20/9/19

Ch-2 (Software Engineering)

* Software Engineering introduction :-

- SE is the application of a systematic, discipline, quantifiable approach to the development, operation and maintenance of Software.
- It is Based on Layered Technology:

Tools
↓
Methods
↓
Process
↓
A Quality Focus

- Quality Focus → Any Engineering approach must rest on an organisational commitment to quality.

Process → It is the glue that holds the technology layers together and enables rational and timely development of computer software. Process defines a framework for a set of Key Process Areas (KPAs) that must be established for

defective delivery of S.E. Technology.

- Method → S.E. methods provides the technical how-to-'s for building software methods incompass a broad array of task that includes requirement analysis, design, program, construction. testing & support.
- Tools → S.E. Tools provide auto-mated / semi-automated support for the process and the methods

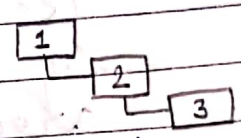
★ A Generic-View of S.E. :-

(i) Analysis
(ii) Design
(iii) Construction
(iv) Verification
(v) Management of Technical-Entities

★ Process Models :-

(i) Linear Sequential or Waterfall Model
(ii) Prototyping Model
(iii) Rapid Application and Development Model (RA
(iv) Component Based development Model (CBD)
(v) Evolutionary Software Process Model :-
  a) Incremental development Model
  b) Spiral Model

1.) Linear Sequential or Waterfall Model -:
The most basic process model.
All the steps are sequentially attached & flows in one direction only.
Advantage: Step by Step working

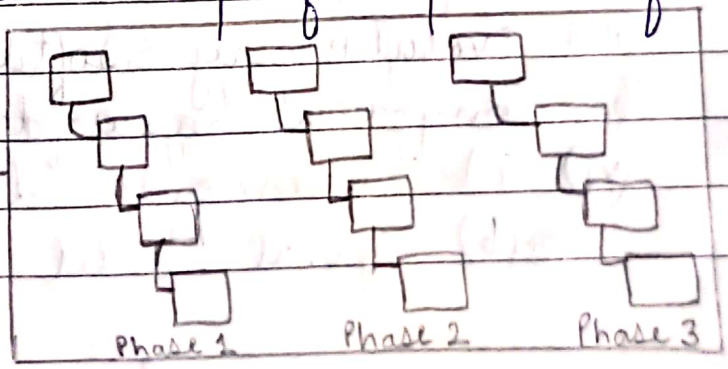


Drawback: If the developer comes on 3rd step and the user tells to modify and change anything, this model fails. It is Because it is not possible to again perform first two steps.

2.) Prototyping Model -
It depends on a formula. User tells requirements and a prototype is made on the base of requirements and modifications are done to satisfy the requirements and a physical original model is made afterwards.
Drawback: User can deny about the non-familiarity of the model, this model fails.

3.) RAD Model -
Best when a user provides a specific period of time for completion of model.
The Best model under a specific period of time / time limits.

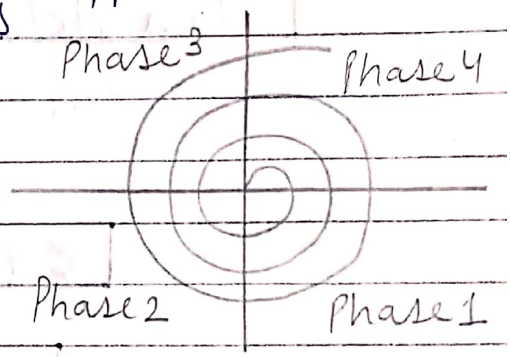
Combination of Waterfall and Prototyping ←



4.) Evolutionary Software Process Model:-

a) Incremental development Model -
Consideration of Production is available here. It mainly focuses on first incremental analysis. Every increment/upgradation is studied and analyzed.

b) Hybrid / Spiral Model (Prototyping + Sequential) -
Based on Risk Driven Approach:
(i) Management Risks
(ii) Technical Risks
(iii) Economical Risks
(iv) Financial Risks



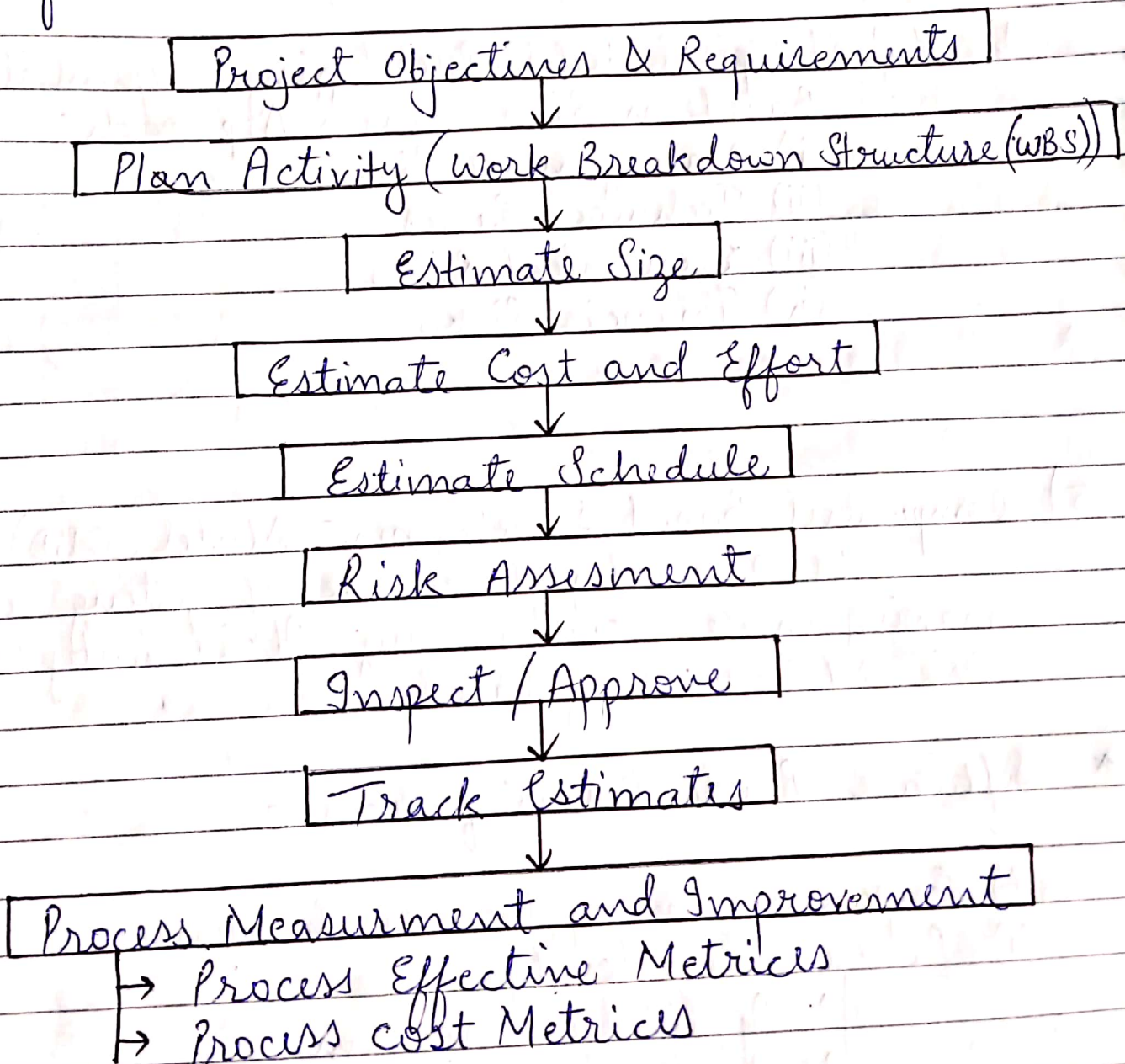
5.) Component Based Development Model (CBD) -
It is developed because of getting rid of making models continously. It is fully Object Oriented Technology based.

* Software Product Cost factors :-

(i) Experience in application domain
(ii) Product Complexity
  → Application Programs
  → Utility Programs
  → System Programs
(iii) Project Size
(iv) Available Time

(v) Programmer Ability
(vi) Level of Technology
(vii) Required level of Technology

★ Imp. Software Cost Estimation Process -



Note: i) Here, Metrics means group of parameters
(ii) Cocomo Model is used for cost estimation.
↓
Imp

* Decomposition Techniques -

- Software cost estimation is a form of problem solving and in most cases the problems to be solved are too complexed to be considered in a single form. Therefore, the problem is decomposed into complements in order to achieve an accurate cost estimate.

- Two Approaches are there:-
  (i) Problem based estimation
    a) FP → Function Point
    b) LOC → Line of Code ($S = (So + 4Sm + Sp)/6$)
  (ii) Process based estimation

- FP → This matrix is used to measure the functionality delevised by the system.
  There are four parameters in function point -
  (i) EI = No. of External Inputs
  (ii) IO = No. of Internal Outputs
  (iii) ILF = No. of Internal Logical Files
  (iv) EIF = No. of External Interface Files
  Formula : FP = Count × WF (weighting factor)
  ↓ Average

* Cost Estimation Model -

(i) Algorithmic (Eg: COCOMO Model) used 80%.
(ii) Non-Algorithmic (Eg: Price to Win) failure Model

→ 10 Marks

- Constructive Cost Model (COCOMO Model) - First invented in early 80's developed by Barry Boehm. It is basically used in cost estimation.

  Category to judge project : (Size → KDLOC)
  a) Organic (Size ≤ 50 KDLOC)
  b) Embedded (Size ≥ 300 KDLOC)
  c) Semi-Detached (Size ≤ 300 KDLOC)

1.) Organic Project → Eg: Small Business system, Library Management
2.) Embedded Project → Complex Projects (Eg: Software used in Military Hardware)
3.) Semi-Detached → Eg: OS, Compiler design

- Three Sub-Models in COCOMO Model: } Hierarchy
  (i) Basic Model
  (ii) Intermidiate Model
  (iii) Advance Model

1.) Basic Model → Formula : $E = A \times (size)^B$ (size → KDLOC)

E → Effort (Person Months (PM)), A → constants, B → constants

Eg:

Project Time	A	B
Organic	2.4	1.05
Semi-Detached	3.0	1.12
Embedded	3.6	1.20

Project Type : Organic
Size : KDLOC

$E = 2.4 \times (30)^{1.05}$

$E = 85$ PM approx.

2) Intermidiate Model → It consider software reliability and complexity as parameters along with size and effort.
There are total four parameters.

Step 1: Calculate E an initiate estimate of effort

Step 2: Identify a set of 15 parameters, all parameters are rated against a numerical value called multiplying factors. → EAF

Step 3: Calculate Total Effort = EAF X Ei

Eg: P+ = organic
Size = 45 KDLOC

P+	A	B
organic	3.2	1.05
Semi-detached	3.0	1.12
Embedded	2.8	1.20

→ $3.2 \times (45)^{1.05} = 174$ PM

EAF = 0.8895 (1.15 X 0.85 X 0.91 X 1.00).

Reliability, Complexity, Application Example, Program Capability

3) Advanced Model → Here, effort is calculated as a program size and a set of cost drivers for each phase of software engineering. This model incorporates all characteristics of the intermediate model and provides procedures for

adjusting the phase wise distribution of the development schedule. It has four phases.

a) Requirement planning & project design (RPPD)
b) Detail Design (DD)
c) Code & Unit Test (CUT)
d) Integration and Test (IT)

For all these rating, cost drivers are assigned multiplying factors and these are called multiplying factors for analyst capability (ACAP). ACAP is calculated using 25 parameters.

Eg: Total Effort = Ei X ACAP
= 174 X ACAP
= 174(ACAP) PM

* System Analysis -

- What is software requirement?

→ Requirement is a condition or a capability possed by software or system component in order to solve a real world problem. Requirements describes how a systems should act, appear or perform.

For this, when users request for software they posses an approximation of what the new system should be capable of doing.

- Feasibility Analysis

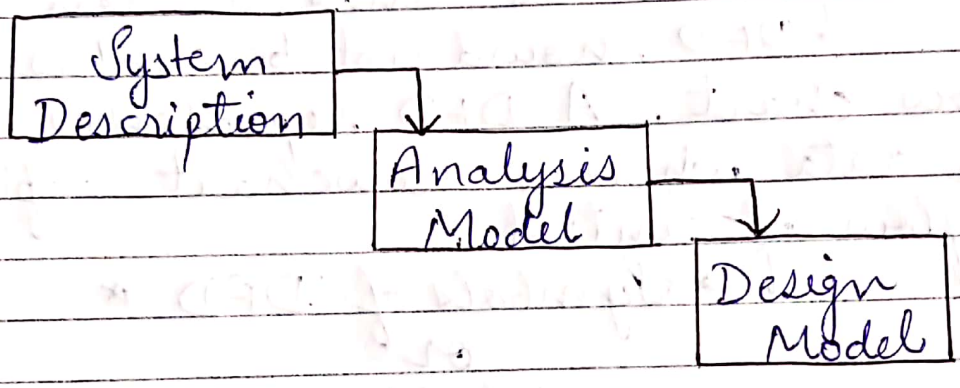
Requirement Analysis according to Ethical rules:-

IEEE defines:

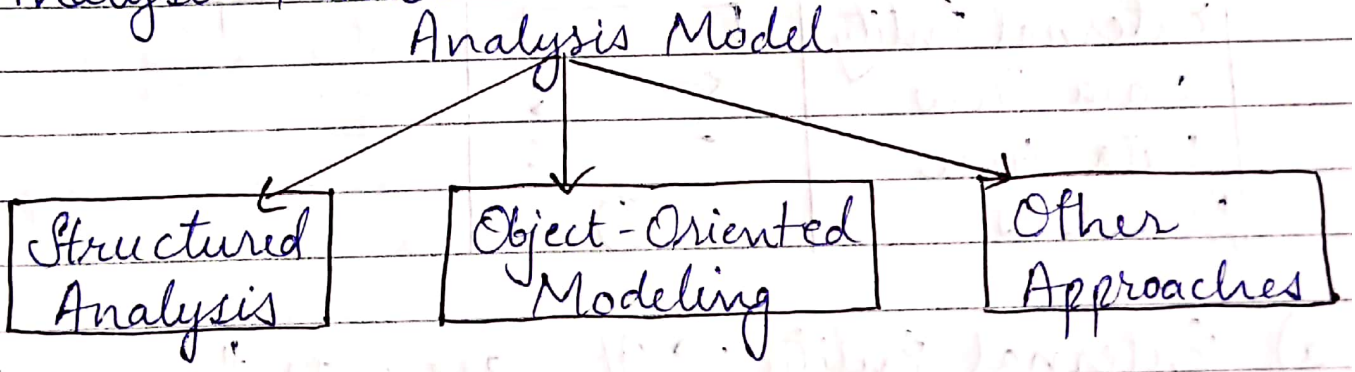
a) The process of studying, user needs to arrive at a definition of a system, hardware or software requirements.

b) The process of studying & refining system, hardware or software requirements.

* Analysis Model as Connector *



- Analysis Model :-



a) Structured Analysis
   - → Data Flow Diagram (DFD)
     - → Level 0 (Context Diagram)
     - → Level 1
     - → Level 2
     - → Level 3 (Detailed Hierarchy)
   - → Data Dictionary

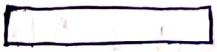
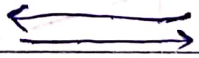
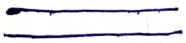
b) Object-Oriented Modeling (OOM)

* Data-Flow Diagram (DFD):

IEEE defines DFD (also known as Bubble Chart or Workflow diagram (WFD)) as a diagram that depicts data sources, data sinks, data storage and processes perform on data as nodes and Logical Flow of Data as links between the nodes.

DFD should not be confused with a flow chart. A DFD represents the flow of data whereas flowchart depicts the flow of control.

* Symbols of DFD *
or
* Notations *

Name	Notation	Description
External Entity		
Data Flow		
Data Store		
Processes		

a) External Entity → It represents the source or destination of data within the system. Each external entity is identified with a meaningful & unique name.

b) Data Flow → Represents the moment of data from its source to destination within the system.

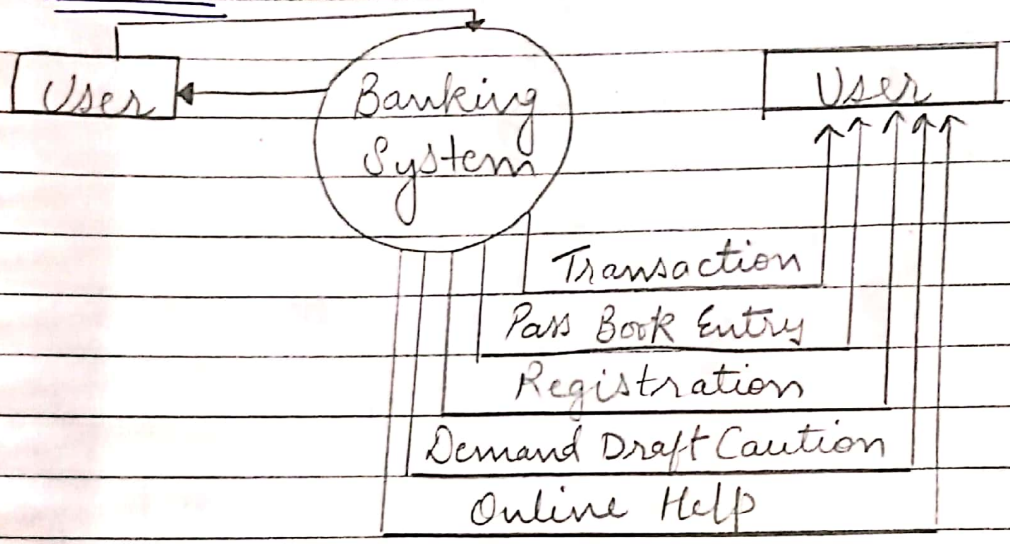
c) Data Store → Indicates the place for storing

information within the system.

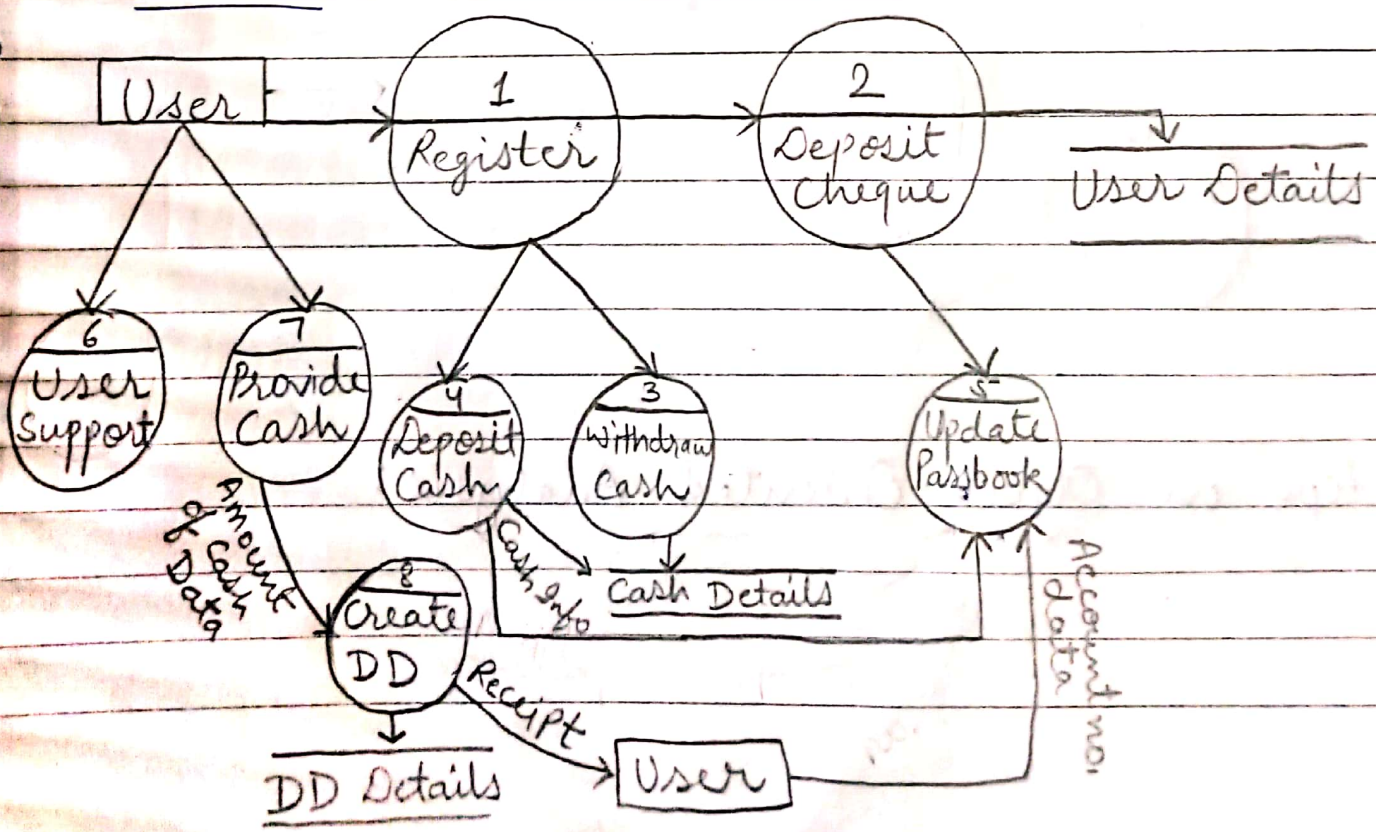
d) Processes → Shows a transformation or manipulation of data within the system. A process is also known as 'Bubble'.

- DFD of Banking System: (Case Study)

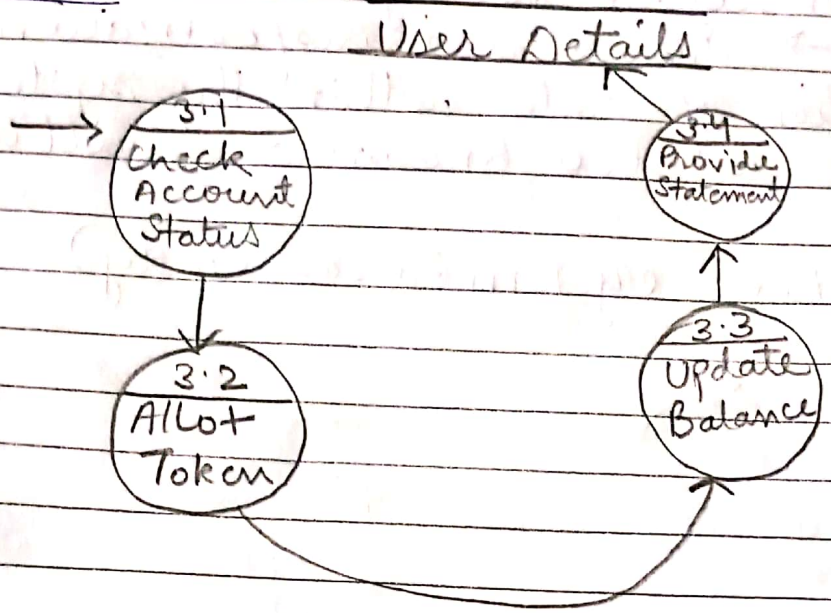
Level 0:



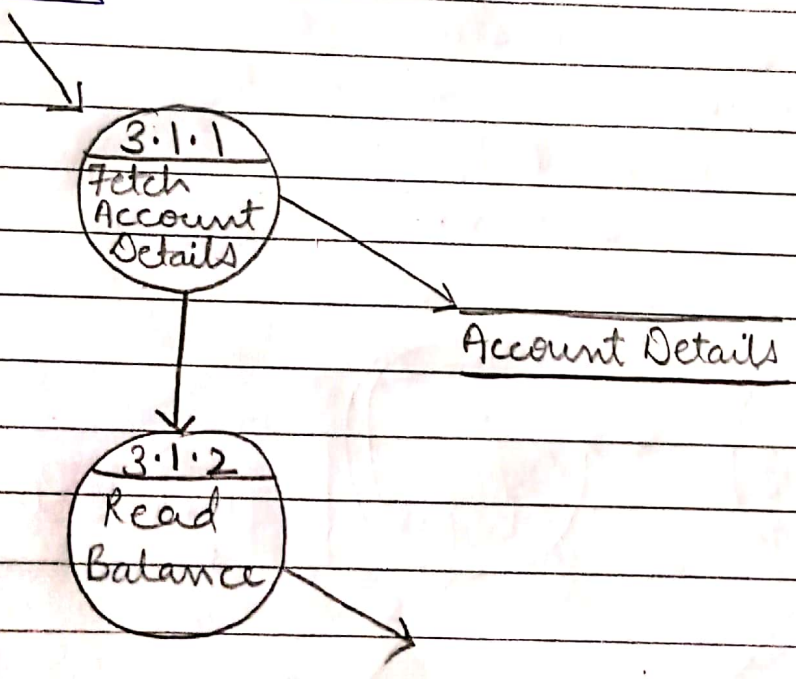
Level 1:



Level 2 :



Level 3 :



★ Steps in Object-Oriented Analysis :