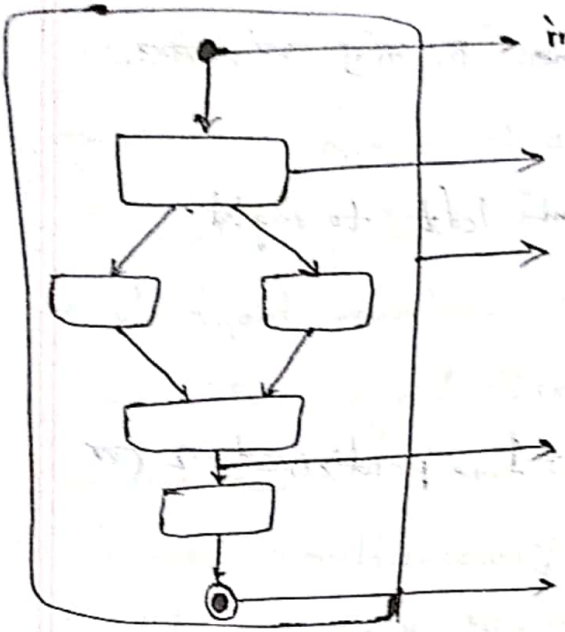
Activity Diagram :

Activity is a group of task.

Activity - তে decision নিতে হয়.

if, else - এর condition থাকতে পারে, অর্থাৎ if/else - এ guard দেয়.



object. status present করে, like clothes রঙ অবস্থা বোঝে. dry or weight.

if. else কোনো জায়গায় split হয়ে থাকে, then join হয়ে একত্রে. তাই এই join হওয়াকে আর merge node বলে।

decision node → ◇ → incoming edge একটা, outgoing edge multiple.

merge node → ◇ → incoming multiple edge হয়, outgoing - এ only one edge.

Parrallel node → fork + join ═══ incoming edge একটা. outgoing edge multiple

synchronously → একসাথে

Sequence Diagram

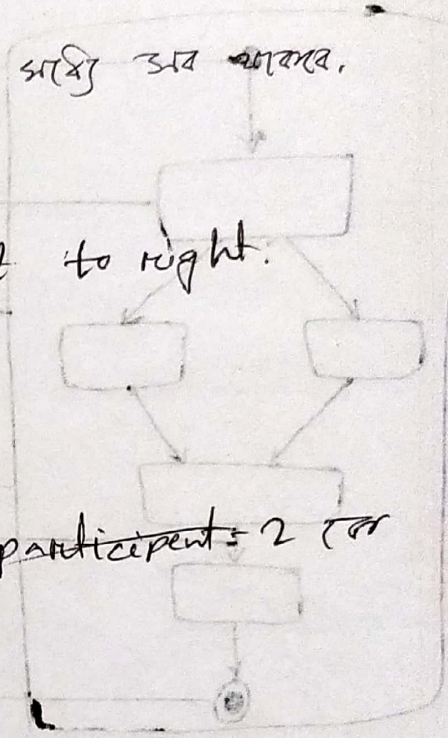
Classe -এর object -গুলোর মধ্যে interaction থাকে

use-case -এ যেমন activity কোথায় গিয়ে কোথায় specifically বলা ছিল না, but sequence -এ আমরা এটা বলবো। কোন activity -র পর কোন activity হবে। তা sequence diagram -এ থাকবে,

Frame use করতে হবে, frame -এর মধ্যে সব থাকবে.

Time will ~~to~~ move from top to bottom.

object, actor/ interaction more from left to right.



---- → life line

~~synchronous~~ = ~~Participent~~ =1, ~~participent~~ = 2 কে msg দিবে.

synchronous - ⟶

Asynchronous - ⟶

Msg return to the sender - - - - -> / Rply msg

object, class হইয়া চলমান যাবে না.
class, object " " যাবে,

State Diagram / stat transition / state chart

এটা Activity diagrame-এর মতোই similar

Activity diagram-এ আমরা একটা activity বা action গুলোর sequence maintain করি, like কোনো action করার আগে guard বা decesion-এর দরকার আছে কিনা, বা কোনো action simultanously হচ্ছে কি না - এই বিষয়গুলো আমরা ~~actit~~ activity diagram-এর মাধ্যমে present করি,

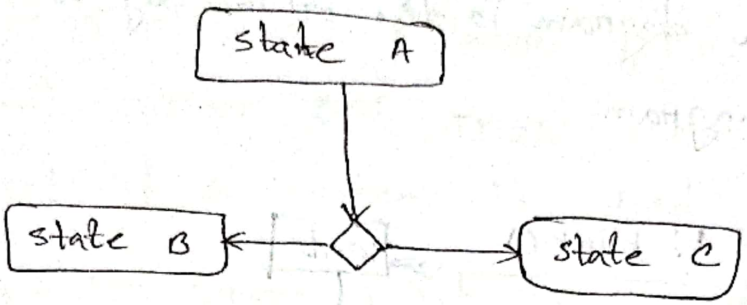
state diagram - এ particalar কোনো একটা object-এর status-ই কোনো একটা event-কে respond করে কি কিভাবে change হয়।

activity diagram-এ multiple ~~object~~ action থাকতে পারে,
state diagram-এ
state diagram is related just only one thing.
একটা application কিংবা এই application-এর অংশই সব থাকবে,

Event-এর মাধ্যমে particalar কোনো object-এর status change হয়,

 → state  → start  → end.

এক state থেকে অন্যের state এ যাওয়ার জন্য decision mode - এ থাকতে হবে।



#fork and join is common for activity diagram. It's rare for state diagram.

Change event: bank like bank account - এ ~~৳~~ 1000 এর বেশি balance থাকলে টাকা তুলতে না (balance <= 1000)

Call event: Particular কোনো action - কে call করা, যেমন. call autopilot.

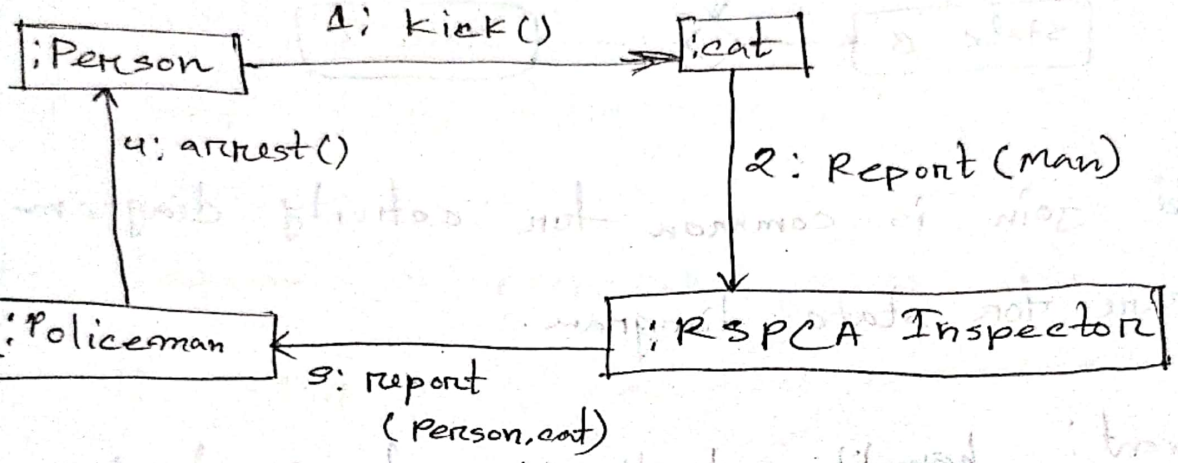
Time event: কোনো transaction - এ যদি ৫ 10 sec inactive থাকে তাহলে/ তখন

Time events marks the designated period of time e.g. after (10 second)

Collaboration Diagram

sequence diagram -কে চিত্রাকারে collaboration diagram দিয়ে design করা যায়,

→ collaboration diagram is the other way to represent sequence diagram.



→ এই diagram -এ কোথায় শুরু থেকে শুরু হচ্ছে তা বুঝানো action number দিয়ে, আর এইখানে যেমন cycle create হয়েছে তা always এমন নাও হতে পারে,

Component & Deployment Diagram

এখানে input এবং output গুলোকে present করা হয়। design-এ
অধ্যায়ে,

module → কোনো একটা software system-কে যদি আমরা part by part ভাগ করে নেই, তাহলে i each of the smaller part, sub-set of a system is called module.

এই module-এর মধ্যে input এবং output যে জিনিসগুলো দিয়ে আমরা করে যারা সেই জিনিসগুলোকে বলা হয়
component - base diagram.

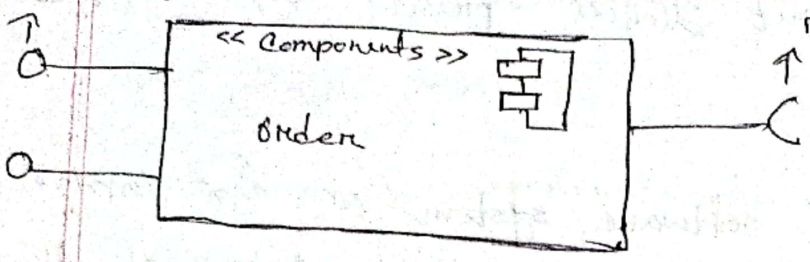
Deployment diagram-এ আমরা দেখাবো যে hardware-এর সাথে কীভাবে software interacting করে একটা system-এর operation করছে।

component হচ্ছে modularity-র একটা part.

Modularity হচ্ছে complex একটা জিনিসকে logically connected যে function-গুলো আছে, তার উপর base করে বিভিন্ন part-এ part-এ ভাগ করা.

→ Total system-কে কিছুসংখ্যক part-এ divide করা হয়, এই devision-কে বলা হয় modularity আর প্রত্যেকটা part-কে বলা হয় module বা component.

Provided / output



Component - এ input ও output - এর জন্য দুই ধরনের Interface থাকে,

1) Provided interface → ~~out component~~ output
2) Required interface. → input.

→ component - এর data other component - এ supply দিবার জন্য output.

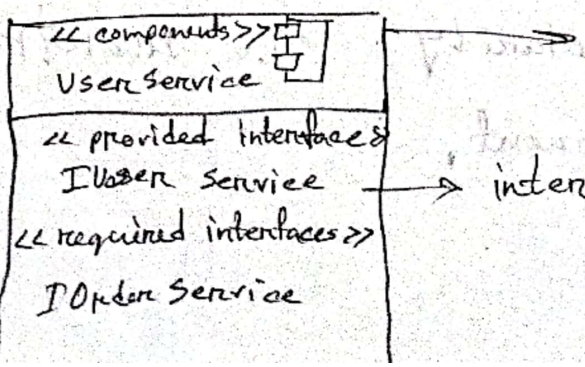
→ কিন্তু component কোন input নিবে,

port can support un-indirectional or bi-directional communication

□ → port, input, output - এর জন্য port use করা হয়ে থাকে,

Comoponent - এ external & internal view থাকতে পারে,

↓ black box view / ↓ white box view



Port - এর মধ্যে দুইটার connector থাকে,
Assembly & deligation connector

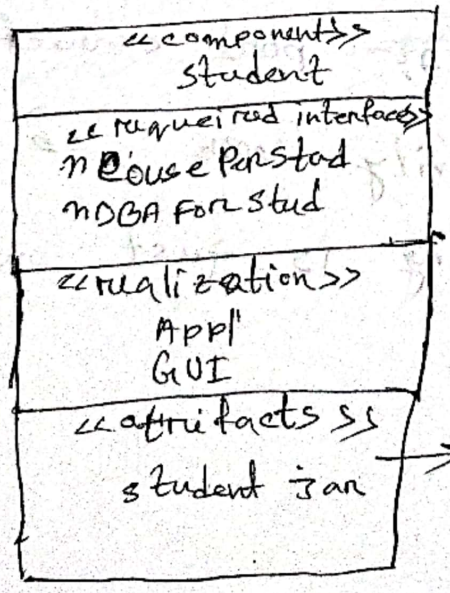
→ একটা component - এর under - এ আরো অনেক component থাকতে পারে।

→ এর আবার package - এ হতে পারে, একটা package - এ under - এ আরেক package থাকতে পারে।

→ internal component কীভাবে external component - এর সাথে connect হতে পারে?

→ Assembly & deligation connector - এর মাধ্যমে.

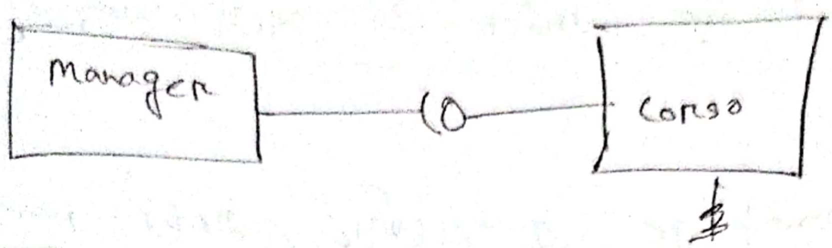
→ interface যে ক্লিনিসগুলো আছে তাদেরকে আমরা realization বলতে পারি.



→ some kind of physical material, যেমন : document.

Assembly Connector:

A component is connected to the other component



~~this~~ This assembly connector is notated by a ball and socket " connection.

Delegation Connector:

external interface/port থেকে internal interface/port এ একইরকম connection দেয়া থাকে, এই connection-ই হলো Delegation। ~~connector~~।

larger component-এ, একটির component-এর port-এর সাথে অন্য sub-component-এর port connected হবে, এর মধ্যে যে interconnectivity থাকে, এই বিষয়গুলোকে বলে delegation. , its working is just one component.

Deployment Diagram:
some physicalb materials might be there.

Software Metrics: It's common measurement entities.
~~module, size~~ ; It helps us to evaluate →
Performance, efficiency, security, Complexity, Testability, Flexibility etc.

Size: (LOC) → Line of code.

Complexity: Cydomatic code complexity,
একটা code -এ কত logic আছে, সেগুলো কতো complex or cyclomatic complexity দিয়ে বের করা হয়.

⇒ Previous similar code-এ, ~~size~~ size দেখে size এসা- যায়

MCCABE's cyclomatic Complexity:

Previous code-এ যেগুলো statement-কে node
দ্বারা represent করেছে.

conditional node-এর incoming edge → একটার
outgoing " → multiple.

⇒ Cyclomatic Complexity = (Total number of edge - total number of node) + 2

⇒ Cyclomatic complexity = total number of Path

⇒ cyclomatic complexity = Total number of decision node + 1.

⇒ " " = Region number.

Weighted Methods Per Class (WMC).

~~It~~ number of methods in the class and their complexity level.

প্রত্যেক method খুঁজে বের করি, এবং method গুলোর complexity-ও বের করি।

DIT → Depth of Inheritance Tree.

It is multi-level inheritance

একের অধিক parent, ...

parent
↓
child
↓
child
↓
child

generation বাড়ে থাকে।

~~ame~~ NOC →

multiple inheritance

একটি parent class যদি অনেক গুলো child class কে inherite করে, তাই ~~হলো~~.

→ DIT বেড়ে গেলে coupling ও বাড়বে।

→ DIT -এর value zero হলে সে class -এর কোনো child class থাকবে না, এটা হল root.

→ child -এর সংখ্যা বাড়ে থাকলে, DIT -এর সংখ্যাও বাড়তে থাকবে।

→ DIT বাড়তে থাকলে maintainability, Portability, efficiency কমবে, এবং Functionality বাড়বে।

→ ~~DIT কমলে~~

NOC → Number of children :
multiple inheritence
এক্ষেত্রে parent class -এর যত multiple child class থাকে।

এখানে immediate child -এর সংখ্যা বলা হবে।
NOC বাড়লে থাকলে functionality বাড়তে থাকবে।

CBC → Coupling between Classes :

complexity দেখার জন্য দুইটি way তে দেখা হয়ে থাকে,

I) cohesiant , II) coupling.

part গুলোকে modue বলে,

→ একটি moduel এর মধ্যে যে function গুলো থাকে তাদের মধ্যে interconnectivity কে ~~তাদেরগুলোকে~~ বলে cohesiont.

→ ~~দুই বা~~ multiple modul এর মধ্যে যে function গুলো থাকে তাদের মধ্যে যে interconnectivity থাকে তাদেরগুলোকে বলে coupling. / ~~CBV~~. CBC.

→ particular কোনো design এর জন্য CBC যত কম থাকে তত ভালো, এবং cohisiont যত বেশি হয় তত ভালো, কারণ আমরা কোনো design এ কে part by part ডেভেলপ করা করবে, তাহলে কাজ করতে সুবিধা হবে, complexity কমবে,

LCOM ⟶ Lack of cohesion in methods

diff between LCOM & CBO & cohesion

LCOM বেশি হলে CBC বেশি হবে and cohesion কম হবে,

coupling বাড়লে cohesion কমে

cohesion কমলে LCOM বাড়ে,

coupling বাড়লে LCOM বাড়ে-

~~LO~~ LCOM = |P| - |Q|, if |P| > |Q|, otherwise 0.

Q = set of all cohesive pair of methods

P = set of all non-cohesive pairs of methods.

COCOMO → Constructive Cost model.

এই model এর সাহায্যে particular software এর effort, time & human resource নির্ণয় করা যায়।

PM = Person month, SLOC = Source line of code.

Effort = PM = Coefficient <Effort Factor> * (SLOC/1000)^P

P = Project এর complexity.

T =

Design Pattern

বিভিন্ন-বিভিন্ন problem/ common design থাকে project-এ,

pattern name এবং objective, এবং step by step process বা solution এবং এর result, এই জিনিসগুলো আগে থেকে define করে রাখলে next time এমন problem face করলে just pattern name call করলেই ঐ ভাবনা মাথায় চলে আসবে,

Adapter: একটা ~~ল্যাপটপের~~ laptop-এর charger এর পিন like একটা পিন data cable এর সাথে connect করার জন্য Adapter use করে, input একটা হলেও output same হয়,

Facade: It is just like a control panel, internal function এবং logical complexity hide-এর জন্য use করা হয়,

Observer: one to many relationship.
teams-এ msg.

Singleton: particular কোনো একটা class এ সবসময় define করা যায় only একটাই object create করা যাবে,

→ One of a kind object.

Strategy: It allows selection of one of several algorithms dynamically.
algorithms select at run time.

Factory: same ভিন্নভিন্ন object -এ select অনুসারে সেই object select হবে run time-এ

Software Matrix: It's a common measurement entity. It helps as to evaluate → Performance, Efficiency, Security Complexity, Testability, flexibility etc.

Size: It's counting the number of lines of code (LOC)

Complexity: Cyclomatic code complexity. → একটা code এ কয় logic আছে, সেগুলো কত- complex বা একটা দিয়ে- বুঝে নেয়া যায়;

MCCABE's Cyclomatic complexity:

→ Cyclomatic complexity = (Total number of edge + total number of node) + 2

→ " " = total number of path

→ " " = total number of decision node + 1

→ " " = Region number

1 – 10 ⟶ simple Program

11 – 20 ⟶ More complex, moderate risk

21 – 50 ⟶ complex, high " "

> 50 ⟶ Un-testable prog, very high risk

diff. between activity & state.

→ Activity হচ্ছে background logic কে বুঝায়।
state - এ object কোনোর মধ্যে step দিয়ে যায় হয়।

⇒ Provided interface → output. ০
component-এর data other component-এ supply দিচ্ছে।

→ Required(I) → input → ভিন্ন component থেকে input নেয়।

→ LCOM & coupling diff.
Lack of cohesion হবেনা হলে CBC বেশি and cohesion কম।

→ sequence & collaboration diff.
Sequence দিয়ে event বা activity গুলোকে sequencially arrang-এর মাধ্যমে বুঝানো হয়।
collaboration এ প্রতিটি sequence-এর জন্য number দিতে হয়।

→ Internal & External view:
External হচ্ছে আমরা যে sign use করি like rectangle এবং provided & required interface-এ sign.
Internal হচ্ছে এগুলো ভাবনা দিয়েছে এবং সঙ্গে হবেনা description.

internal view

Delegation connector: যা একটি external interface থেকে internal interface - এ একপ্রকারের connection তৈরি করে।

এইরকম - এইর