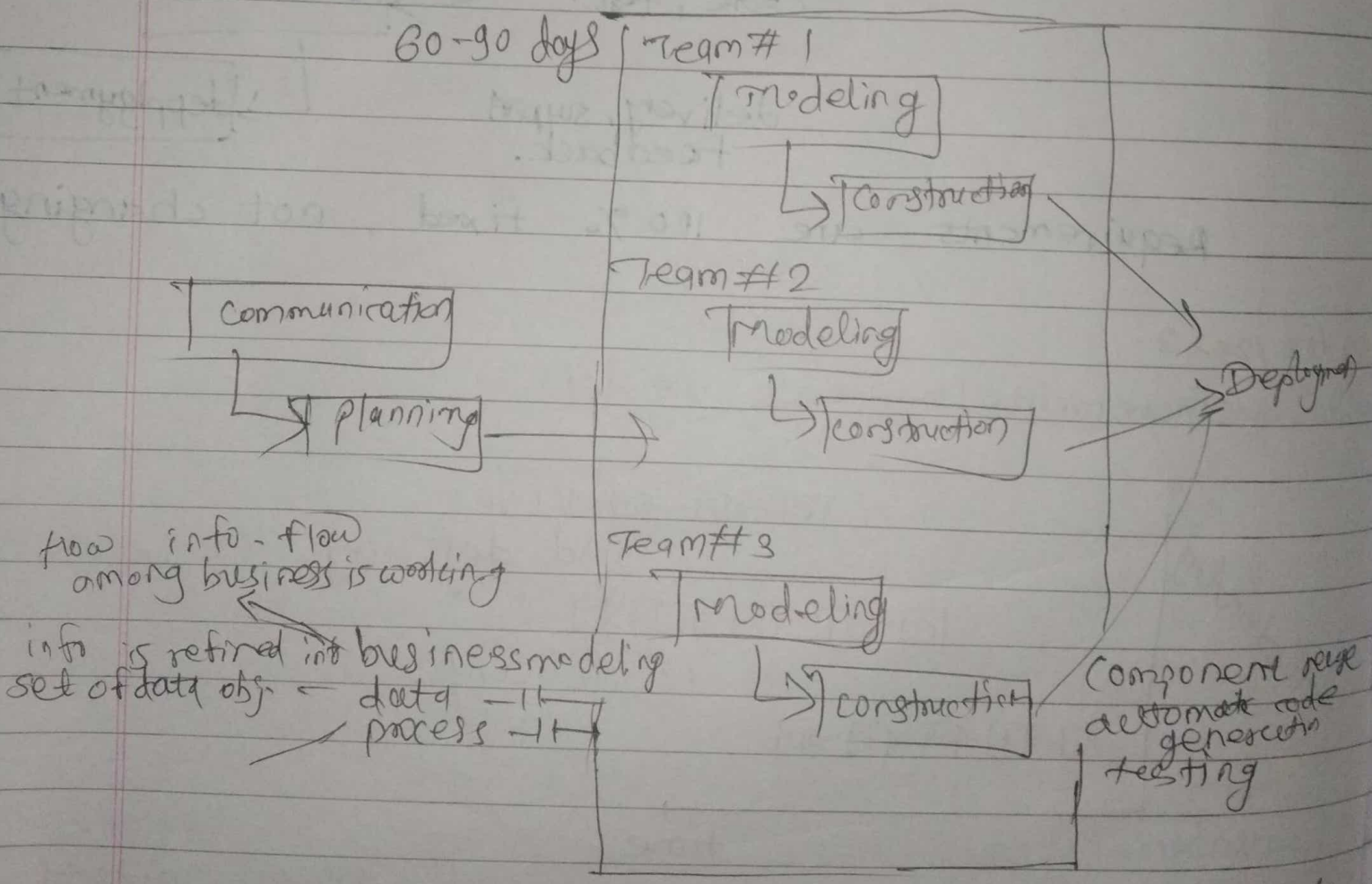
Delivers s/w in small but working pieces & each build on below 3rd on 2nd.

- Even if requirements are changing, th can fulfill in 2nd or 3rd delivery
- It particularly useful when enough staffing is not available for whole project.
- Lower risk.

elicit (to obtain)

- early req^. act as prototype to help elicit req^. for later increment.
- core functionalites are many times tested.

* RAD (Rapid Application Development



makes heavy use of reusable s/w compents with an extremly short development cycle.

Requirement Engineering

16/02/2023

* requirement – a function, constraint or other property that sys. must provide to fulfill need of intended user(s)
engg. – implies that systematic & repeatable techniques should be used.
req. engg. – requ for a product are defined, managed & tested

RE starts with communication activity & continues to modeling activity.

* Characteristics of good requirements –
1) clear & unambiguous
   - standard struct
   - has only one possible interpretation
   - one line, one requ.
2) correct – requ. contributes to real need.
3) Understandable – reader can easily under meaning of it
4) verifiable – can be tested
5) complete
6) consistent
7) Tracable

* RE tasks – 7 tasks
1) Inception – establish basic understanding of proble & nature of sol^n.
2) Elicitation – Draw out req^n. from stakeholders
3) Elaboration (highly structured) – create analysis model that represent behavioural aspects. information, function and

4) Negotiation - agree on a deliverable syst that is realistic for developer & customers. walk through

5) Specification - describe reqn. formally.

6) validation - review the reqn. specifications for errors, ambiguities, commissions & conflicts.

7) Requirement mangement - manage changing Traceability tables - (feature ~~to~~ table, source dependency, subsys.) reqn.

Why re elicitation is difficult?

- Problem of scope
  - bounday is not defined
  - specify unnecessory technical details
- Problem of understanding
  - customer not sure of what they need.
- problems of ~~viola~~ volatility -
  reqn. may change over time.

ucd - used case diagram.

* Requirement Mangement -
Tracebility table.
(not any standard templet, varying from project to project)
Req^. analysis -
Specific s/w's operational characteristics.

Analysis Modeling principles -
1. info domain of a problem must be repr & un
2. function that are s/w performs must be defin
3. behaviour of s/w must be represented
4.

→ i. data flow into sys. out of sys & data stored
2. function - direct benefit to end users & also provide internal support for thoes features that are user visible.
3. Behaviour - interaction within sys as well as environment.

Partitioning -
Analysis rules of Thumb:-
level of abstraction should be relatively high.
Don't get bogged into details.

Two approches -
1. Structured analysis -
data obj. are modeled in a way that defines their attributes & relationship
process. data flow diagram.

2. Object oriented analysis -
Focus on definition classes.
UML.

Next task ERD & SRS.

* Elements of analysis model - 4 ele.
A] Scenario - based - uses cases - text, case - diag, activity diagrams, swim lane diagrams
B] Flow oriented - DFD, control flow, processing narrative
C] Class based - class diagrams, analysis package, CRC models, collaboration diag.
D] Behavioral diagram - state diagram, sequence diagrams

(-- items are part of UML.)

Every functional req^n. have work task (reason ~~every~~ atomicity)

6/03/2023

* Data flow diagrams - structural analysis.
Graphical tool useful ~~to~~ for communication with user mangers & other personnel.
Focus on movement of data bet^n external entities & process & bet^n process & data stores.
person or organization that send data into system

Why DFD?
- DFD Elements
1) External entity (sink/source) receive data from sys.
2) Process - models what happens to data
3) Data stores - represent permanent data that is used by system.
4) Data flows - actual flow of data bet^n elements

do not process data

Name :
Date : / /

Gane

Symbol.	Gane & Sarson symbol.	De Macro & Yordan symbol
Noun — External entity	Name	Name
verb phrase — process	Name (rounded rectangle.)	Name (oval)
Noun — Data store.	D1 | Name	D1 Name
Name of data — Data flow	Name →	Name →

It is okay to any one of the method but stick to only one throughout proj.

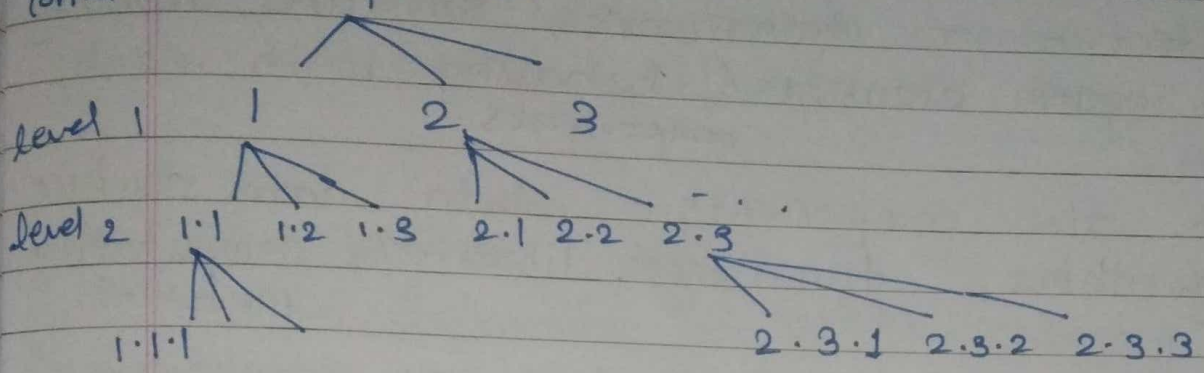
* Symbol Naming —

* Decomposition of DFD — top-down approch.

Level 0	context diagram (words diag)	expla- contains only one process
Level 1	overview diagram	utilizes all four ele.
Level 2	Detailed diagram	a breakdown of level1 proc

There is no rule as to how many levels of DFD can be used.

* Rules for level 0 diagram — (only imp things)
- 1 process represent entire system
- Data arrows show input & op
- Data stores Not show, they are within system



Rules of Data flow

Data can flow

- 1) process to process
- 2) external entity to proces or 3) vice versa
- 4) process to store & back.

can not flow

- 1) store to store
- 2) store to external entity
- 3) external entity to external entity
- 4) —11— to store.

Two types of DFD - physical & logical.

Unit 3

System Architecture & Design overview

09/03/2023

Architecture — framework / structure depicting how system elements/components behaving with each other.

- s/w components - data, data structure, algorithms, functions, libraries, computational modules.

* s/w architecture - structure or structures of system which comprise the s/w compo, the externally visible properties of those comp. & relationship among them.
It is not operational s/w but it is representation that enables s/w engineer to
  - to analays effecti effectiveness
  - alternative architecture

* Importance -
* Data design - imp part of s/w design
- 1] Component level -
  design of data structure & associate algorithms.
- Principles are applicable to data design.
- The systematic analysis principles applied to function and behavior should be applied to data
- all data structures and the operations to be performed on each should be identified.
- data dictionary.

semantic model - ERD

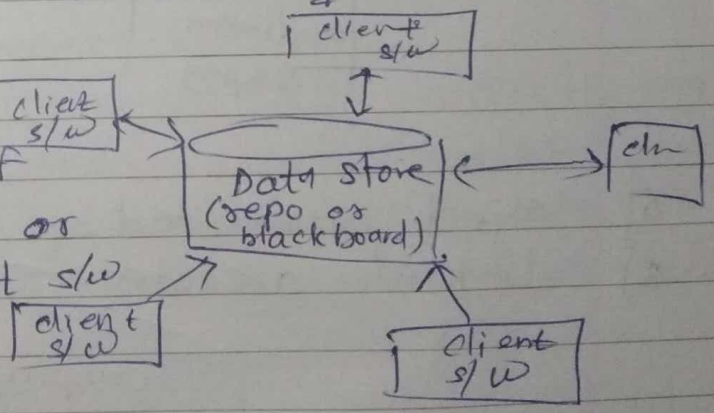
- low level data design decesion should be deffered until late in design process.

* Architectural style -
Style describles a system category that encompasses
1) set of components -
2) set of connectors. that enable "communication, co-ordination & cooperation" among components
3) Contraints how comp. integrated to form the system.
4) Semantic models that enable a designer to understand overall propesties of system.

Types - 5

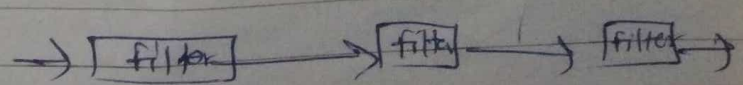
1] Data - centered architectured -
(client - passive & reposit ory - active it becomes black-boxed architeation)
repo - passive & client - active.

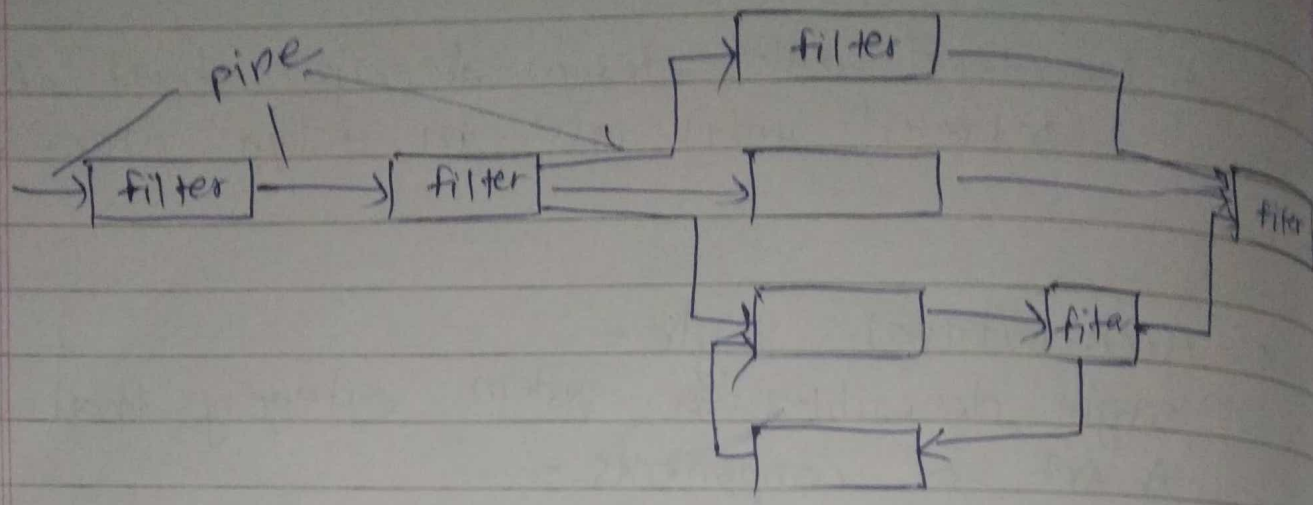
client s/w accesses the data independent of any changes to data or action of other client s/w



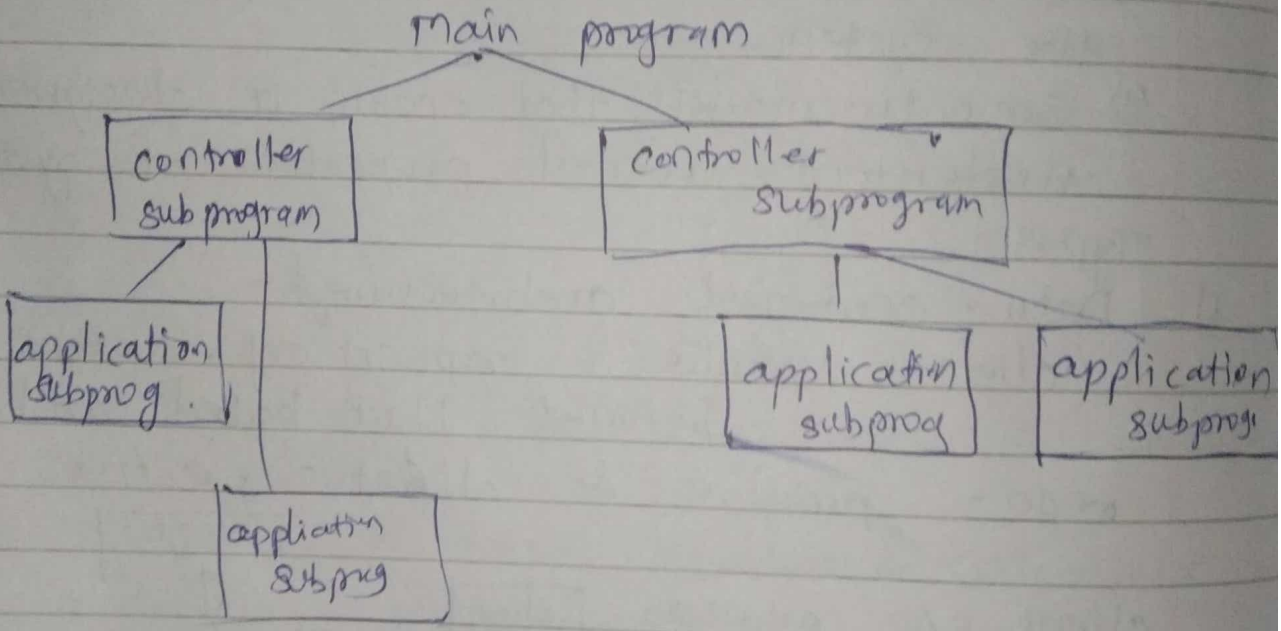
2] Data - flow architecture -
pipes & filters.

Batch sequential



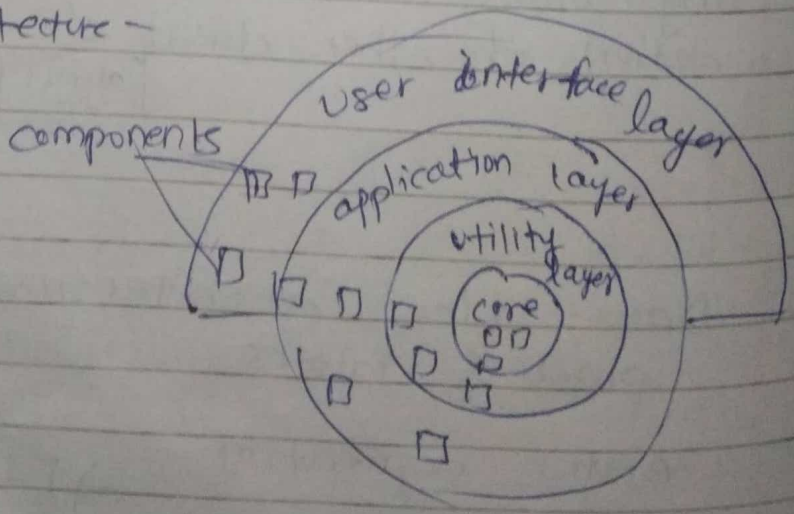


3] call & return -



4] object oriented.

5] Layered architecture -
OS.



13 of 2023

* Design Engg. -
- is to produce a model or representation that depict: 1) firmness - not have bug that inhibit its functions
2) Commodity - suitable to its intended us.
3) delight - pleasurable to use.
- provides details data structures, architecture, interfaces & components are are necessary for implementation.
- foundation for all s/w engg. activities.
without ~~with~~ - risk, test, quality.
which can't possible to detect.
- s/w design iterative process through which req^n. are translated into "blueprint" for constructing s/w
- As design iteration occur, subsequent refinement leads to design representation at much lower layers of abtraction.

Good of design process :-
- design must implement all explicit req^n.
- must be readable, understandable guide for those who generate code & who test
- should provide comple picture of s/w

Quality guidelines :-
more cohesion, less coupling

Design principles - design process & design model. (why both)
- design process should not suffer tunnel vision
consider alternatives.

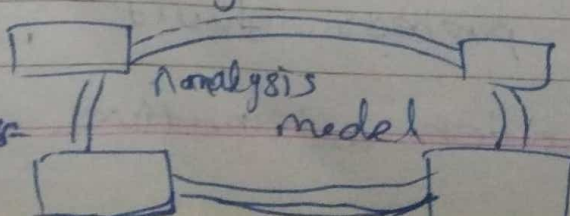
- The design should be traceable to analysis model
- design should not reinvent the wheel - use already exit pattern because time short & less / limited resource
- should minimize the intellectual distance bet^n. the s/w & problem as it exits in real world, it should self-explanatory
- should exhibit uniformity and integration
- should be structured to accomodate change.
- design is not coding, coding is not design
- should be structured to degrade gently even when unusual data, events or op^r. conditions are encountered.
- should be assessed for quality as it is being created ~~or~~ not after the fact
- should be reviewed to minimize conceptual (semantic) errors

* Design concepts - provide framework ^ to get the things on right way abstraction, refinement, modularity, archi. data structure, control hierarchy, structural partitioning, s/w procedure, info. hiding. - (find out def^ & clear concepts)

* s/w design - 4 designs data/class, architec interface design, component de

Elements of analysis mode - 4 ~~part~~ ele.

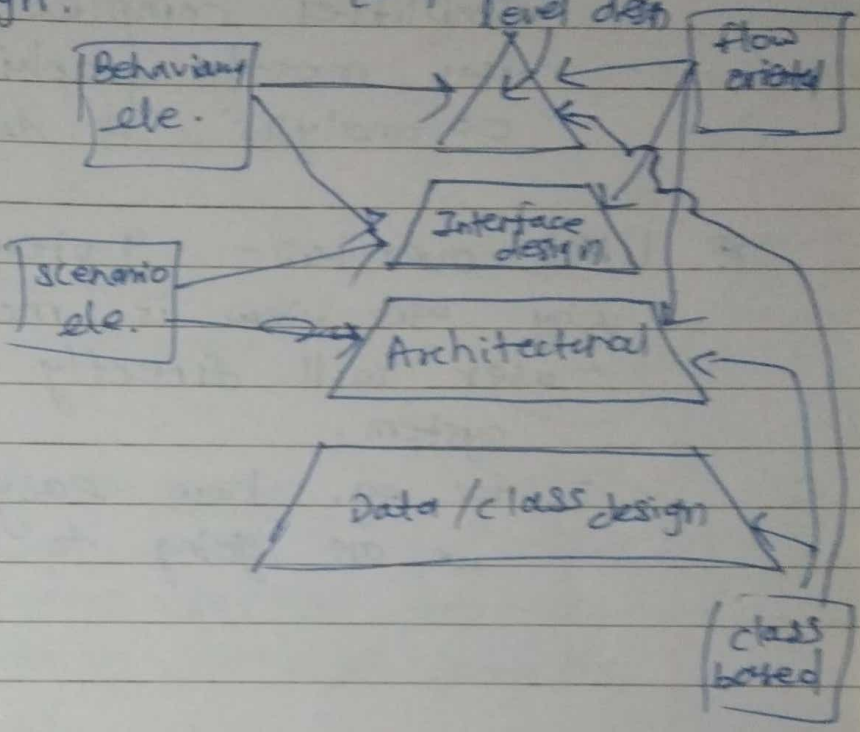
scenario ele, flow oriented, behavioral ele, class based.



see this in req^. engg. notes.

Translating Analysis & Design -

Data / class design - class based elements
architectural - data flow / flow oriented
interface design.



architectural,

16/03/2023 -

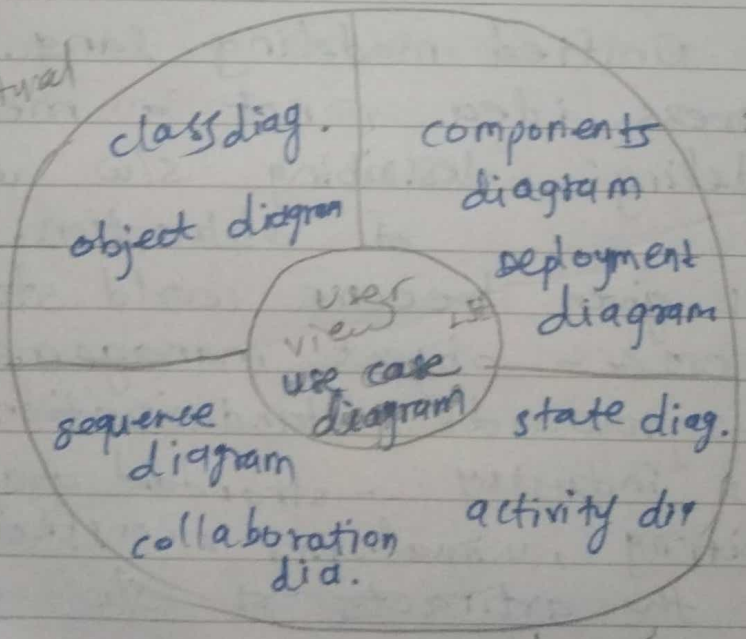
UML - Unified modeling lang.
- express idea, not a methodology
- modeling: describing s/w with high level of abtraction.
- UML given become world standard, given by OMG - object management group - works towards standarization of objects. It is industry - standard graphical lang for specifing, visualizing, constructing & documenting the artifacts of s/w sys.
  100 % object oriented.
- Use graphical notation: more clearly than natural lang (imprecise) & code (too detailed)
- helps acquire overall view of sys.

- UML not dependent on any one lang. or technology
- UML moves us from fragmentation to standardization.
- Simplifies complex process of s/w design.
- uses mostly graphical notations to express OO analysis & design of s/w project.

* UML overview - 4 views user, implementation structural, behavioral view.

Why user view is more imp?
- user will directly interacting with end prod/ system.
- for eg. how easy to use that product we are going to buy that.

1st use case diagram 2nd structural view
3rd behavioral diagram. 4th implementation diagram.



- use case
- class
- sequence
- collaboration
- state diagr.

behavioral view
3rd

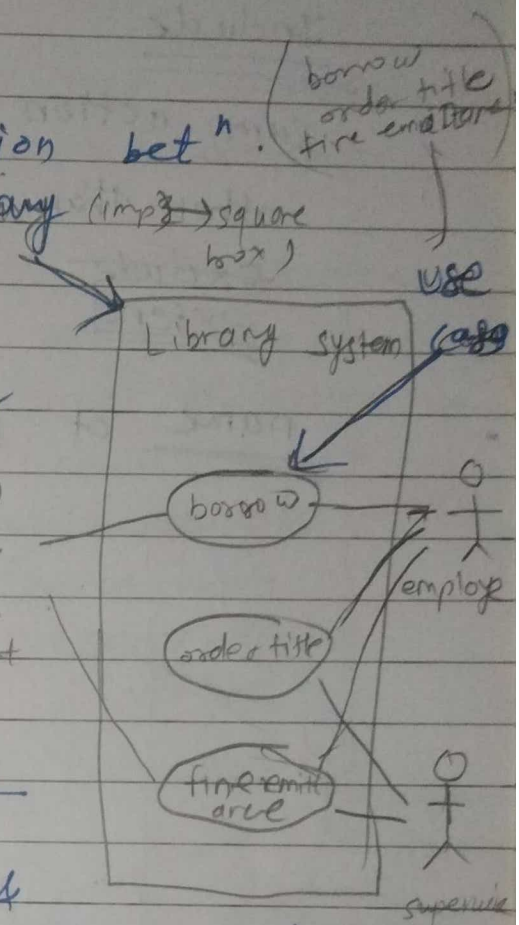
either have class or object diagram.

since object is instantaneous of class.

— most widely used.

preliminary -

1) Use-case diagrams -
- is set of use cases
- model with interaction bet^n.



stickman
stickmen

i) external user & sys. Actor
ii) system itself
more precisely, an actor is user playing specific role

- describing a set of user scenarios
- capturing user requirments.
- contract bet^n end user & s/w developers (for small projects)
- primilary stage

left i/p proces, right access process.
lines - association, no any arrows (since only interaction is shown) thick solid line.
Nouns - actors verbs - use cases

use cases - set of all possible scenarios that describe interaction bet^n user & sys.
Actor - may be human or inhuman.
Association - communication bet^n an actor & use cases — solid line.
Generalization —▷ solid line hollow △.
type of association.

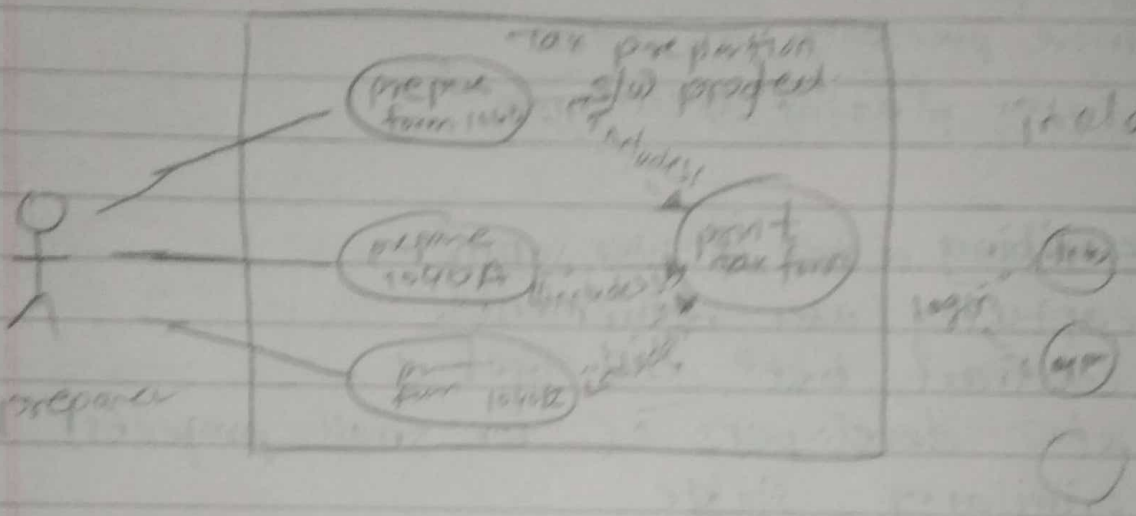
use.
starting from base case ending at include use case

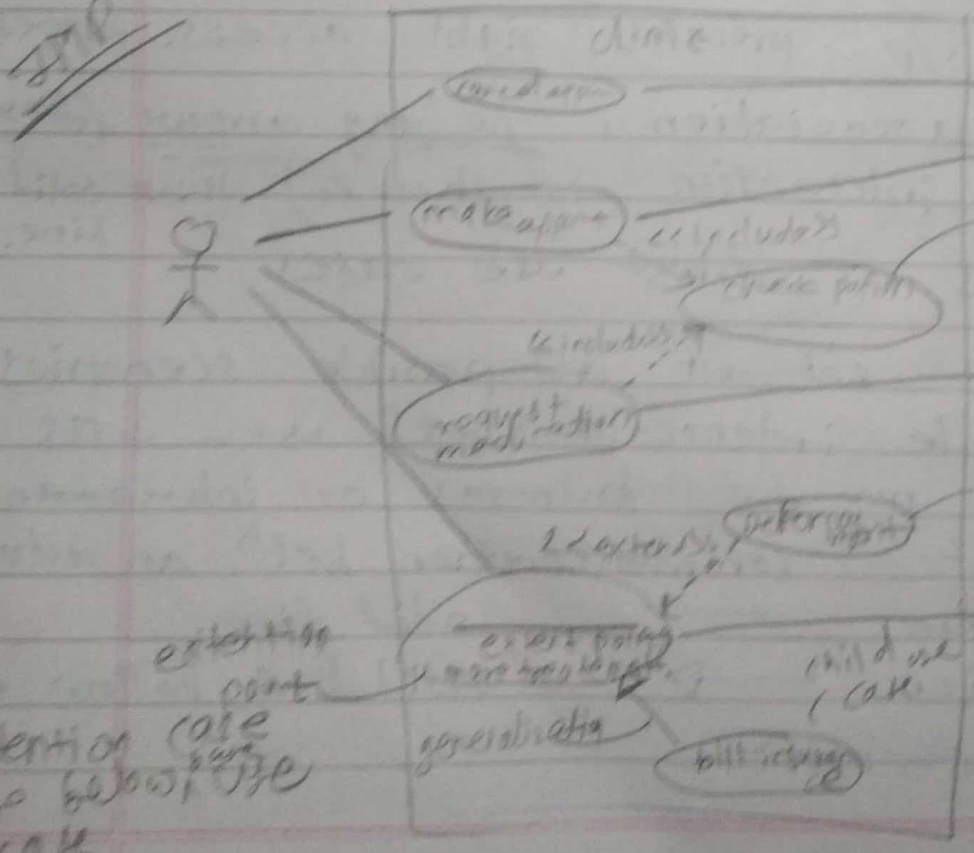
Include <<includes>> dotted line with arrow head

Same action repeated - include

other than parent / base behaviour - extend <<extended>>

name of project within boundary





Ufr →

23/03/2023

* Class diagram - (structural)
  - depicts classes and their interrelationship
  - used for describing structure & behaviour in use case
  - required capturing, end-user interaction.
  - detailed class diagrams are used by developers
  - provide a conceptual model of the system in terms of entities & their relationships.
  - each class represented by rectangle subdivided into 3 compartments.
  - name, attributes (hidden), operations visible

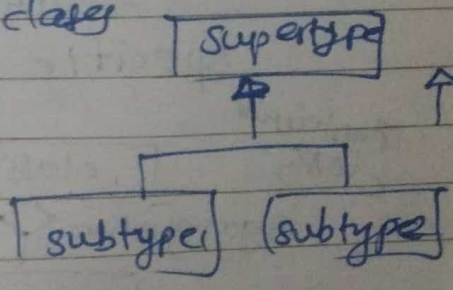
+ public visibility
# protected visible
- private visibility.

(encapsulation)

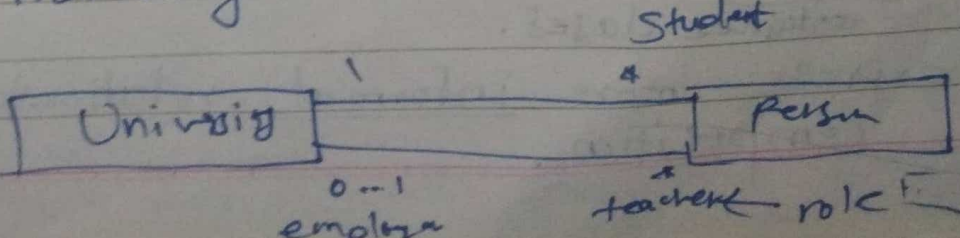
OO relationships
Generalization - (parent-child relationship) (tip of arrow always towards parent/base) (among related classes) - inheritance
(used for abstracting details upto several levels)

Association (student enrolls for course)
- aggregation
- composition

relationship bet^n instances of classes (how objects are related)



Have two ends -
- role names (eg. enrolls)
- multiplicity (cardinality)
- Navibility



Multiplicity –

1 — one & only one

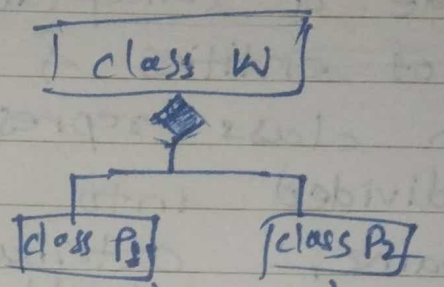
0..1 — zero or 1

M..N — from M to N (natural lang)

0..* — from zero to any +ve integer

1..* — from one to any +ve integer

Composition – when bond between classes strong, solid diamond will be

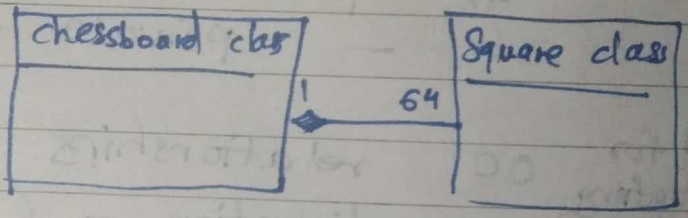


whole class

whole will deleted parts will automatically gets deleted

part classes.

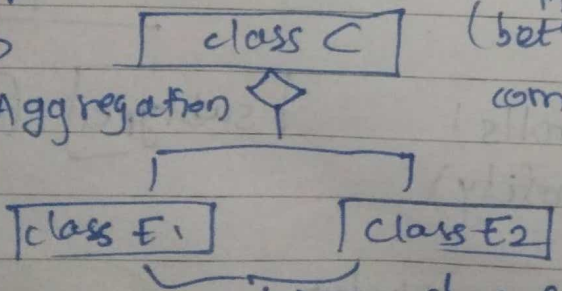
eg. chess if chess board will throw away, all 64 square belongs ~~to only~~ to on that board go as well.



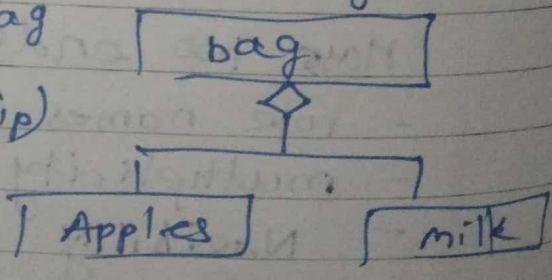
Aggregation – relationship among instances of related classes.

specific kind of container - containee relat – apple can survive without bag, but milk can't (both milk & bag composition relationship)

container class



containee classes.



express more informal relationship than composition.

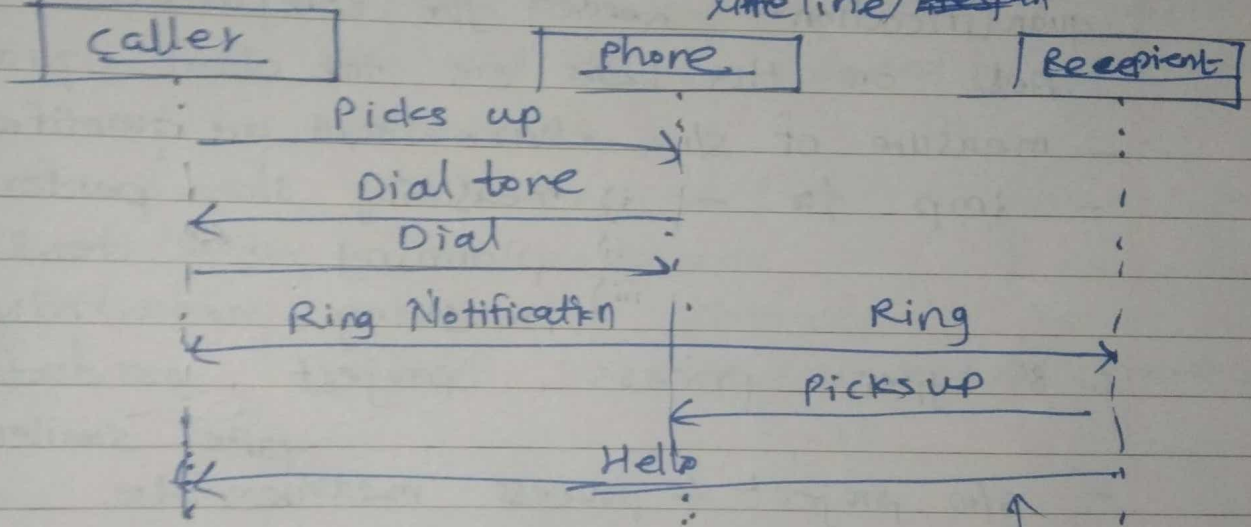
Aggregation vs. Composition
- Compo. is strong form of association
- Com have only one owner.

* Interaction diagrams-
- show how obj. interacts with one another
-

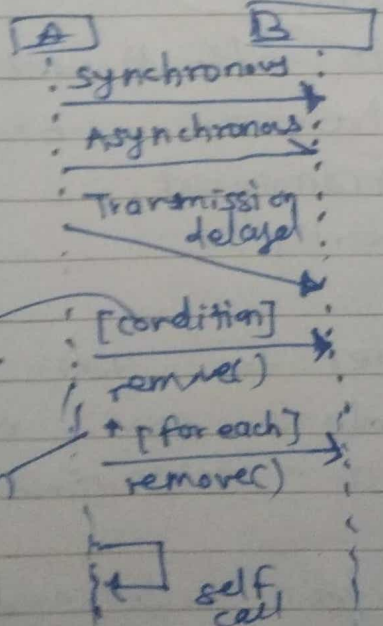
i) sequence
ii) collaboration

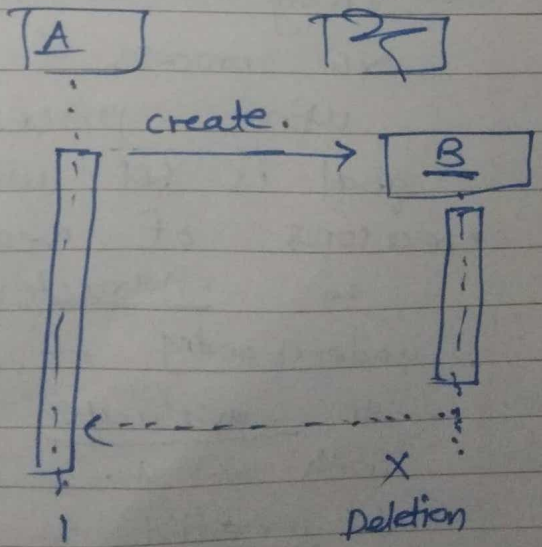
↓ below — within name of square → obj. not closes.

by dotted line - lifeline/~~lifespan~~



(flow of interaction)





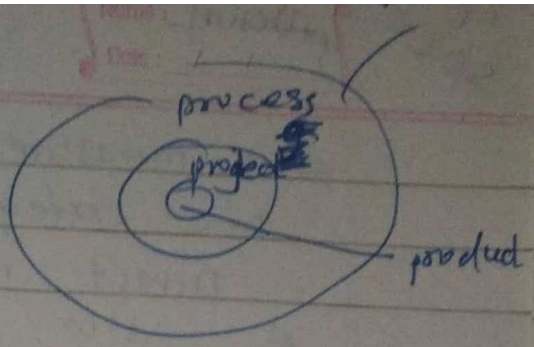
B should end before A ends

Use sequence diag. when the transfer of info is the focus of attention.
Use collaboration when concentrating on data. Collaboration dia. are equivalent to Seq. dia.
Ask anyone thursday's notes.

03/04/2023

* Software metrics — measuring / measing w/o
  - units to be needed for measure.
  - quantification needed for comparison & analysis. Next on this basis can talk about quality.
  - measure of s/w char. that are quantifiable/countable
  - imp for — i) measuring s/w performance
    ii) planning work items
    iii) measuring productivity, etc.
  - 3 types: process, project, product. (quite similar. since o/p of project is product)
  - s/w project & process metrics are quantitative measures: used a management tool. also helps product metrics.
  - They offer insight into effectiveness of s/w process & project that are conducted using process as framework.
  - goal is set wealth productivity
* reasons of measure
  - to characterize in order to - gain an understanding of proc., product, envi, resource
  - to evaluate in order to - determine status with r.t. plans
  - to predict in order to gain understanding of relationship among process & product.
  - To improve in order to - identify roadblocks, root causes, inefficiencies & other oppo. for improving product quality, performance

other options -
1) Opinion mining
2) Sign lang recognization



* Process metrics -
1) - strategic decision. (strategic for long time)
used for making
- the intend is to provide process indicators that lead to long-term s/w process improvement
- product char. helps to find out process metrics
- outcomes of process can be assed by
  - errors uncoverd before release of s/w
  - defects delivered to and reported by end user.
  - work product delivered
  - Human effort expended
  - calendar time expended
  - conformance to the schedule
  - time & effort to complete each generic activity

Project metrics -
helps project manager to asses states of ongoing project, tract potential risk, adjust project flow/ task, uncover problem areas befor their status become critical, etc.
- 1st appli. during estimation
  - metrics from past - basis for estimating time & effort.

productivity rate -
reviews -
used to - 1) minimize the development schedule the adjustments necessary to avoid delays & mitigate potential & risks
2) asses product quality on ongoing basis & when necessary, to modify technical approch to improve quality.

S/w measurement -

two categories

- Direct measure of - 1) s/w process (cost, effi
a) s/w product (lines of code produced, exec
speed, defects)
- indirect measures of
· s/w product (functionality, quality, complex
efficiency, reliability, maintainability, etc)
project metrics consolodate to form proje
process metrics

ate - 16/04/2023

S/w metrics - use. for project proper planning, for goal setting, ~~we~~.

- product oriented.

ofunction oriented metrics - CPMCD

function points -

function point analysic - we can ~~used~~ meanse ~~it~~ s/w in function point.

- unit of measurement of s/w.

ate - 20/04/2023

Testing in nutshell -

deconstructive phase - for finding errors (bugs)

Two ways - manual & automated.

- process used to identify correctness, ~~completeness~~ & quality of developed product / computer s/w.
- It checks for 1) specification - (SRS)
a) functionality 3) Performance
- It is process of executing a program/ appli. under +ve & -ve conditions by manual or automated systems.