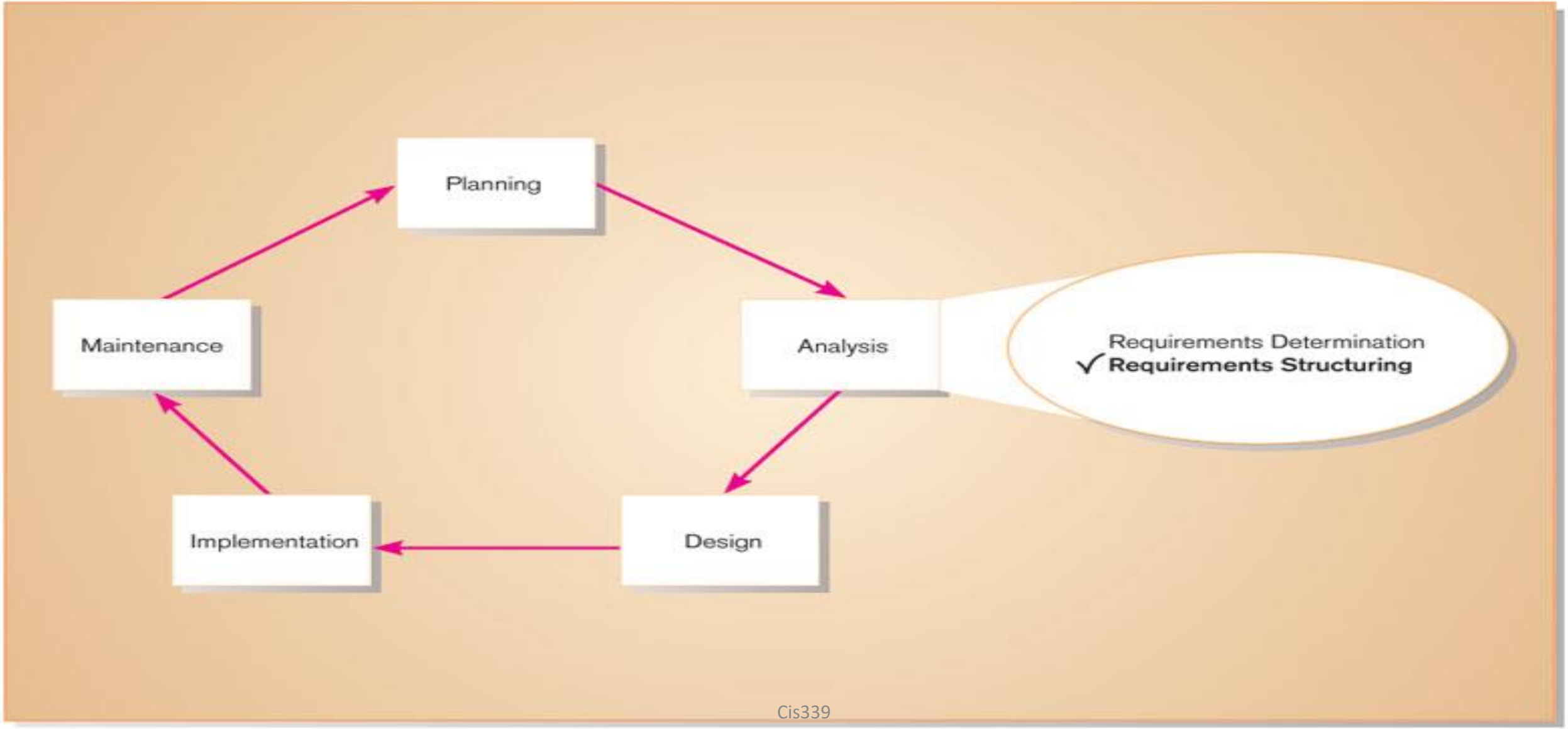
Systems Analysis and Design

Chapter 5
System Analysis

5.2 Structuring System Requirements:
a. Process Modeling

Process Modeling

Figure 7-1 Systems development life cycle with the analysis phase highlighted



Process Modeling

- Graphically represent the processes that capture, manipulate, store and distribute data between a system and its environment and among system components
- Utilize information gathered during requirements determination and organized it into meaningful representation.

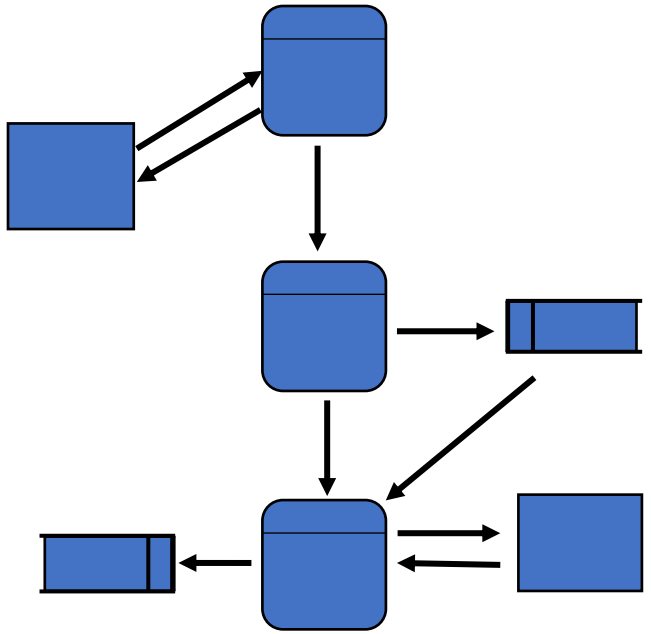
- **Data flow diagrams (DFD)**
  - Graphically illustrate movement of data between external entities and the processes and data stores within a system

Deliverables for Process Modeling

- Deliverables and Outcomes
  - Set of coherent, interrelated data flow diagrams, such diagrams are:
    - Context data flow diagram (DFD)
  - Scope of system
    - DFDs of current system
  - Enables analysts to understand current system
  - it shows what data processing functions performed by the current system
    - DFDs of new logical system
  - Show data flows, structure and functional requirements of new system
    - Description of each DFD component, entries for all of the objects included in all diagrams (in data dictionary or CASE repository)

Data Flow Diagram (DFD)

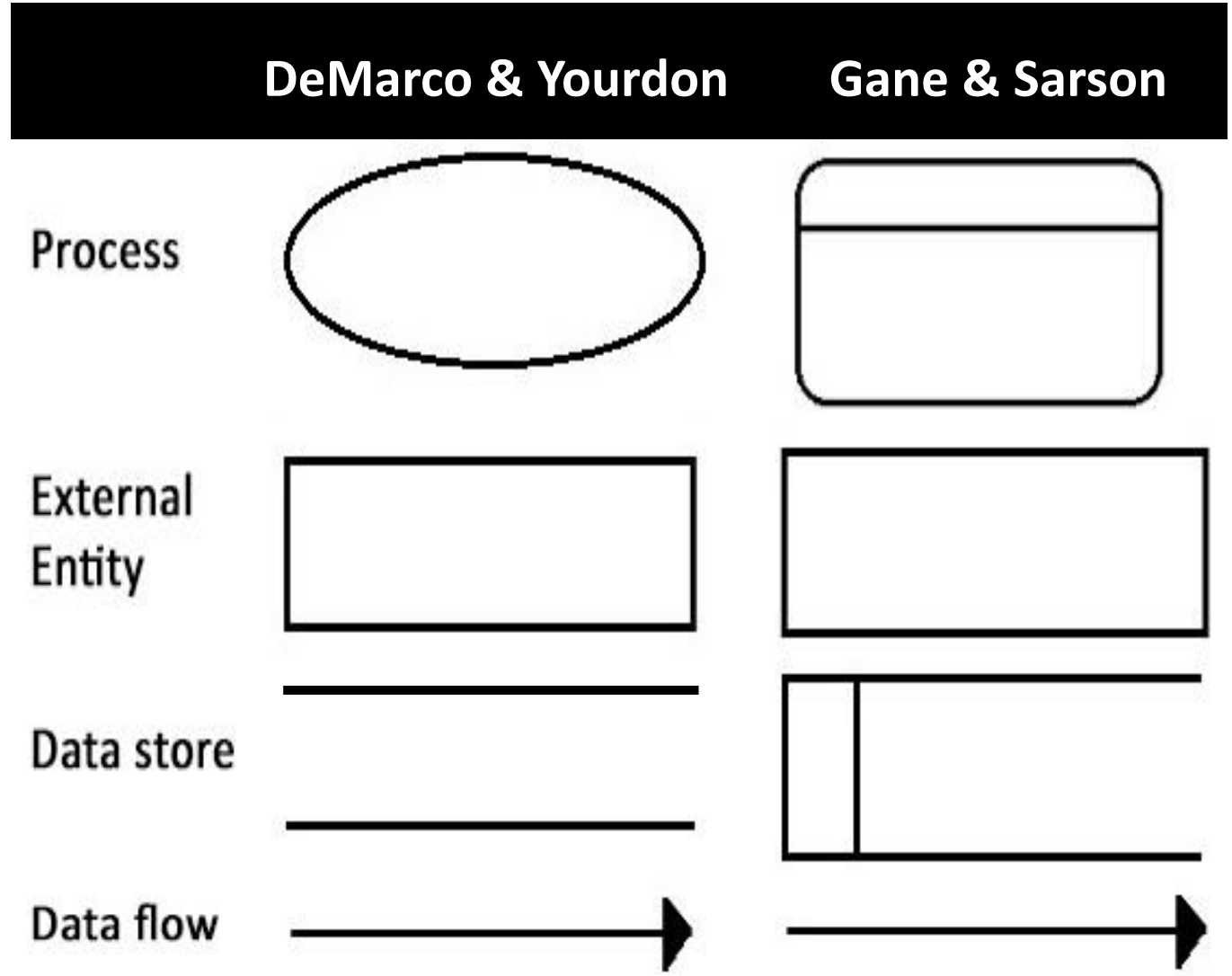
- Data Flow Diagram (DFD) is a tool that describes the flow of data through a system and the work or processing performed by the system.
- It shows the flow of data from external entities into the system, how the data moves from one process to another and its logical storage.



Symbols used in DFD

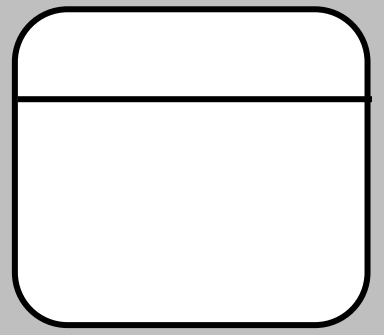
- Data Flow Diagram consists of four notations
  - Process
  - External Entity (Source / Sink)
  - Data Store
  - Data flow

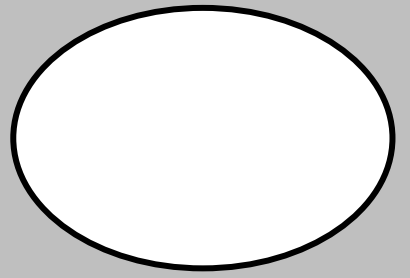
- Two different standard sets can be used
    - DeMarco and Yourdon
    - Gane and Sarson



Components of DFD

- **Process**
  - Process is a procedure that manipulates the data and its flow by taking incoming data, changing it, and producing an output with it.
  - Depicts work or action performed on data so that they are transformed, stored or distributed
  - Number of process as well as name are recorded
  - A process is represented by a rounded rectangle (Gane & Sarson) or an oval (DeMarco & Yourdon).



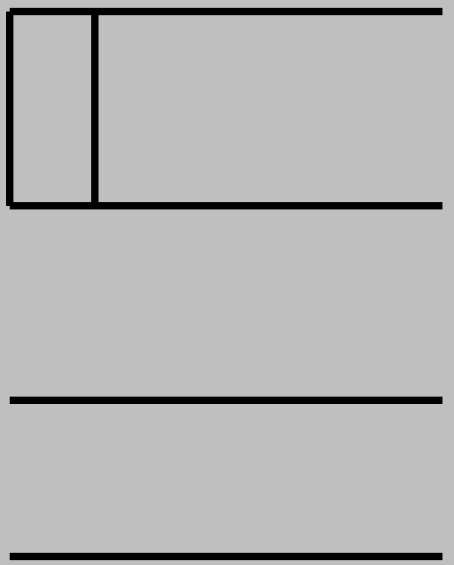


Components of DFD

- **External Entity or Source / Sink**
  - An external entity, which is also known as terminator, source, sink, or actor, is an outside system or process that sends or receives data to and from the diagrammed system.
  - External entities are either the sources (origin) or destinations of information, so they’re usually placed on the diagram’s edges.
  - An external entity is represented by a square or a rectangle

Components of DFD

- **Data Store**
  - A data store represents files or repositories that hold information for later use
  - It depicts data at rest
  - It may represent data in database, computer -file, notebook etc.
  - It does not generate any operations but simply holds data for later access.
  - A data store is represented by an open ended box (Gane & Sarson) or parallel lines (DeMarco & Yourdon).
  - The name of the store as well as the number are recorded in between lines



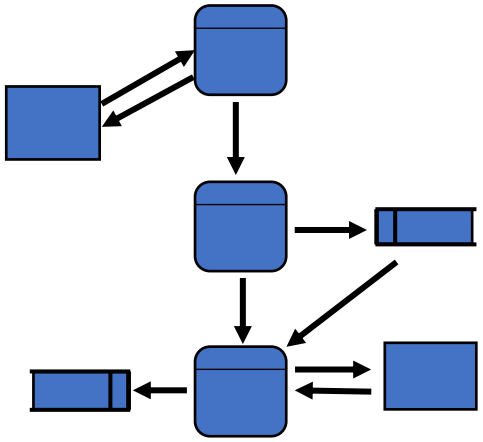
Components of DFD

- **Data flow**
  - Data flow is the path the system's information takes from external entities through processes
  - Depicts data that are in motion and moving as a unit from one place to another in the system.
  - represented by arrows.
  - Labels are shown along with the arrows to represent the data

Data Flow Diagramming Definitions

- **Context Diagram (Context Level DFD)**
  - A data flow diagram (DFD) of the scope of an organizational system that shows the system boundaries, external entities that interact with the system and the major information flows between the entities and the system
  - Contains a single process which represents the whole system
  - Doesn't include the data stores as they are internal to the system

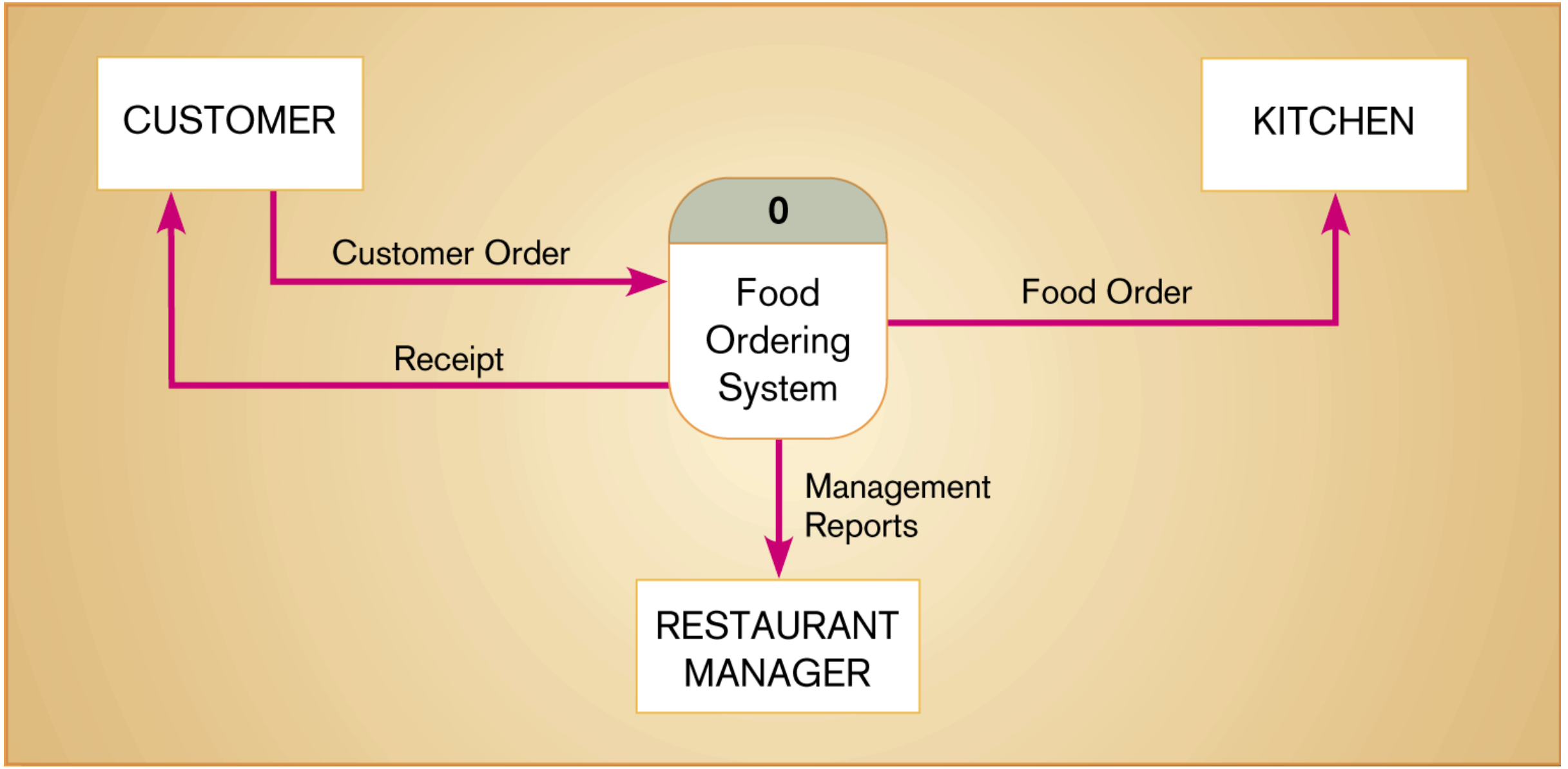
- **Level-0 DFD**
  - A data flow diagram (DFD) that represents a system's major processes, data flows and data stores at a high level of detail



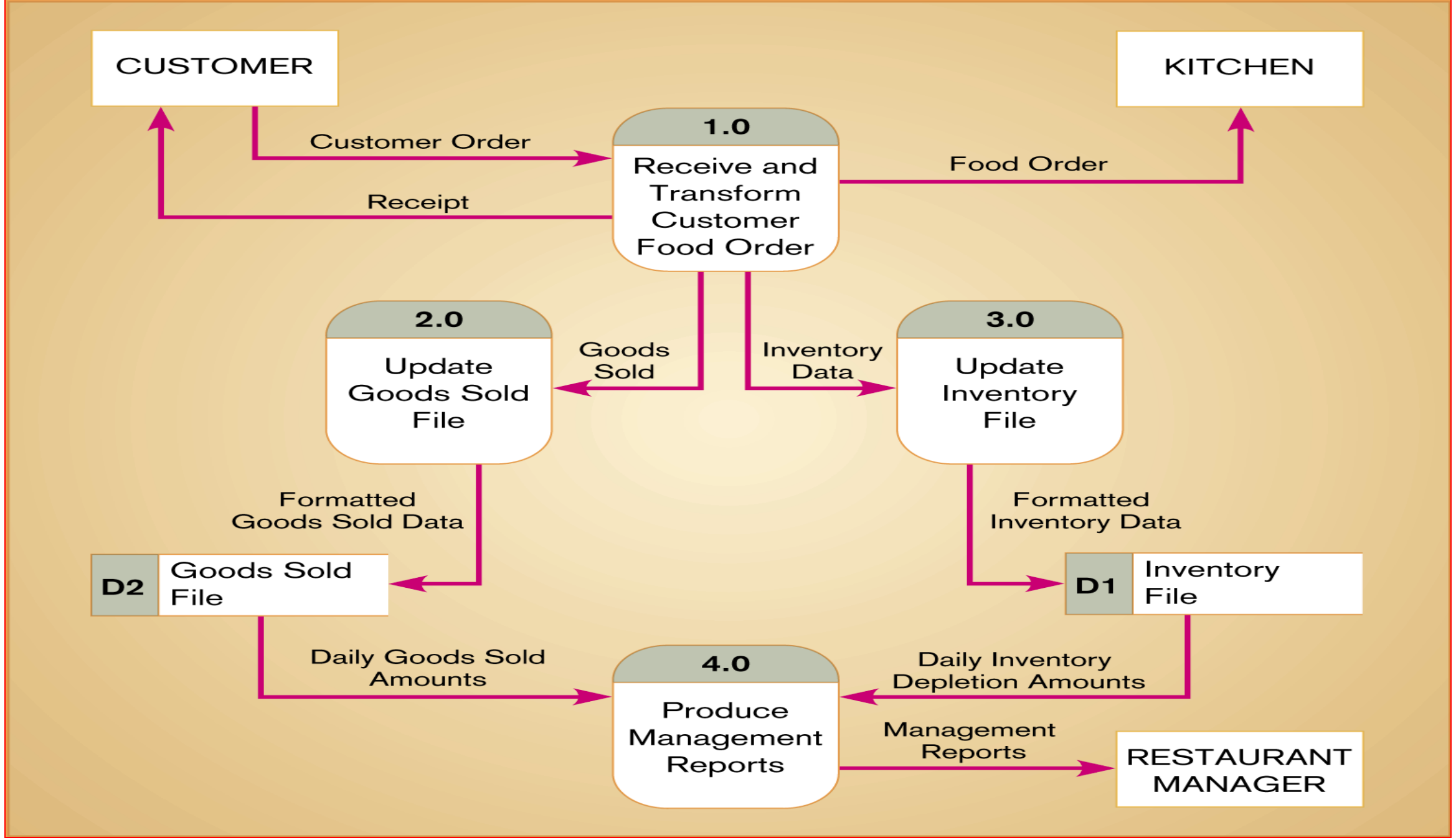
Developing DFDs: An Example

An automated food ordering system

- Context Diagram contains one process, no data stores, four data flows, and three sources/sinks
- Next step is to expand the context diagram to show the breakdown of processes level-0 DFD



Context diagram of food ordering system



Level-0 DFD of food ordering system

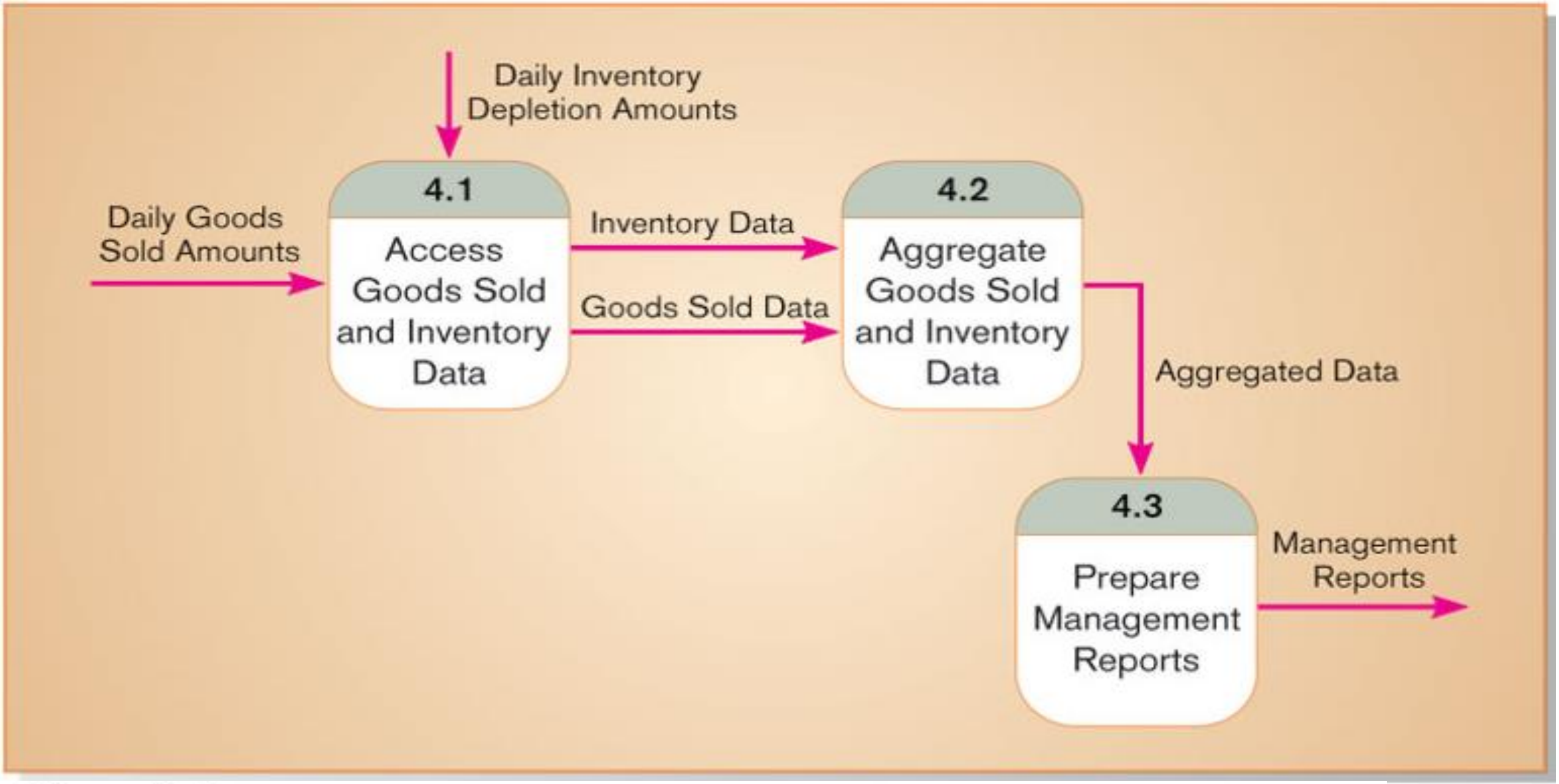
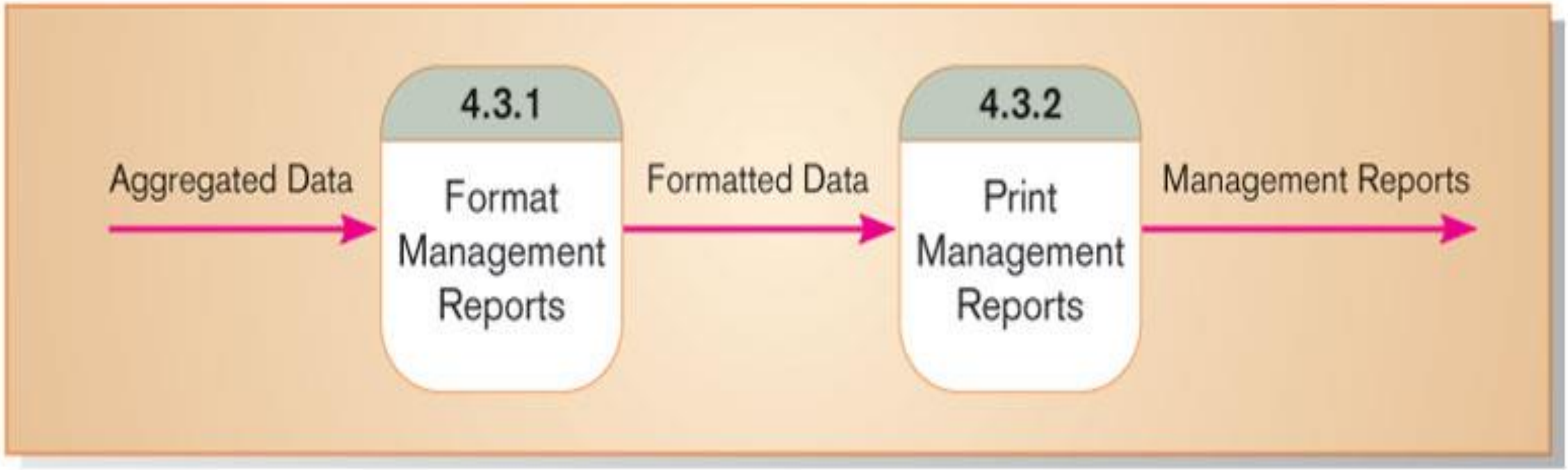


Figure 7-8 Level-1 diagram showing the decomposition of Process 4.0 from the level-0 diagram for Hoosier Burger's food ordering system

Figure 7-9 Level-2 diagram showing the decomposition of Process 4.3 from the level-1 diagram for Process 4.0 for Hoosier Burger's food ordering system



Data Flow Diagramming Rules

- **Basic rules that apply to all DFDs**
  - Inputs to a process are always different than outputs
  - Objects always have a unique name
    - In order to keep the diagram uncluttered, you can repeat data stores and sources/sinks on a diagram
    - Every process has a unique name.

Data Flow Diagramming Rules

- **Process**
  - No process can have only outputs (a miracle)
  - No process can have only inputs (black hole)
  - A process has a verb phrase label
- **Data Store**
  - Data cannot be moved directly from one store to another
  - Data cannot move directly from an outside source to a data store
  - Data cannot move directly from a data store to a data sink
  - Data store has a noun phrase label

Data Flow Diagramming Rules

- **Source/Sink**
  - Data cannot move directly from a source to a sink
  - A source/sink has a noun phrase label

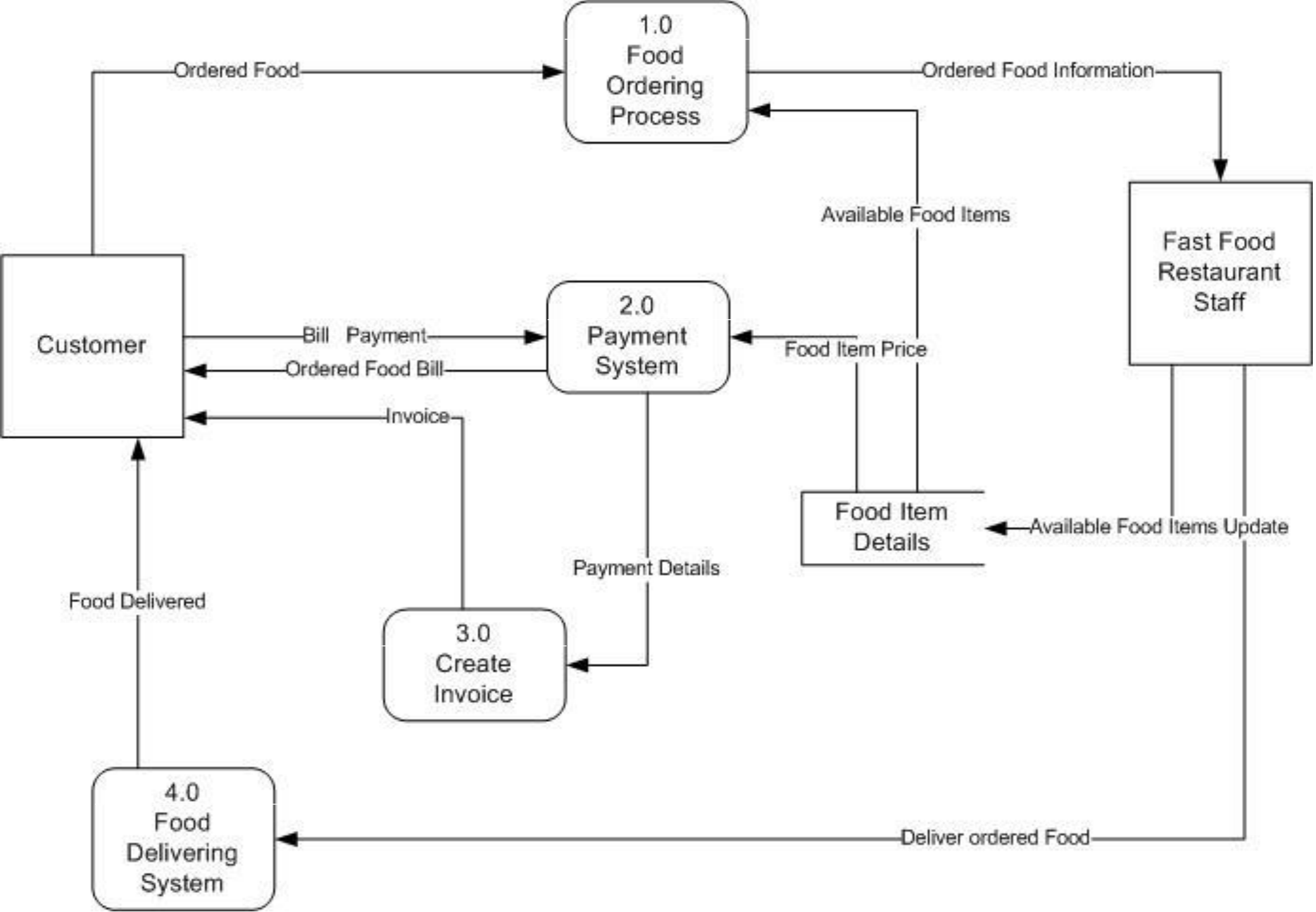
- **Data Flow**
  - A data flow has only one direction of flow between symbols
  - A fork means that exactly the same data goes from a common location to two or more processes, data stores or sources/sinks

Data Flow Diagramming Rules

Data Flow (Continued)

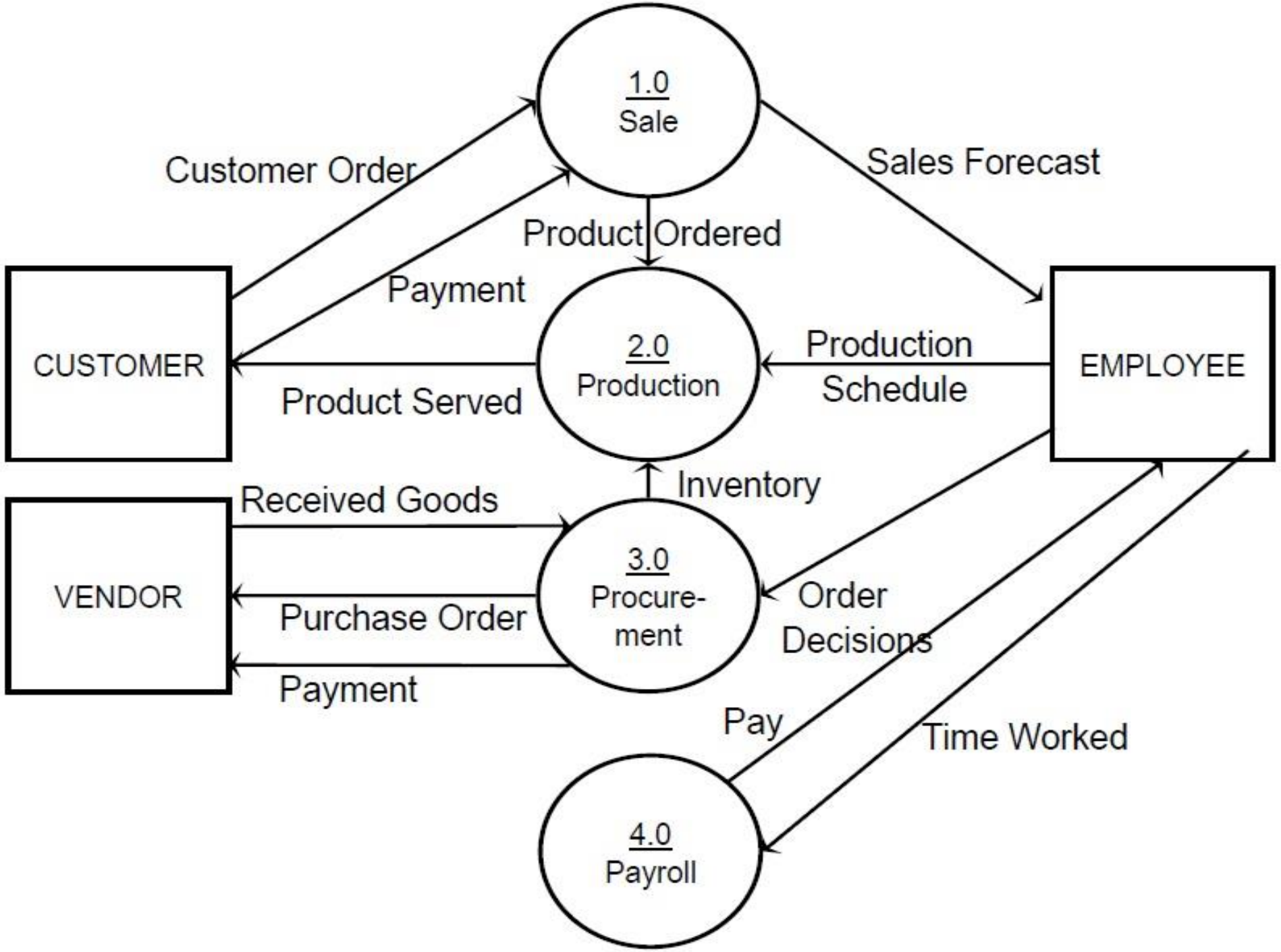
- A join means that exactly the same data comes from any two or more different processes, data stores or sources/sinks to a common location
- A data flow cannot go directly back to the same process it leaves
- A data flow to a data store means update
- A data flow from a data store means retrieve or use
- A data flow has a noun phrase label

Some other Examples of Data Flow Diagram (DFD)



Example 2

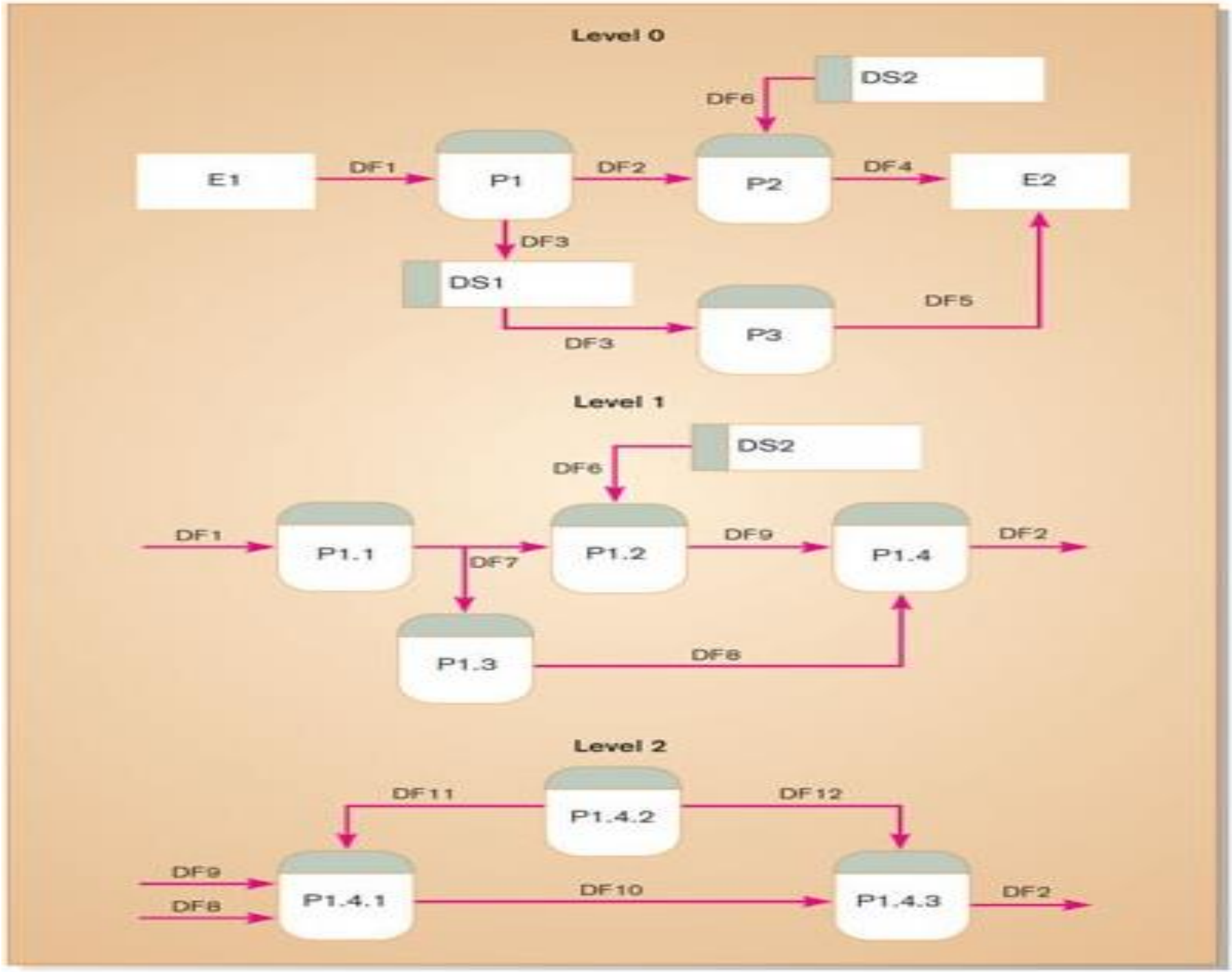
Data Flow Diagram of Restaurant (Using Gane & Sarson notations)



Example 3

Data Flow Diagram of Lemonade System (Using DeMarco and Yourdon notation)

What is wrong of the following, if any?



Decomposition of DFDs

- Functional decomposition, Is an iterative process of breaking a system description down into finer and finer detail.
  - Creates a set of charts in which one process on a given chart is explained in greater detail on another chart.
  - Continues until no sub-process can logically be broken down any further.
  - Lowest level is called a primitive DFD

- Level-N Diagrams
  - A DFD that is the result of *n* nested decompositions of a series of sub processes from a process on a level-0 diagram

Balancing DFDs

- **Conservation Principle**: conserve inputs and outputs to a process at the next level of decomposition.
- **Balancing:** conservation of inputs and outputs to a data flow diagram process when that process is decomposed to a lower level.
- Balanced means:
  - Number of inputs to lower level DFD equals number of inputs to associated process of higher-level DFD
  - Number of outputs to lower level DFD equals number of outputs to associated process of higher-level DFD

Balancing DFDs

Figure 7-10a An unbalanced set of data flow diagrams - Context diagram

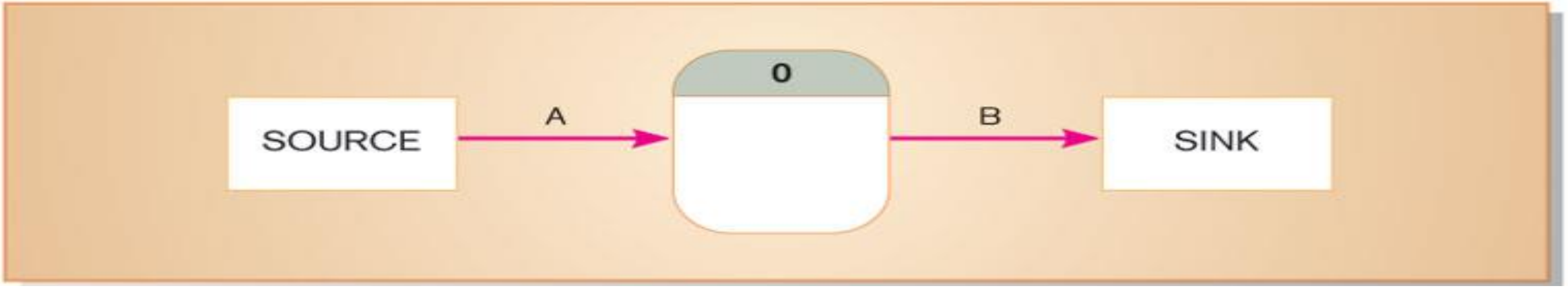
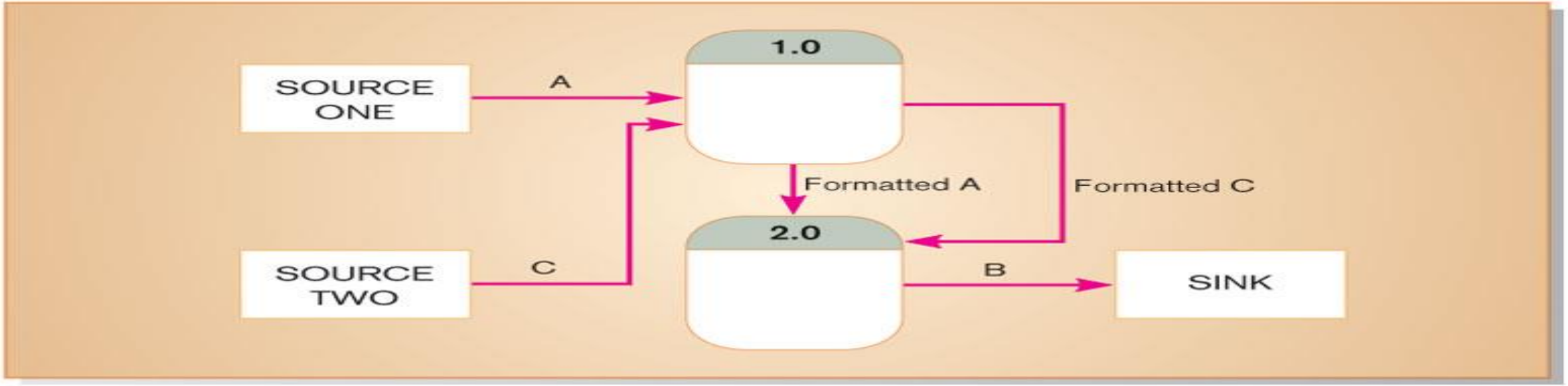


Figure 7-10b An unbalanced set of data flow diagrams - Level-0 diagram



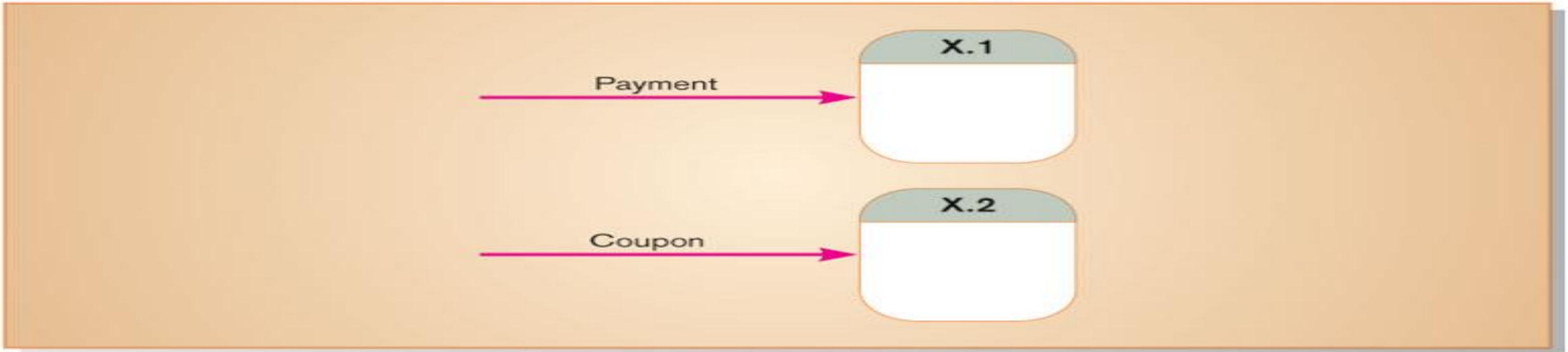
Balancing DFDs

- **Data flow splitting** is when a composite data flow at a higher level is split and different parts go to different processes in the lower level DFD.
- The DFD remains balanced because the same data is involved, but split into two parts.

Balancing DFDs

Figure 7-11a Example of data flow splitting - Composite data flow

Figure 7-11b Example of data flow splitting - Disaggregated data flows



Four Different Types of DFDS used in the systems development process:

- **Current Physical**
  - Process labels identify technology (people or systems) used to process the data.
  - Data flows and data stores identify actual name of the physical media.
- **Current Logical**
  - Physical aspects of system are removed as much as possible.
  - Current system is reduced to data and processes that transform them.

Four Different Types of DFDS

Four Different Types of DFDS used in the systems development process:

- **Current Physical**
  - Process labels identify technology (people or systems) used to process the data.
  - Data flows and data stores identify actual name of the physical media.
- **Current Logical**
  - Physical aspects of system are removed as much as possible.
  - Current system is reduced to data and processes that transform them.

- **New Logical**
  - Includes additional functions.
  - Obsolete functions are removed.
  - Inefficient data flows are reorganized.
- **New Physical**
  - Represents the physical implementation of the new system.

Guidelines for Drawing DFDs

- **Completeness**
  - **DFD must include all components necessary for system**
  - **Each component must be fully described in the project dictionary or CASE repository**
- **Consistency**
  - **The extent to which information contained on one level of a set of nested DFDs is also included on other levels**
- **Timing**
  - **Time is not represented well on DFDs**
  - **Best to draw DFDs as if the system has never started and will never stop.**
- **Iterative Development**
  - **Analyst should expect to redraw diagram several times before reaching the closest approximation to the system being modeled (How many?)**
- **Primitive DFDs**
  - **Lowest logical level of decomposition**
  - **Decision has to be made when to stop decomposition**

Guidelines for Drawing DFDs

- **Rules for stopping decomposition (continued)**
  - **When every data flow does not need to be split further to show that data are handled in various ways**
  - **When you believe that you have shown each business form or transaction, on-line display and report as a single data flow**
  - **When you believe that there is a separate process for each choice on all lowest-level menu options**

Using DFDs as Analysis Tools

- **Gap Analysis** is the process of discovering discrepancies between two or more sets of data flow diagrams or discrepancies within a single DFD.
- **Inefficiencies in a system can often be identified through DFDs, such inefficiency is a violation of DFD drawing rules, such an obsolete data are captured but not used any where in the system .**