

# Chapter 2

# Process Modeling

# Contents:

- 2. STRUCTURING SYSTEM REQUIREMENTS :Process Modeling (5hrs)**
  - a. What is Process Modeling
  - b. Introduction to Data flow diagrams (DFD)
  - c. Data flow diagramming rules
  - d. Context Diagrams
  - e. Using Data Flow Diagrams in the Analysis Process

# Requirement Structuring

- The amount of information gathered during requirements determination could be huge, especially if the scope of the system under development is broad.
  - Such a large amount of information must be organized in order to be useful.
- Because of the dangers of excessive analysis, today's systems analysts focus more on the system to be developed than on the current system.
  - JAD (Joint Application Design) and prototyping techniques keep analysis effort at minimum.
- Agile methodologies have been proven much popular and effective development model in terms of limiting analysis and producing better quality systems.

# Methods for Requirement Gathering

```
graph TD; A[Methods for Requirement Gathering] --> B[Traditional Method]; A --> C[Modern Method]
```

Traditional  
Method

Modern  
Method

# Traditional methods for Requirement Gathering

1. Interview and listening
2. Direct observation (User and their environments)
3. Analysis of existing procedures and relative documents

# Interview

- Ask people about their work, the information they use to do it, and the types of information processing that might supplement their work.
- Gathering facts, opinions, and speculation is easier through this process.
- Observing body language, emotions, and other signs of what people want and how they assess current systems is a vital aspect
- Analyst must make sure to choose right questions (to ask) and make neutral notes of everything that sounds important.

# Direct observation

- People often do not have a completely accurate appreciation of what they do or how they do it, especially when infrequent events, issues from the past, or issues for which people have considerable passion are involved.
- Because people cannot always be trusted to interpret and report their own actions reliably, analyst can supplement what people tell him/her by watching what they do in work situations.
- It helps to understand the difference between what user claims the requirement to be, and what system really fits in.

# Analyzing current procedure & documents

- Examining system and organizational documentation helps to discover more details about current systems and the organization they support.
- Helps to study legacy system specifications, Problems with existing systems, Opportunities to meet new needs, Organizational Guidelines about new system, etc.

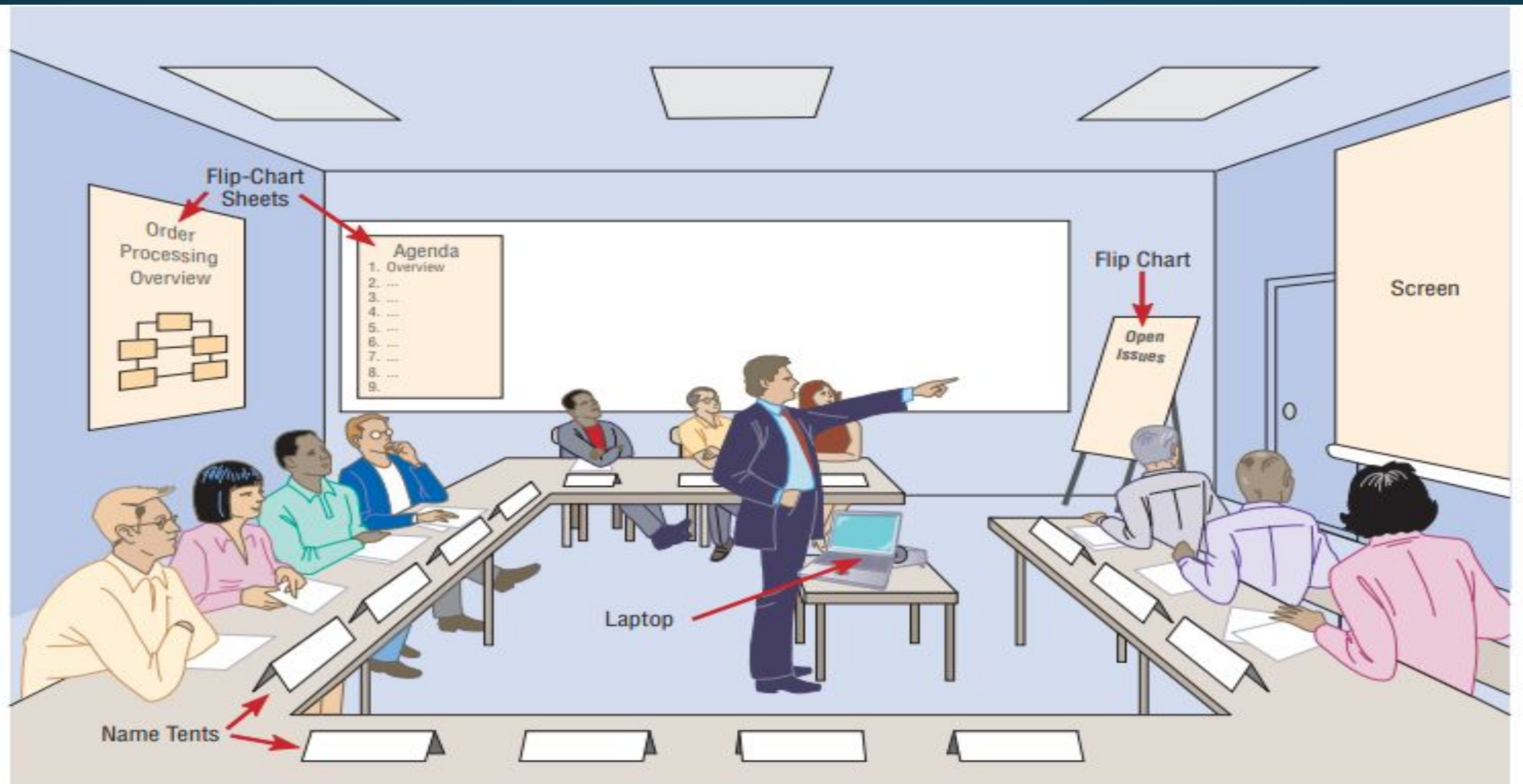


# Modern methods

- Joint Application Design (JAD)
- Prototyping

# JAD

- The primary purpose of using JAD in the analysis phase is to collect systems requirements simultaneously from the key people involved with the system.
  - The result is an intense and structured, but highly effective, process.
- Having all the key people together in one place at one time allows analysts to see the areas of agreement and the areas of conflict.
- Meeting with all these important people for over a week of intense sessions allows you the opportunity to resolve conflicts or at least to understand why a conflict may not be simple to resolve



**FIGURE 5-6**

A typical room layout for a JAD session.

Source: Based on Wood and Silver, 1989.

# Prototyping

- Prototyping is a repetitive process in which analysts and users build a rudimentary version of an information system based on user feedback.
- To establish requirements for prototyping, you still have to interview users and collect documentation.
  - Prototyping, however, allows you to quickly convert basic requirements into a working, though limited, version of the desired information system.
  - The user then views and tests the prototype.

**TABLE 5-9 Stages of System Implementation of WebStore**

**Stage 1 (Basic Functionality)**

Simple catalog navigation; two products per section—limited attribute set

25 sample users

Simulated credit card transaction

Full shopping cart functionality

**Stage 2 (Look and Feel)**

Full product attribute set and media (images, video)—commonly referred to as “product data catalog”

Full site layout

Simulated integration with Purchasing Fulfillment and Customer Tracking Systems

**Stage 3 (Staging/Preproduction)**

Full integration with Purchasing Fulfillment and Customer Tracking Systems

Full credit card processing integration

Full product data catalog



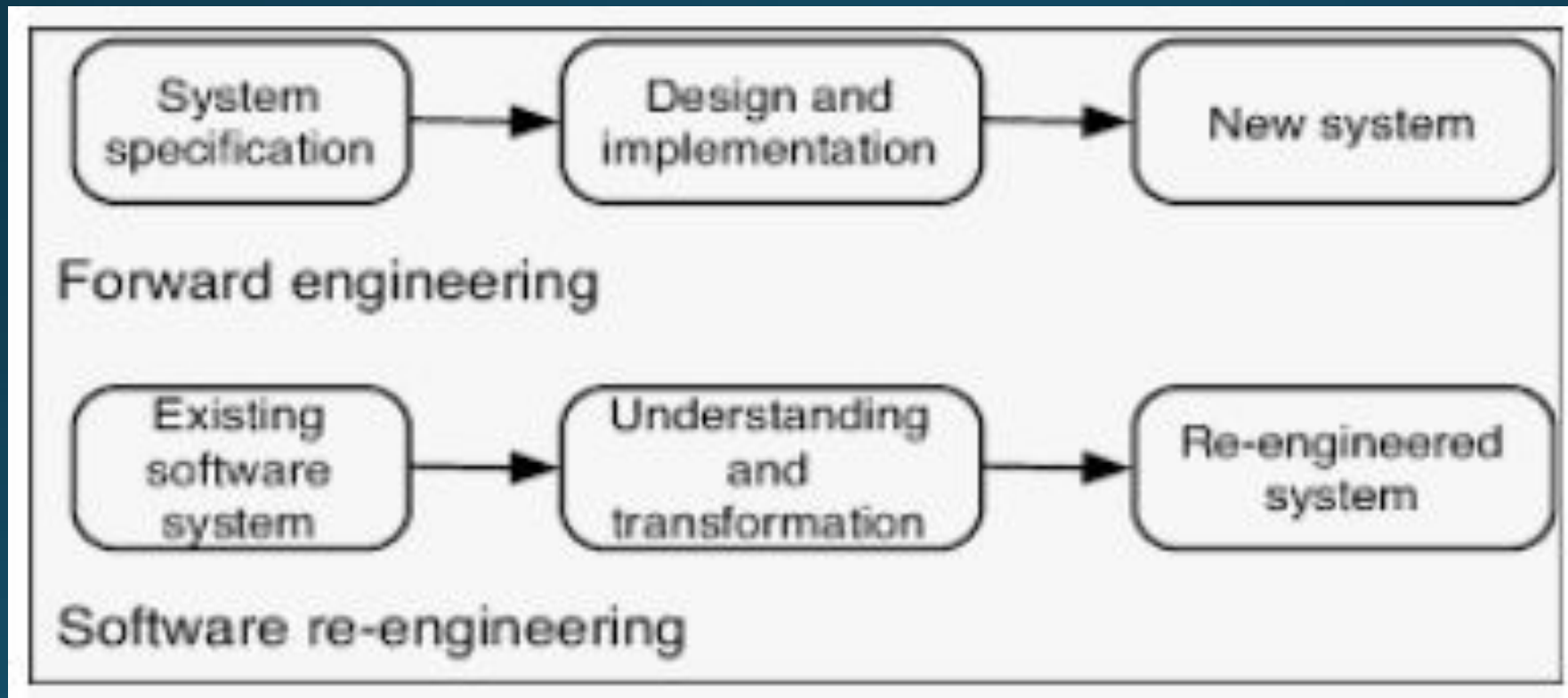
# Radical methods for requirement gathering

- In some organizations, though, management is looking for new ways to perform current tasks.
  - These ways may be radically different from how things are done now, but the payoffs may be enormous.
  - E.g. Fewer people needed to do the same work; better customer relationships, or efficient processes
- The overall process by which current methods are replaced with radically new methods is referred to as Business process Reengineering (BPR).



- **Reengineering:**

- It is a process of software development which is done to improve the maintainability of a software system.
- Re-engineering is the examination and alteration of a system to reconstitute it in a new form.






# BPR (Business Process Reengineering)


- The idea behind BPR is not just to improve each business process but, in a systems modeling sense, to reorganize the complete flow of data in major sections of an organization to eliminate unnecessary steps, combine previously separate steps, and become more responsive to future changes.
- BPR has been described as a radical new approach for business improvement, with the potential to achieve dramatic improvement in business performance.
- BPR concepts are actively applied in both corporate strategic planning and information systems planning as a way to improve business processes radically

# **Steps in Business Process Reengineering**

**Define Objectives and Framework**



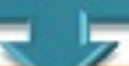
**Identify Customer Needs**



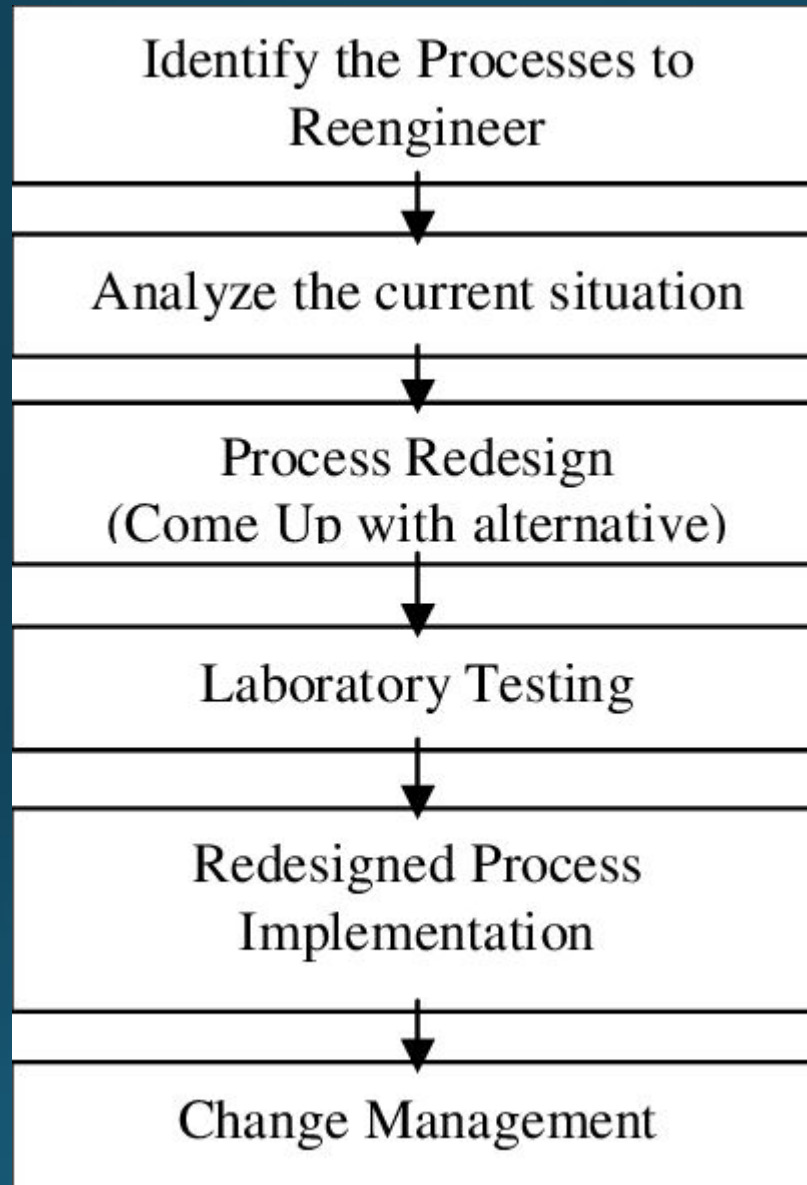
**Study the Existing process**



**Formulate a Redesign Plan**



**Implement the Redesign Plan**





# Structuring the System Process Requirements

# Background

- The two parts to the analysis phase are determining requirements and structuring requirements.
- As part of structuring, team members organize the information into a meaningful representation of the existing information system and of the requirements desired in a replacement system.
- Analysts use a tool called process modeling to structure information based on how data flow through an information system, the relationships among the data flows, and how data come to be stored at specific locations.
- These process model diagrams concentrate on the movement of data between processes.

# What is process modeling?

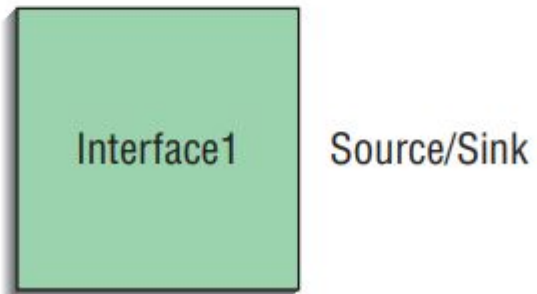
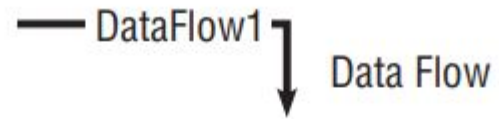
- Graphical representation of the processes or actions that capture, manipulate, store, and distribute data between a system and its environment and among components within a system.
- Concepts:
  - It detects how the system is operating.
  - Focus on: information flow
  - Illustrates the process flow without going into the detail about physical implementation.
- structured analysis technique used to increase software development productivity.
- A common form of a process model is a data-flow diagram (DFD).

# Data Flow Diagram (DFD)

- A graphical tool that allows analysts (and users) to show the flow of data in an information system.
- A data-flow diagram is a graphic that illustrates the movement of data between external entities and the processes and data stores within a system.



# Data Flow Diagram Symbols



Data Flow: Data that are in motion from one system component to another

Data Store: A data at rest (external file or database)

Process: work or actions performed on data so that they are transformed, stored, or distributed.

Source/sink: the origin and/or destination of the data.

# General Description of DFD components

## 1. Data Flow

- Data moving as a unit from one place in a system to another
- A data flow can be composed of many individual pieces of data that are generated at the same time and that flow together to common destinations.
- E.g. result of query to database, contents of a document, data on computer display system

## 2. Data Store

- A data store may represent one of many different physical locations for data, including a file folder, one or more computer-based file(s), or a notebook.
- It focuses on data movement and handling within the system, but not on physical device configuration.
- E.g.: A data store might contain data about customers, students, customer orders, or supplier invoices.

# General Description of DFD Components

## 3. Process

- The work or actions performed on data so that they are transformed, stored, or distributed.
- Process can be performed manually or by the use of computers

## 4. Entity (Source/Sink)

- External entities that lie outside the system.
- Before processing, data need to enter the system via some interface.
  - Once processed, data or information leave the system and go to some other place.
- a source or sink is a “black box”.
  - What a source or sink does with information or how it operates is not considered.

# Steps for Developing DFDs

## 1. Develop a context diagram.

- The boundary or scope of a system, and the system's relationship to its environment, is represented by a data-flow diagram called a context diagram.
- It gives the overview of the system.
- context diagram contains only one process, multiple data flows, sources and sinks, but no data stores.
- A single process (labeled “0” [zero]) represents the entire system, while the sources and sinks represent its environmental boundaries.

# Example: Context Diagram for Online shopping system



**Figure: Context Diagram of Online Shopping System**

# Developing DFDs

## 2. Break down system into individual sub-systems.

- After drawing the context diagram, analyst needs to think about which processes are represented by the single process in CFD.
- The system is then broken down based on various processes or functions carried out by individual modules of system.
  - E.g. of these functions can be data capture system, data store management system, data production and distribution module, etc.

# Relationship between Context diagram and DFDs:

Traditional method:

Context diagram

DFD : level-0, level-1, ...

Modern method:

Context diagram/DFD level-0

DFD : level-1, level-2, ...

# DFD-Levels

- Levels of DFD:
  - DFD level-0
  - DFD level-1
  - DFD level-2, beyond.
- More the level, more will be the details.



# A level-1 DFD Example

Example is shared in pdf format.

# Developing DFDs

## 3. Keep Breaking down bigger system into smaller individual sub-systems.

- Data flows generated by various processes can be analyzed and further broken down into smaller parts.
- If the data flows go from process to external entities(source/sink) then their further breakdown is not required.
- Breakdown of level-0 DFD produce more detailed description of level-0 processes.
  - This breakdown is then labeled as level-1 DFD
- Breakdown process goes on till each process and sub-process are explained in detail.
  - For academic level, only DFD level-1 and below are used.

# DFD level-2 diagram

Example is shared in pdf format.

# Rules while making DFDs

## A. Process related rules

1. No process can have only outputs.
  - The only object that has output data flow only, is a source.
2. No process can have only inputs.
  - The only object that has input data flow only, is a sink.
3. A process has generally verb-phrase label [E.g. “produce sales report”] or a system name with functional description [E.g. Sales Report Management System].

## B. Data Store related rules

1. Data cannot directly move from one data store to another.
  - Data must be moved by a process.
2. A data store has generally noun-phrase label [E.g. “Customer info”].
3. Errors in data store:
  - Black whole error
    - Only input but no output in data store
  - Magical error
    - Output without any input in data store

## C. Source/Sink related rules

1. Data cannot move directly from a source to a sink i.e. one entity to another entity.
  - Data must be moved by a process if the data are of any concern to our system.
2. A source/sink has a noun-phrase label. [E.g. “Garage”, “Student”].

## D. Data Flow related rules

1. A data flow has only one direction of flow between symbols.
2. A fork in a data flow means that exactly the same data go from a common location to two or more different processes, data stores, or sources/sinks.
3. A join in a data flow means that exactly the same data come from any of two or more different processes, data stores, or sources/sinks to a common location.
4. A data flow to a data store means update and a data flow from a data store means retrieve or use.
5. A data flow has a noun-phrase label [E.g. “Daily sales records”].

# Balancing DFDs

- When we decompose a DFD from one level to the next, a conservation principle is at work.
- We must conserve inputs and outputs to a process at the next level of decomposition.
  - E.g. Process 1.0 that appears in a level-0 diagram, must have the same inputs and outputs when decomposed into a level-1 diagram.
- If a combined data flow is taken as input to a process in a level, the data flow can be broken into individual units as the process breaks down in next level.



# What features explored in DFD

## 1. Completeness

- All required components must be depicted in DFD

## 2. Consistency

- System shown at one level of DFD should correspond well with system shown at other levels.

## 3. Timing considerations

- DFDs do not do a good job of representing time.

## 4. Iterative nature of drawing DFDs

- drawing the same diagram over and over again, in an iterative fashion for a good approximation of the system.

## 5. Drawing primitive DFDs

- Analyst to make sure when to stop decomposing processes.

# Use of DFDs for analysis

- Because DFDs depict how system responds to data flow in different level, it can be used to analyze gap between how system interacts on one level and another.
- DFDs help to analyze any redundant data flows, unused data, or same data originating from different sources(processes).
  - This helps to understand how the organization is performing its activities efficiently.
- Comparison of 2 similar DFDs help to determine what features can be added removed from existing system to make a better, more effective system.

# Assignment:

- What are the advantages and limitations of DFD.