Somerville Chapter 8

Pressman Chapter 8

Objectives

- To explain why the context of a system should be modeled as part of the RE process
- To describe data modeling , functional modeling, behavioural modeling, and scenario-based modeling
- To introduce some of the notations used in the Unified Modeling Language (UML)

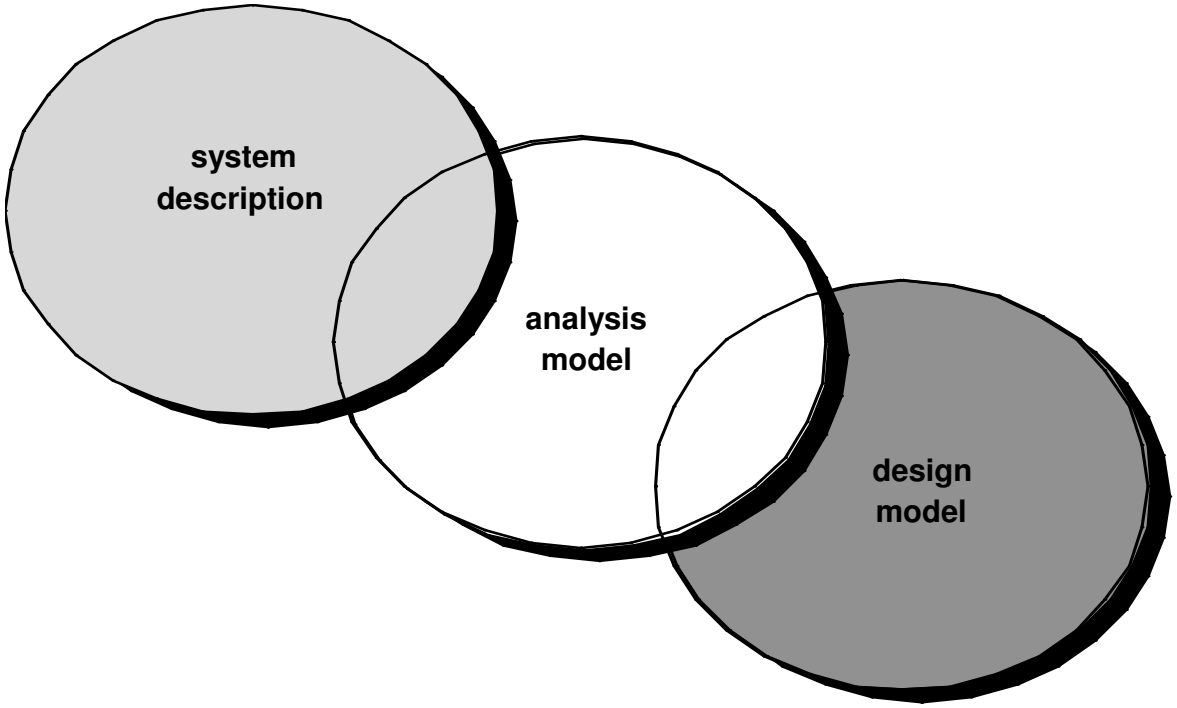
Topics covered

- Context models
- Data models
- Scenario-based models
- Flow-oriented models
- Behavioural models
- Class based models

Requirements Analysis

- Requirements analysis
  - specifies software's operational characteristics
  - indicates software's interface with other system elements
  - establishes constraints that software must meet
- Requirements analysis allows the software engineer (called an *analyst* or *modeler* in this role) to:
  - elaborate on basic requirements established during earlier requirement engineering tasks
  - build models that depict user scenarios, functional activities, problem classes and their relationships, system and class behavior, and the flow of data as it is transformed.

Analysis Phase: What is it?



Three objectives:

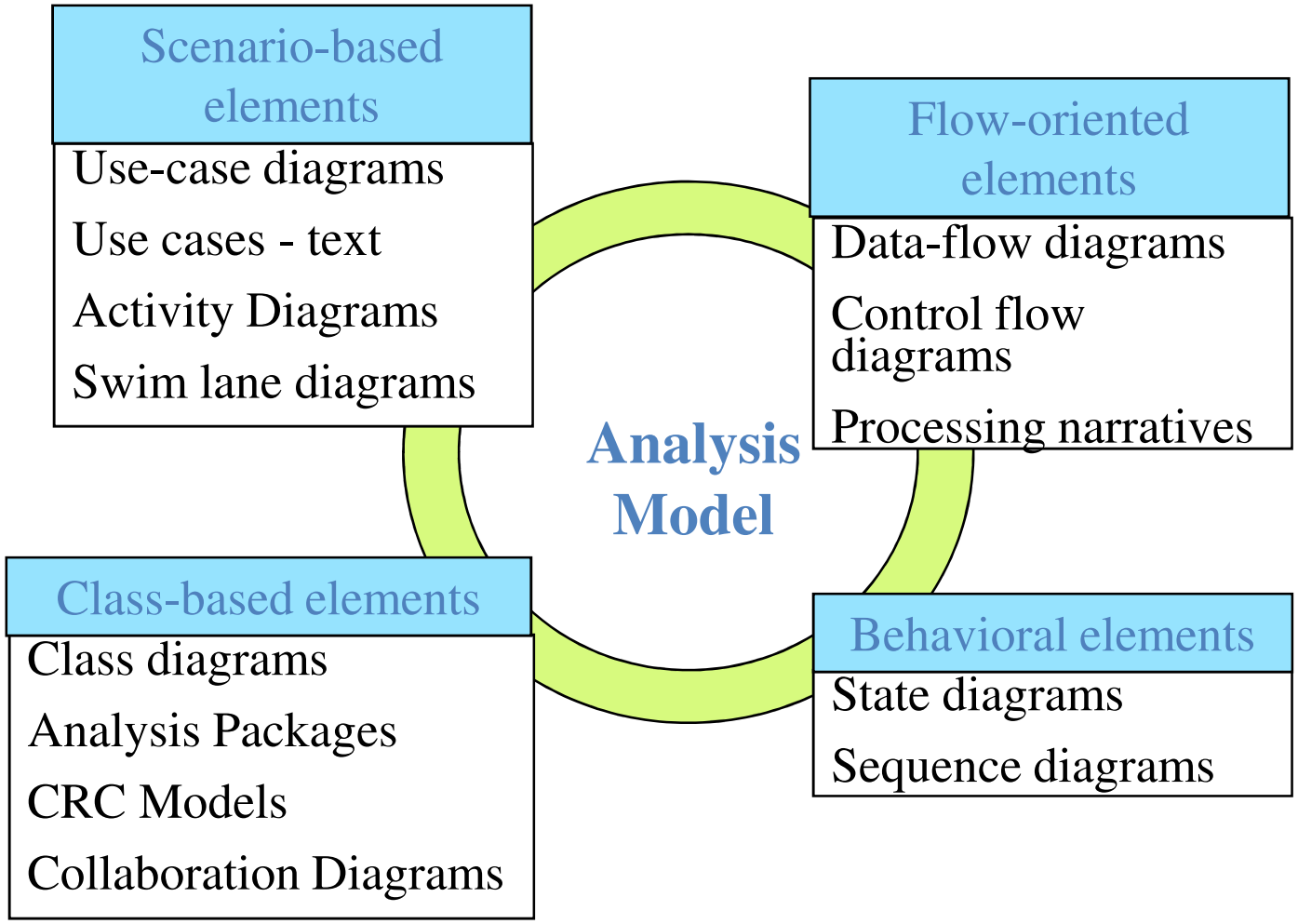
- To describe what the customer requires
- To establish a basis for the creation of a software design
- To define a set of requirements that can be validated once the software is built

Analysis Modeling Approaches

- **Structured analysis:**
  - The **data**: The model defines data objects, their *attributes* and *relationships*.
  - The **processes** that *transform* the data: The model shows how they transform the data objects as they *flow* through the system.
- Object-oriented analysis:
  - Focus: Classes and their inter-relationships
  - UML is predominantly object-oriented

But don't be too dogmatic!

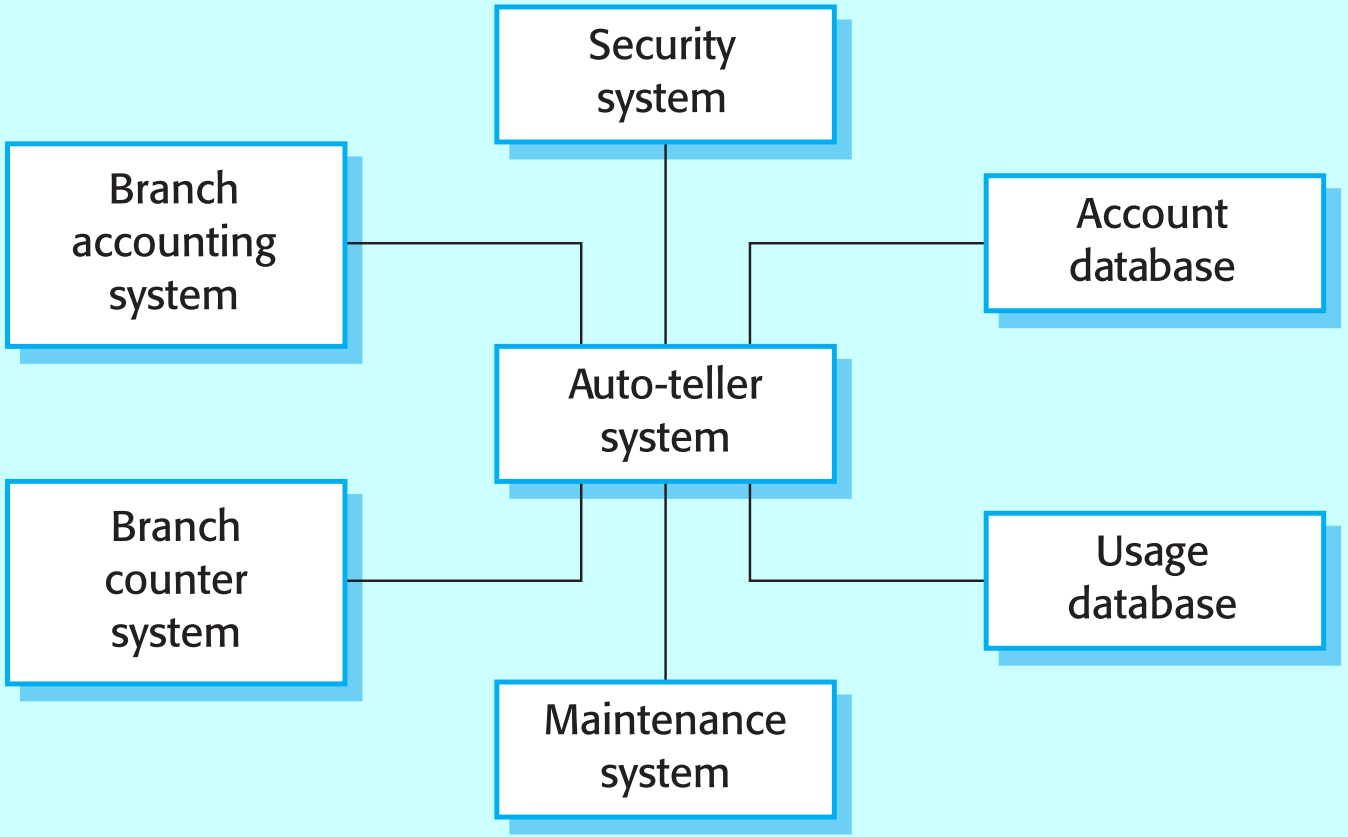
Elements of the Analysis Model



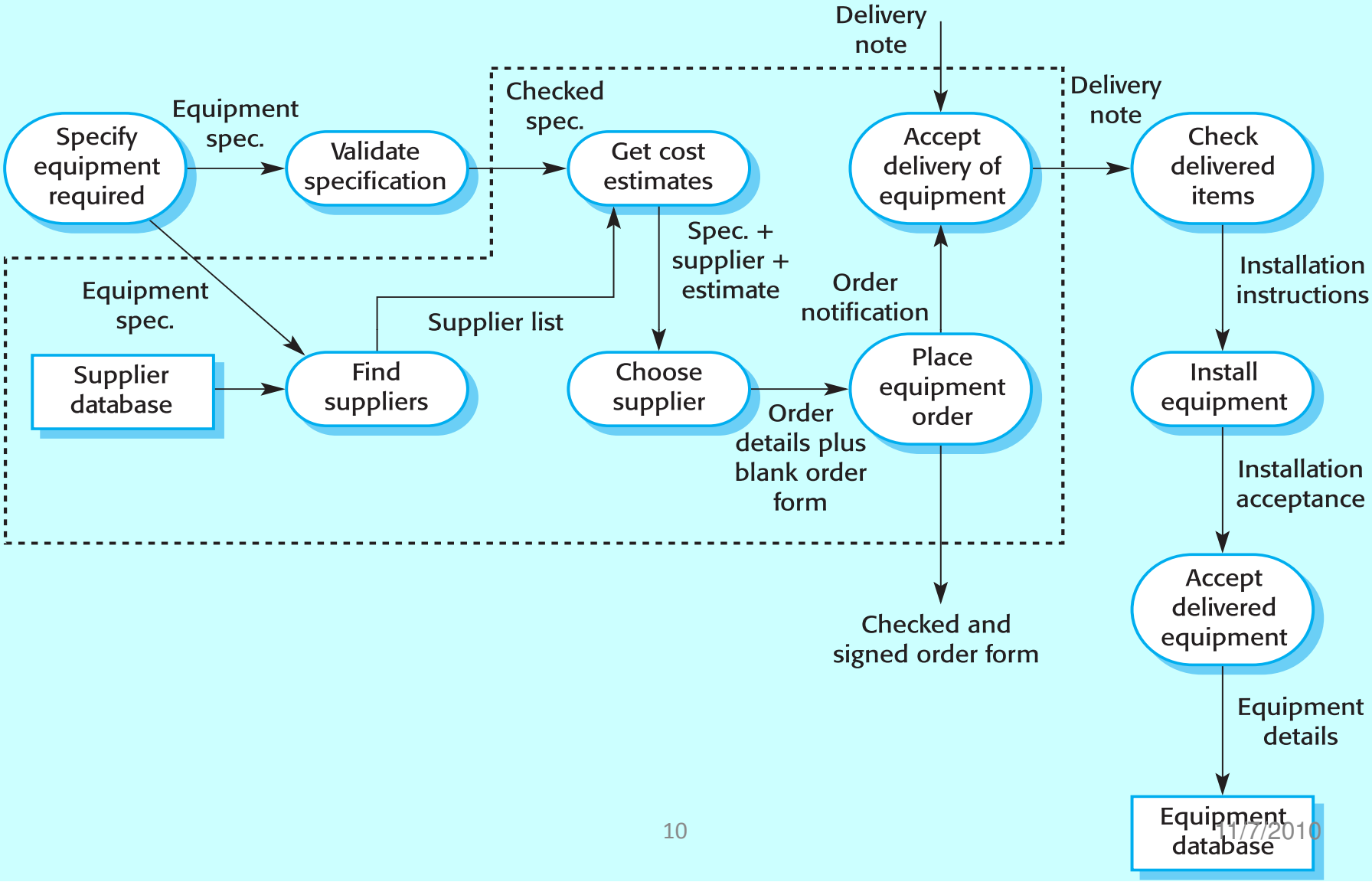
Context models

- Used at early stages in requirement elicitation and analysis.
- Context models are used to illustrate the operational context of a system - they show what lies outside the system boundaries.
- Social and organisational concerns may affect the decision on where to position system boundaries.
- Simple architectural models supplemented by process models are used for context modeling.
  - Arch. Models show environment of the system
  - Process models show how the system is related to its environment

Example: Architectural model of an Information System containing a bank Auto-teller network



Example: Process model of equipment procurement



Data Modeling

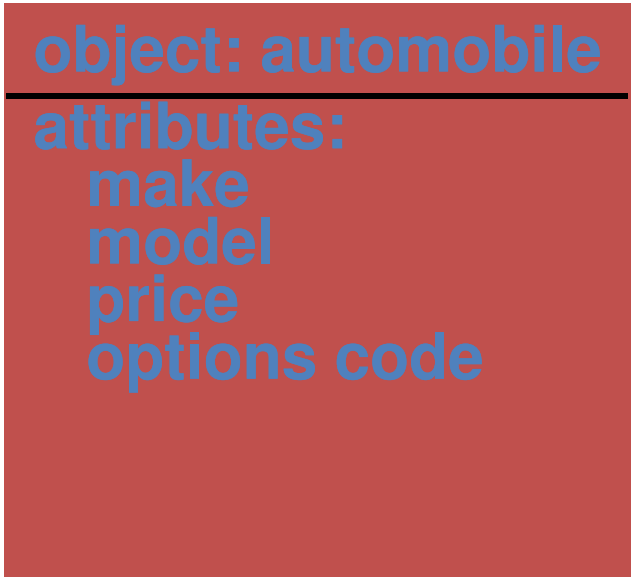
- Analysis modeling often begins with data modeling
- Examines data objects independently of processing
- Focuses attention on the data domain
- Indicates how data objects relate to one another

Typical Data Objects

- *external entities* (printer, user, sensor)
- *things* (e.g, reports, displays, signals)
- *occurrences or events* (e.g., interrupt, alarm)
- *roles* (e.g., manager, engineer, salesperson)
- *organizational units* (e.g., division, team)
- *places* (e.g., manufacturing floor)
- *structures* (e.g., employee record)

Attributes of Data Objects

A data object contains a set of attributes that act as an aspect, quality, characteristic, or descriptor of the object



What is a Relationship?

relationship —indicates “connectedness”;
a "fact" that must be "remembered"
by the system and cannot or is not computed
or derived mechanically

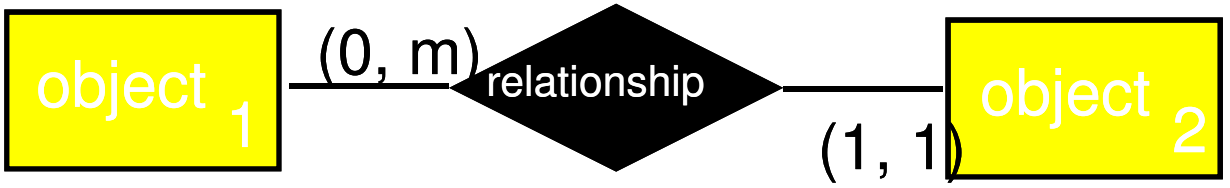
- objects can be related in many different ways

Entity Relationship Diagrams(ERD)

- Used to describe the logical structure of data processed by the system.
- Widely used in database design.

ERD Notation

One common form:

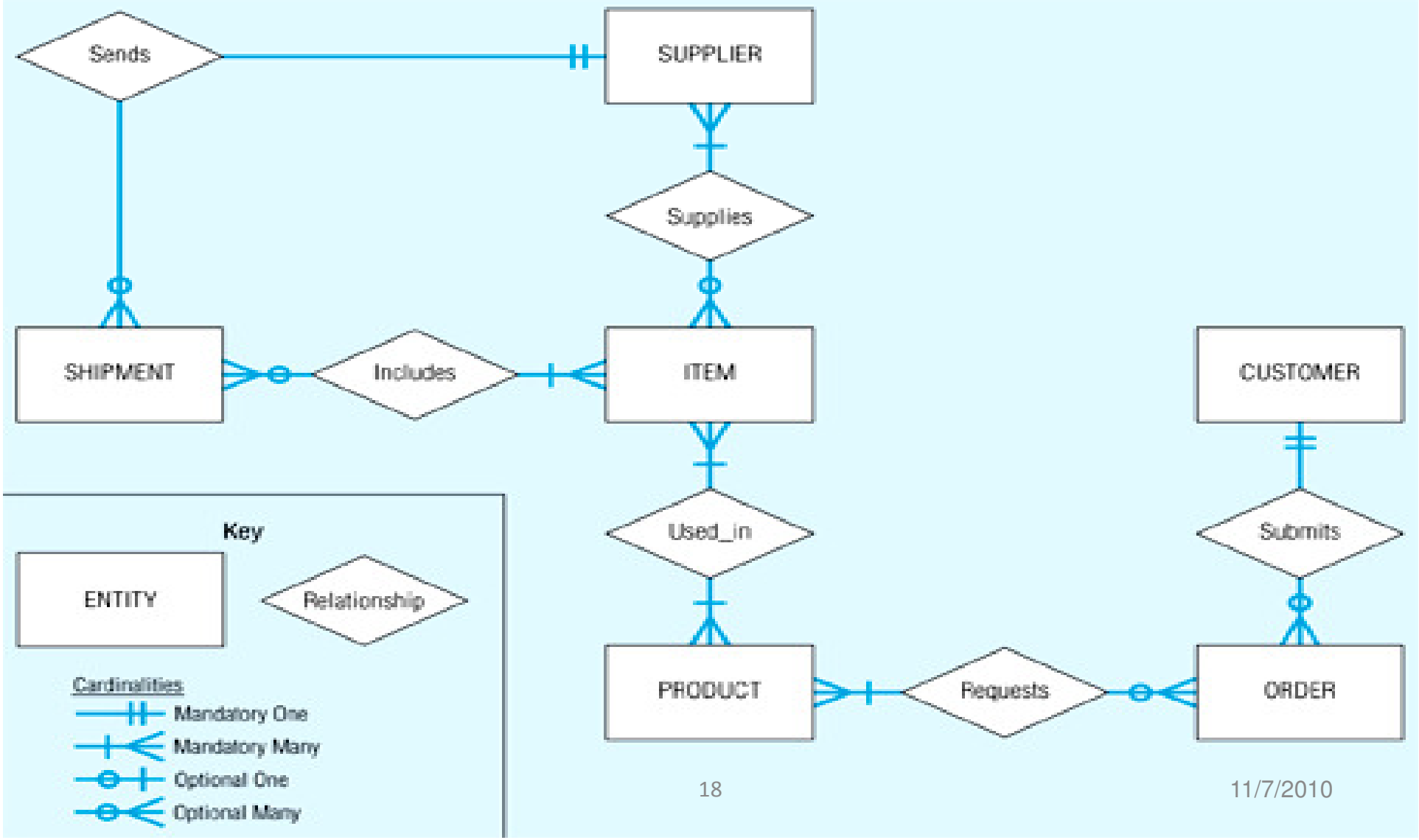


Another common form:

ER: Cardinality

- The number of instances of entity B that can 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
  - This is also called "modality".
- Maximum Cardinality
  - The maximum number of instances of entity B that may be associated with each instance of entity A

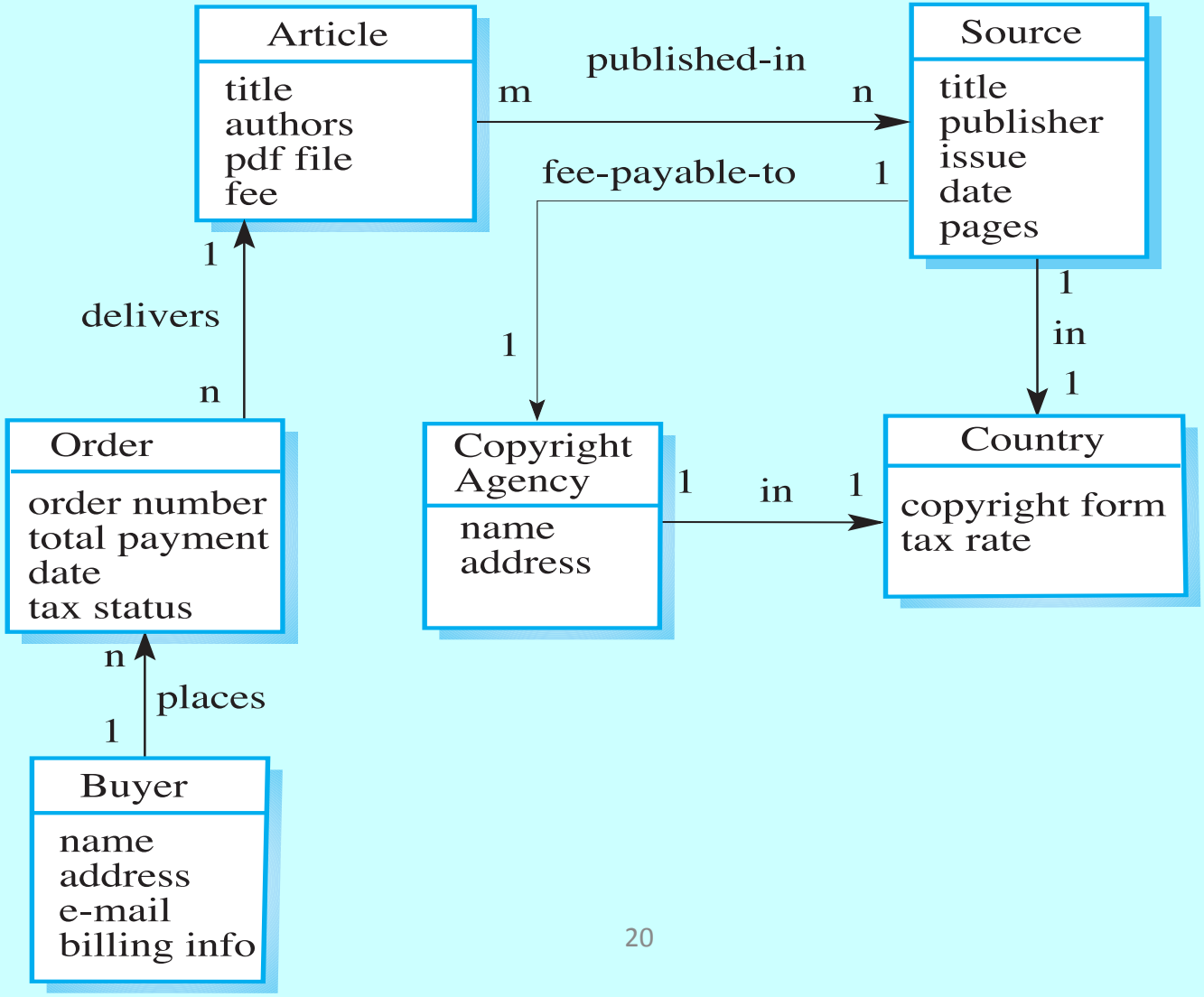
Example ERD


Sends
SUPPLIER
Supplies
SHIPMENT
Includes
ITEM
CUSTOMER
Used_in
Submits
Key
ENTITY
Relationship
Cardinalities
Mandatory One
Mandatory Many
Optional One
Optional Many
PRODUCT
Requests
ORDER

ER Diagrams in UML

- No specific notation provided in the UML but objects and associations can be used.

Example: Library semantic model in UML



Data dictionaries

- Data dictionaries are lists of all of the names used in the system models.
- Descriptions of the entities, relationships and attributes are also included.
- Advantages
  - Support name management and avoid duplication;
  - Store of organisational knowledge linking analysis, design and implementation;

Example: Data dictionary entries

Name	Description	Type	Date
Article	Details of the published article that may be ordered by people using LIBSYS.	Entity	30.12.2002
authors	The names of the authors of the article who may be due a share of the fee.	Attribute	30.12.2002
Buyer	The person or organisation that orders a co py of the article.	Entity	30.12.2002
fee-payable-to	A 1:1 relationship between Article and the Copyright Agency who should be paid the copyright fee.	Relation	29.12.2002
Address (Buyer)	The address of the buyer. This is used to any paper billing information that is required.	Attribute	31.12.2002

Object-oriented Analysis

- The intent of OOA is to define
  - Classes
  - Relationship of classes
  - behavior of classes

that are relevant to the problem to be solved

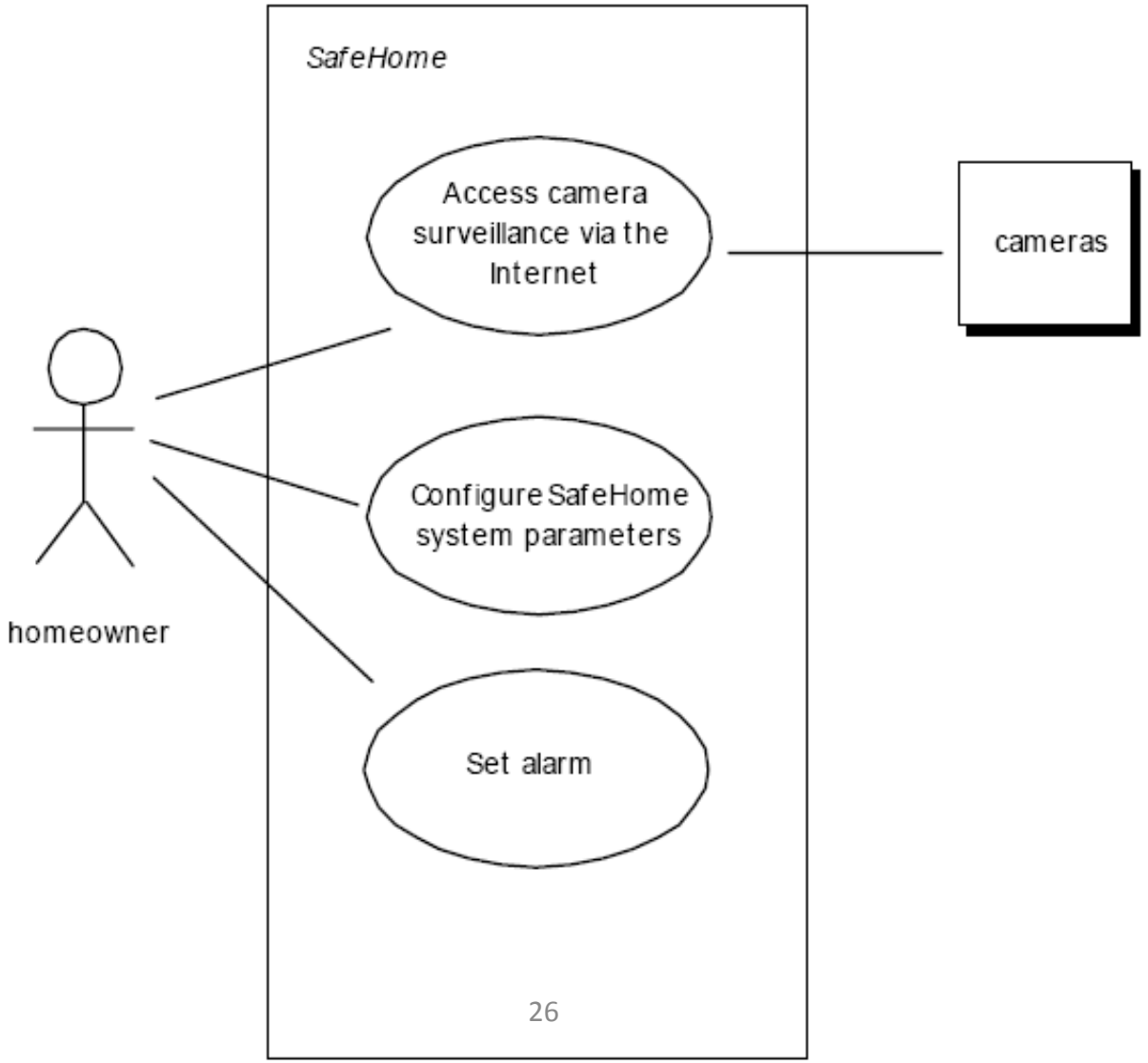
Scenario-based Modeling

- Use-Cases
  - a scenario that describes a “thread of usage” for a system
  - *Actors represent roles people or devices play as the* system functions
  - *Users can play a number of different roles for a given* scenario

Use-cases

- Developing a use case
  - What are the main tasks or functions that are performed by the actor?
  - What system information will the actor acquire, produce or change?
  - What information does the actor desire from the system?

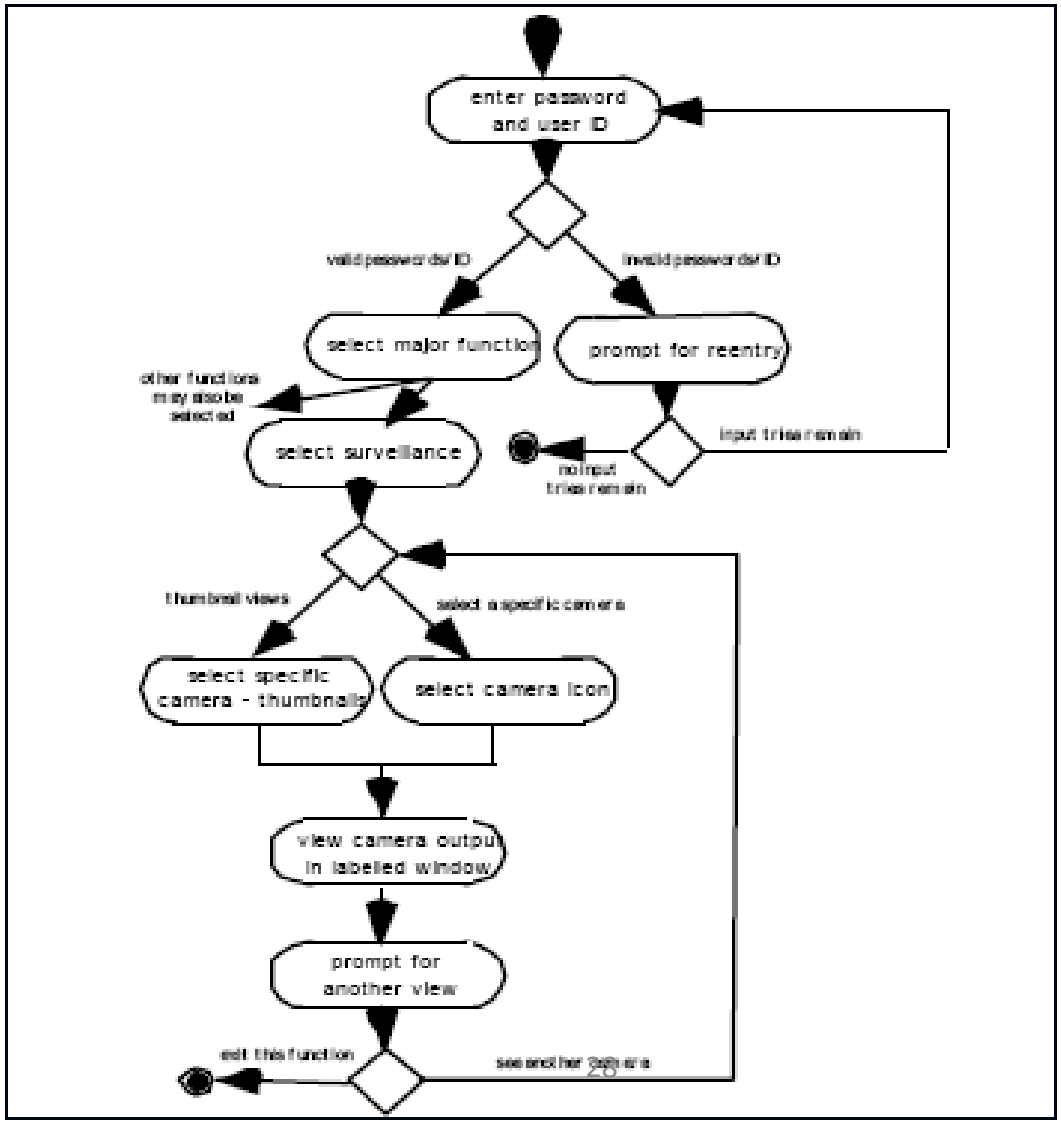
Use-case Diagram



Activity Diagrams

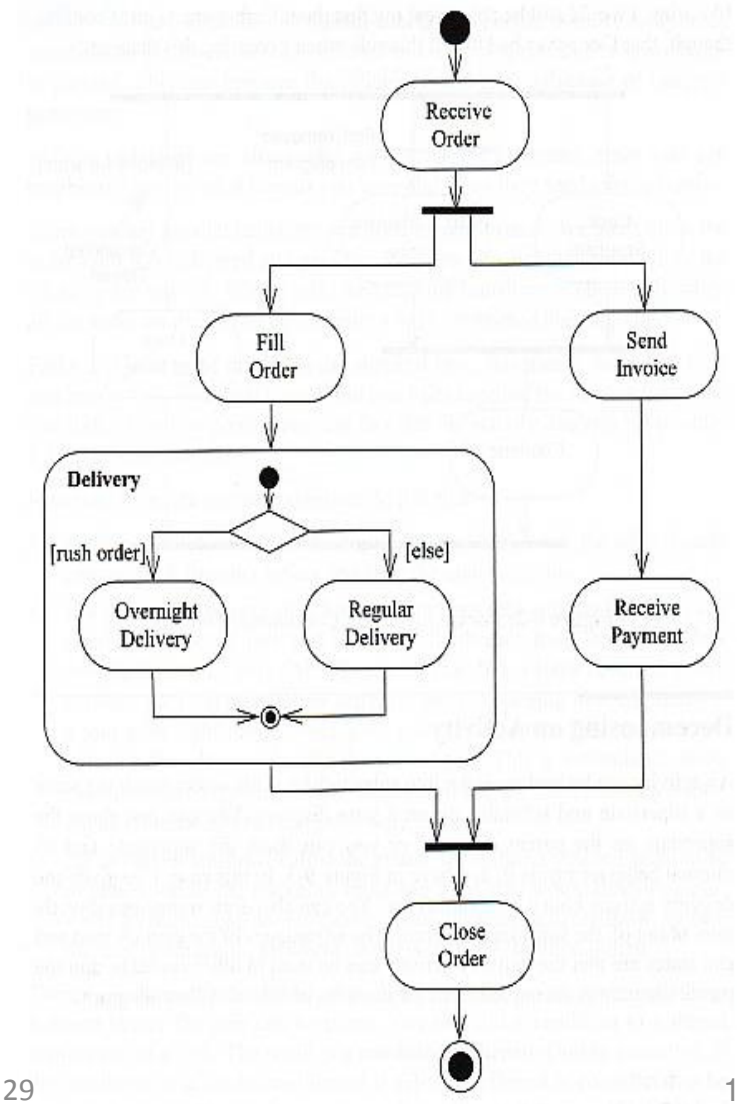
- Supplements the use-case by providing a diagrammatic representation of procedural flow

Example: Activity Diagram



Activity Diagram Example

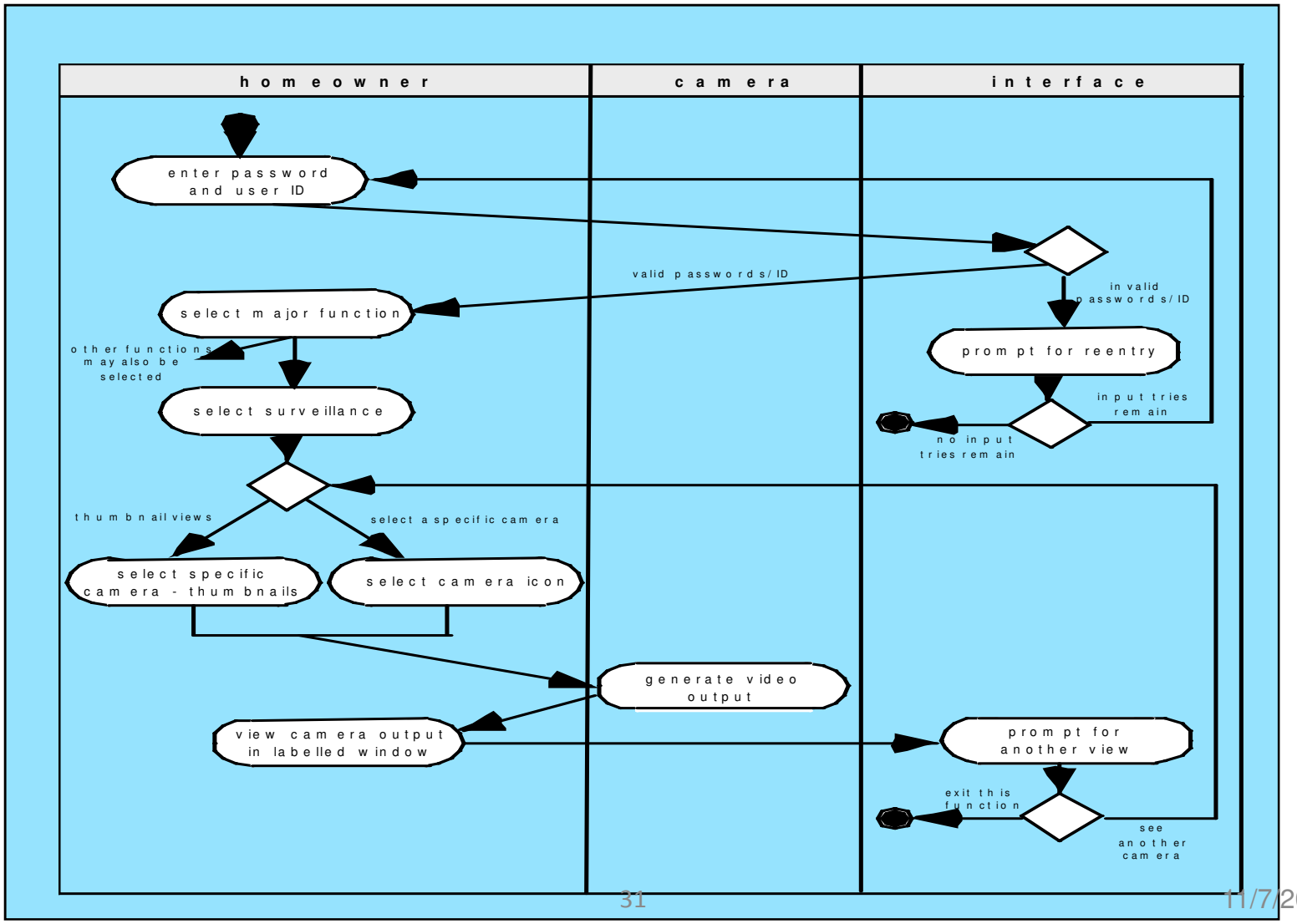
- To show concurrent activity, activity diagrams allow branches and joins.



Swim-lane Diagrams

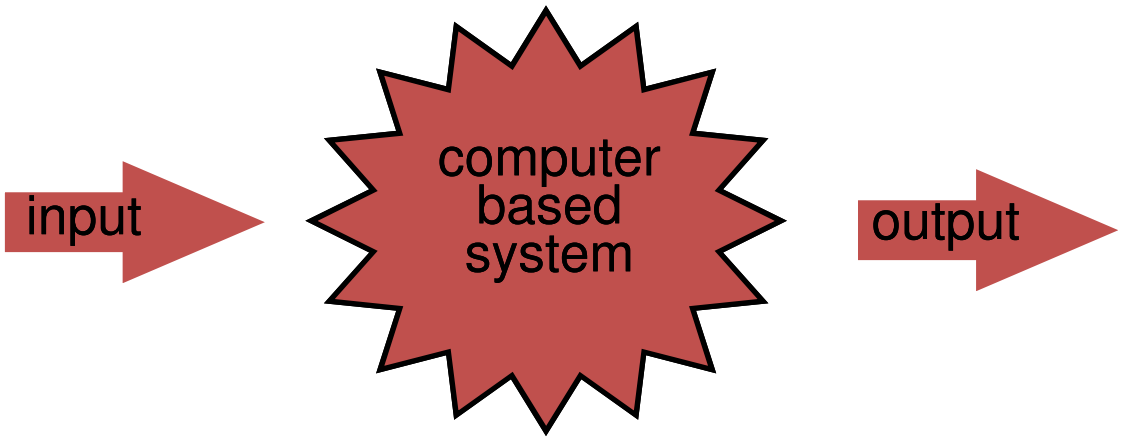
- Allows the modeler to represent the flow of activities described by the use-case
- This diagram indicates which actor or analysis class has responsibility for the action described by an activity rectangle

Swimlane Diagram Example



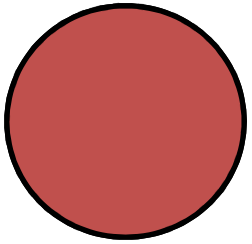
Flow-oriented Modeling

Every computer-based system is an information transform ....

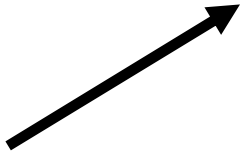


Flow Modeling Notation

external entity



process



data flow

data store

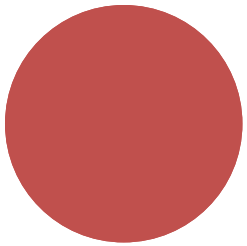
External Entity

A producer or consumer of data

Examples: a person, a device, a sensor

Another example: computer-based system

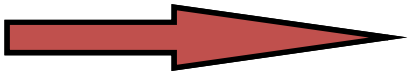
Process



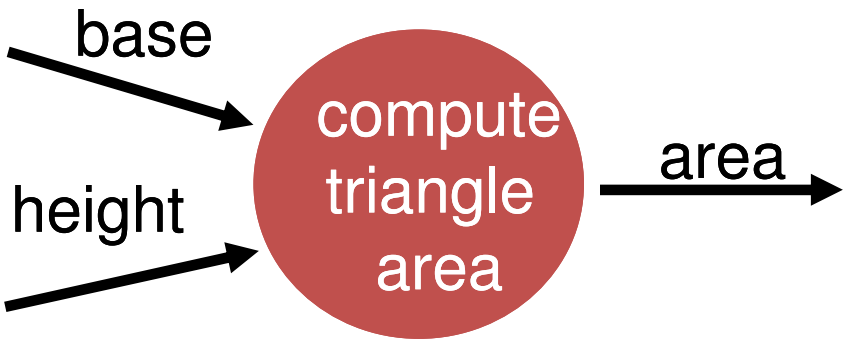
A data transformer (changes input to output)

Examples: compute taxes, determine area, format report, display graph

Data Flow

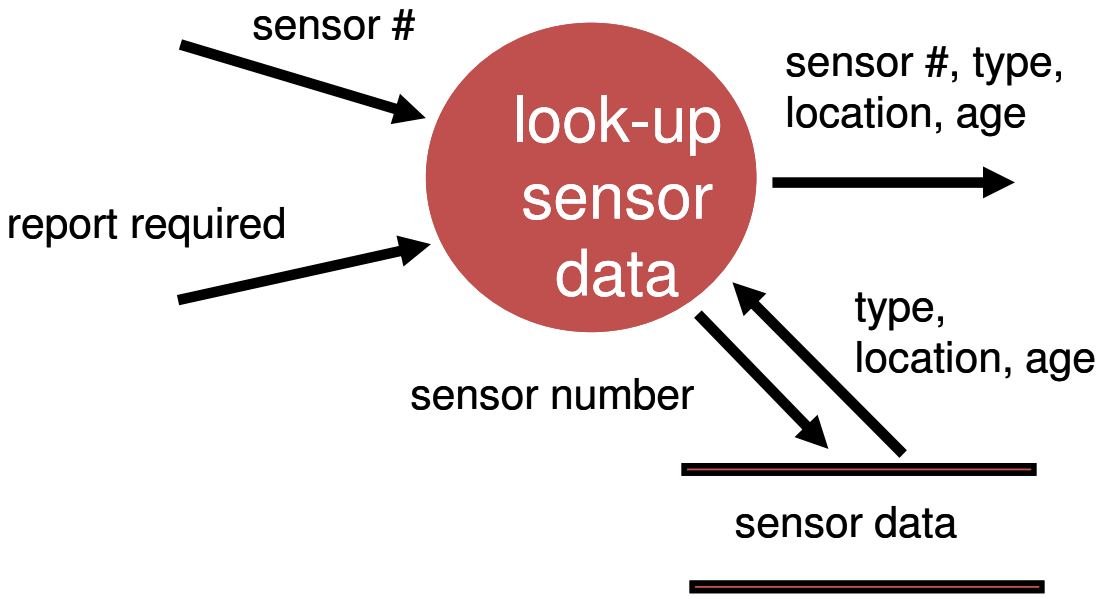


Data flows through a system, beginning as input and be transformed into output.



Data Stores

Data is often stored for later use.



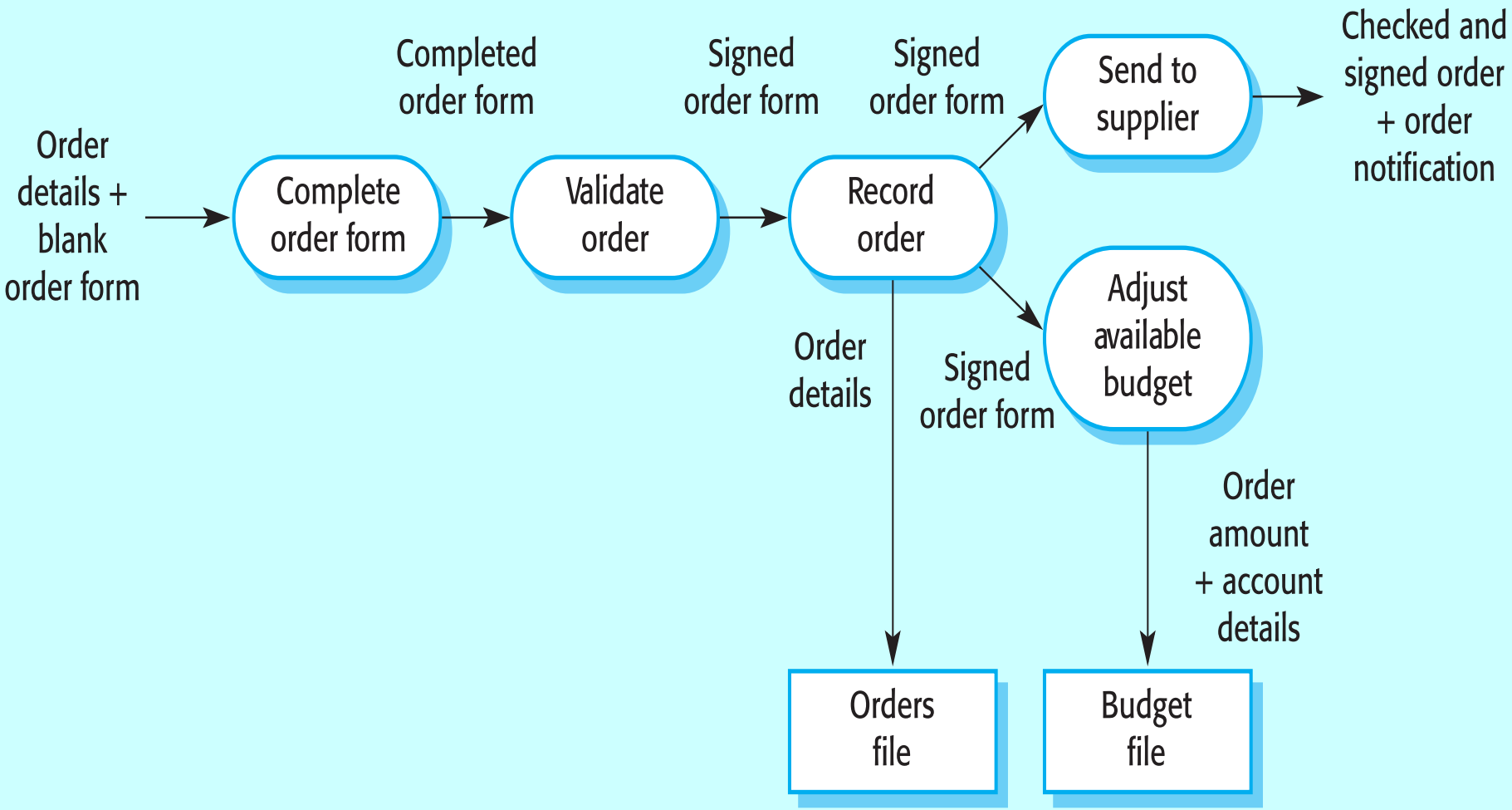
Data-flow models

- Data flow diagrams (DFDs) may be used to model the system's data processing.
- DFDs model the system from a functional perspective.
- These show the processing steps as data flows through a system.
- Simple and intuitive notation that customers can understand.
- Show end-to-end processing of data.

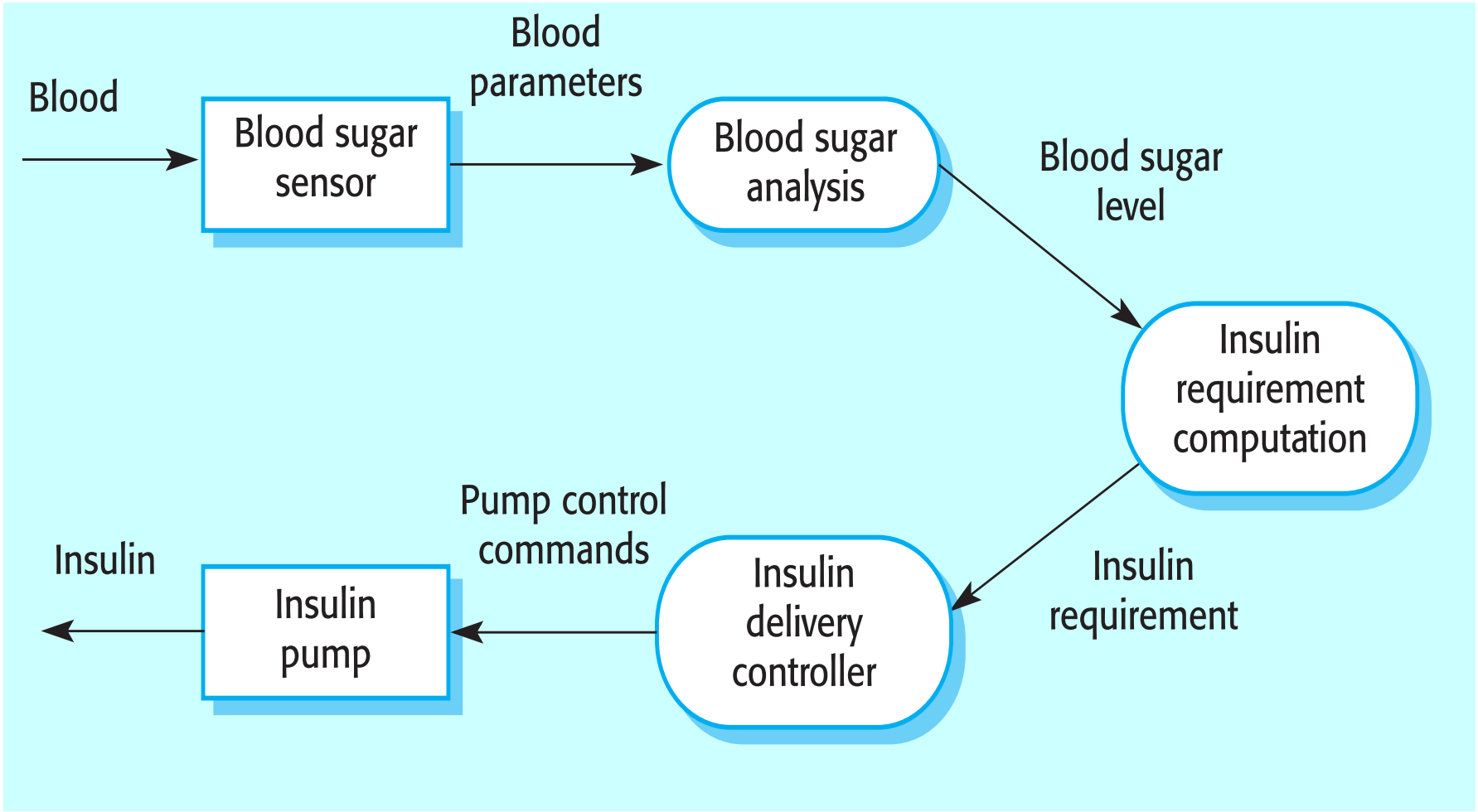
Data flow diagrams

- Data flow diagrams may also be used in showing the data exchange between a system and other systems in its environment.
- Focus on what transforms happen , how they are done is not important
- Usually major inputs/outputs shown, minor are ignored in this modeling
- No loops , NO conditional thinking , …
- No algorithmic design/thinking.

Example: Order processing DFD



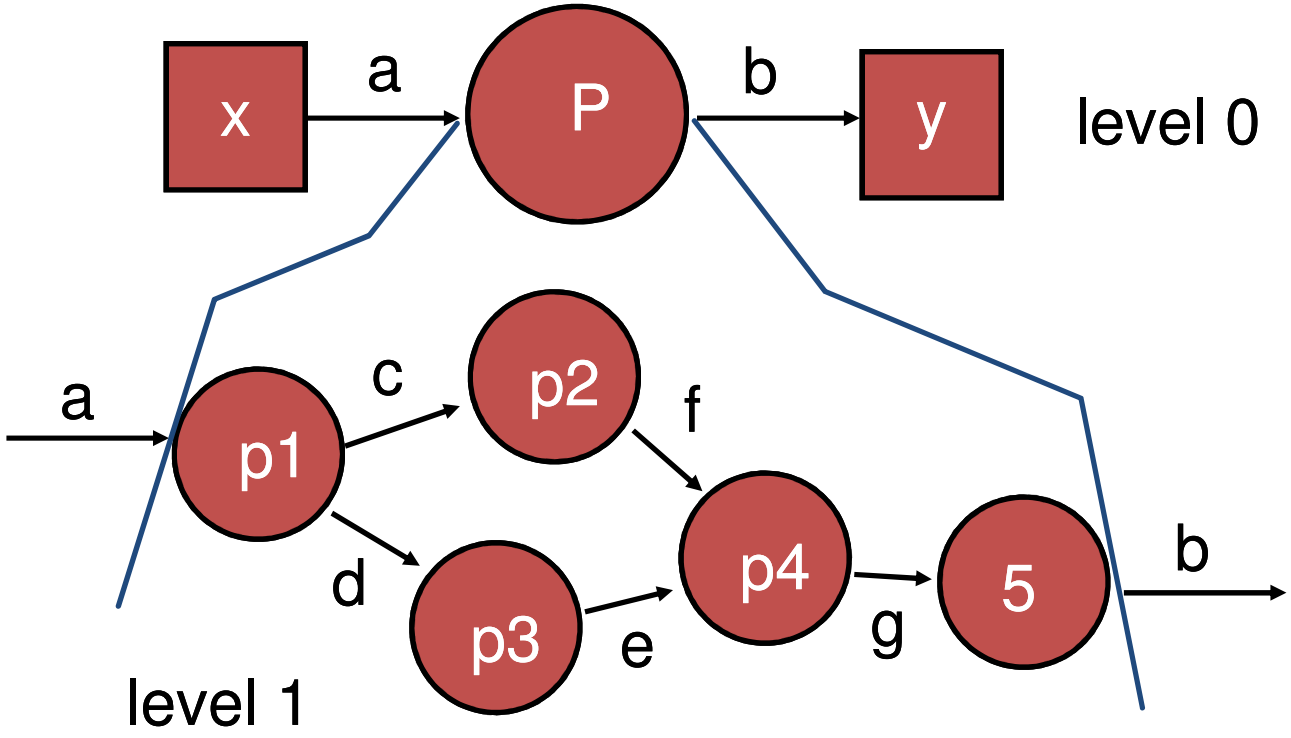
Example: Insulin pump DFD



Leveled DFD's

- DFD of a system may tend to get very large
- Can organize it hierarchically
- Start with a top level DFD with a few abstract processes
- Then draw a separate DFD for each process (called "exploding")
- Preserve I/O when exploding

The Data Flow Hierarchy



Top level DFD

- Also serves as a context model
- Views the entire system as a single transform and identifies the context
- Is a DFD with one transform (system), with all inputs, outputs, sources, sinks for the system identified

Flow Modeling Notes

- each bubble is refined until it does just one thing
- the expansion ratio decreases as the number of levels increase
- most systems require between 3 and 7 levels for an adequate flow model
- a single data flow item (arrow) may be expanded as levels increase (data dictionary provides information)

DFD Drawing – Common Errors

- Unlabeled data flows
- Missing data flows
- Consistency not maintained during refinement
- Missing processes

Class-based modeling

- Object models describe the system in terms of object classes and their associations.
- An object class is an abstraction over a set of objects with common attributes and the services (operations) provided by each object.

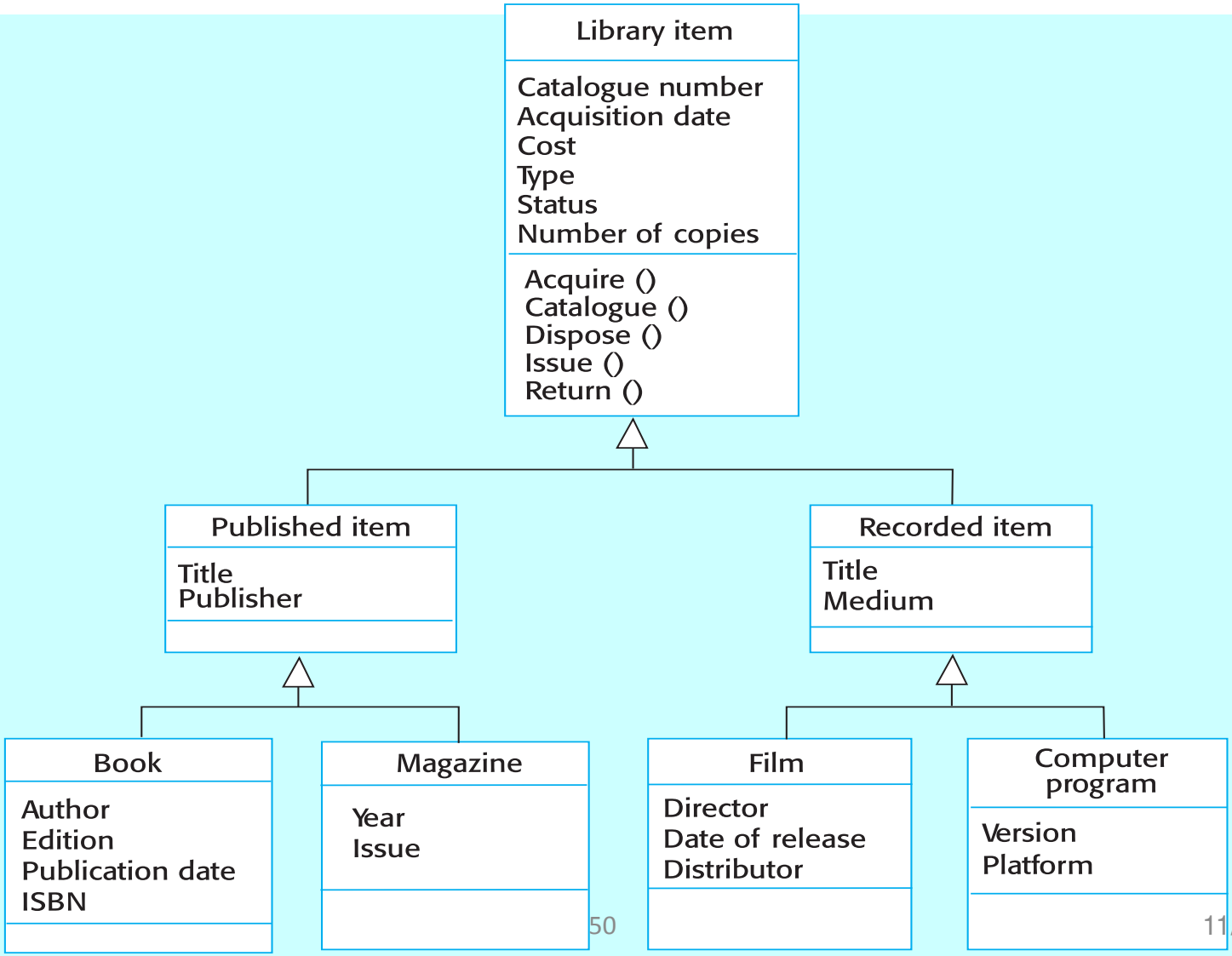
Class-based models and the UML

- The UML is a standard representation devised by the developers of widely used object-oriented analysis and design methods.
- It has become an effective standard for object-oriented modelling.
- Notation
  - Object classes are rectangles with the name at the top, attributes in the middle section and operations in the bottom section;
  - Relationships between object classes (known as associations) are shown as lines linking objects;

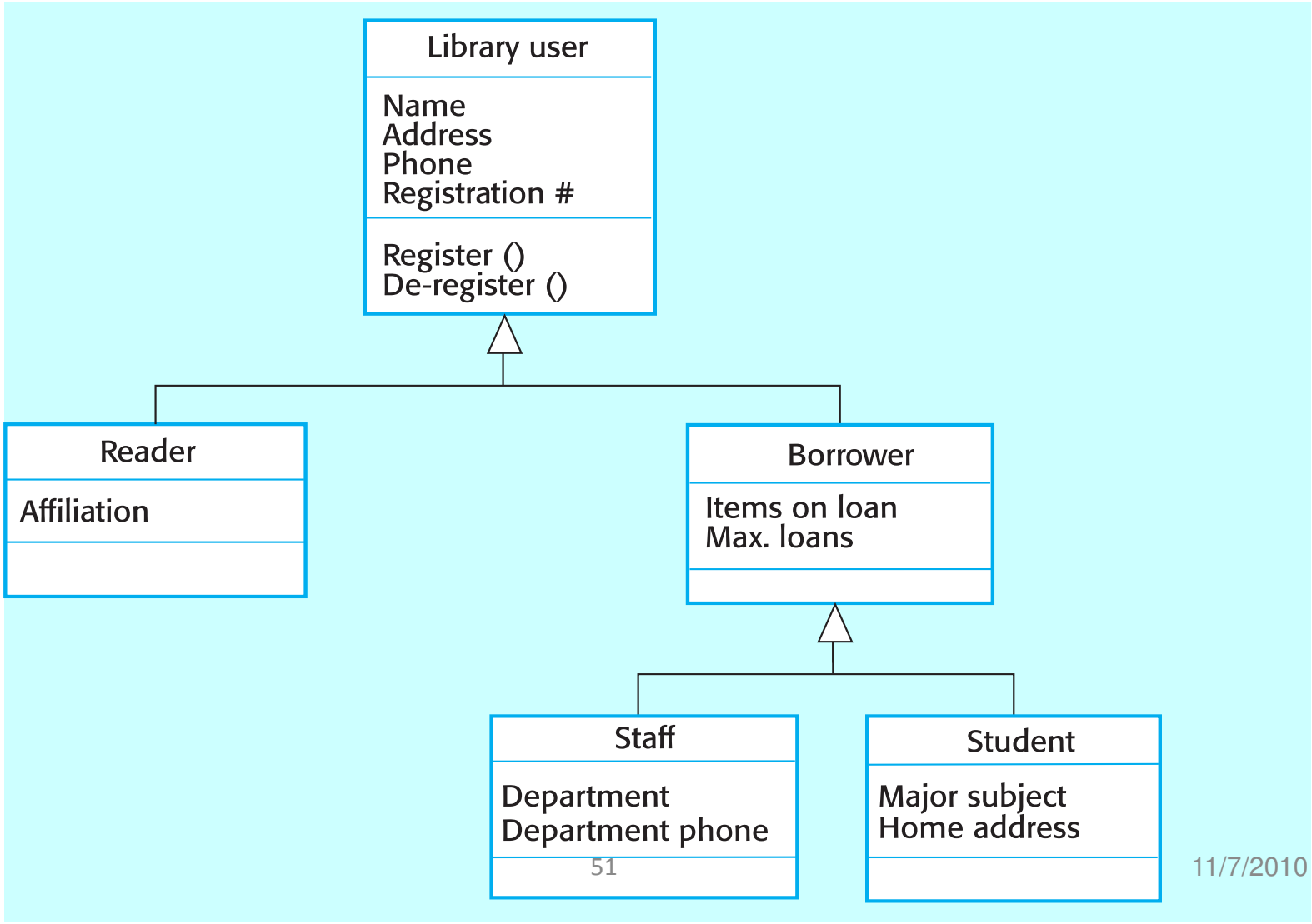
Inheritance

- Organise the domain object classes into a hierarchy.
- Classes at the top of the hierarchy reflect the common features of all classes.
- Object classes inherit their attributes and services from one or more super-classes; these may then be specialised as necessary.
- Inheritance is referred to as generalisation and in UML it is shown as ‘upwards’ arrow.

Example: Library item class hierarchy in UML



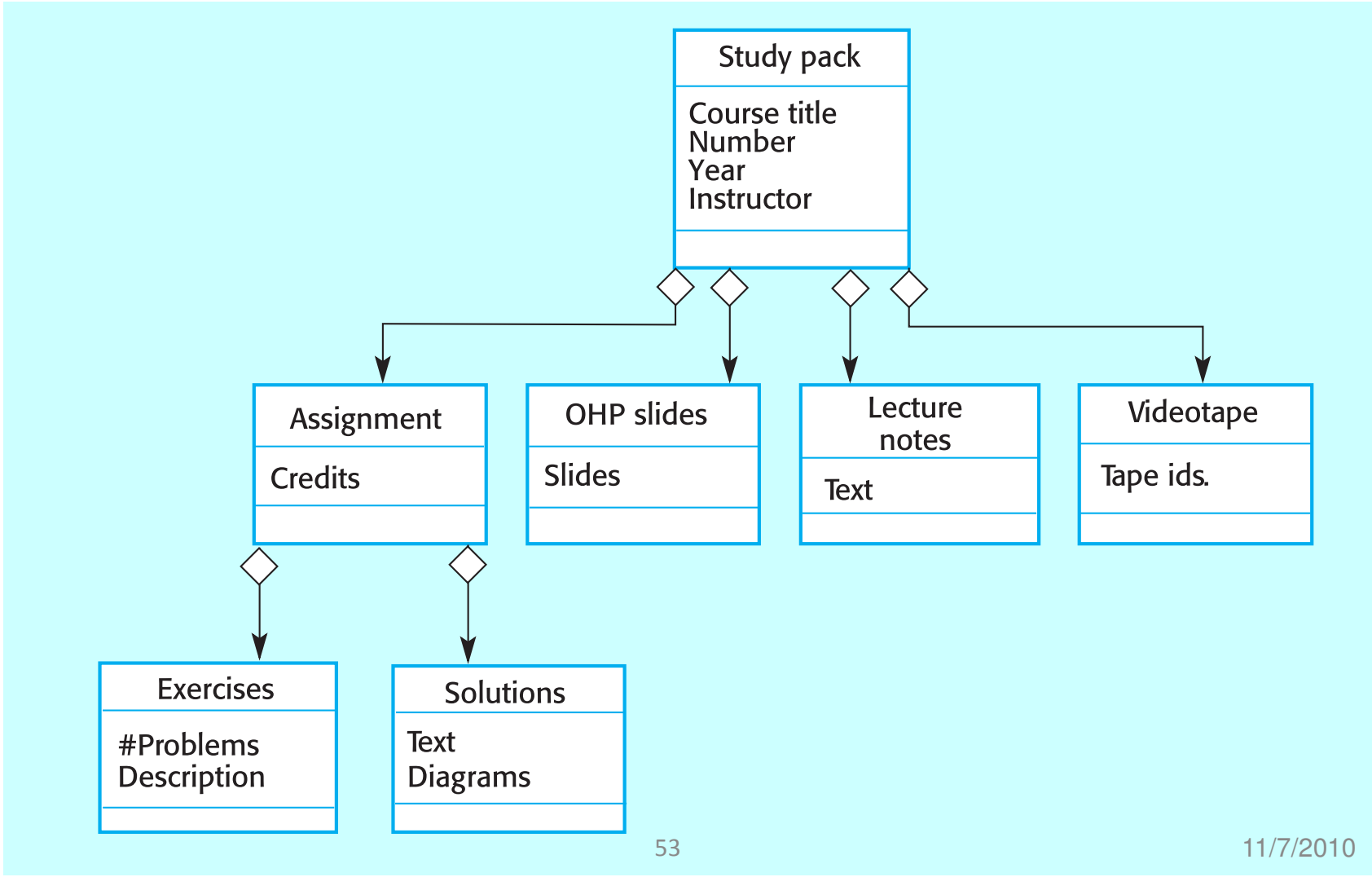
Example: Library user class hierarchy in UML



Aggregation

- An aggregation model shows how classes that are collections are composed of other classes.

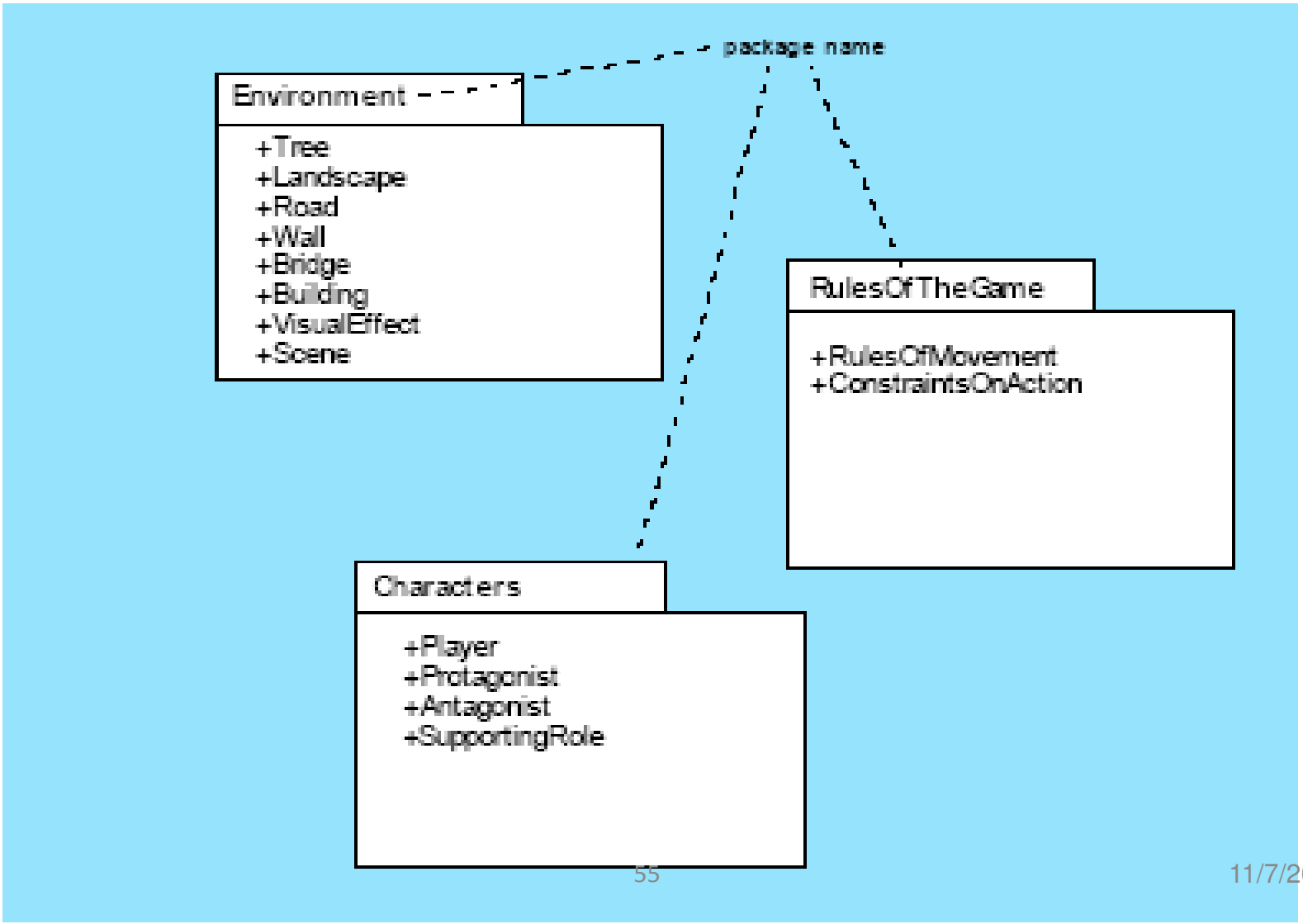
Example: Study pack as an aggregation in UML



Analysis Packages

- Various elements of the analysis model (e.g., use-cases, analysis classes) are categorized in in a manner that packages them as a grouping
- The + sign preceding the analysis class name in each package indicates that the classes have public visibility and are therefore accessible from other packages.
- A minus sign indicates that an element is hidden from all other packages and a # symbol indicates that an symbol indicates that an element is accessible only to packages contained within a given package.

Analysis Packages



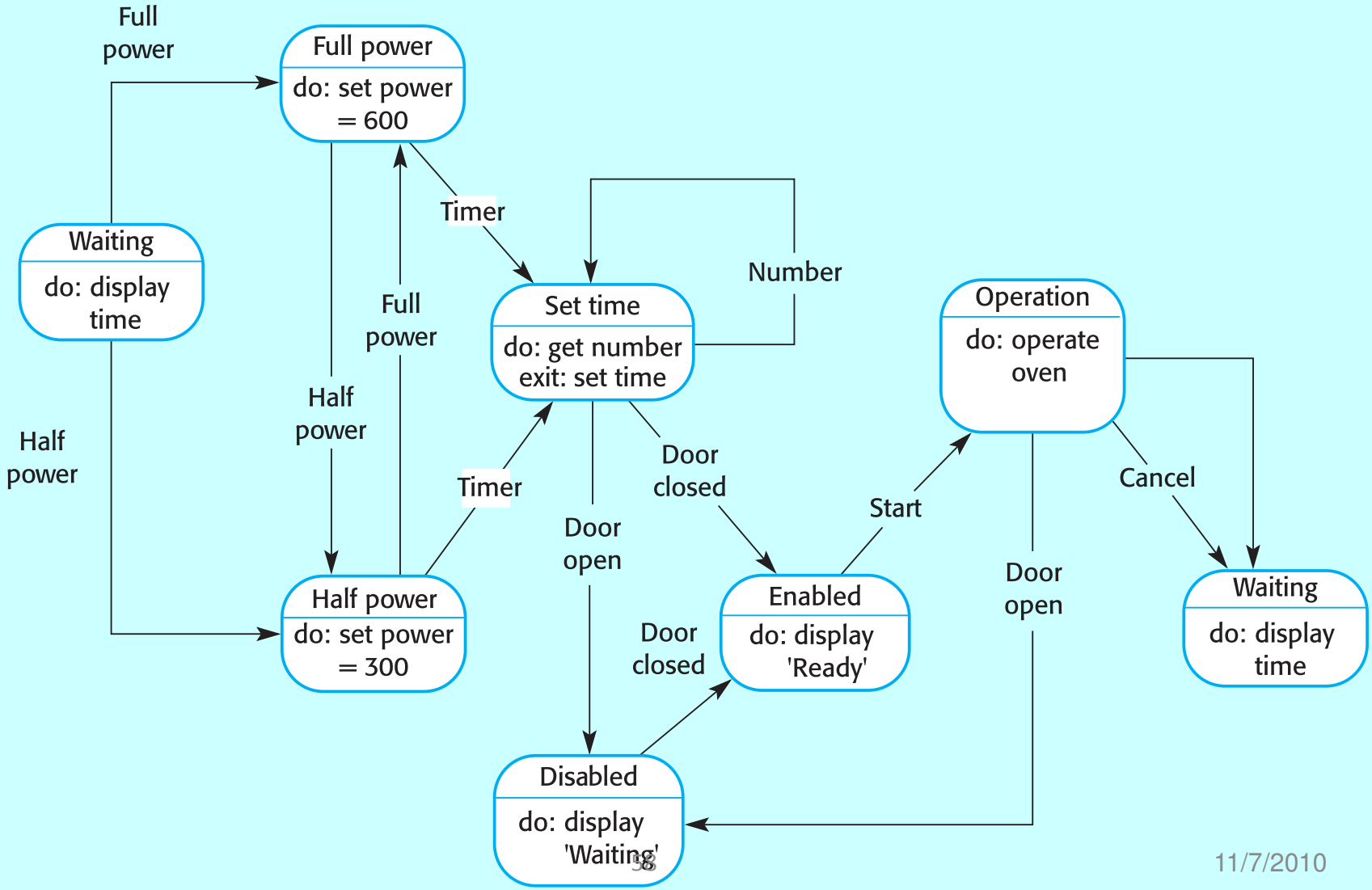
Behavioural models

- Behavioural models are used to describe the dynamic behaviour of a system.
- Express behavior of a system as a function of events

State based models

- These model the behaviour of the system in response to external and internal events.
- They show the system's responses to stimuli, so are often used for modelling real-time systems.
- State based models show system states as nodes and events as arcs between these nodes.
- When an event occurs, the system moves from one state to another.

Example: Microwave oven state model



Microwave oven state description

State	Description
Waiting	The oven is waiting for input. The display shows the current time.
Half power	The oven power is set to 300 watts. The display shows ‘Half power’.
Full power	The oven power is set to 600 watts. The display shows ‘Full power’.
Set time	The cooking time is set to the user’s input value. The display shows the cooking time selected and is updated as the time is set.

Microwave oven state description

State	Description
Disabled	Oven operation is disabled for safety. Interior oven light is on. Display shows 'Not ready'.
Enabled	Oven operation is enabled. Interior oven light is off. Display shows 'Ready to cook'.
Operation	Oven in operation. Interior oven light is on. Display shows the timer countdown. On completion of cooking, the buzzer is sounded for 5 seconds. Oven light is on. Display shows 'Cooking complete' while buzzer is sounding.

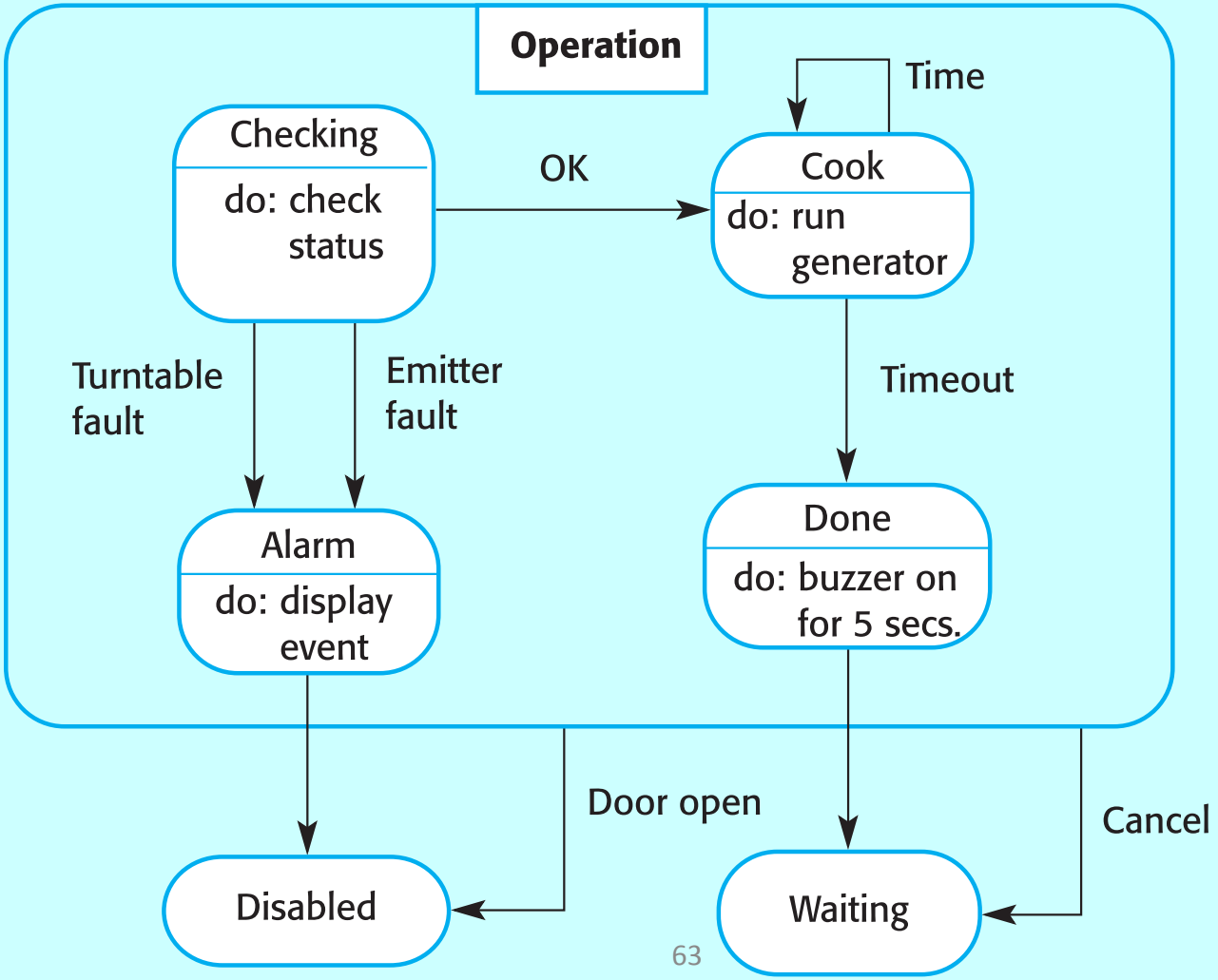
Microwave oven stimuli

Stimulus	Description
Half power	The user has pressed the half power button
Full power	The user has pressed the full power button
Timer	The user has pressed one of the timer buttons
Number	The user has pressed a numeric key
Door open	The oven door switch is not closed
Door closed	The oven door switch is closed
Start	The user has pressed the start button
Cancel	The user has pressed the cancel button

State Modelling with UML Statecharts

- Allow the (hierarchical) decomposition of a model into sub-models (see next slide).
- A brief description of the actions is included following the ‘do’ in each state.
- Can be complemented by tables describing the states and the stimuli.

Example: Microwave oven operation state expansion



Key points

- A model is an abstract system view. Complementary types of models provide different system information.
- Context models show the position of a system in its environment with other systems and processes.
- Data flow models may be used to model the data processing in a system.
- State machine models model the system's behaviour in response to internal or external events

Key points

- data models describe the logical structure of data which is imported to or exported by the systems.
- Class based models describe logical system entities, their classification and aggregation.
- Scenario based models show the interactions between actors and the system objects that they use.