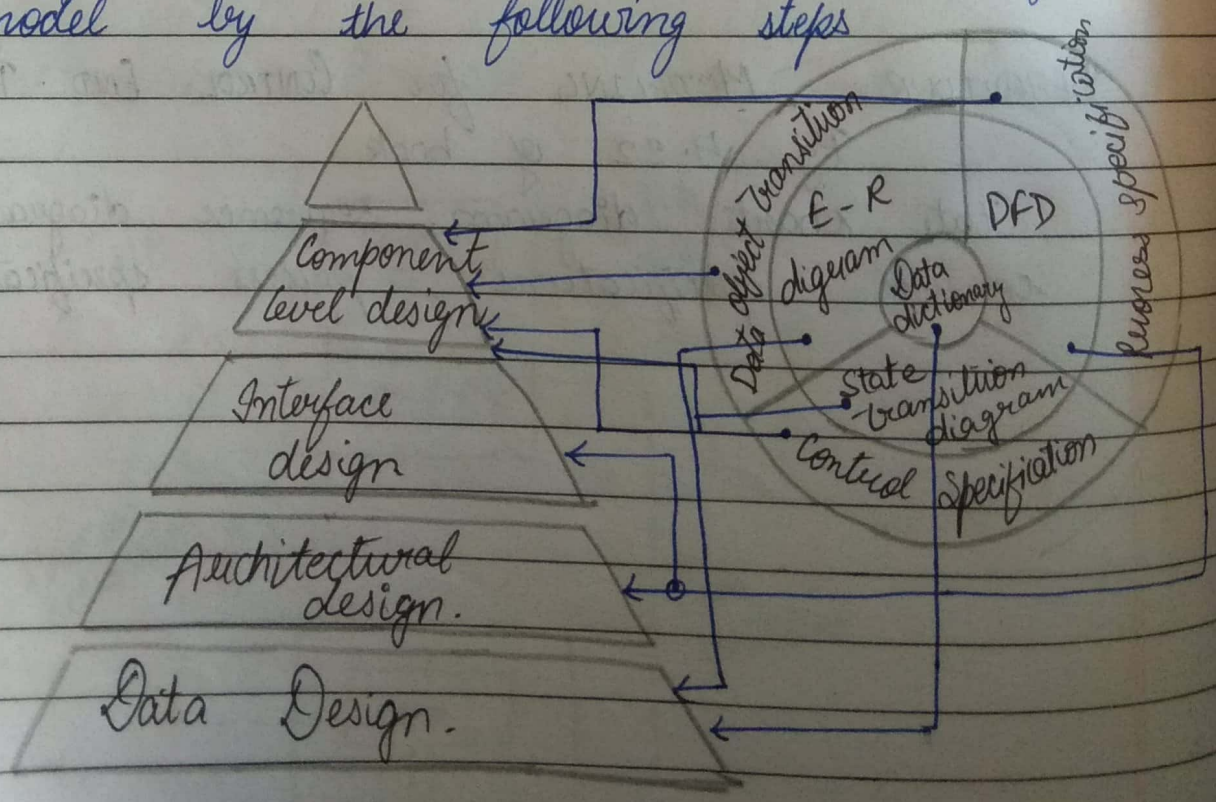
27/10/18

SOFTWARE DESIGN

Design is the process of applying various techniques and principles for the purpose of defining a process or a system which could permit the development of system.

When the software is decomposed into various modules with specific relationships design is produced to define architecture of system.

Analysis model is converted into design model by the following steps



Scanned by CamScanner

Step 1 → Data design with data dictionary and ER diagram.

ER diagram gives data objects and there relationships. Data dictionary provides details about the data that helps creating the data model of the system which further creates data structure for the system

Step 2 → Architectural design with data flow

Data flow diagram gives structural elements of the system which consists of various components of data model

Step 3 → Interface design with state transition diagram and data flow diagram.

Interface majouly deals with intercommunication part of the software and user computer communication or interaction dynamics.

Step 4 → Component level design

It includes control specification, process specification and data - object design to create component of the system. Further integration of which provides complete information of system to users

Scanned by CamScanner

DESIGN FUNDAMENTALS

① **Decomposition and modularity** : Decomposition is a technique which breaks the system as per functional similarities in the system. Higher leve description of the functions are used to implement and built lower level explaination of component execution and relation with other components.

Types of decompositions

Data oriented decomposition :- Data with similar data structure will be grouped under same module

Event oriented :- Events that will be happening in the system which will produce similar impact on the system will be grouped under same module

Object oriented :- Objects functioning with similar class and same interelations will be defined under similar module

User based :- This is the highest level of decomposition majouly depends on all possible users and their input which will be served to the system.

Scanned by CamScanner

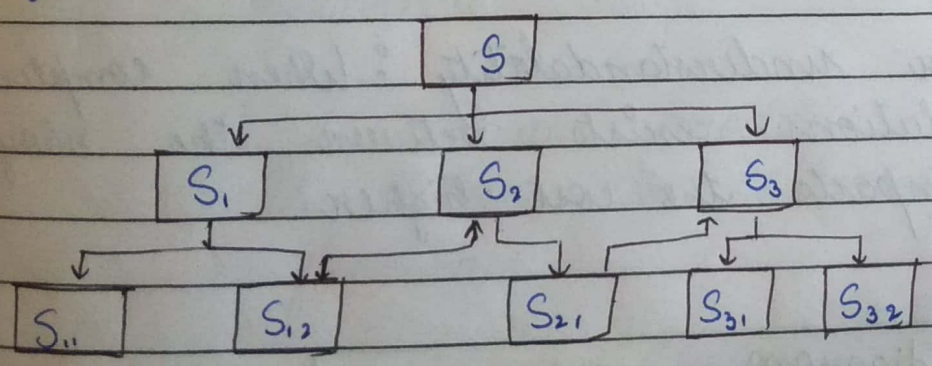
Modularity is dividing system into smaller modules such that when system is reintegrated it will produce results similar to sequential execution of the system.

Need for modularity: i) If system consists of P_n processes such that execution of

$E(P_1 + P_2 + \cdots + P_n) > E(P_1) + E(P_2) + \cdots + E(P_n)$

then it is better to produce the processes as the individual entity and then integrating it into the system.

ii) Abstraction is well defined with the help of modularity where the system can be used further for multiple applications.



If S_{11} could be used in other application, it is reusable without any desired complexity

(iii) Easy data structure modification is possible in modular elements of the system.

(iv) Changes can be reflected easily when the global data is present in the system.

Scanned by CamScanner

Date: __/__/__ Page no: ____

Cost Dependancy on Modularity:

i) It is not effective, if system consists of P_n process such that
$E(P_1 + P_2 + \cdots + P_n) > E(P_1) + E(P_2) + \cdots + E(P_n)$

ii) When the modular decomposibility is taking greater amount of time and effort it is not efficient to use modularity as an approach.

iii) Modular composability, ie, reintegration of all components is taking larger amt of time than it is not useful to use modularity.

iv) Modular understandability: When complex correlations exists between the subsystem it impacts the cost higher.

Peer diagram with more intercelations

v) Modular security: When increasing modules increase security req^t, modularity must not be used.

Scanned by CamScanner

18/11/18

2. Architecture : The overall structure of the software and the ways in which it provides integrity comes under architecture. Architectural design consist of various models —

a) Structural model
It defines component of the system and interaction between the components

b) Framework model
It defines the levels of abstraction that the system will possess

c) Process model
It defines the behaviour of the system.

d) Functional model
It defines the heirarchy of the functions.

3. Patterns : Common solutions to a common problem is generated under pattern designing
Patterns could be devided into —

a) Creational pattern
These patterns create themselves when they meet specific conditions

Scanned by CamScanner

b) Structural Pattern
It depends on the building type of applications

c) Behavorial Pattern
Change in the behaviour of system can be categorized under behavorial pattern.

4 Information Hiding : A good split of modules is required while they are communicating. So, only necessary information is visible. This is taken care under this.

5 Functional Independence : It is defined an enhancement over modularity, abstraction and information hiding. Isolation bet" interacting modules is major goal to achieve.
We perform two operations —

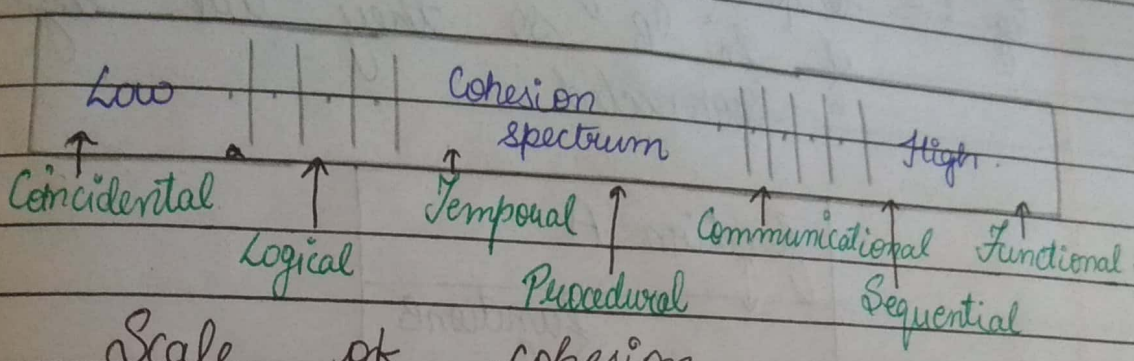
☆ a) Cohesion :
Degree to which modules perform one operation.

☆ b) ~~Cohesion~~ Coupling
Degree to which a module is connected with another module

Scanned by CamScanner

COHESION

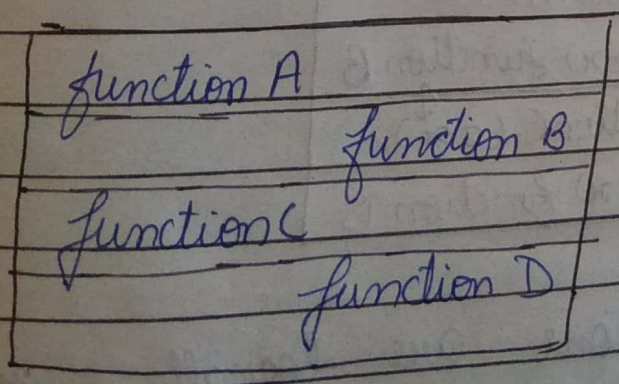
Cohesion is the concept that tries to capture intramodule bond. It means how closely elements of the module are related to each other.



Scale of cohesion.

1. Coincidental Cohesion : ~~Worst~~ Worst degree of cohesion is coincidental.

It happens when there is no meaningful relationship exist among the elements of module So the module tasks are loosely related or not related with each other.

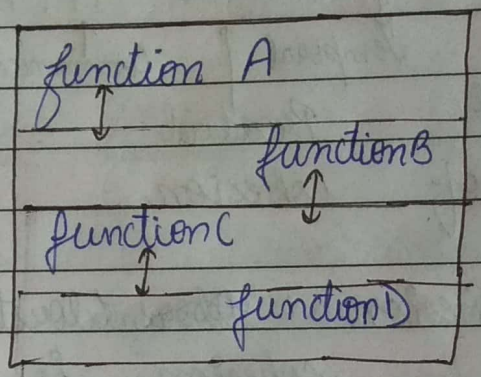


no interaction among function

2 Logical Cohesion

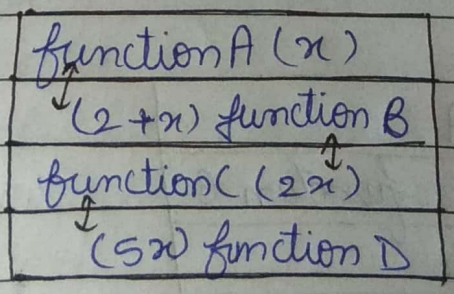
Its a higher level cohesion where functions are logically related with data or sequence

Eg :- Output of fun^n A is served as ~~an~~ input of fn^n B so they are logically connected.



3 Temporal Cohesion

Functions are related to each other with a component of time



All functions are logically related to each other with respect to constraint time x.

4 Procedural

In this, execution of fn follows a sequential order that makes them cohesive with respect to each other.

fun A (first execution)
↑
fun B (sec. execution)
↑
fun C (third execution)

in a sequence

B can't be executed, until A is executed. Similarly C can't be executed, until B. So, they are connected through executional sequence.

5 Communication Cohesion

Communicational cohesion has elements that depends on the data structure where all functions works on similar data access same data and manipulate same data

Data ← Function A
Function B
Function C

Scanned by CamScanner

6 Sequential Cohesion

In this, input and output of the functions are exchanged and they are bound with each other with several change in state of data.

funA
↓
funB
↓
funC

7 Functional Cohesion.

It is the strongest cohesion where functions are interrelated and combinely doing a single task.

funA

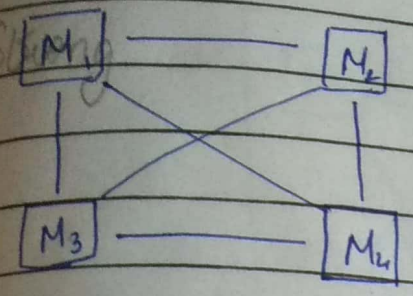
funA$_1$ → funA$_2$ → funA$_3$

COUPLING

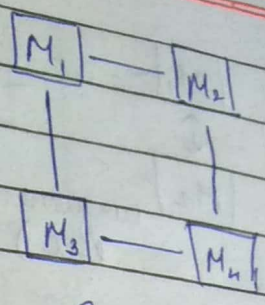
Coupling betn modules is strength of interconnections between them and it measure the level of interdependency. Coupling can be majorly categorised into —

Strong coupling
Loose coupling
No coupling

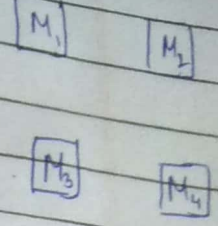
Scanned by CamScanner



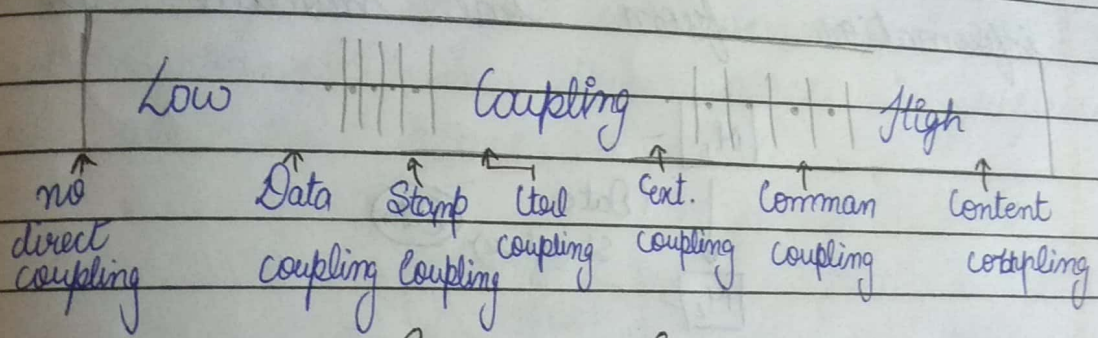
Strong
(High dependency)



Loose
(Moderate dependency)

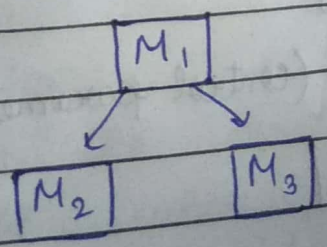


Independent
(No coupling)

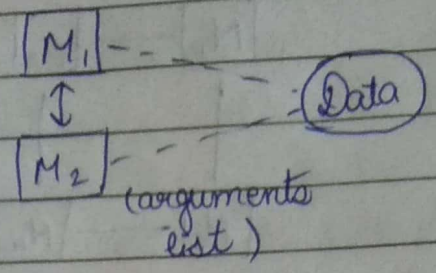


Scale of Coupling.

1 No direct coupling :
In this, there are modules which are not having interaction directly with each other will come under no direct coupling

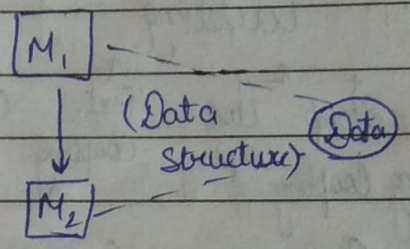


2 Data coupling :
In this, data is passed from one module to another mostly with argument list



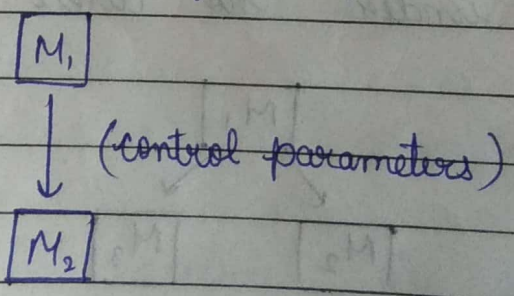
3 Stamp coupling :

Its a variation of data coupling where data structure is used to pass information from one module to another



4. Control coupling :

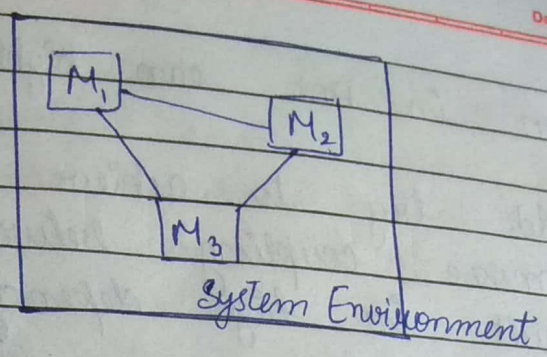
When one module passes parameters to control activity of another module



5 External coupling :

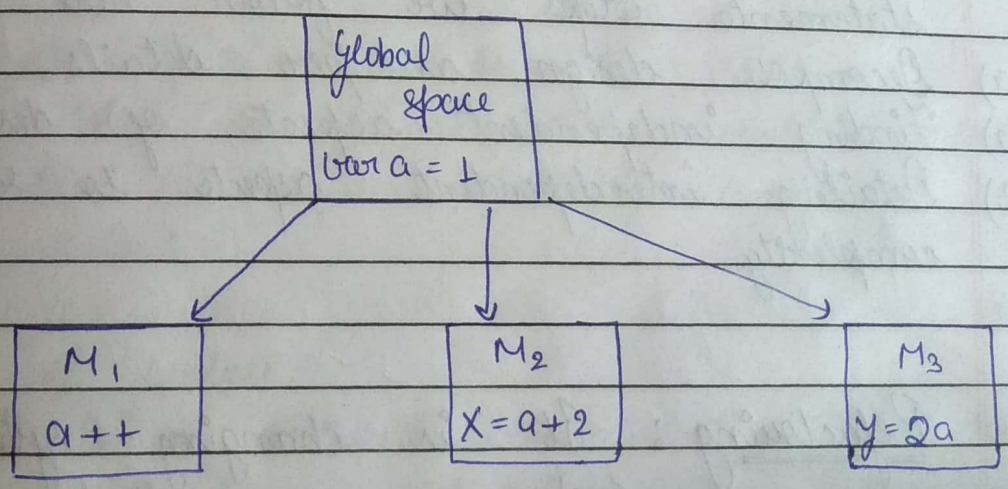
Modules are tied to an external environment and then they are connected with each other.

Scanned by CamScanner



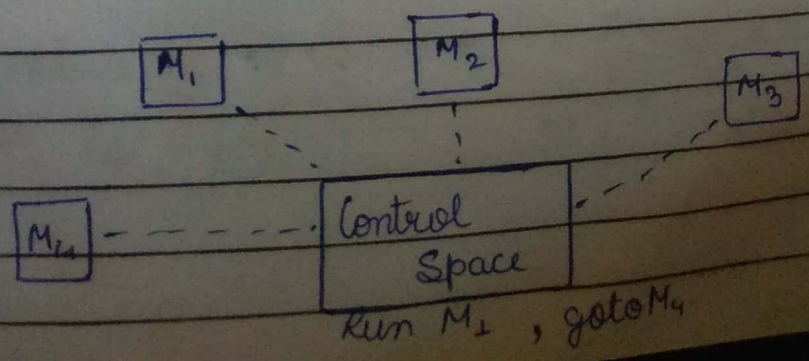
6 Common Coupling:

In this, data is used through a common space and whenever the data is used change in one place is reflected over all the places



7. Content Coupling:

It is the least desirable coupling this is used when control information is maintained within boundary of another module,



modification in one can highly impact another

We should try to achieve low coupling because more coupling between modules can result in high dependency and risk factor

6 Refinment: It is a process of elaboration where several instructions are decomposed into more detailed statements. Steps we follow here are —

a) Decompose design as per details.
b) Finding independent aspects of design
c) Detailing interdependent aspects to resolve complexity.

7 Refactoring: It is changing system in such a way that it does not alter external behaviour of the system.

Scanned by CamScanner

Effective Modular Design

Modularity can increase the complexity of software and makes program structure modification easy. Points to be considered while achieving modularity :-

1. Coupling and cohesion balance

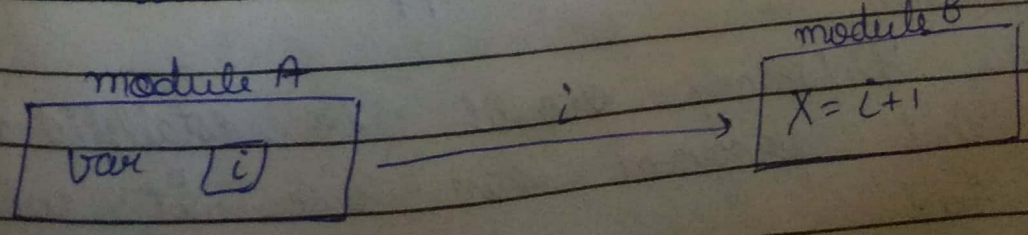
We should try to achieve modular indepency that reduce the coupling of the software and increasing relations in different functio of the system to increase cohesion of the structure.

2. In depth layering of project

We should create reasonable distribution of control as per the dependency of input and output.

3. Scope of control

There should be minimum effect of changing decision made in any module to another module. For eg, there is a control variable i which is in ~~control~~ module A and is used again in module B for finding control of variable x, module B will be fully dependent on module A.



Scanned by CamScanner

4 Connection in module interface
Interface designing should be consistent with module dependency so that information can be easily passed

Parts of Effective Modular Design

1 DATA DESIGN
Data design is first of the activities during designing of software. It creates a top view for the system with highest level of abstraction.
This contains data structures (logical representation of data objects) that are identified during data analysis.

2 Principles of Data Designing.

a) Data objects should be identified and on the basis of that required structure should be applied.
b) All specific operations required must be included in data structures
c) Low level data design decision can take place later when upper level is refined properly.
d) Data dictionary should be estaiblished properly so that program can use it when required.
e) Data libraries should be developed in case of

Scanned by CamScanner

repeated data functions.

STRUCTURE OF DATA DESIGN

Data is designed on three levels.

1. Component level
   At this leve your define data structure and algorithm.
2. Application level
   Here we define databases.
3. Business level
   Here are records and files.

DATA MODELING.

Entity - relationship diagrams and data dictionary is defined during process of data modeling.

⇒ Explain ER diagram and data dictionary.

Data structures.

It is logical representation of data where functionalities are present to access the data in a proper fashion. Data architecture is built with data structures. Multiple data structures are available as per the degree of complexity, performance and effectiveness req^d in the system

Data base.
Data base is collection of similar type of data in an arranged manner so, that access of data is easier

Data Warehouse
Warehouse is an environment to store large independent and arranged data bases. They are arranged as per subject, timeline, type.

23/11/18

2 ARCHITECTURAL DESIGN

software structure

Architectural design is developed to create a which represents the relationship between different modules

ADVANTAGES.

1. User interaction: Architectural design represents highest abstraction level
Hence it is easier for user to understand system. ~~System analysis~~

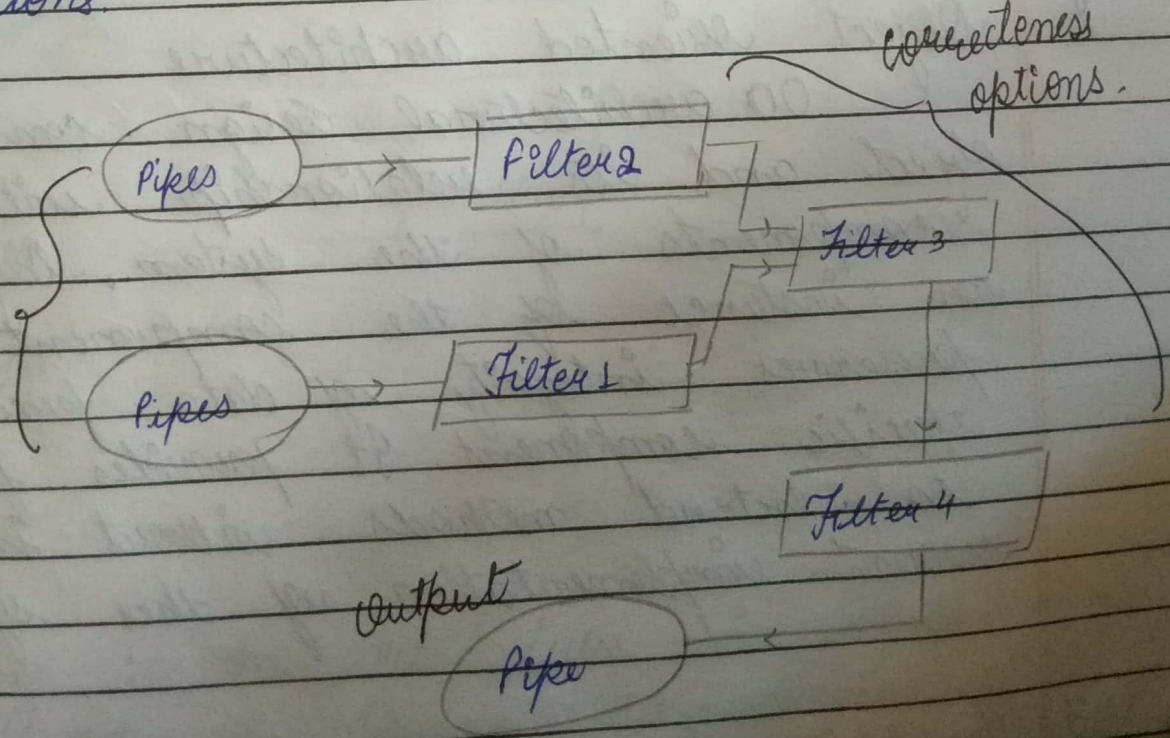
2 System analysis: Architectural design is useful to make decisions about different system components that whether they meet user req" or not. This is also useful to analyse performance reliability and maintainability of software.

Scanned by CamScanner

3. Component reuse : System design represents components and interoperability. Hence modules can be identified which can support similar req^n types in different set of systems

Architectural Styles.

1. Pipes and filters : This system has streams of data called pipes for input and output and a component called filter which defines correctness of the system and works independently from another filters to produce required operations.



Scanned by CamScanner

Properties

a) Effect on system can be analysed with the amount of input and output with composition of filters.

b) New filters can be added easily as filters are independent of each other

c) Filter reuse is easy whenever a similar system appears.

Disadvantage :- Highly dependent on data

2. Object oriented architecture

OO architectural design consists of object and its relationship with different components of the system. Object is an instance of the component that preserves integrity of data present in specific component. It provides functionality to extend methods without impacting current implementation of the system.

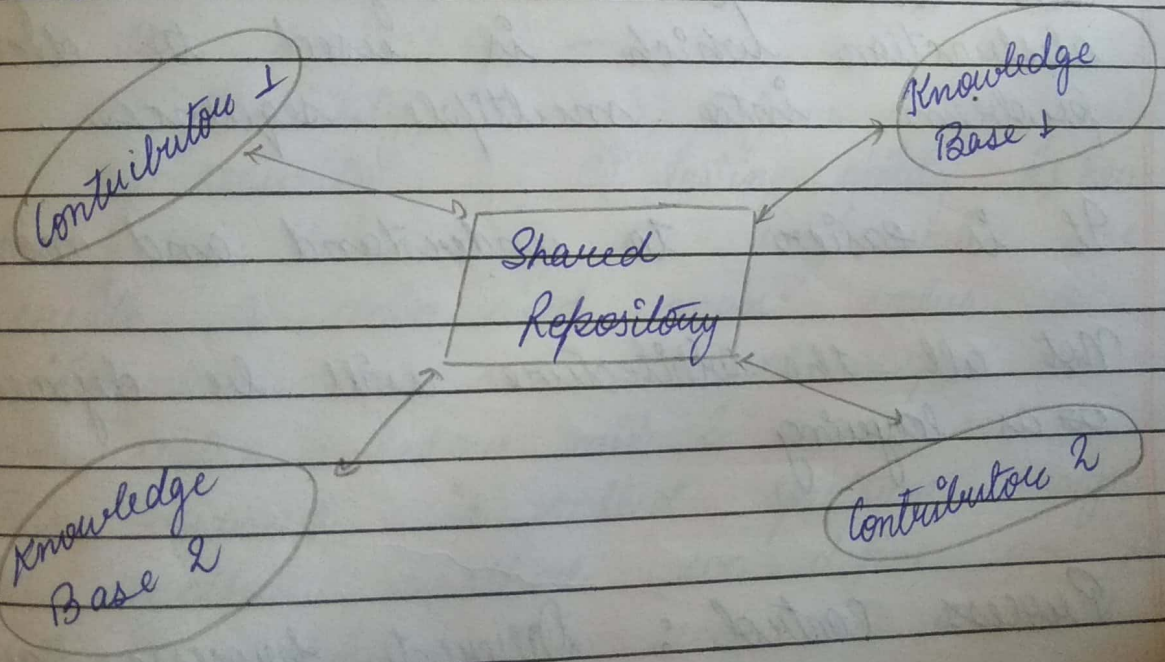
3 Implicit invokation architecture

This design model is event driven which defines which method to call

Scanned by CamScanner

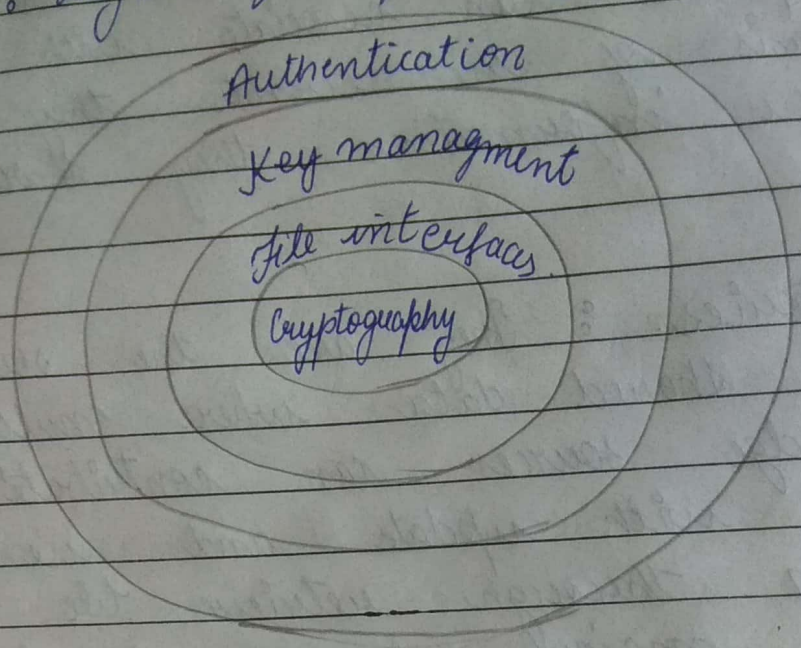
when one or more events takes place. A component can associate with another on basis of the events they invoke and the information they share.

4. Repositories : They are the source of shared data where multiple knowledge sources can contribute to the system. with update and use the system through retrieve. We use this system majorly because it is open and it could be provided to multiple users.



5. Layering : An architecture is made from no. of layers to create a hierarchy between different systems which provides security inside out to the ~~dis~~ client and the system

Eg : Layers for providing Security.



Advantages.

a) Each level provides higher level of abstraction which is used to decompose problem into multiple sequences.

b) It is easier to understand and modify

c) Not all the architecture will be defined ~~to~~ as layering

3. Process control : Different processes are created for multiple components to provide specific output to the system We define different values called set points from where ctrl will be transfered from one system to another.

Scanned by CamScanner

24/11/18

Date: / / Page no:

ANALYSING ARCHITECTURAL DESIGN

1 Trade off Method

This is an iterative evaluation process to analyse the design whether it fits scenarios constraints, environment.
Various quality attributes are considered to analyse design as per users perspective.
Alternatives are discussed & on basis of elimination and modification final design is defined which later used for the system.

2 Quantitative Method.

In this method a criteria is created on the basis of —

a) flexibility b) performance c) reliability
d) security e) testing ability f) reusability

On the basis of the req^d value we create a min and max value for these constraints and we do not expect to use the value below min & above max.
This method is called Spectrum analysis.
In another method, we define an ideal system which will serve as a basis for achieving system design. This method is called design analysis.

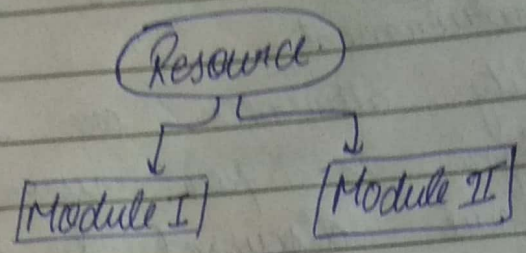
3 Complexity Analysis Method

This method is used to evaluate interdependencies of the system. Major divided

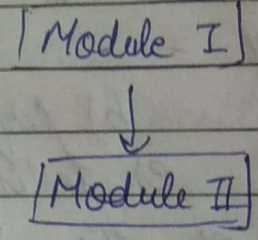
Scanned by CamScanner

into :-

a) Sharing dependencies, where 2 modules use the same module



b) Flow dependency : When one module is executed only after execution of another module as they are in a particular sequence



c) Constraint dependency : When system provides limitations for the set of activities in a module, constraint dependencies appears in system.

Eg : u, v are two OS, but either of one can run at a time because RAM can be used by only one at a time.

Scanned by CamScanner

PROCESS (PROCEDURAL) DESIGN

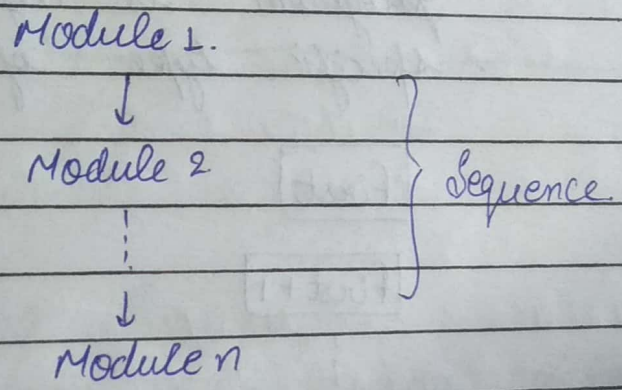
In this we define the process in detail with the help of multiple graphical language for use to understand the process clearly.

Process Designing Methods.

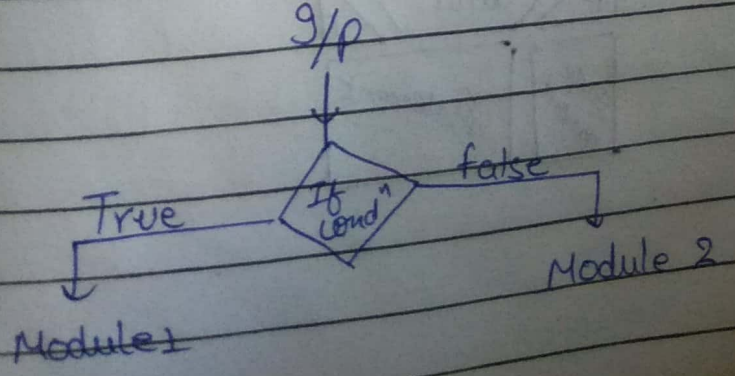
1. Structured Programming.

A simple logical constructs are used to define the program. It majorly use – ~~sequence~~

a) Sequence : When 1 thing happens after another.

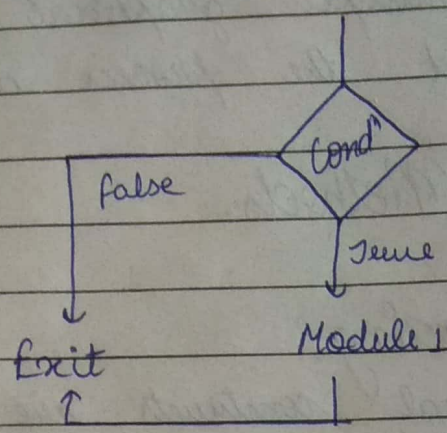


b) Condition : If it is true one part is executed & if it's false another part is executed

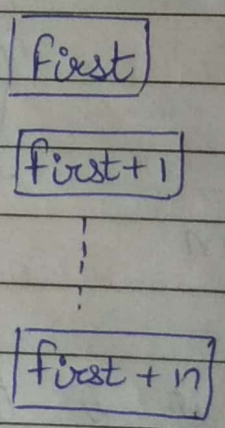


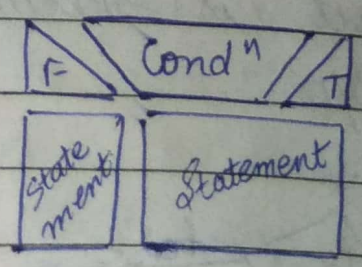
Scanned by CamScanner

c) Repitatation : Untill cond^n is false, a particular module is executed.



d) Box diagrams : These are another graphical notations where each part of program is defined with a specific type of box.





Scanned by CamScanner

2 Tabular Notations.

It is used to design a decision table which consist of combinations of conditions possible actions, when certain rule is applied

	1	2	3	4	5	---	n.
conditions.							
Actions.							

Rules.

3 Programming Language.

Different programme design languages are used which is mostly in user understandable language or pseudo code to define the design with index, operation, nesting and data structures.

Scanned by CamScanner

23/11/18

Date: / / Page no:

★) Architectural design → is develop to create a software structure that represents the control relationship b/w diff modules:

The adv. of Arci. design :

1) User interaction : → A.d. represents highest level of alestraction here it is easier for the user to understand the system.

2) System analysis ⇒ The arc. design is useful to make decesions about diff. system components that whether they meet user requirement or not.
This is also useful to analyse performance reliability and maintainability of software.

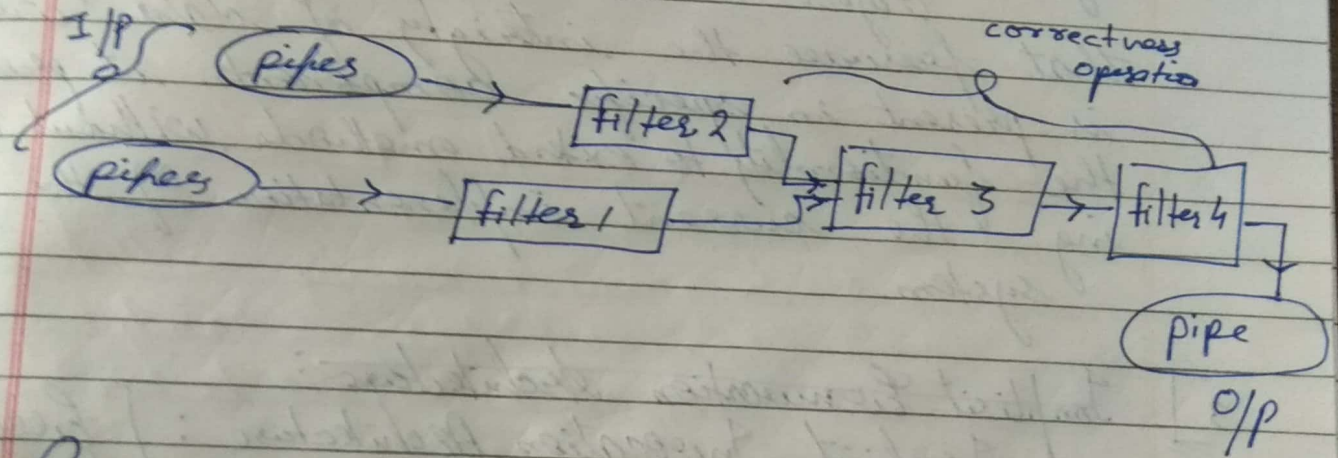
3] Component Reuse : System design represents components and their interoperability hence modules can be identified which can support similar type of requirements in diff.ⁿ set of systems.

Scanned by CamScanner

23/11/18

*) Architectural Styles :

1] Pipes and filters : It / This system has a stream of data called pipes for input and output and a component called filter which defines correctness of system and works independently from another filters to produce required operation



*) Properties of pipes and filter system:

1) Effect on the system can be analyse with the amount an input and output with the composition and filter.

2) New filter can be added easily, as filter are independent of each other.

3) Reuse of filter is easy, whenever a similar system appears.

Scanned by CamScanner

23/11/8

⇒ Diasadv: → It is highly dependent on data

2] Object Oriented Architecture:

The OOA design consist of object and its relationship with diff" components of the system.

Object is an instance of a component that presurve the intrigity of data which is present in specific components. it provide the functionality to extend methods without impacting the current implementation of the system.

3] ~~Implicit Invocation Architecture~~:

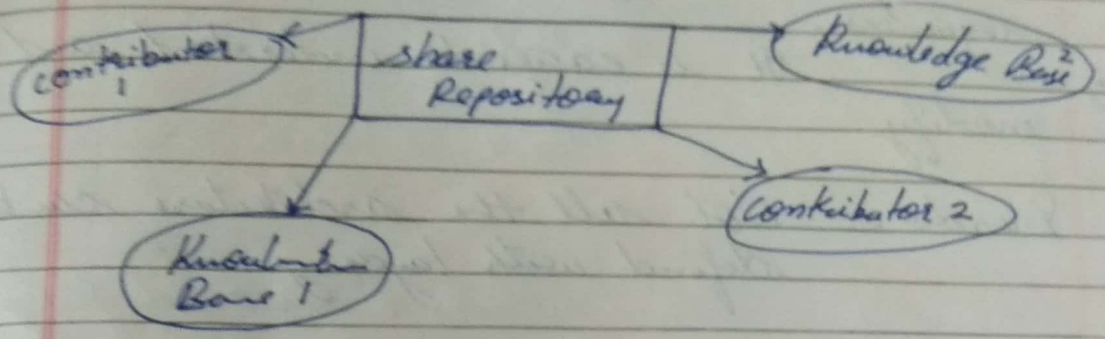
Implicit Invocation Architecture: [Event Driven]

This design model is event driven, which defines which method to call when one or more events takes place. A components can associate with another on the basis of the events they invoke and the info. they share.

4] Repositories: These are the source of share data where multiple knowledge sources can contribute to the system with update and use the system

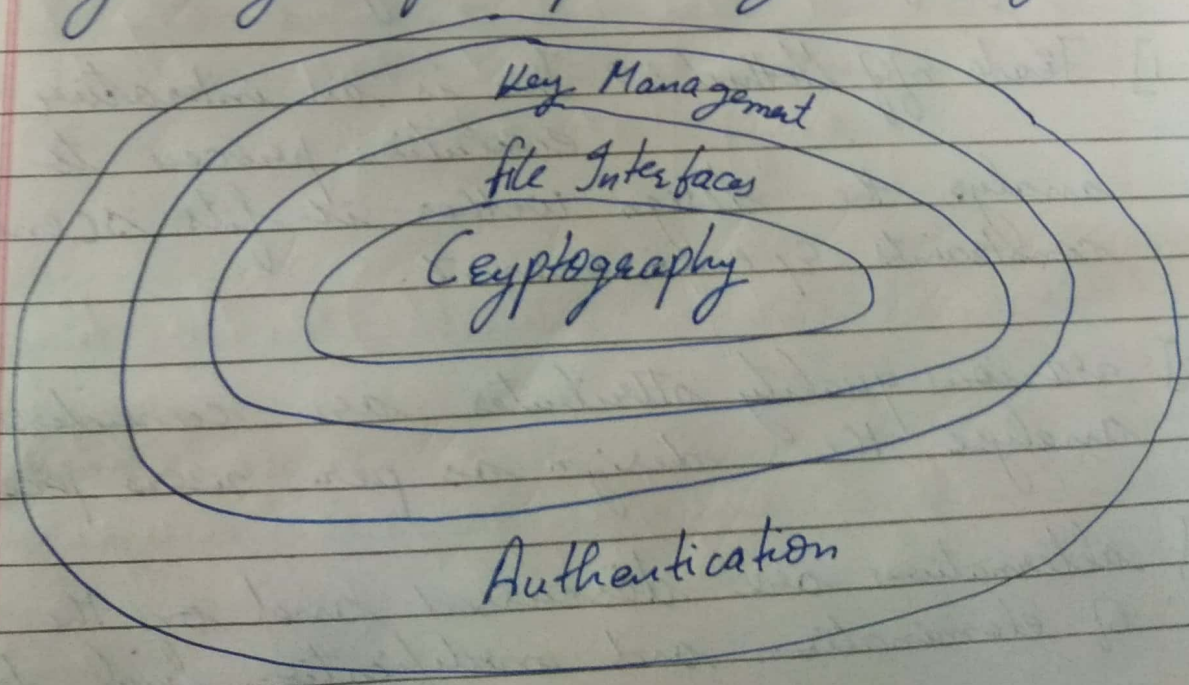
23/11/18

through retrieve. We use this majorly because it is open and it could be provided to multiple users.



5] Layering : An architecture is made from number of layer to create a hierarchy b/w diff. system which provide security inside out to the client and the system.

Eg: layers for providing security:



Scanned by CamScanner

23/11/18

Advantages: ·) Each layer provide higher lvl of abstraction which is use to decompose the problem into multiple sequences.

·) It is easier to understand and modify.

Disadv: Not all the architectures can be defind with layering.

6] Process Control: Different processes are created for multiple components to provide specific output to system. We define diff^n values called setpoints from where the control will be transferred from one system to another.

Analysing architectural Design ⇒

1] Trade off Method :⇒ This is an interative evolution process to analys the design wether it fits scenerio's constraints & environment.

2] various quality attributes are considerd to analyse the design^n as per users prespective

3] alternatives are discussed and on the basis of elemination and modification final design

24/11/18

is defined which later used for the system.

2*] Quantitative Methods ⇒ In this method a criteria is created on the basis of

i) flexibility
ii) performance
iii) reliability
iv) security
v) Testing ability
vi) Reusability.

On the basis of the required value we create a minimum and max. value for these constraints and we do not expect to use the value below min. and above maxi. this method is called spectrum analysis.

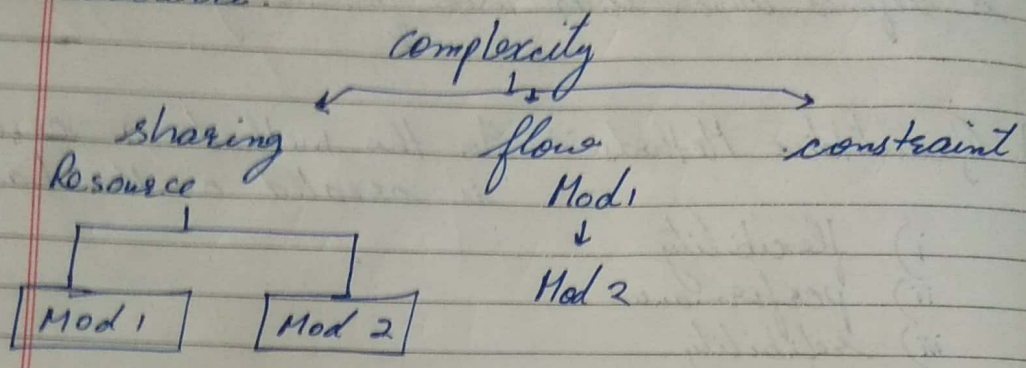
*) In an other method we define an ideal system which will serve as a basis for achieving the system design this method is called design analysis method.

3*] Complexity Analysis Method : This method is used to evaluate inter dependencies of the system major by divided into 3 parts :

i) Sharing dependencies : where 2 module use the same

Date: 24/11/18 Page no.

resource.



*) flow def. → When 1 module is executed only after the another module is executed as they are in a particular execution.

*) constraint def → when a system provides limitation for set of activities in a module constraint depend. appear in the system.

Eg: V, U are modue which req" RAM at same time but RAM could be provided to only 1 at a time.

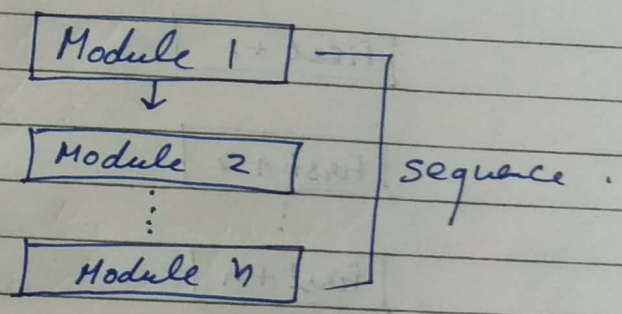
A) Process Design (Procedural) ⇒ In this we define the process in detail with the help of multiple graphical language for user to understand the process clearly.

Date: 24/11/18

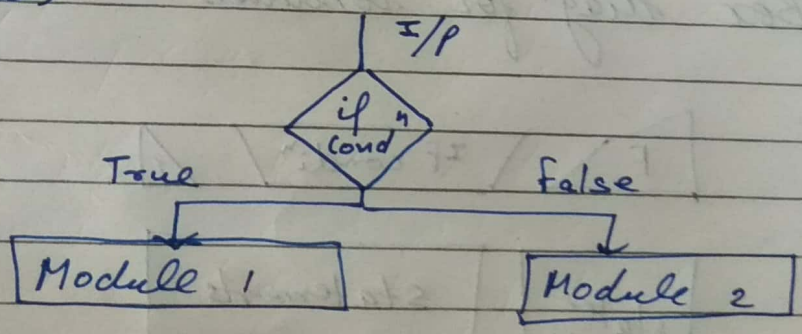
Process Designing Methods:

i) Structured Programming: A simple logical constructs are use to define the program. It majorly use to define the program. It majorly used sequence:

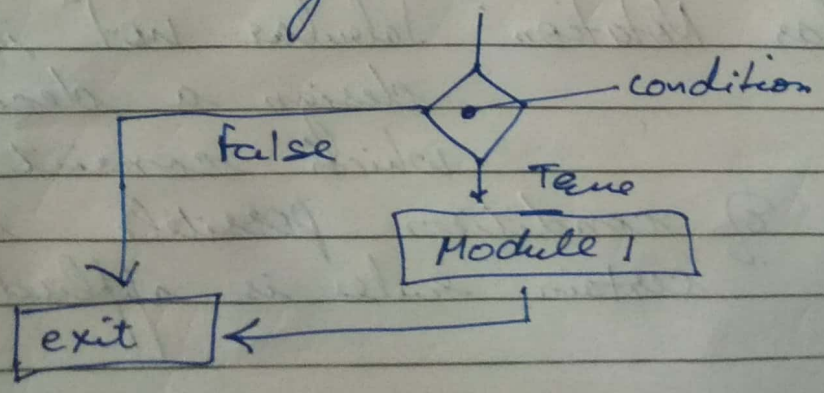
Sequence :-



ii) Condition →



iii) Repetation: iteration of step again and again till result is obtained.

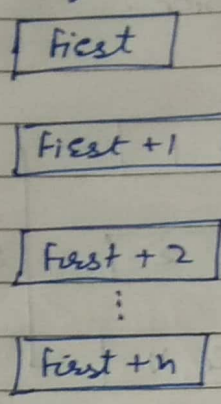


Scanned by CamScanner

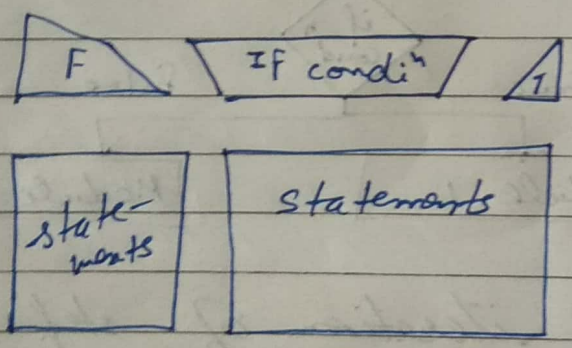
24/11/13

iv) Box diagrams : are another graphical/notation method where each part of the program is defined with a specific type of box.

a) Box dia. for sequence :



b) Box diagⁿ for condition :-



★ Tabular Notation : Tabular notⁿ is used to design a decision table which consist of combination of condition possible action when certain rules is applied.

24/11/18

Date: / / Page no:

Decision Table

		1	2	.....	n
condition					
action					

*3] Programming language : different program design languages are used which is mostly in user understable language or pseudo code to define the design with index operation nesting and data structures.

Design documentation:

Scanned by CamScanner

Unit -5

Date: / / Page no:

UML Diagram

UML → Unified Modeling Language

Use Case Diagram

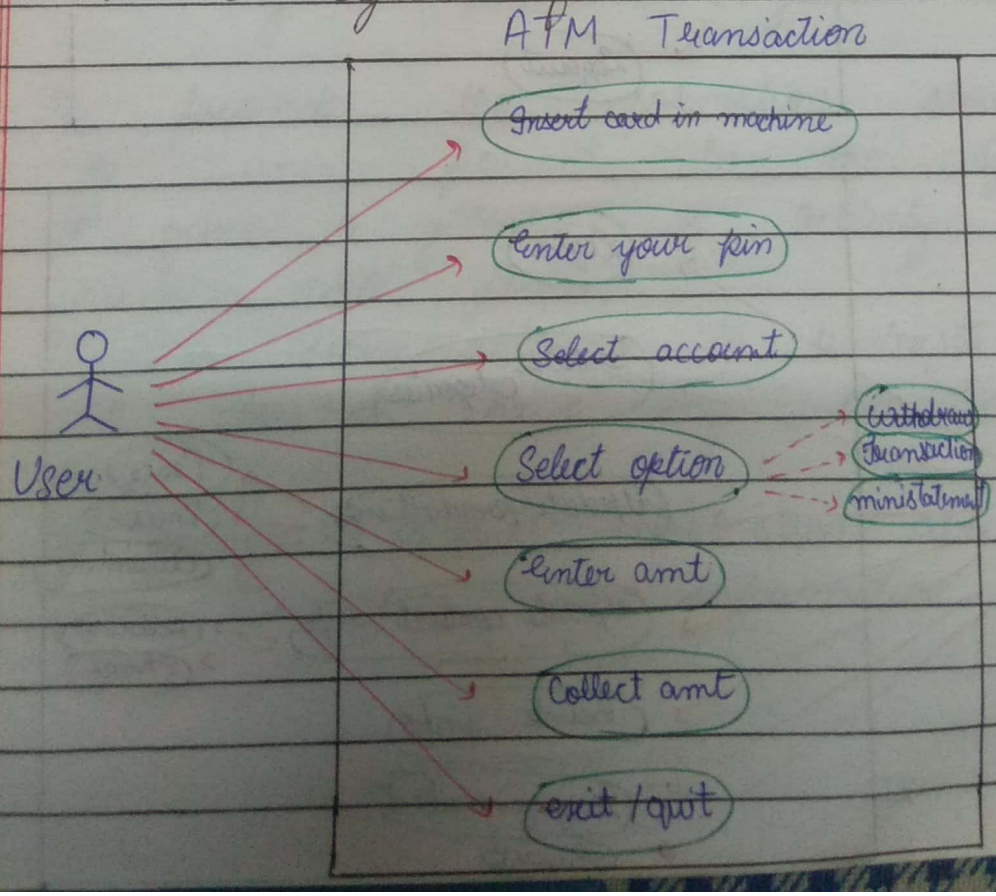
→ User / Actor

→ Communication

→ Environment

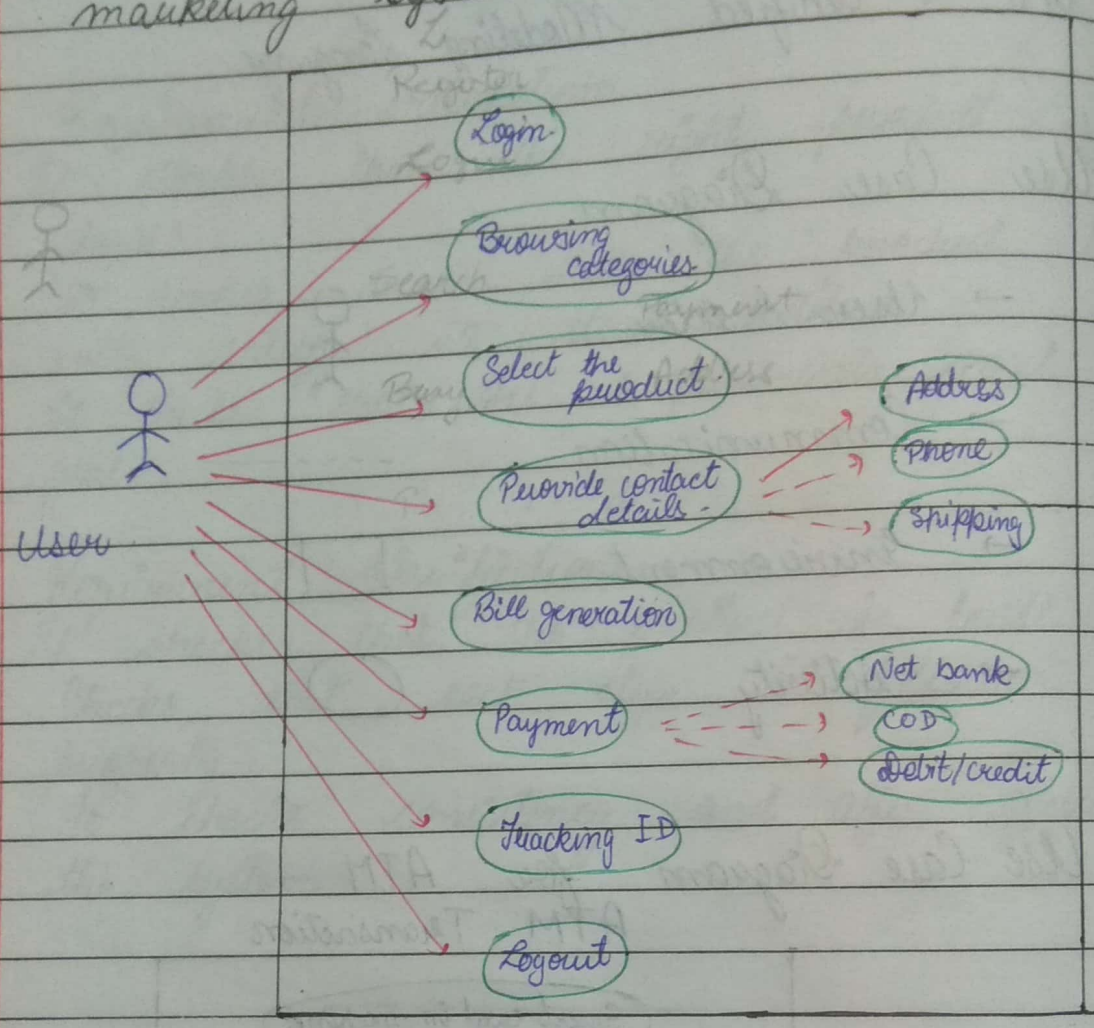
→ Activity

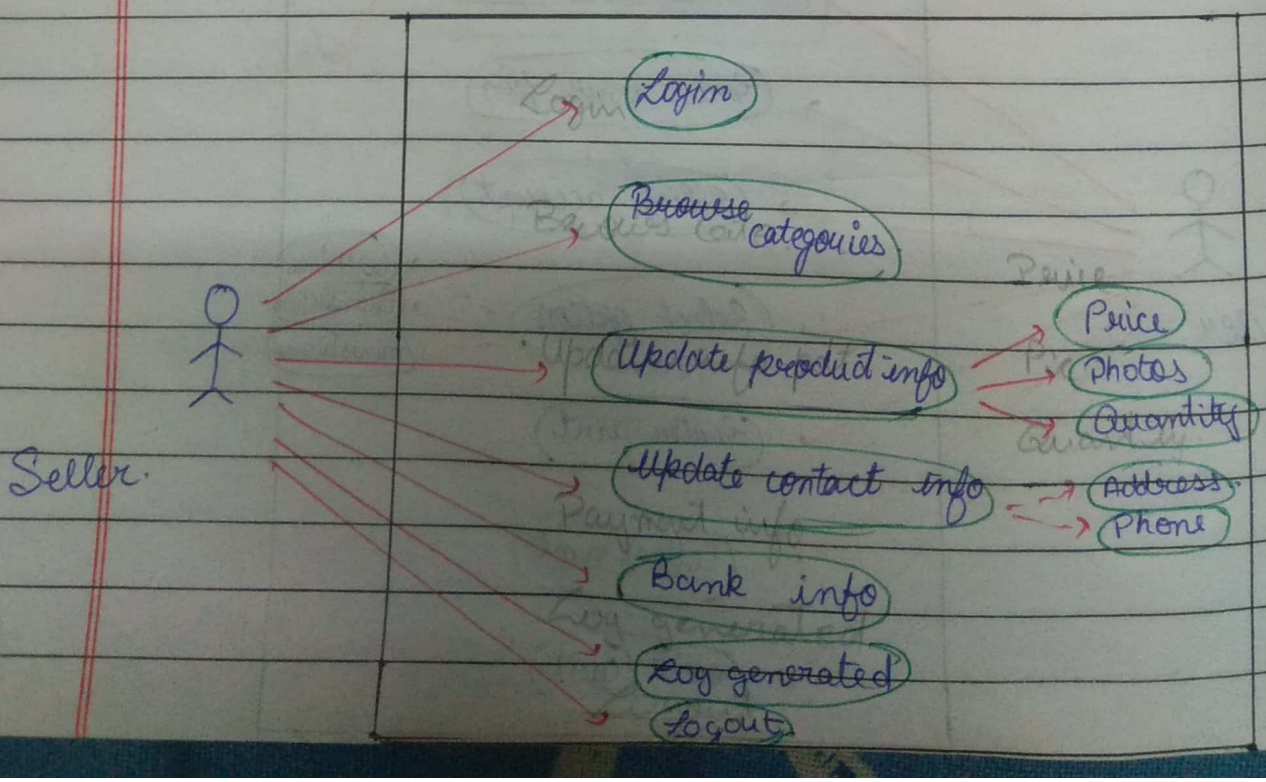
Use Case Diagram for ATM.



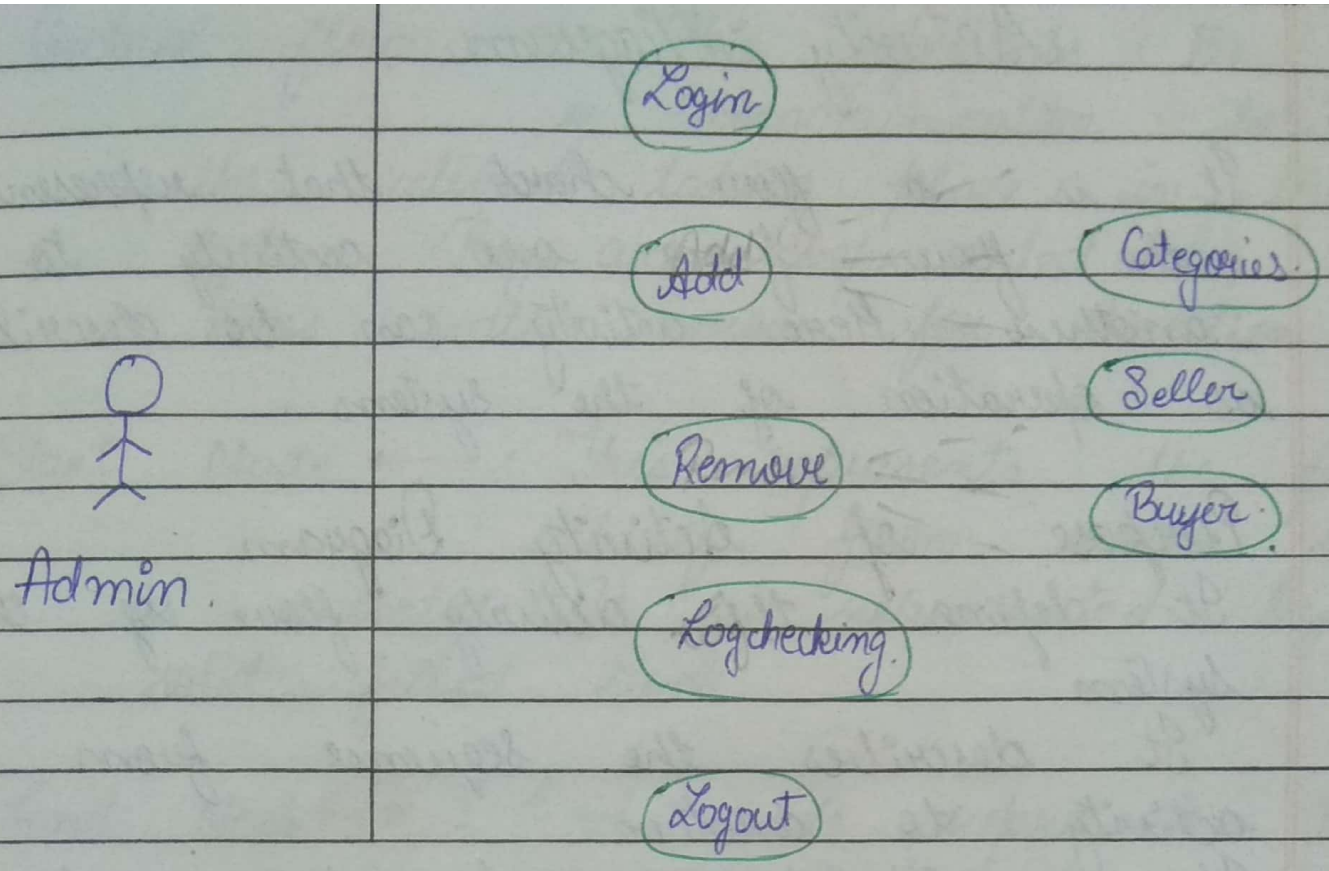
Scanned by CamScanner

Ques. Create a use case diagram for online marketing system.





Scanned by CamScanner



Use of Use Case Diagram.

It provides the information about set of users present in the system
It gives a glimpse of activity that the user will perform.
It is very simple to understand because it dosnot have any condition or loop
It provides pictorial information about the user interface and information available inside it

Scanned by CamScanner

Activity Diagram

It is a flow chart that represents the flow from one activity to another. Here activity can be described as operation of the systems.

Purpose of Activity Diagram

1. It defines the activity flow of the system
2. It describes the sequence from one activity to another.
3. It models parallel branched and concurrent flow of the system.
4. It describes the logic in the system.
5. It clarifies the complex architecture of the system.

Components of activity diagram

1. Activity : It represents the action performed in the system. It is represented by a rectangle with circular edges.

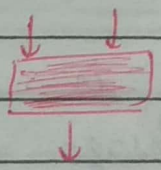
2. Decision node : This represents the decision or the condition that appears in the system. This mostly deals with singular input and two or more output branches.

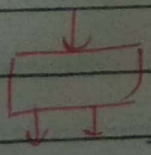
Scanned by CamScanner

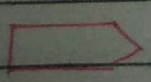
3 Control flow ⟶ : This represents the flow of communication between the actions taking place in the system. They are represented by solid lines with arrow for direction.

4. Start Node •⟶ : This represents the first activity from which system is intiated. It is represented by solid filled circle.

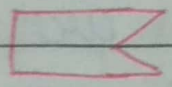
5. End Node ⊙ : End node represents the last activity that will be performed in the system. It is represented by a circle encapsulated by filled circle.

6 Join / Syncronization  : When two or more activities needed to be combined because they leads to similar activity we use join to connect both of them

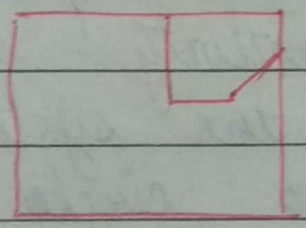
7 Fork  : When the activity has to be splitted in to two or more consecutive entities we use fork to split them

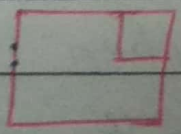
8 Sent message  : When a type of signal has to be sent from

One activity to another, we use sent message

Recieve message  : When a signal is accepted by an activity it will be shown with recieving message.

History pseudo code (H) : It represents history of last invoked transition

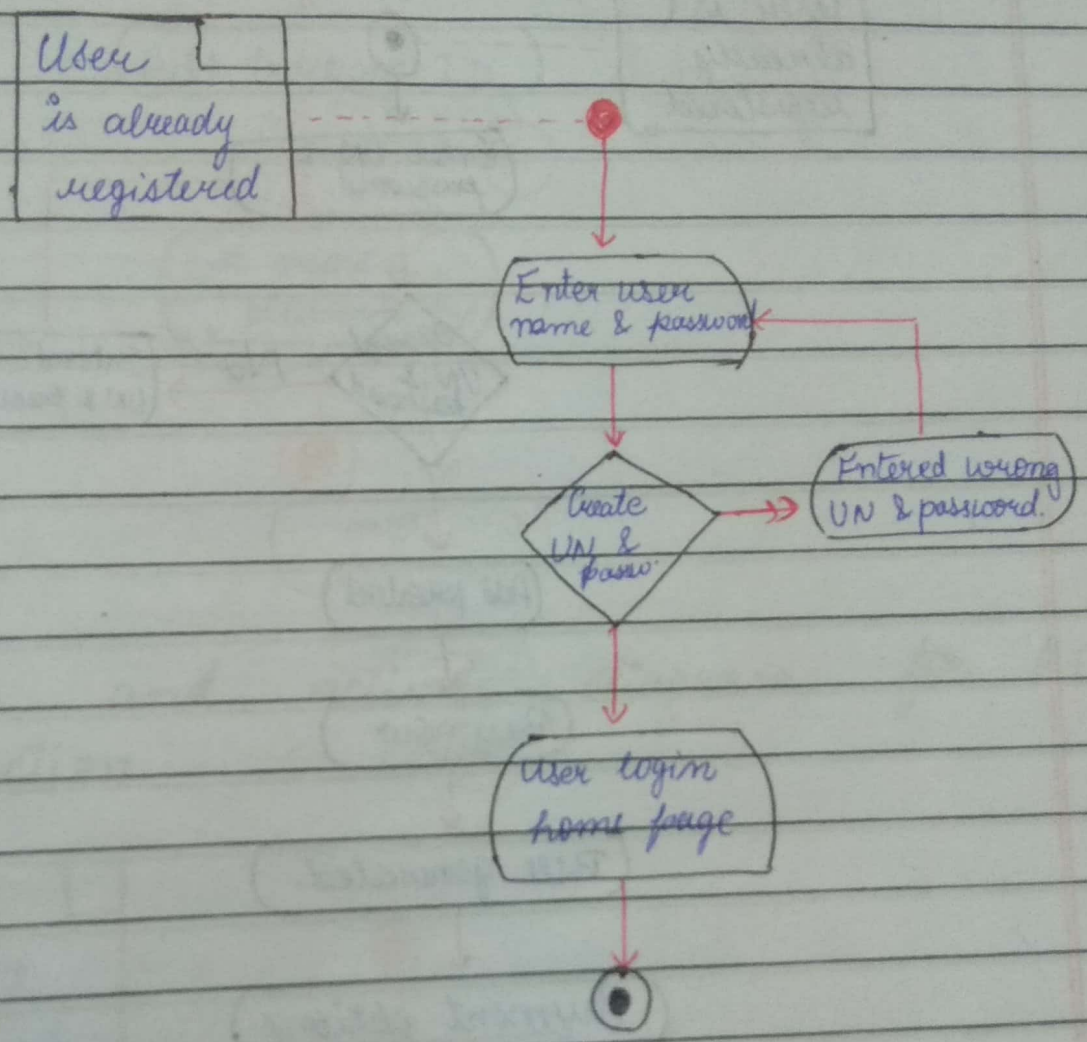
Loop  : Loop symbol is used when a single activity has to be initiated and iterated multiple times.

Note  : It is used to define additional message that will be required in the system.

Scanned by CamScanner

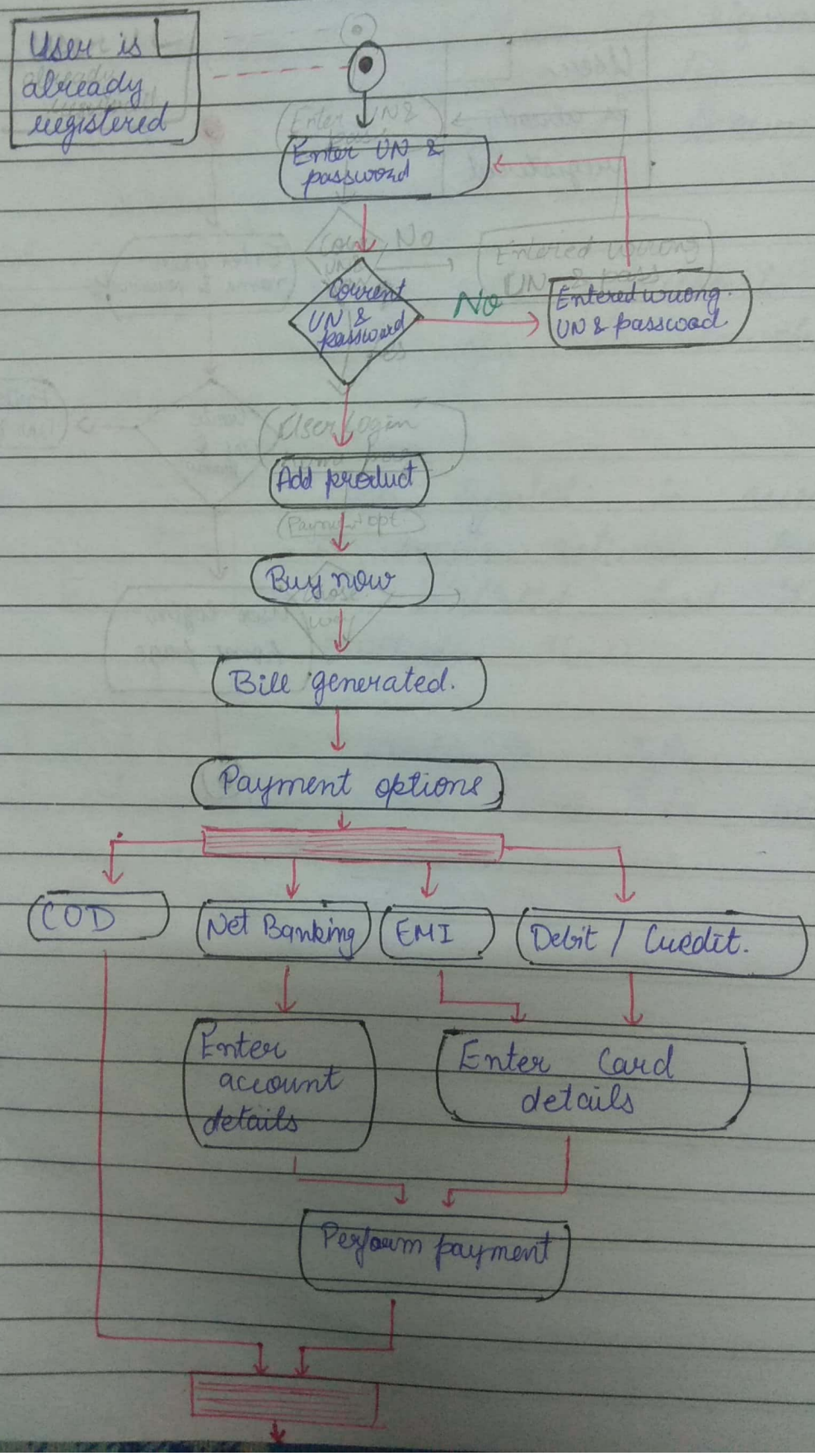
Date: __/__/____ Page no: _____

Ques. Draw an activity diagram for login activity of a System.

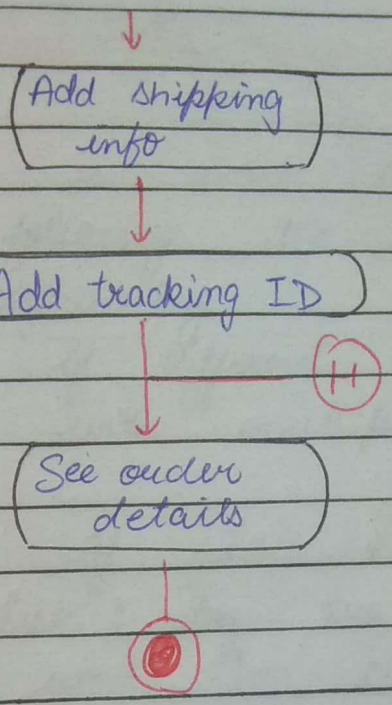


Scanned by CamScanner

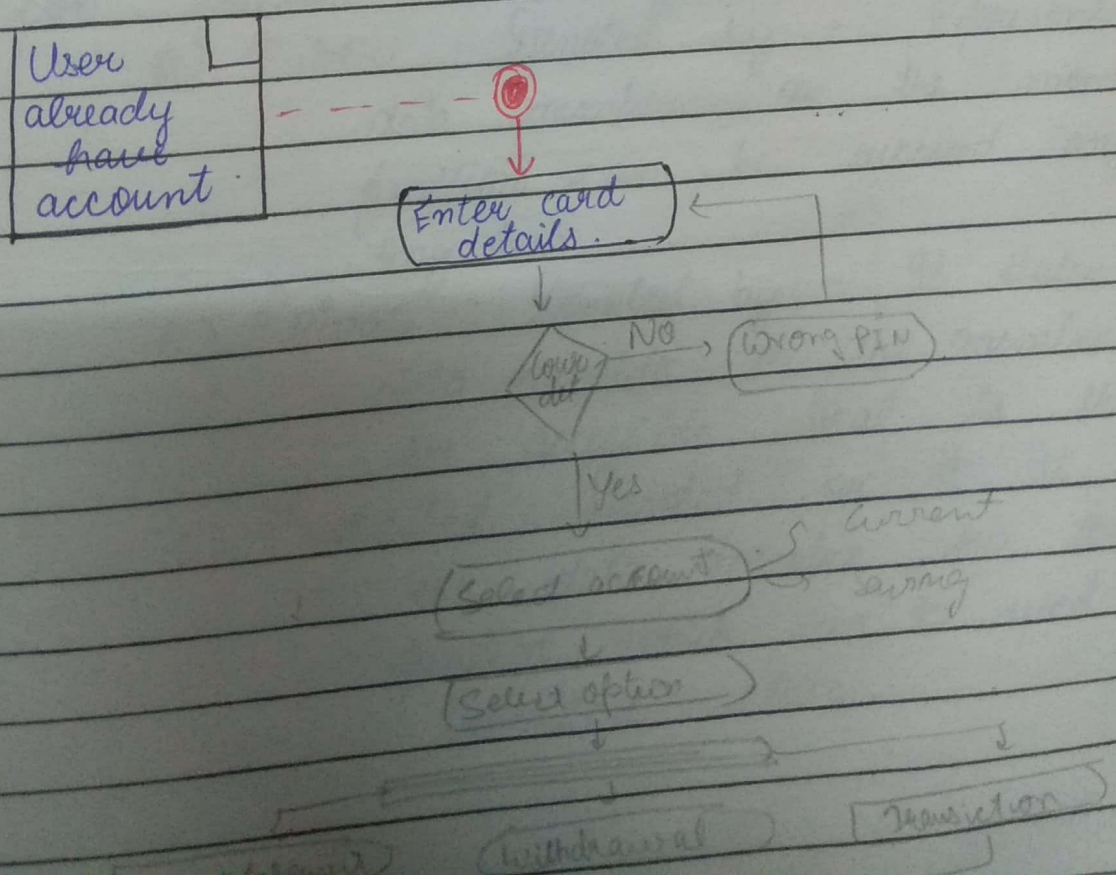
Ques. Create an activity diagram for online payment system.



Scanned by CamScanner



Ques. Create and activity diagram for ATM Transaction



Scanned by CamScanner

Class Diagram

Class digram is used to define the software which is following the concept of object oriented programing
It consists of different classes connected with each other with multiple constraints

Components of Class Diagram

1 Data member : They are variables that are used in class diagram

2 Member function : These are the operations that will be performed by the class.
Member functions and data members can contain three type in their access :

a) Public : Denoted by +. Represents that data member or the member function can be accessed anywhere by any class

b) Private : Denoted by -. It states that data member or the member function are accessible local to the class only

c) Protected : Denoted by #. It represents the data member or the member function which can be used only in inherited class.

Scanned by CamScanner

Date: __/__/__ Page no:

Structure of Class

Class name
Data members
Member function

Create a class for student registration

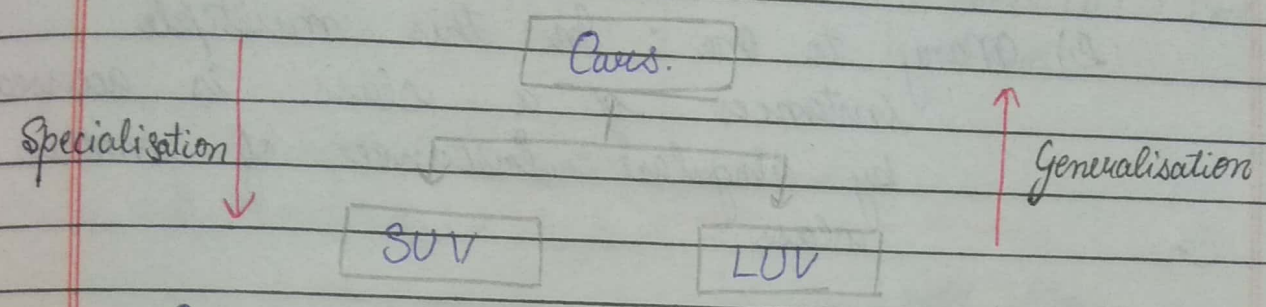
Enrollment
+ email : char [30] + name : char [30] - password : char [12] + sem : int \# course : char [5]
+ enter details ( ) : int + display details (char [ ], int)

Scanned by CamScanner

Date: __/__/__ Page no: ____

Constraints of a Class

1 Inheritance : It helps a class to use the property of the existing class and we preform this with the help of specialisation and generalization.



Generalisation is a process of abstraction where we do not consider differences and aggregat similar functionalities into a singlur entity or class

Specialisation is performed where entity is having multiple branches which ~~alternatively~~ represents a different set of attributes all-together

When we perform speciallisation we creat heirarcy In this the classes at the lower level will contain all properties of classes at higher level

2 Maping Cardinality : A class can be connected to another class in such a manner

a) One to one : When a class can access or be connected to only singular instance of another class

Eg : Student — Course

b) One to many : In this a class can be related to another class with its multiple expenses

eg: University 1 —— N Courses

c) Many to One : In this multiple instances of a class is accessed by singular instance of another class

eg : name ← 1 Students

d) Many to many : In this relationship many instances of other class is connected to many instances of one class

eg :

Author 1 —— N Books

Ques Create a class for online billing system which will contain different classes for every operation

Tracking.

Track ID : char
~~Courier name~~ :
Courier ID :
Bill date :
Ship date :
Deliver date :
Track order
Cancel order.

Return order
Exchange order

Date: / / Page no:

Costumer

+ email : char [30]
+ name : char [30]
- passcoord : char [12]
- phone no : int
- address : char [50]

+ Enterdetails ( ) : int
# Storedetails ( )

Login

- User ID : char [15]

+ Enterdetailog ( ) : int
- Generate ID ( ) : int
+ Redirect ( )

Add item

- item : int [30]
- itemdet : char [50]
- itemcost : int [30]

+ additem ( )
+ delitem ( )
+ searchitem (

Bill Generation

cost : int
item : int [30]

- Generatetotal ( )
Addtaxes ( )

Payment

amt : int
Payopt : ch

Applycoupon ( )
Selectopt ( )

Pay from Card

Card no. : char
holdername
CVV
expiredate

generateOTP ( )
Make payment ( )
Changestates ( )

COD

- Address

Change Status ( )

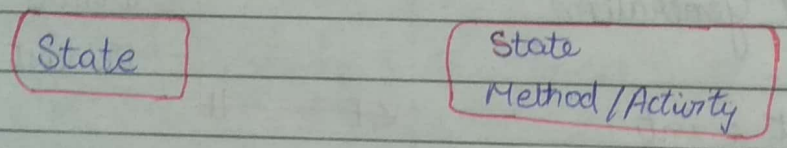
Scanned by CamScanner

State Chart Diagram

It shows behaviour of the classes and object in response to series of events in the system. This UML models the dynamic control flow from one state to another of an object in the system

Components of State Chart Diagram

1 State : It represents different situations during life of an object. Represented by rectangle with circular edges. It can and can't include methods

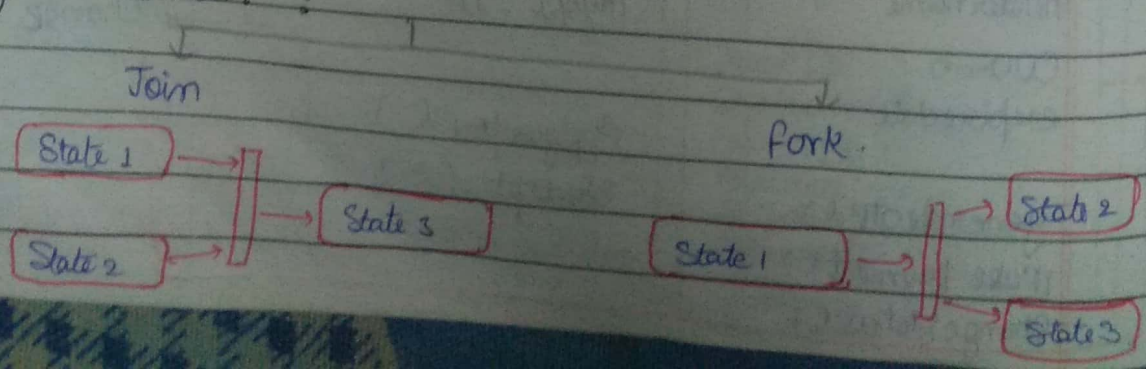


2 Transation : ⟶ It denotes the change in the activity and defines a path.

3 Initial state : ●⟶

4 Final State : ◉

5 Syncronisation :

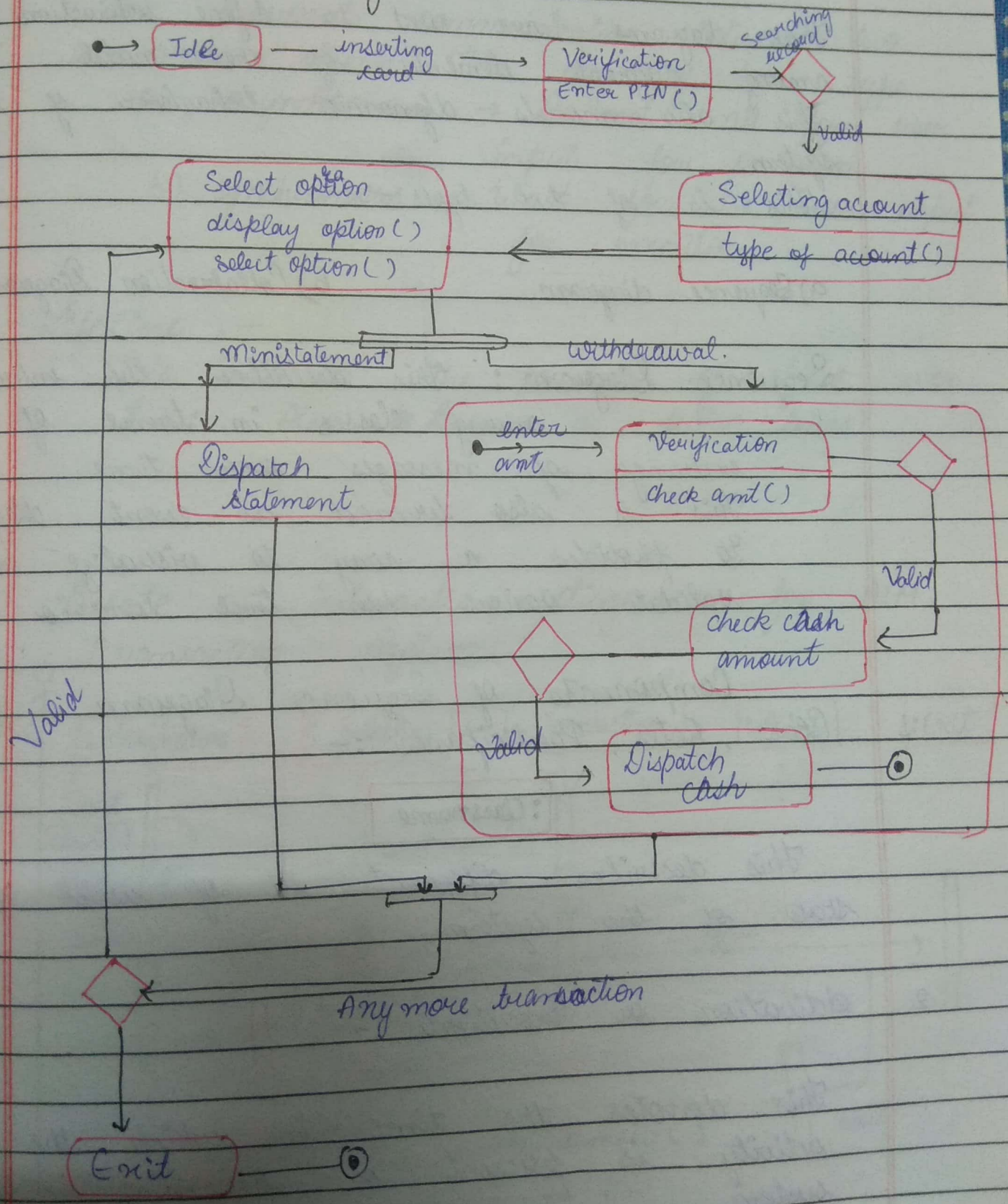


Ques

Scanned by CamScanner

Date: __/__/__ Page no: ____

Ques Draw a state chart diagram for ATM Transaction System



Scanned by CamScanner

Interaction Diagram:

These diagrams are used to define interaction among different elements of the model. This also models dynamic behaviour of the system.

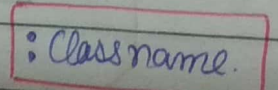
This is of two types —

a) Sequence diagram b) Collaboration Diagram

Sequence Diagram : This describes the interaction among classes in terms of exchange of messages over time. This is also termed as event diagram. It provides a way to visualize and validate various run time scenerio.

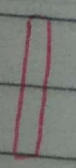
Components of Sequence Diagram

1 Class, Roles, Participants :—



This describes different set of users or state of the system.

2 Activation or Execution :—



This denotes the time in which the activity is present or happening in the system.

Scanned by CamScanner

3 Messages :-

message

This represents communication between two or more objects. This are of two type

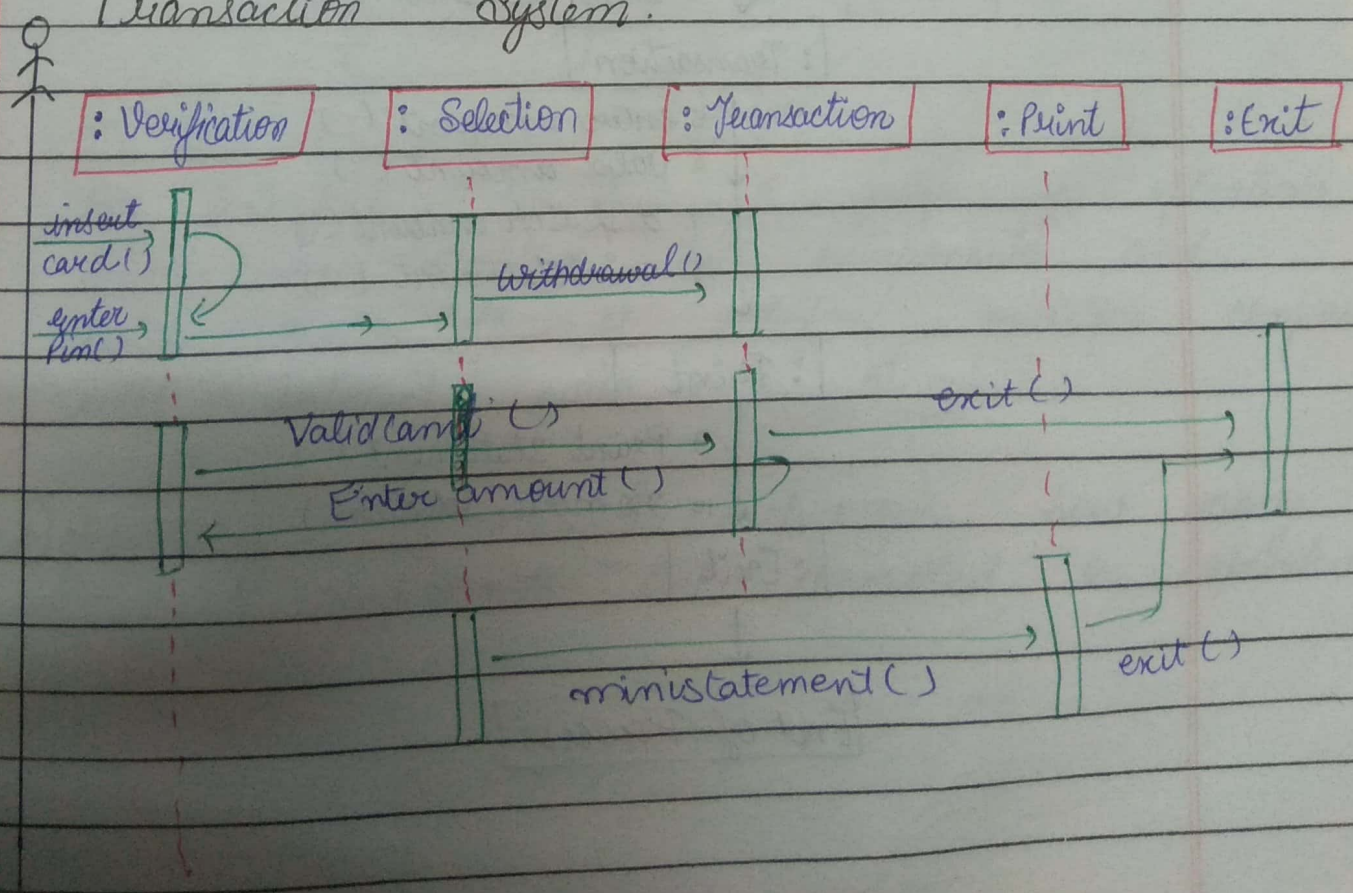
a) asynchronous → Do not wait for user input for execution.

b) synchronous → Wait for the user input for execution

4 Lifeline :-

This indicates the presence of object over time. Also known as swim lines.

Ques Create a interaction diagram for ATM Transaction System.

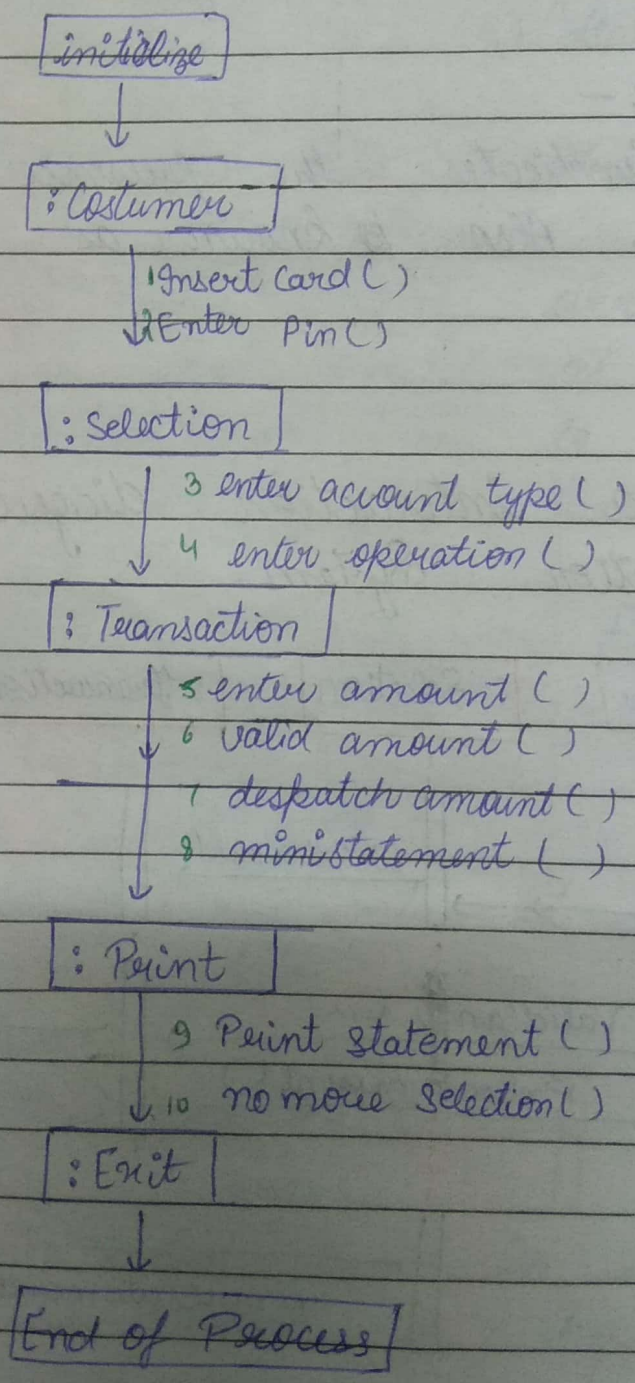


Scanned by CamScanner

Collaboration Diagram :

Here the sequence diagram is indicated by numbering the methods.

The number indicates how many methods are called and in which sequence they are called.



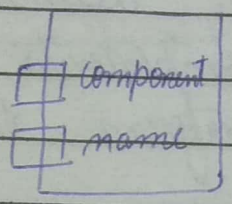
Scanned by CamScanner

Component Diagram

It describes the organisation and connection of physical and software components in the system

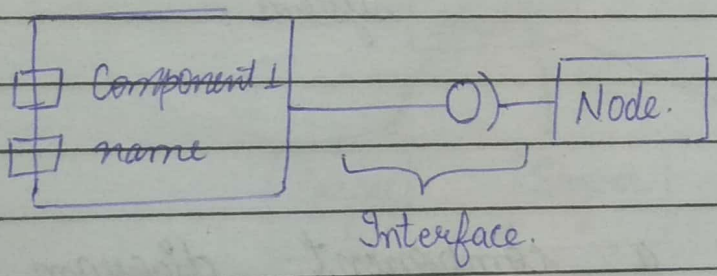
Components of this diagram

1. Component :



It is a logical unit block of a system and having abstraction slightly higher than abstraction of the classes

2. Interface :

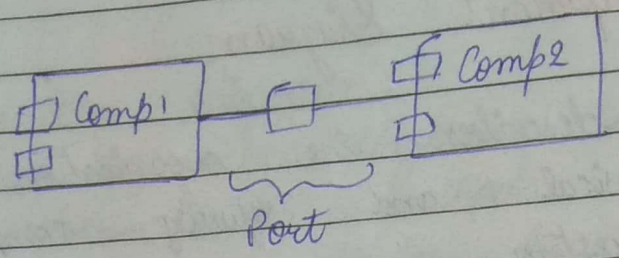


An interface describes a group of operations used or created by components and with the help of interface, multiple component will interact with each other

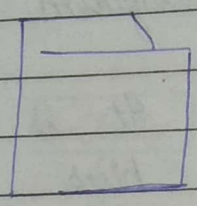
3. Dependency : ---> It represents how many components are connected or dependent on each other.

Scanned by CamScanner

4 Port :

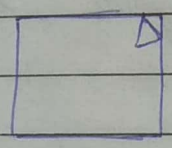


5 Package :



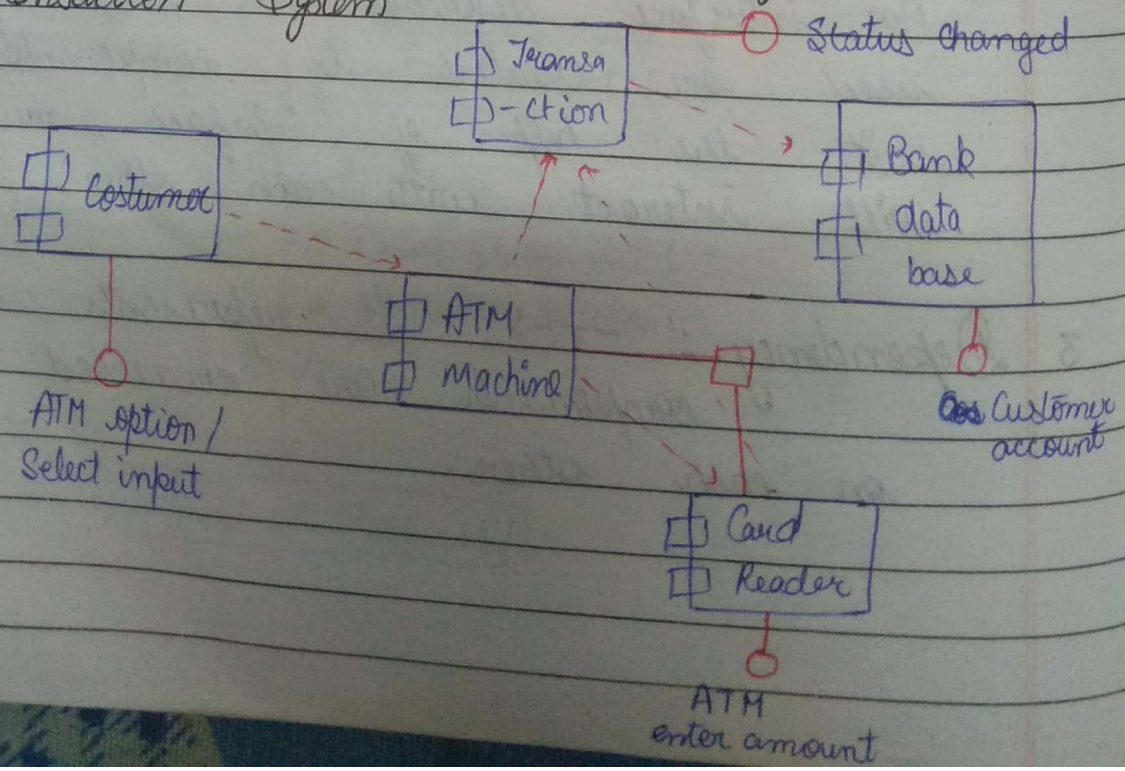
It defines a group of components used to perform similar set of task

6 Note :



It provides additional message to the user about the system

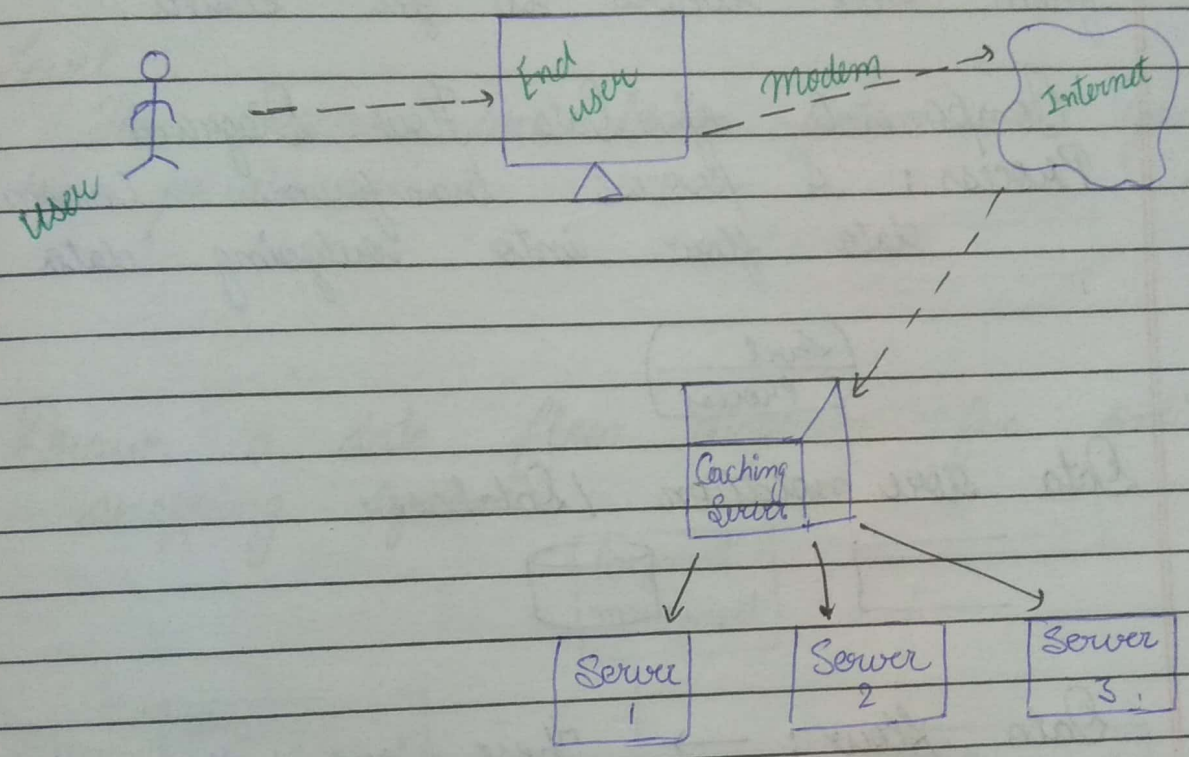
Ques Draw a component diagram for ATM Transaction System



Scanned by CamScanner

Deployment Diagram

This diagram visualise the topology of the physical components of the system where the software component is deployed.



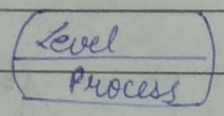
Scanned by CamScanner

Data Flow Diagram

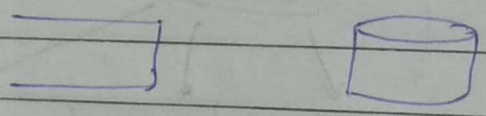
This diagram display how data is processed in terms of input and output. It indicates and focuses on flow of information that were the data comes from and where it get stored

Components of Data Flow Diagram

1 **Process**: A process transforms incoming data flow into outgoing data flow

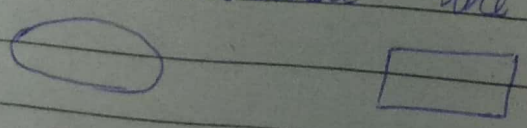


2 Data store notation / Database:



3. Data flow: → These are the pipelines through which packets of information flows

4 External entity: These are the objects outside the system



Ques

Scanned by CamScanner

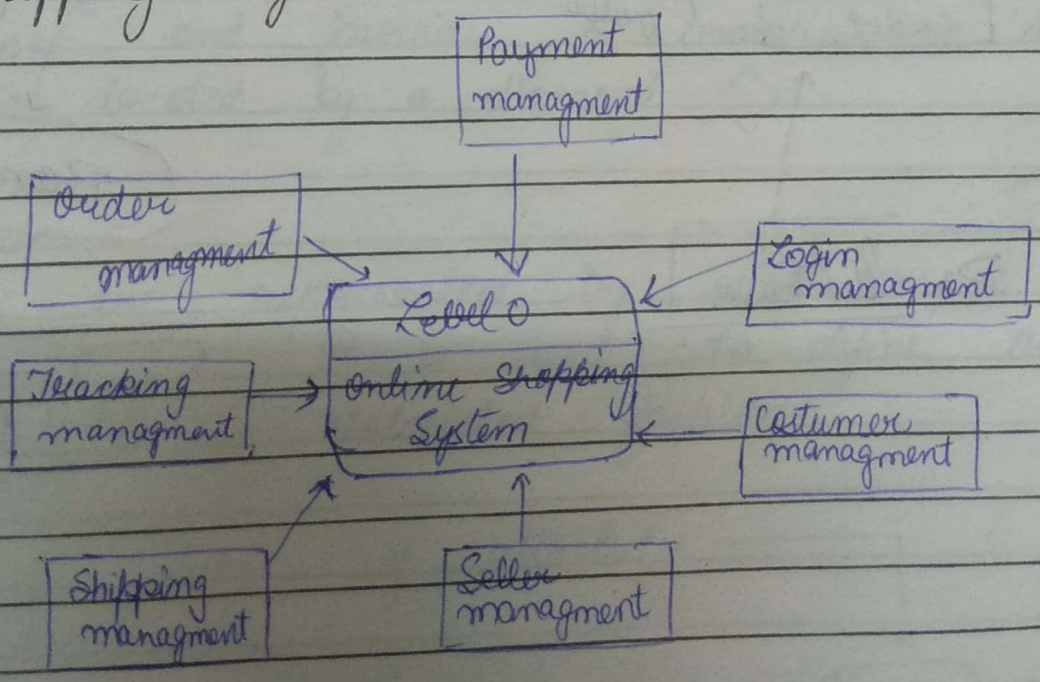
Date: __/__/____ Page no: _____

Level of Data Flow Diagram

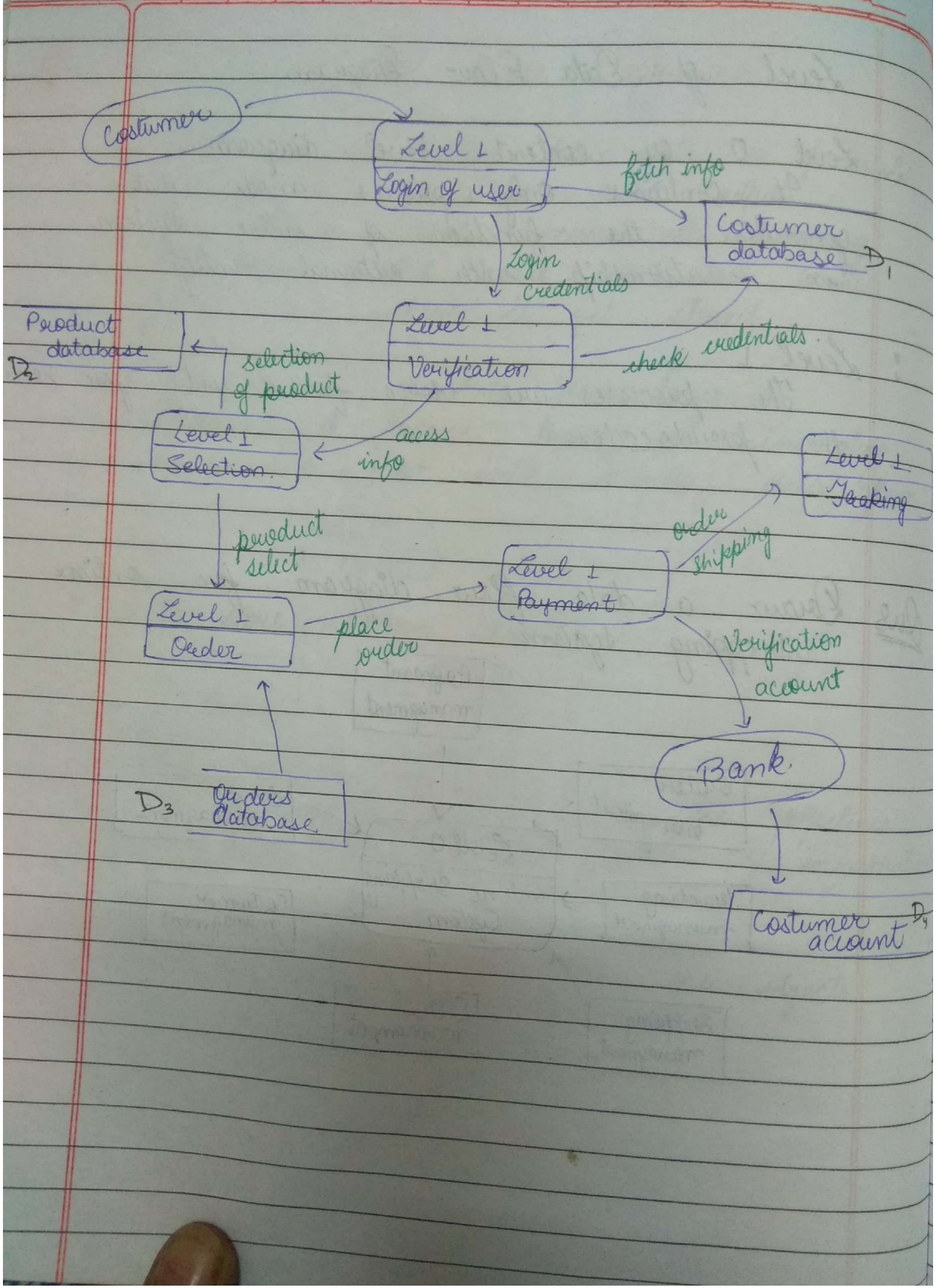
: Level 0 or context level diagram
It contains only 1 process node that generalize the function of entire system in relationship with external entity

: Level 1
The processes are broken further until you reach the pseudocode.

Ans Draw a data flow diagram for online shopping system.



Scanned by CamScanner



Scanned by CamScanner

Entity Relationship Diagram

These diagrams are used to express overall logical structure of a database. This defines different components of a system and the relationship between them.

Components of E-R Diagram :

1 Entity :

It models the object or things used in the system. Represented by a rectangle

2 Relationship :

This defines the connection between two entities and provides a communication channel. It is denoted by a diamond

3 Attributes :

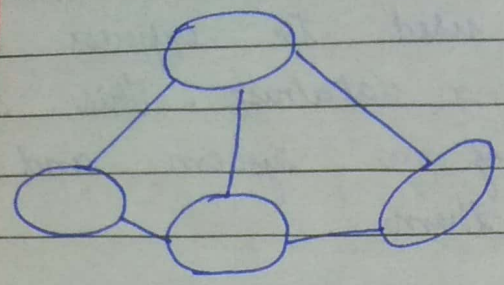
These are represented by oval shaped ~~circle~~ circle . It is used to define the entity in a elaborated fashion.

Types of attributes

Primary attributes
Unique and contains all values (can not be NULL)

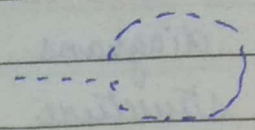
Multi-valued attribute
Can contain more than one value

Scanned by CamScanner



Composite attribute

Composed of multiple attributes

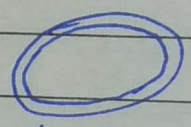


Derived attribute:

Value is calculated from another attributes.

Types of Entities:

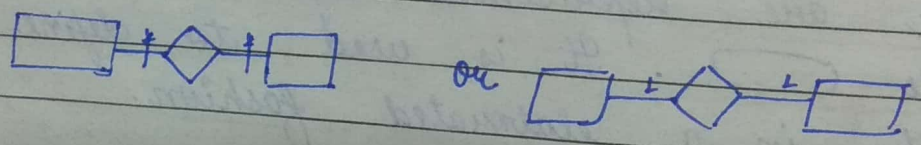
Weak Entity



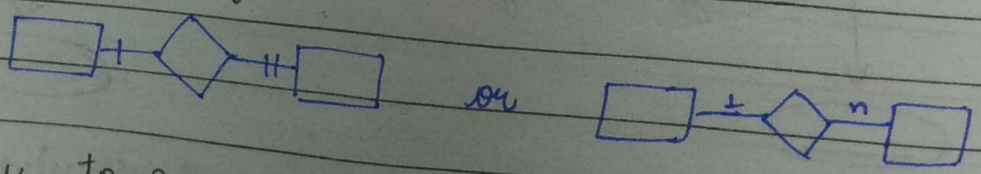
Completely dependent on another entity for its existence

Types of Relationship

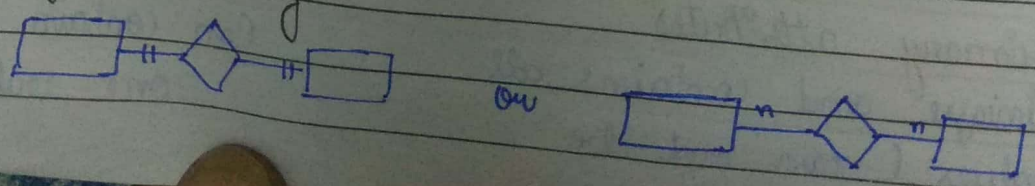
1 One to one



2 One to many

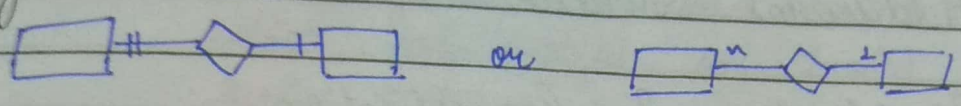


3. Many to many



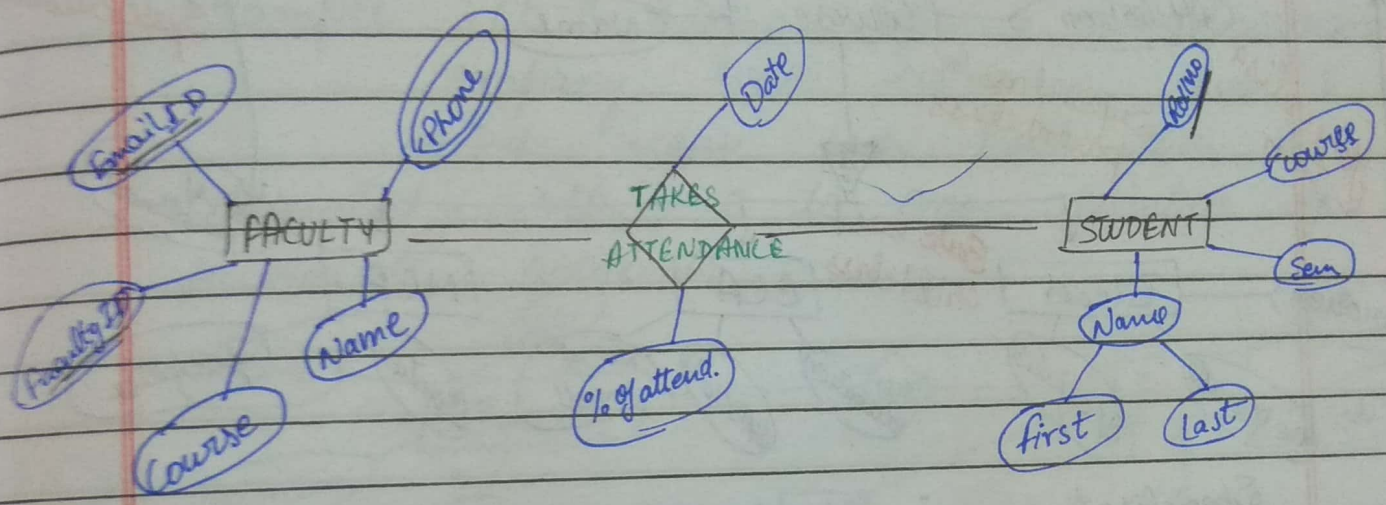
Scanned by CamScanner

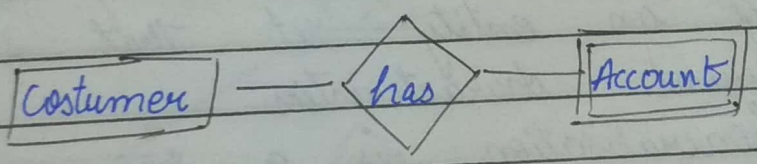
4 Many to one



Example :

2 entities → faculty & students
connected with takes attendance.



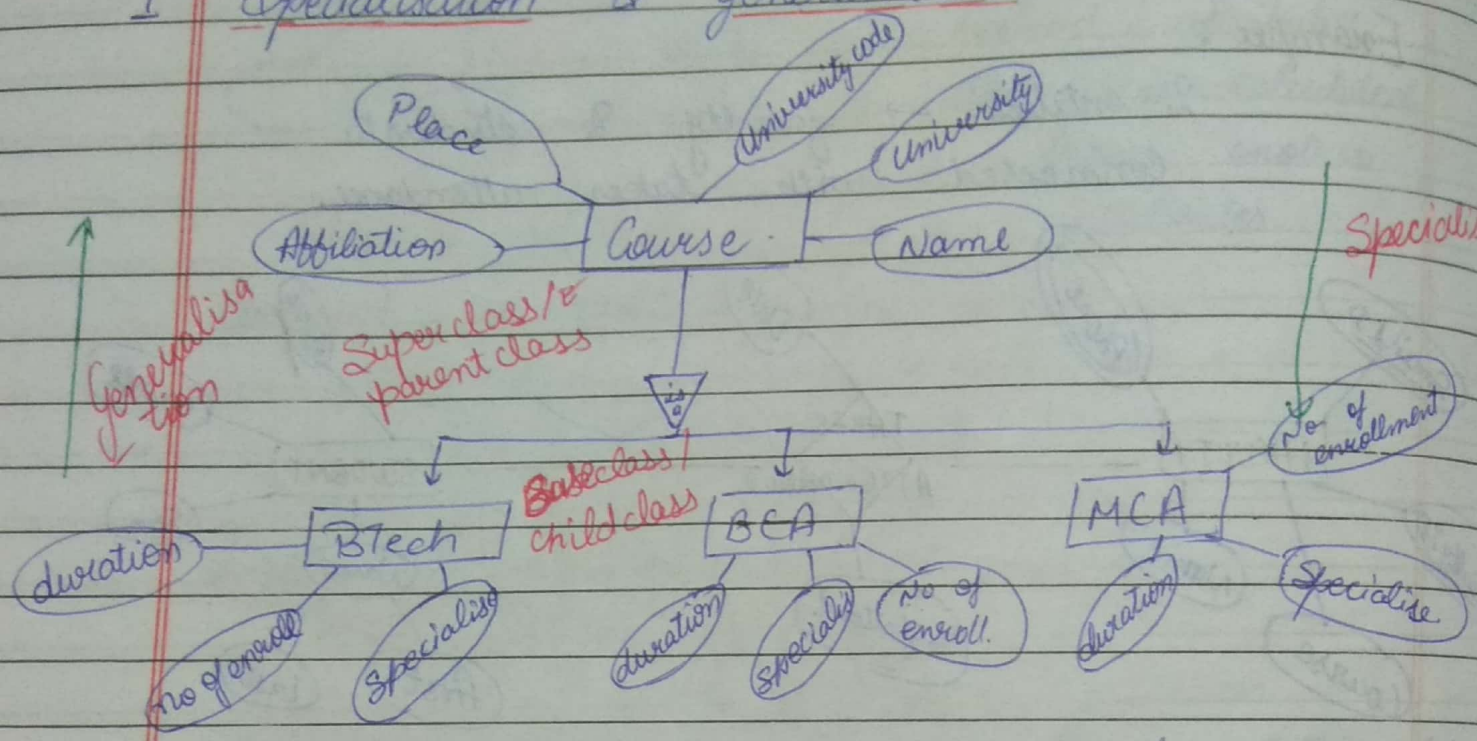


Account is a weak entity
The weak entity cannot have its own primary key. It uses primary key of the dominant entity.
There will be no relevance of account if ~~cust~~ customer is deleted

Scanned by CamScanner

Additional Functionalities in E-R Diagram

1 Specialisation & Generalisation

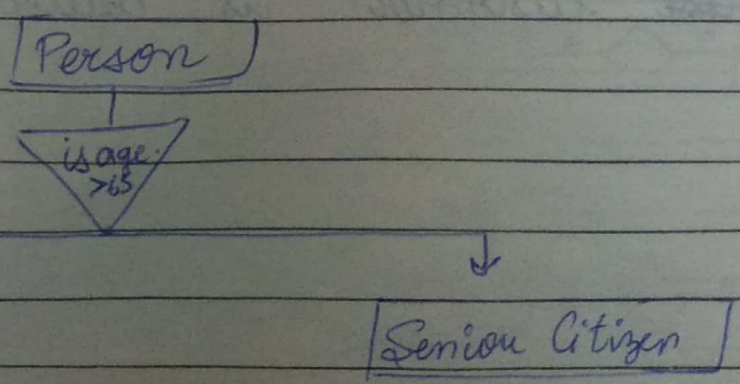


Specialisation is a process of identifying subset of an entity set that share some different characteristics

Generalisation is a process of abstracting similar properties by ignoring difference in properties in a singular set.

We use "is a" relationship to join different entities.

Condition on 'is-a' relationship

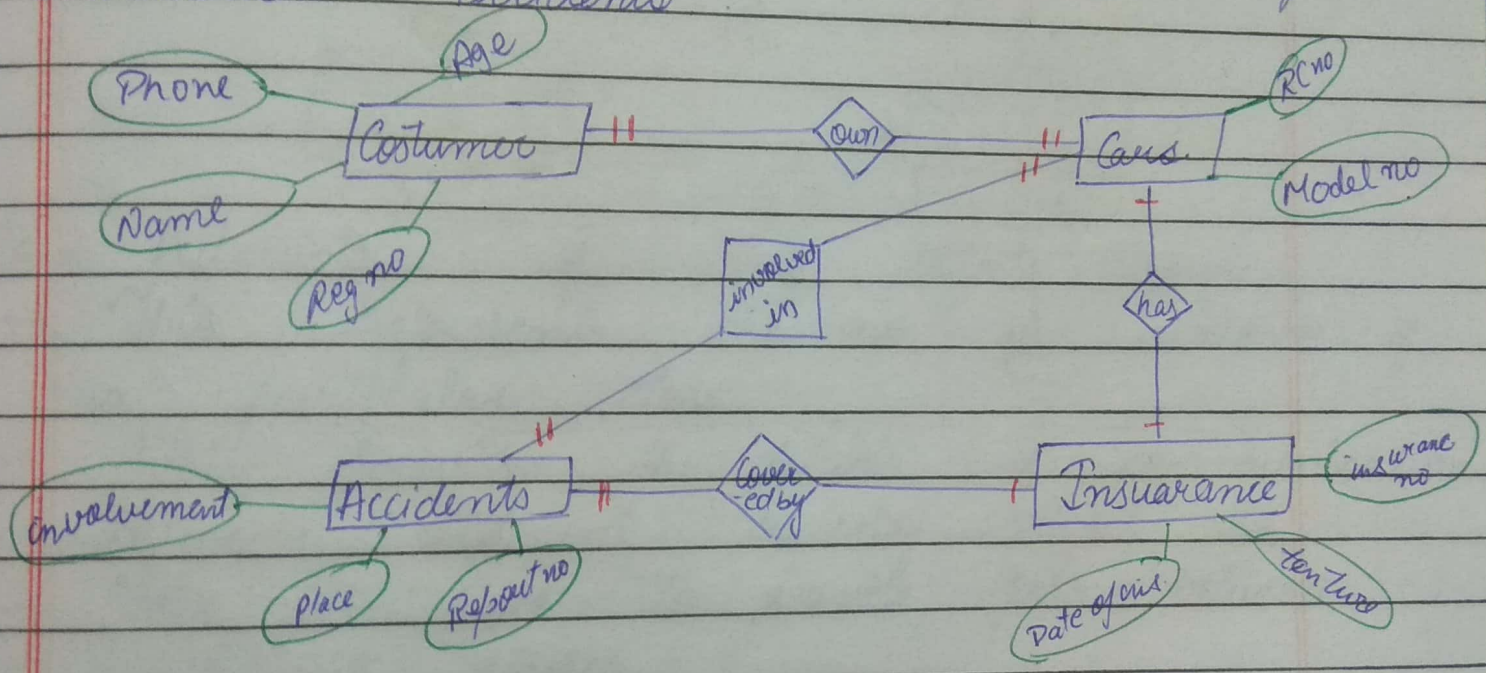


Scanned by CamScanner

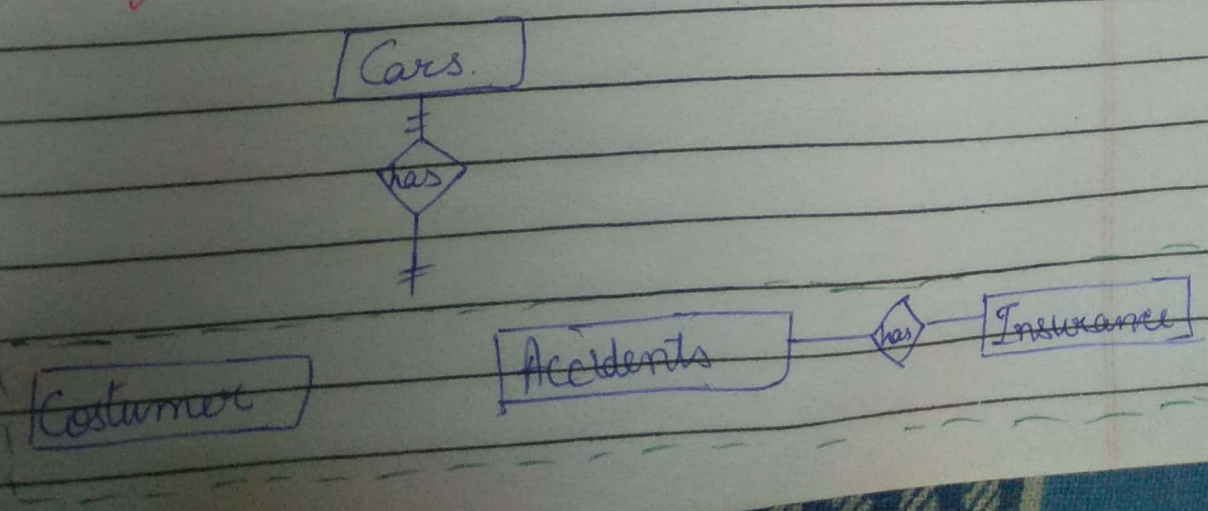
2 **Aggregation** : Aggregation is an abstraction through which relationships are treated as higher level entities.

It allows us to indicate that a relationship set participates in another relationship set.

Example : E-R Diagram for a car insurance company where costumers own 1 or more cars each. Each car is associated with 0 to any number of recorded accidents.

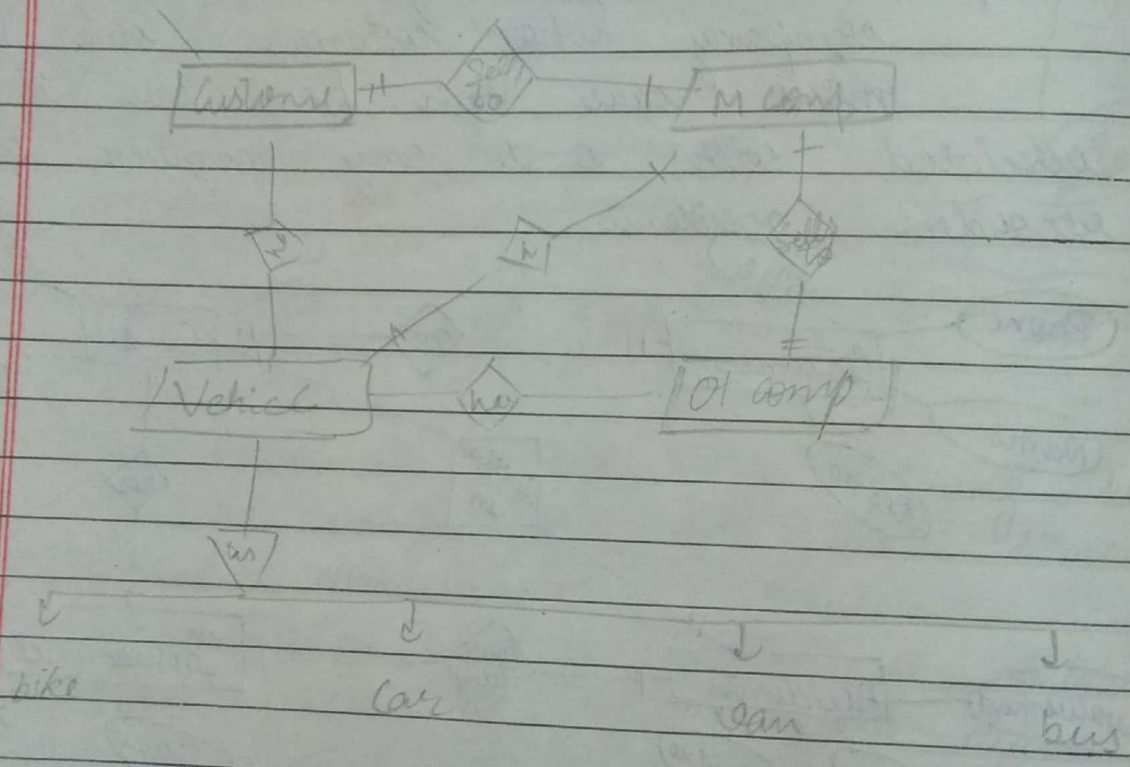


Aggregated diagram

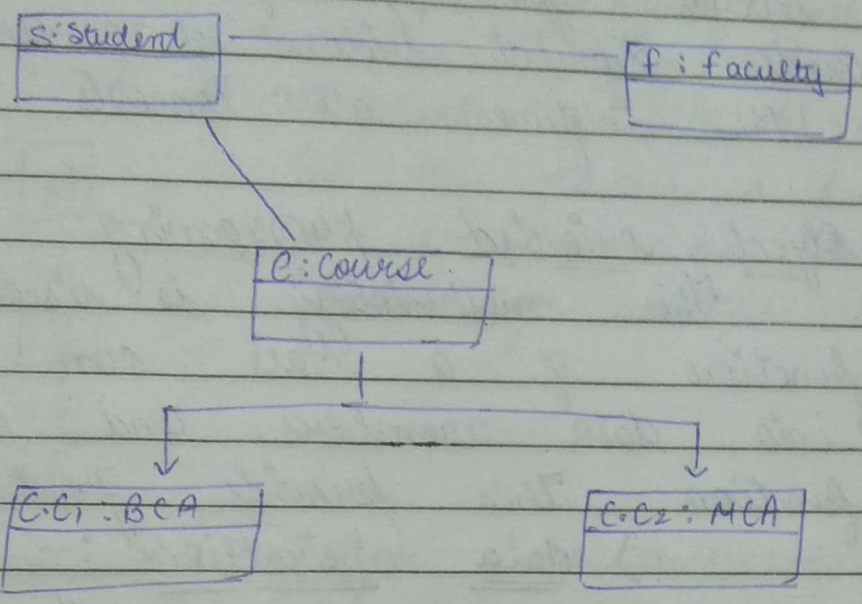


Scanned by CamScanner

Example : E.R. Digram for a motor vehicle Sells Company. Company sells motor cycles, passenger cars, vans, buses to ~~custom~~ customers directly and other companies as well. Justify your placement of attribute as well heirarchi of system.



3 Object diagram : It represents different set of objects present in different classes of the system



Advantages of using object diagram

→ This represents the no of incidences of a particular class.

→ This provides the details of communication between different set of object

→ This is used to provide path between different classes.

Scanned by CamScanner

Unified modeling language (UML)

It is a graphical language used in object oriented development. This includes a several type of system model that provides different views of a system.

UML defines all concepts of OOP language

⇒ **Object oriented programming**

This methodology is used when a function of a class can be segregated into data members and member function. This provide features -

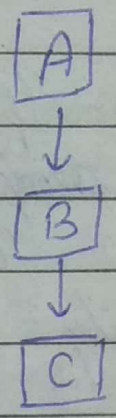
a) **data abstraction** :- It denotes showing only essential details to the user and hidding all the unneccesary complex functionality

eg: When a user register to a particular website, the only thing visible to user is the form, that he has to fill rest of the things running in background like inserting data into database providing unique id to the user are not shown to the user.

Scanned by CamScanner

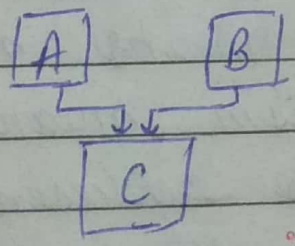
b) Inheritance :- It is a basic to provide code reusability. Here we can use the function again and again with the help of inheritance. Types of inheritance -

i) Multilevel inheritance :-



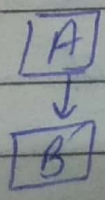
In this process a functionalities or operations are transferred in terms of multiple levels

ii) Multiple inheritance :-



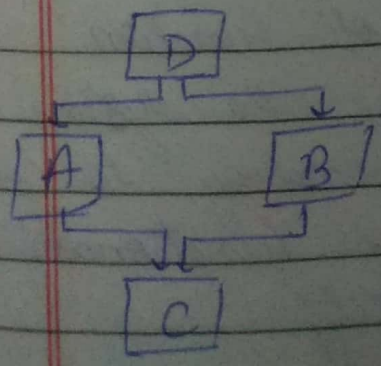
In this a singular class is extracting the properties of more than one parent class

iii) Single inheritance :-



In this a single child class inherits properties from only a single parent class

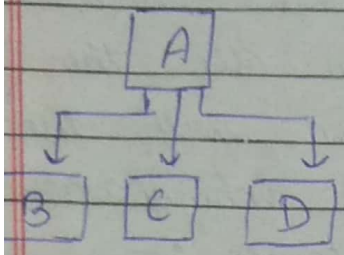
iv) Hybrid Inheritance :-



Combination of any two or more inheritance is known as hybrid inheritance

Scanned by CamScanner

iv) Hierarchical Inheritance :-



When a singular class passes its property to more than one ~~po~~ child classes it is called heirarchical inheritance.

c) Data encapsulation :- It is performed by ~~tiny~~ binding data member and member function into a single entity.

d) Polymorphism :- It is constructed by poly and morphism where poly means many and morphism means form. It is used to represent data into more than one form. Method overloading and method overridding are used for this process.

e) Classes :- It represent the blueprint of function performed by the system. It is a methodology to combine similar set of functionality as a single entity.

Scanned by CamScanner

Date: __/__/__ Page no: ____

f) Objects :- These are instance of classes used to access data members and methods of the class

⇒ Object oriented Architecture

It defines creating a model based on the concepts of oop. It consist of a design that can define fundamentals of abstraction in the system and independent objects.

Characteristics of object oriented design -

a) Prototyping Support :- It helps to design class and functionalities which could better be understood by user.

b) Integrated environment :- It creates an environment for the system where different set of objects can communicate with each other

c) Security of access : The design can better represent the aspects of security by providing different access specifiers

Scanned by CamScanner

f) Information sharing :- It provides mechanism for sharing information among multiple components with messages or controls

⇒ Object Oriented Design

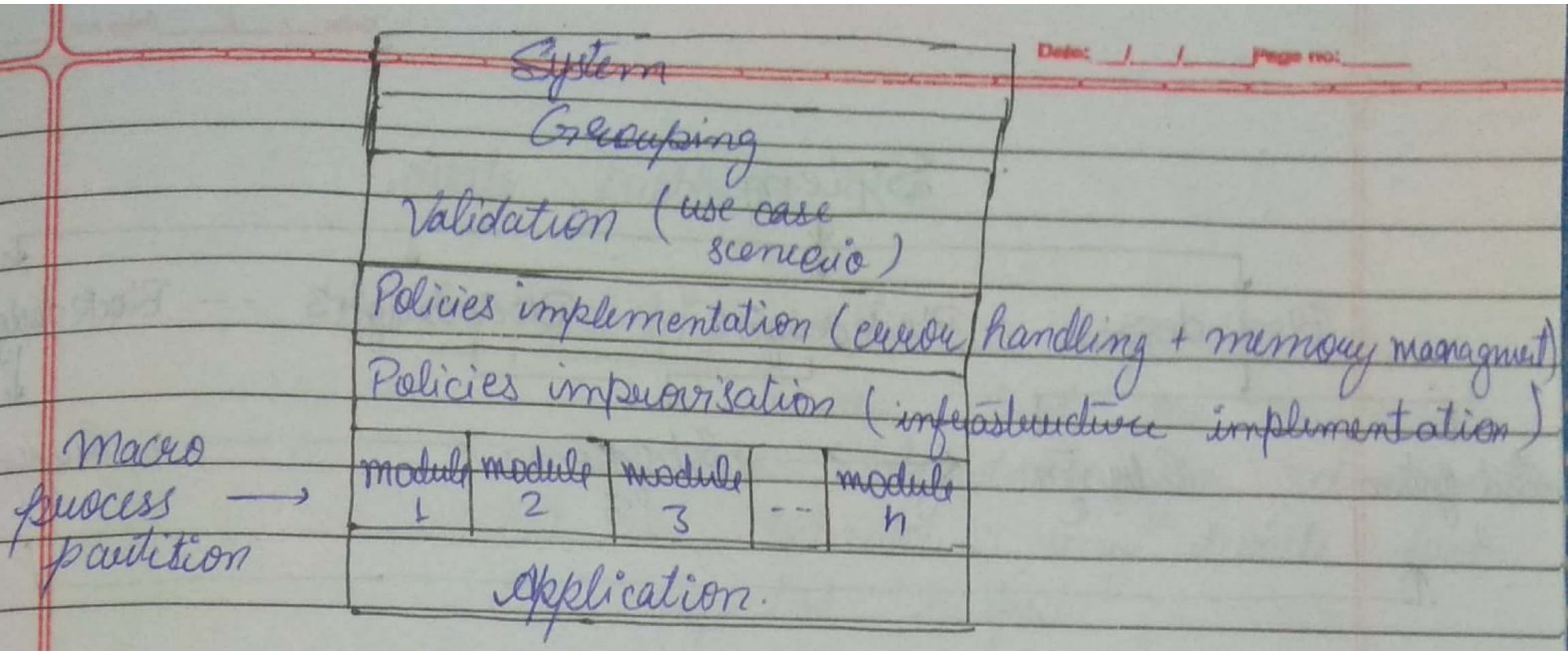
Object oriented system draws upon class definition that are derived from object oriented analysis process. It develops a process to design a product containing following features

a) Abstraction
b) Functional independency
c) Modularity

Various OOD approaches

a) Booch method :- In this method macro development process is applied. Consist of planing activity, clustering similar objects in separate partitions, layering objects by levels of abstraction and creating a prototype and validating it by appling different usage scenarios. In macro development process domain specific policies for memory managment error handling and infrastructure functions are defined

Scanned by CamScanner

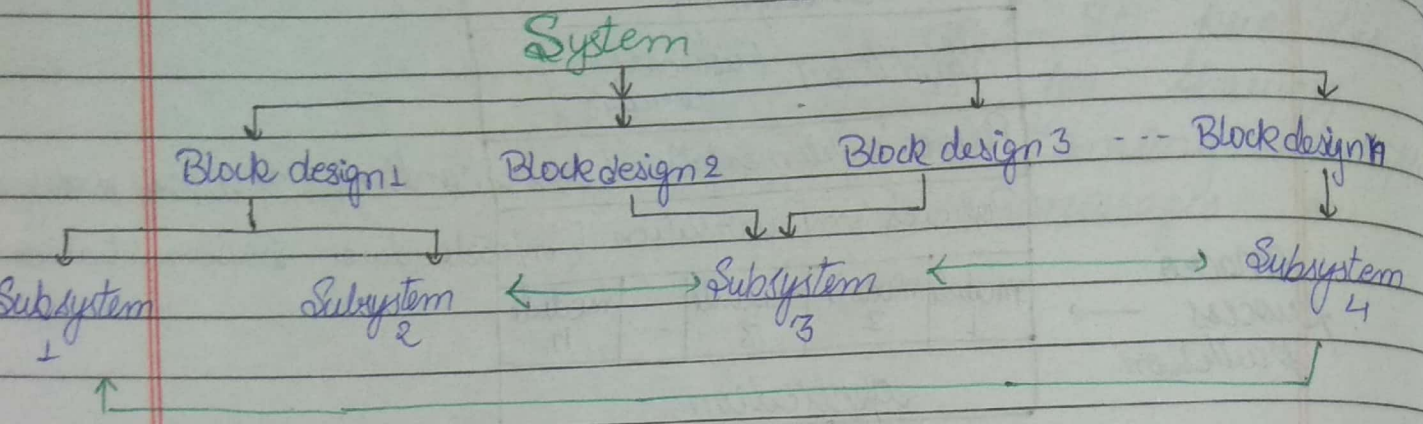


b) Rumbaugh Method :- In this method, we deivde the process into 2 levels of abstraction

System Design	Object Design
It focuses on components, their layout and what is their place in the system	Emphasise on detailed architecture of communication of individual objects

c) Jacobson Method :- In this, first a primary design of the object is created called as blocks then they are categorised according to their working areas. Communication betn these blocks are organised and they work as sub systems.

Scanned by CamScanner



d) Codd & Youdan Method :- This method addresses the application as well as the infrastructure of the application. It applies all the object oriented objects to design the system.
The basic steps are -
i) Describe subsystem
ii) Define datamanagment, interface support and task managment
iii) Define object design
iv) Create relation between different objects
v) Create communication model
vi) Review design and iterate

Design issues in OOD -
a) Decomposibility :- It is a facility to help the designer to decompose a large problem into subsystems but the levels and referencial integrity can

Scanned by CamScanner

create problem

b) Composibility :- This is a facility that insures that the components could be integrated after decomposition but its behaviour may deviate from the expected output.

c) Understandability :- Decomposing elements should not increase complexity of the system

d) Continuity :- Changes in one module should not affect the working of another module.

e) Protection :- If error appears in one module it must not propogate to other modules

Object description

It can be performed in two forms :-

Protocol description → It describes the object by messages it recieves & the related operations it performs after recieving the message.

Implementation description :→ It define implementation details for each object & contains its internal details, data structure, procedural design & description operation.

Scanned by CamScanner

Case characteristics → The difference betn the information contain in protocol description & in implementation description in terms of users & suppliers of services are defined under case characteristics. It provides detail about group of procedure & people who will access it

Object Relationship Model

It establish relation between elements of class & object relationships. During this, responsibility of each class is defined & properly understood. A connection is established between the classes to find out common relationship. For this we ~~do~~ draw class - responsibility - collaboration (CRC) model.

Scanned by CamScanner

A CRC model is a collection of standard index cards that represent classes. Cards are divided into three sections. At the top of card their is the name of the class. Body of the card contains responsibilities at left and collaborations at right.

optional { class name class type (~~eg~~ Eg: Device) class characteristics (Eg: Tangible, inclusive)	
Responsibility	Collaboration

Index card

The part of CRC diagram —

1) Classes → Classes are entities, things, places or structures in any organisational unit. It could be understood as all nouns are potential classes. Classes can be of 3 types :-

a) Device class → This class is used in defining external entities of the system.

b) Property class → This class represents some important property of the problem environment.

Eg → Classes for collecting pH of water in water pollution control system.

c) Interaction class → They models interaction from other objects.

Eg → Recieving message is an interaction.

Scanned by CamScanner

class that models a network routing protocol system.

OBJECTS :- Objects are instances of classes used to define the class & access the class in the required manner. Six potential characteristics are defined by Cott & Youten to define the object.

a) Retain information → Potential objects will be useful during the analysis if & only if they contain the information about system function.

b) Needed Services → Potential object must have a set of identifiable operations that can change the value of its attributes.

c) Multiple attributes → Single object can access multiple attributes of a system.

d) Common attributes → Object can potentially change values at all occurences.

e) Common operations → Set of operations can be defined that applies to all the occurances of the object.

f) Essential requirements → It should always be related to & connected to all

Scanned by CamScanner

the problem environment of the system.

RESPONSIBILITIES :- It consists of attributes and operations of a class. These operations defines ~~function~~ features of a class collectively & exhibits behaviour of the system.

⇒ Guidelines on Responsibility —

1) It should be as general as possible

2) Information and behaviour related to the responsibility should reside within the same class.

3) Responsibilities should be shared when appropriate.

COLLABORATION :- It represents a contract on communication between multiple classes. It defines relationship between multiple classes and segregation of responsibility.

⇒ Relationship between collaborations can be of multiple types —

1) Is part of relationship : In this we represent which one is a part of another class. It is mostly used to model inheritance.

Scanned by CamScanner

Date: __/__/____ Page no: ____

Eg :- Employee is a part of organisation.

2) Has knowledge of relationship : This is required when one class acquires information from another class.

Eg :- Class student has knowledge of attendance.

3) Depends upon relationship : This happens when two classes have dependency on each other.

Eg : Customer has account

Scanned by CamScanner

Object Oriented Analysis Modeling

It is phase of software development where the system requirements are specified and classes and relations are identified

Three techniques are used in object oriented analysis modeling n

Object modeling —

It is used to develop static structure of the software system in terms of objects. It identifies objects into which functionalities can be grouped It also identifies main attributes and operations. Hence, process of object modeling includes :

1. Identification of objects and grouping them into classes.
2. finding relationship among classes
3. Creating user object model digram

Dynamic Modeling —

The behaviour of the system wrt time and ext changes are examined under dynamic modeling It can be defined as a way of describing how individual objects responds to events that are triggered by other objects or outside world. So, overall it includes —

1. State of each object
2. Events and capabilities of actions.
3. Validation of transaction in states via state transation diagram.

Scanned by CamScanner

Functional Modeling —

It shows processes performed by the object and how data is changed by processes. So, overall it provides :-

1. Input & output of system
2. Functional dependancies of the system
3. Identity constraints of the system
4. Optimisation criteria of the system.

Data Modeling

It can be used to represent content of the stored data and relationships that exists among them. There is no direct mechanism for representing relationship but graphical notations of object oriented analysis can be used.

Data, Attributes and Relationships

When data modeling is used as requirement analysis technique it begins by creating models for data obj. Data is a representation of information that must be understood by the software.

Attributes : defines object in clearer manner.

It can be of multiple types.

1. Naming attributes → It creates name of the obj.
2. Identifier → It is a unique and not null attribute that defines the object and can be used as primary key attribute.
3. Descriptive attribute → This describes the obj in such a manner that it could be understood easily.

Scanned by CamScanner

Data modularity (self)

Date: / / Page no:

(4) Referential attributes → They create a relationship between multiple attributes by creating a referential entity

Model (naming attribute)	Id number	Body type	Color	Owner
750IOL	A I	Sports	Black	Aman
LAHOD	B23	Sports	Blue	Barden
PL	N425L	Utility	Red	Catelin
	Identifier	Descriptive attribute		Referential Attribute

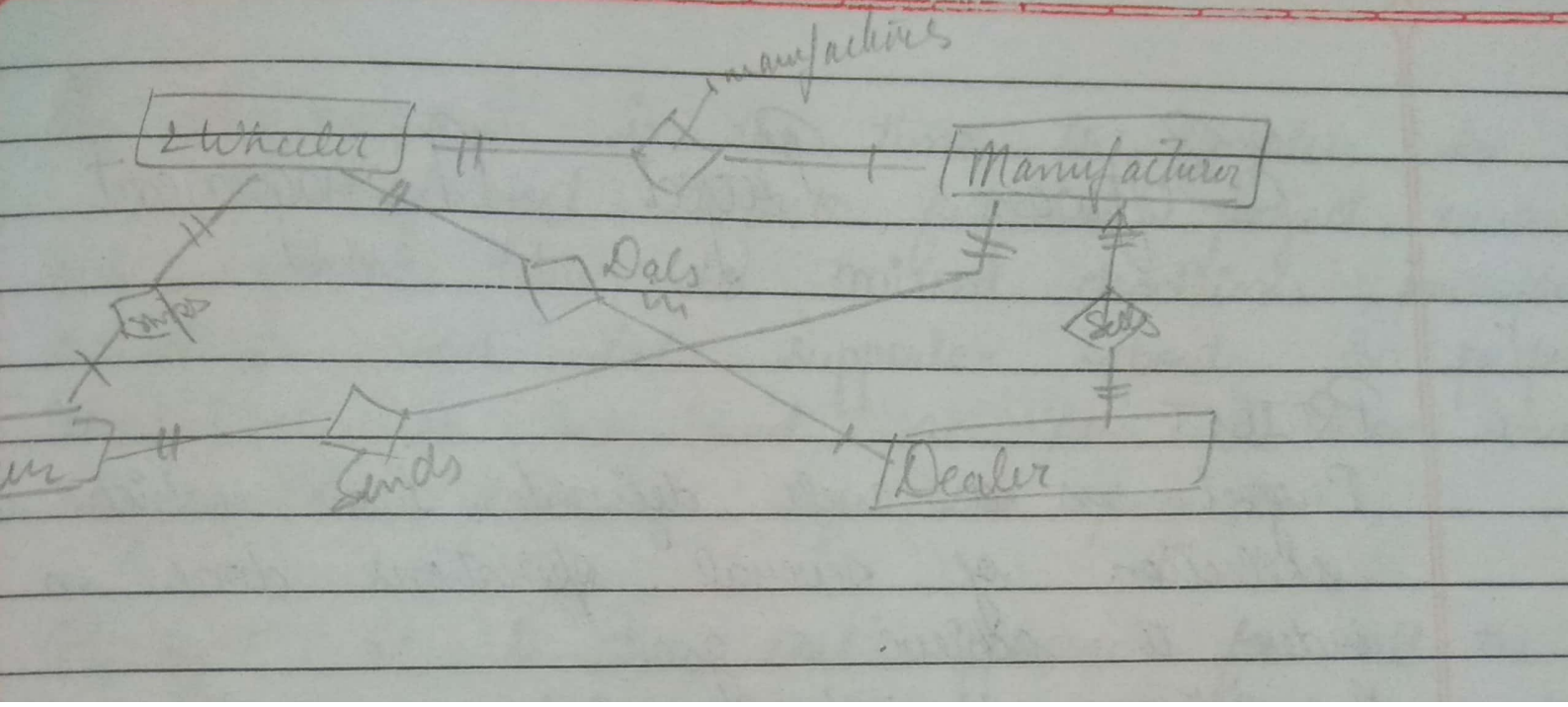
Relationships : are defined to connect two objects of the system. Normalisation is applied to make optimized relationship between different set of objects.

Entity Relationship Digram : Done earlier

Draw an ER-diagram which defines the process of Transportation of a two wheeler. It contains 2-wheeler, Shipper, manufacturer and dealer as the data objects!

Scanned by CamScanner

Date: __/__/____ Page no: _____



Scanned by CamScanner