BIT 1st Year

Semester 2

IT 2405

Systems Analysis and Design Chapter 5

Modeling Methods

How to simplify, present / document a complex problem?

The answer is just Simple, use **MODELS**

Model

Is a presentation of reality. Just a picture is worth a thousand of words, most system models are pictorial representations of reality.

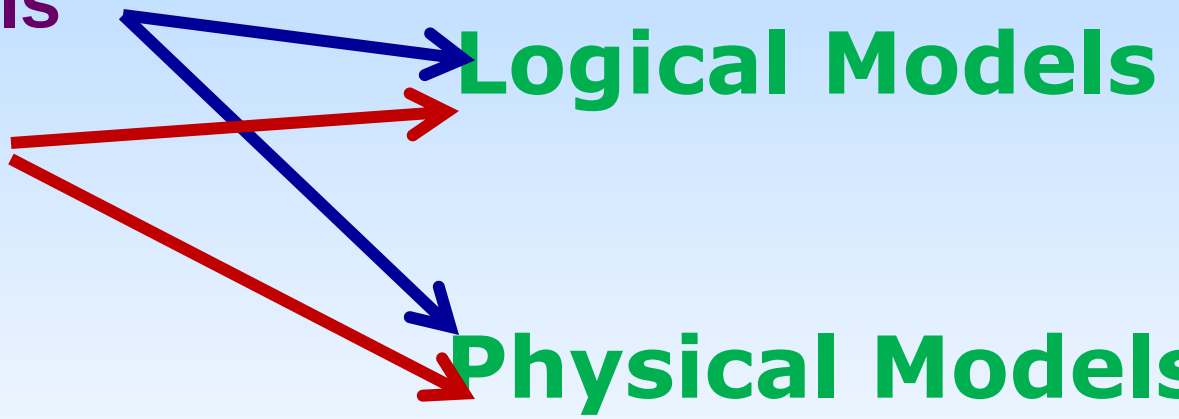
SAMPLE FLOOR PLAN

UCSC

Structured Methodologies

Models are divided into

- **Process Models**
- **Data Models**



Process Modeling

Introduction

- Technique for organizing and documenting the structure and flow of data through a system's process and the logic, policies, and procedures to be implemented by a system's process.
- Consists of various types of process models.

Process Modeling

Models

Logical Models

Other names:

~Essential model
~Conceptual model
~Business model

Physical Models

Other names:

~ Implementation Model
~ Technical model

Logical Process Models

- Show what a system is or does.
- Implementation – independent
  - depict the system independent of any technical dependence
- Illustrates the essence of the system
- Used to Depict business and non technical requirements
- Used to document system's Process focus from the systems owners' and users' perspective
- Encourage creativity
- Reduce the risk of missing business requirements
- Allows better communication with end-users in non-technical / less technical languages.

Physical Process Models

- Show not only what a system is or does. But also how the system is physically and technically implemented.
- Implementation –dependent
- Reflect technology choices and the limitations of those technology choices
- Used to Depict technical designs

Process Modeling

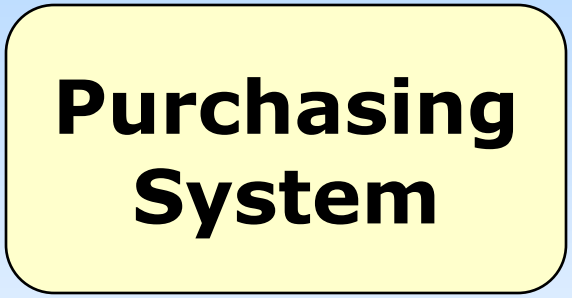
Program Structure Charts
Logic Flow Charts
Decision Tables, are some examples for various types of process models found in early software engineering methods and programming.

Data Flow Diagram : Popular System Analysis Process Model.

Data Flow Diagrams

- Shows the flow of data through the system and the processing performed by the system
- Other words : bubble chart, transformation graph, and process model
- Some analysts draw a **decomposition diagram** before DFD
- There exist several competing symbol sets for DFDs.
  - **Gane and Sarson notation is widely popular**

Elements in a DFD
(Gane and Sarson Symbols)



A Process

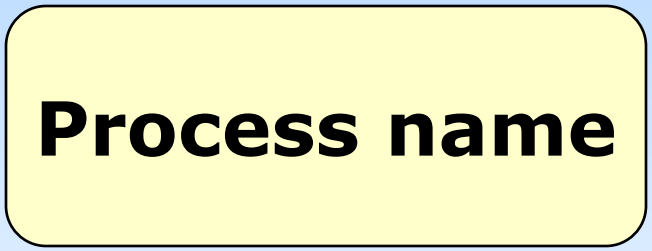
An External Agent

Invoice

A Data Flow

A Data Store

Elements in a DFD
(Gane and Sarson Symbols)



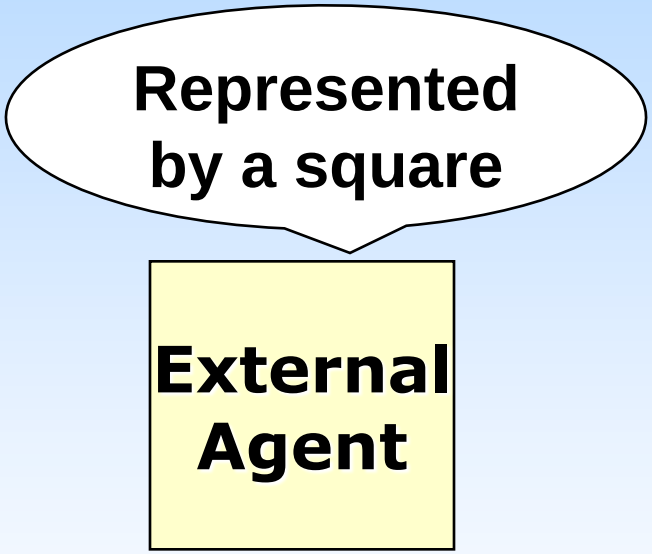
A Processes or Work to be done

Represented by a rounded rectangle

- A Process is work performed by a system In response to incoming data flows or conditions and it transforms incoming data flow into outgoing data flow.

A Synonym is transform

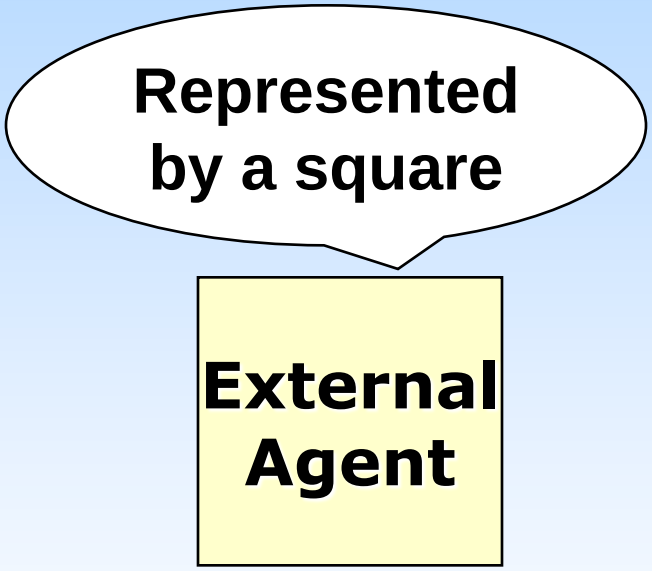
Elements in a DFD
(Gane and Sarson Symbols)



An External Agent

An external agent is an outside person (e.g. supplier, customer), organization unit (e.g. other dept), system (other business systems), or organization (e.g. Bank) that interact with the system. Also called an external entity.

Elements in a DFD
(Gane and Sarson Symbols)



An External Agent

External Agents

Provide the net inputs into the system and receive net outputs from the system being defined.

External

external to the system being analyzed or designed.

Elements in a DFD
(Gane and Sarson Symbols)

A Data Store

Represented by the open-end box

A Data Store is an "inventory" of data. That is, stored data intended for later use (data at rest). Also known as a file or database.

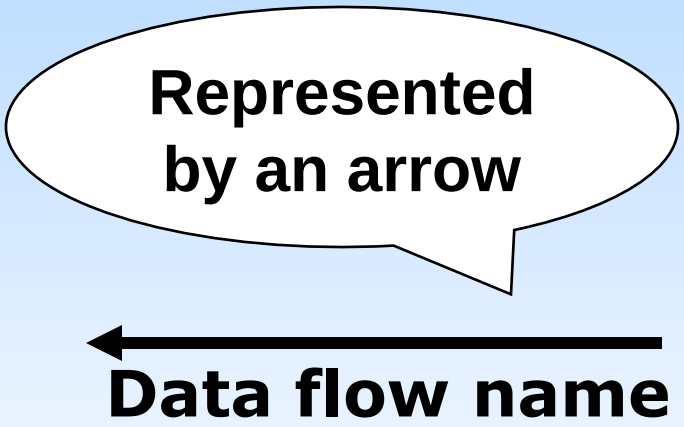
Elements in a DFD
(Gane and Sarson Symbols)

A Data Store

- Data stores should describe “things” about which the business wants to store data.
- These include
  - Persons: Customer, Employee
  - Places: Building, Room, Campus
  - Objects: Book, Machine, Product
  - Events: Invoice, Order, Registration, Renewal
  - Concepts: Course, Fund, Stock

UCSC

Elements in a DFD
(Gane and Sarson Symbols)

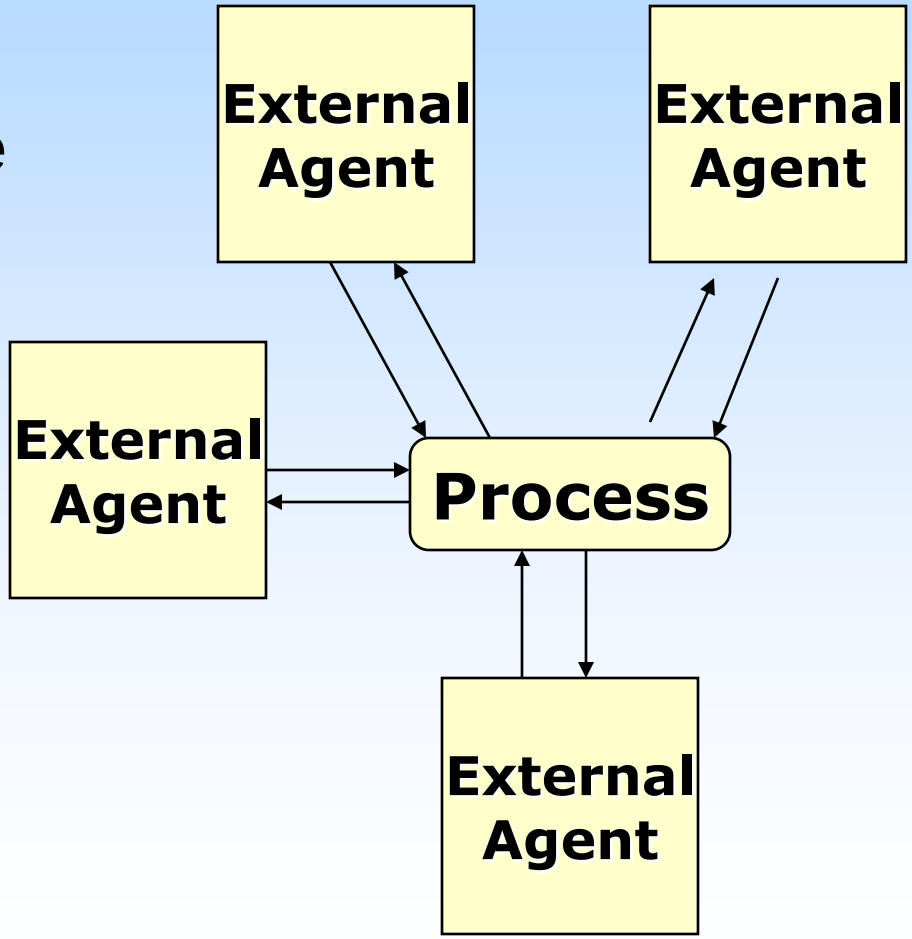


A Data Flow

- Represent inputs or outputs, to or from the processes.
- The arrow head indicates the direction of data flow.
- Label the arrows with the name of the data that moves through it.
- Data in motion

The Context Data Flow Diagrams

- A diagram that shows the system as a “black box” and its main interfaces with its environment.
- Used to document the scope of the system
- Also known as environmental model.



The Context Data Flow Diagrams

- Used to clarify and agree the scope of the investigation
- Shows the interfaces between the system under investigation and the external agents with which it communicates
- Subject to constant change
  - **Because the scope of any project is always subject to change**

The Context Data Flow Diagrams

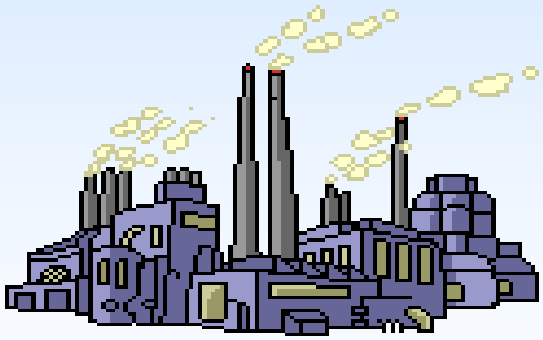
- Can be drawn without considering the **Document Flow Diagram**
- Need to identify
  - the **data flows** and
  - the **external agents** needed for the context diagram

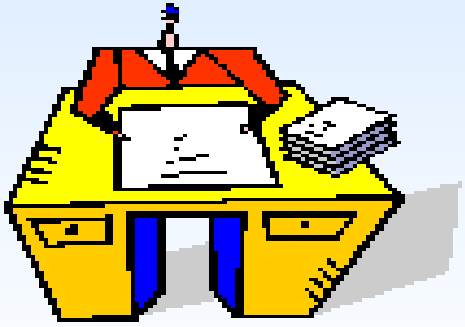
The Context Data Flow Diagrams

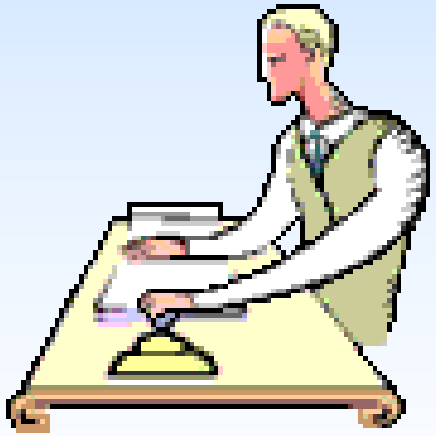
- Think the system as a container
- Distinguish the inside from the outside
- Ignore the inner workings of the container
- Find out the net inputs to the system
  - Business transactions a system must respond to
- For each net input determine its source (External Agents)
- Find out the net outputs from the system
  - Responses produced by the system
- For each net output find the destination (External Agents)
- Identify any external data stores,
  - Files or databases of other systems

Task 1

Identify all sources and recipients of data to/from the system.

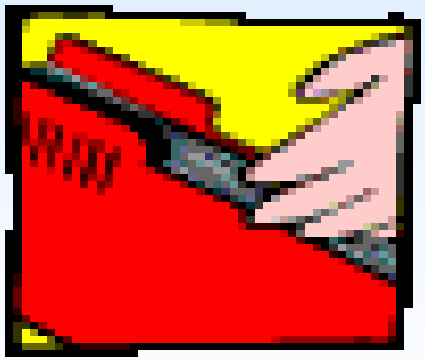


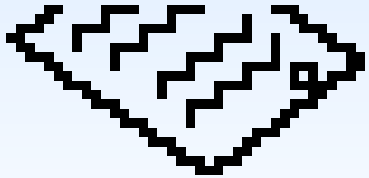




Task 2

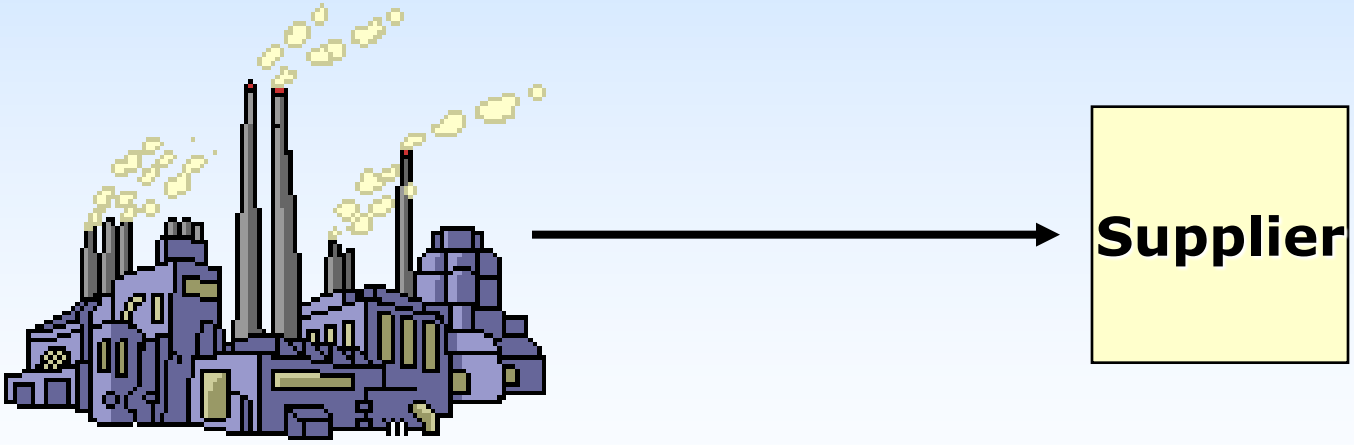
- Identify major data flows to and from the System





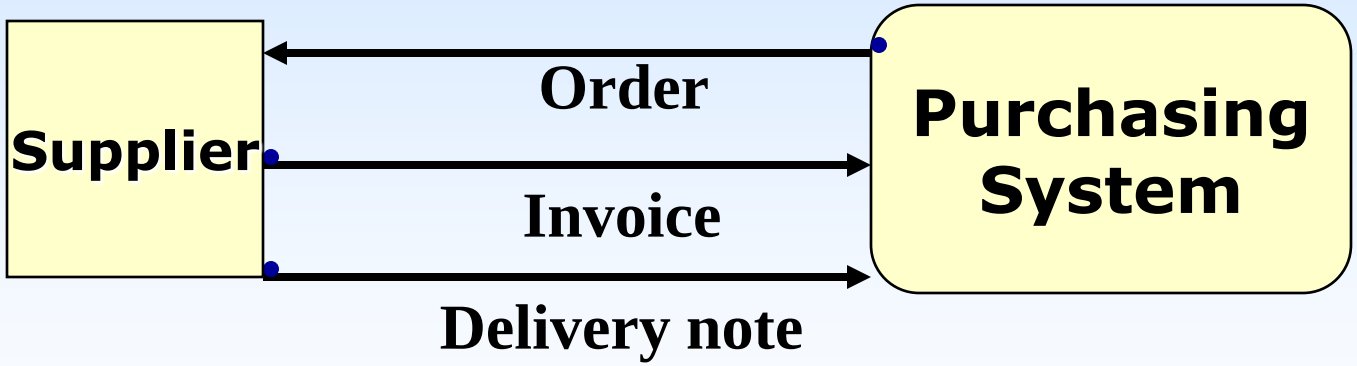
Task 3

- Convert each source or recipient into external entities



Task 4

- Add the data flows between each external entity and the process representing the entire system.



Data Flow Diagrams

- Draw Context Diagram
- Level 0 (Top Level) Data Flow Diagram
- Level 1 Data Flow Diagram
- Continue up to elementary functions

Bank Payment System

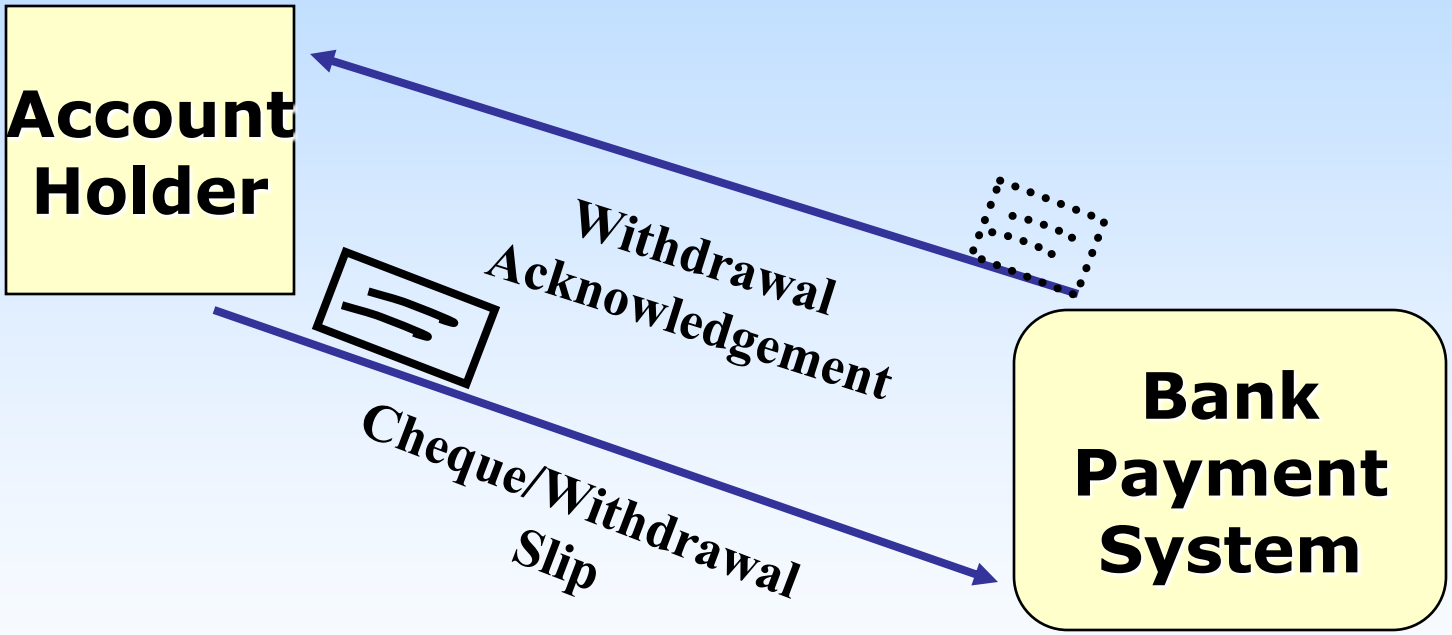
❖ Consider a system in a bank whereby account holders get their withdrawals effected.

Whenever an account holder wants to withdraw some cash, he presents a cheque or withdrawal slip.

The account is checked for the appropriate balance.

If balance exists, the cash is paid and the account is updated.

Context Diagram

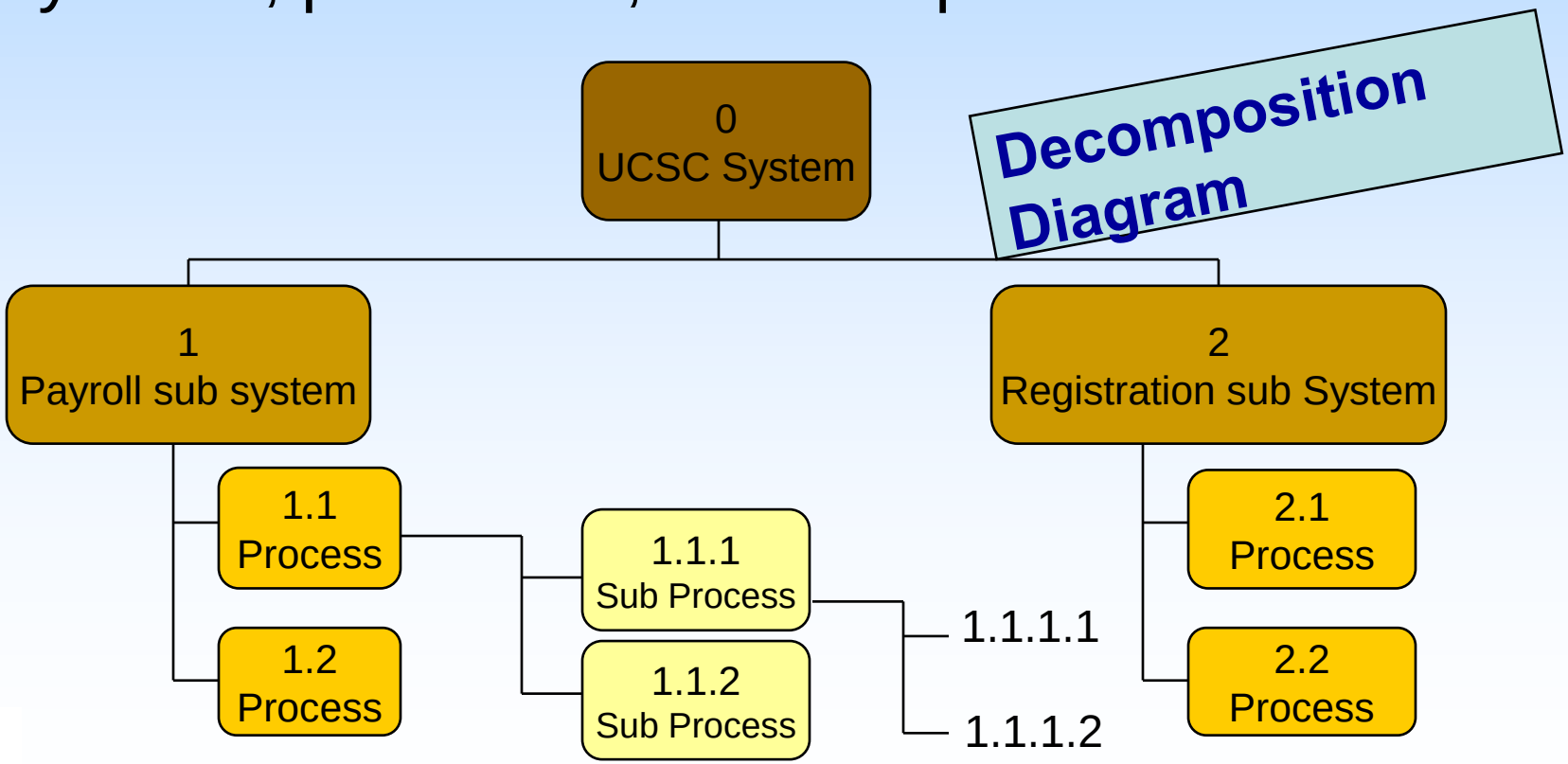


Decomposition

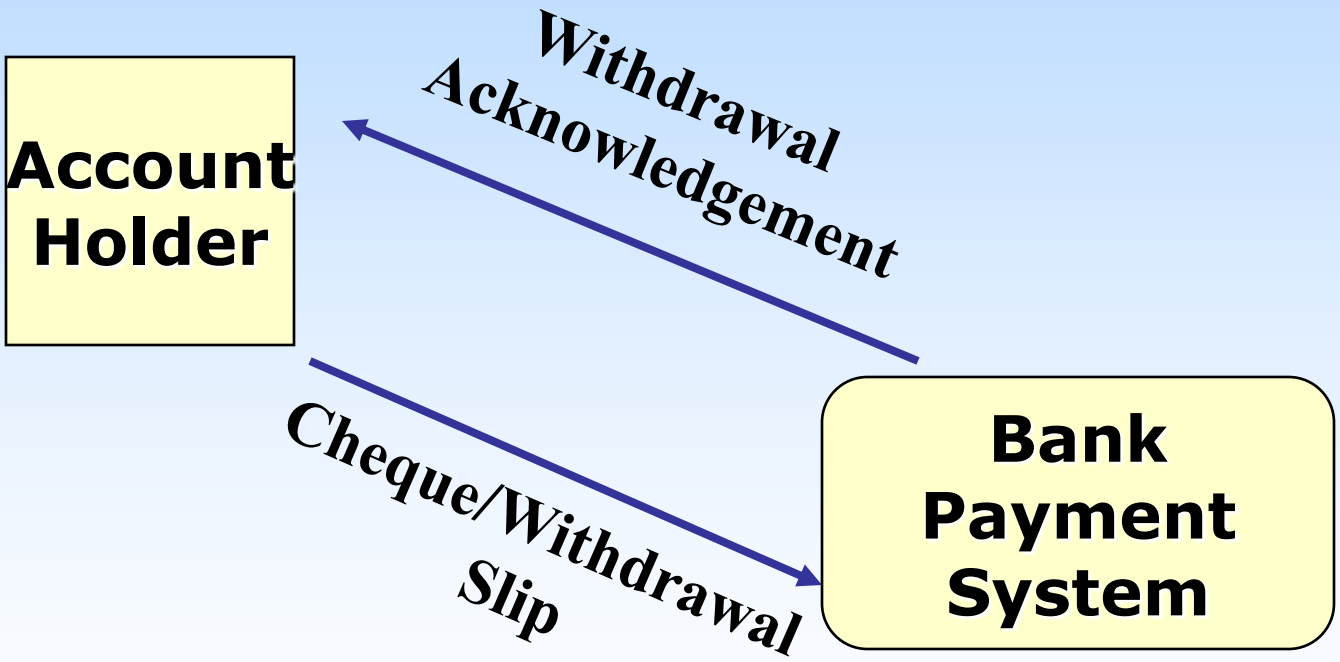
- Is the act of breaking a system into its component subsystems , processes and sub processes.
- Top level function is then decomposed to its component functions

Process Decomposition

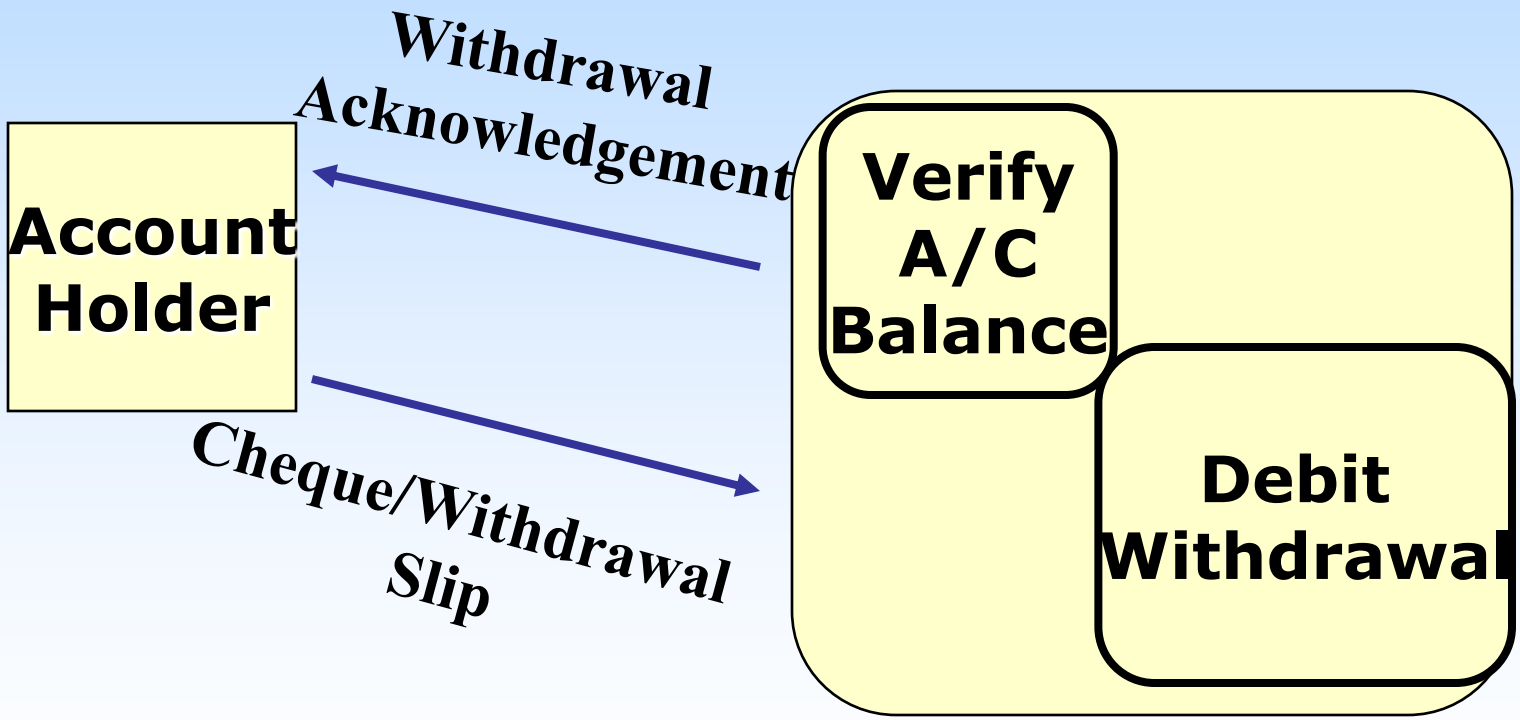
Is an act of breaking a system into its component subsystems, processes, and sub processes.



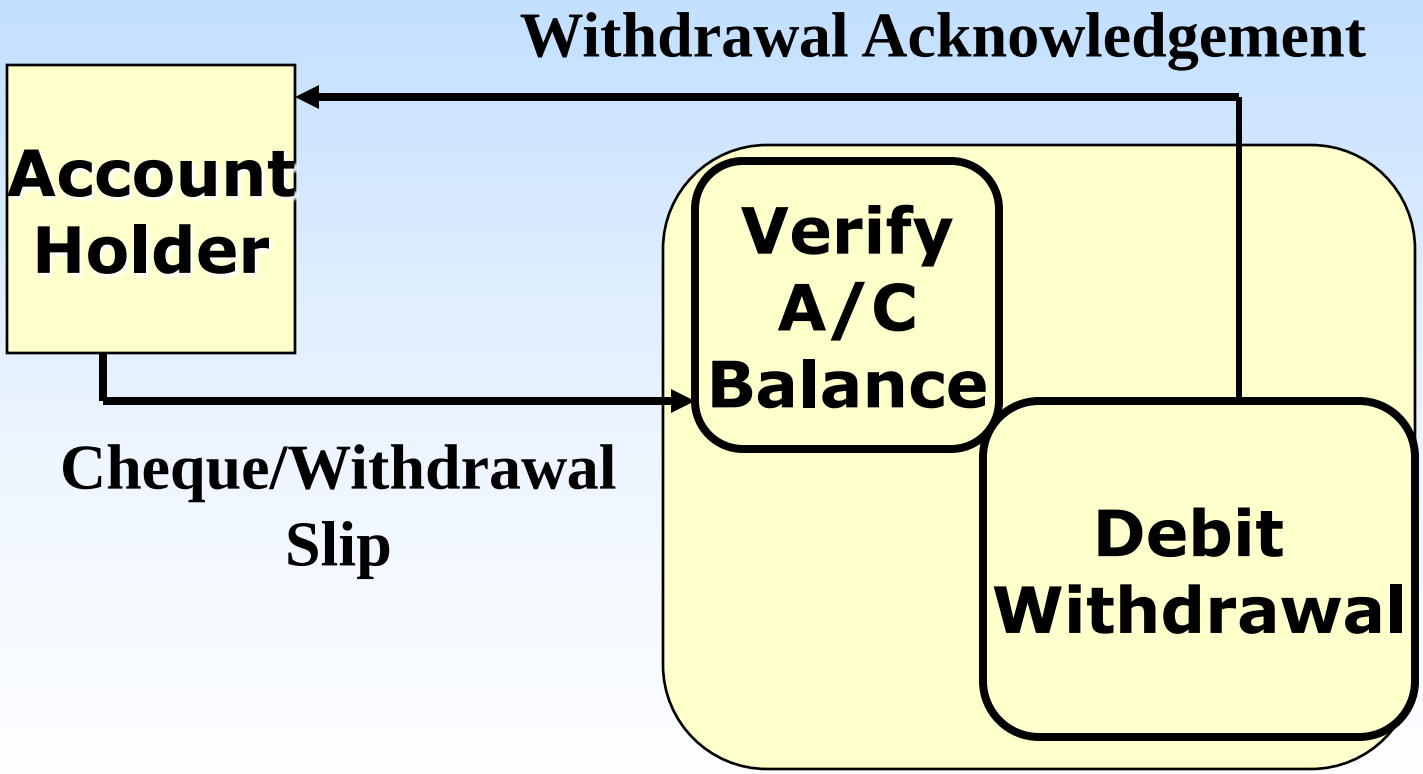
Context Diagram



Top Level DFD – Step 1



Top Level DFD – Step 2



Top Level DFD – Step 3

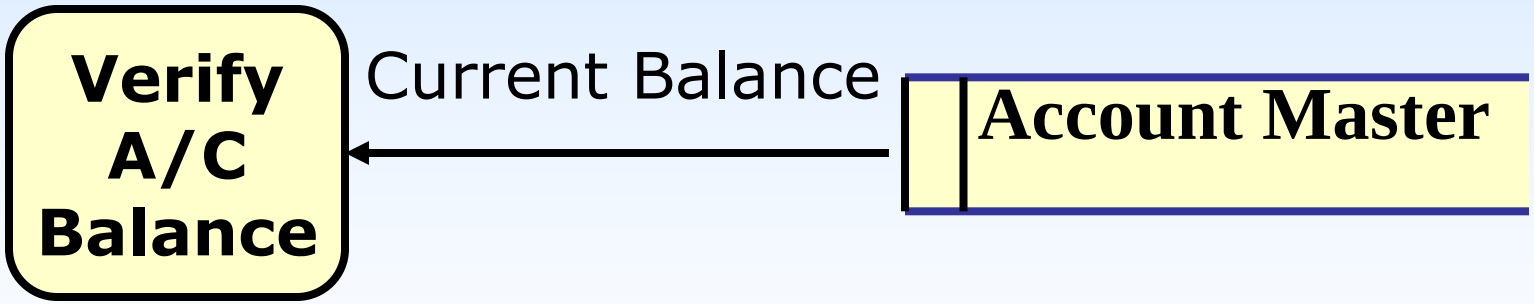
- Identify the Data Stores

Account Master

Top Level DFD – Step 4

- Identify the other data flows.

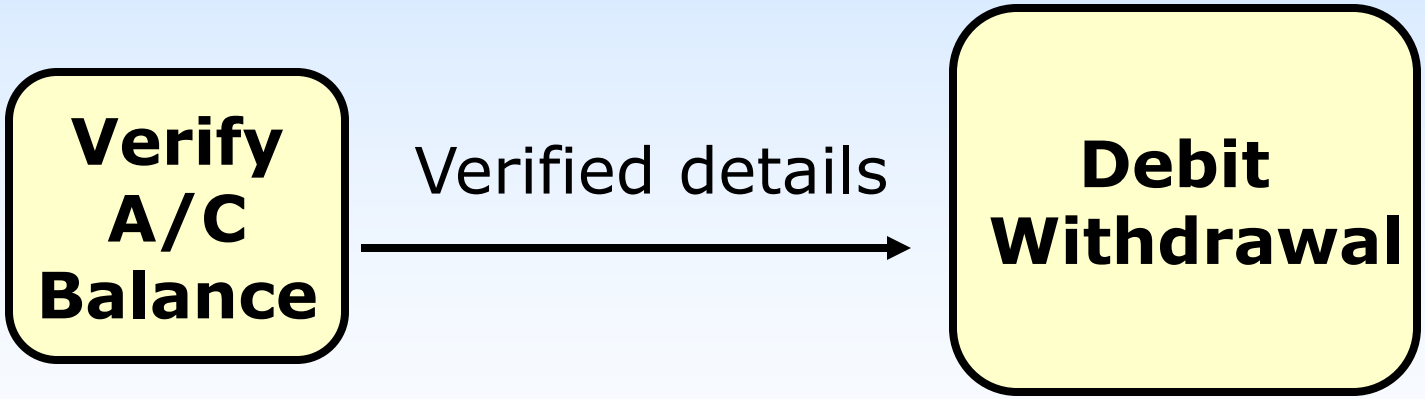
  Get current balance



Top Level DFD – Step 4

- Identify the other data flows.

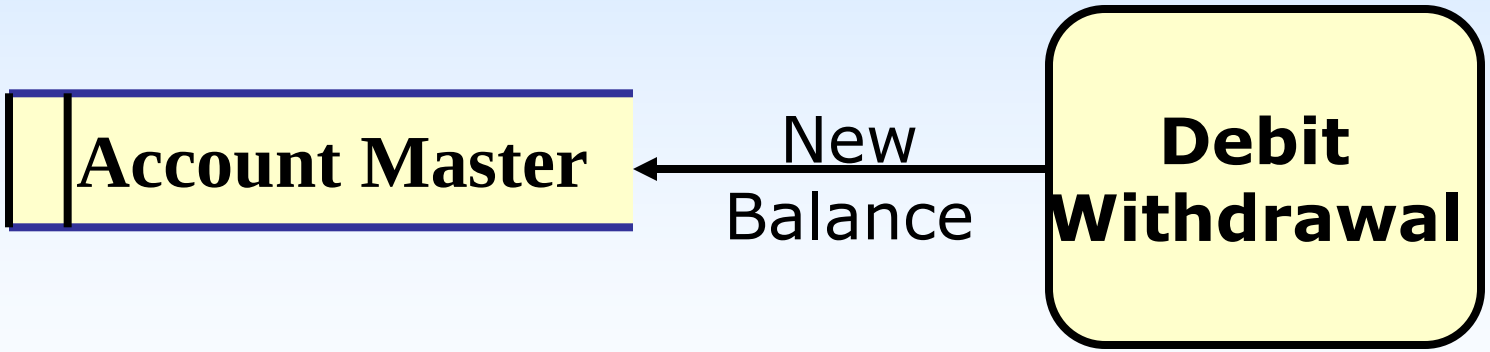
  Transfer the verified details



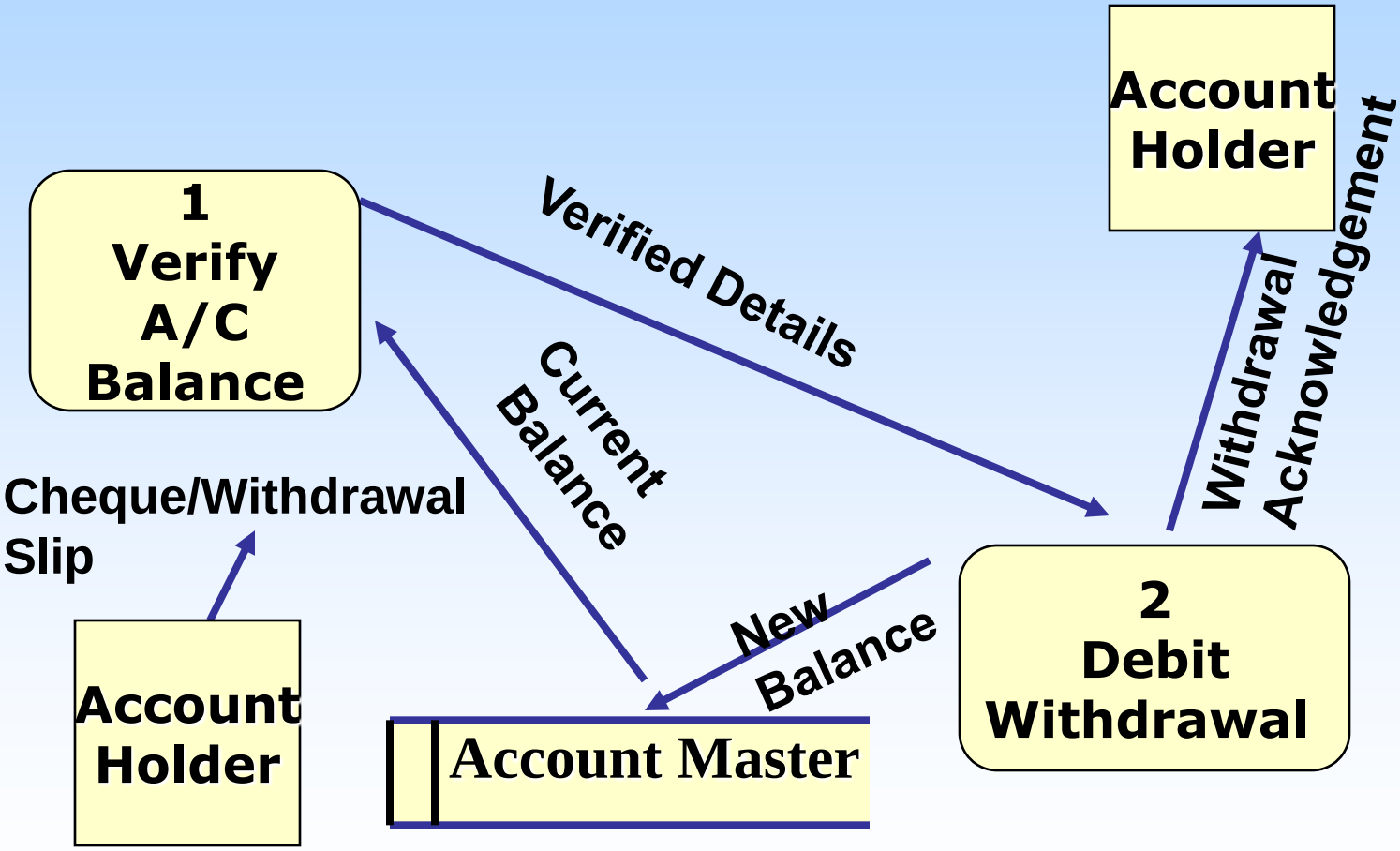
Top Level DFD – Step 4

- Identify the other data flows.

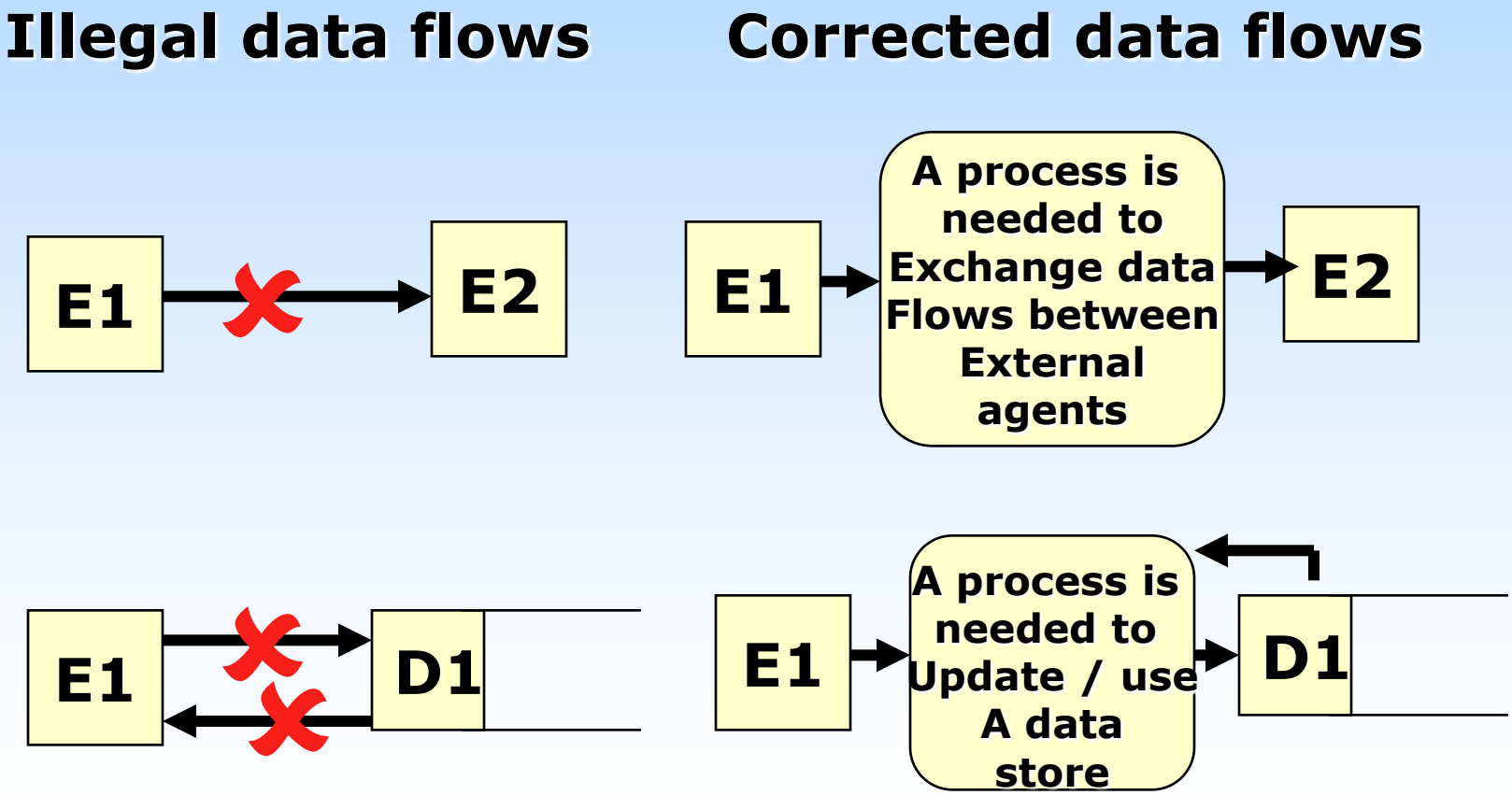
  update new balance



Top Level Diagram

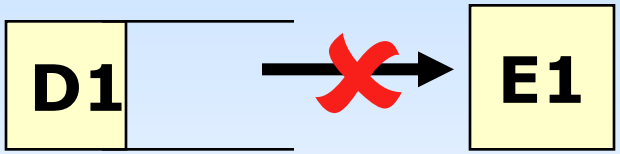


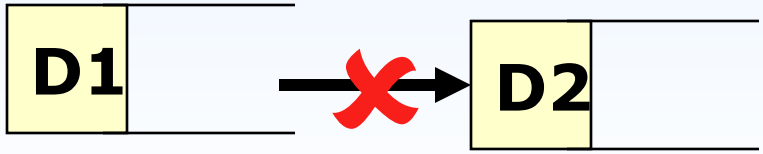
Illegal Data Flows



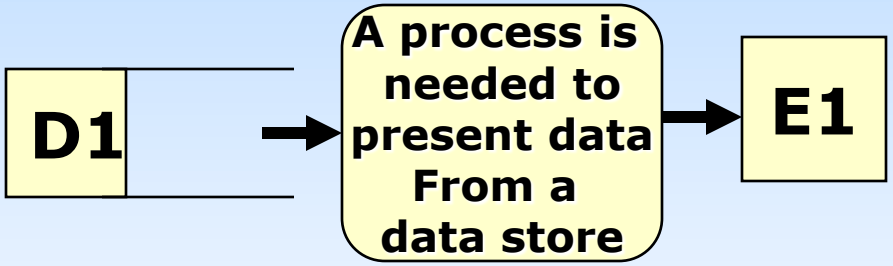
Illegal Data Flows

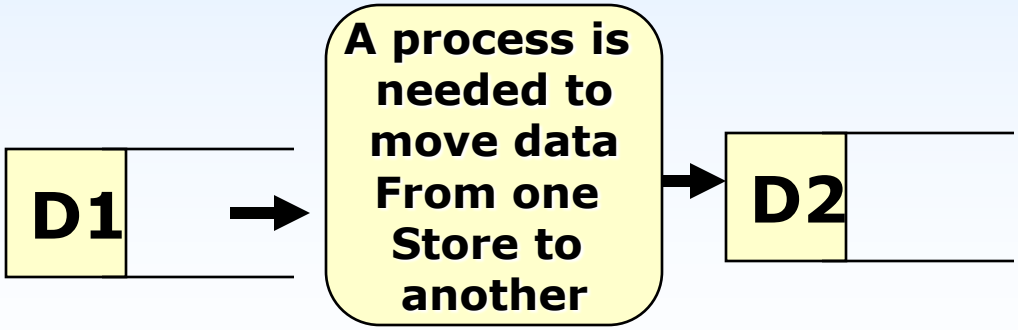
Illegal data flows



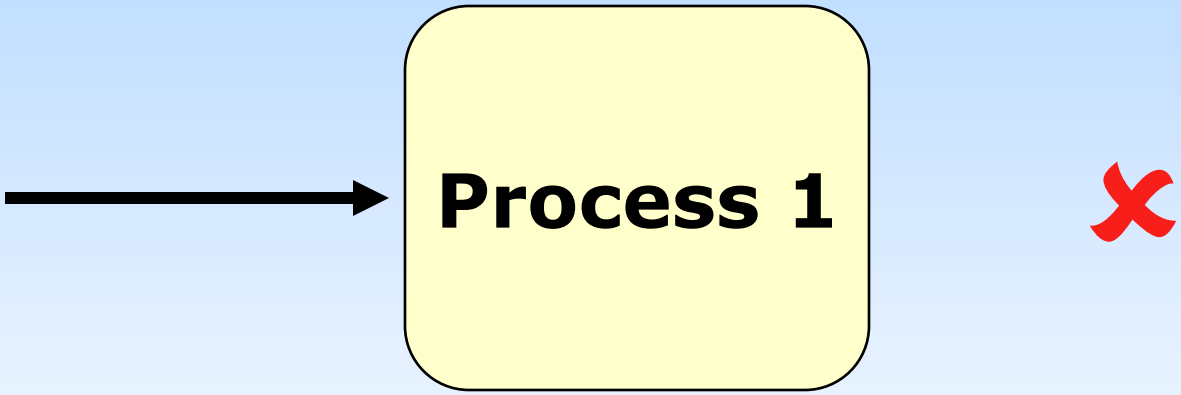


Corrected data flows





Another Common error



No data flow should ever go unnamed

CASE STUDY
Library System

Library System

Library supports

- Lending
- Cataloging
- Registration of Members and Books
- Reservation
- Inquiries
- Correspondence

All activities are done manually

Library System

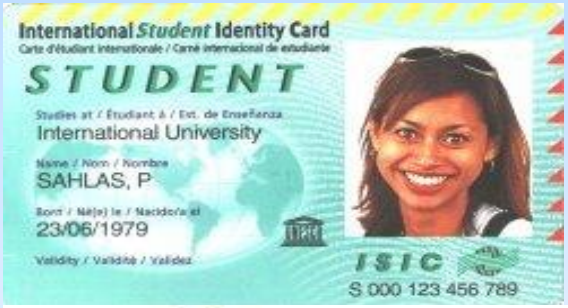
To Analyze and Design a Library System

- What are the Documents in the system?

- Study the physical movements of documents

Library System

- Documents in the system
  - Application form
  - Student Id
  - Membership card
  - Reminders
  - Borrowing slips etc.
  - Reservation ready notice



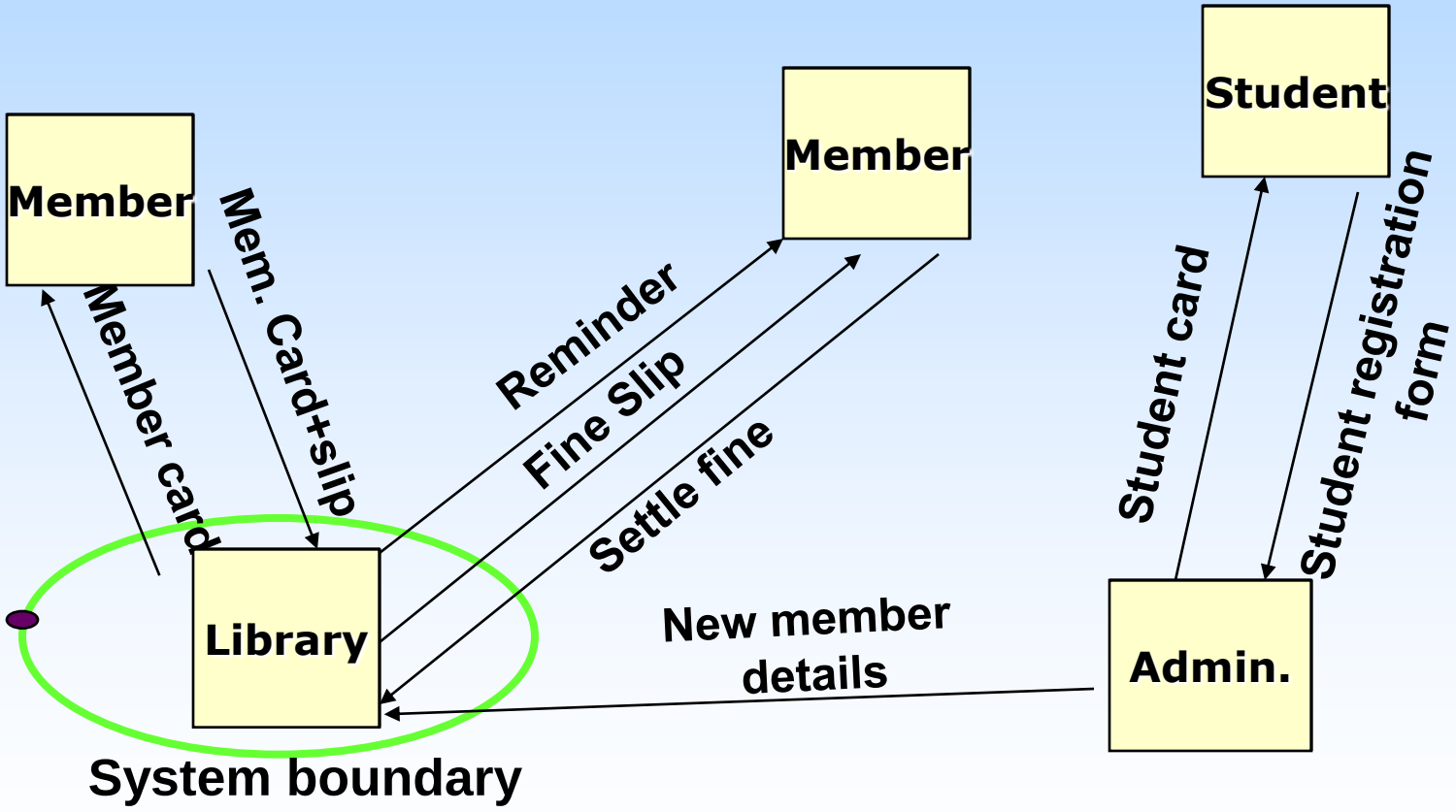
Library System

- Identify the physical movements of documents.
  - **Document Flow Diagram**
    - **Modeling method or technique used to illustrate movements of documents.**

Converting Document Flow Diagrams to DFDs

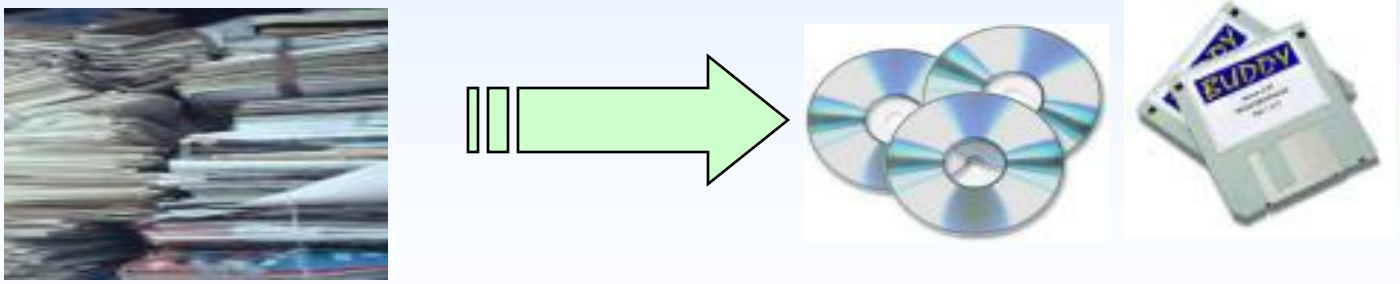
- What process generates this document flow?
- What process receives this document ?
- Is the document stored by a process?
- Where is the document stored?
- Is the document created from stored data?

Document Flow Diagrams for the Library System

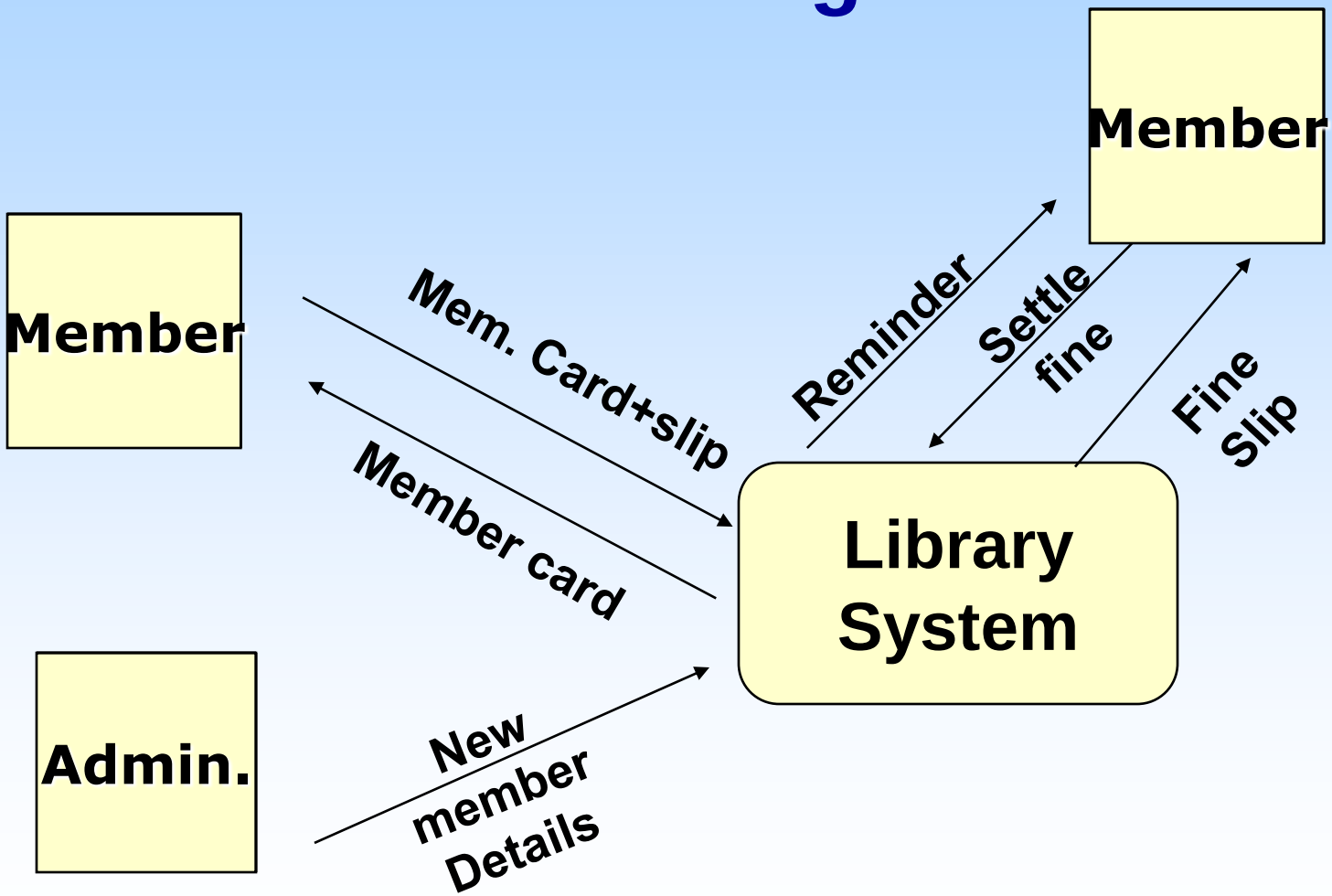


Data Flow Diagrams (DFDs)

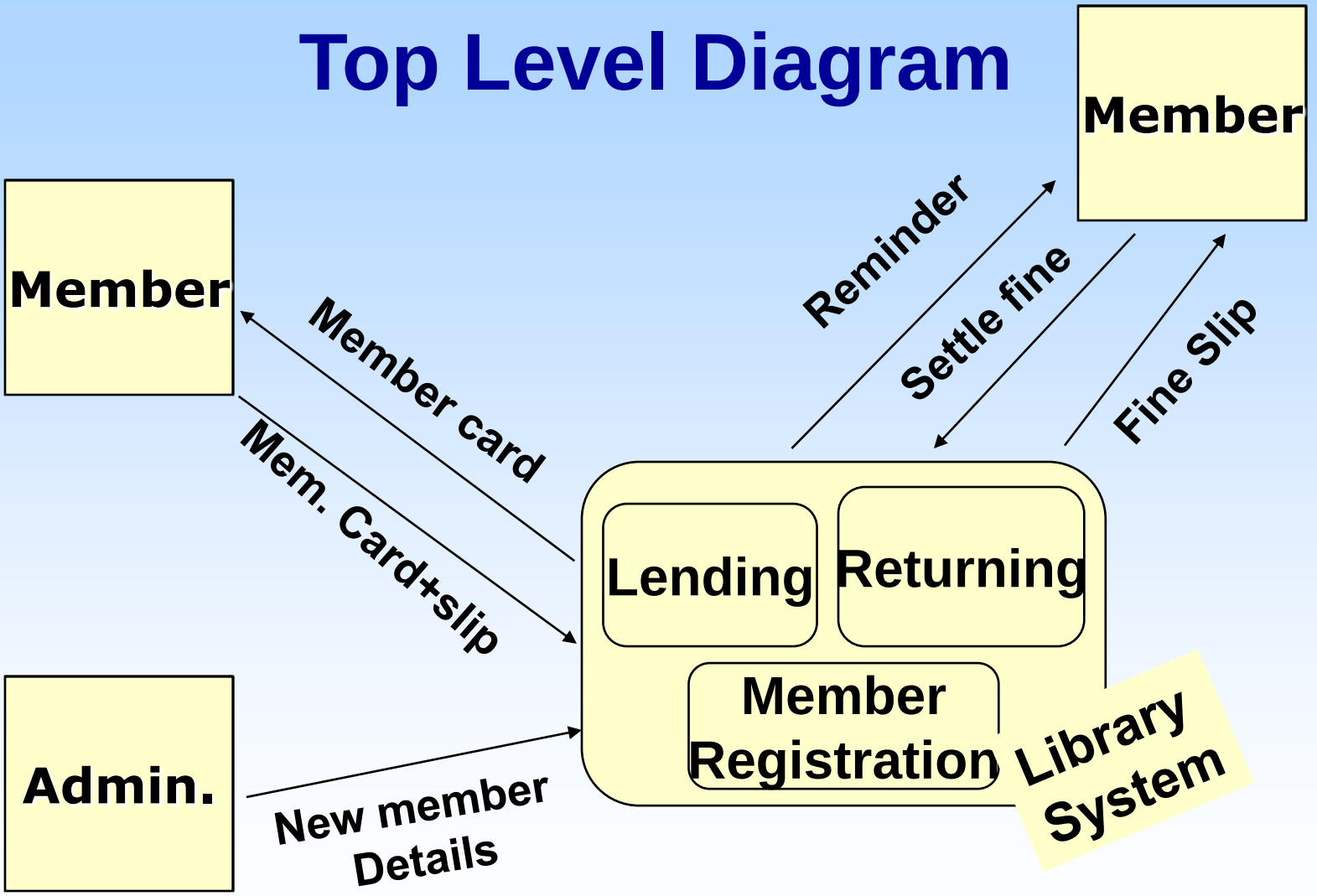
- DFDs handle transformation from physical document to logical data
- Advances in technology mean that electronic means are increasingly supplementing the paper based documents.



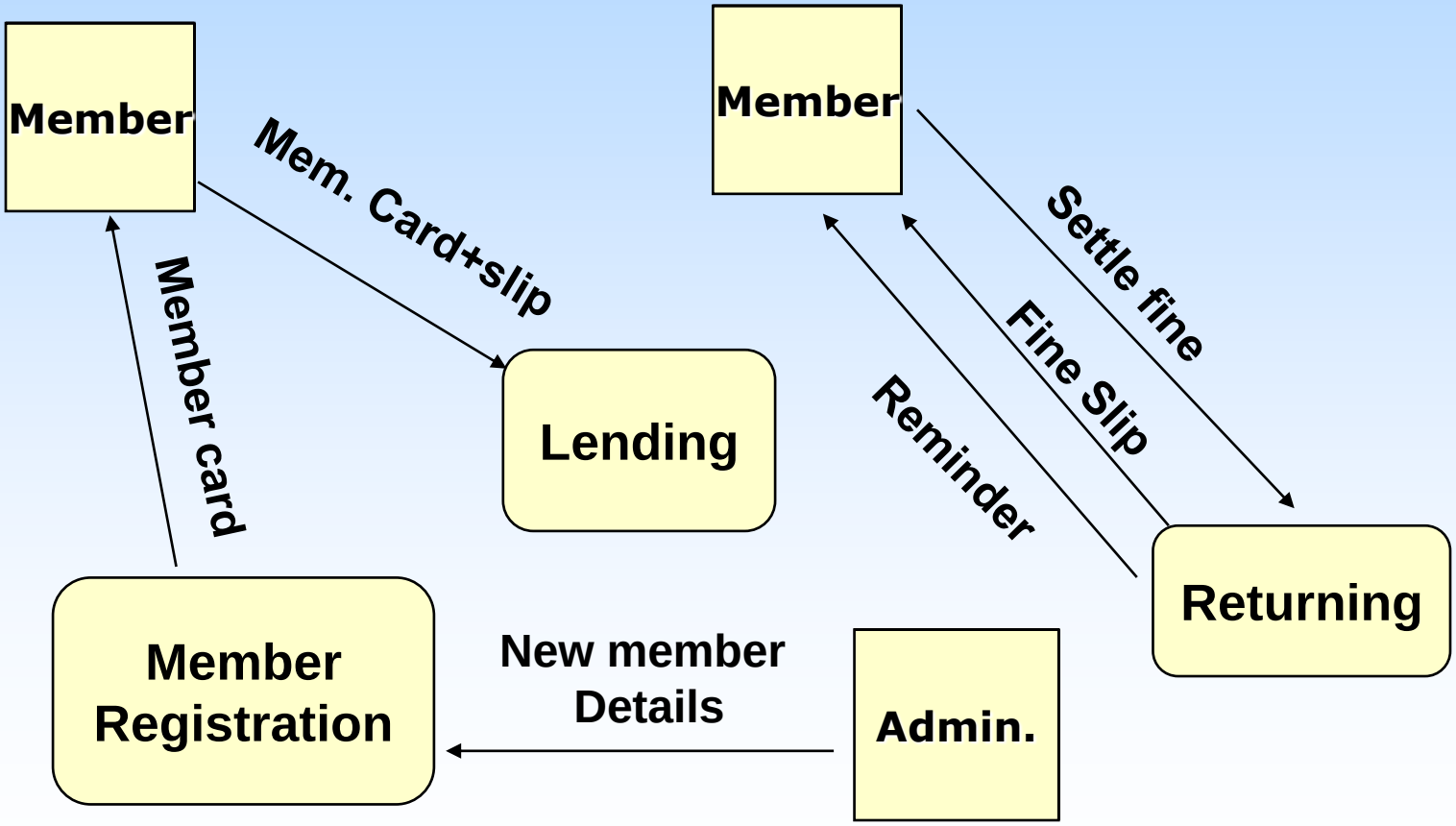
Context Diagram



Top Level Diagram



Top Level Diagram



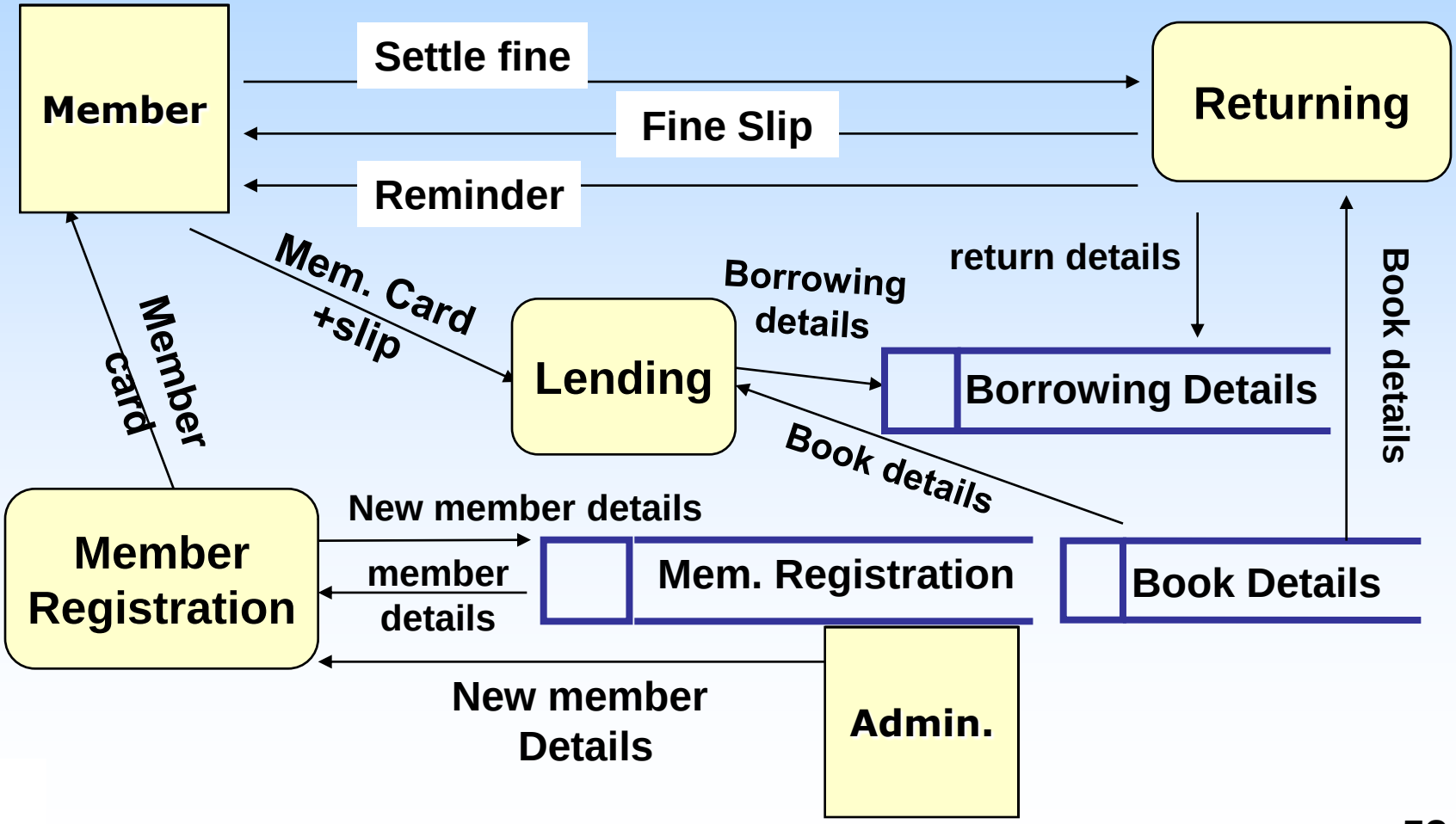
Data Stores

Mem. Registration

Borrowing Details

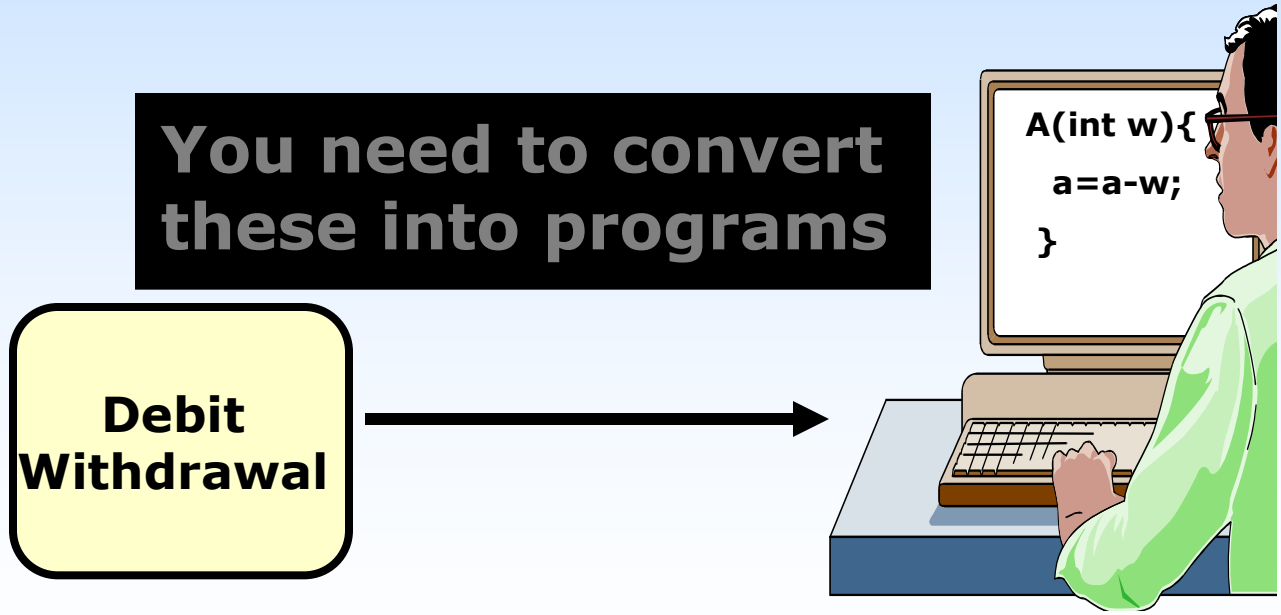
Book Details

Top Level Diagram



Documenting Elements in DFD

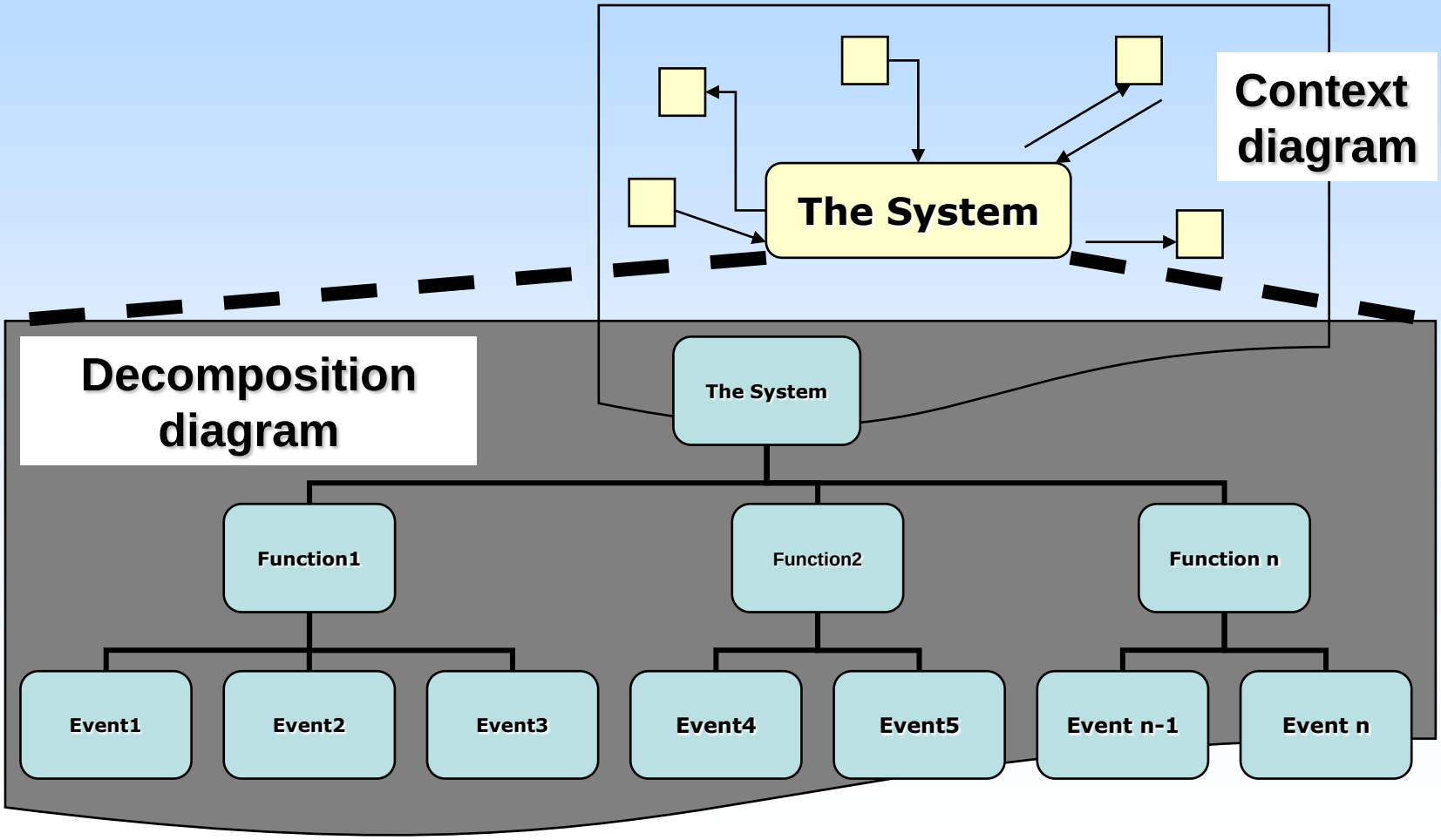
- Element name is not enough.
- More important for processes



The Functional Decomposition Diagram

- Shows the top-down functional decomposition / the structure of the system
- Break the system into its component subsystems , processes and sub processes
- Top level function is then decomposed to its component functions
- Provides an outline for drawing the data flow diagrams

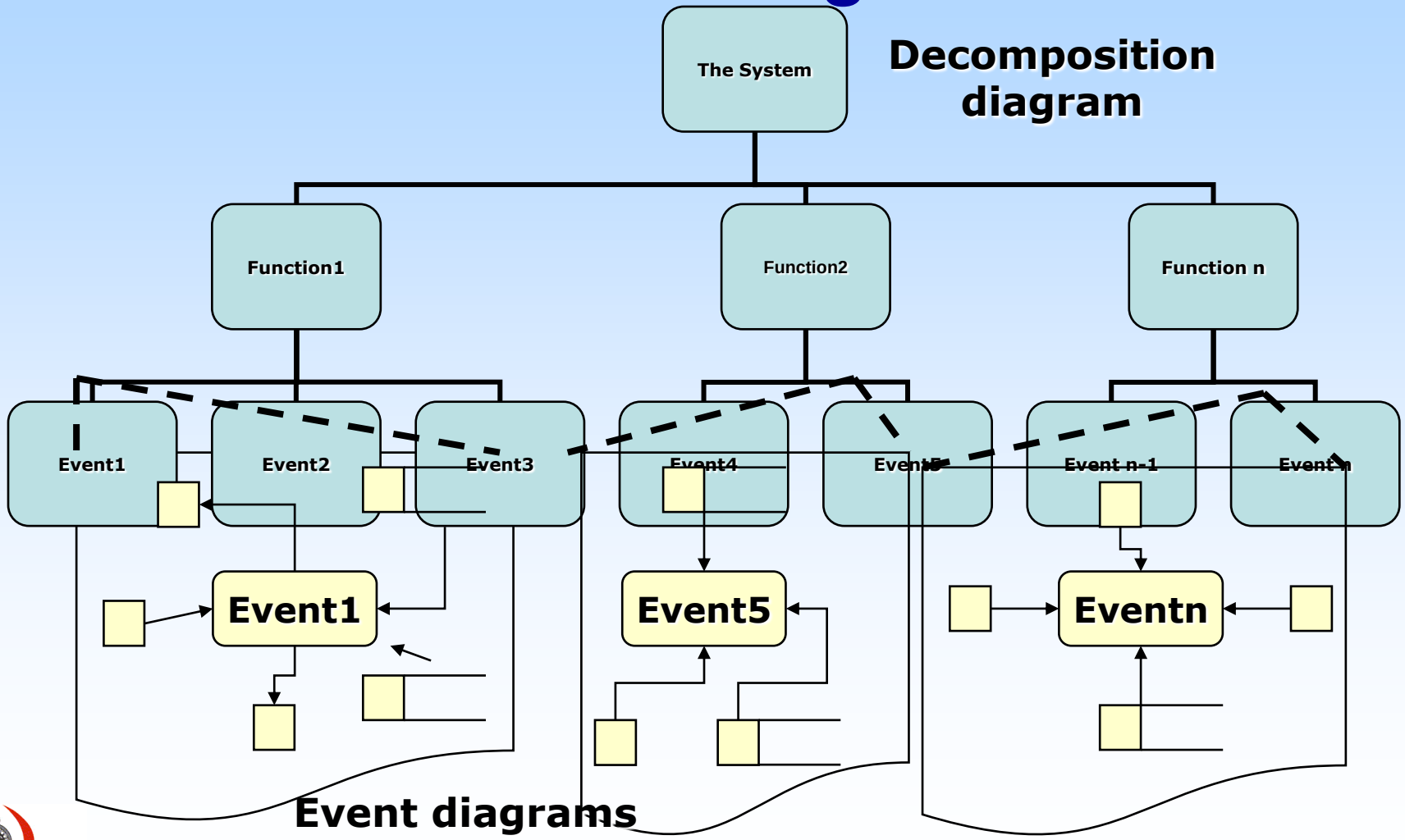
The Functional Decomposition Diagram



Event Diagram

- A data flow diagram that depicts the context for a single event
- Shows the inputs, outputs, and data store interactions for the event.
- Users are not overwhelmed by the overall size of the system
- A powerful communication tool between users and technical professionals

Event Diagram



Event Diagram

- For each event, illustrate the following
  - **The inputs and their sources**
    - **Sources are shown as external agents**
    - **The data structure for each input should be recorded in the repository**
  - **The outputs and their destination**
    - **Destinations are depicted as external agents**
    - **The data structure for each output should be recorded in the repository**
  - **Any data stores from which records must be read**
  - **Any data stores from which records must be created, deleted, or updated**

Process Descriptions

- Structured English
- Decision Table
- Decision Tree

Eg. A Process that has to determine whether a customer is to be given credit

Structured English

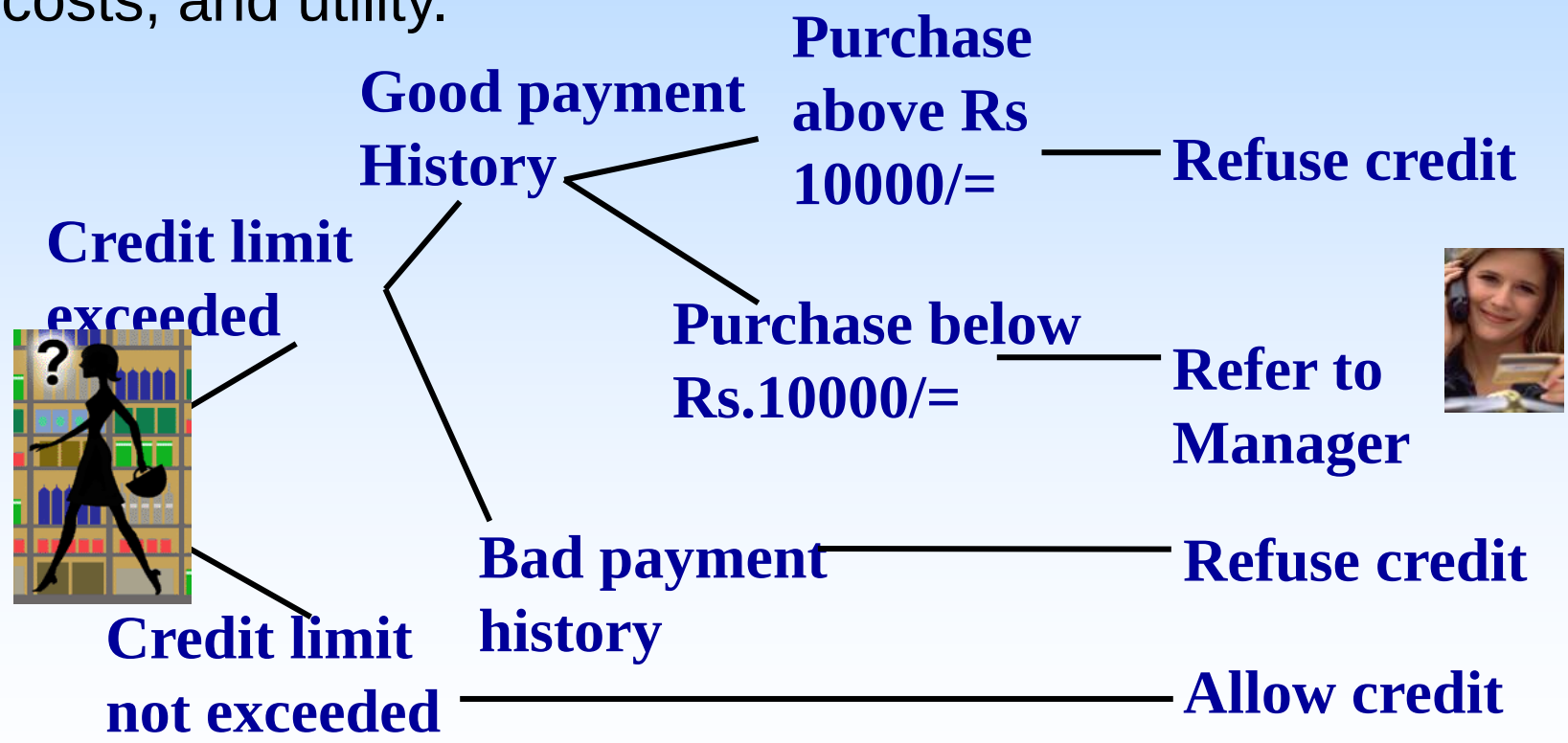
- A language and syntax, based on the relative strengths of structured programming and natural English, for specifying the underlying logic of elementary processes on process models.

Structured English

```
IF credit limit exceeded
THEN
      IF Customer has bad payment history
      THEN refuse credit
      ELSE
        IF purchase above Rs.10000/=
        THEN refuse credit
         ELSE refer to manager
ELSE  allow credit
```

Decision Tree

A graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility.



Decision Table

- A tabular form of representation that specifies a set of conditions and their corresponding actions
- Very useful for specifying complex policies and decision making rules

Decision Table

Y-TRUE
N-NOT TRUE
X-TAKE ACTION

Condition	Credit limit exceeded	Y	Y	Y	Y	N	N	N	N
	Good payment history	Y	Y	N	N	Y	Y	N	N
	Purchase above Rs.10000/=	Y	N	Y	N	Y	N	Y	N
Action	Allow Credit					X	X	X	X
	Refuse	X		X	X				
	Refer Manager		X						

Data Modeling

- A technique for defining business requirements for a database
- Also known as database modeling
- There are several notations
- Actual model is called an ERD – Entity Relationship Diagram
  - **Shows data in terms of the entities and relationships described by the data.**
  - **There exist several notations for an ERD**
  - **Martin notation is widely used.**

Data Modeling…

Entity Relationship Diagrams

Shows data in terms of the entities and relationships described by data.

Entities

An entity is something about which the business needs to store data.

Synonyms – entity type and entity class

Data Modeling…

Entity Relationship Diagrams…

Entity: is a class of

Persons **(Customer, Employee)**

Places **(Building, Room)**

Objects **(Book, Product)**

Events **(Flight, Invoice)**

Concepts **(Account, Fund)**

about which we need to capture and store data.

Data Modeling…

Entity Relationship Diagrams…

Entity Instance

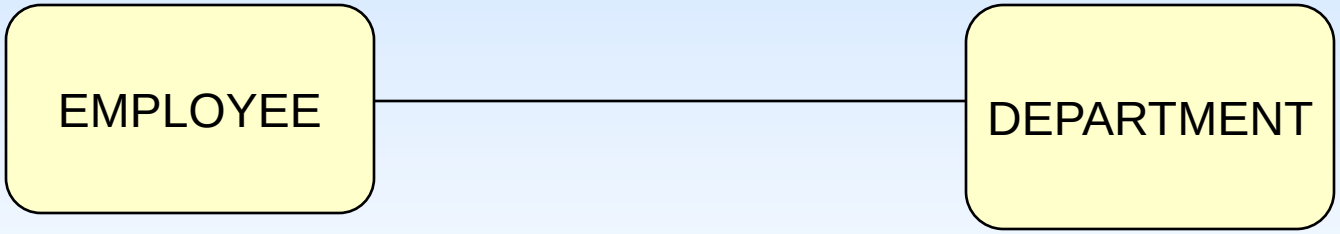
An entity instance is a single occurrence of an entity. Every entity must have an identifier or key to uniquely identify each instance.

Data Modeling…

Entity Relationship Diagrams…

Symbol:

Consider Martin notations.



The named rounded rectangle represent the entity. –

A line represent the relationship. –

Data Modeling…

Entity Relationship Diagrams…

Attribute:

is a descriptive property or characteristics of an entity. Sometimes called as element, property, or field.

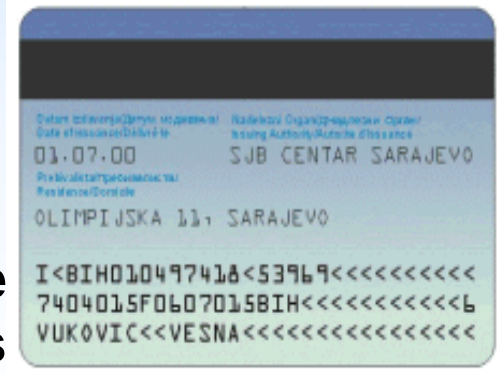
Data Modeling…

Entity Relationship Diagrams…

EMPLOYEE
NIC_NO
Name
Address
Age
….
….
….

A key is an attribute, or group of attributes that assumes a unique value for each entity instance. It is sometimes called an *identifier*.

This person can be identified using his ID number.



Data Modeling…

Entity Relationship Diagrams…

Compound Attribute is one that actually consist of other attributes.

Synonyms- composite attribute, concatenated attribute

Example :

Student name

Last Name
First Name
Middle Name

Address

Street Address
Postal Code
Country
City

Data Modeling…

Entity Relationship Diagrams…

The values for each **attribute** are defined in terms of three properties:

1. Data type – What type of data can be stored in that attribute (Number, Date, Text etc).
2. Domain – What values an attribute can legitimately take on.

   Refer to table 8-2 in pg 273 Ref1

3. Default – Is the value that will be recorded if not specified by the user.

Data Modeling…

Entity Relationship Diagrams…

Relationships

Natural business association that exists between one or more entities

E.g.. A Current Student is enrolled in one or more curricula

Data Modeling…

Entity Relationship Diagrams…

Cardinality

Defines the minimum and maximum number of occurrences of one entity that may be related to a single occurrences of the other entity.

Exactly one

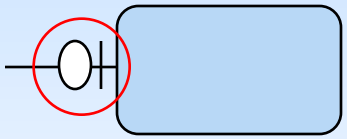
or

Data Modeling…

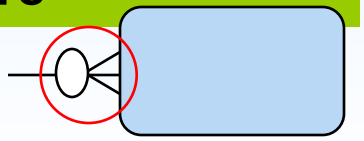
Entity Relationship Diagrams…

Cardinality

Zero or one



Zero, one or more

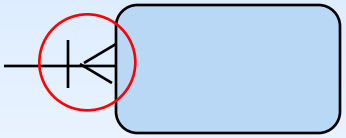


Data Modeling…

Entity Relationship Diagrams…

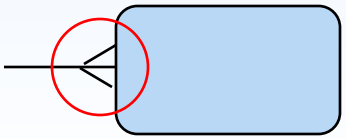
Cardinality…

One or more

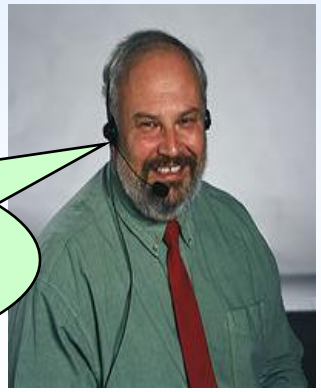


I have to work at least in one, or more projects.

More than one



I am working on many projects.



Data Modeling…

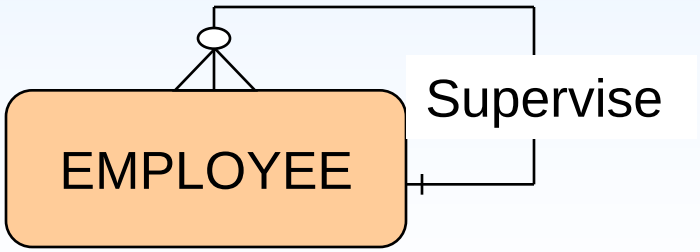
Entity Relationship Diagrams…

Degree

Number of entities that participate in the relationship

Degree =1

Recursive Relationship – Relationship that exists between different instances of the same entity.



Data Modeling…

Entity Relationship Diagrams…

Degree…

Degree =2

Binary Relationship - When two different entities participates in a relationship

Data Modeling…

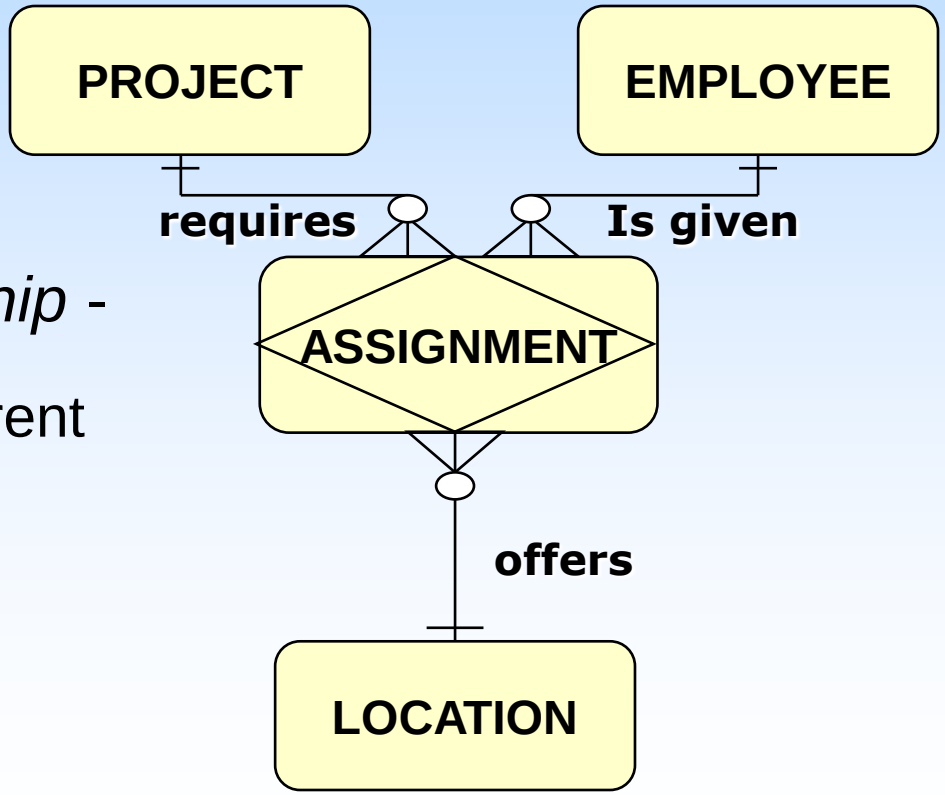
Entity Relationship Diagrams…

Degree…

Degree =3

Ternary or 3-ary Relationship -

When more than two different entities participates in a relationship.



Synchronization of System Models

- Data and process models represent different views of the same system
- These views are interrelated
- Thus, modelers need to synchronize the different views to ensure consistency and the completeness of the total system specification.

Synchronization is the process of maintaining consistency between the different types of models

Object Modeling

- A technique for identifying objects within the systems environment and identifying the relationships between those objects.
- Object Modeling techniques prescribe the use of methodologies and diagramming notations that are completely different from the ones used for data modeling and process modeling.

Object Modeling Methods

- In the late 80s and early 90s
  - Booch Method – Grady Booch
  - Object Modeling Technique (OMT) – James Rumbaugh
  - Object-Oriented Software Engineering – Ivar Jacobson
- To avoid problems of having many different methods, In 1997,
  - Unified Modeling Language (UML) - Grady Booch, James Rumbaugh, Ivar Jacobson

System Concepts for Object Modeling

- Objects
  - **Something that is or is capable of being seen, touched, or otherwise sensed and about which users store data and associate behavior**
  - **Types of objects**
    - **Person – e.g. employee, customer, instructor, student**
    - **Place – e.g. warehouse, building, room, office**
    - **Thing – e.g. product, vehicle, computer, videotape**
    - **Event – e.g. an order, payment, invoice, application**
    - **Sensual – e.g. phone call, meeting**

System Concepts for Object Modeling…

- Attributes
  - The data that represents characteristics of interest about an object

    e.g. Object : Customer

    Attributes : Customer no, first name, last name, home address, work address, contact no,…etc.

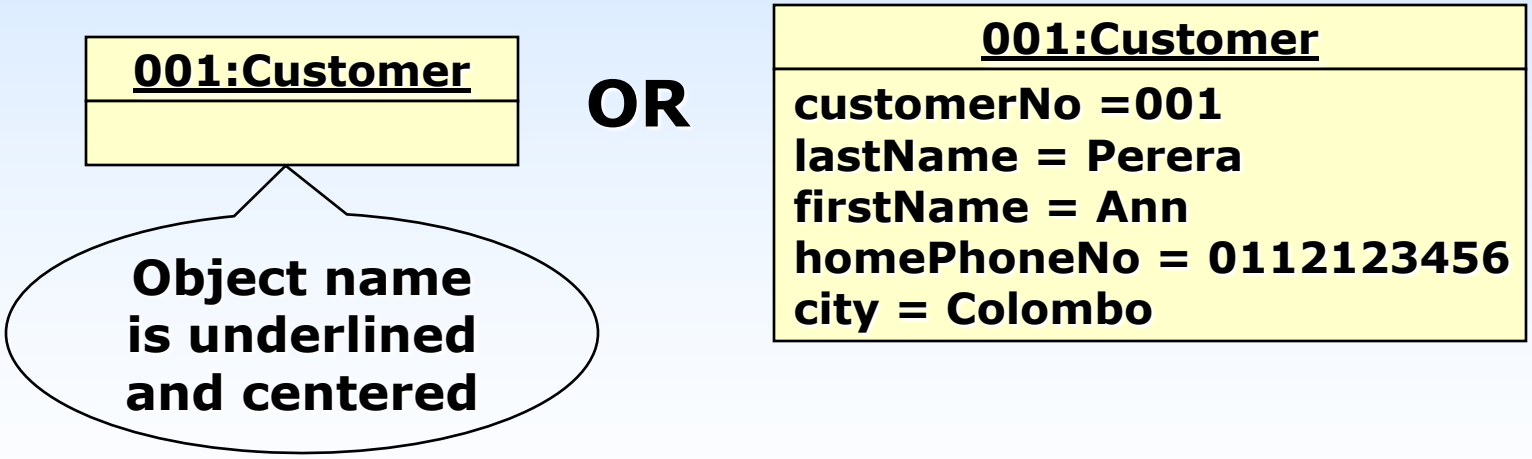
System Concepts for Object Modeling…

- Object instance
  - **Each specific person, place, thing, or event, as well as the values for the attributes of that object.**
  - **Sometimes referred to as an Object.**
  - **Drawn using a rectangle with the name of the object instance**
  - **The name consists of the attribute that uniquely identifies it, followed by a colon and then the name of the class in which the object has been categorized.**

System Concepts for Object Modeling…

- Object instance

  e.g. A “CUSTOMER” Object Instance



System Concepts for Object Modeling…

- Behavior
  - The set of things that an object can do and that correspond to functions that act on the object's data or attributes.
  - Also known as a method, operation or service

e.g. Object : Door

behavior : open, shut, lock or unlock

System Concepts for Object Modeling…

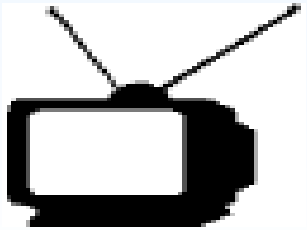
- Encapsulation
  - **Packaging of several items together into one unit (both attributes and behavior of the object)**
  - **The only way to access or change an object's attribute is through that object's specific behavior.**
  - **Objects *encapsulates* what they do.**
    - **That is, they hide the inner workings of their operations**
      - **from the outside world**
      - **and from other objects**

System Concepts for Object Modeling…

Encapsulation

When an object carries out its operations, those operations are hidden.

E.g. When most people watch a television show,
- they usually don't know or care about the complex electronics that sit in back of the TV screen
- or the operations that are happening.

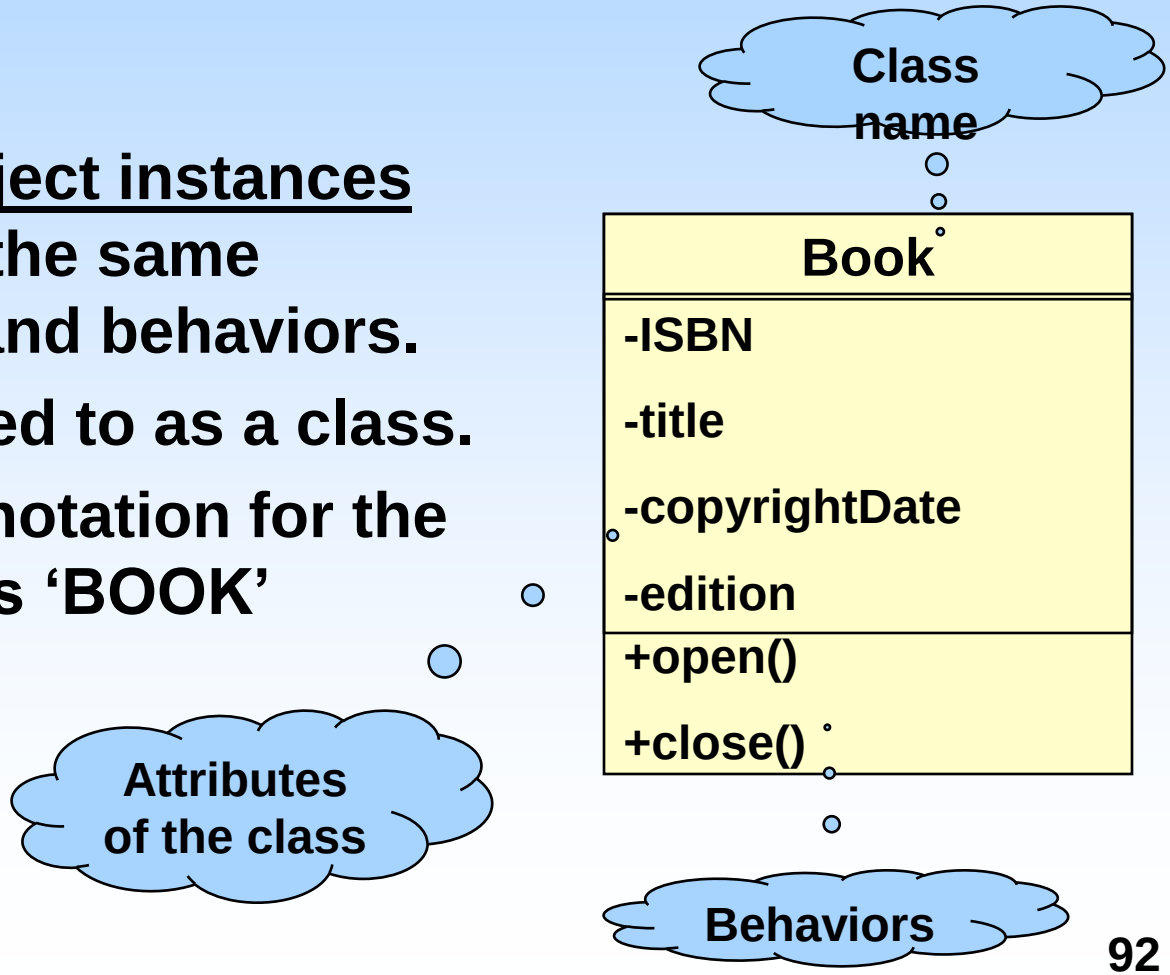


The TV hides its operations from the person watching it.

System Concepts for Object Modeling…

- Object class
  - **A set of <u>object instances</u> that share the same attributes and behaviors.**
  - **Also referred to as a class.**

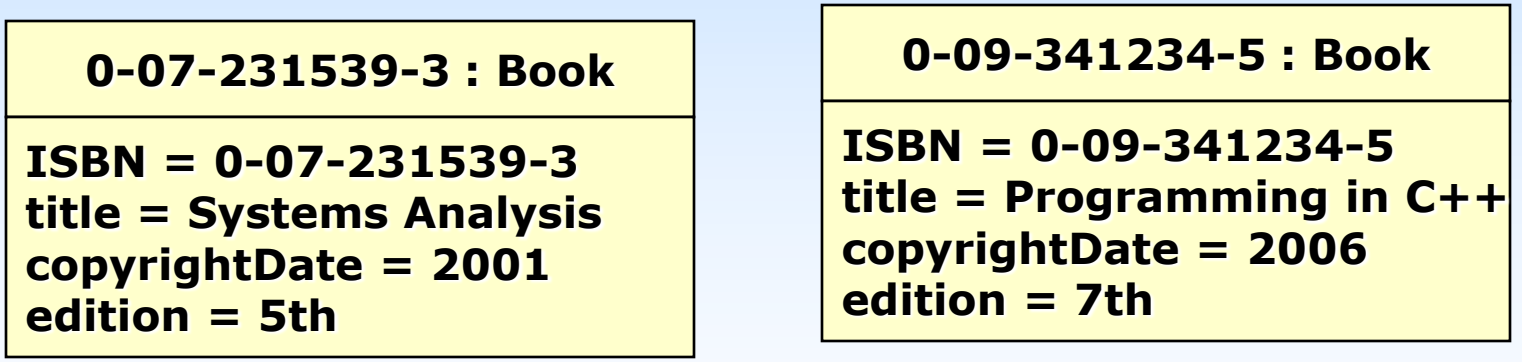
    e.g. UML notation for the object class 'BOOK'



System Concepts for Object Modeling…

An Object instance

e.g.



System Concepts for Object Modeling…

- Inheritance
  - The concept wherein methods and/or attributes defined in an object class can be inherited or reused by another object class.

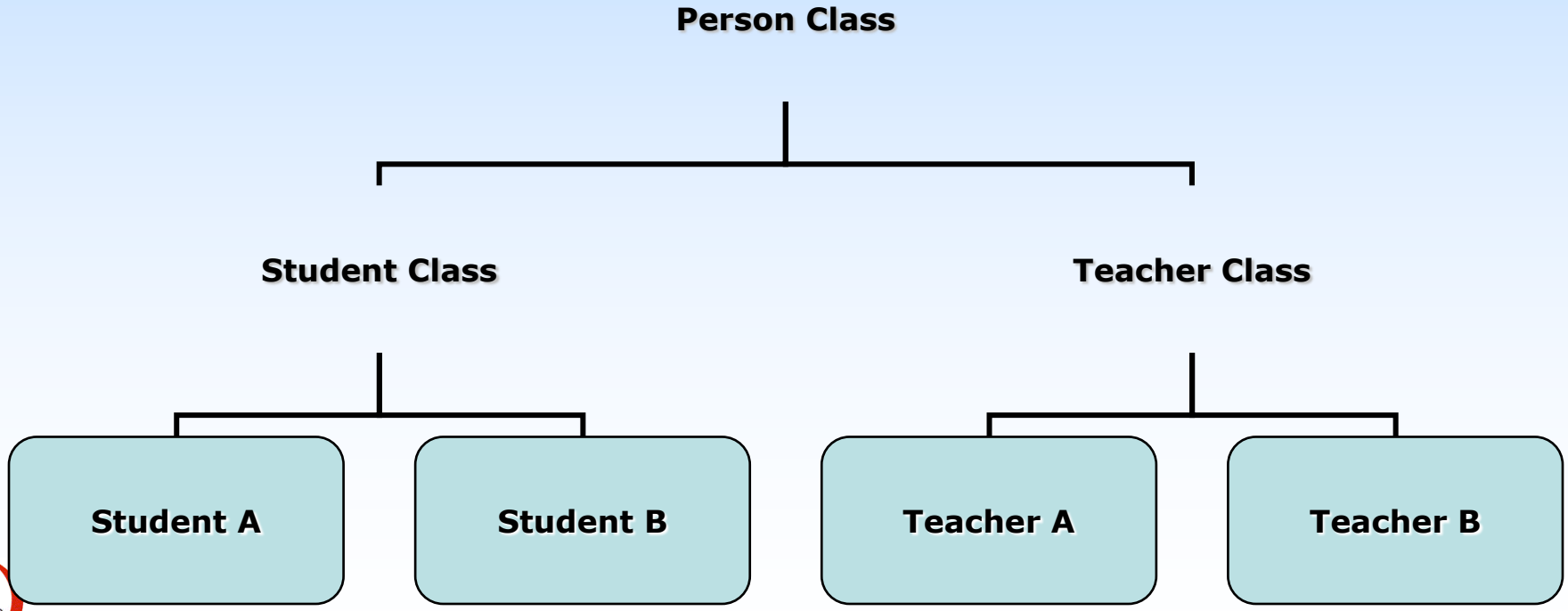
    e.g. some individuals in the room might be classified as STUDENTS and TEACHERS.

    Thus, STUDENT and TEACHER object classes are members of the object class PERSON

System Concepts for Object Modeling…

- Inheritance

  e.g. Cont…



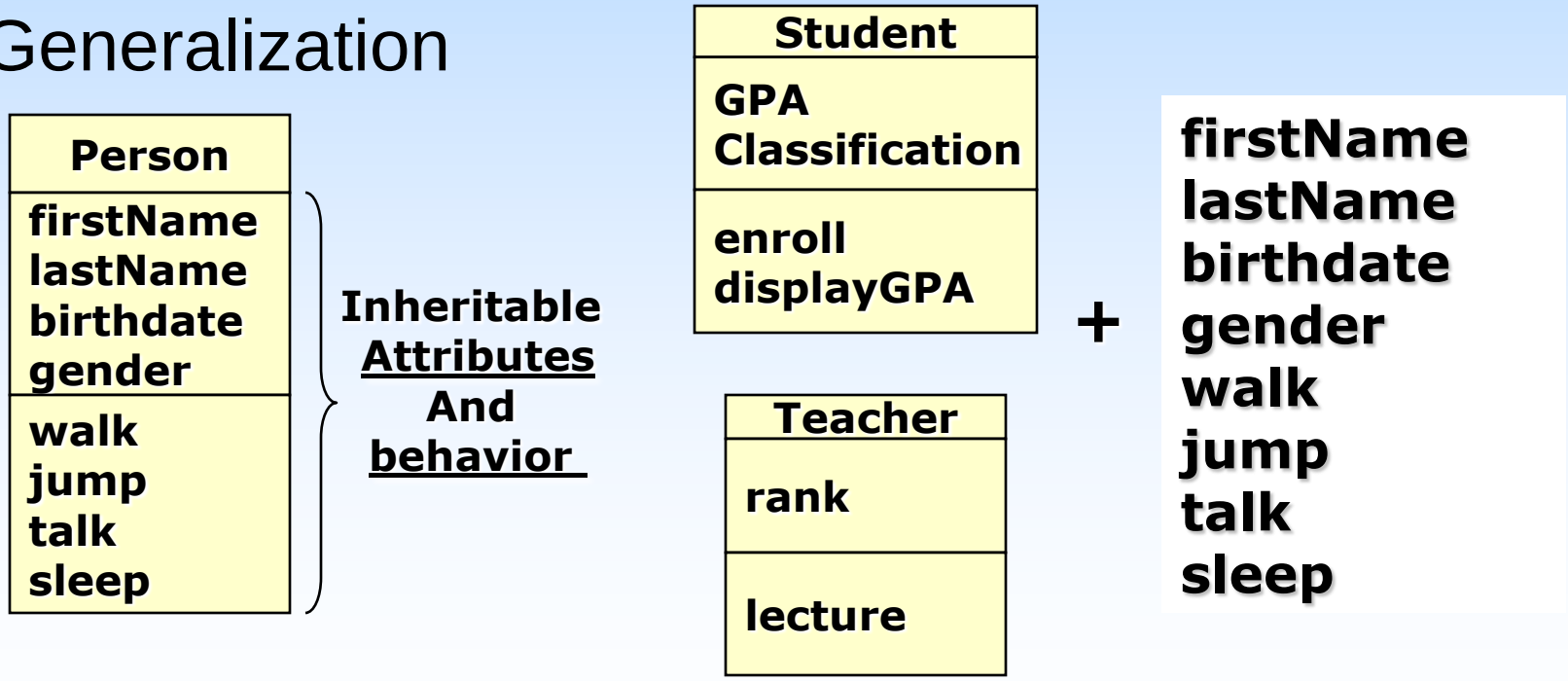
System Concepts for Object Modeling…

- Generalization / Specialization
  - A technique wherein the attributes and behaviors that are common to several types of object classes are grouped / abstracted into their own class called a super type.
  - The attributes and methods of the supertype object class are then inherited by those object classes (subtype)
  - Sometimes abbreviated as gen/spec.

System Concepts for Object Modeling…

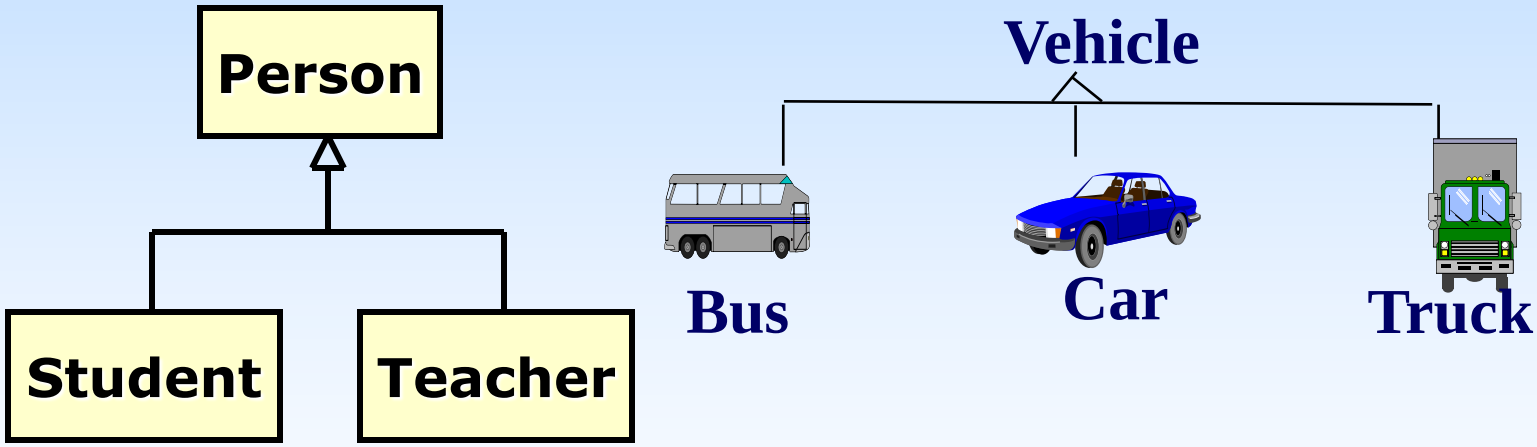
Specialization

Generalization



System Concepts for Object Modeling…

- Generalization / Specialization



*** Specialized classes inherits from the parent class**

System Concepts for Object Modeling…

- Object Class Relationships
  - A natural business association that exists between one or more objects and classes

e.g. You interact with a text book by reading it,

with a telephone by using it,

People interact with each other by communicating with them.

System Concepts for Object Modeling…

- Object / Class Association
  - When you turn on your TV, in object oriented terms, you are in an *association* with your TV.
  - An association is unidirectional (one way) or bi-directional (two way).

    eg. *is married to*
  - Some times an object might be associated with another in more than one way.

    Gihan *is a co-worker of* Damith

    Gihan *is a friend of* Damith

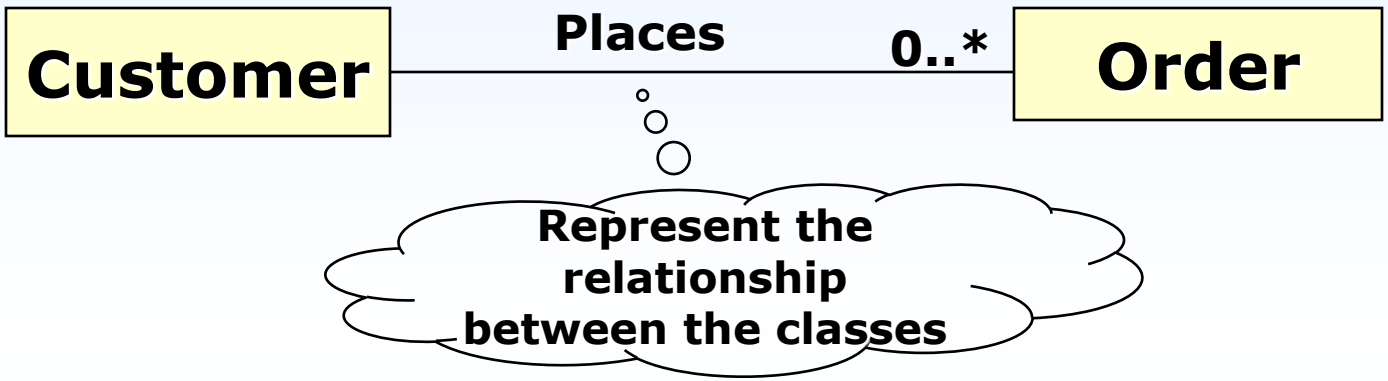
System Concepts for Object Modeling…

- Object / Class Association

  e.g.

  A CUSTOMER PLACES zero or more ORDERS

  An ORDER IS PLACED BY one and only one CUSTOMER



System Concepts for Object Modeling…

- Multiplicity
  - The minimum and maximum number of occurrences of one object class for a single occurrence of the related object class.

    e.g. Exactly one -> **1** or *leave blank*

    Zero or 1 -> **0..1**

    Zero or more -> **0..*** or ***

    1 or more -> **1..***

    Specific range -> **7..9**

Refer Figure 10-5 pg 377 Ref1 for more details

UCSC

System Concepts for Object Modeling…

- ***Aggregation***
  - A relationship in which one larger “whole” class contains one or more smaller “parts” classes. Conversely, a smaller “part” class is part of a “whole” larger class.

    e.g.

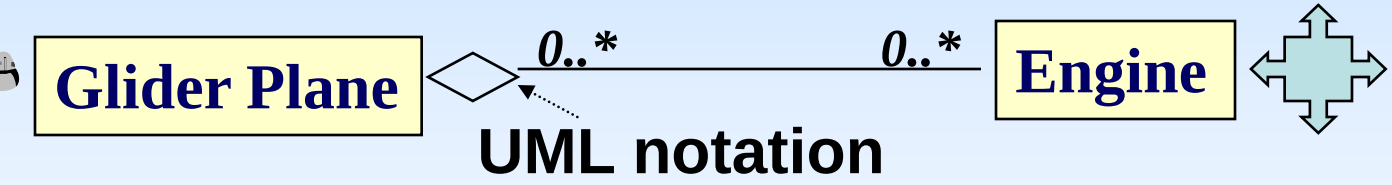
    A club – a club is made up of several club members

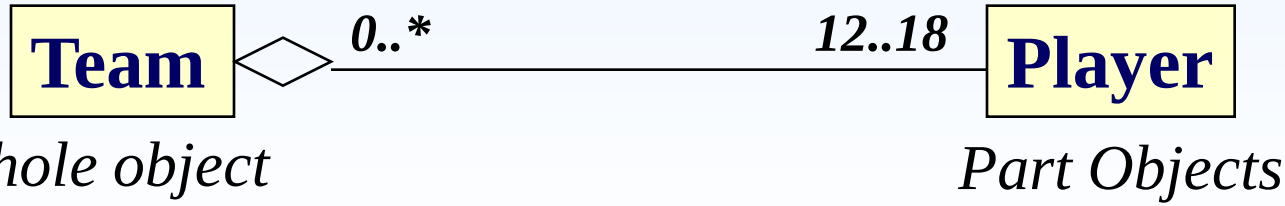
    A computer – a computer contains a case, CPU, motherboard, power supply …etc.

System Concepts for Object Modeling…

- ***Aggregation (Removed in UML 2.*)***

  some more examples…





System Concepts for Object Modeling…

- ***Composition***
  - An aggregation relationship in which the "whole" is responsible for the creation and destruction of its "parts".
  - If the "whole" were to die, the "part" would die with it.
  - A stronger form of aggregation.
    - The relationship between club and club member would not be composition, because members have a life out-side the club and can, belong to multiple clubs.

System Concepts for Object Modeling…

- ***Composition***
  - Drawn with a filled diamond.

Each "part" can belong to only one "whole", therefore, multiplicity needs to be specified only one for the "part"

Components will live and die with the whole object

System Concepts for Object Modeling…

- Polymorphism
  - Literally meaning “many forms”, the concept that different objects can respond to the same message in different ways.

e.g. Consider the WINDOW and DOOR objects