System Models

Classification

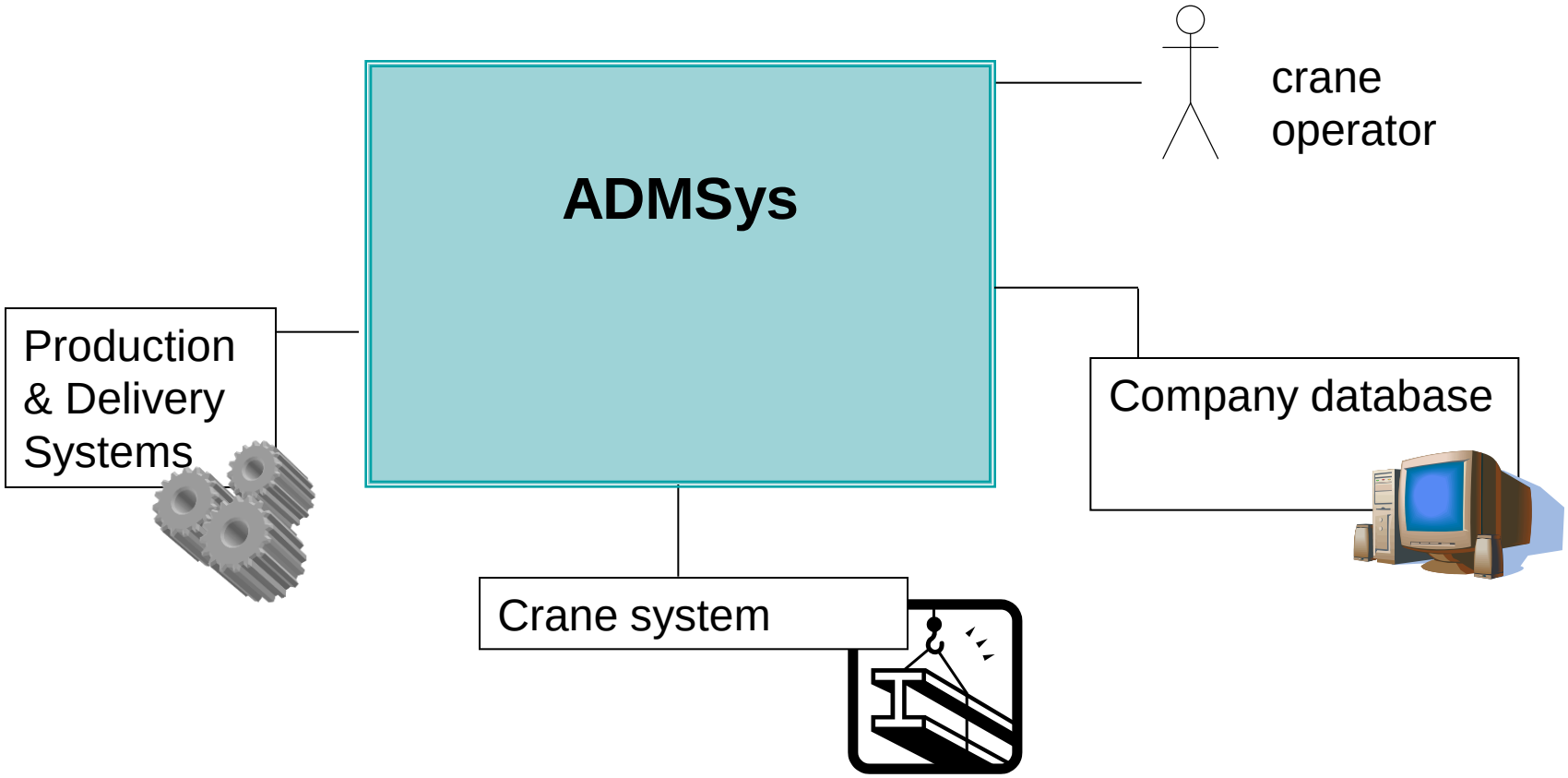
- Context models
- Behavioural models
  - Data-flow models
  - State machine models
- Data models
- Object models
  - Inheritance models
  - Object aggregation
  - Object behaviour models

Context models

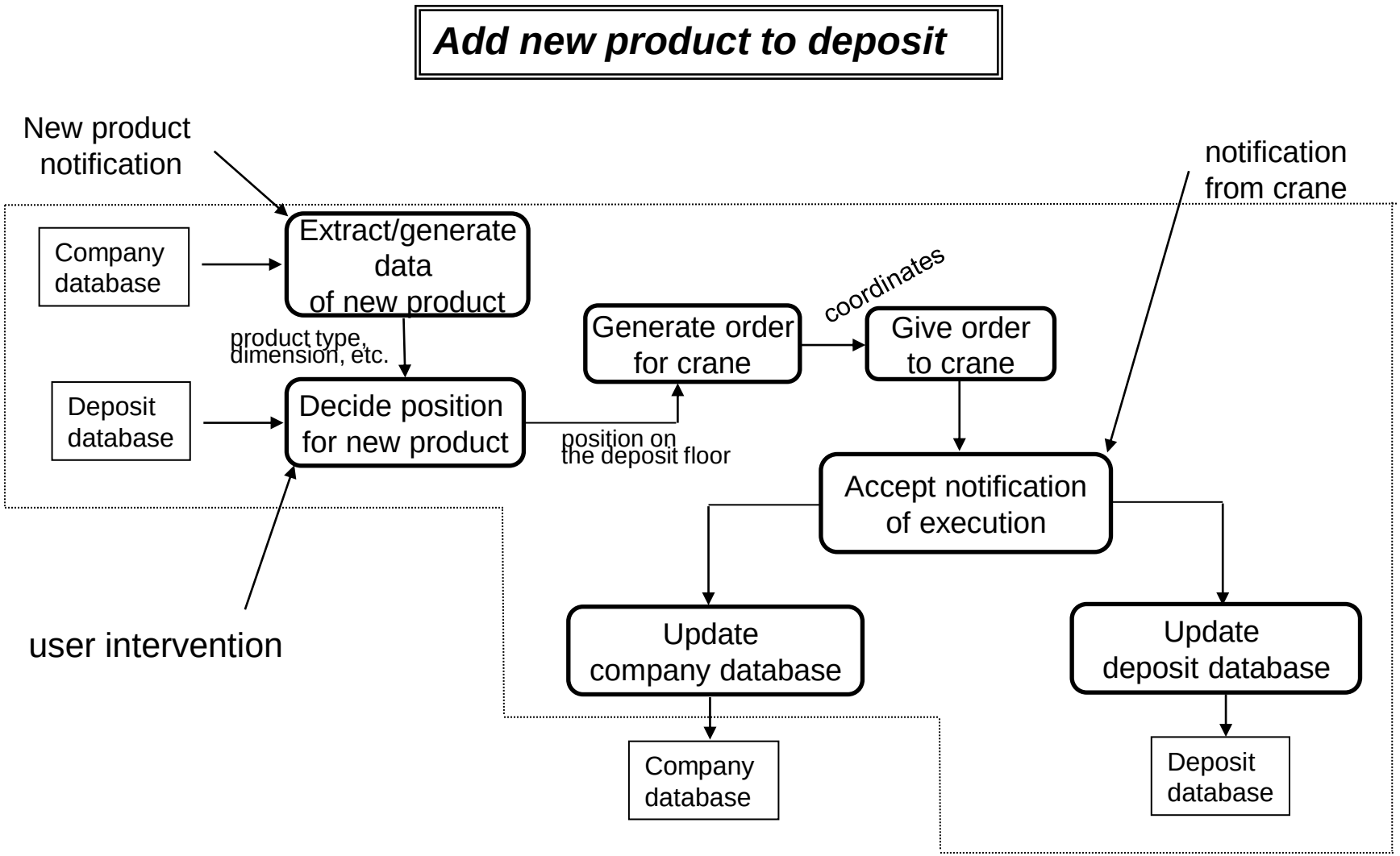
- Very general *architectural models*
  - illustrate the operational context of a system
    - show what lies outside the system boundaries
    - show the system and its relationship with other systems
- Supplemented with more detailed models
  - **Process models** – show process activities
  - **Data-flow models** – transfer of data

Context models

Context model example: ADMSys



Process model: example



Behavioral models

Behavioral models

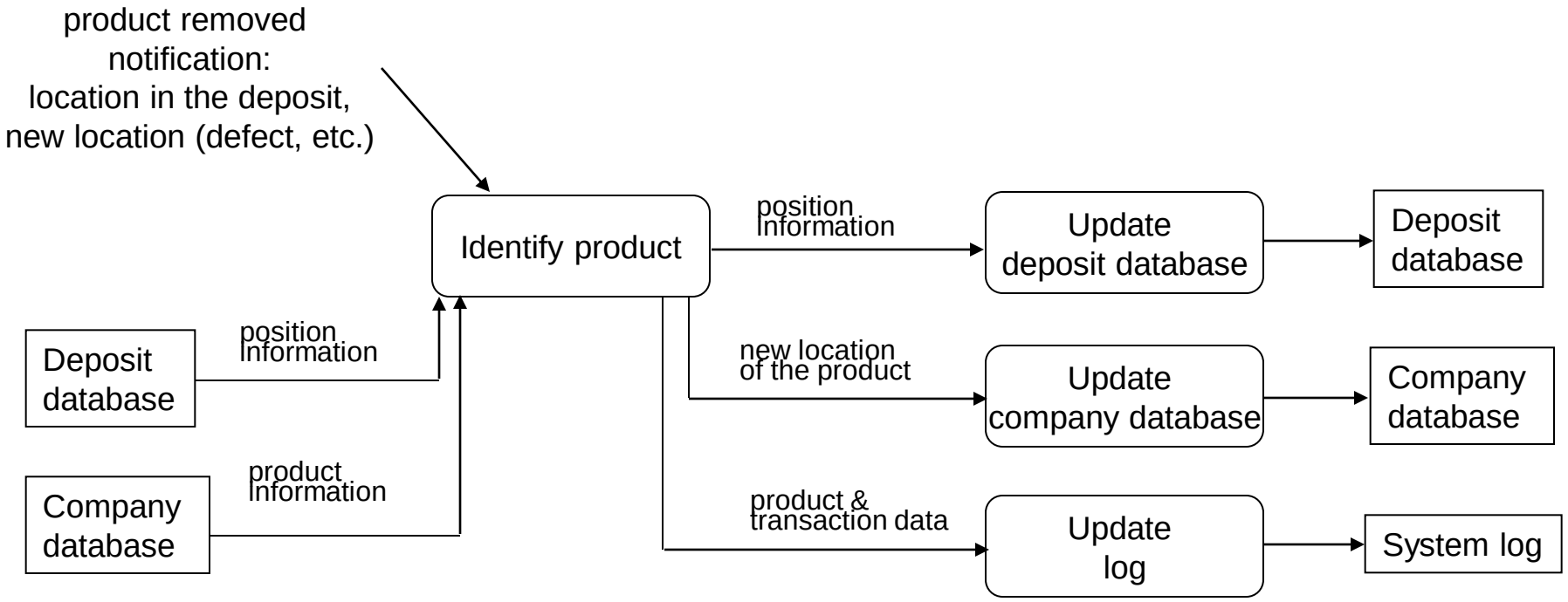
- Describe the overall behavior of a system
- Types:
  - Process models
    - show main processing steps of the whole system, or
    - only a feature / functional requirement;
  - Data flow models
    - show how data is processed as it moves through the system;
  - State machine models
    - show the systems response to events.

Data-flow models

- data-flow diagrams:
  - model the system's data processing
  - show the *processing steps* as data ***flows*** through a system
  - intrinsic part of many analysis methods
  - should use simple and intuitive notations that customers can understand
  - show end-to-end processing of data

Data-flow diagrams: Example

- Removal of a product from the deposit



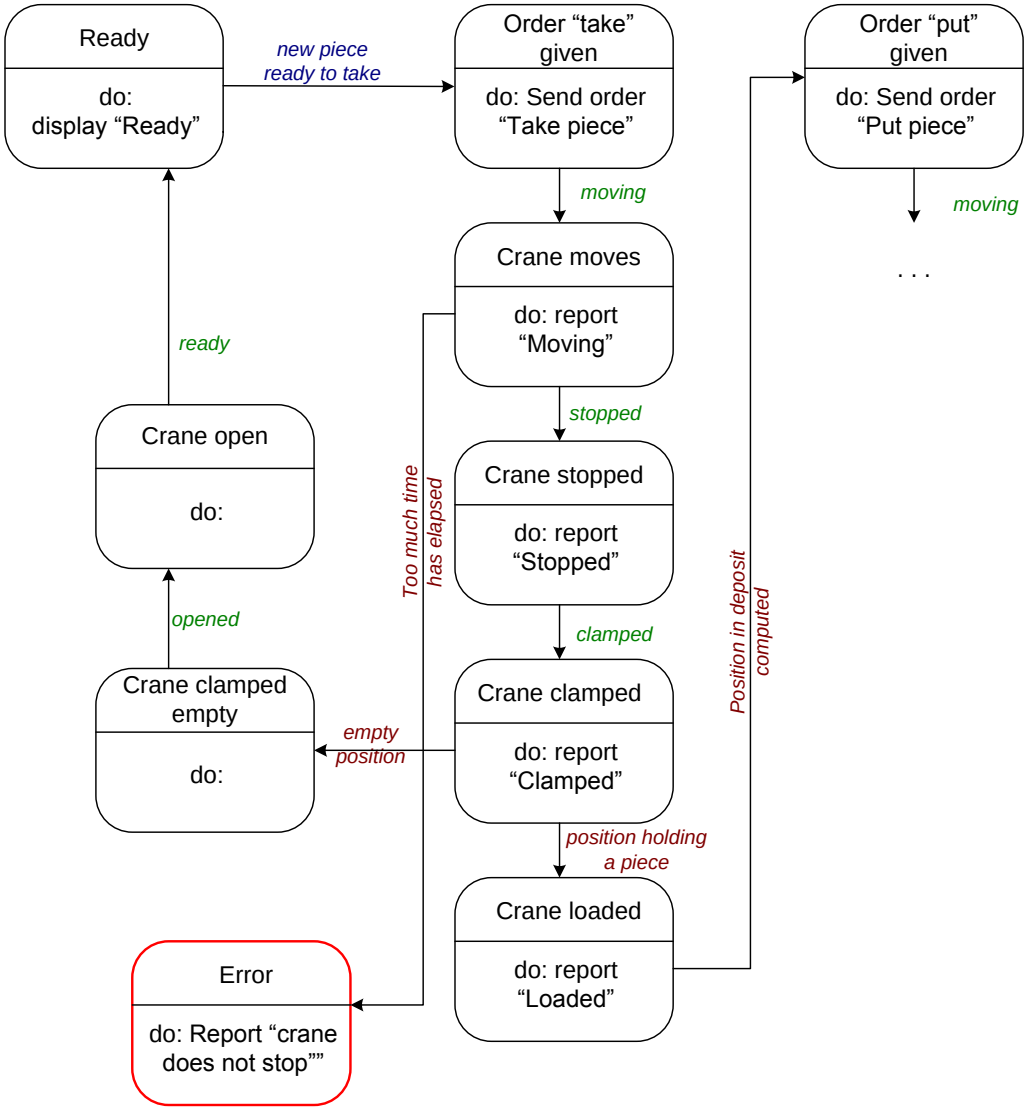
State machine diagrams

- model the *behaviour* of the system in response to external and internal *events*.
- show the system's *responses* to *stimuli*.
- often used for modelling *real-time systems.*
- show ***system states*** as nodes and ***events*** as arcs between these nodes.

- **the system *stays* in a state**,
- when an event occurs, the system ***changes*** *into* another state.
- *state charts* - used to represent state machine models.

State machine diagrams: example

States
of the crane sub-system
in ADMSys
(excerpt)



State machine models: Description

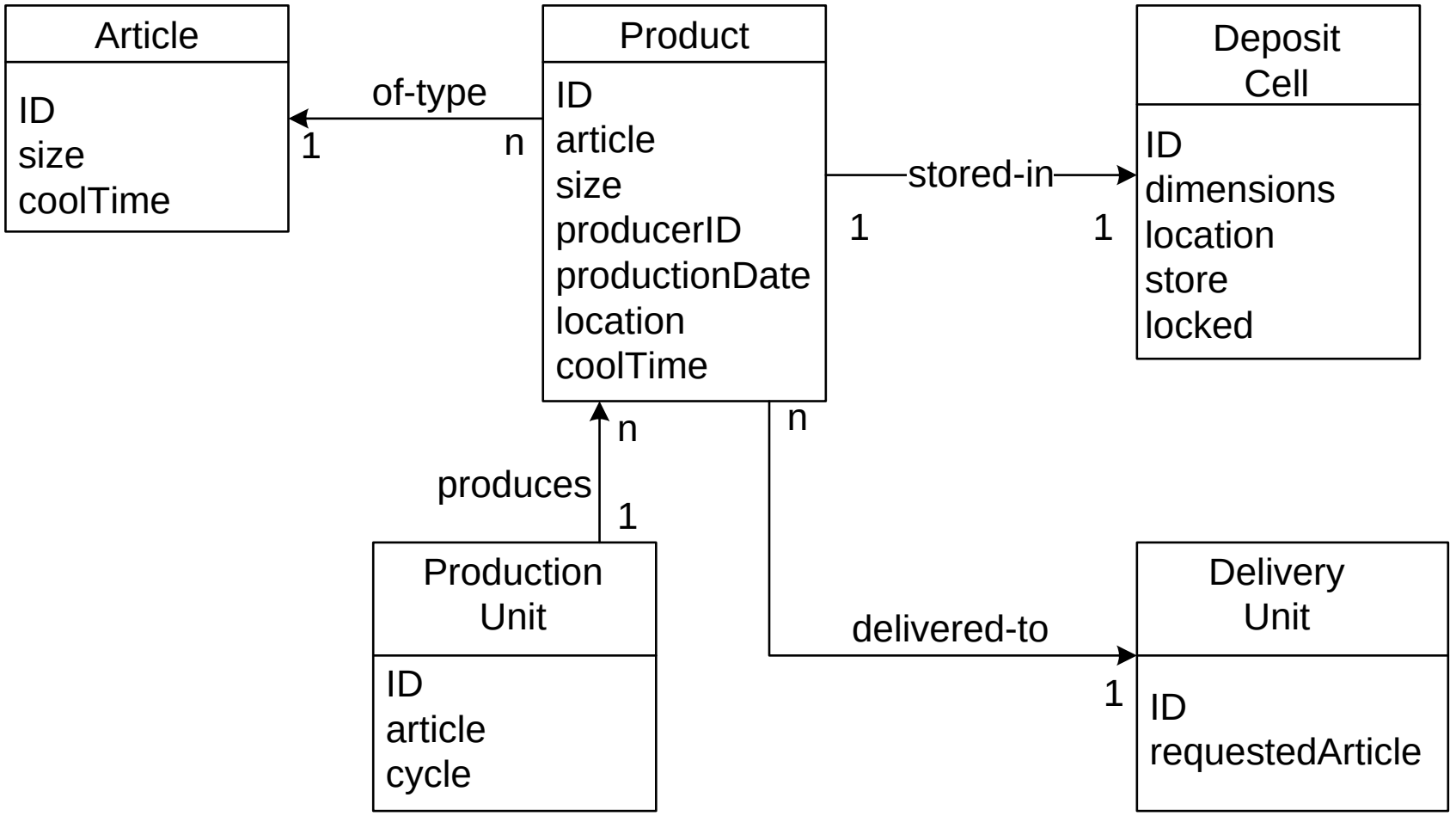
State	Description
Ready	The crane component is waiting for an order.
Order “take” given	The crane component launches an order “take piece” to the crane PLC.
Crane moves	The crane component is set to “moving”. The display shows “moving”.
Event	**Description**
New piece ready to take	The production unit has announced that a new piece is on the outlet.
Empty position	The deposit database has announced that at the crane position there is an empty cell.
Clamped	The crane PLC has announced that the crane clamped.

Data Models

Data models

- sometimes called “semantic data models”
- describe the *logical structure* of data
- an *entity-relation-attribute model* sets out
  - the ***entities*** in the system,
  - the ***relationships*** between these entities
  - the ***attributes*** of each entity
- used in database design
- easily implemented using relational databases

Data models


Article
ID
size
coolTime
of-type
1
n
Product
ID
article
size
producerID
productionDate
location
coolTime
stored-in
1
1
Deposit Cell
ID
dimensions
location
store
locked
n
produces
n
1
Production Unit
ID
article
cycle
n
delivered-to
1
Delivery Unit
ID
requestedArticle

Data models: description

- collect detailed descriptions of all entities, relations and attributes in *dictionaries* or *repositories*:
  - For name management;
  - As a storage for organisational information.

Name	Description	Type	Date
Article (Product)	Name of the article to which a product belongs.	Attribute	23.04.2006
Product	Details of the product stored in the deposit.	Entity	23.04.2006
stored-in	A 1-1 relationship between a product and the cell in the deposit which holds it.	Relation	23.04.2006

Object models

Object models

- describe the system in terms of object classes and their associations.
- Object class
  - abstraction over a set of objects with common attributes, and the services (operations) provided by each object.
- Types of object models:
  - Inheritance models;
  - Aggregation models;
  - Interaction models.

Object models

- natural way of reflecting the real-world entities manipulated by the system
- object class identification
  - difficult
  - requires a deep understanding of the application domain

- *object classes reflecting domain entities are / should be **reusable** across systems*

Object models and the UML

- UML – Unified Modelling Language
  - standard representation
  - devised by the developers of widely used *object-oriented analysis* and *design* methods.
- effective standard for object-oriented modelling
- notation
  - *Object classes* are rectangles with the *name* on 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* is referred to as generalization and is shown ‘upwards’ rather than ‘downwards’ in a hierarchy.

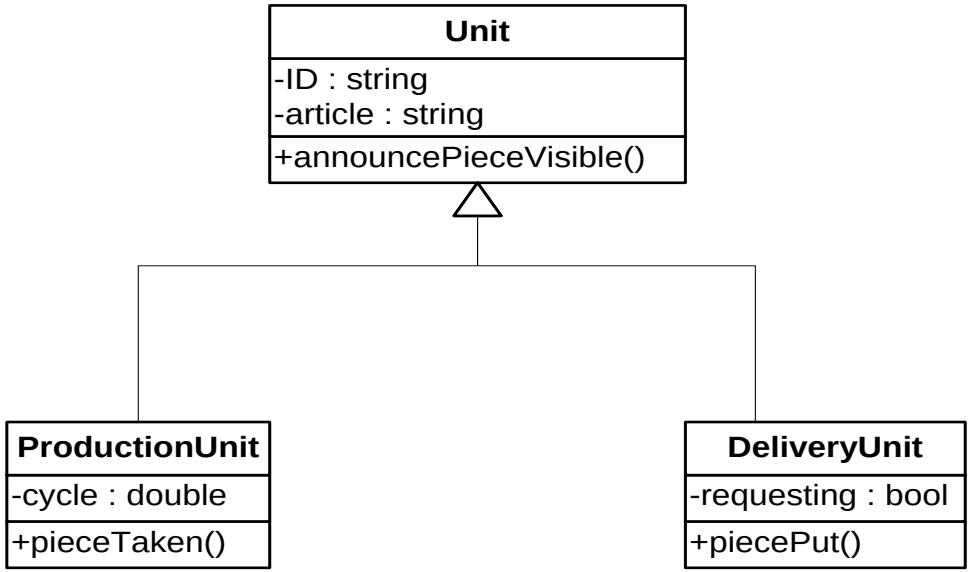
Inheritance models

- organize classes into a *hierarchy*
- classes at the top reflect common features
  - sometimes they are in fact *interfaces*
- classes inherit their attributes and services from one [or more] super-classes
- classes may be as specialised as necessary

- *class hierarchy design – a difficult process*

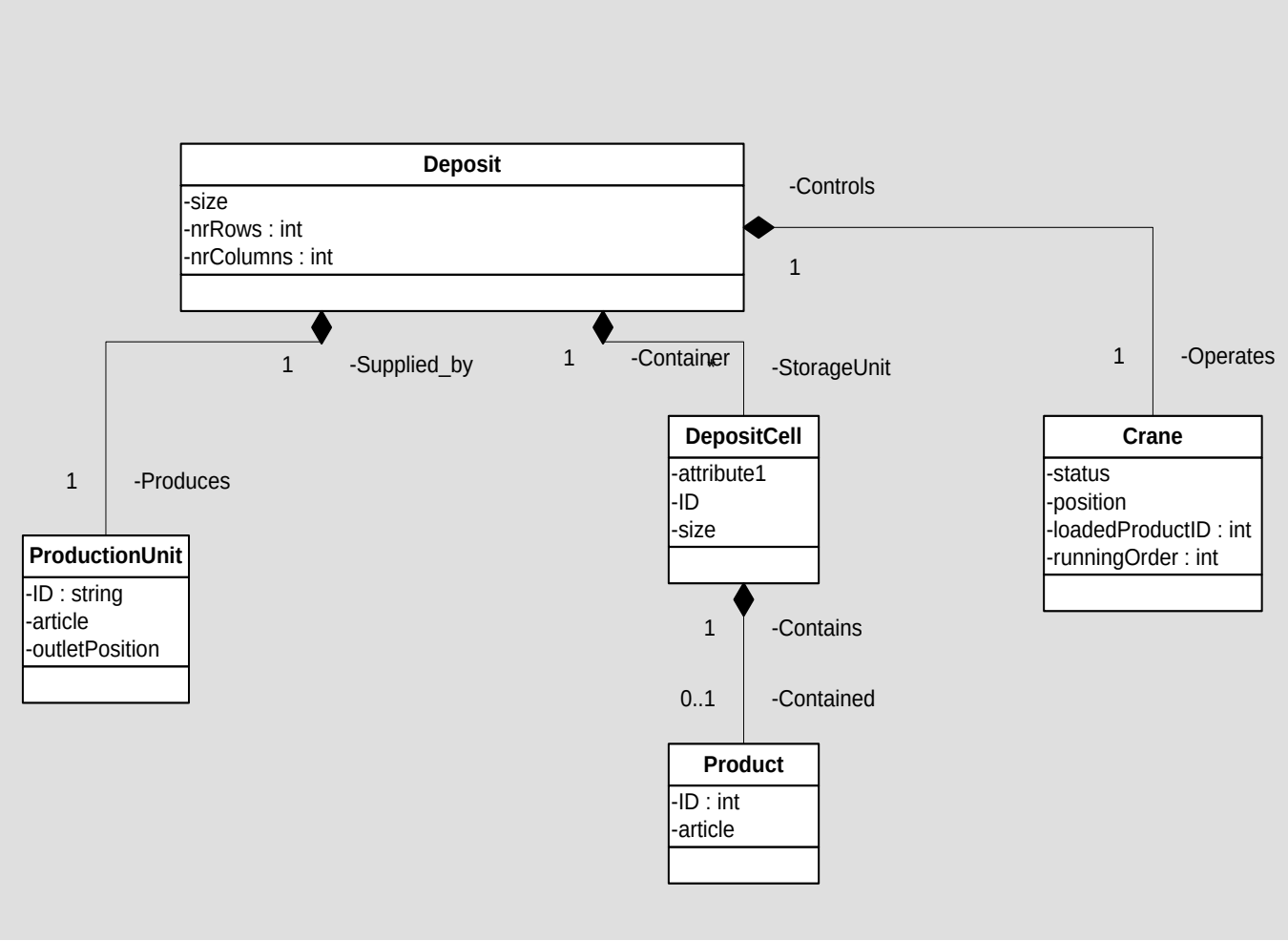
Inheritance model: example

From ADMSys:



Aggregation models

- show how classes are composed by other classes



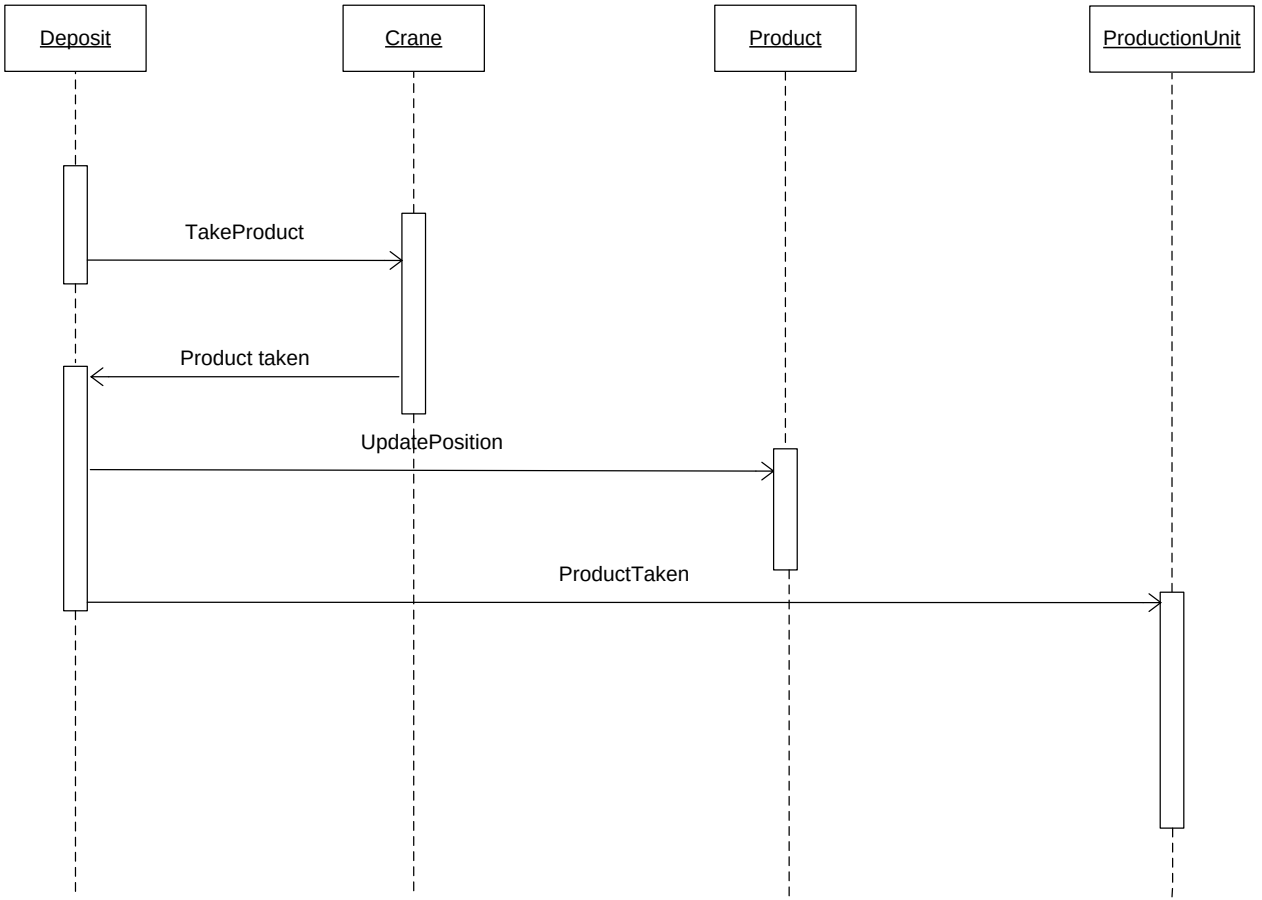
Object behaviour models

- show interactions between objects
- produce some particular system behaviour that is specified as a use-case

- sequence diagrams (or collaboration diagrams) in the UML are used to model interaction between objects

Object behaviour models: example

- a new product needs depositing…



Homework

Identify possible states in a state diagram for the simulation of a coffee machine with water tank.