UNIT IV

User Application Analysis :-

Application Interaction Model :→

- App Interactn model is built because boundary of system is determined.
- Then identify usecase, prepare scenarios & design sequence diagram.

Steps to build app interaction model :-

(1) Determining system Boundary :-

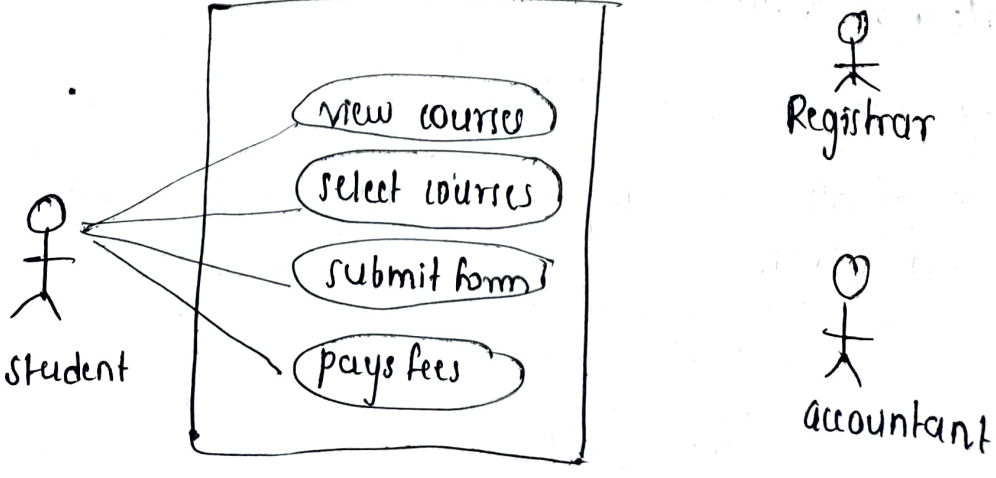
- It is the scope of app. It help to specify the functionality.

2) finding actors :-

- Actors are external objects that interact with system.
- actor can be human, external device & sw systems.
- Ex:- online course Registratn system, student, accountant are actors.

3) finding use cases :-

- use case represent fundamental unit of functionality & behaviour of system.
- Ex:- online course registration system, use case :-



4). finding initial & final events :-

-) Initial event is request for some service or chain of activity
-) final event is complete operational scope of app.
-) ex:- student views courses. → initial
  student pays fees & confirms admission → final.

5) preparing normal scenarios :-

-) scenario is sequence of events among interacting objects.
-) for example :-

submit form → system display registration form.
student fill form
student click submit
form validatn by Registrar.

pays fees → system displays option
student select payment mode.
student pays fees
admission confirmatn.

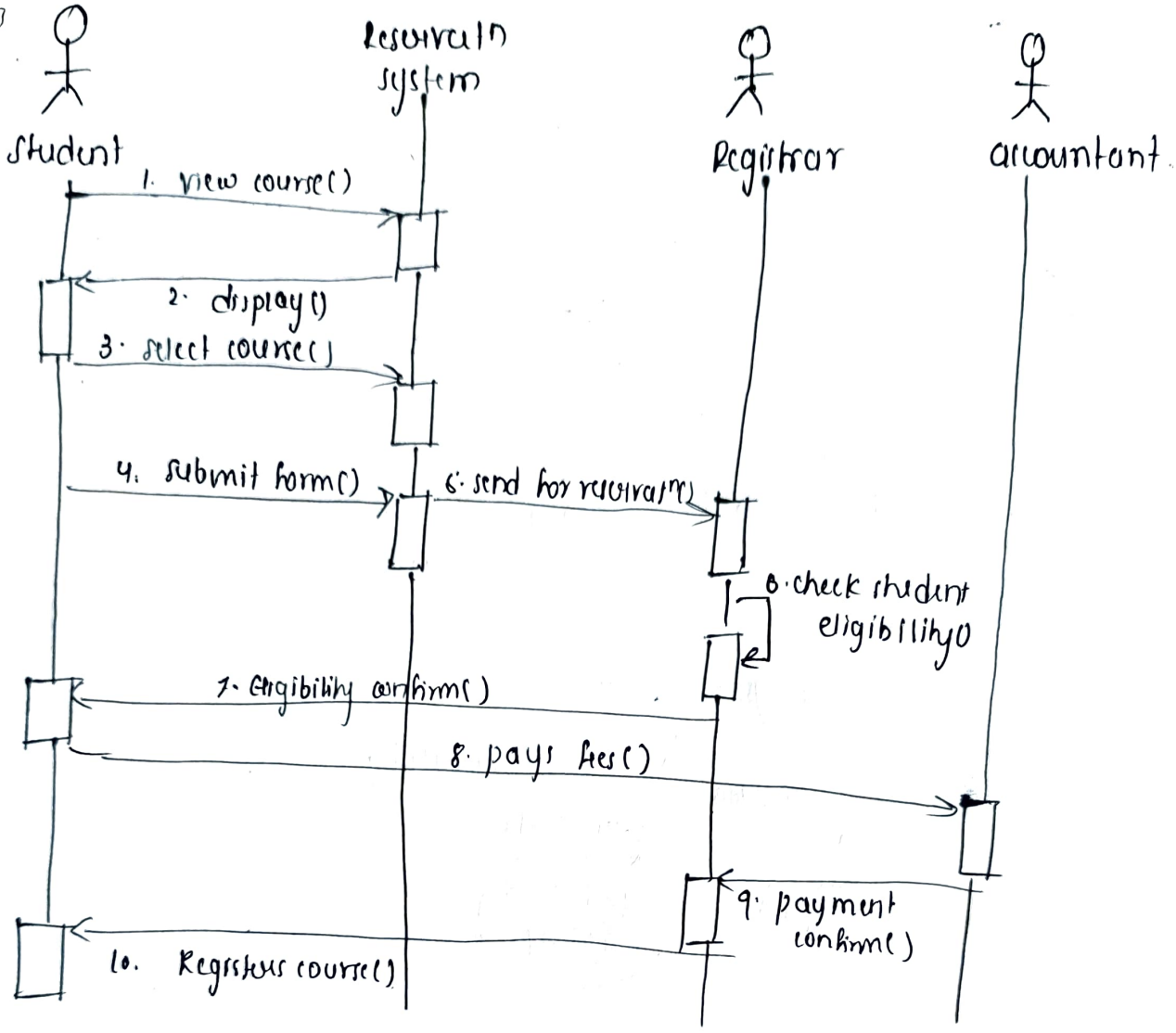
6) adding variations & Exception scenarios :-

→ additional scenarios can be added to usecase modeling.

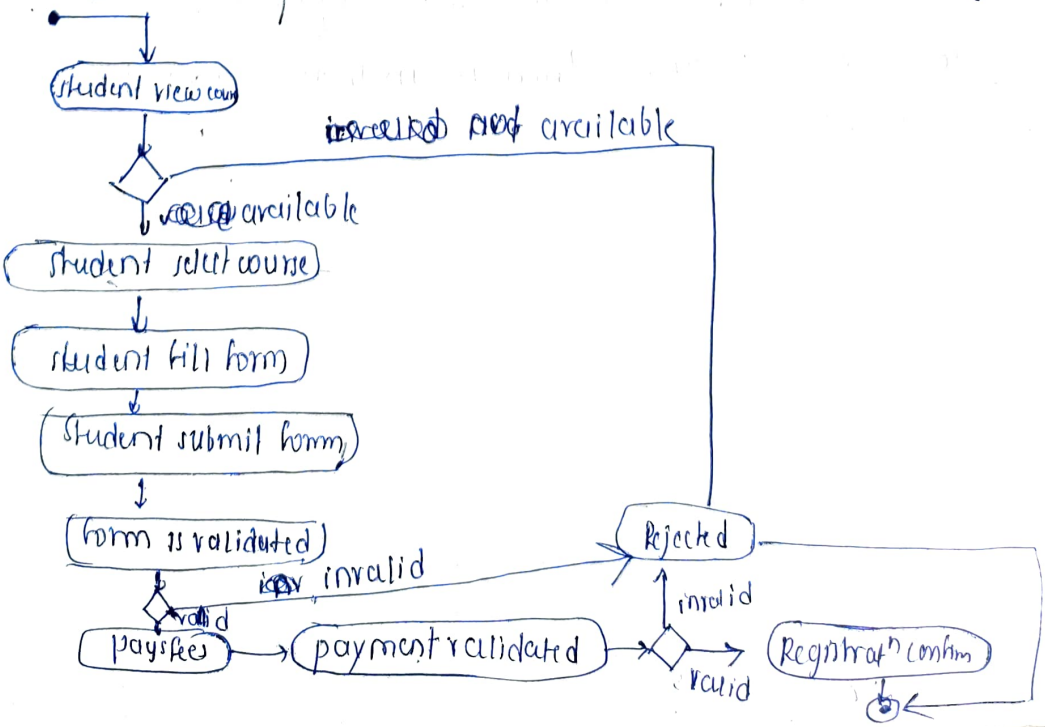
7) filling external event :-

→ scenarios are observed to find external events.
→ external events are inputs, decisions, interrupts & interactn from user.

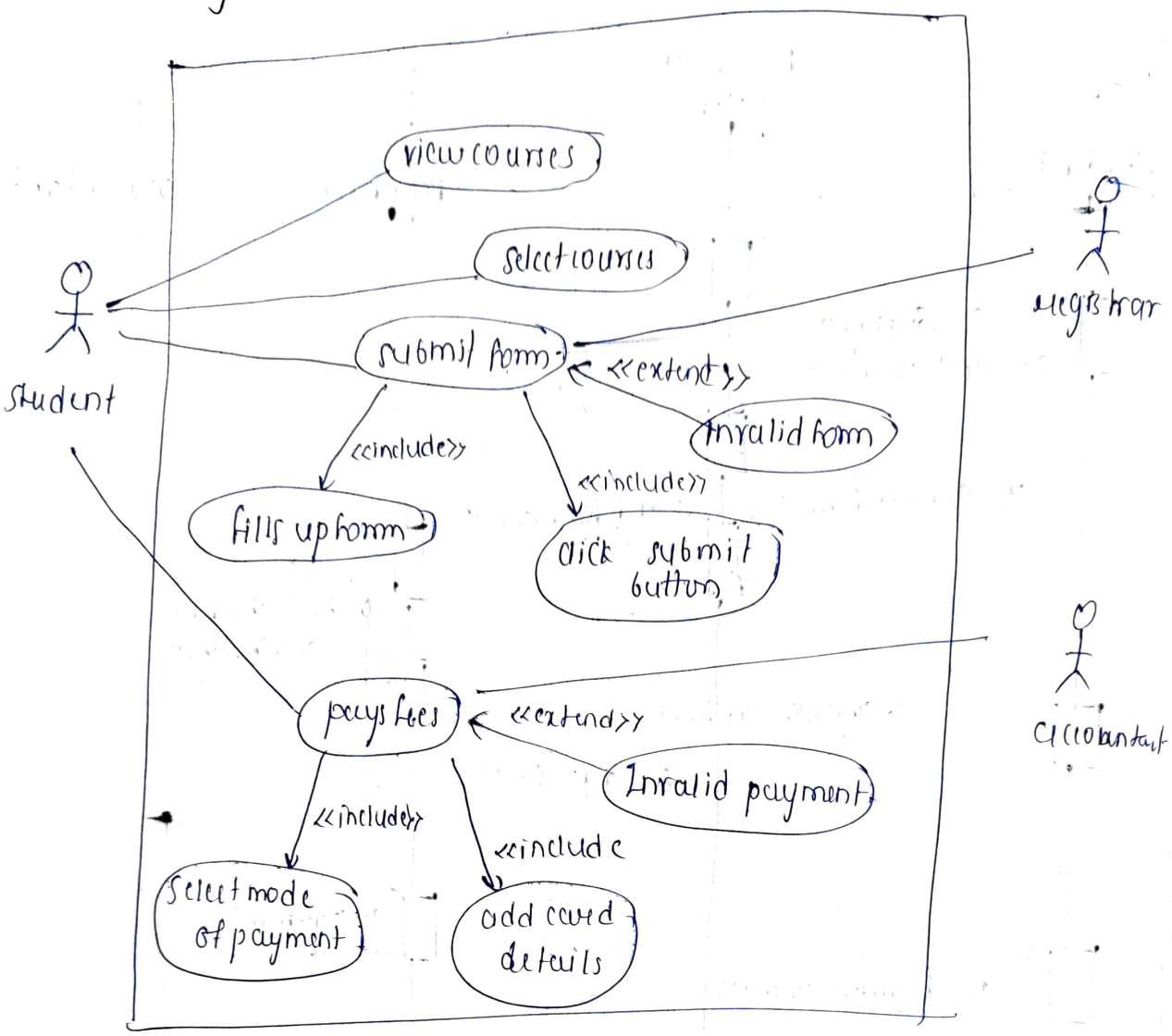
Example :- sequence diagram for online reservation system.



8.) create activity diagram :-



9. Organizing actors & usecases :-



10. verifying against class model :-

• app model must be tested against class model. If both models are consistent then it indicates all data & event parameters are consistent.

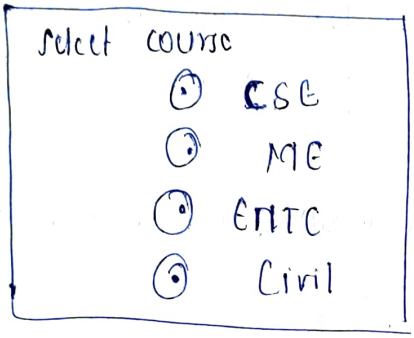
Application class model :-

App class model can be constructed using :-

1) specification of UI :-

- UI is a collection of objects which allows user to access domain objects of system
- look & feel of UI can be tested by decoupling the app from the interface.
- Ex :-

Online course registration system



2) defining boundary class :-

- Boundary is a class that helps in communicatn blw system & external source.
- It understands the format of external source & converts information to internal system. & vice-versa.
- example :- Keyboard.

3) determining controllers :-

- It is ~~object~~ an active object that controls internal operations of app.
- It receives signal from outside world, then invokes required operations & send the signal to outside world.
- ex :- checking eligibility of student.

4) verifying against interaction model :-

•) If some value is requested by actor from system, then that value must appear as attribute of UI class.

Application State Model :-

•) It focuses on app classes.

* Steps :-

1) identify the app classes with states.

•) In application class model the classes that are computer oriented & contain multiple operations are considered for being the states.

•) user interface classes & controller classes are good candidate for state.

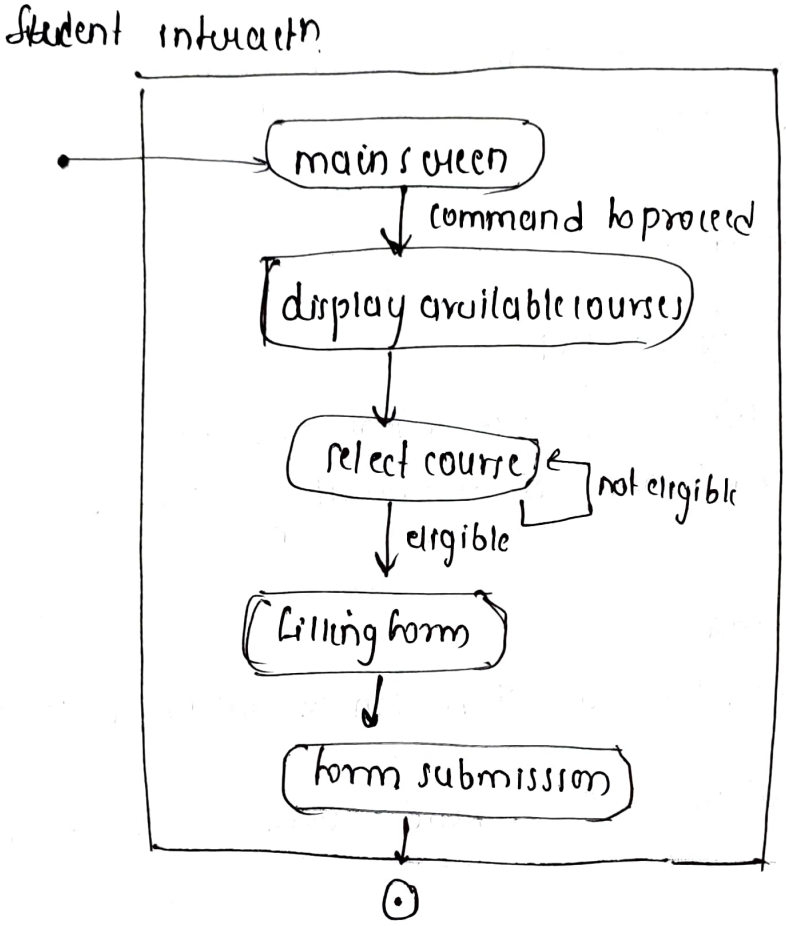
2). find events :-

from interaction model, study the scenarios & find out the events.

3) design state diagram :-

4) Check against other state diagram :-

-) Each state diagram is checked for its completness & consistency
-) State must have predecessors & successors. & starting & terminatn points.



5) check against class model :

-) state diagrams must be consistent with domain & app class model.

6) check against interactn model :

- Behaviour in state model must match with behaviour exhibited by sequence diag.

Addition operations :-

* operations from class model :-

→ classes have attributes & associations and their values need to be read or written. Hence set of operations can be obtained from class model.

* operations from usecase :-

→ complex functionalities of system can be obtained from usecases.

* shopping list operations :- for online course reservatn system.

System Design :-

It is first stage in which developer decides structure & style.

- It helps in taking important decision :-

Decisions are :-

(1) Estimating performance :-

•) performance Estimation process should be fast & progressive in nature.

•) Basic need, the sw developer should try to simplify assumptions made during requirement analysis phase of sw development.

•) Main aim is to determine feasibility of system.

•) This process works in two activities predictn & approximatn.

•) sw developer predict & approximate the performance of system & then followed by estimation process.

(2) Making Reuse plan :-

•) Reuse plane is considered as key advantage of object oriented technology. Reuse should be well planned.

Reusable things include :-

(1) Libraries :-

•) A library is collection of classes that are convenient & valuable in various circumstances.

•) characteristics of good class libraries :-

1) Efficiency : library provide efficient implementation of alg.

2) consistency : operations should have consistent names & signatures across classes.

3) coherence : class library should organized about limited themes.

4.) Generic :- library should use parameterized class whenever required.

•) some problems arises when the class from libraries are integrated
   1) argument validation
   2) Error handling
   3) control paradigm
   4) Garbage collection
   5) Group operations.

2) Frameworks :-

•) It describes broad approach to solve particular problem.

•) It is used in order to design & develop complete s/w system app & expanded by s/w designers.

•) Frameworks categorized into two categories :-

→ Blackbox framework :- internal structure of s/w prog. is hidden from end user.

→ whitebox framework :- end user & designers can view all the components.

3) patterns :-

•) A pattern is an established & confirmed s/n to general problem.

•) there are patterns for analysis, architecture, design & implementation phases of SDLC.

•) pattern is more suitable to apply.

Breaking system into subsystems:

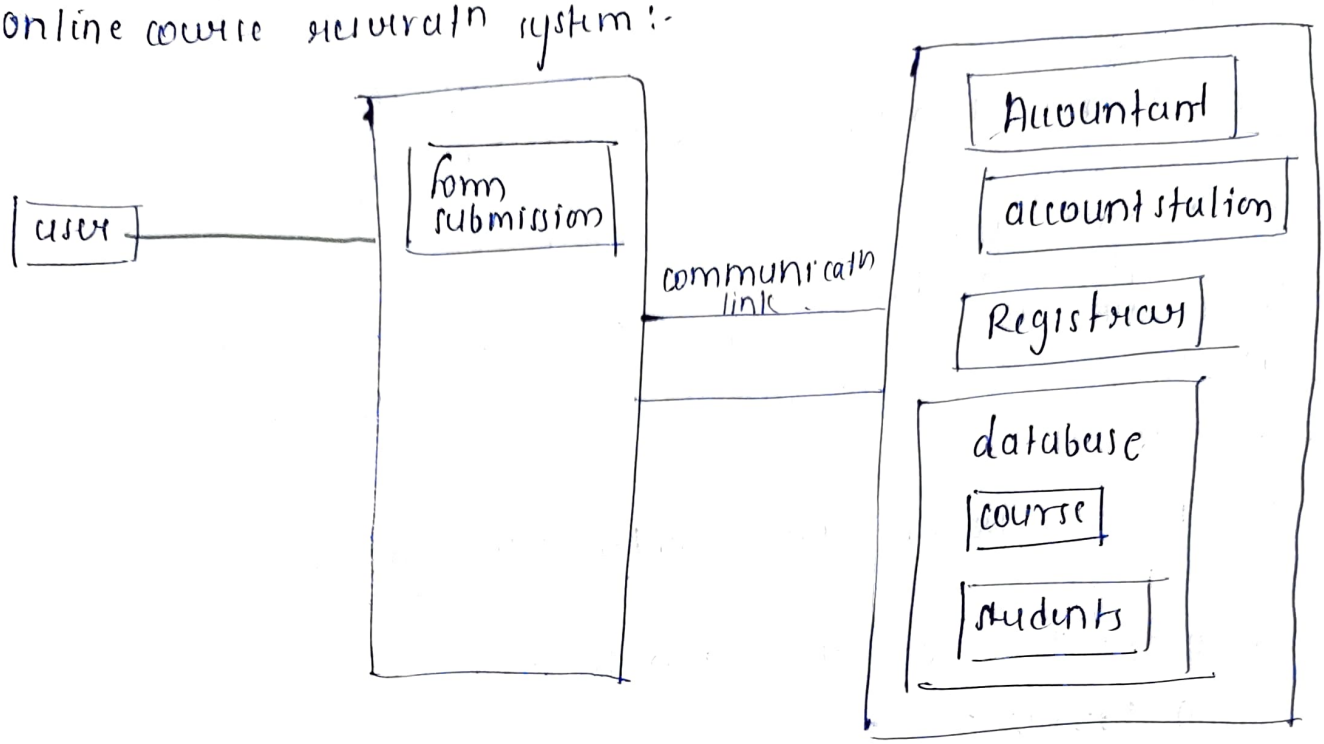
:) A subsystem is group of classes, associations, operations, events & constraints that are interrelated & well defined.
:) subsystem is known by its service.
:) A service is cluster of related functions that share I/O procedures drawing pictures purpose/or collection of functionalities that it provides
:) There are two relationships between subsystems :-
1) client/server - client invoke server, server then provide service to its client.

2) peer to peer : each subsystem calls other subsystem.

:) decomposition of system can be :-
1) layers :-
:) It is collection of tiers that are built one below the other.
:) Lower layer provide implementation basis for layer just ~~layer~~ ~~at~~ above it.
:) objects in each layer are independent but they can communicate with each other.
:) client server relationship exist blw lower & upper layers.
:) There are two types :- closed & open
2) partitions :-
:) It divide system vertically
:) subsystems created by partitioning are loosely coupled.
:) Each subsystem provide a knowledge about other
:) It is have peer to peer relationships.

•) Combining layer & partition:-

Ex. Online course registration system:-



☆ identifying concurrency :-

•) using state model concurrency blw objects can be identified.

•) If two objects receive events at same time without interacting then they are called inherently concurrent objects.

•) If two subsystems are inherently concurrent then it is not necessary to implement them using separate blw.

•) A thread of control is execution path in state diagram on which only single object at a time is active.

Allocation of subsystems :-

) System is broken down into subsystems & each concurrent subsystem is allocated to hardware unit.

The allocatn is as :-

1) Estimating hlw resource requirements.
2) Making hlw - slw tradeoffs. → which subsystem must be implemented on hlw & which on slw.
3) Allocating tasks to processors. → communicat? & computatn limits
4) determining physical connectivity. (logistic)

Management of data storage :-

•) data can be stored in data structure, files & databases.

•) personal computer system app can make use of file & ds as per requirements.

•) data kind, suitable for file are :-
- → sequentially accessed data
- → data with low data density.
- → data with high volume.

•) databases are type of data store which are managed, controlled by dbms.

•) data kind, suitable for dbs :-
- → Large amount of data
- → data that synchronized the updates
- → data accessed by apps.
- •) data should be protected against malicious access

•) slw designer must consider object oriented dbms for domain apps.

Global resources handling :-

-) slw system designer should recognize global resources.
-) types of global resource :-
  - → physical units :- processors, communicatn satellites
  - → space :- button on mouse
  - → logical Names :- object IDs, filenames
  - → access to shared data :- databases
-) when physical units are considered as resource, it can regulate & control all activities in scenario.
-) global resourse have self governing control.
-) If resourse is logical entities then there are chances of conflicts.
-) for this purpose guardic objects are used.
-) It controls the access of resources.
-) the shared global resource can partitioned logically & each partition can controlled by separate guarding object.

Choosing slw control strategy :-

-) control is used to control interactn blw objects.
-) there are two types of control in slw system :-
  1) External control
  2) Internal control.

External controls :-

) It manages externally visible events among objects.
) It is implemented in three ways :-

★ procedure driven control :-

•) In this system control, control exists within sw program coding.
•) It is easy to implement by means of languages.
•) It is responsibility of sw designer to translate events into operations.
•) drawback is to, concurrency inherency is significal among object in scenario.

★ Event driven control :

•) In this, it is associated with the circumstances where measurement method is event based in nature.
•) It offers acceptable & compliant control.
•) it is more challenging to implement as compared to procedure driven.

★ concurrent control :

•) In this, one task can wait for input but at same time another task can continue its execution.
•) If there are multiple CPUs then the tasks execute concurrently.
•) 3 basic classes of concurrent contro :- pessimetic, semi optimistic, optimistic.

Internal control :-

•) it get executed when many lower level operations get executed on same or another object.

Handling Boundary conditions :-

~~Termination~~ Initialization :-) System begin execution from initial state to steady state.

-) In initial state, system initializes global vars, parameters

~~Fault~~ Termination :-) In which task to be accomplished get completed.

Failure :-) arises due to internal breakdown, exhaustion of system resource or errors.

Setting Tradeoffs priorities :-

-) making compromise in one goal in order to achieve some other more important goal.
-) during design, it is responsibility of system designer to set priorities of system.
-) It affects entire functionality of system.

Selecting Architectural style :-

1) Batch Transformation :-

-) In this, the info transformation is executed once on complete input dataset.
-) Main objective is to calculate an answer & is achieved by app for receiving inputs.
-) In this, SW developer should breakdown complete transformatn into stages
  -) create class model for input, output
  -) detailed out each stage until each operatn get simplified
  -) restructure the stages to obtain optimizatn

continuous Transformation:-

·) In this output varies when input is getting changed.

·) stages :-

→ break the transformatn into stages.
→ creat class model for input, output.
→ incremental changes to each stage can be identified
→ add objects to obtain optimizat?

3) Interactive interface :-