Structured Analysis – or (mostly) Domain Modeling

Seng 321

What is the purpose of Analysis?

- Book: “through study of a problem domain, the achievement of understanding of, and the documentation of the characteristics of that domain and the problems (requiring *software* solutions) that exist within the domain.”
  - Structure of problem domain
  - Problem domain data
  - Innate properties and behavior of problem domain/subdomains
  - Significant events and phenomena in problem domain
  - Requirements: effects the system should produce within the problem domain

Traditional ‘Structured’ Modeling

To complement the **Textual Requirements** it also includes:

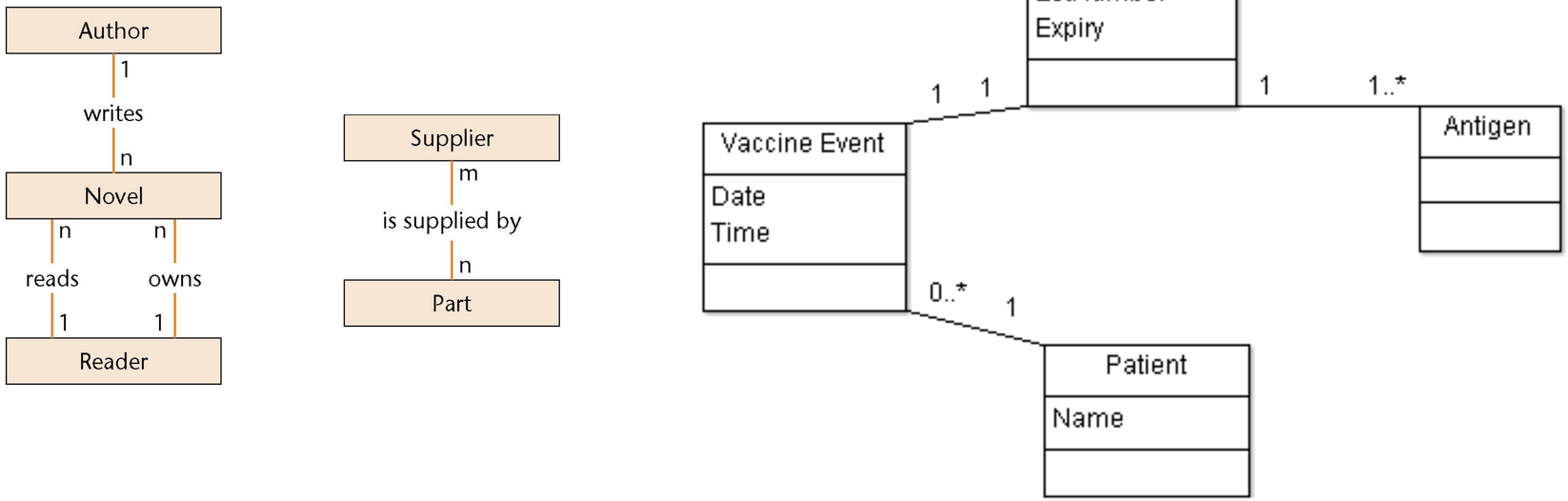
Data modeling artifacts**:**

- Entity Relationship Diagrams (ERDs)
- Data dictionary (DD)

Process modeling artifacts**:**

- Data flow diagrams (DFDs)

Entity-relationship Diagrams



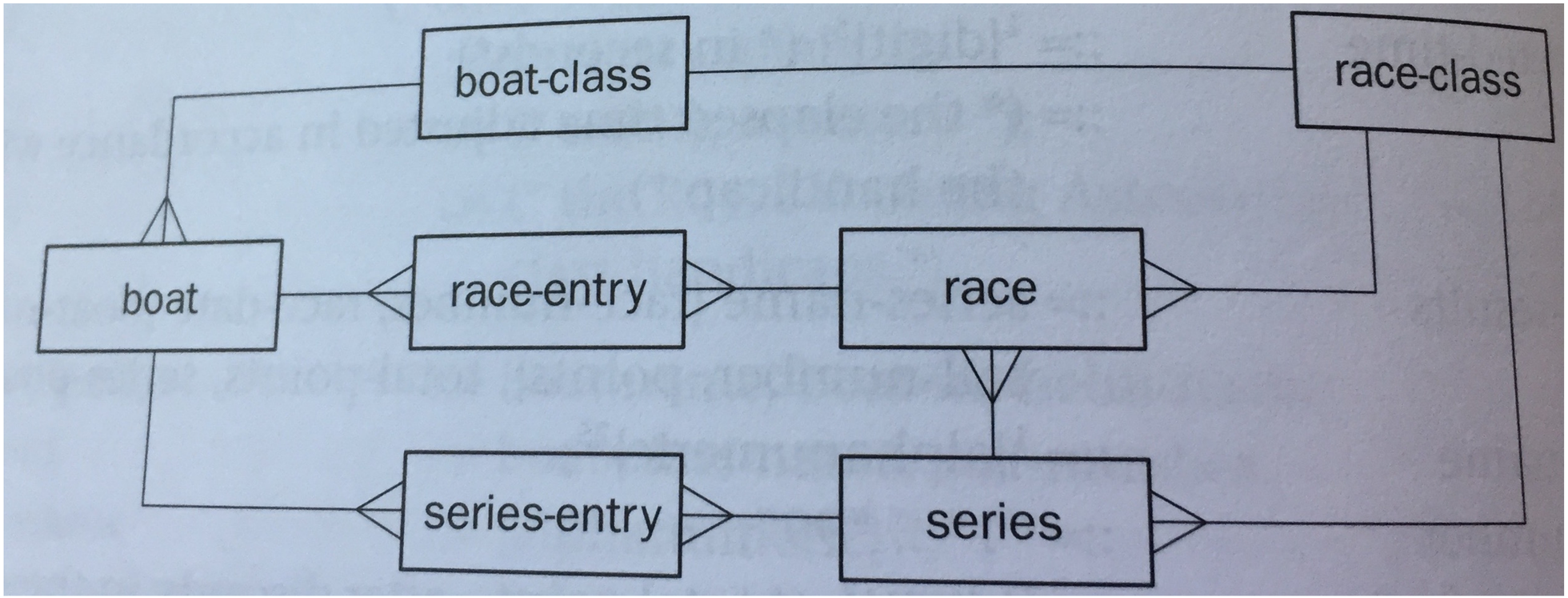
Vaccine: A specific instance of vaccine, containing 1 or more antigens and having an assigned lot number and expiry date.

Data Modeling

- Identify possible domain entities as **nouns** in requirements notes
- Narrow this list to just those mentioned in identified use cases
- Draw **data entities** with relationships —> **Data Entity Relationship** diagrams
- Refine **relationships** with cardinality (numbers of each class compared to their related classes)
- Describe each domain entity in the **Data Dictionary**

Entity-relationship Diagrams

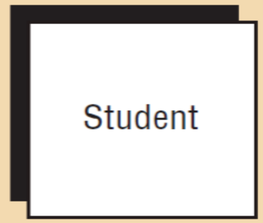
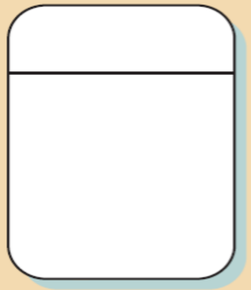
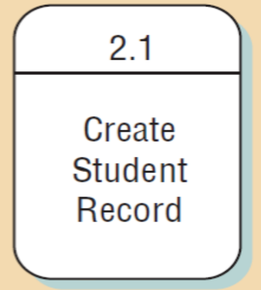
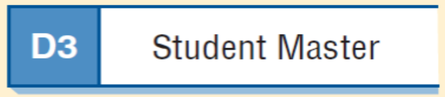
From the Boat race Case study in the textbook



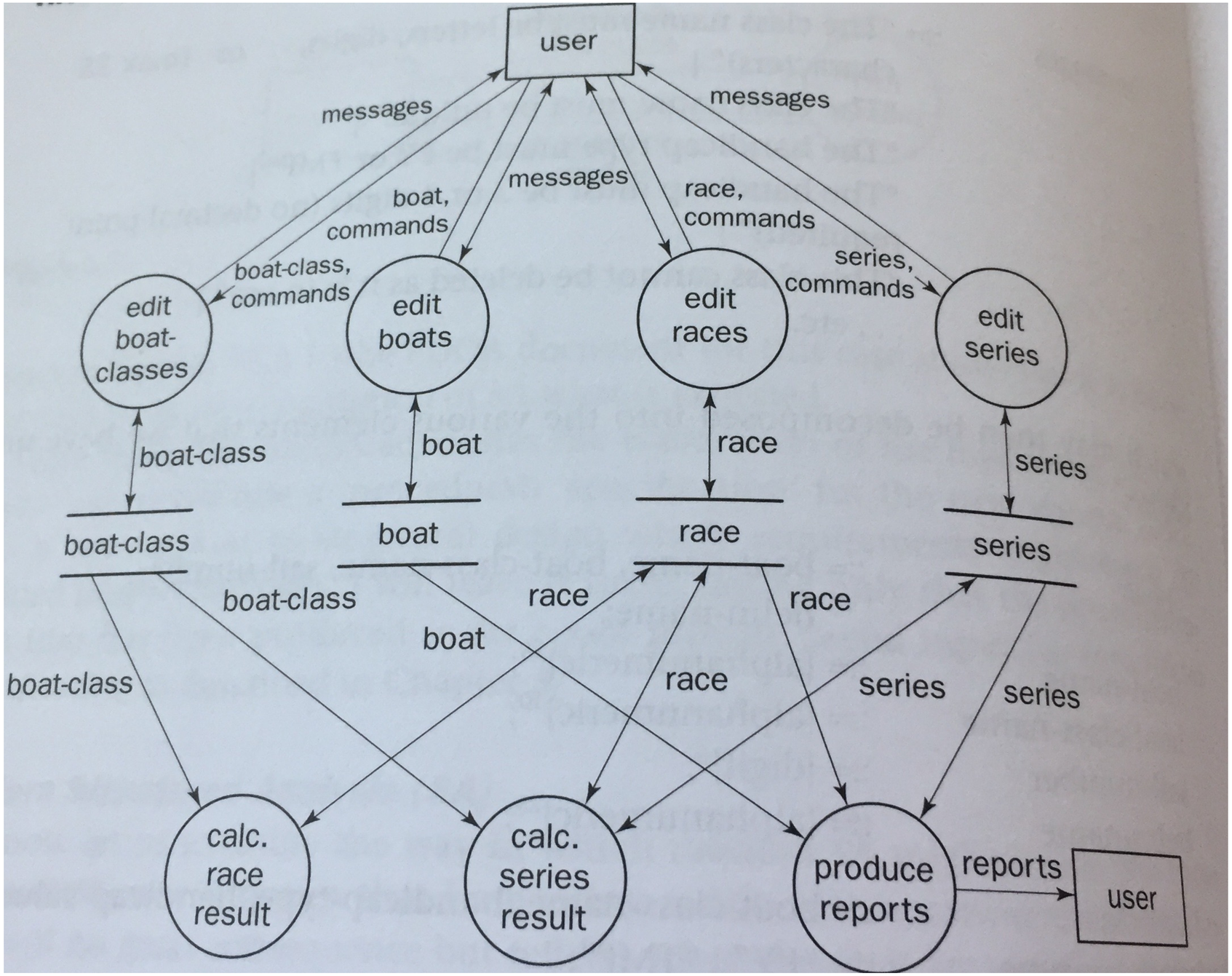
Process-oriented modeling (data flow modeling)

- Identify possible processes/functions as **verbs (actions on the data you defined)** in requirements notes
- Narrow this list to just those mentioned in identified use cases
- Draw **process bubbles** with relationships (data flow)—> **Data Flow Diagrams**
- Refine **diagrams at different levels of detail**

Data Flow Diagram (DFD) components

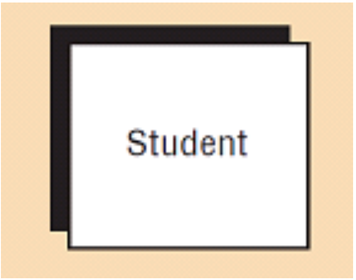
Symbol	Meaning	Example
	Entity	
	Data Flow	New Student Information 
	Process	
	Data Store	

Another example, from the textbook



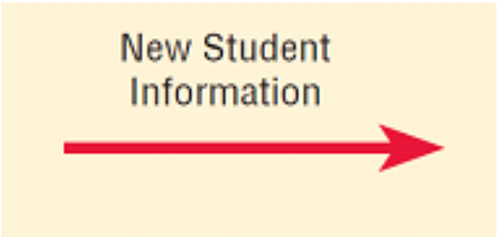
External Entities

- Similar to actors in Use Cases
- Represent another department, a business, a person, or a machine
- A source or destination of data, outside the boundaries of the system
- Should be named with a noun



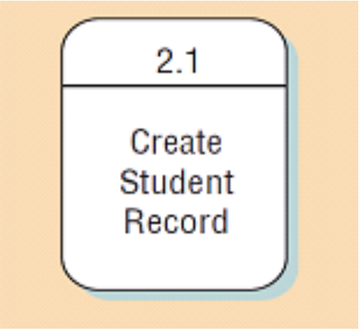
Data Flow

- Shows movement of information from one point to another
- Described with a noun
- Arrowhead indicates the flow direction
- Represents info about a person, place, or thing



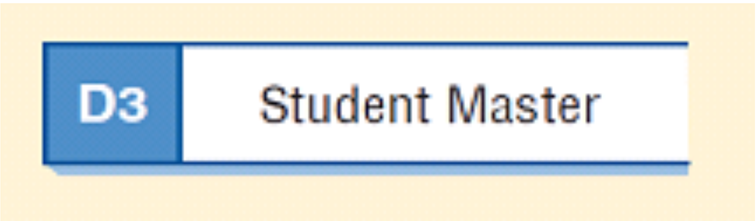
Process

- Similar to use cases or steps in a use case
- Each process transforms information
- Describes work performed in a system
- Naming conventions:
  - At context level – use system name
  - At subsystem level, name with subsystem name + the word “subsystem”
  - At more detailed levels use verb-adjective-noun for detailed processes

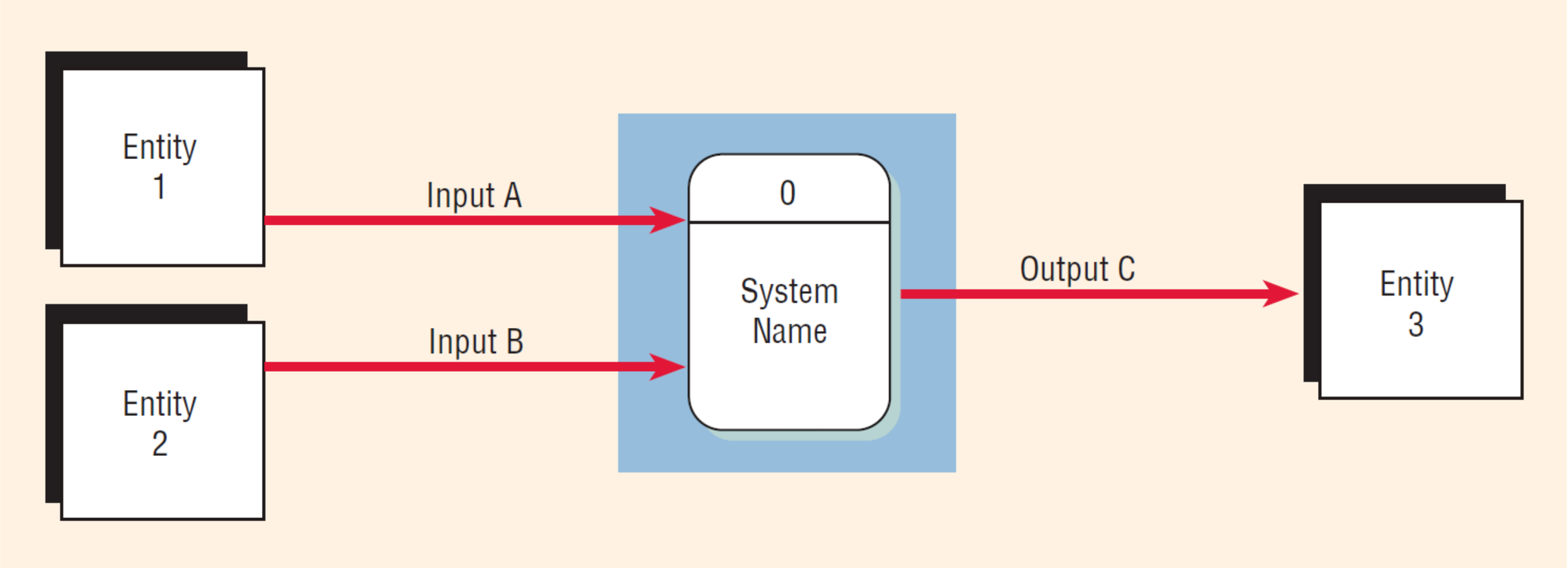


Data Store

- A place where information is stored
- Allows examination, addition, and retrieval information
- Named with a noun, describing the information
- Can be given a unique reference number, such as D1, D2, D3
- May represents a:
  - Filing cabinet
  - Database
  - Computerized file



Context diagram example



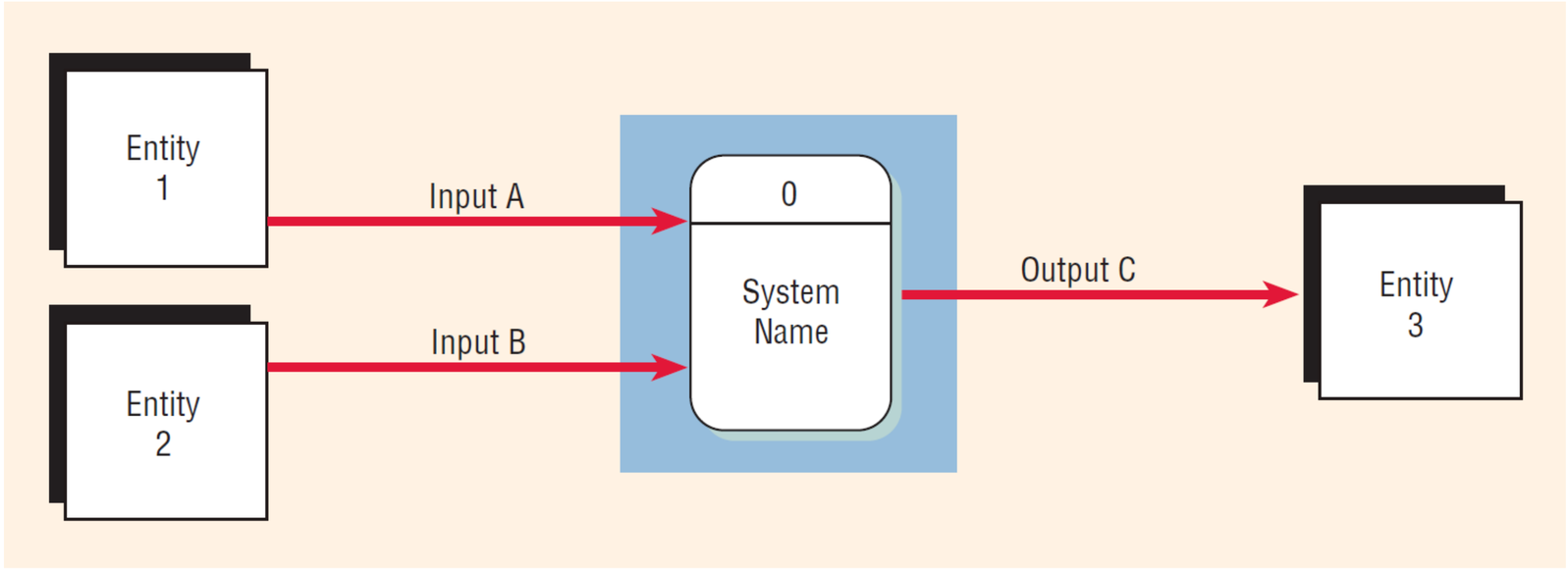
Steps in developing DFDs (basics)

1. Make list of business activities and things (potential entities)
2. Create context level diagram including external entities and summary data flows
3. Create a child diagram for each complex process on diagram 0. Include local data stores and detailed processes.
4. Check for errors and make sure names of processes and data stores are meaningful.

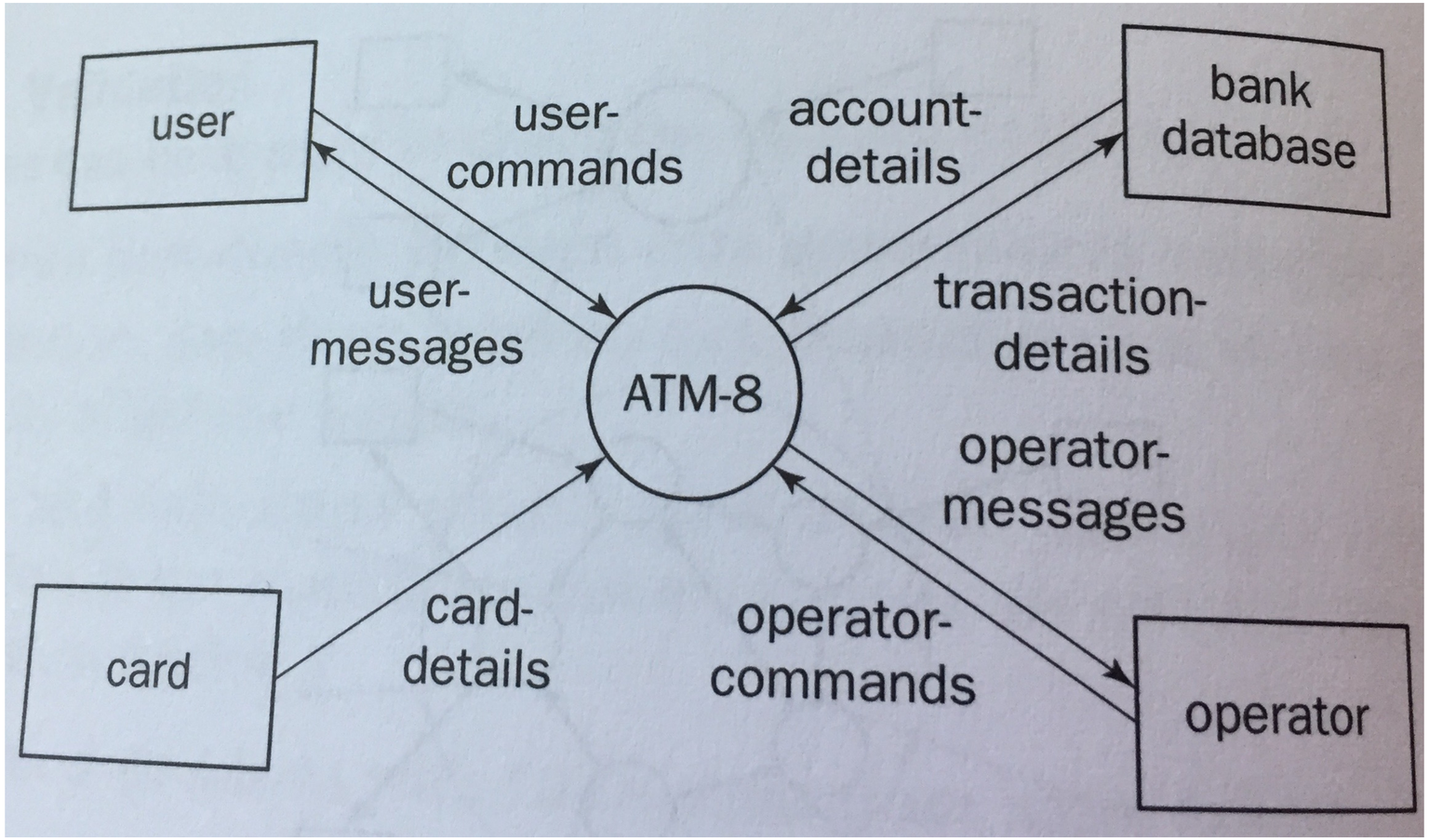
Creating the Context Diagram (DFD 0)

- The highest level in a data flow diagram
- Contains only one process, representing the entire system
- The process is given the number 0
- All external entities, as well as major data flows are shown

Context diagram (DFD 0) example



Another example of Context Diagram



Context diagram (DFD 0) example

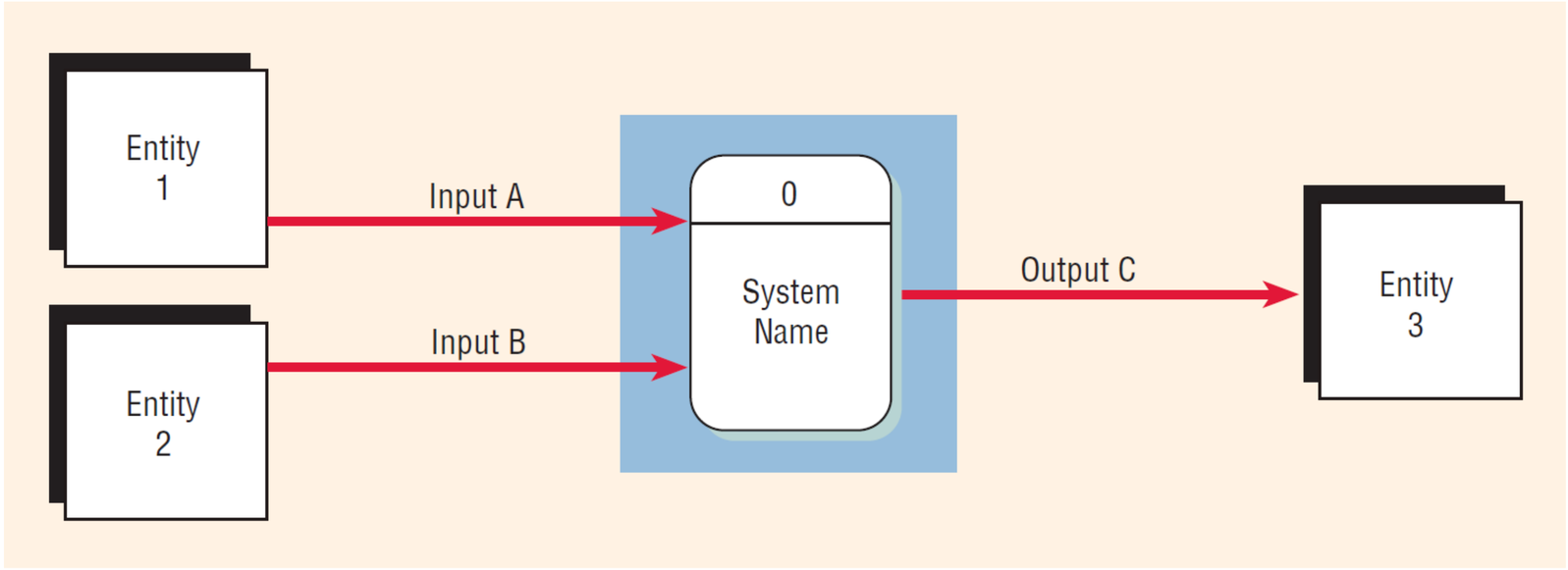
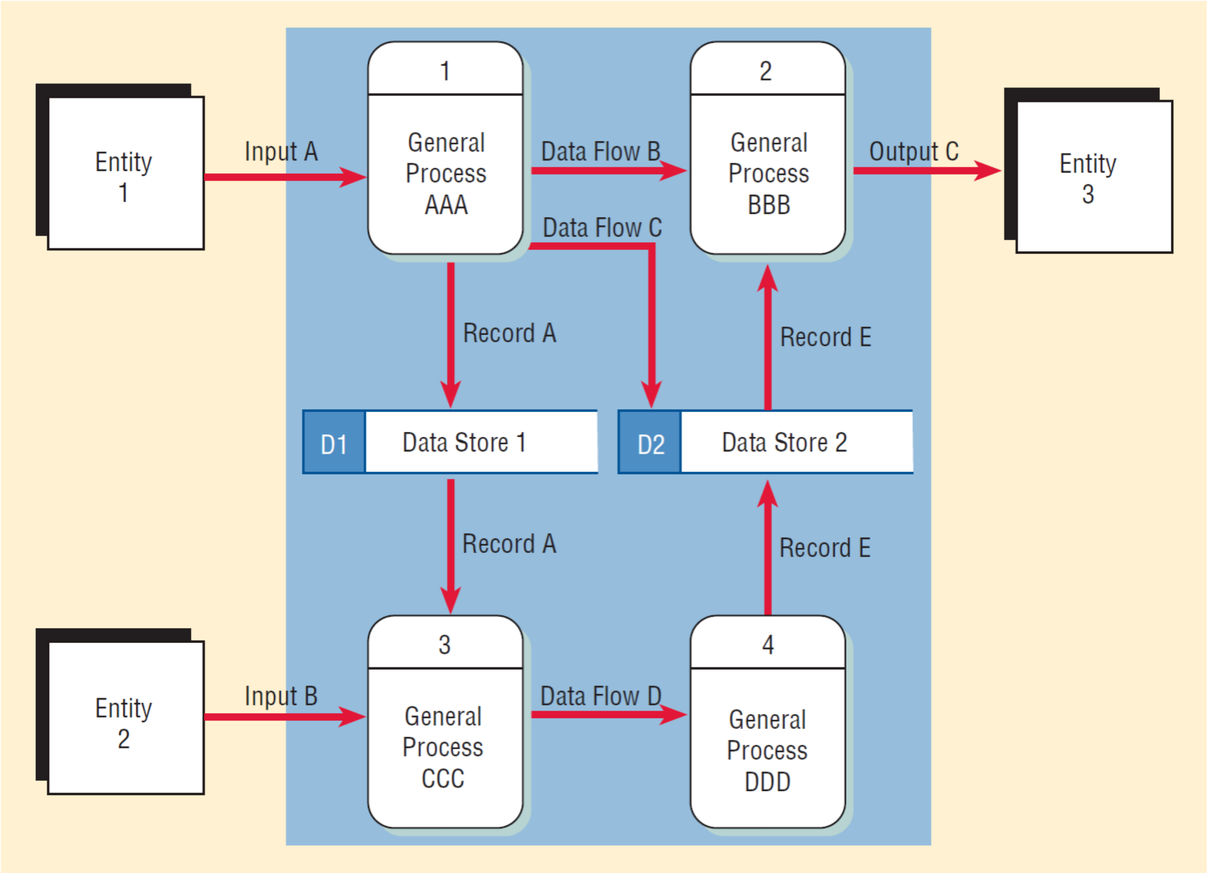


Diagram 1 example



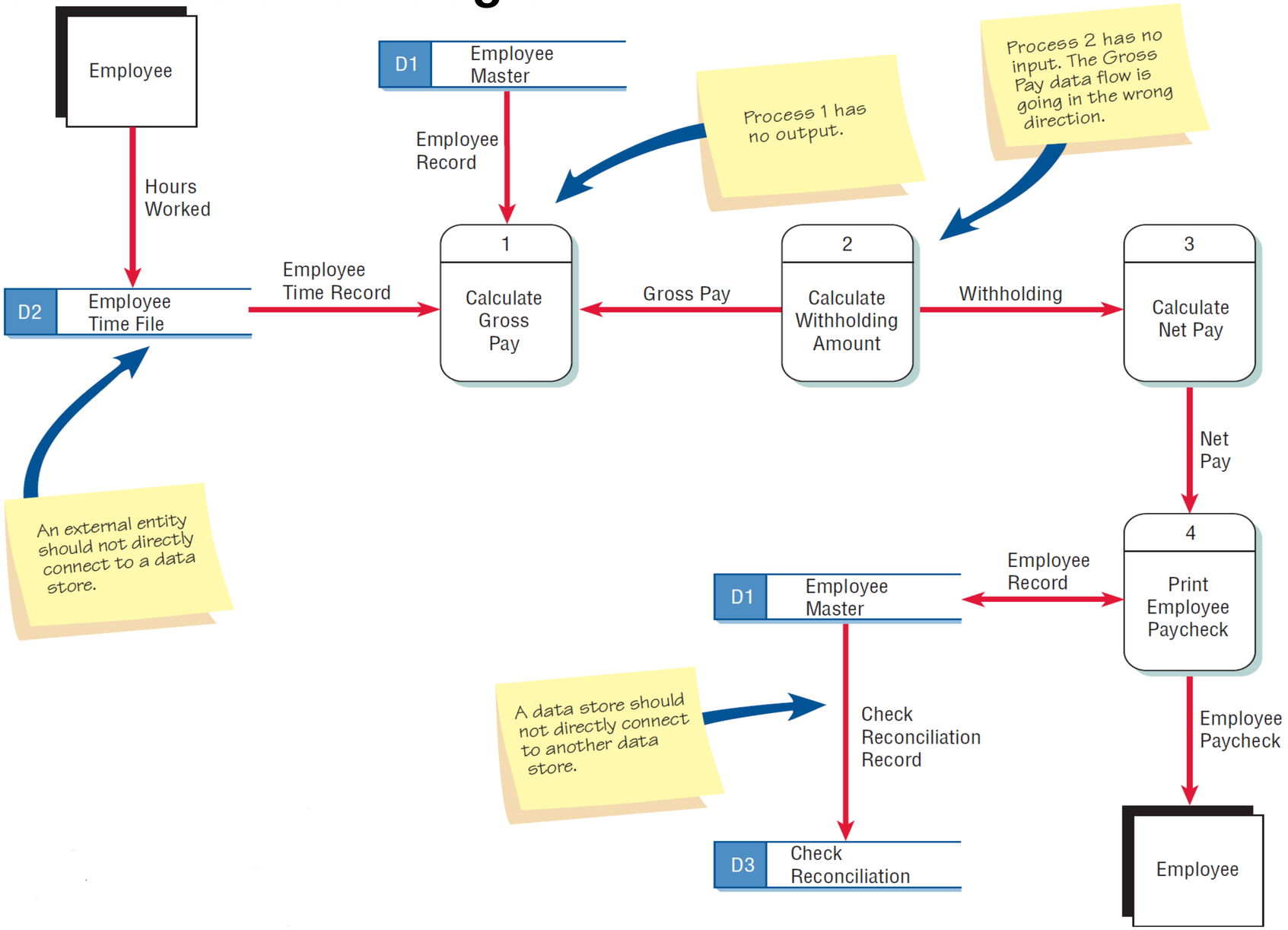
Data Flow Diagram Levels

- Data flow diagrams are built in layers
- The top level is the Context level
- Each process ‘blows up’ to a lower, more detailed level
- The lower level diagram number is the same as the parent process number
- Processes that do not create a child diagram are called primitive

Checking the Diagrams for Errors (Continued)

- Incorrectly labeling processes or data flow
- Including more than nine processes on a data flow diagram
- Must ensure diagram is consistent (balanced) with its parent diagram
  - Same external entities and input and output flows as parent

Typical errors in a data flow diagram



Object Oriented Modeling

- Could model processes with DFD
- But Data modelling involves objects
- Difference from ERDs is that there are actions placed on objects (detailed during Design phases, not requirements analysis)
- Here, just define the objects and their attributes

Object and their attributes, then relationships

object class	attributes
boat	boat-name, boat-class-name,
boat class	boat-class-name, handicap-ty
race-class	race-class-type, race-class-na
handicap-race-class	handicap-type, minimum-han
helm	helm-name
race	race-class-name, race-date,
race-entry	boat-class-name, sail-numbe race-time, result
race-officer	
series	series-name, race-class-nam of-races-to-count
series-entry	boat-class-name, sail-numbe