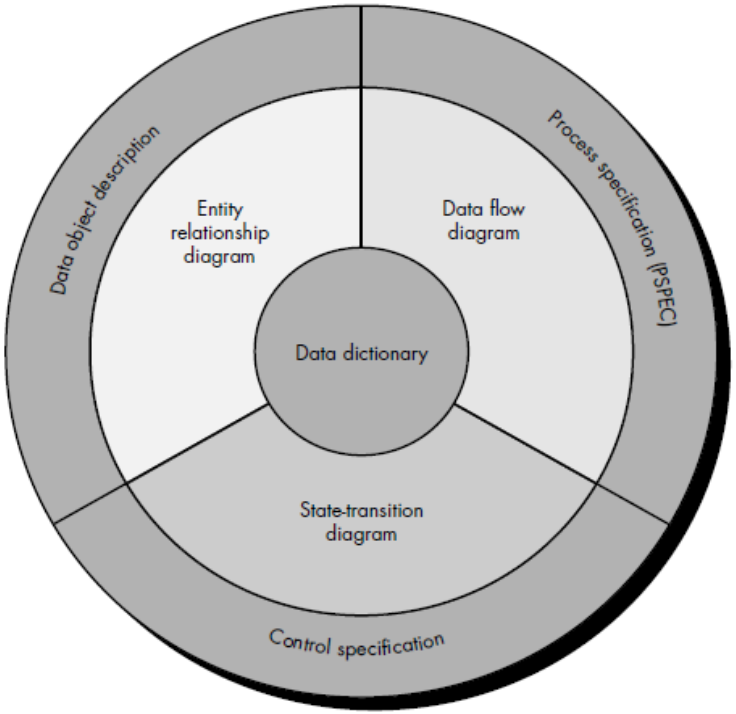
Structured Analysis Approach

Structured/Procedural Analysis

- Structured analysis describes a system as a hierarchy of functions
- The approach takes into account data and activities aspects
- Data aspect is modelled with Entity Relationship (ER) model
- Activities aspect is modelled with Data Flow Diagram (DFD)

Elements of Analysis model-Structured Analysis



Data dictionary is a repository that contains description of all data objects consumed or produced by the software

Data object description is textual description of entities and attributes in the ER

Process specification description is textual description of each function in DFD

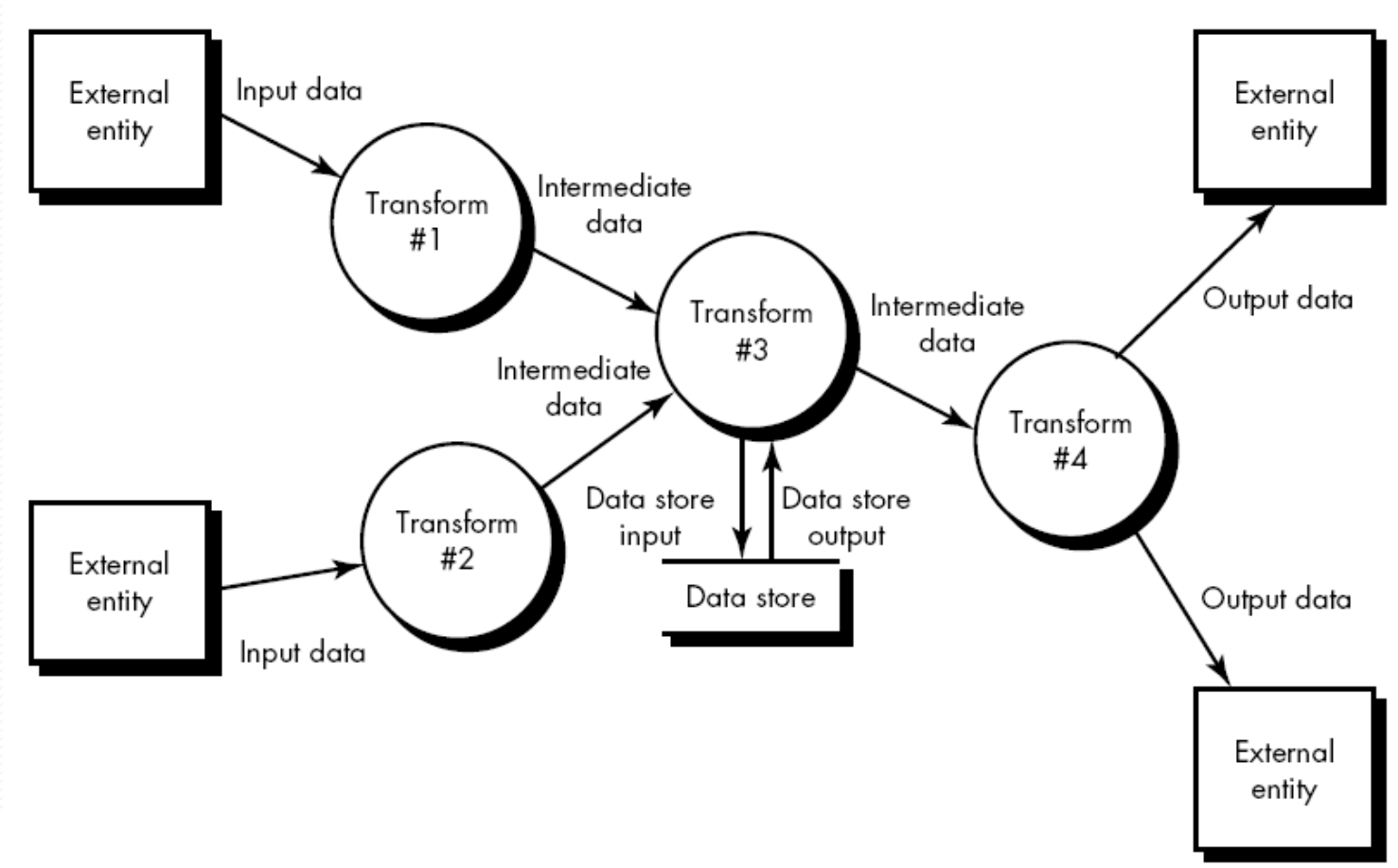
Control specification is textual description of Data conditions within a function

We just talk about ER model and Data Flow Diagram !

Data Flow Diagram

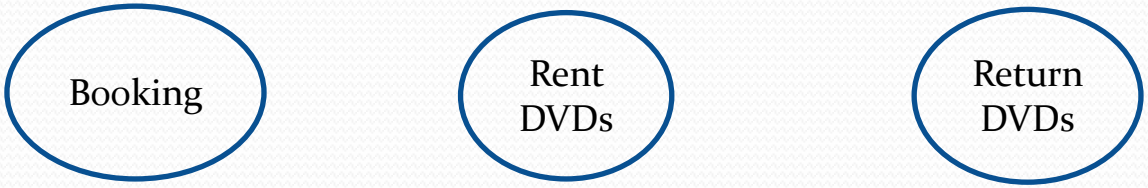
- A *data flow diagram (DFD)* is a graphical representation that depicts information flow and the transformation that are applied as data move from input to output
- DFD's purposes:
  - to provide an indication of how data are transformed as they move through the system, and
  - to depict the functions (and subfunctions) that transform the data flow
- Constructs:
  - Information flow
  - Process (bubble)
  - Data store
  - Source/Sink (external entity – human, hardware, other system)

DFD format


External entity
Input data
Transform #1
Intermediate data
Transform #3
Intermediate data
Intermediate data
Transform #4
External entity
Output data
External entity
Transform #2
Data store input
Data store output
Data store
Input data
Output data
External entity

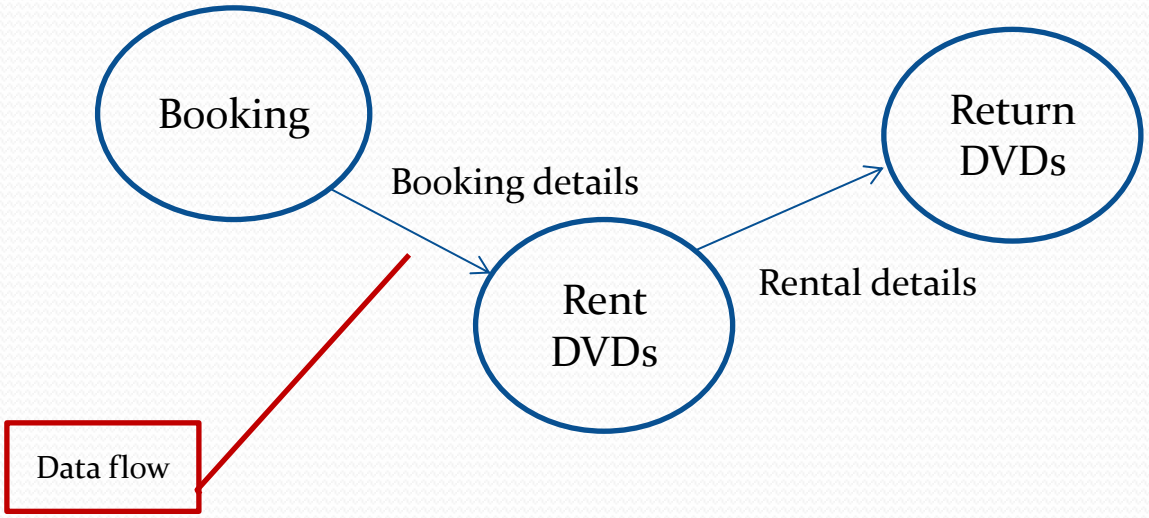
Process

- Work or action performed on data (inside the system)
- Receive input data and produce output data
- A process can receive one or several input data and produce one or several output data
- A process can connect to any other constructs
- Examples of process – Video store:



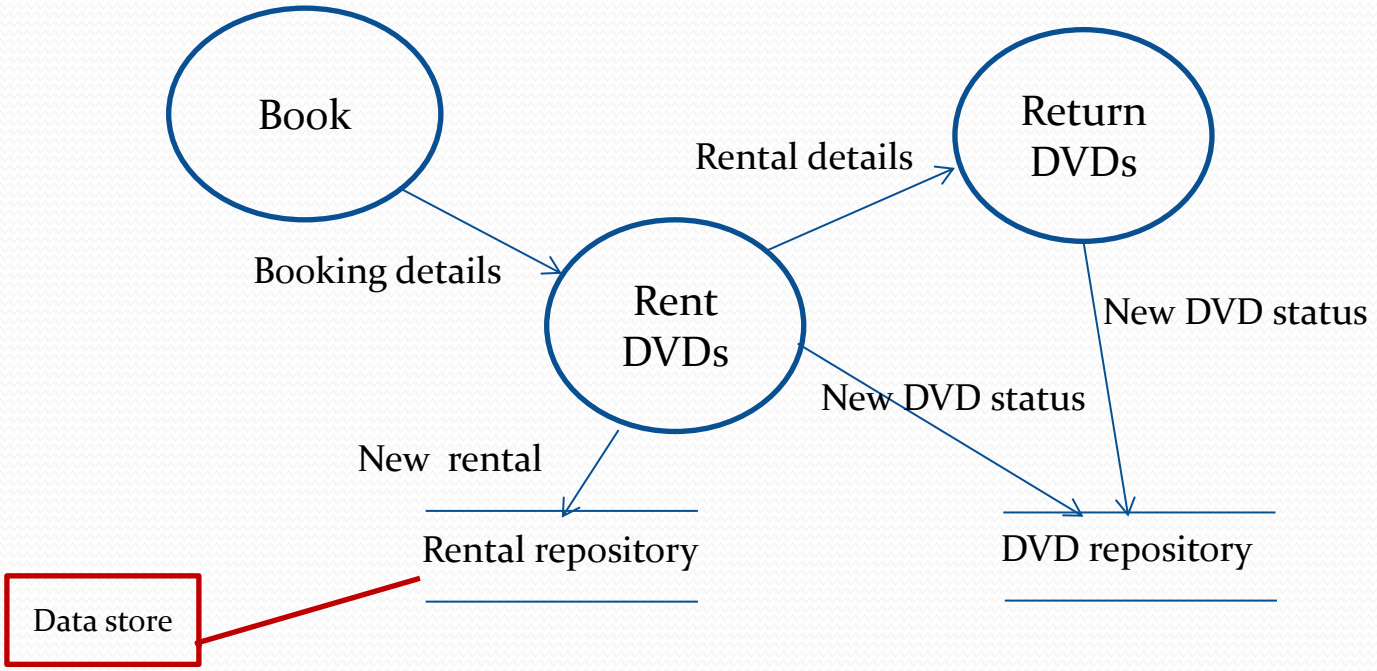
Data Flow

- A path for data to move between parts in/outside the system
- Arrows depict the direction of data movement
- Labels should be a noun (represent data)



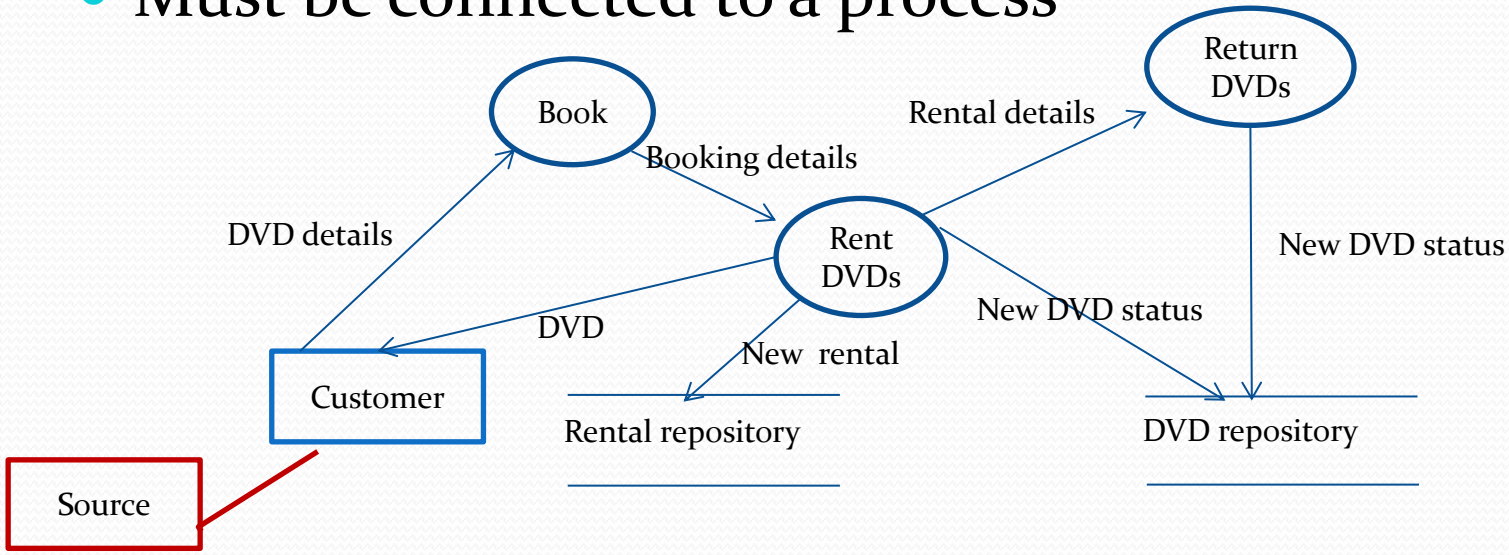
Data store

- Represent data that the system stores
- Have at least one incoming/outgoing data flow for each data store



Source/sink (external entity)

- External entity that is origin or destination of data (outside the system)
- Source: entity supplies data to the system
- Sink: entity receive data from the system
- Must be connected to a process



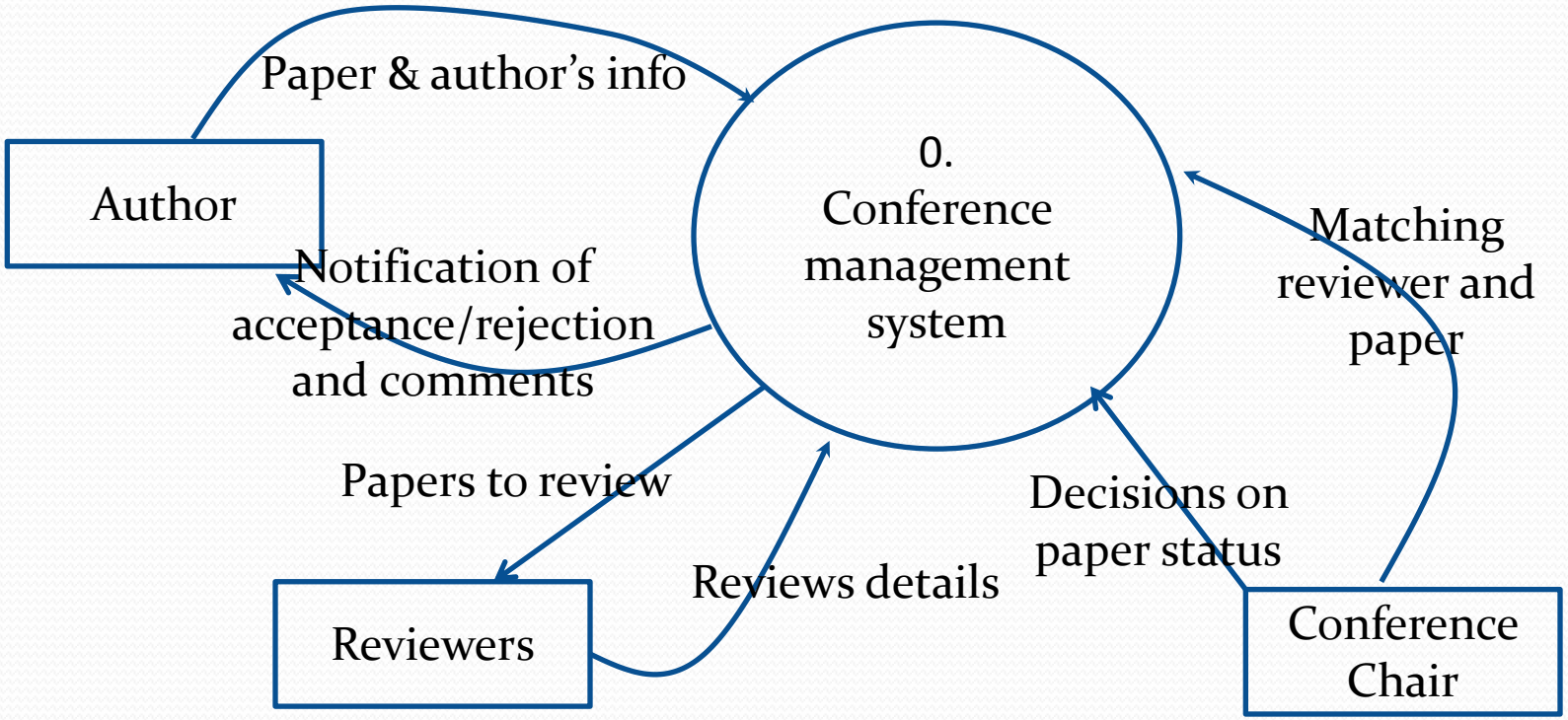
Context Diagram

- DFD may be used to represent a system or software at any level of abstraction. In fact, DFDs may be partitioned into levels that represent increasing information flow and functional detail.
- Top level view of the system – level 0 – called Context diagram
- Show the system boundaries, external entities that interact with the system, and information flow between external entities and the system

Example- Conference management system

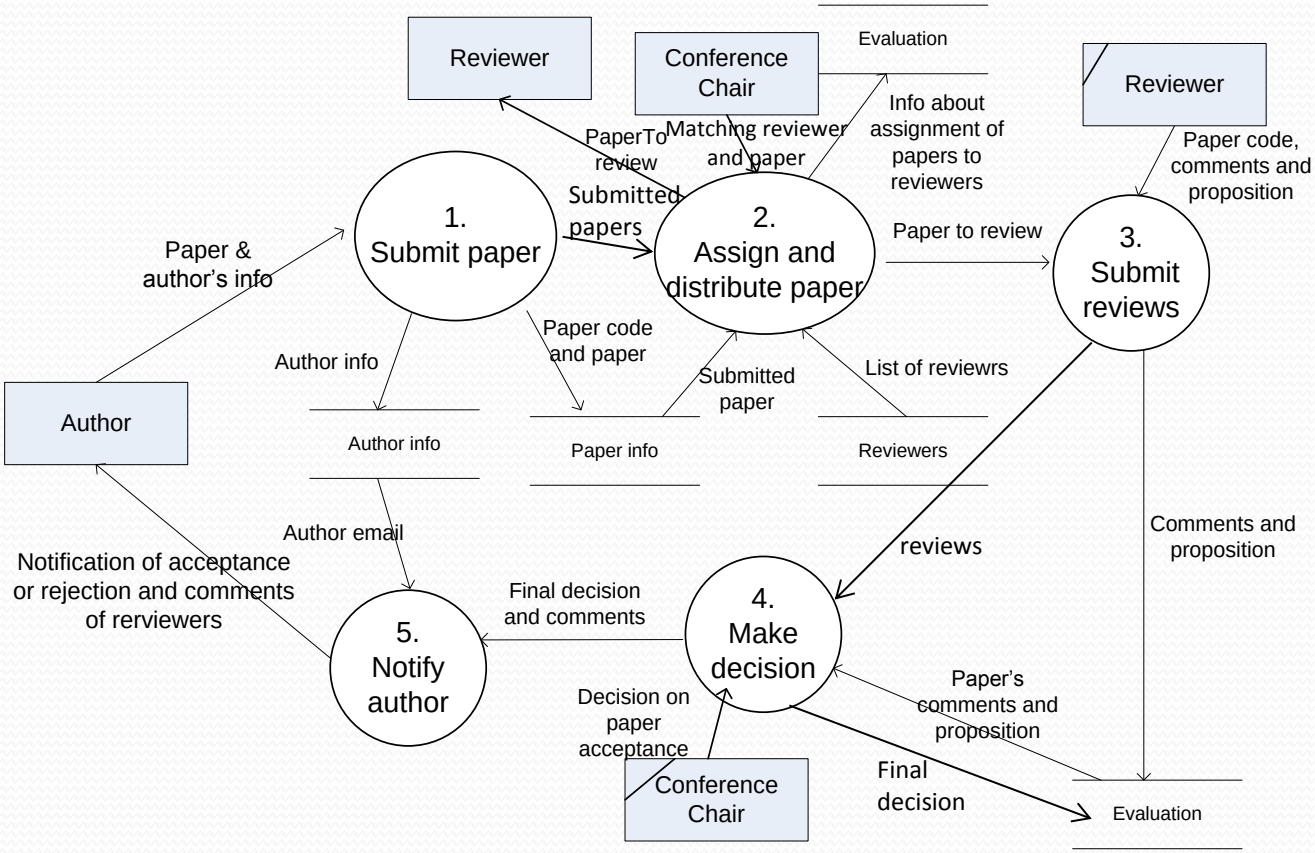
- A scientific conference needs a system to manage paper submissions and the reviewing process. Researchers and practitioners can submit their scientific papers. There is a list of reviewers(program committee) who can review and evaluate the submitted papers. The conference chairs will distribute the papers to relevant reviewers. The reviewers then submit their reviews/comments and score to the system. The conference chair will make the decision about accepting the papers to be presented and published or rejecting the papers. The conference chair then notify the authors about the status of their submission.

Level 0 – DFD - Example

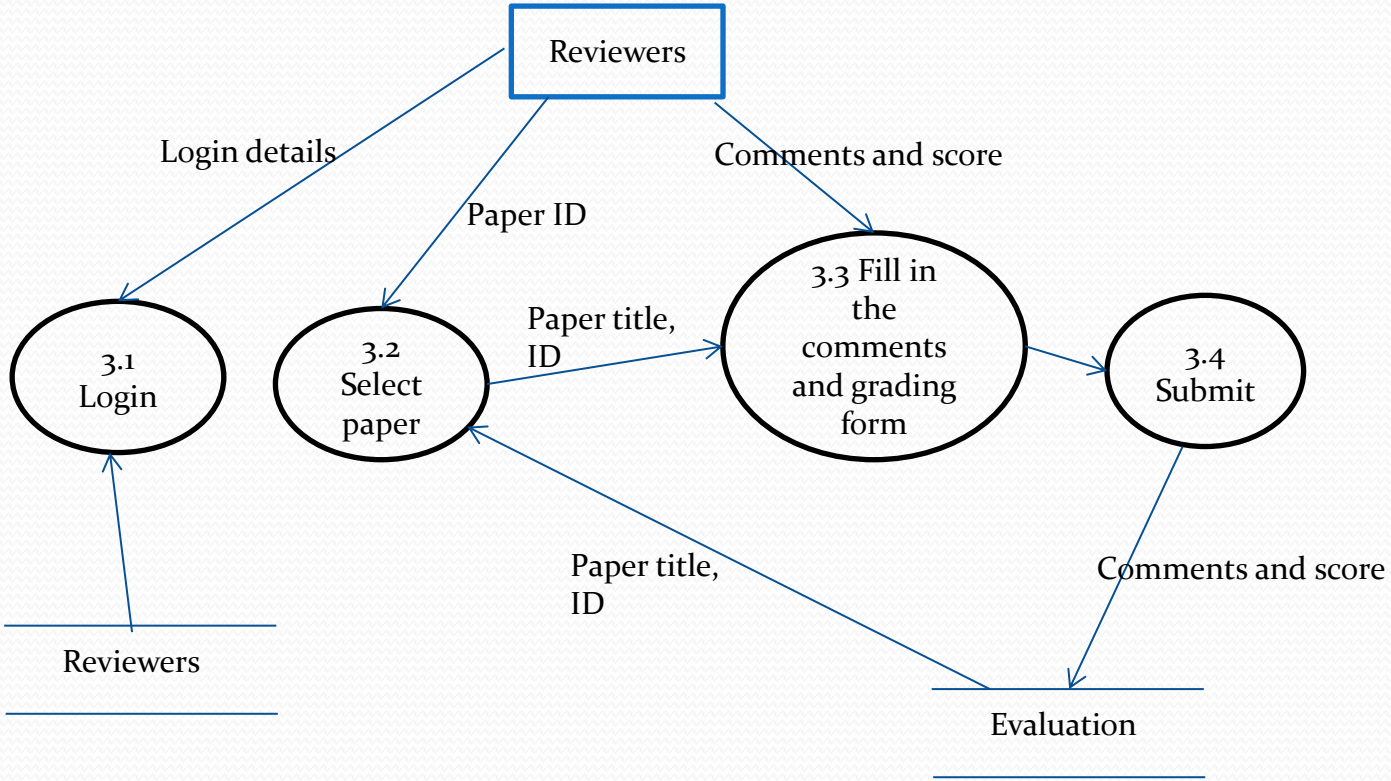


Lower level – Level 1

- Functional decomposition (top-down)



Level 2 – Submit review



DFD guidelines

- The level zero of DFD should depict the software/system as a single bubble;
- Primary input and output should be carefully noted;
- Refinement should begin by isolating candidate processes, data objects, and stores to be represented at the next level;
- All arrows and bubbles should be labelled with meaningful names;
- Information flow continuity must be maintained from level to level, and
- One bubble at a time should be refined

Data Modeling

- Examines data objects independently of processing
- Focuses attention on the data domain
- Creates a model at the customer's level of abstraction
- Indicates how data objects relate to one another

Entity Relationship Diagram

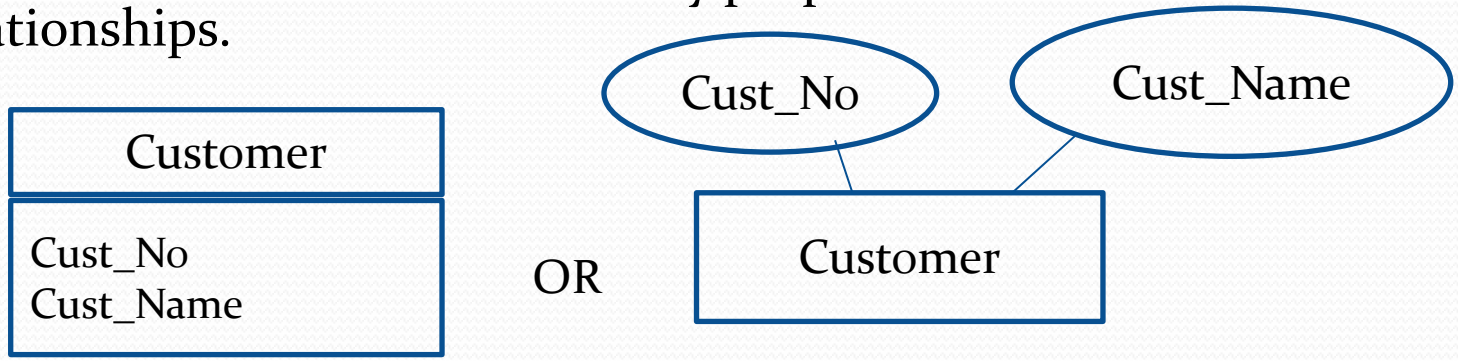
- Proposed by Peter Chen in 1977 which allows to describe objects/entities and their relationship within the system
- Composed of:
  - Data objects/Entity
  - Attributes
  - Relationships
  - Cardinality

Construct description

- *Entities* represent classes of objects (facts, things, people,…) of interest that have properties in common and an autonomous existence.

- *Relationships* represent logical links (connectedness) between two or more entities. Between two data objects/entities there may be several relationships.

- *Attributes* describe the elementary properties of entities or relationships.



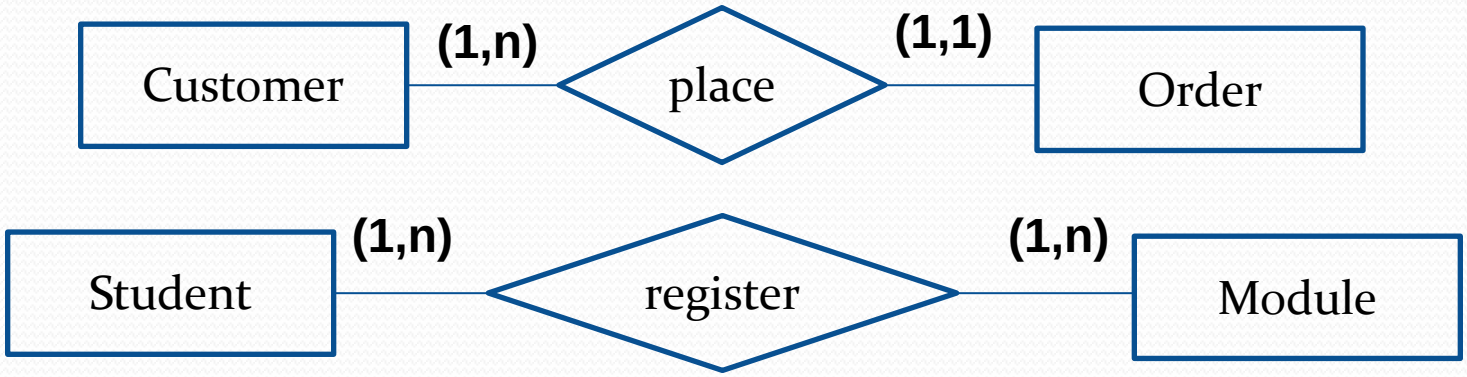
Construct description

- *Cardinality* is the specification of the number of occurrences of one data object that can be related to the number of occurrences of another data object
  - 1:1 One library is managed by one manager
  - 1:N One book can have many copies
  - N:M Many staff may work in many offices: 1 member of staff may work in M offices and 1 office may have N members of staff working in it.

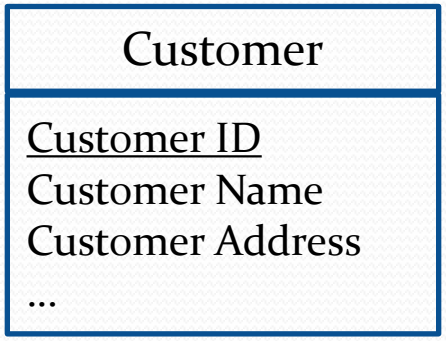
Note: The cardinality in ER is located at the opposite side of the relationship compared to the multiplicity in Classes diagram

Construct description

- Example of cardinality



- *Identifiers* (or keys) consist of one or more attributes which identify uniquely instances of an entity.



Classes diagram

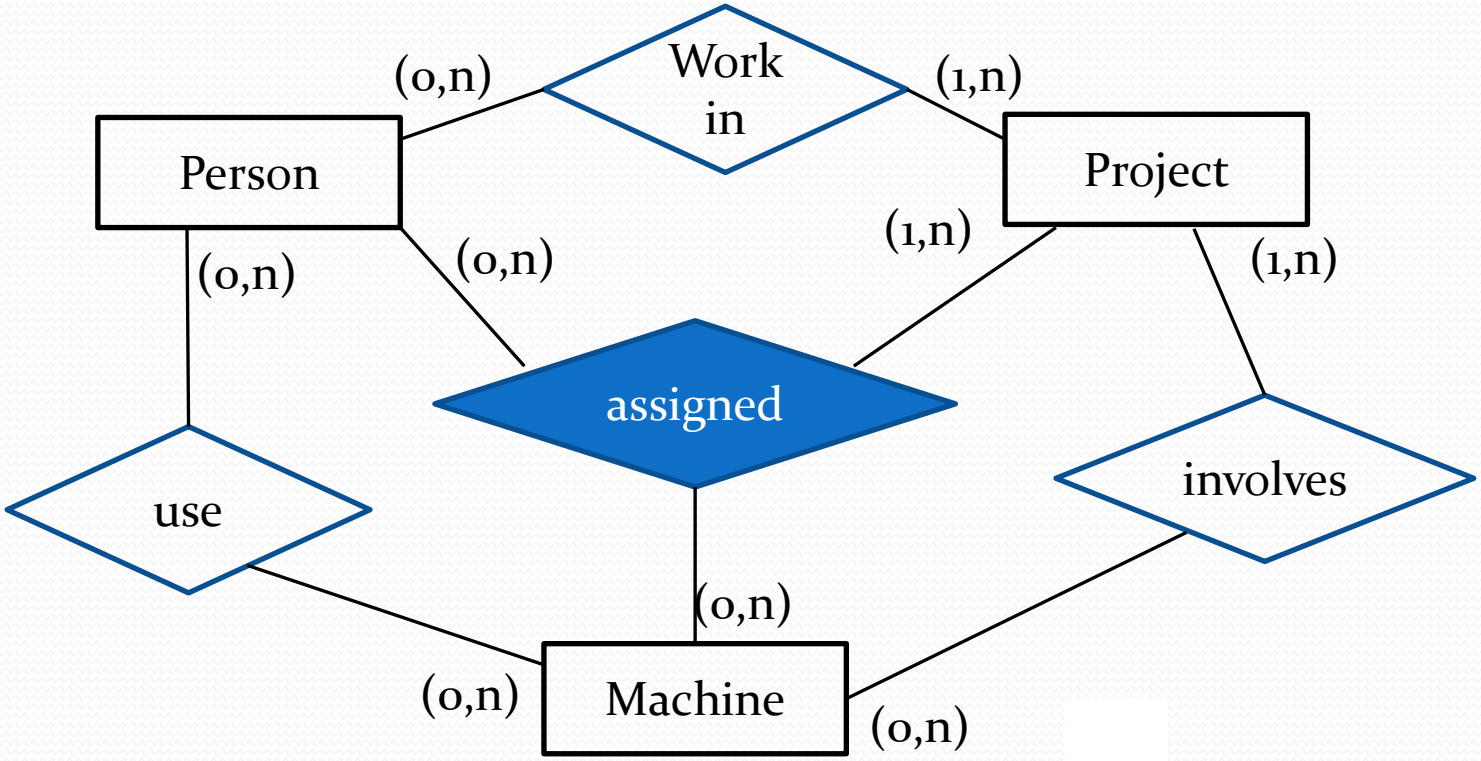
Modality

- The modality of a relationship is 0 if there is no explicit need for the relationship to occur or if the relationship is optional

- The modality of a relationship is 1 if the relationship is mandatory

Ternary relationship

Can we replace a ternary relationship with 3 binary relationships ?

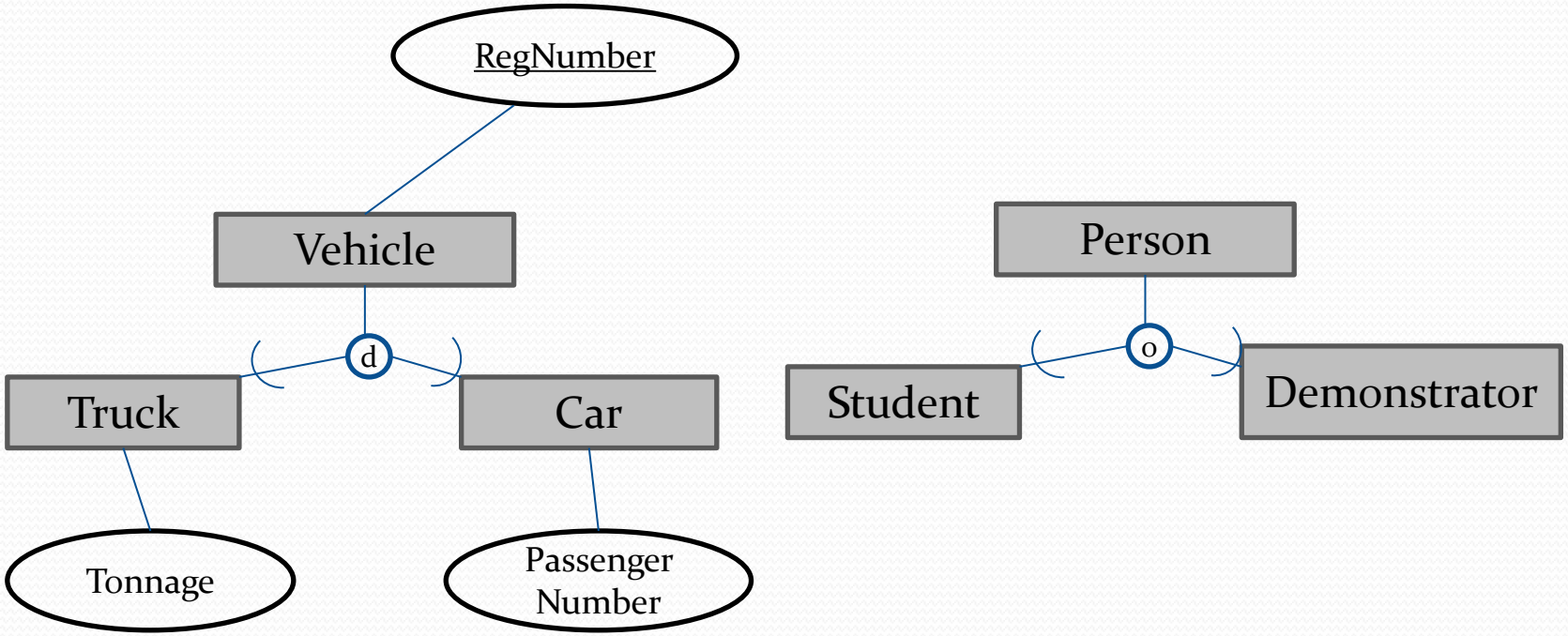


Three binary relationships don't have the same semantic as a ternary relationship.
Replacing it causes losing information

Specialization (Extended ERD)

- *Specializations* represent logical links between an entity E, known as parent entity, and one or more entities E1,…,En called child entities, of which E is more general, in the sense that they are a particular case.
  - A generalization is **total** if every instance of the parent entity is also an instance of one of its children, otherwise it is **partial**.
  - A generalization is **exclusive** (**disjoint**) if every instance of the parent entity is at most an instance of one of the children, otherwise it is **overlapping**.

Examples



d: disjoint
o: overlapping

Choices…

- Should a concept be modeled as an entity or an attribute?
- Should a concept be modeled as an entity or a relationship?
- Identifying relationships: Binary or ternary? Specialization?
- Constraints on the ER Model:
  - A lot of data semantics can (and should) be captured.
  - But some constraints cannot be captured by ER diagrams.

Some rules of thumb

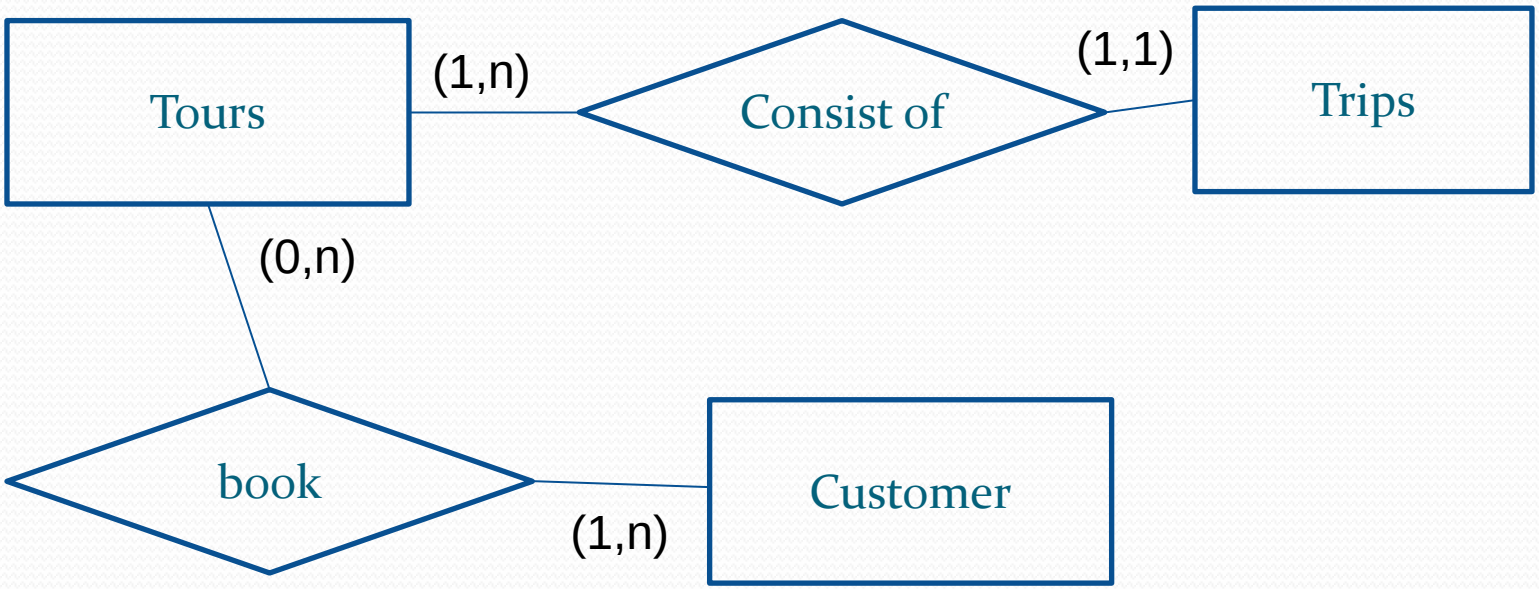
- If a concept has significant properties and/or describes classes of objects with an autonomous existence, it is appropriate to represent it as an entity.
- If a concept has a simple structure, and has no relevant properties associated with it, it is convenient to represent it with an attribute of another concept to which it refers.
- If a concept provides a logical link between two (or more) entities, it is convenient to represent it with a relationship.
- If one or more concepts are particular cases of another concept, it is convenient to represent them in terms of a generalization relationship.

(J. Mylopoulos, 2004)

Example 3

- A tourist company organises tours packages to Africa. A tour has a departure date and a return date, destination country, a price, and includes several trips to visit different places/cities. Each trip has a departure location, departure date and time, and a destination location/city. Customer can book a tour or several tours.

ERD



Adding attributes. Country, cities and places are attributes or entities ?