UNIT -3

Object Oriented Analysis and Design

Object Oriented Analysis (OOA)

Object Oriented Analysis (OOA) is the first technical activity performed as part of object oriented software engineering. OOA introduces new concepts to investigate a problem. It is based in a set of basic principles, which are as follows-

1. The information domain is modeled.
2. Behavior is represented.
3. Function is described.
4. Data, functional, and behavioral models are divided to uncover greater detail.
5. Early models represent the essence of the problem, while later ones provide implementation details.

The above notes principles form the foundation for the OOA approach.

Object Oriented Design (OOD):

An analysis model created using object oriented analysis is transformed by object oriented design into a design model that works as a plan for software creation. OOD results in a design having several different levels of modularity i.e., The major system components are partitioned into subsystems (a system level "modular"), and data their manipulation operations are encapsulated into objects (a modular form that is the building block of an OO system.).

Object Modeling Technique (OMT)

Object Modeling Technique (OMT) is real world based modeling approach for software modeling and designing. It was developed basically as a method to develop object-oriented systems and to support object-oriented programming. It describes the static structure of the system.

Object Modeling Technique is easy to draw and use. It is used in many applications like telecommunication, transportation, compilers etc. It is also used in many real world problems. OMT is one of the most popular object oriented development techniques used now-a-days. OMT was developed by *James Rambaugh*.

Purpose of Object Modeling Technique:

- To test physical entity before construction of them.
- To make communication easier with the customers.
- To present information in an alternative way i.e. visualization.
- To reduce the complexity of software.
- To solve the real world problems.

Object Modeling Technique's Models:

There are three main types of models that has been proposed by OMT:

1. **Object Model:**
   Object Model encompasses the principles of abstraction, encapsulation, modularity, hierarchy, typing, concurrency and persistence. Object Model basically emphasizes on the *object* and *class*. Main concepts related with Object Model are classes and their association with attributes. Predefined relationships in object model are aggregation and generalization (multiple inheritance).
2. **Dynamic Model:**
   Dynamic Model involves states, events and state diagram (transition diagram) on the model. Main concepts related with Dynamic Model are states, transition between states and events to trigger the transitions. Predefined relationships in object model are aggregation (concurrency) and generalization.
3. **Functional Model:**
   Functional Model focuses on the how data is flowing, where data is stored and different processes. Main concepts involved in Functional Model are data, data flow, data store, process and actors. Functional Model in OMT describes the whole processes and actions with the help of data flow diagram (DFD).

Phases of Object Modeling Technique:

OMT has the following phases:

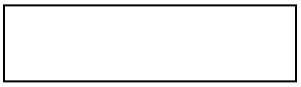
1. **Analysis:**
   This the first phase of the object modeling technique. This phase involves the preparation of precise and correct modelling of the real world problems. Analysis phase starts with setting a goal i.e. finding the problem statement. Problem statement is further divided into above discussed three models i.e. object, dynamic and functional model.
2. **System Design:**
   This is the second phase of the object modeling technique and it comes after the analysis phase. It determines all system architecture, concurrent tasks and data storage. High level architecture of the system is designed during this phase.
3. **Object Design:**
   Object design is the third phase of the object modelling technique and after system design is over, this phase comes. Object design phase is concerned with classification of objects into different classes and about attributes and necessary operations needed. Different issues related with generalization and aggregation are checked.
4. **Implementation:**
   This is the last phase of the object modeling technique. It is all about converting prepared design into the software. Design phase is translated into the Implementation phase.

What is DFD(Data Flow Diagram)?

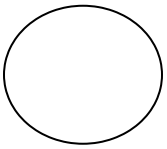
DFD is the abbreviation for **Data Flow Diagram**. The flow of data of a system or a process is represented by DFD. It also gives insight into the inputs and outputs of each entity and the process itself. DFD does not have control flow and no loops or decision rules are present. Specific operations depending on the type of data can be explained by a flowchart. Data Flow Diagram can be represented in several ways. The DFD belongs to structured-analysis modeling tools. Data Flow diagrams are very popular because they help us to visualize the major steps and data involved in software-system processes.

Using any convention's DFD rules or guidelines, the symbols depict the four components of data flow diagrams.

1. **External entity:** an outside system that sends or receives data, communicating with the system being diagrammed. They are the sources and destinations of information entering or leaving the system. They might be an outside organization or person, a computer system or a business system. They are also known as terminators, sources and sinks or actors. They are typically drawn on the edges of the diagram.

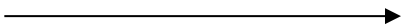


2. **Process:** any process that changes the data, producing an output. It might perform computations, or sort data based on logic, or direct the data flow based on business rules. A short label is used to describe the process, such as "Submit payment."



3. **Data store:** files or repositories that hold information for later use, such as a database table or a membership form. Each data store receives a simple label, such as "Orders."

4. **Data flow:** the route that data takes between the external entities, processes and data stores. It portrays the interface between the other components and is shown with arrows, typically labeled with a short data name, like "Billing details."



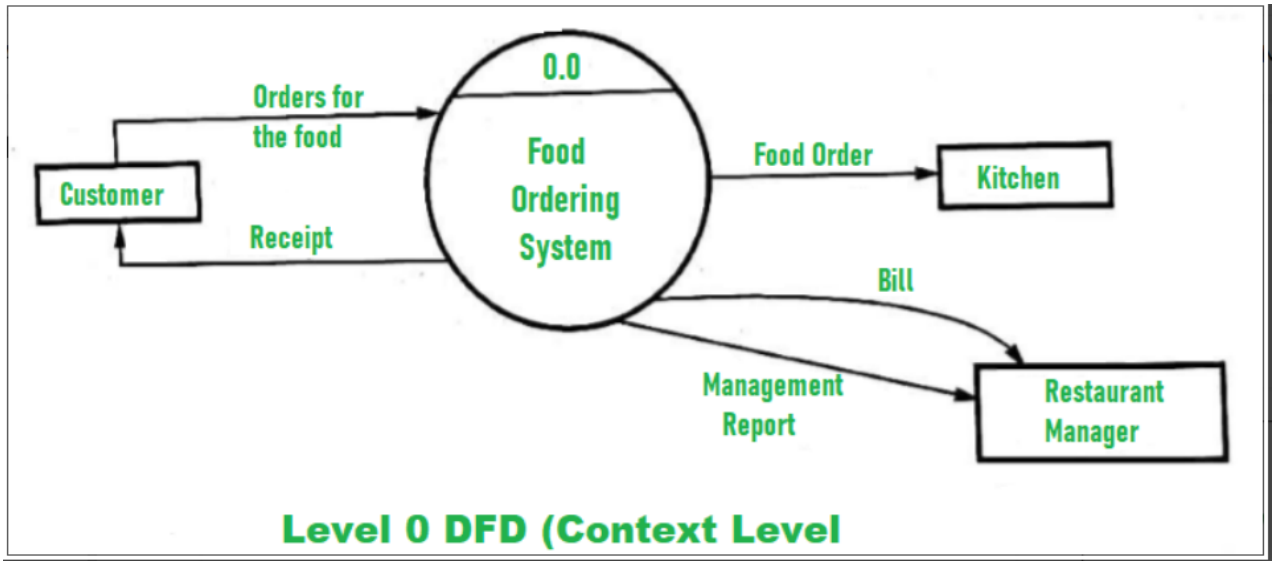
DFD levels

A data flow diagram can dive into progressively more detail by using levels and layers, zeroing in on a particular piece. DFD levels are numbered 0, 1 or 2, and occasionally go to even Level 3 or beyond. The necessary level of detail depends on the scope of what you are trying to accomplish.

DFD Level 0

It is also called a Context Diagram. It's a basic overview of the whole system or process being analyzed or modeled. It's designed to be an at-a-glance view, showing the system as a single high-

level process, with its relationship to external entities. It should be easily understood by a wide audience, including stakeholders, business analysts, data analysts and developers.



Food Ordering System has the following input :

- Food order is input as the customer's order for food.

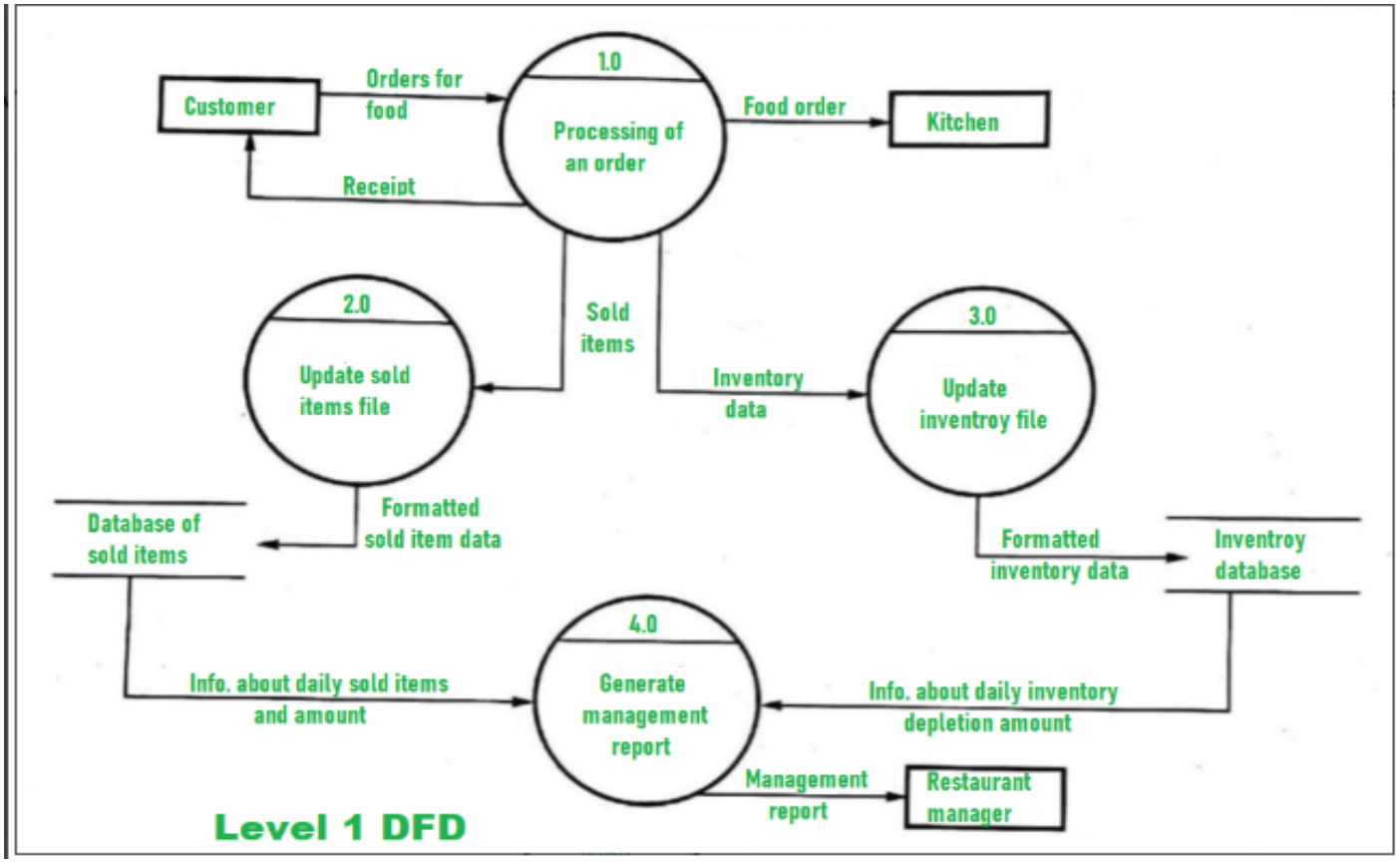
Food Ordering System has the following output:

- Receipt of the order.
- For further processing the order, the food order is passed to the kitchen.
- The restaurant manager gets the report of Bill and Management.

Level 1 DFD

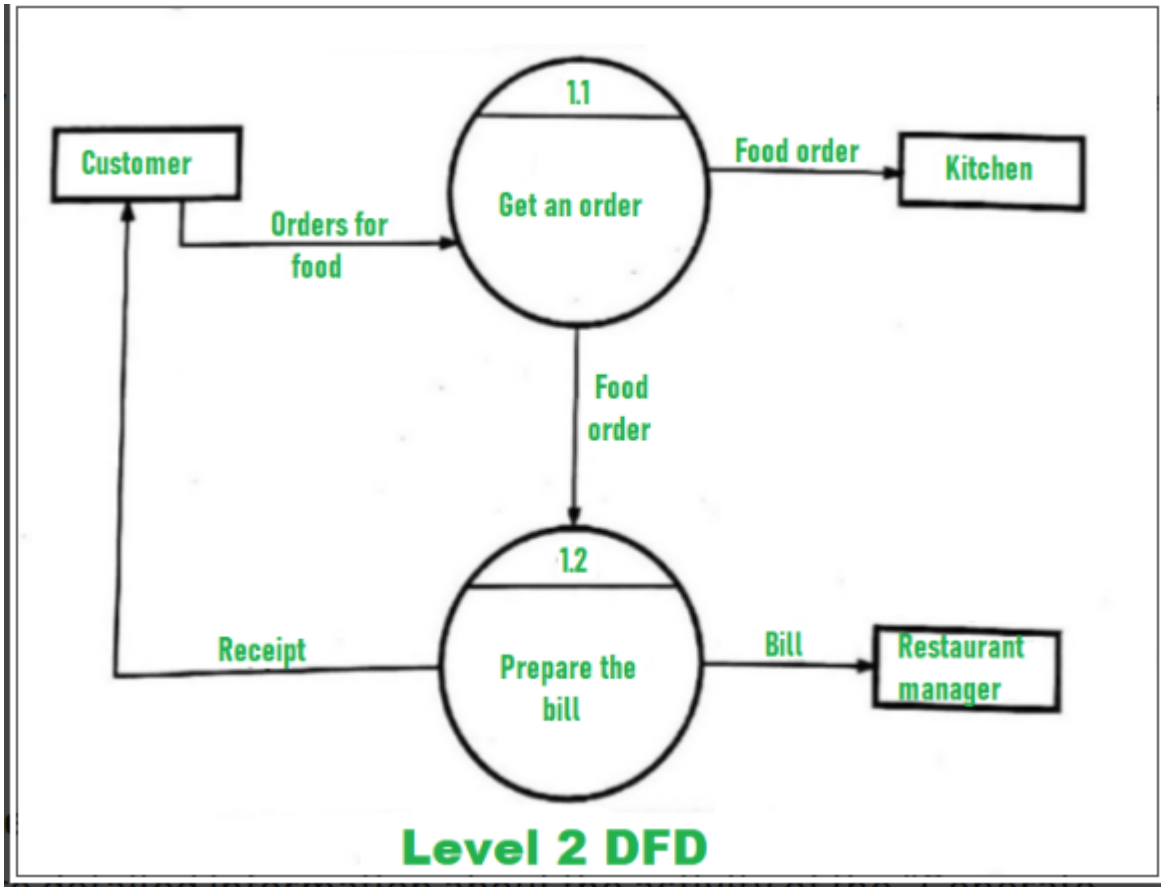
For processing the order, process 1.0 is responsible. For food, the housekeeping activities involved are represented by processes 2.0, 3.0, and 4.0. The detailed information about daily sold items should be available to create and report management and the list of items that are available 'in-stock' should be kept by maintaining the inventory data (describes the records of datasets such as their name, their content, source, many useful information, etc.) at the same time.
Hence, two data stores are used in this level of DFD given below :

- Database of Sold items
- Inventory database



In the end, with the use of the amount of daily sold items and daily inventory depletion, it is easy to prepare a report of management. Further, the restaurant manager gets this report of management.

- **Level 2 DFD –**
  Detailed information about “Processing of an Order” is shown below :



- Progression to Levels 3, 4 and beyond is possible, but going beyond Level 3 is uncommon. Doing so can create complexity that makes it difficult to communicate, compare or model effectively.

Structured Analysis and Structured Design (SA/SD)

Structured Analysis and Structured Design (SA/SD) is a diagrammatic notation that is designed to help people understand the system. The basic goal of SA/SD is to improve quality and reduce the risk of system failure. It establishes concrete management specifications and documentation. It focuses on the solidity, pliability, and maintainability of the system.
Basically, the approach of SA/SD is based on the **Data Flow Diagram**. It is easy to understand SA/SD but it focuses on well-defined system boundary whereas the JSD approach is too complex and does not have any graphical representation.
SA/SD is combined known as SAD and it mainly focuses on the following 3 points:

1. System
2. Process
3. Technology

SA/SD involves 2 phases:

1. **Analysis Phase:** It uses Data Flow Diagram, Data Dictionary, State Transition diagram and ER diagram.
2. **Design Phase:** It uses Structure Chart and Pseudo Code.

1. Analysis Phase:
Analysis Phase involves data flow diagram, data dictionary, state transition diagram, and entity-relationship diagram.

1. **Data Flow Diagram:**
   In the data flow diagram, the model describes how the data flows through the system. We can incorporate the Boolean operators and & or link data flow when more than one data flow may be input or output from a process.

2. **Data Dictionary:**
   The content that is not described in the DFD is described in the data dictionary. It defines the data store and relevant meaning. A physical data dictionary for data elements that flow between processes, between entities, and between processes and entities may be included.
   .

3. **State Transition Diagram:**
   State transition diagram is similar to the dynamic model. It specifies how much time the function will take to execute and data access triggered by events. It also describes all of the states that an object can have, the events under which an object changes state, the conditions that must be fulfilled before the transition will occur and the activities were undertaken during the life of an object.

4. **ER Diagram:**
   ER diagram specifies the relationship between data store. It is basically used in database design. It basically describes the relationship between different entities.

Design Phase:
Design Phase involves structure chart and pseudocode.

1. **Structure Chart:**
   It is created by the data flow diagram. Structure Chart specifies how DFS's processes are grouped into tasks and allocate to the CPU. The structured chart does not show the working and internal structure of the processes or modules and does not show the relationship between data or data-flows
2. **Pseudo Code:**
   It is the actual implementation of the system. It is an informal way of programming that doesn't require any specific programming language or technology.

Jackson System Development (JSD)

Jackson System Development (JSD) is a method of system development that covers the software life cycle either directly or by providing a framework into which more specialized techniques can

fit. JSD can start from the stage in a project when there is only a general statement of requirements.
However many projects that have used JSD actually started slightly later in the life cycle, doing the first steps largely from existing documents rather than directly with the users.

Phases of JSD:
JSD has 3 phases:

1. **Modelling Phase:**
   In the modelling phase of JSD the designer creates a collection of entity structure diagrams and identifies the entities in the system, the actions they perform, the attributes of the actions and time ordering of the actions in the life of the entities.

2. **Specification Phase:**
   This phase focuses on actually what is to be done? Previous phase provides the basic for this phase. An sufficient model of a time-ordered world must itself be time-ordered. Major goal is to map progress in the real world on progress in the system that models it.

3. **Implementation Phase:**
   In the implementation phase JSD determines how to obtain the required functionality. Implementation way of the system is based on transformation of specification into efficient set of processes. The processes involved in it should be designed in such a manner that it would be possible to run them on available software and hardware.

JSD Steps:
Initially there were six steps when it was originally presented by Jackson, they were as below:
1. Entity/action step

2. Initial model step

3. Interactive function step

4. Information function step

5. System timing step

6. System implementation step

Later some steps were combined to create method with only three steps:

1. Modelling Step

2. Network Step

3. Implementation Step

Merits of JSD:

- It is designed to solve real time problem.

- JSD modelling focuses on time.

- It considers simultaneous processing and timing.
- It is a better approach for micro code application.

Demerits of JSD:

- It is a poor methodology for high level analysis and data base design.
- JSD is a complex methodology due to pseudo code representation.
- It is less graphically oriented as compared to SA/SD or OMT.
- It is a bit complex and difficult to understand.