University Institute of Computer Science and Application
(UICSA) RDVV, Jabalpur.

Power point presentation
on
Object Oriented Analysis and Design
"Functional Modeling"
M.C.A (IV sem)

BY:-

MRS. ABHILASHA PANDEY

Functional Modeling

- Functional Modelling gives the process perspective of the object-oriented analysis model and an overview of what the system is supposed to do. It defines the function of the internal processes in the system with the aid of Data Flow Diagrams (DFDs). It depicts the functional derivation of the data values without indicating how they are derived when they are computed, or why they need to be computed.

Data Flow Diagrams

- Functional Modelling is represented through a hierarchy of DFDs. The DFD is a graphical representation of a system that shows the inputs to the system, the processing upon the inputs, the outputs of the system as well as the internal data stores. DFDs illustrate the series of transformations or computations performed on the objects or the system, and the external controls and objects that affect the transformation.
- The four main parts of a DFD are –

- Processes,
- Data Flows,
- Actors, and
- Data Stores.

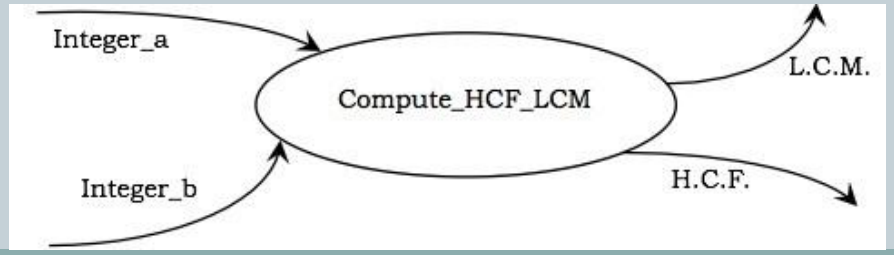
- The other parts of a DFD are –

- Constraints, and
- Control Flows.

Features of a DFD

- **Processes**

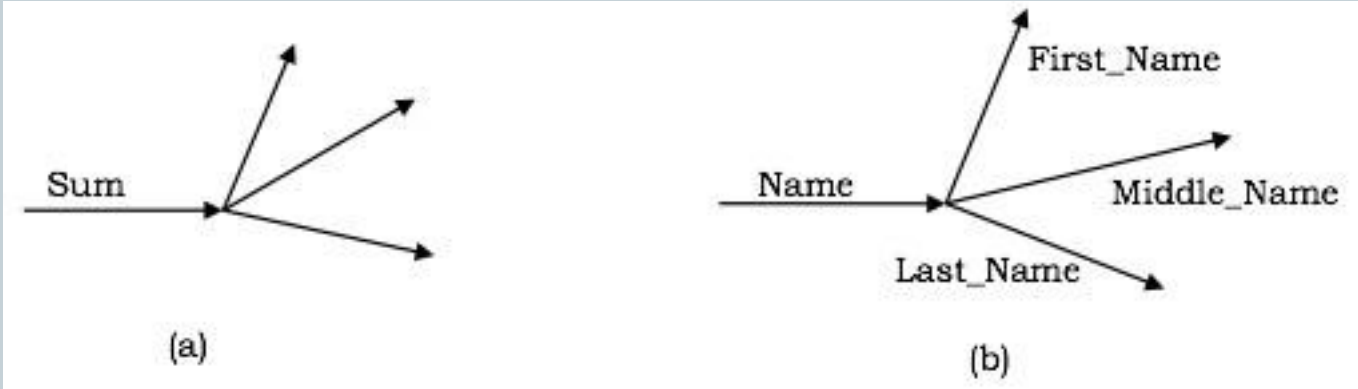
- Processes are the computational activities that transform data values. A whole system can be visualized as a high-level process. A process may be further divided into smaller components. The lowest-level process may be a simple function.
- **Example** – The following figure shows a process Compute_HCF_LCM that accepts two integers as inputs and outputs their HCF (highest common factor) and LCM (least common multiple).
-



Data Flows

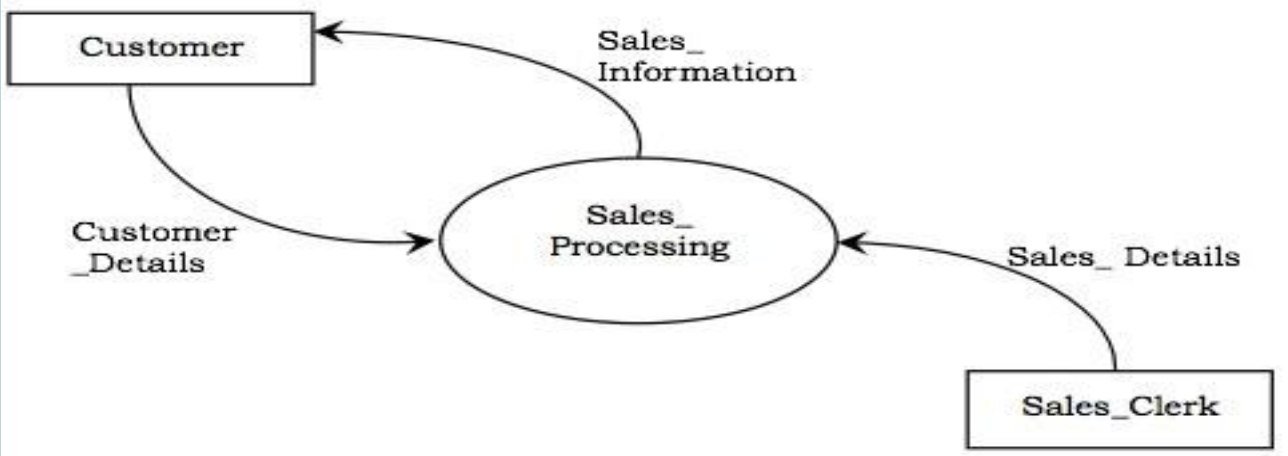
- Data flow represents the flow of data between two processes. It could be between an actor and a process, or between a data store and a process. A data flow denotes the value of a data item at some point of the computation. This value is not changed by the data flow
- **Representation in DFD** – A data flow is represented by a directed arc or an arrow, labelled with the name of the data item that it carries.

Example:-



Actors

- Actors are the active objects that interact with the system by either producing data and inputting them to the system, or consuming data produced by the system. In other words, actors serve as the sources and the sinks of data.
- **Representation in DFD** – An actor is represented by a rectangle. Actors are connected to the inputs and outputs and lie on the boundary of the DFD.
- **Example** –



Data Stores

. Data stores are the passive objects that act as a repository of data. Unlike actors, they cannot perform any operations. They are used to store data and retrieve the stored data. They represent a data structure, a disk file, or a table in a database.

Example – The following figure shows a data store, Sales_Record, that stores the details of all sales. Input to the data store comprises of details of sales such as item, billing amount, date, etc. To find the average sales, the process retrieves the sales records and computes the average.

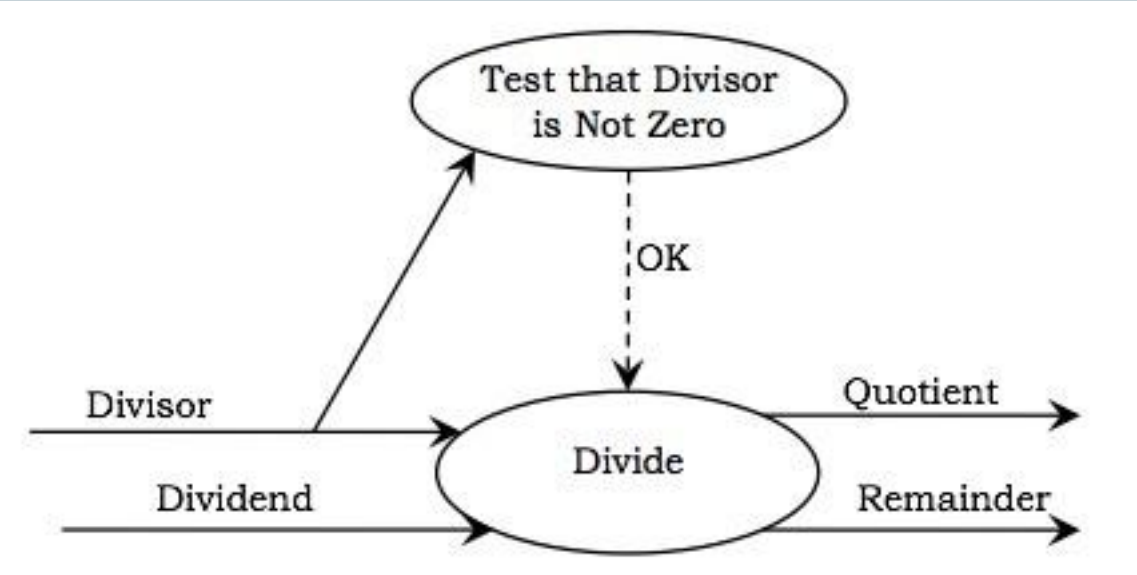
Constraints

- Constraints specify the conditions or restrictions that need to be satisfied over time. They allow adding new rules or modifying existing ones. Constraints can appear in all the three models of object-oriented analysis.
- In Object Modelling, the constraints define the relationship between objects. They may also define the relationship between the different values that an object may take at different times.
- In Dynamic Modelling, the constraints define the relationship between the states and events of different objects.
- In Functional Modelling, the constraints define the restrictions on the transformations and computations.

Control Flows

A process may be associated with a certain Boolean value and is evaluated only if the value is true, though it is not a direct input to the process. These Boolean values are called the control flows.

- **Representation in DFD** – Control flows are represented by a dotted arc from the process producing the Boolean value to the process controlled by them.
- **Example** –

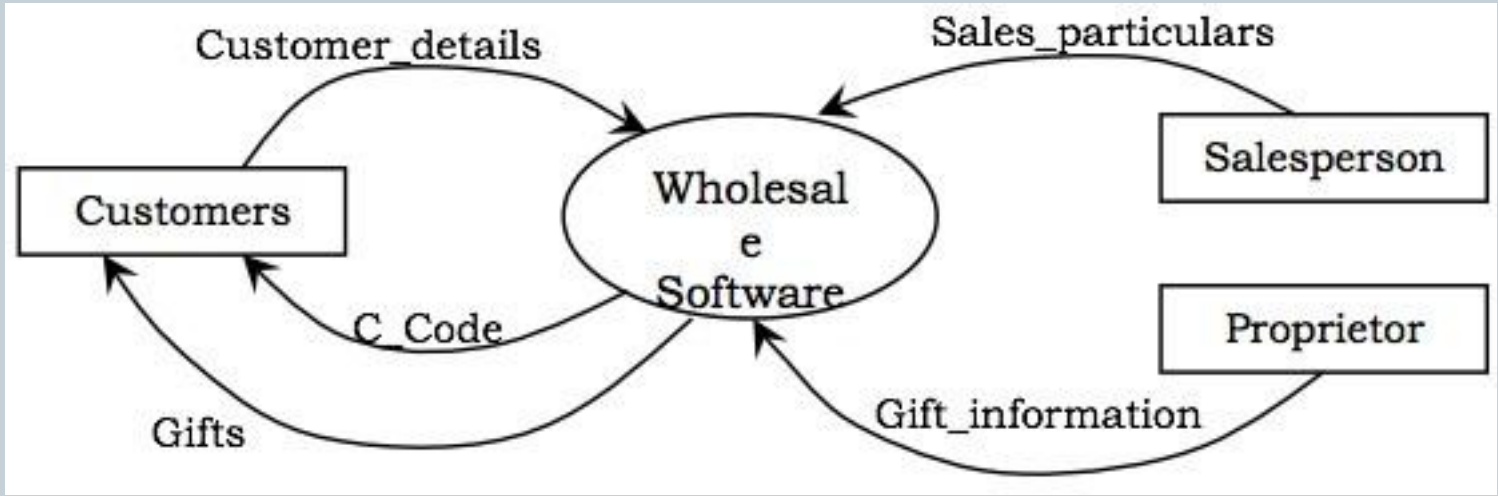


Developing the DFD Model of a System

- In order to develop the DFD model of a system, a hierarchy of DFDs are constructed. The top-level DFD comprises of a single process and the actors interacting with it.
- At each successive lower level, further details are gradually included. A process is decomposed into sub-processes, the data flows among the sub-processes are identified, the control flows are determined, and the data stores are defined. While decomposing a process, the data flow into or out of the process should match the data flow at the next level of DFD.

Example

- Let us consider a software system, Wholesaler Software, that automates the transactions of a wholesale shop. The shop sells in bulks and has a clientele comprising of merchants and retail shop owners. Each customer is asked to register with his/her particulars and is given a unique customer code, C_Code. Once a sale is done, the shop registers its details and sends the goods for dispatch. Each year, the shop distributes Christmas gifts to its customers, which comprise of a silver coin or a gold coin depending upon the total sales and the decision of the proprietor.
- The functional model for the Wholesale Software is given below. The figure below shows the top-level DFD. It shows the software as a single process and the actors that interact with it.
- The actors in the system are –
- Customers
- Salesperson
- Proprietor


Customer_details
Sales_particulars
Customers
Wholesal
e
Software
Salesperson
C_Code
Proprietor
Gifts
Gift_information

Advantages of DFD

* DFDs depict the boundaries of a system and hence are helpful in portraying the relationship between the external objects and the processes within the system.

* They help the users to have a knowledge about the system.

* The graphical representation serves as a blueprint for the programmers to develop a system.

* They are used as a part of the system documentation.

Disadvantages of DFD

- DFDs take a long time to create, which may not be feasible for practical purposes.
- DFDs do not provide any information about the time-dependent behavior, i.e., they do not specify when the transformations are done.
- They do not throw any light on the frequency of computations or the reasons for computations.
- The preparation of DFDs is a complex process that needs considerable expertise. Also, it is difficult for a non-technical person to understand.

* The method of preparation is subjective and leaves ample scope to be imprecise.