Chapter II

Modeling Tools for System Analyst

2.1 System Analyst

2.1.1 Introduction

System analyst is a person who can start from a complex problem, break it down logically, and identify the reasonable solutions

Installing a system without proper planning leads to great user dissatisfaction and frequently causes the system to fall into disuse.

Systems analysis and design, as performed by systems analysts, seeks to understand what humans need to analyze data input or data flow systematically, process or transform data, store data, and output information in the context of a particular business.

User involvement throughout the systems project is critical to the successful
development of computerized information systems.

2.1.2 Roles

The analyst plays many roles, sometimes balancing several at the same Time. The three primary roles of the systems analyst are consultant, supporting Expert, and agent of change.

Consultants

The systems analyst may be hired to address information systems issues within a business.

- Advantage: Outside consultants can bring with them fresh perspective
- Disadvantage: the true organizational culture can never be known to an outsider.

Supporting experts

- **T**he analyst draws on professional expertise concerning computer hardware and software and their uses in the business.
- He serves as a resource for those who are managing a systems project.
- He supports business decision making.

Change agents

- Acting as an agent of change.
- Develops a plan for change.
- Works with others in facilitating that change.

2.1.3 Skills

2.1.3.1 Technical Skills for Systems Analysis

- Constant re-education is necessary as technology changes rapidly
- Activities to keep skills up-to-date

  - Trade publications
  - Professional societies
  - Attend classes or teach at a local college
  - Attend courses sponsored by organization
  - Conferences and trade shows
  - Browse Websites
  - Participate in new groups and conferences

- Understanding of a wide variety of technologies is required
  - Microcomputers, workstations, minicomputers and mainframe computers
  - Programming languages
  - Operating systems
  - Database and file management systems
  - Data communication standards
  - Systems development tools and environments
  - Web development languages and tools
  - Decision support system generators

2.1.3.2 Management Skills for Systems Analysis

- Four categories
  - Resource Management
  - Project Management
  - Risk Management
  - Change Management

Resource Management

- Systems analyst needs to know how to get the most out of the resources of an organization, including team members
- Includes the following capabilities
  - Predicting resource usage
  - Tracking resource consumption
  - Effective use of resources
  - Evaluation of resource quality
  - Securing resources from abusive use
  - Relinquishing resources when no longer needed

Project Management

- Two Goals
  - Prevent projects from coming in late
  - Prevent projects from going over budget
- Assists management in keeping track of project’s progress
- Consists of several steps
  - Decomposing project into independent tasks

  - Determining relationships between tasks
  - Assigning resources and personnel to tasks

Risk Management

- Ability to anticipate what might go wrong in a project
- Minimize risk and/or minimize damage that might result
- Placement of resources
- Prioritization of activities to achieve greatest gain

Change Management

- Ability to assist people in making transition to new system
- Ability to deal with technical issues related to change
  - Obsolescence
  - Reusability

2.1.3.3 Interpersonal Skills for Systems Analysis

- Four types of skills:
  - Communication skills
  - Working alone and with a team
  - Facilitating groups
  - Managing expectations

Communication Skills

- Effective communication helps to establish and maintain good working relationships with clients and colleagues
- Skills improve with experience
- Three types used by Systems Analyst
  - Interviewing and Listening
    - Means to gather information about a project
    - Listening to answers is just as important as asking questions
    - Effective listening leads to understanding of problem and generates additional questions
  - Questionnaires
    - Advantages:
      - Less costly than interviews
      - Results are less biased due to standardization
    - Disadvantages
      - Less effective than interviews due to lack of follow-up
  - Written and Oral Presentations
    - Used to document progress of project and communicate this to others

- Communication takes several forms:
    - Meeting agenda
    - Meeting minutes
    - Interview summaries
    - Project schedules and descriptions
    - Memoranda requesting information
    - Requests for proposals from vendors and contractors
    - Oral presentations

<u>Working alone and with a team</u>

- Working alone on aspects of project involves managing:
    - Time
    - Commitments
    - Deadlines
- Team work involves establishing standards of cooperation and coordination

<u>Facilitating groups</u>

- Involves guiding a group without being a part of the group
- Useful skill for sessions such as Joint Application Development (JAD)

<u>Managing expectations</u>

- Managing expectations is directly related to successful system implementation
- Skills for successful expectation management
    - Understanding of technology and workflows
    - Ability to communicate a realistic picture of new system to users
    - Effective education of management and users throughout systems development life cycle

2.2Context Diagram

2.2.1 Process Modeling

Process is a particular course of action intended to achieve a result.

It is a formal way of representation to show how a business system operates.

It involves graphically representing the functions, or processes, which capture, manipulate, store, and distribute data between a system and its environment, and between components within a system.

External Entities

The double square is used to show an external entity that can send data to or receive data from the systems.

External entity shows the initial source and final recipient of data and information.

External entity should be named with a noun, describing that entity.

External entities may be:

A person, such as CUSTOMER or STUDENT.

A company or organization, such as BANK or SUPPLIER.

Processes

A processes may represent either:

- A whole system when naming a context process.

  (INVENTORY CONTROL SYSTEM)
- A subsystem. (INVENTORY REPORTING SUBSYSTEM)
- Work being done, an activity by using a verb-adjective-noun format for detailed processes. (PRINT BACKORDERED REPORT, ADD INVENTORY RECORD)

A process must be given a unique identifying number indication its level in the diagram.

Data Stores

- Name with a noun, describing the data
- Data stores are usually given a unique reference number, such as D1, D2, D3.
- Include any data stored, such as:

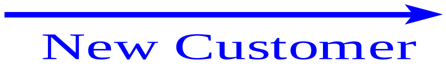
A computer file or database.

A transaction file.

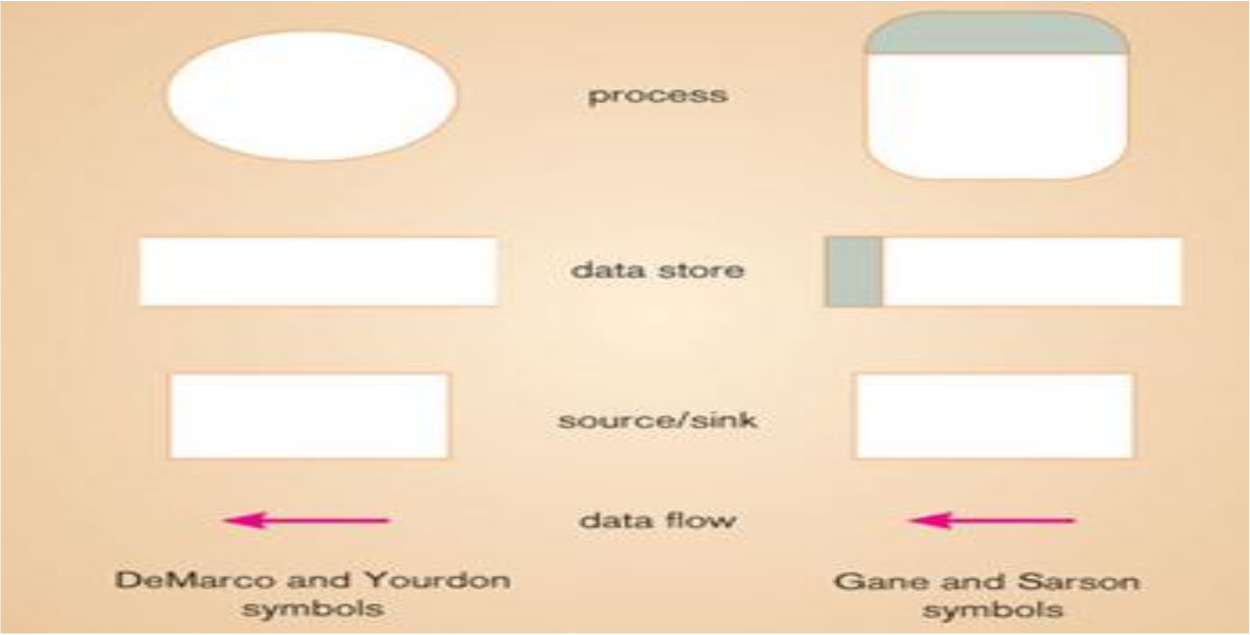
A set of tables.

Data Flow

- Data flow shows the data about a person, place, or thing that moves through the system.
- Names should be a noun that describes the data moving through the system.
- Arrowhead indicates the flow direction.



Symbols and Definitions

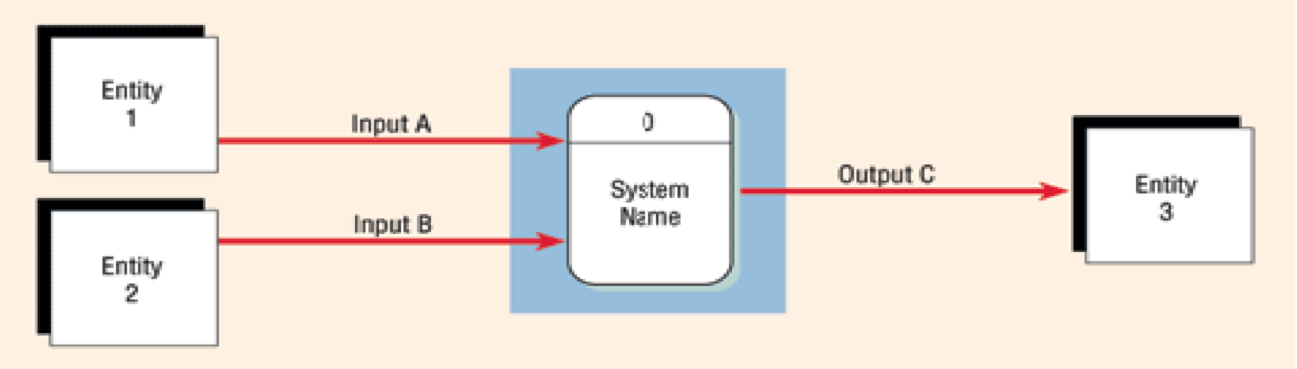


2.2 Context Diagram or Context-Level DFD

- Context diagram is an overview of an organizational system that shows:
  - The system boundaries.
  - External entities that interact with the system.
  - Major information flows between the entities and the system.

- It contains only one process, representing the entire system.
- The process is given the number zero.
- All external entities are shown on the context diagram as well as major data flow to and from them.

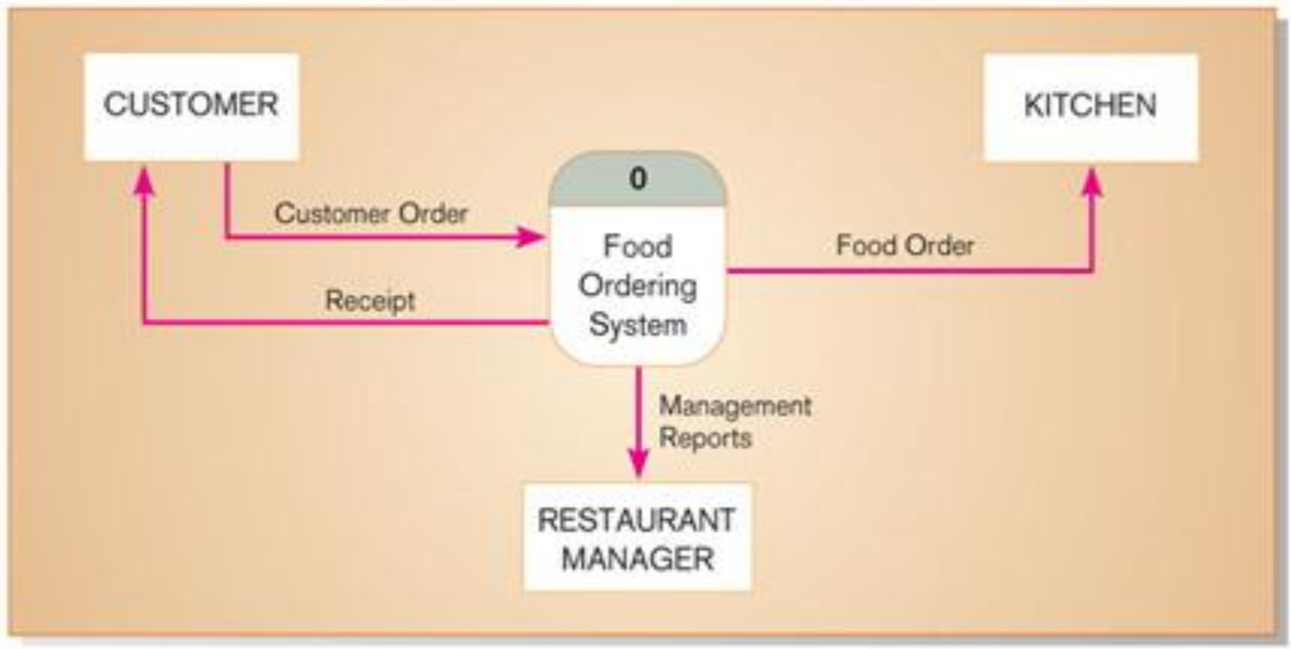
- The diagram does not contain any data stores.



For example:

Food Ordering System

- **Overview of the organizational system**



2.3Data Flow Diagram(Level 0, Level 1, Level 2)

2.3.1 Data Flow Diagram: (DFD)

It is one of the most commonly used techniques available for process modelling.

It is a picture of the movement of data between external entities, the processes and stores within a system

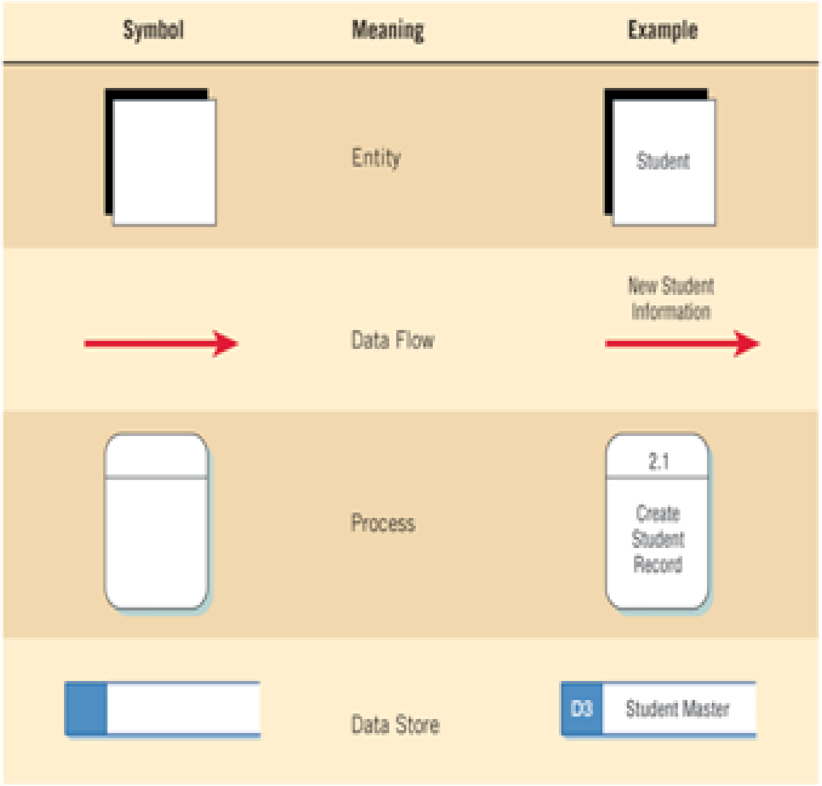
DFD Level

- Context DFD
  - Overview of the organizational system
- Level-0 DFD
  - Representation of system’s major processes at high level of abstraction
- Level-1 DFD
  - Results from decomposition of Level 0 diagram
- Level-*n* DFD
  - Results from decomposition of Level *n*-1 diagram

Basic Symbols

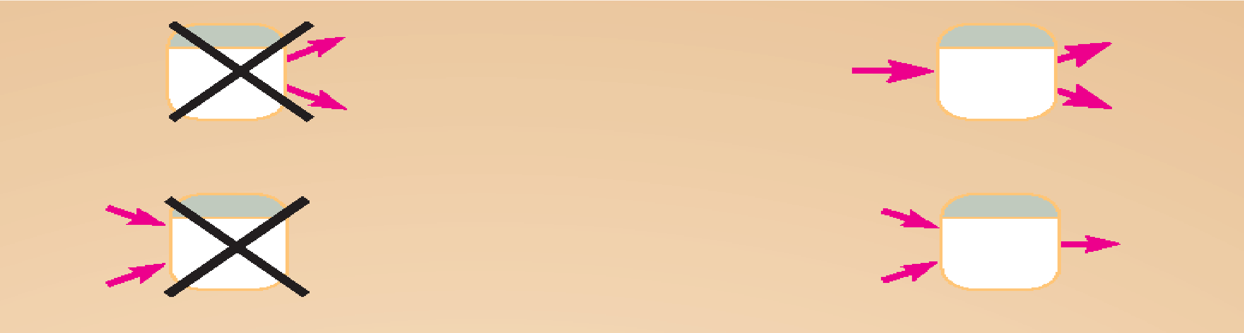
- A double square for an **external entity**-a source or destination of data.
- An arrow for **movement of data** from one point to another.
- A rectangle with rounded corners for the occurrence of transforming **process.**
- An open-ended rectangle for a **data store.**

Figure 7.1
The four basic symbols used in data flow diagrams, their meanings, and examples.

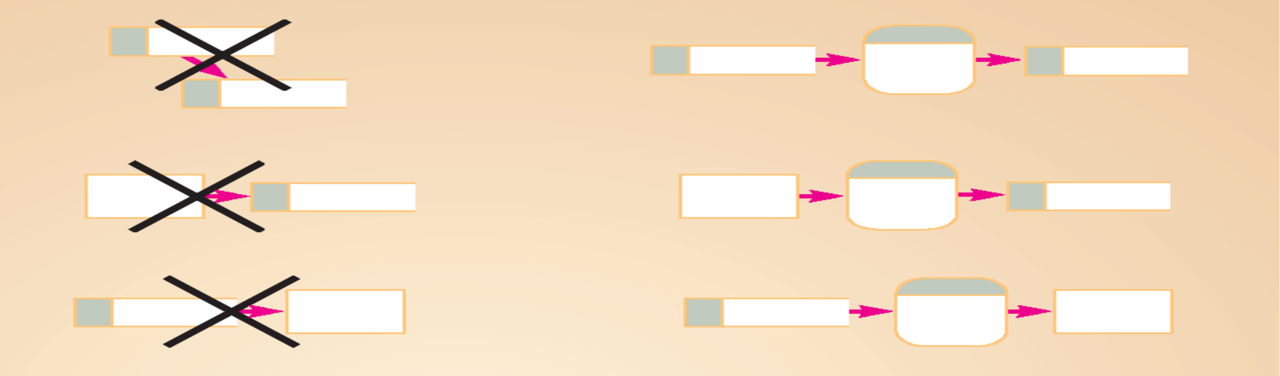


2.3.2 DFD Diagramming Rules

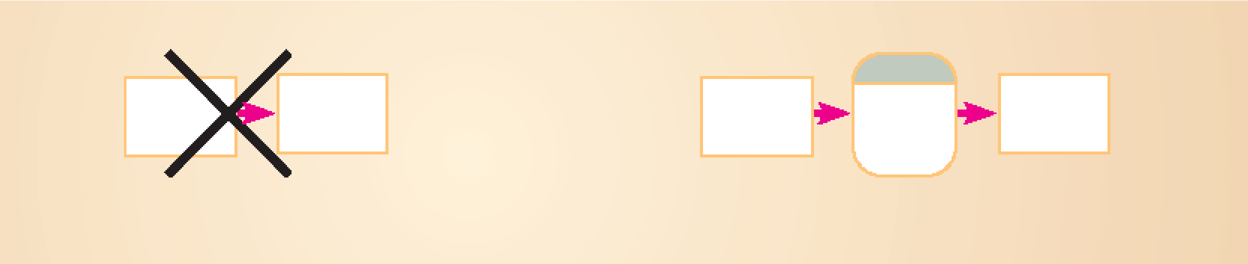
2.3.2.1 Process



2.3.2.2 Data Store

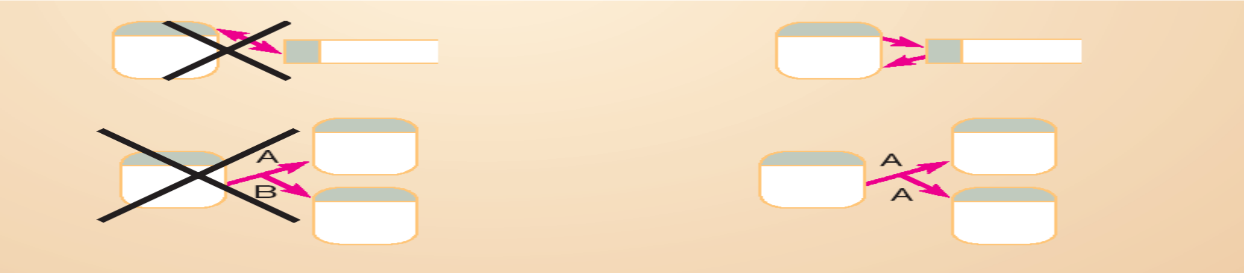


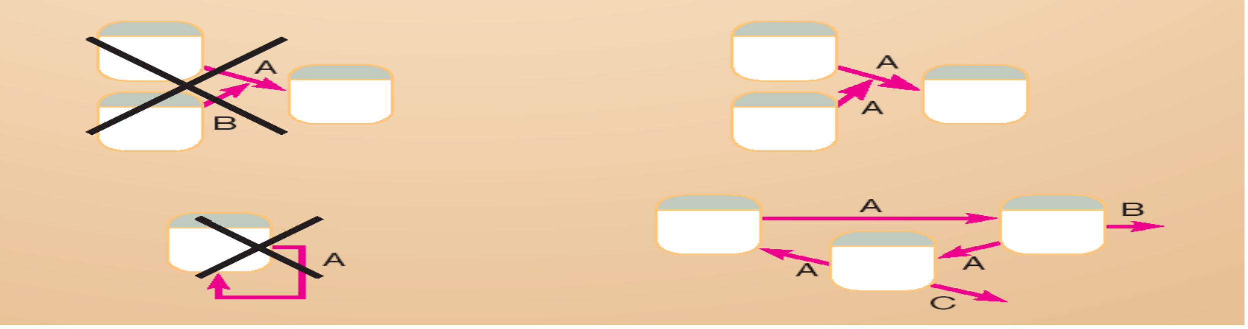
2.3.2.3 Source/Sink



2.3.2.4 Data Flow

- Data flow from a process to a data store means update (insert, delete or change).
- Data flow from a data store to a process means retrieve or use.



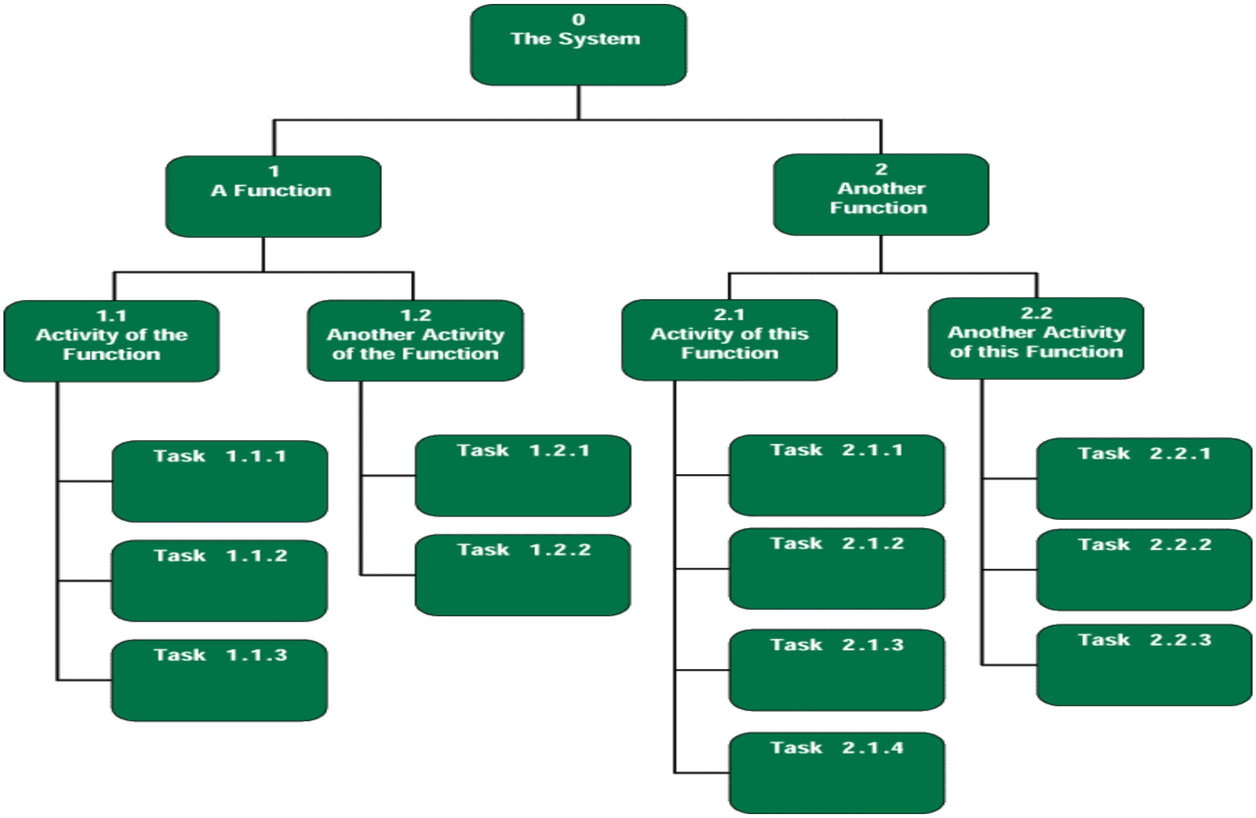


2.3.3 Concepts How to make DFD

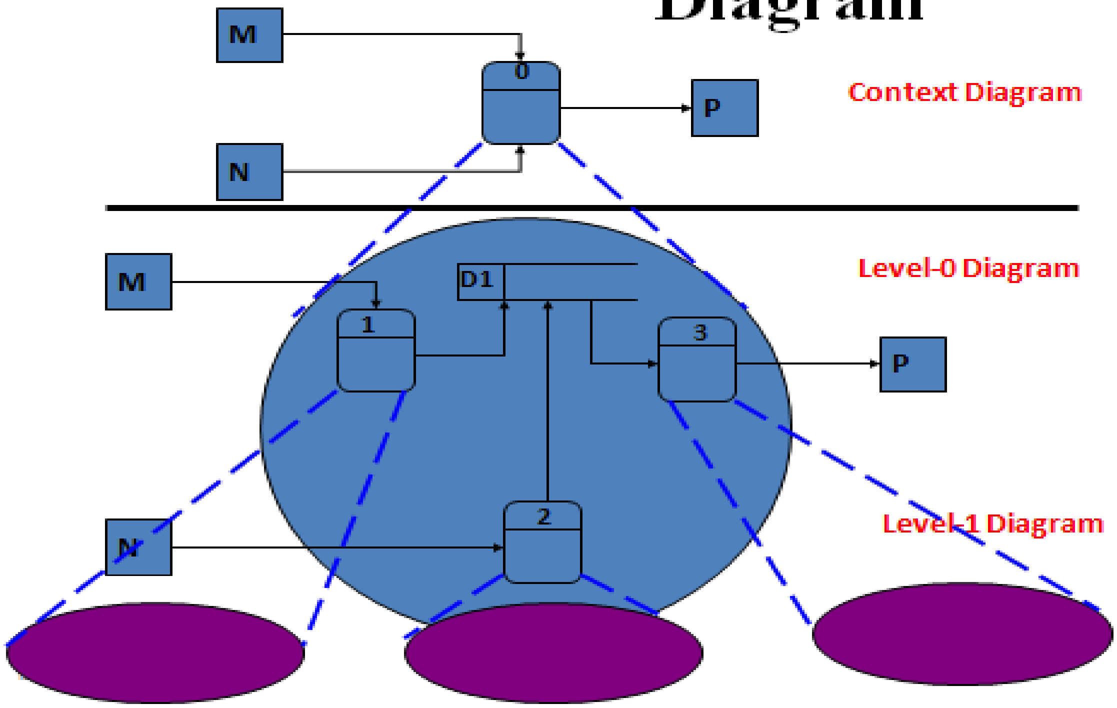
2.3.3.1 Functional Decomposition

- An iterative process of breaking a system description down into finer and finer detail.
- High-level processes described in terms of lower-level sub-processes**.**

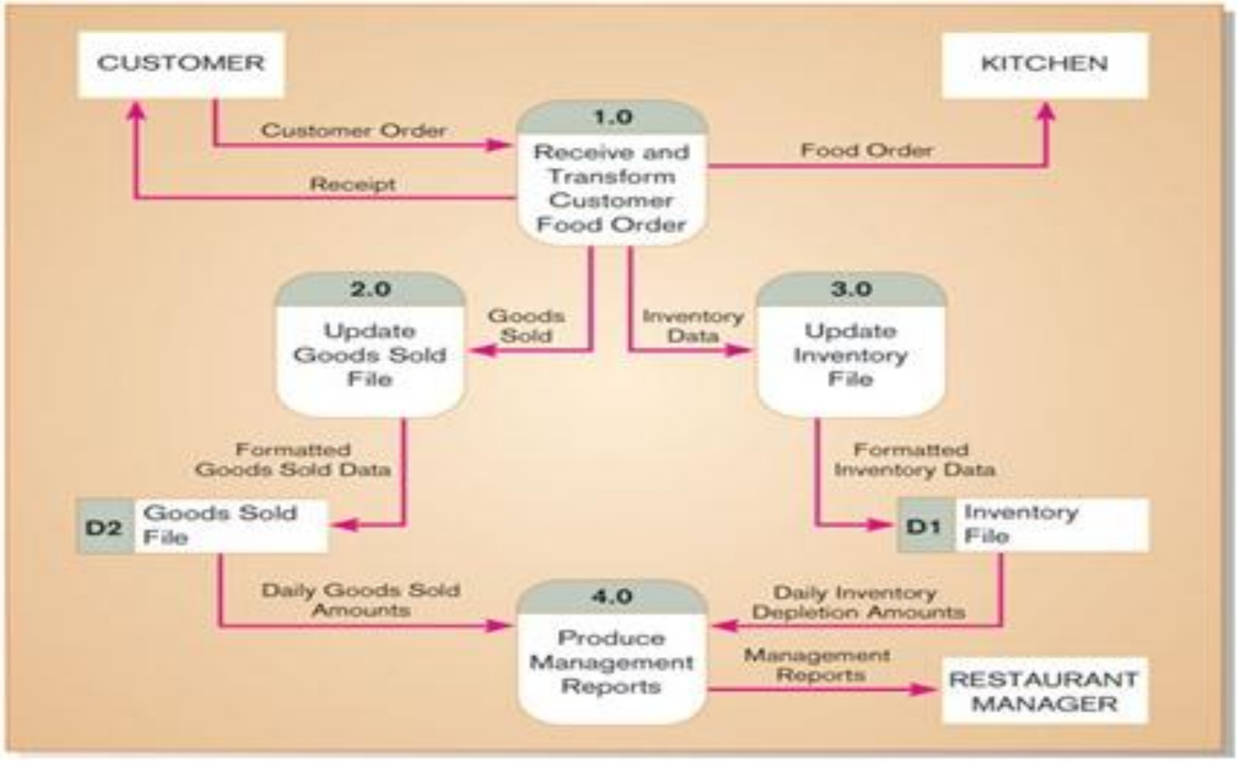
Decomposition Diagram



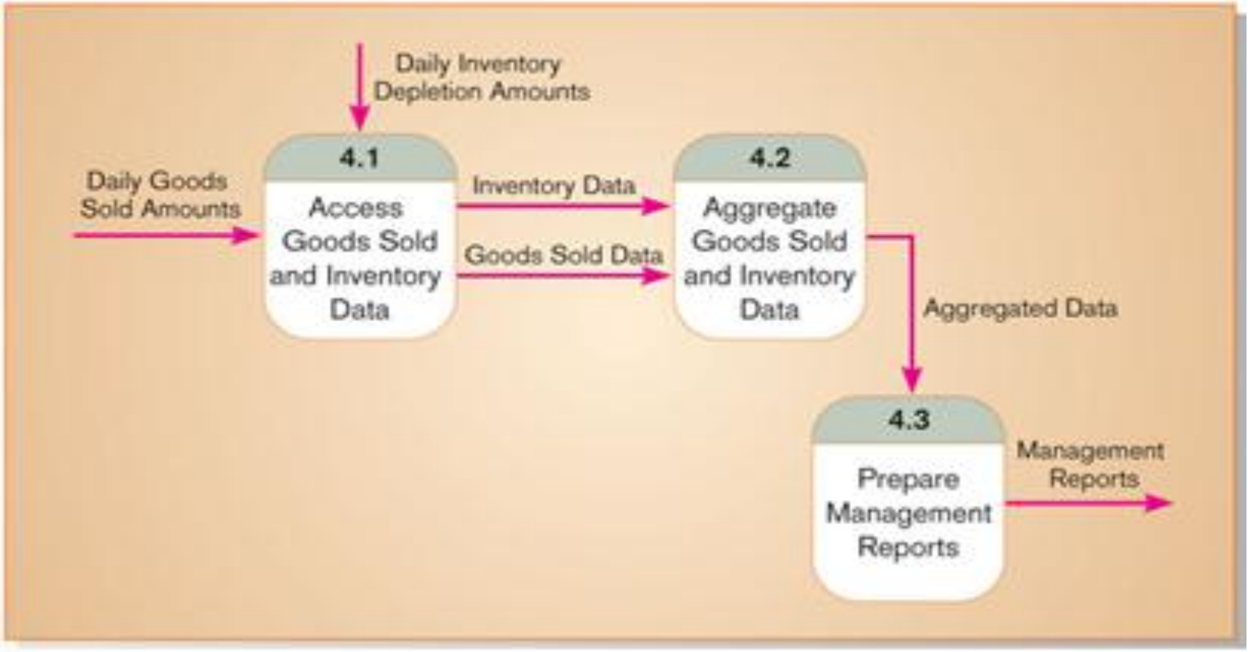
Decomposition of Context Diagram



Level 0 DFD Food Ordering System

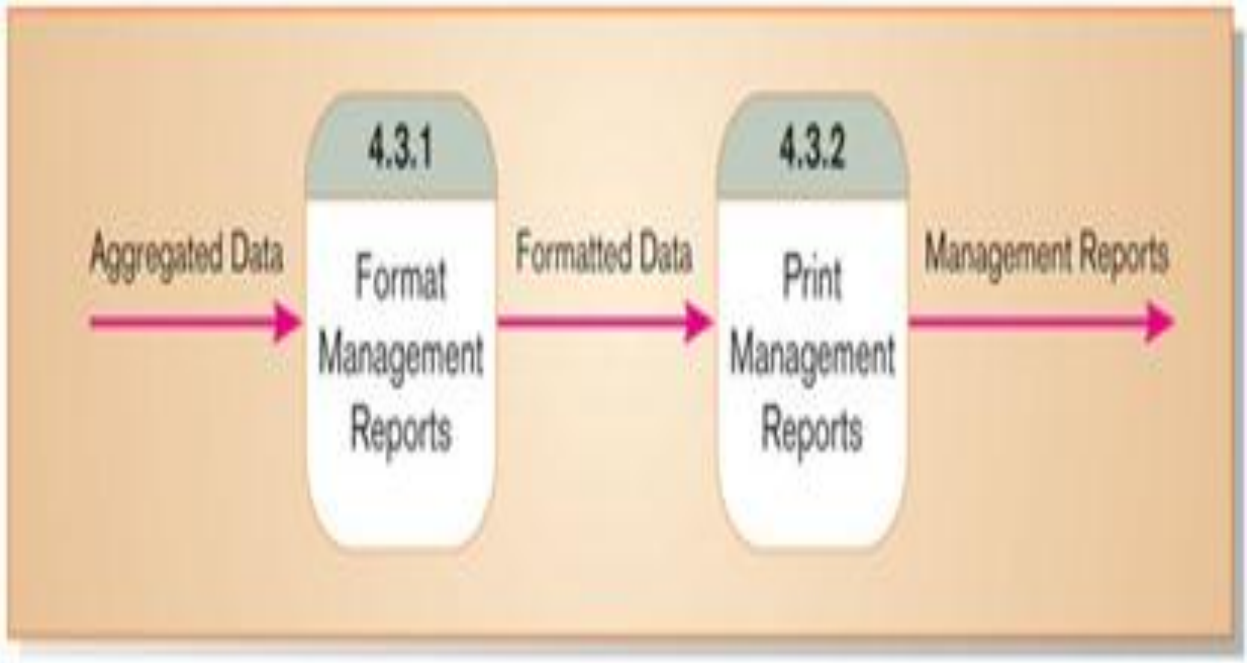


Level 1 DFD Food Ordering System


Daily Inventory Depletion Amounts
Daily Goods Sold Amounts
4.1
Access Goods Sold and Inventory Data
Inventory Data
Goods Sold Data
4.2
Aggregate Goods Sold and Inventory Data
Aggregated Data
4.3
Prepare Management Reports
Management Reports

This is a Level-1 DFD for Process 4.0.

Level 2 DFD Food Ordering System



This is a Level-2 DFD for Process 4.0.

DFD Balancing

- The conservation of inputs and outputs to a data flow 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

Unbalanced DFD

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

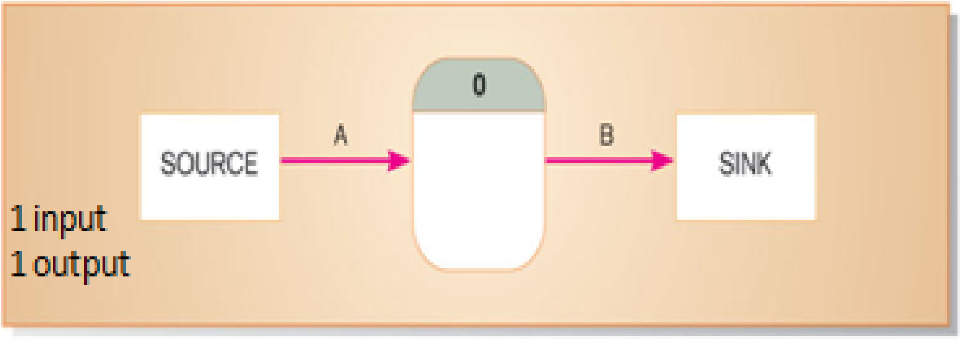
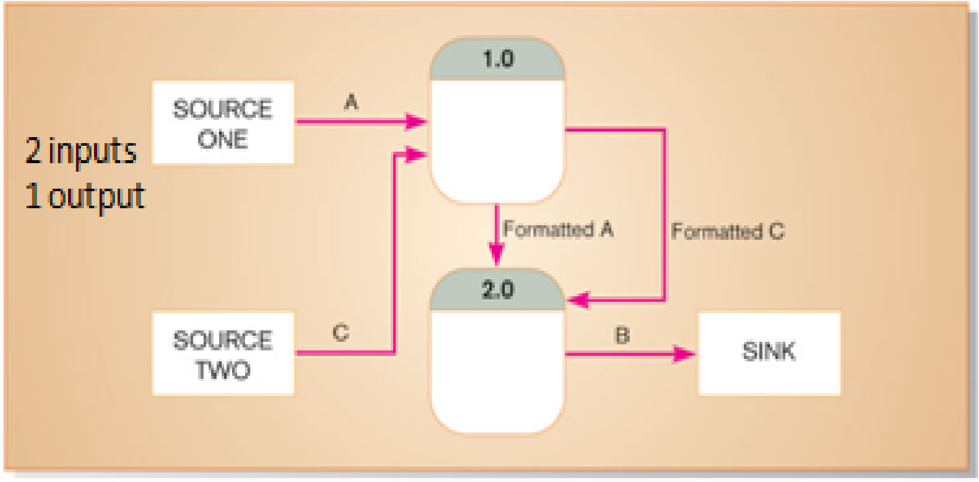


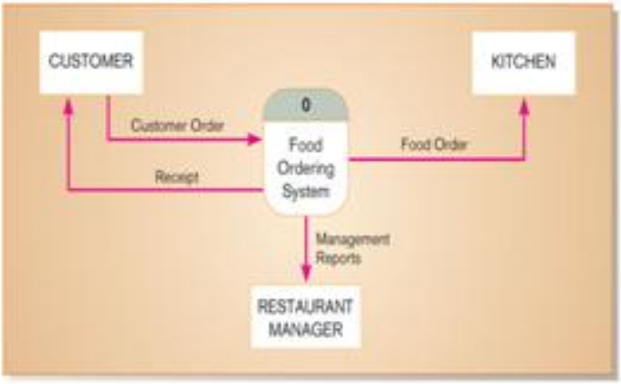
Figure 7-10b An unbalanced set of data flow diagrams - Level-0 diagram



This is unbalanced because the process of the context diagram has only one input but the Level-0 diagram has two inputs.

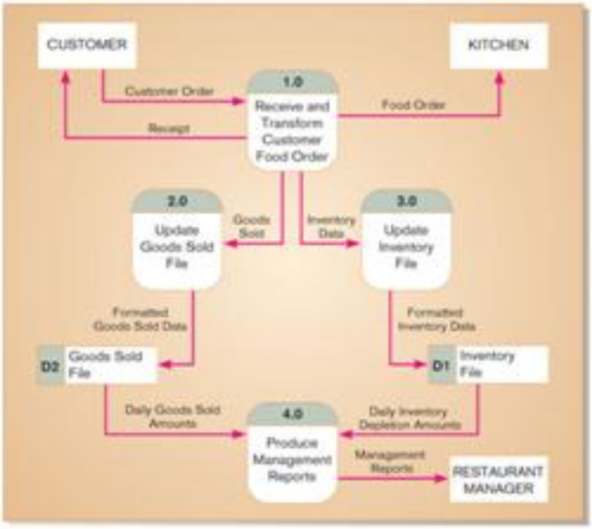
Balanced DFD

Figure 7-4 Context diagram of Hoosier Burger's food ordering system



1 input
2 outputs

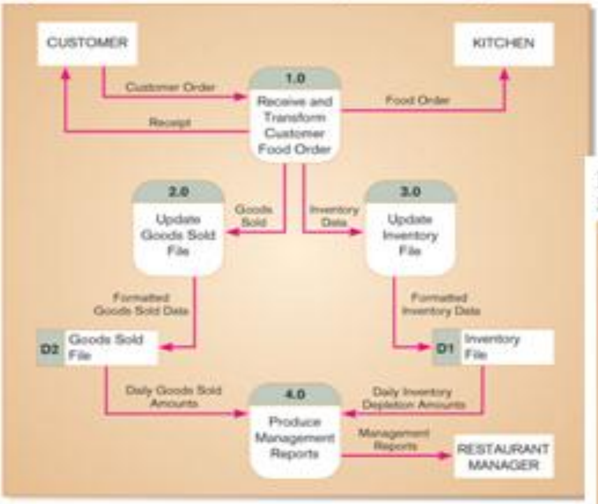
Figure 7-5 Level-0 DFD of Hoosier Burger's food ordering system



These are balanced because the numbers of inputs and outputs of context diagram process equal the number of inputs and outputs of Level-0 diagram.

Balanced DFD (cont.)

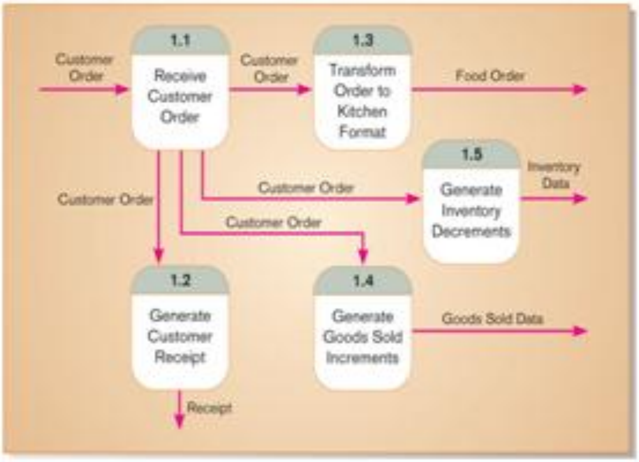
Figure 7-5 Level-0 DFD of Hoosier Burger's food ordering system



1 input
4 outputs

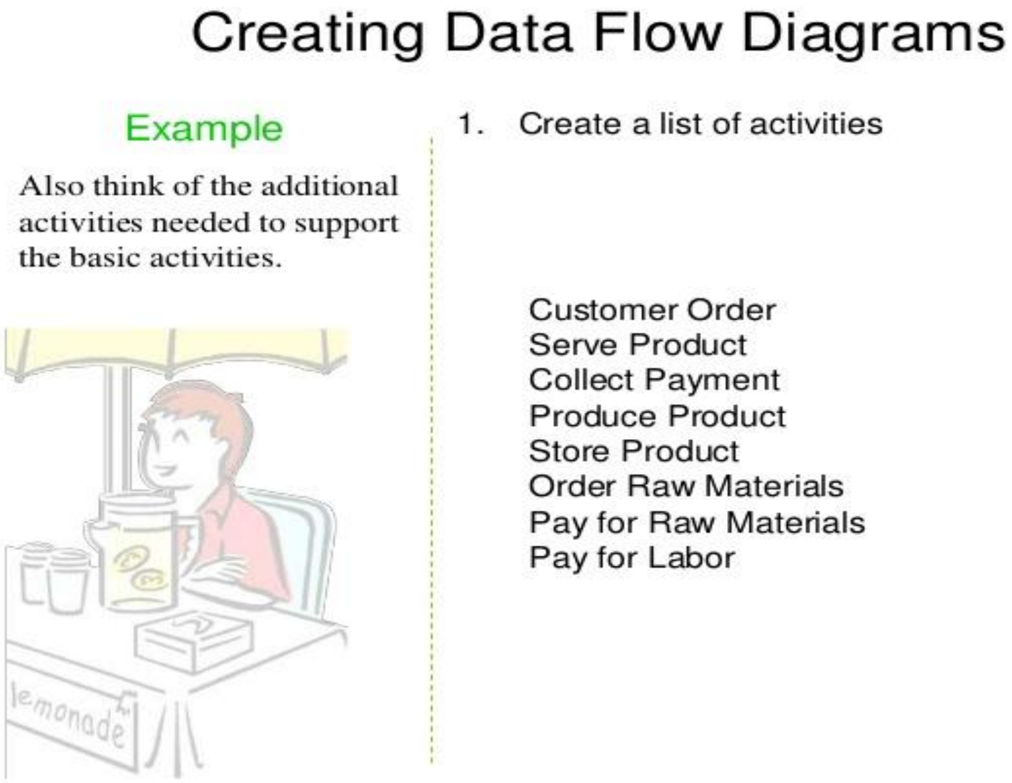
These are balanced because the numbers of inputs and outputs to Process 1.0 of the Level-0 diagram equals the number of inputs and outputs to the Level-1 diagram.

Figure 7-7 Level-1 diagram showing the decomposition of Process 1.0 from the level-0 diagram for Hoosier Burger's food ordering system



Guidelines for Drawing DFDs

- Completeness
  - DFD must include all components necessary for system.
- Consistency
  - The extent to which information contained on one level of a DFDs is also included on other levels.
- Timing
  - Time is not represented on DFDs.
- Iterative Development
  - Analyst should expect to redraw diagram several times before reaching the closest approximation to the system being modeled.
- Rules for stopping decomposition
  - When each process has been reduced to a single decision, calculation or database operation
  - When each data store represents data about a single entity
  - When the system user does not care to see any more detail
  - 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, online 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



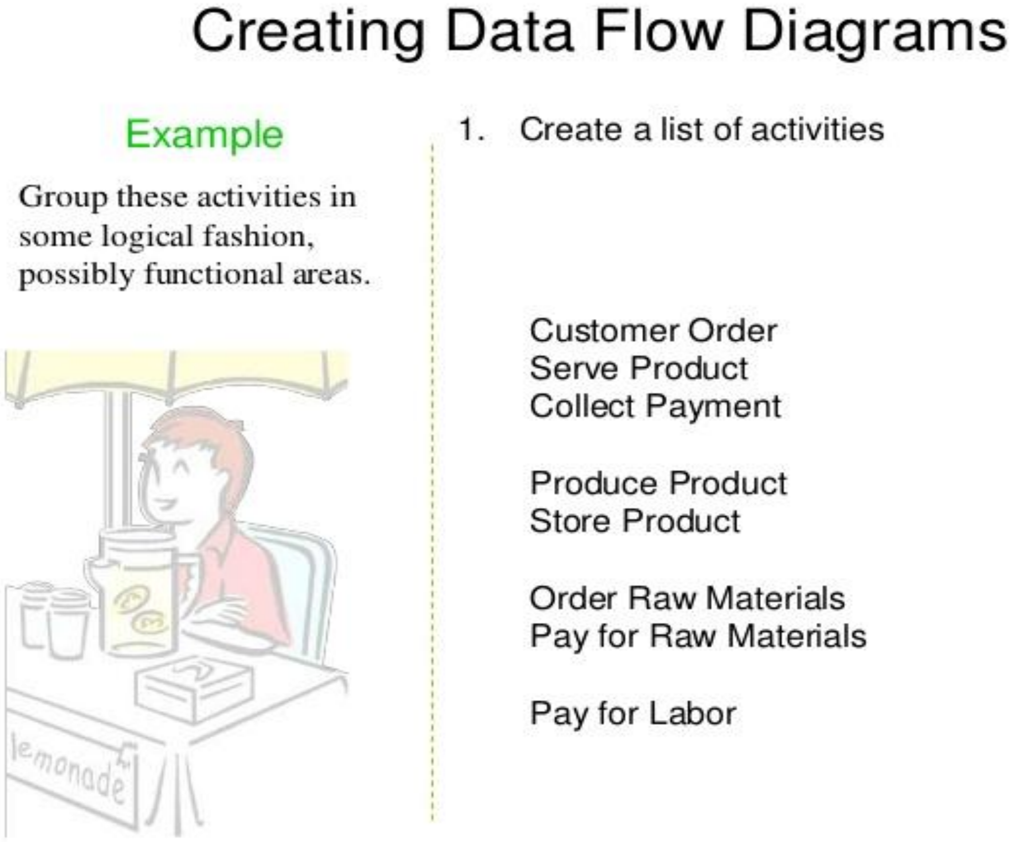
Creating Data Flow Diagrams

Example

Also think of the additional activities needed to support the basic activities.

1. Create a list of activities

Customer Order
Serve Product
Collect Payment
Produce Product
Store Product
Order Raw Materials
Pay for Raw Materials
Pay for Labor



Creating Data Flow Diagrams

Example

Group these activities in some logical fashion, possibly functional areas.

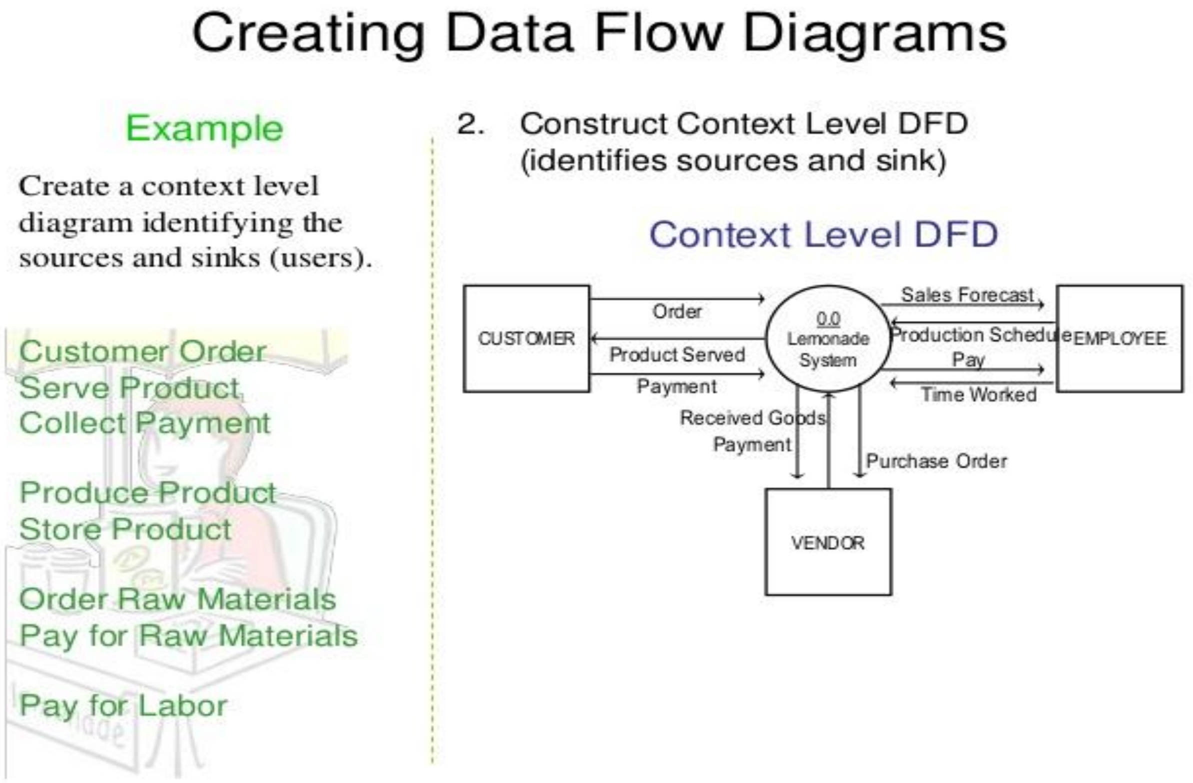
1. Create a list of activities

Customer Order
Serve Product
Collect Payment

Produce Product
Store Product

Order Raw Materials
Pay for Raw Materials

Pay for Labor



Creating Data Flow Diagrams

Example

Create a context level diagram identifying the sources and sinks (users).

Customer Order
Serve Product
Collect Payment

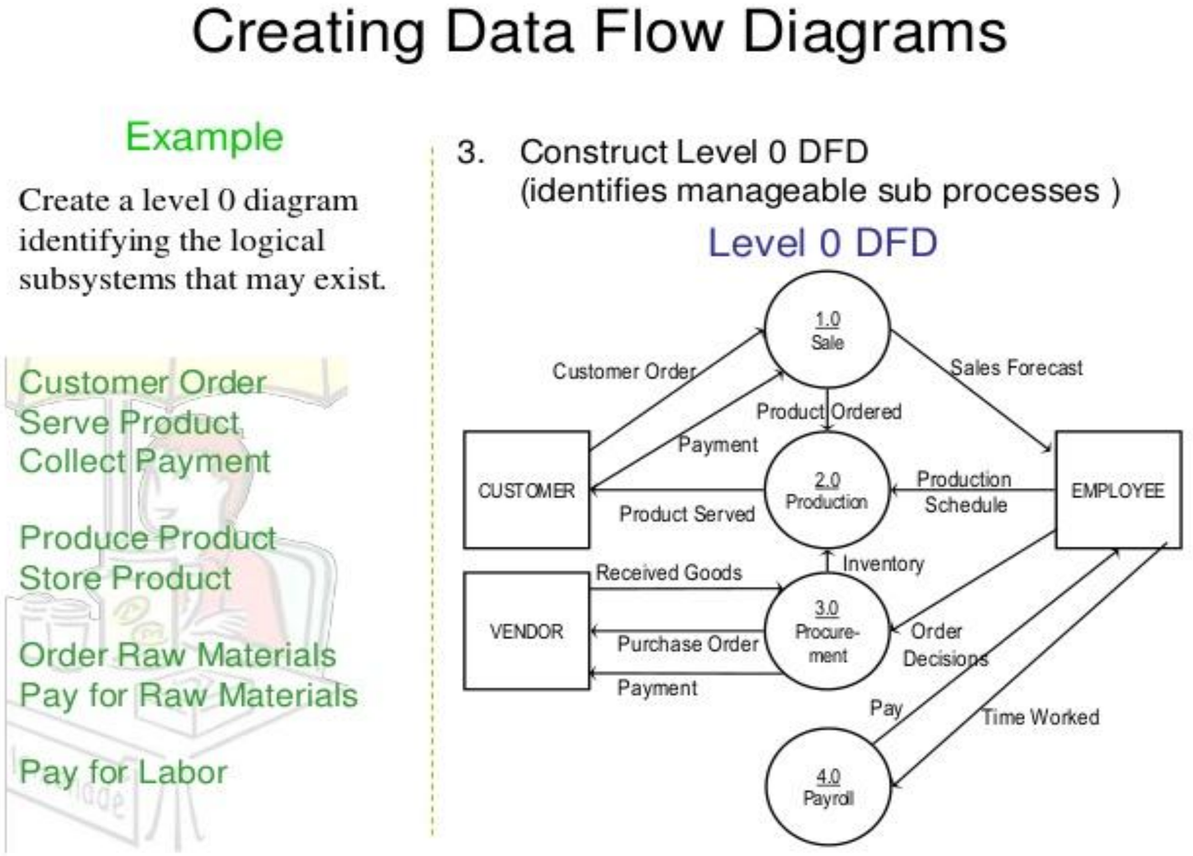
Produce Product
Store Product

Order Raw Materials
Pay for Raw Materials

Pay for Labor

2. Construct Context Level DFD (identifies sources and sink)

Context Level DFD



Creating Data Flow Diagrams

Example

Create a level 0 diagram identifying the logical subsystems that may exist.

Customer Order
Serve Product
Collect Payment

Produce Product
Store Product

Order Raw Materials
Pay for Raw Materials

Pay for Labor

3. Construct Level 0 DFD (identifies manageable sub processes)

Level 0 DFD

Creating Data Flow Diagrams

Example

Create a level 1 decomposing the processes in level 0 and identifying data stores.

Customer Order
Serve Product
Collect Payment

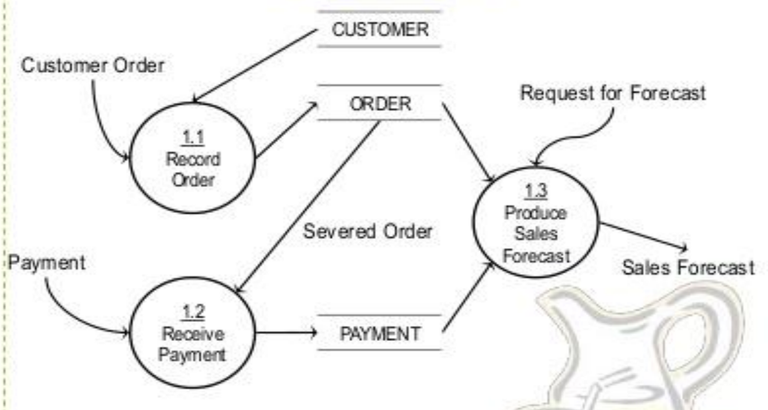
Produce Product
Store Product

Order Raw Materials
Pay for Raw Materials

Pay for Labor

4. Construct Level 1- n DFD
(identifies actual data flows and data stores)

Level 1 DFD



Creating Data Flow Diagrams

Example

Create a level 1 decomposing the processes in level 0 and identifying data stores.

Customer Order
Serve Product
Collect Payment

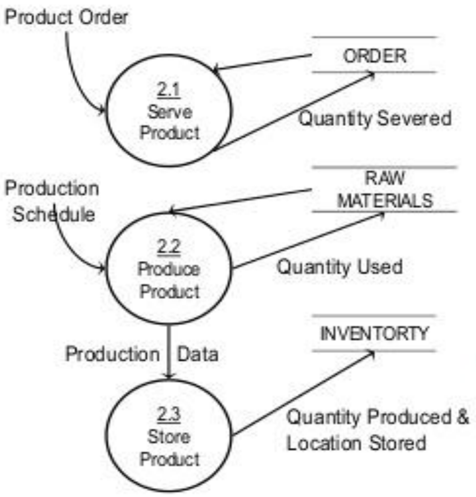
Produce Product
Store Product

Order Raw Materials
Pay for Raw Materials

Pay for Labor

4. Construct Level 1 (continued)

Level 1 DFD



Creating Data Flow Diagrams

Example

Create a level 1 decomposing the processes in level 0 and identifying data stores.

Customer Order
Serve Product
Collect Payment

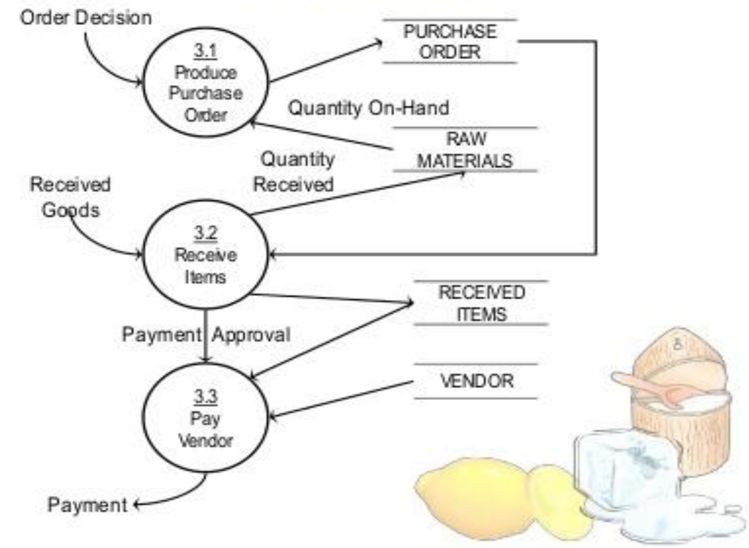
Produce Product
Store Product

Order Raw Materials
Pay for Raw Materials

Pay for Labor

4. Construct Level 1 (continued)

Level 1 DFD



Creating Data Flow Diagrams

Example

Create a level 1 decomposing the processes in level 0 and identifying data stores.

Customer Order
Serve Product
Collect Payment

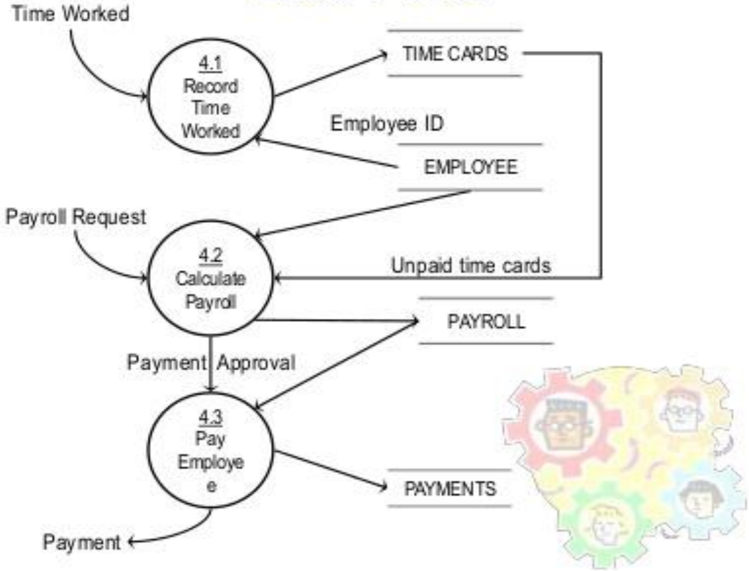
Produce Product
Store Product

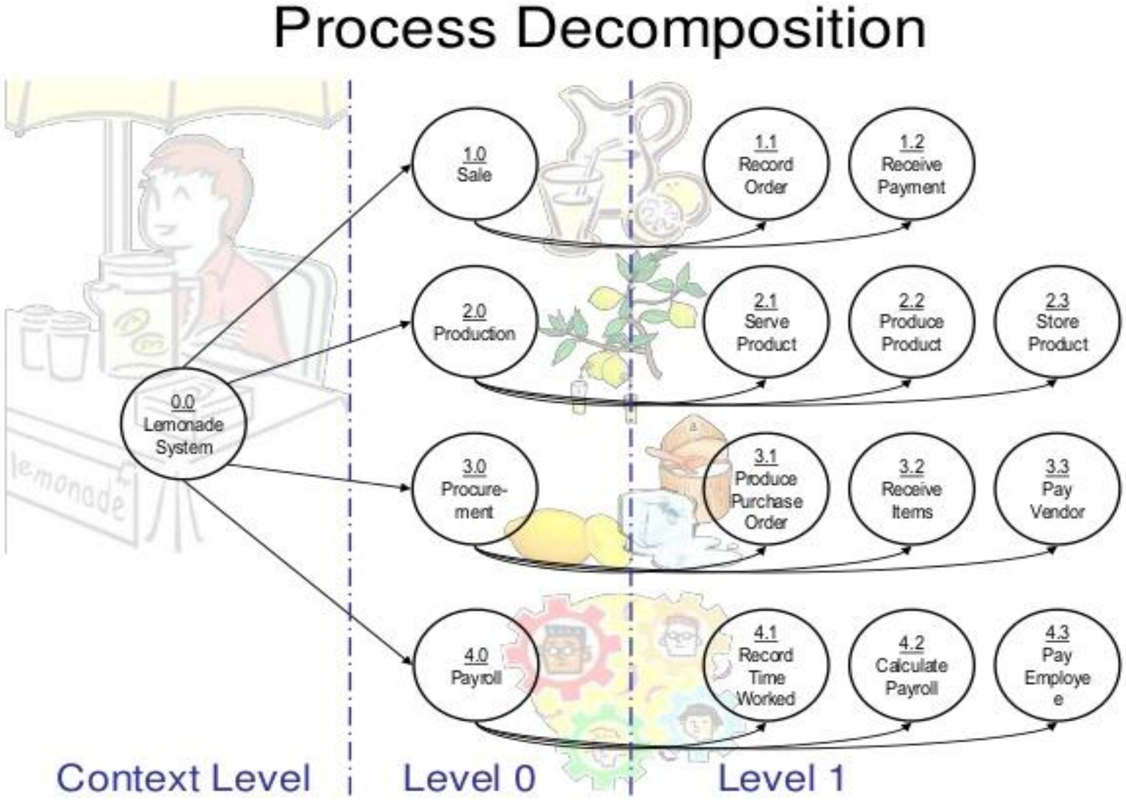
Order Raw Materials
Pay for Raw Materials

Pay for Labor

4. Construct Level 1 (continued)

Level 1 DFD





2.4Computer-Aided Software Engineering (CASE) Tools

Computer-Aided Software Engineering (CASE)

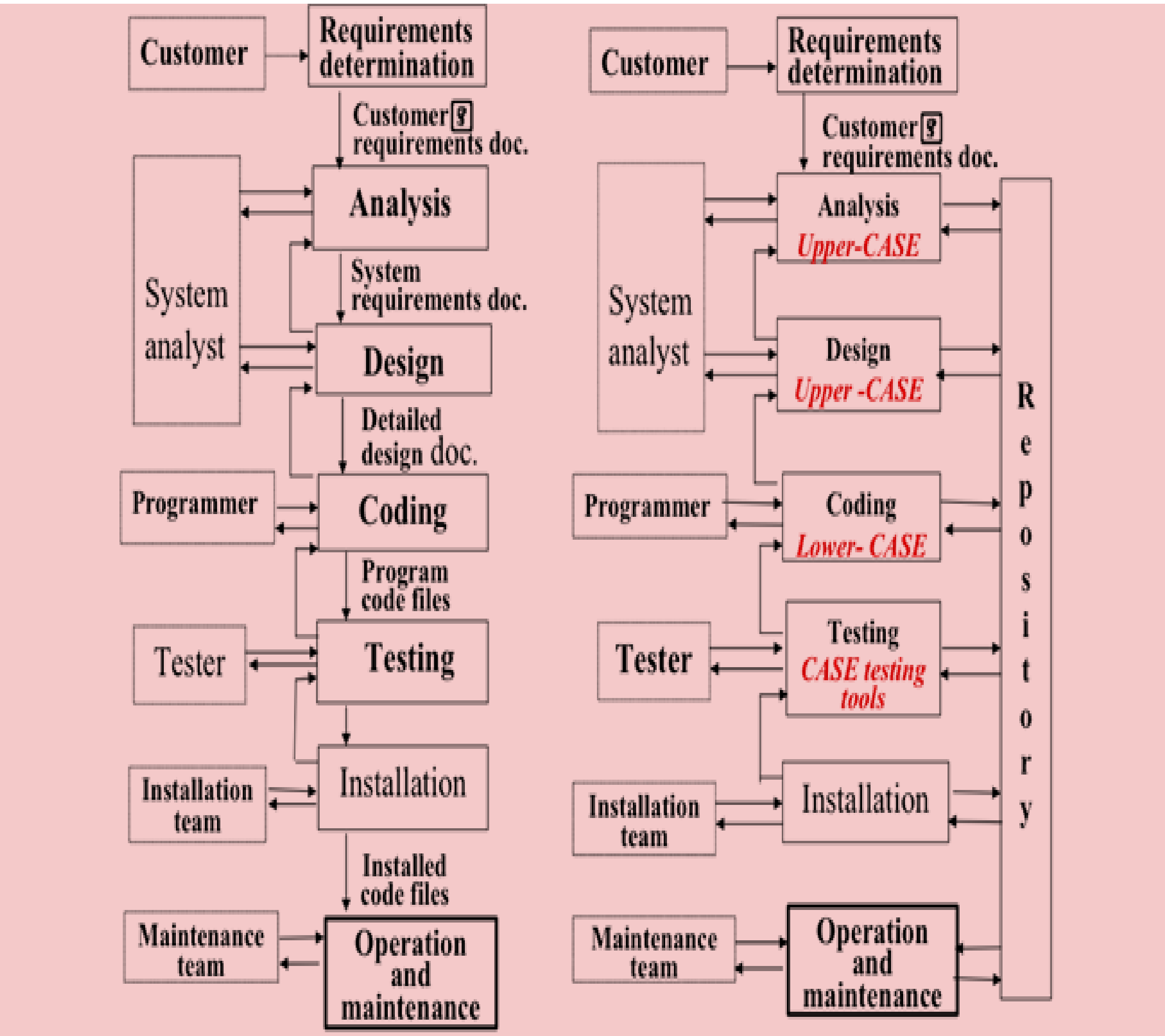
- *Automated software tools* used by *systems analysts* to develop information systems
- Used to *support* or *automate* activities throughout the systems development life cycle (SDLC)
- Increase *productivity*
- Improve overall *quality* of systems

Four reasons for using CASE tools are:

- To increase analyst productivity.
- Facilitate communication among analysts and users.
- Providing continuity between life cycle phases.
- To assess the impact of maintenance.

Components of CASE tools

- **Upper-CASE tools (front-end tools)**
  - Assist developer during requirements, analysis, and design workflows or activities
- **Lower-CASE tools (back-end tools)**
  - Assist with implementation, testing, and maintenance workflows or activities
- **Integrated CASE tools (I-CASE)**
  - provide support for the full life cycle



Traditional development life cycle   Real CASE tool-supported development life cycle

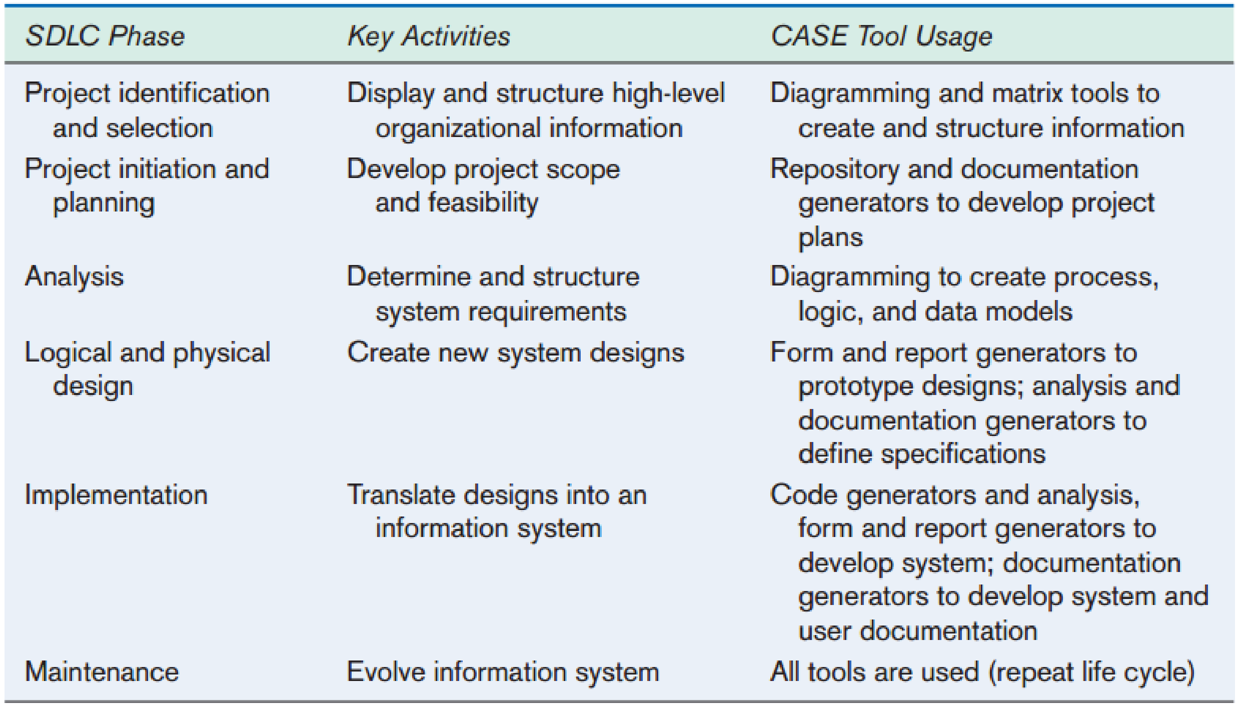
CASE versus Traditional Systems Development

Traditional Systems Development	CASE-Based Systems Development
Emphasis on coding and testing	Emphasis on analysis and design
Paper-based specifications	Rapid interactive prototyping
Manual coding of programs	Automated code generation
Manual documenting	Automated documentation generation
Intensive software testing	Automated design checking
Maintain code and documentation	Maintain design specifications

The Role of CASE in Conceptual Data

- CASE tools provide two important functions:
  - Maintain E-R diagrams as a visual representation of structured data requirements
  - Link objects on E-R diagrams to corresponding visual representation in a repository

TABLE 1-3 Examples of CASE Usage within the SDLC



SDLC Phase	*Key Activities*	*CASE Tool Usage*
Project identification and selection	Display and structure high-level organizational information	Diagramming and matrix tools to create and structure information
Project initiation and planning	Develop project scope and feasibility	Repository and documentation generators to develop project plans
Analysis	Determine and structure system requirements	Diagramming to create process, logic, and data models
Logical and physical design	Create new system designs	Form and report generators to prototype designs; analysis and documentation generators to define specifications
Implementation	Translate designs into an information system	Code generators and analysis, form and report generators to develop system; documentation generators to develop system and user documentation
Maintenance	Evolve information system	All tools are used (repeat life cycle)

Automated software tools used by systems analysts to develop information systems

- Project management tools (Openproj, MS Project)
- Diagramming tools (Dia, Visio)
- Computer display and report generators (VB)
- Analysis tools (Visual Analyst)
- Documentation generators
- Code generators
- Integration of tools via a repository

2.5 E-R Diagram

Introduction to Entity-Relationship (E-R) Modeling

- Entity-Relationship (E-R) Diagram
  - A detailed, logical representation of the entities, associations and data elements for an organization or business
- Notation uses three main constructs
  - Entities
  - Relationships
  - Attributes

Entities

Specifies distinct real world items in an application

- In the University database context, an individual student, faculty member, a class room, a course are entities.

Entity Set or Entity Type -

Collection of entities all having the same properties.

- Student entity set –collection of all student entities.
- Course entity set –collection of all course entities.

Every member of an entity set is described by its attributes

Attributes

Attributes specify properties of members of entity set & also specify properties of relationships

Student entity

- ƒ Stud Name –name of the student.
- ƒ RollNumber –the roll number of the student.
- ƒ Sex –the gender of the student etc.

All entities in an Entity set/type have the same set of attributes.

Types of Attributes

•**Simple Attributes**

Having atomic or indivisible values.

Example: Dept. – A string
Phone Number –an eight digit number

•**Composite Attributes**

Having several components in the value.

Example: Qualification with components
(Degree Name, Year, University Name)

•**Derived Attributes**

Attribute value is dependent on some other attribute.

Example: Age depends on DateOfBirth.
So age is a derived attribute.

•Single-valued

Having only one value rather than a set of values.

For instance, PlaceOfBirth–single string value.

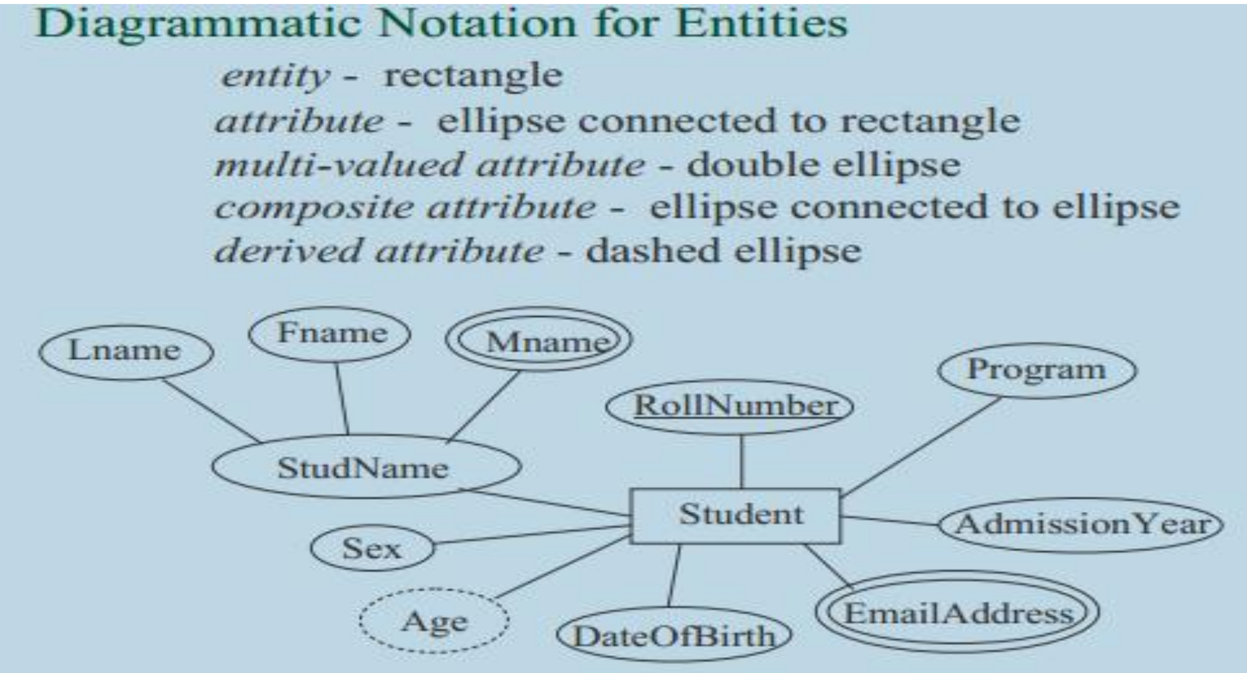
•Multi-valued

Having a set of values rather than a single value.

For instance, Courses Enrolled attribute for student

Email Address attribute for student

Previous Degree attribute for student.



Diagrammatic Notation for Entities

entity - rectangle

attribute - ellipse connected to rectangle

multi-valued attribute - double ellipse

composite attribute - ellipse connected to ellipse

derived attribute - dashed ellipse

RELATIONSHIP

Meaningful dependencies between entities

For example: vendor <u>supplies</u> items

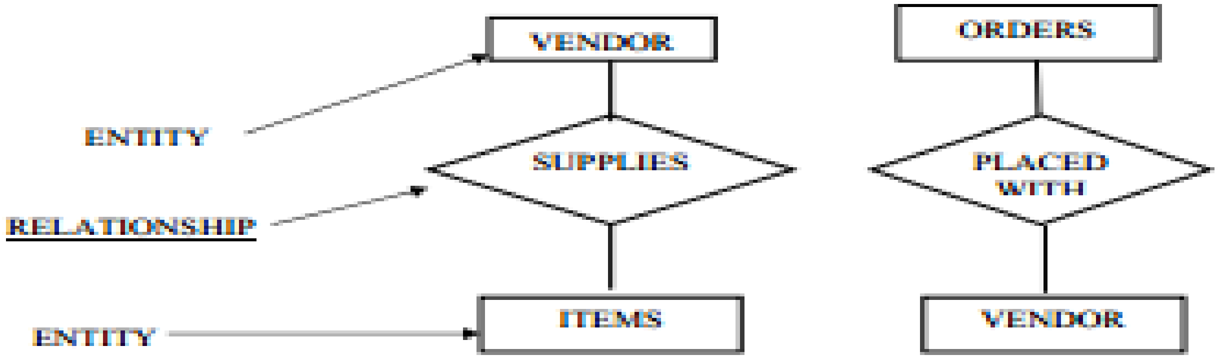
Teacher <u>teaches</u> courses

Relationships are underlined above

Entity-Relationship Diagram

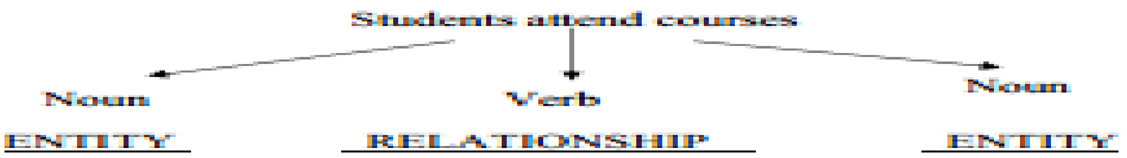
- Some entities depend on one another, for example, entity vendor and entity items are related as vendors supply items.
- These relationships are described by entity-relationship diagrams (or ER diagrams).

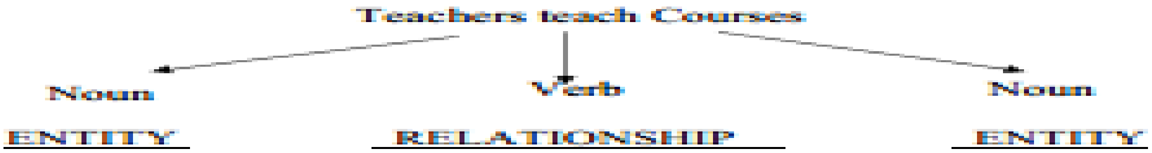
- In an ER diagram entities are represented by rectangles and relationships by diamond shaped boxes



How to Identify Entities and Relationships

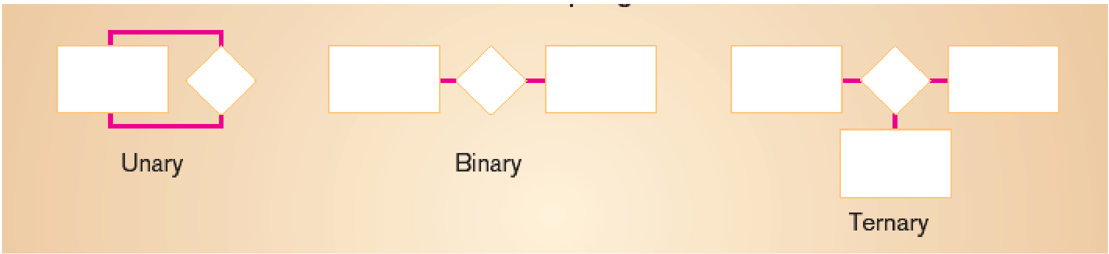
When a word statement is used to describe applications, nouns normally are entities and verbs relationships.

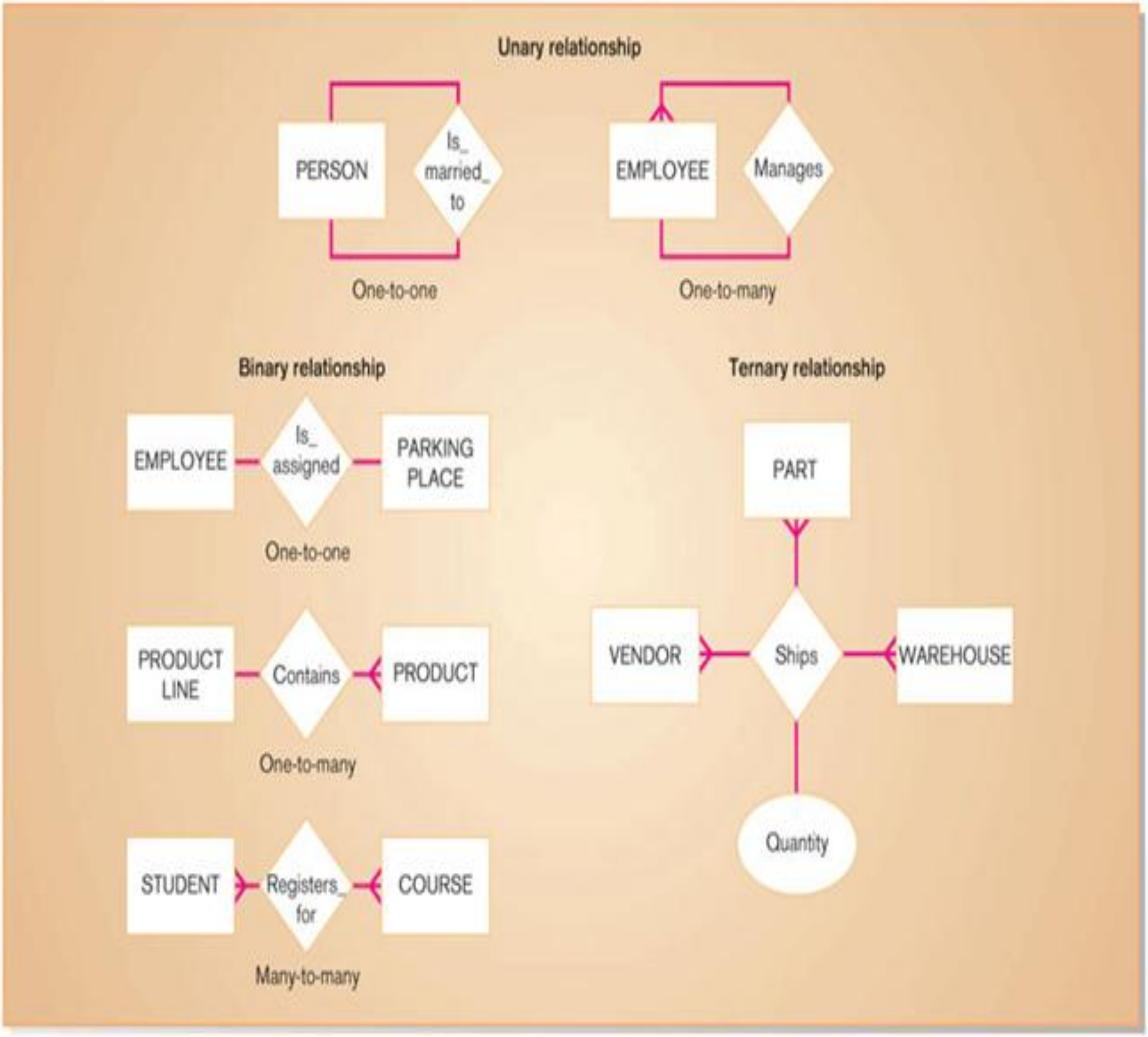




Degree of Relationship

- Degree: number of entity types that participate in a relationship
- Three cases
  - **Unary:** between two instances of one entity type
  - **Binary:** between the instances of two entity types
  - **Ternary:** among the instances of three entity types

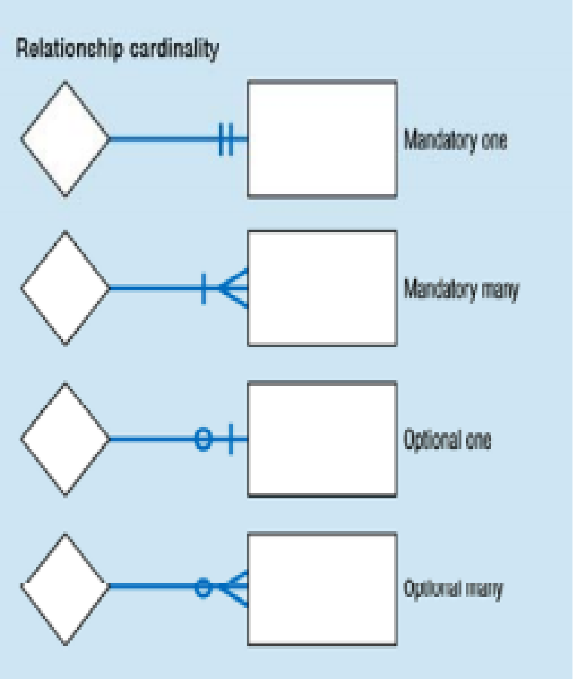




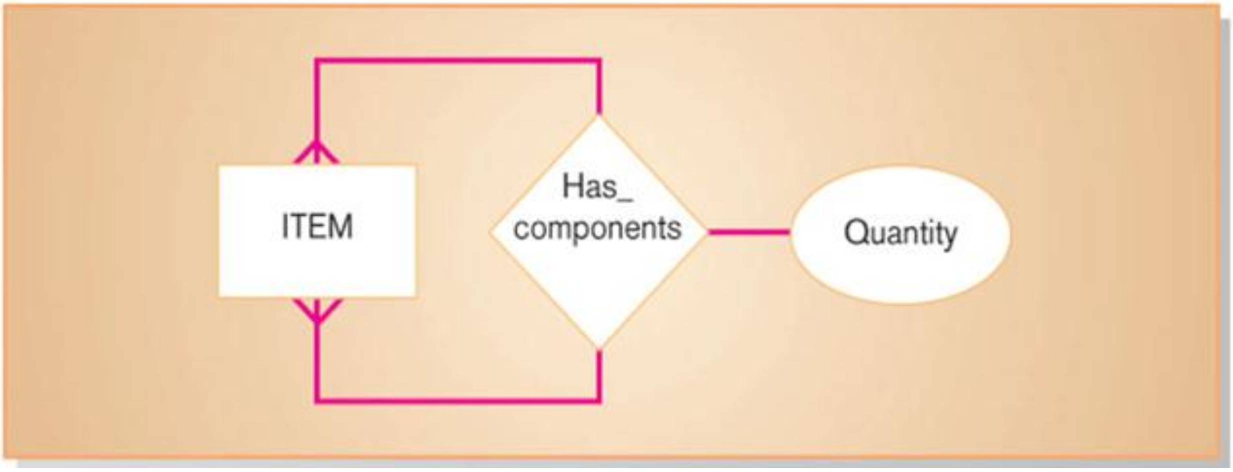
Cardinality

- The number of instances of entity B that can or must be associated with each instance of entity A
- Minimum Cardinality
  - The minimum number of instances of entity B that may be associated with each instance of entity A
- Maximum Cardinality
  - The maximum number of instances of entity B that may be associated with each instance of entity A
- Mandatory vs. Optional Cardinalities
  - Specifies whether an instance must exist or can be absent in the relationship

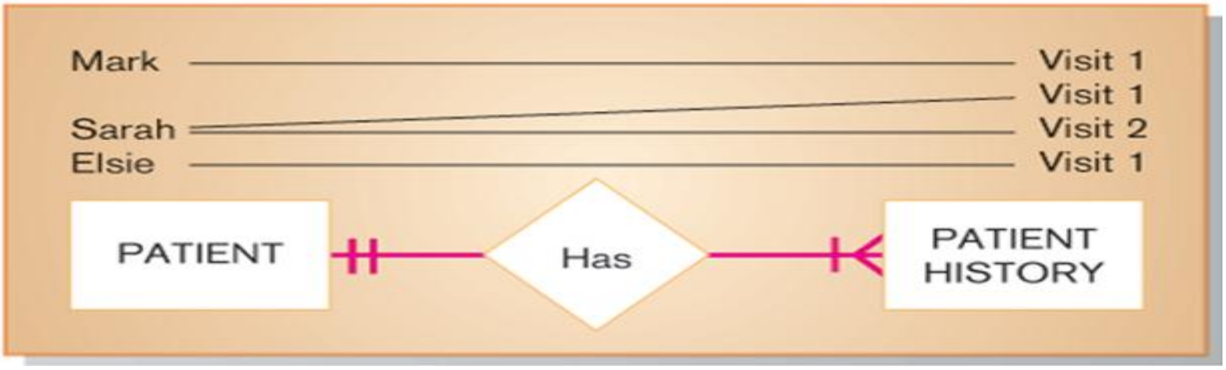
- Cardinality Constraints - the number of instances of one entity that can or must be associated with each instance of another entity.
- Minimum Cardinality
  - If zero, then optional
  - If one or more, then mandatory
- Maximum Cardinality
  - The maximum number



Unary Relationship Example

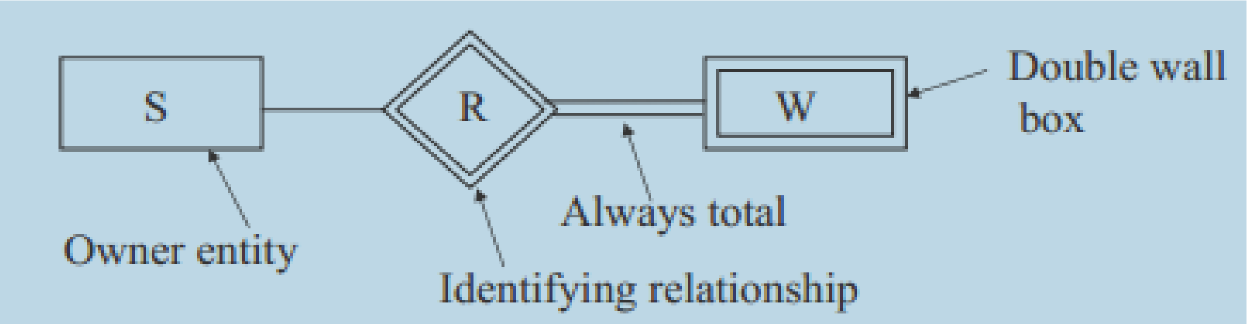


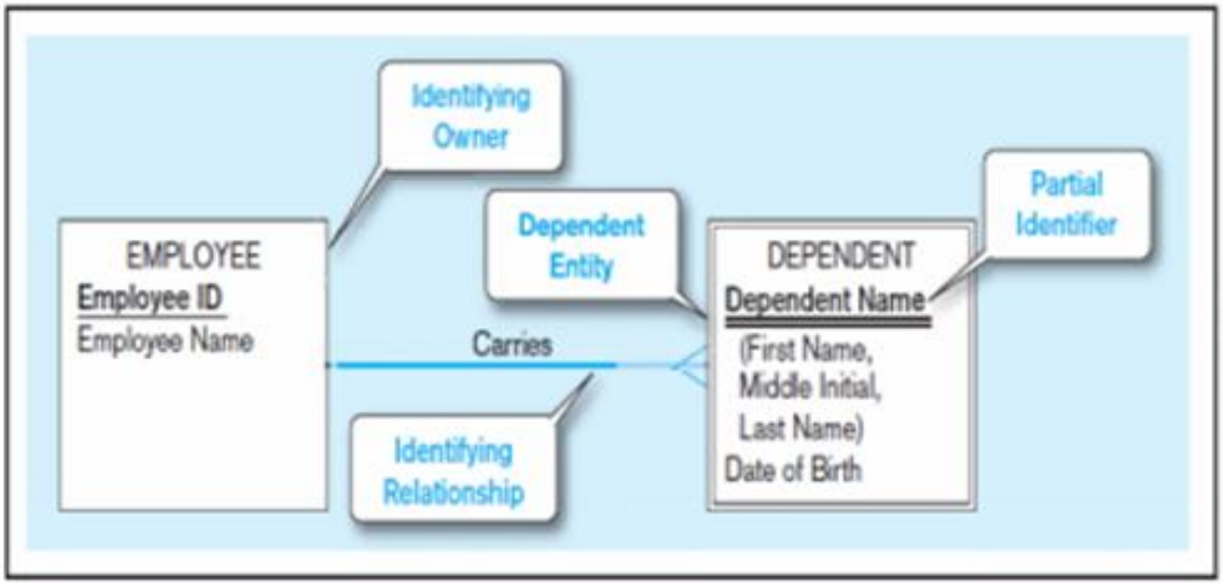
Binary Relationship Example



Strong vs.Weak Entities

- Strong entity
  - Exists independently of other types of entities
  - Has its own unique identifier
  - Identifier underlined with single-line
- Weak entity
  - Dependent on a strong entity …cannot exist on its own
  - Does not have a unique identifier (only a partial identifier)
  - Partial identifier underlined with double-line
  - Entity box has double line
- Identifying relationship
  - links strong entities to weak entities

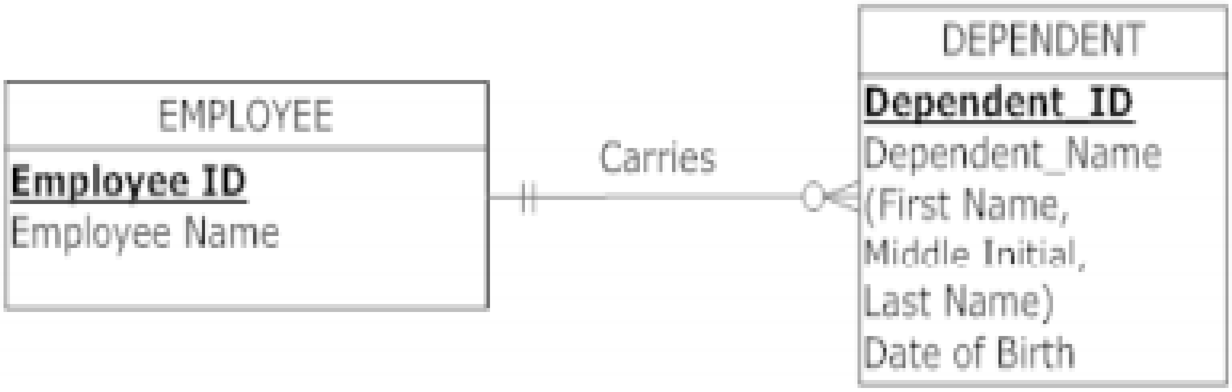




Strong entity

Weak entity

Weak Entity → Strong Entity



Strong entity

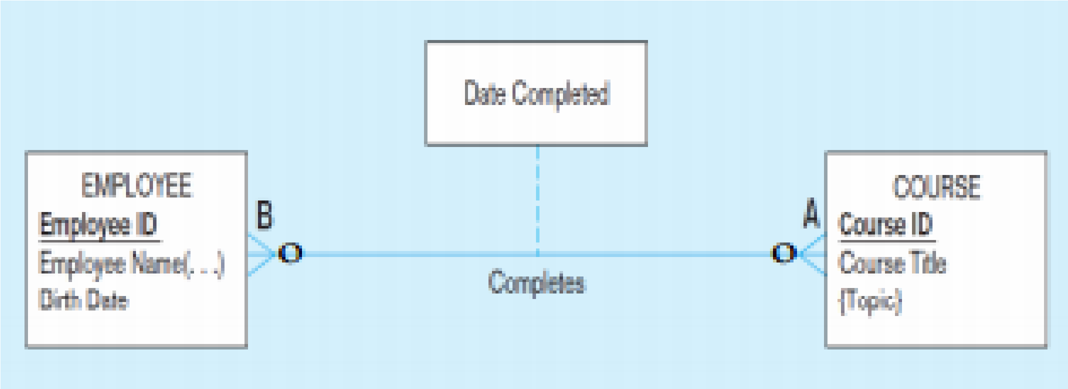
Now a
strong entity
(unique ID)

Associative Entities

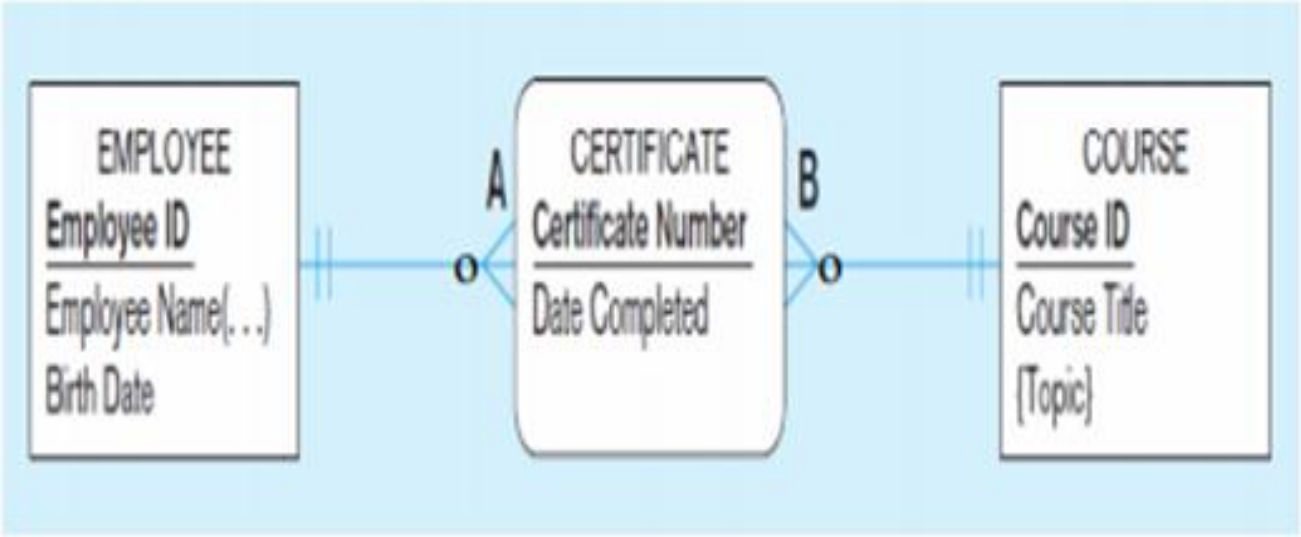
- One of the hardest concepts in E-R modeling
- *An associative entity is a relationship transformed into an entity*
- Each *instance* of an associative entity represents an *instance* of the relationship
- Needed to represent ternary relationships, and for cases when we need to convert a relationship into an entity, to relate it to other entities.

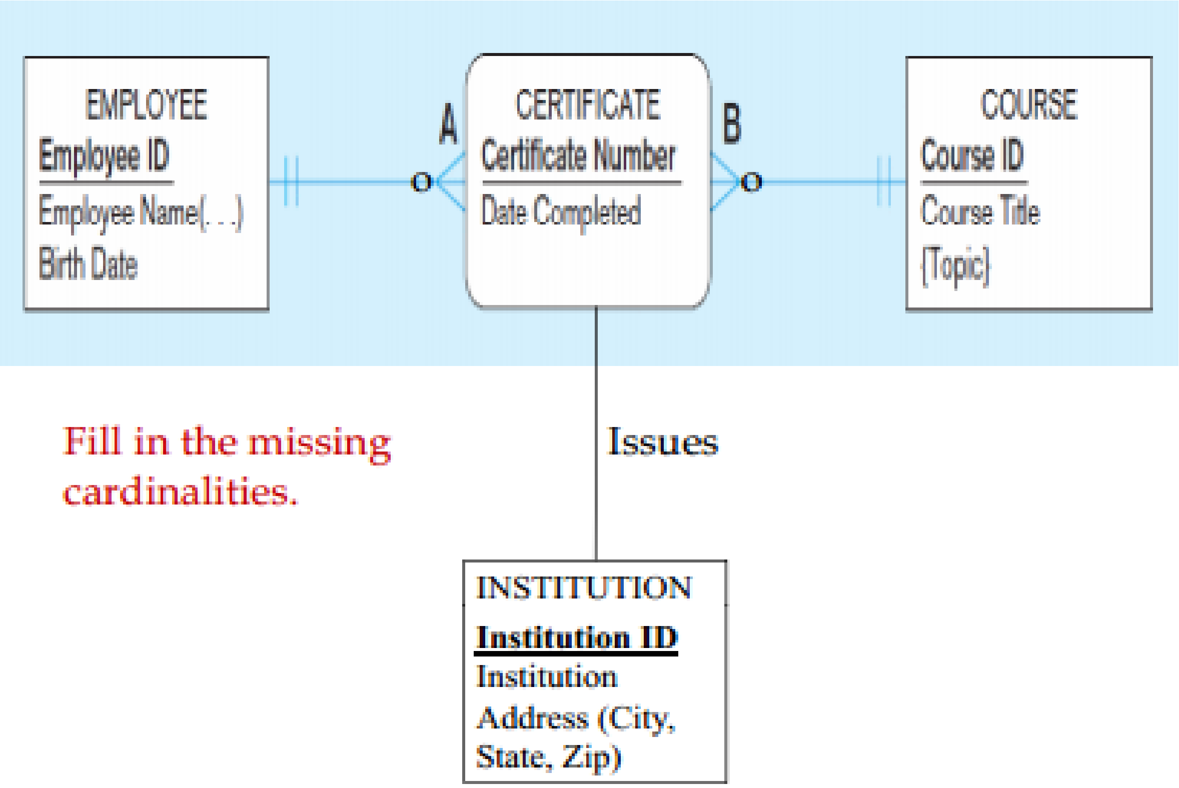
Associative Entities - Example

- Previous model: a database of courses taken by employees. For each employee we keep its SSN (identifier), name and birth date, and for each course we keep its id and title. We also keep the date the employee completed that course.

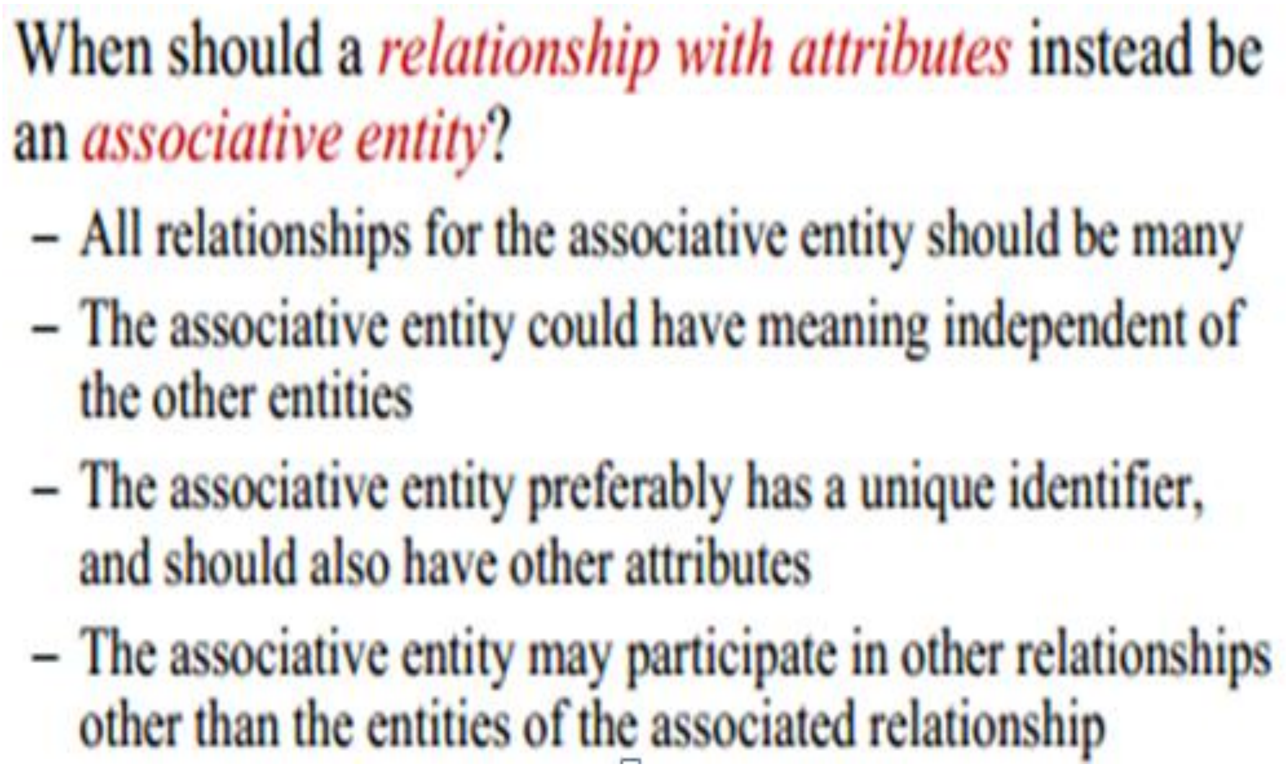


- Suppose that we also want to record information about the institutions issuing the certificates. HOW???





Fill in the missing cardinalities.



When should a *relationship with attributes* instead be an *associative entity*?

- All relationships for the associative entity should be many
- The associative entity could have meaning independent of the other entities
- The associative entity preferably has a unique identifier, and should also have other attributes
- The associative entity may participate in other relationships other than the entities of the associated relationship

Assignment II

1. Draw a DFD diagram of following up to level 2.
   Customer sends enquiry to commercial department; receives quotations from the sales department and places an order. Based on the customer order, the work order is sent to the planning department for planning scheduling and control, in turn, the planning department raises a job order on the "shop floor". On completion, delivery note and invoice are made out costing department also prepares an order wise comparative statement of estimated and actual costs.

2. For the following situation, draw a context-level diagram and a level-0 data flow diagram.
   Kellogg State Bank provides car and home loans to its banking customers. Initially, a potential loan customer meets with a Kellogg loan officer, requests a loan for a certain amount and time frame, and completes a loan application. Next, the loan officer determines the customer's credit standing, the type of loan required, and available interest rates. While the loan officer can authorize car loans for credit worthy customers, a loan committee must approve all home loans.

3. Draw a DFD diagram of the following up to level 2
   Front office of Hotel is responsible for all room reservations, room allocations and final settlement of bills. Any company or person can reserve rooms for their future stay. They have to indicate from what date to what day they need the room. They also have to indicate how many rooms are required. Sometimes the reservations could be cancelled or the dates or number of rooms changed. For reservation, cancellation or modification or rooms, customer receives an acknowledgement from the hotel.

4. Draw a DFD of hostel mesh management system up to level 2.

5. Draw DFD up to level-2 the following specification of the system.

   Suppose you are going to design an information system for a departmental store where a customer sends enquiry to commercial department; receives quotations from the sales department and places an order. Based on the customer order, the work order is sent to the planning department for planning scheduling and control, in turn, the planning department raises a job order on the "shop floor". On completion, delivery note and invoice are made out costing department also prepares an otherwise comparative statement of estimated and actual costs.

6. Draw a DFD diagram of student information system up to level 2.

7. Draw the Entity- Relationship Diagram (ERD) for the following scenario:

A salesperson may manage many other salespeople. Salespeople is managed by salesperson. A salesperson can be an agent for many customers. A customer is managed by one salespeople. A customer can place many orders. An order can be placed by one customer. An order lists many inventory items. An inventory item may be listed on many orders. An inventory item is assembled from many parts. A part may be assembled into many inventory items. Many employees assemble an inventory item from many parts. A supplier supplies many parts. A part may be supplied by many suppliers.

8. Design the E-R diagram of the following:
   a) Customer with draws money from his account.
   b) Student attends classes.
9. What are the management skills needed by system analysts?
10. What are the three relationship types of E-R diagrams? How are these relationships paired to build an E-R diagram?
11. Differentiate between physical DFD and logical DFD?
12. What do you mean by case tools? Explain the case tools in data modeling.
13. What is CASE? What are the roles of CASE in data modeling?
14. What is CASE? What are its objectives? Explain usages of CASE in SDLC.
15. What are the system analysis and design tools?