Faculty of ECMS / School of Computer Science

Software Engineering & Project
System Modelling

System Modelling

Lecture 10

Chapter 8 (5 in Edition 9)
in the course text book

Objectives

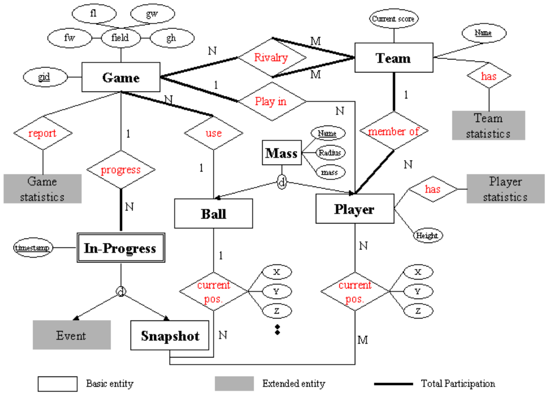
- To introduce several system models, which *describe system specification in a more technical way*
- To describe context modelling, behavioural modelling, data modelling and object modelling
- To introduce some of the notations used in the Unified Modeling Language (UML)

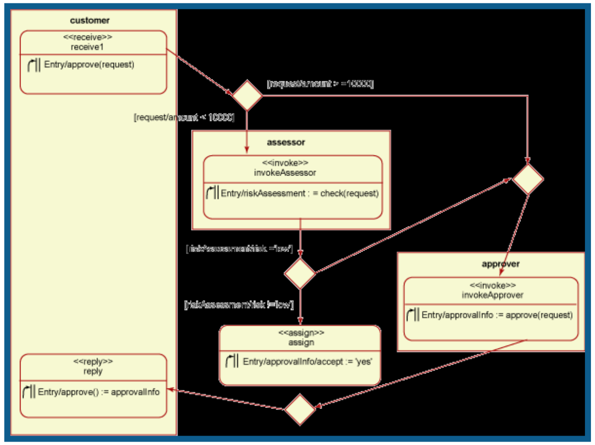
Outline

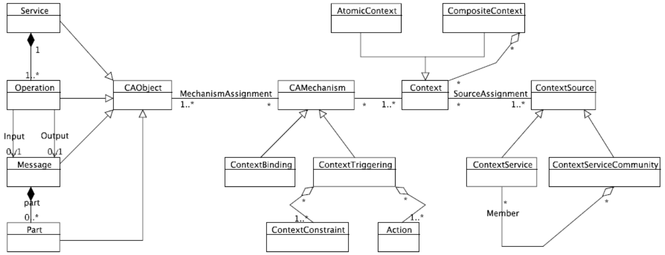
- System modelling: an overview
- Context models
- Behavioural models
- Data models
- Object models

System modelling

- Creating a set of abstract models of a system, with each model presenting a different view or perspective of the system
- Normally in graphical representations that are easy to understand
- System modelling helps the analyst to understand the functionality of the system and the models are used to communicate with stakeholders
- Engineers use these models to discuss design proposals and implementation




customer
<<receive>>
receive1
Entry/approve(request)
[request/amount > =10000]
[request/amount < 10000]
assessor
<<invoke>>
invokeAssessor
Entry/riskAssessment : = check(request)
[riskAssessment/risk ="low"]
[riskAssessment/risk !=low"]
approver
<<invoke>>
invokeApprover
Entry/approvalInfo := approve(request)
<<assign>>
assign
Entry/approvalInfo/accept := 'yes'
<<reply>>
reply
Entry/approve() := approvalInfo



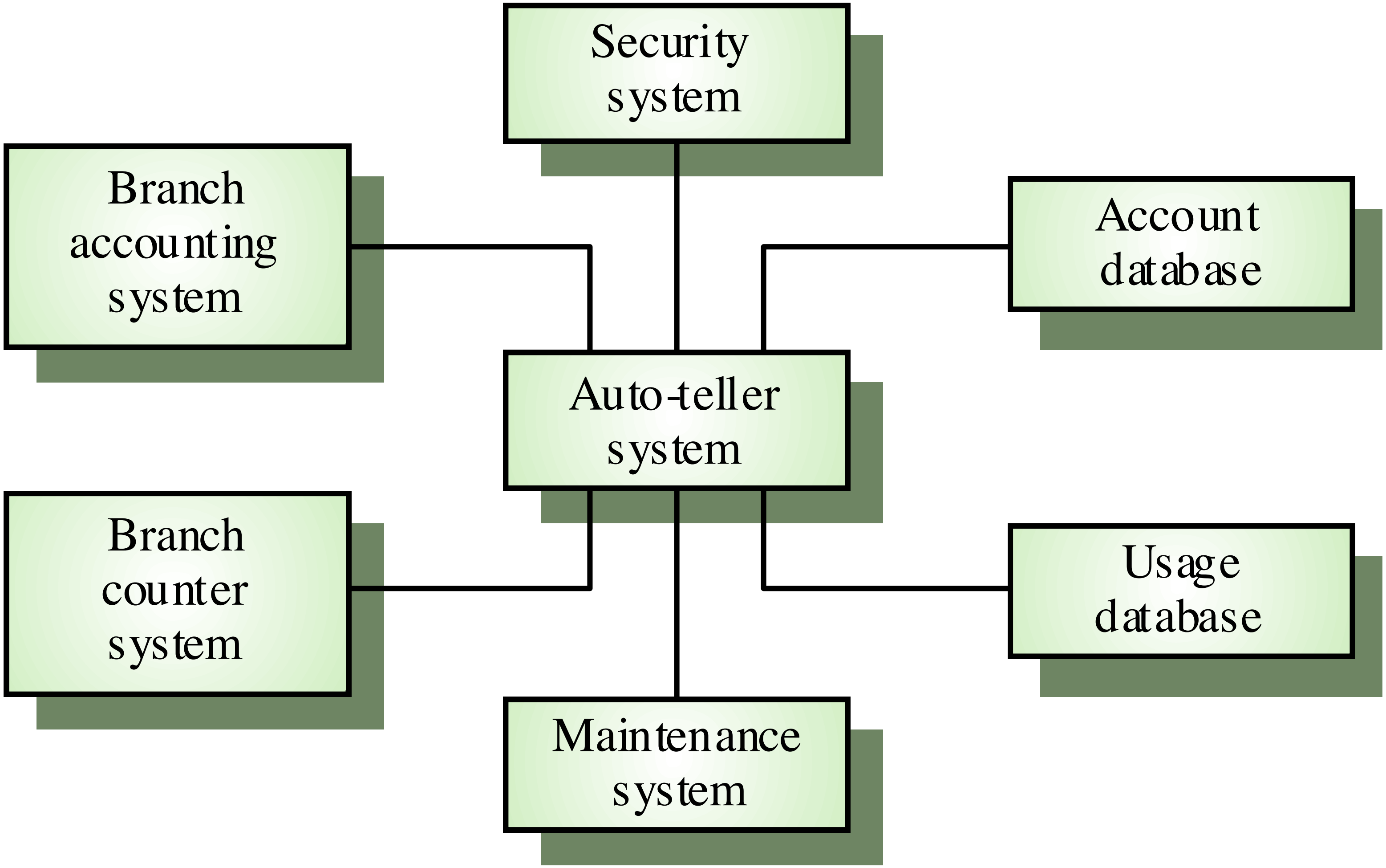
Model types

- We can develop different models to present the system from different perspectives
  - *External perspective* showing the system's context or environment (Context Model)
  - *Behavioural perspective* showing the behaviour of the system (Behavioural Model)
  - *Structural perspective* showing the system architecture (Architectural Model)

Context models

- Context models are used to illustrate the boundaries of a system (i.e., what is in the system, what is not)
- In some cases, the boundary is clear
  - Replace an existing system
- In other cases, the boundary is unclear
  - Decided during the requirement engineering process
- Social and organisational concerns may affect the decision on where to position system boundaries (political judgement)
- Architectural models show the system and its relationship with other systems

Context Models Example: ATM System

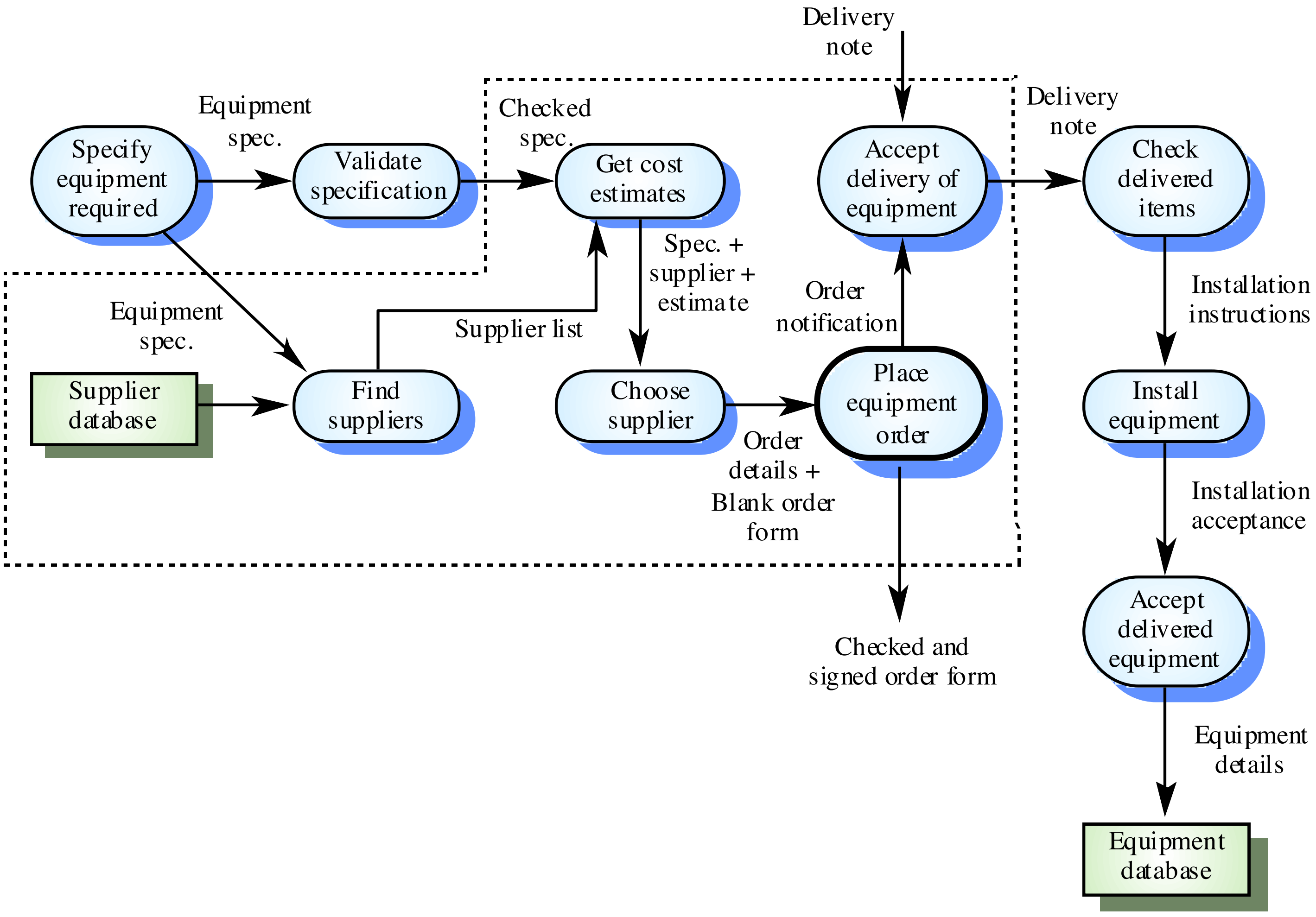


Context Models (Cont.)

- Architectural models are normally supplemented by other models

- ***Process models*** show the overall process and the processes that are supported by the system

Computer-supported equipment procurement process



Behavioural models

- Behavioural models are used to describe the overall behaviour of a system

- Two types of behavioural model are shown here
  - *Data-flow models* that show how data is processed as it moves through the system
  - *State machine models* that show the systems response to events (control-flow)

- These models show different perspectives and may be used separately or together, depending on the type of the system to be developed.

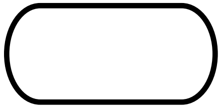
Data-flow models

- Data flow diagrams are used to model the system's data processing
- These show the processing steps as data flows through a system
- Intrinsic part of many analysis methods
- Simple and intuitive notation that customers can understand
- Show end-to-end processing of data

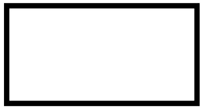
Data Flow Diagrams

- Also called **DFD**s, *data flow graph* or *bubble chart*.
- Widely used semiformal graphical notation.
- DFDs model the system from a functional perspective
- Depicts
  - Information flow and organization
  - Transformations applied to data from input to output

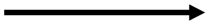
DFD - Notational Elements



functional processing



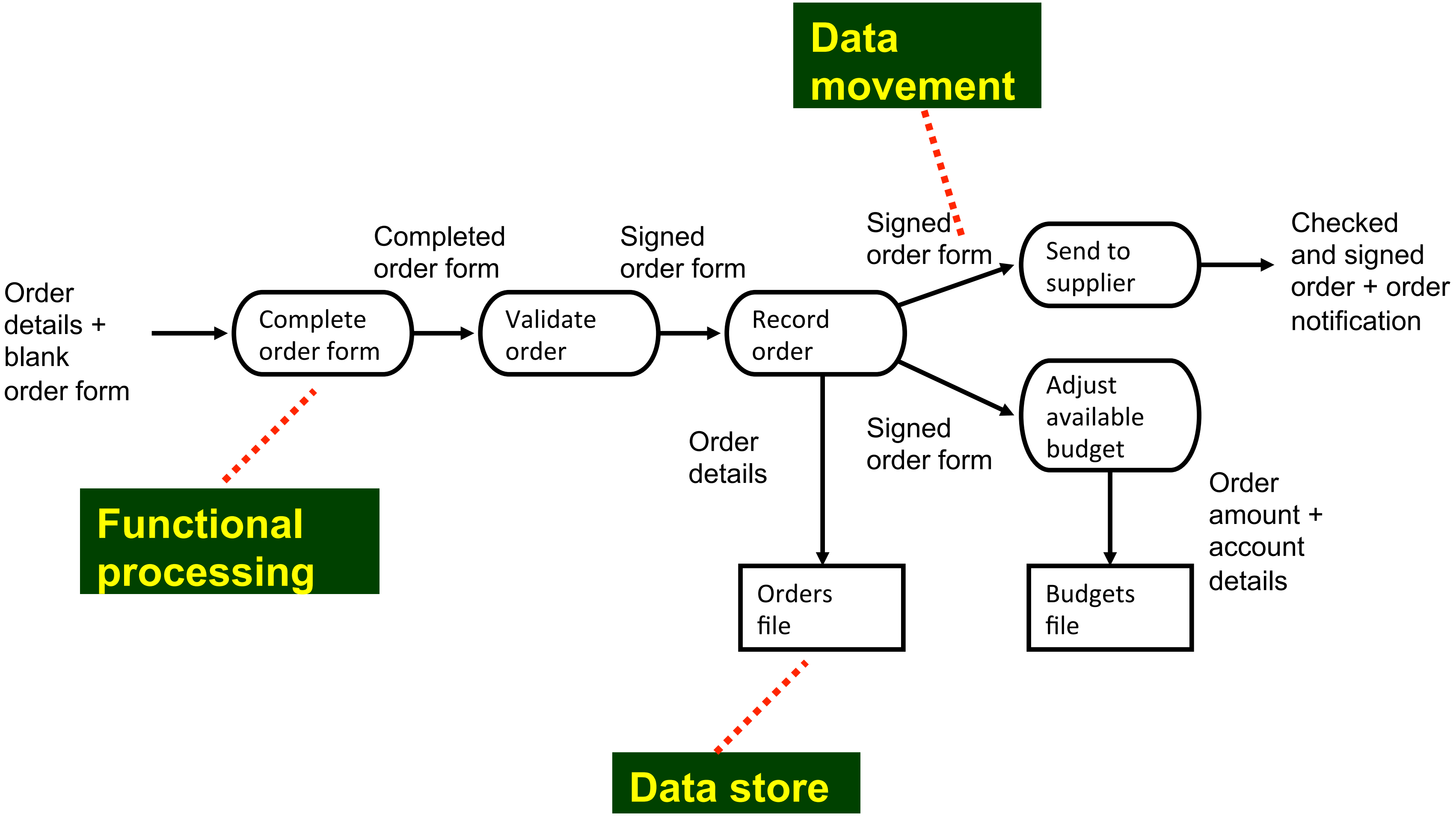
data store



data movement

- Represent transformation of input data to output data.
- Transformation annotated with descriptive name.
- Represent a data store.
- Annotation with descriptive name.
- Show direction of data flow.
- Annotated with name of data that flows along path.

DFD Example: Order Processing



State Machine Models

- Model the behaviour of the system in response to external and internal events

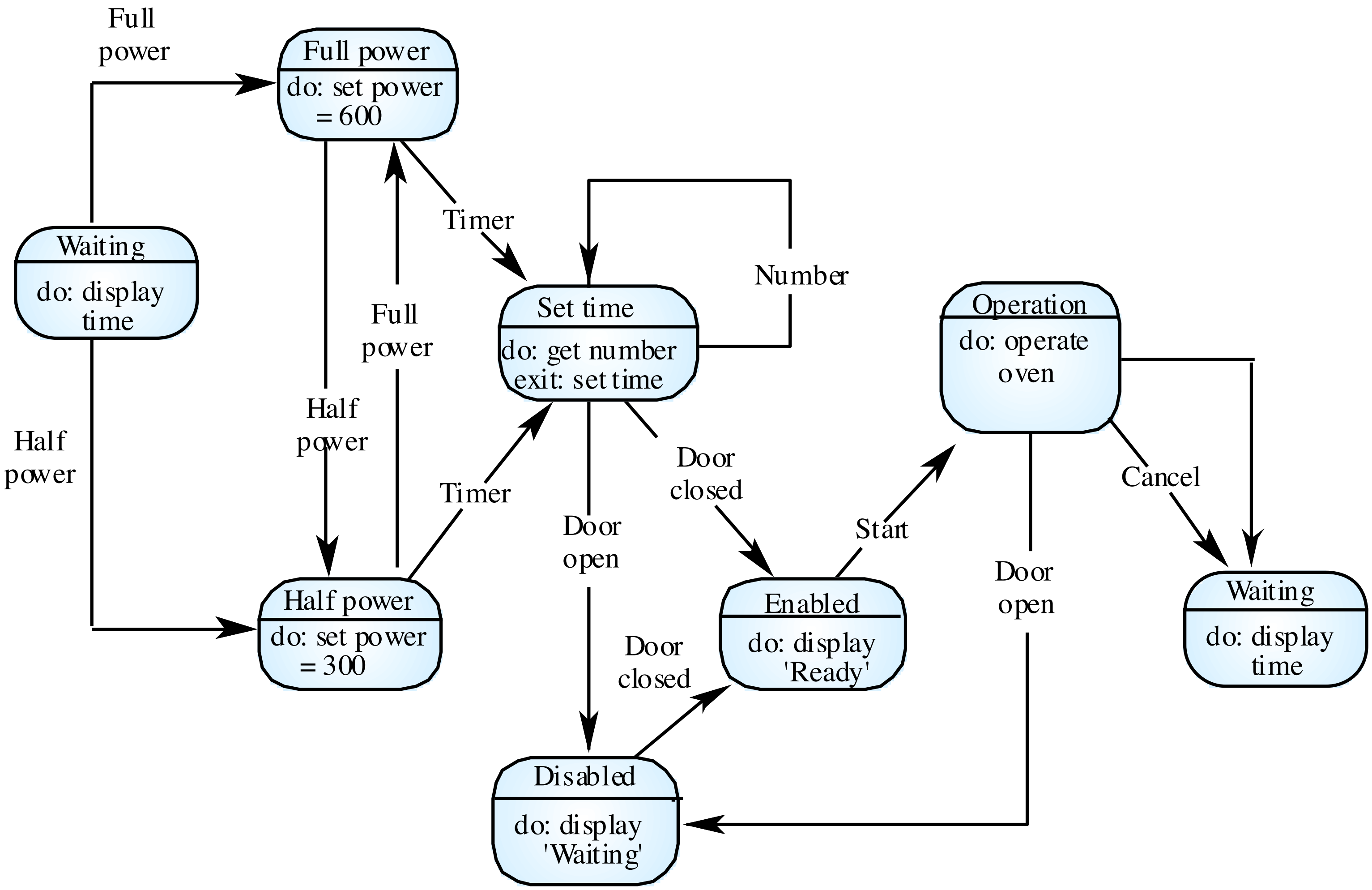
- They show the system’s responses to stimuli so are often used for modelling real-time systems

- State machine models show system states as nodes and events as arcs between these nodes. Assume that at any time, the system is in one of a number of possible states. When an event occurs, the system moves from one state to another (transition)

Statecharts

- Statecharts are an integral part of the UML and are used to represent state machine models.
- Allow the decomposition of a model into sub-models (see following slide)
- *Rounded rectangles* represent system states. A brief description of the actions is included following the ‘do’ in each state; *labelled arrows* represent stimuli (events)
- Can be complemented by tables describing the states and the stimuli.

Statecharts Example: Microwave Oven



Statecharts Example: Microwave Oven (Cont.)

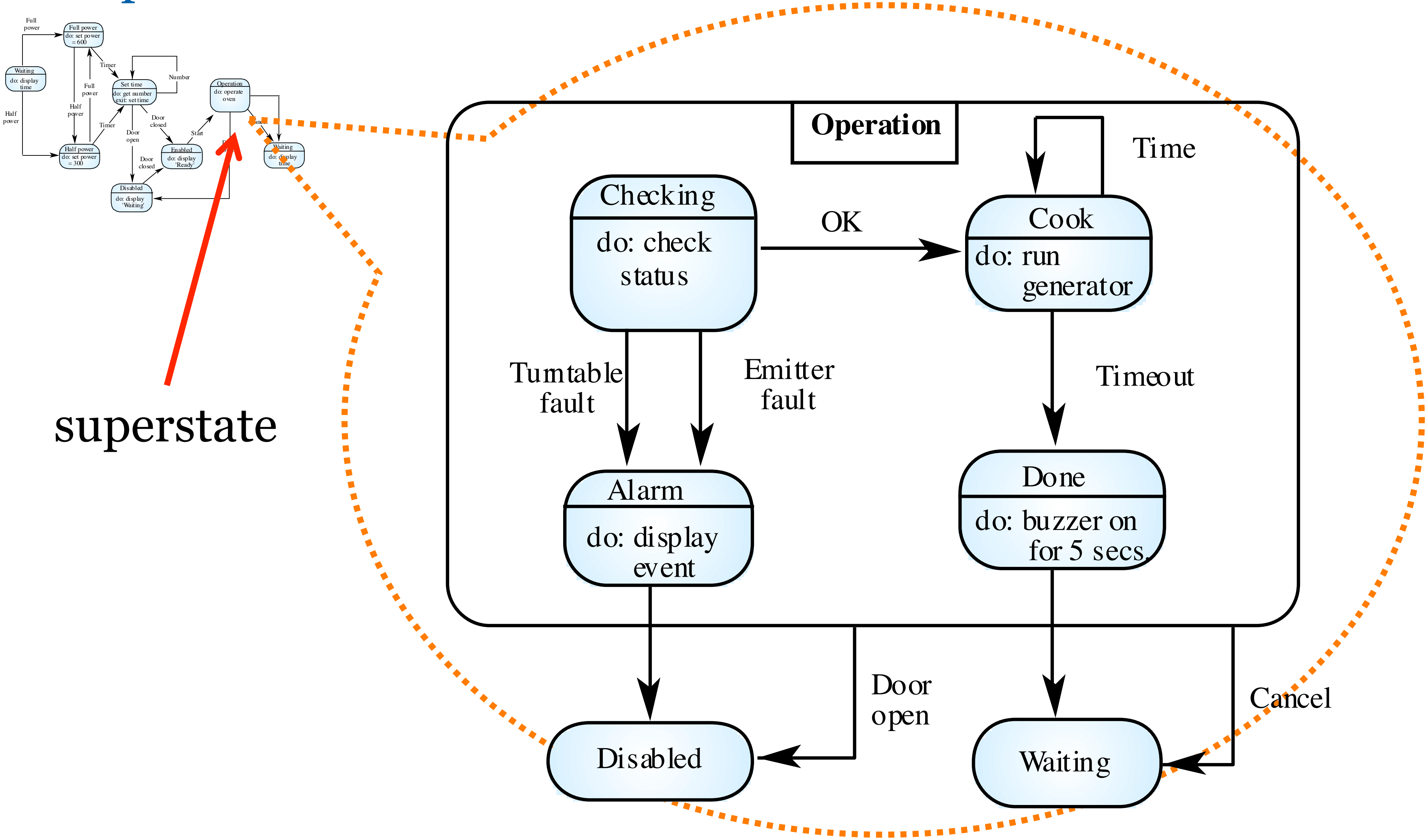
State	Description
Waiting	The oven is waiting for input. The display shows the current time.
Half power	The oven power is set to 300 watts. The display shows ’Half power‘.
Full power	The oven power is set to 600 watts. The display shows ’Full power‘.
Set time	The cooking time is set to the user‘s input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

Statecharts Example: Microwave Oven (Cont.)

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

Statecharts Example: Microwave Oven

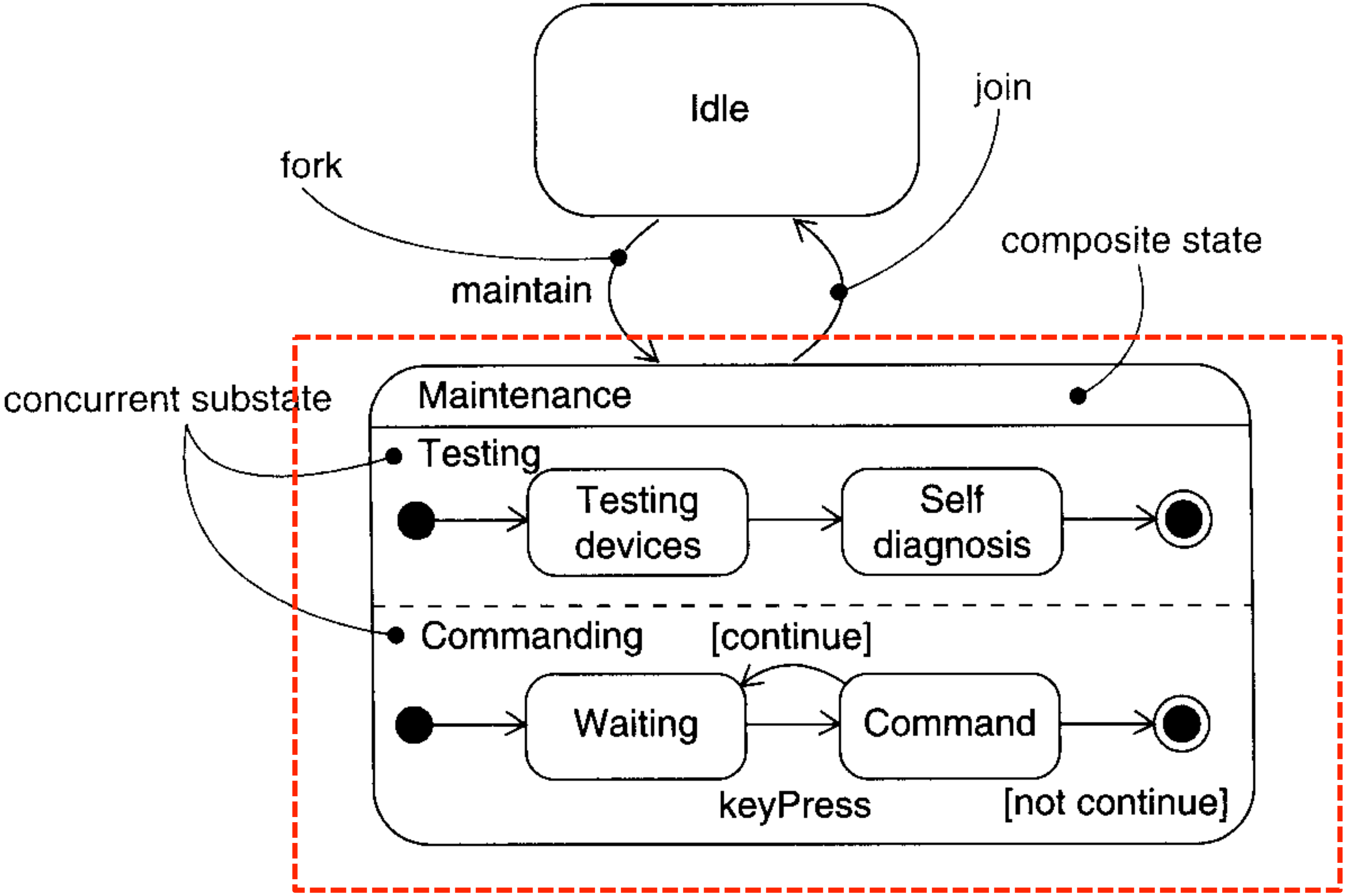
Superstate



superstate

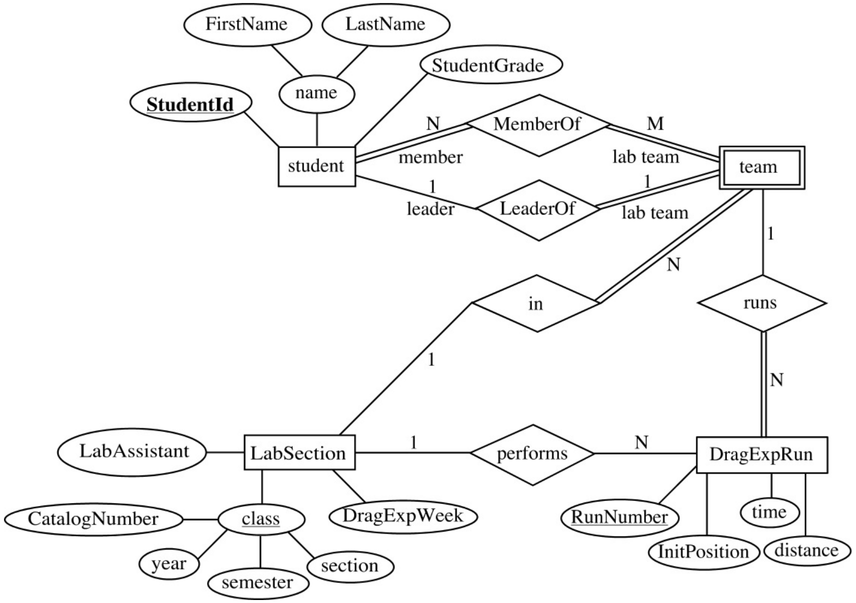
Statecharts Example: Microwave Oven

Composite State



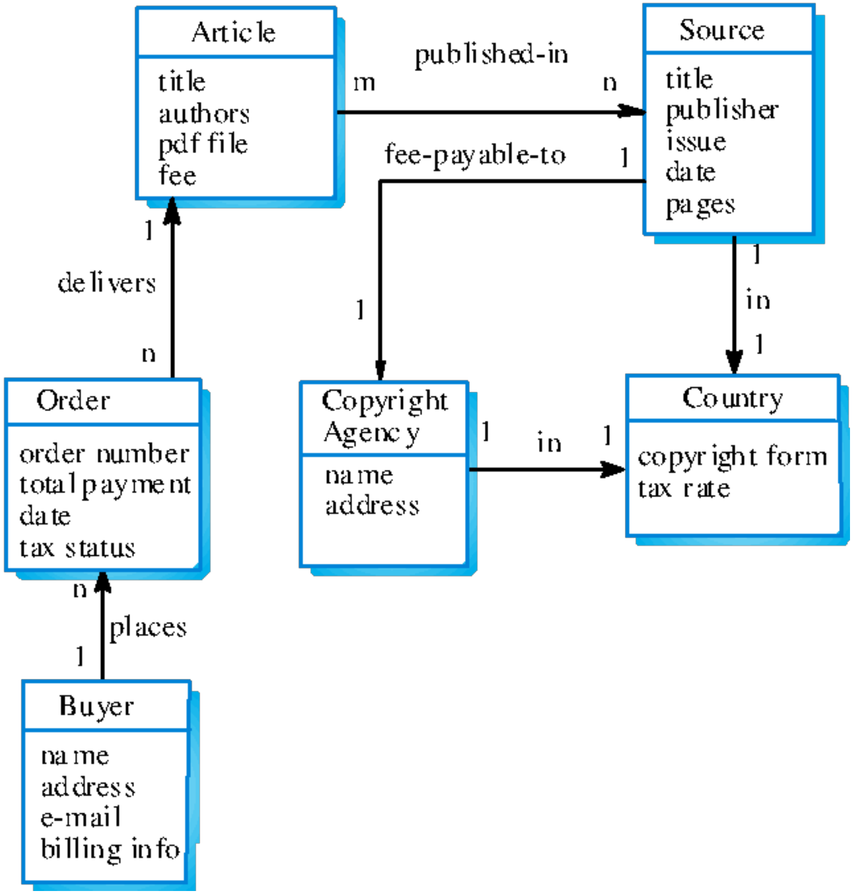
Semantic data models

- Used to describe the logical structure of data processed by the system
- An entity-relation-attribute (**ERA**) model (Peter Chen, 1976) sets out the entities in the system, the relationships between these entities and the entity attributes
- Widely used in database design. Can readily be implemented using relational databases



Data models

- No specific notation provided in the **UML** but objects and associations can be used
- Entities in ER model as simplified object classes (no operations)
- Attributes as class attributes
- Relations as named associations between classes



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
  - Contain more detailed information;
  - Support name management; avoid duplication;
  - Store of organisational knowledge; link analysis, design, implementation, and evolution in one place
- Many CASE workbenches support data dictionaries

Data Dictionaries Example

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 copy 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 models

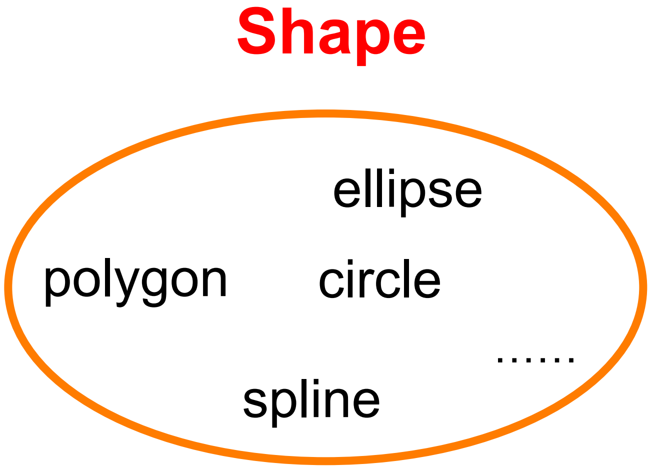
- Object models describe the system in terms of object classes and their associations (in an object-oriented approach)
- An object class is an abstraction over a set of objects with common attributes and the services (operations) provided by each object
- Natural ways of reflecting the real-world entities manipulated by the system (e.g., cars, airplanes, tables, with *clearly identifiable attributes*)
- More abstract or complex entities are more difficult to model using this approach (e.g., the concept of a library system)

Object models (Cont.)

- Various object models may be produced
  - Inheritance models
  - Aggregation models
  - Interaction models

Inheritance models

- 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



Shape

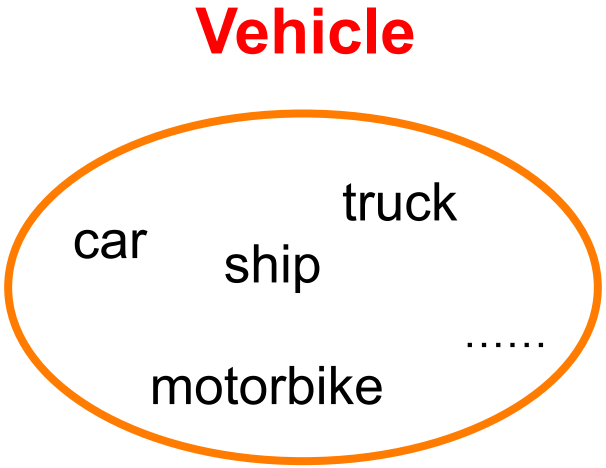
ellipse

polygon

circle

……

spline



Vehicle

truck

car

ship

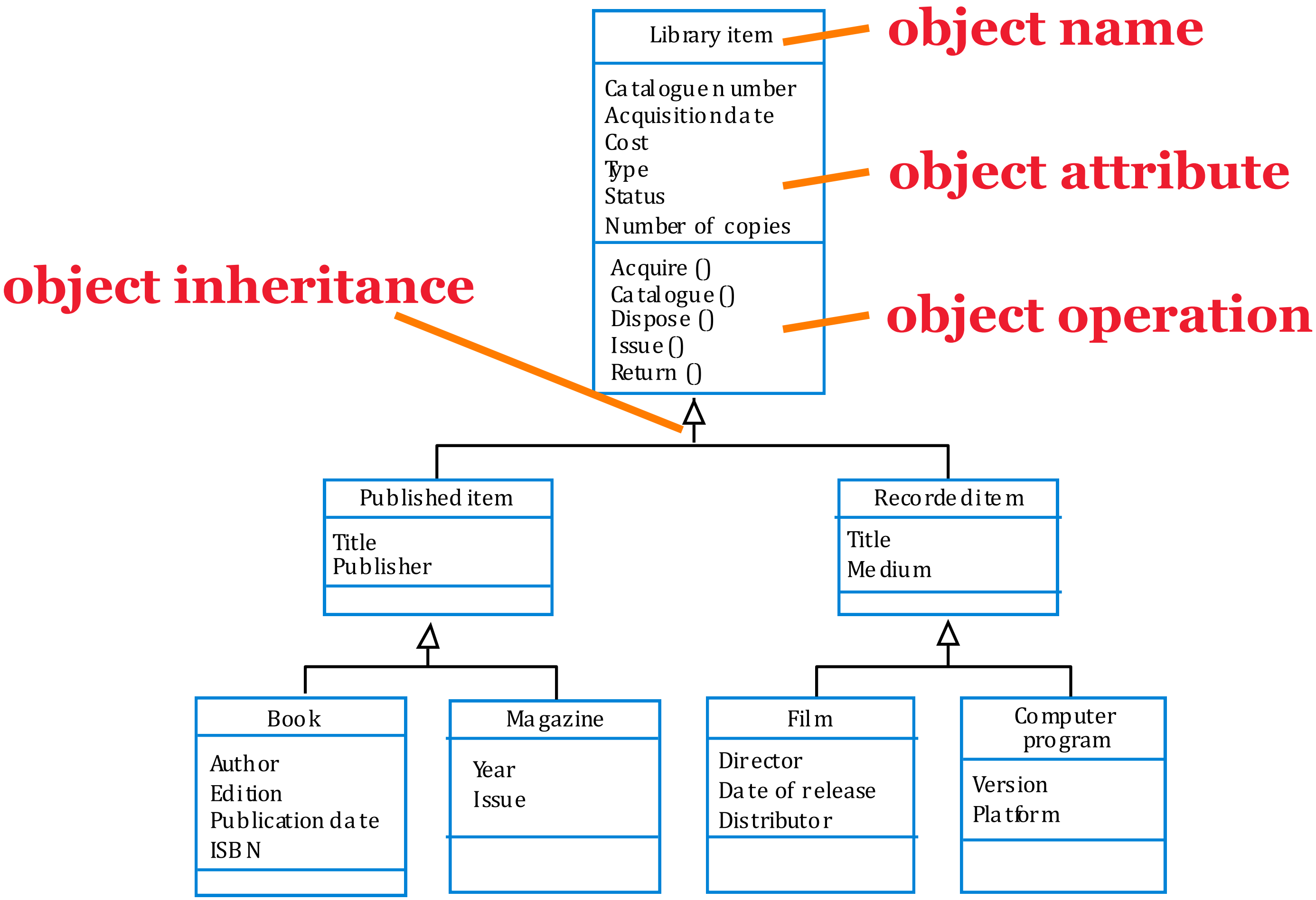
……

motorbike

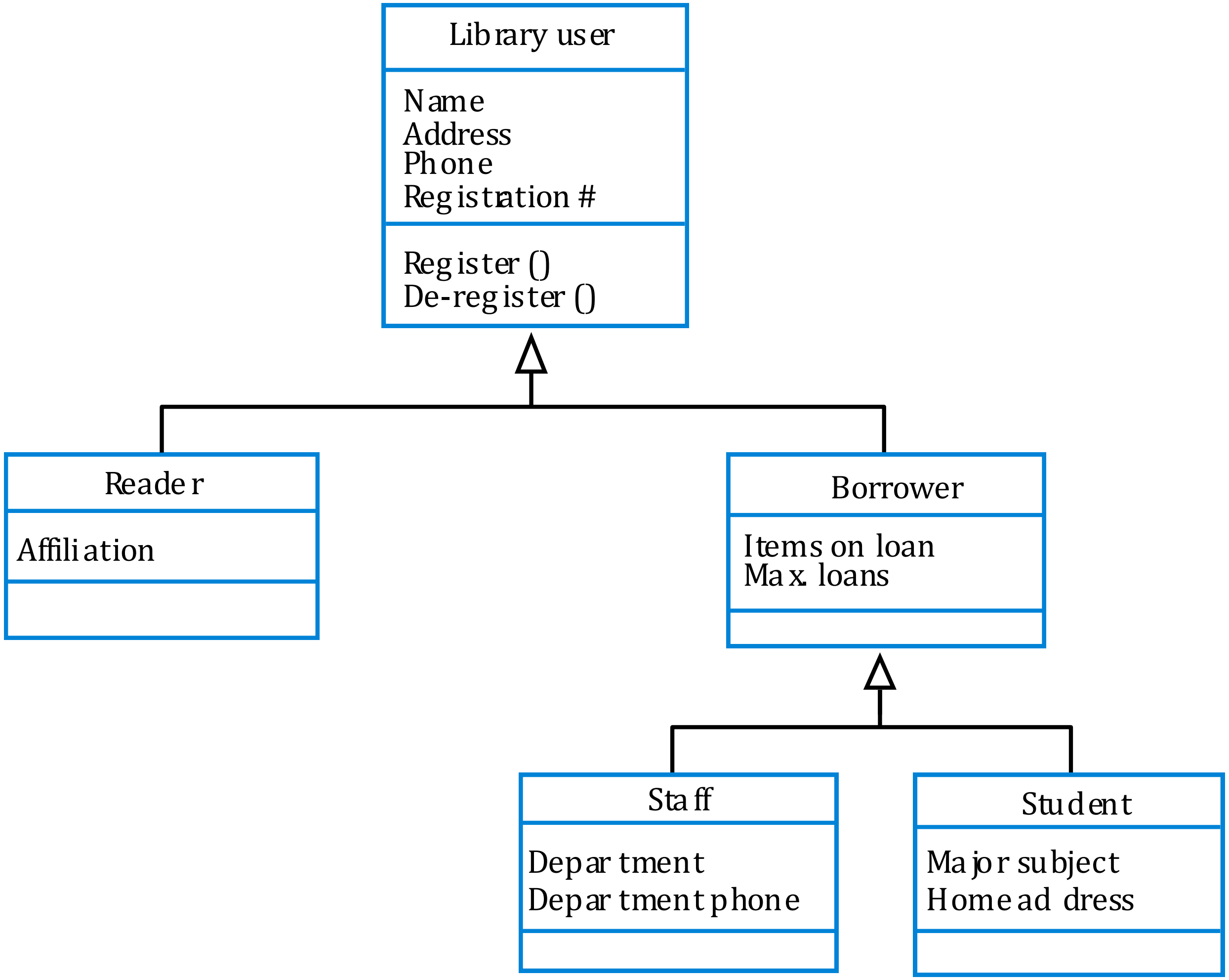
Object models and the UML

- 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** is referred to as generalisation and is shown 'upwards' rather than 'downwards' in a hierarchy.

UML Example: Library item class hierarchy

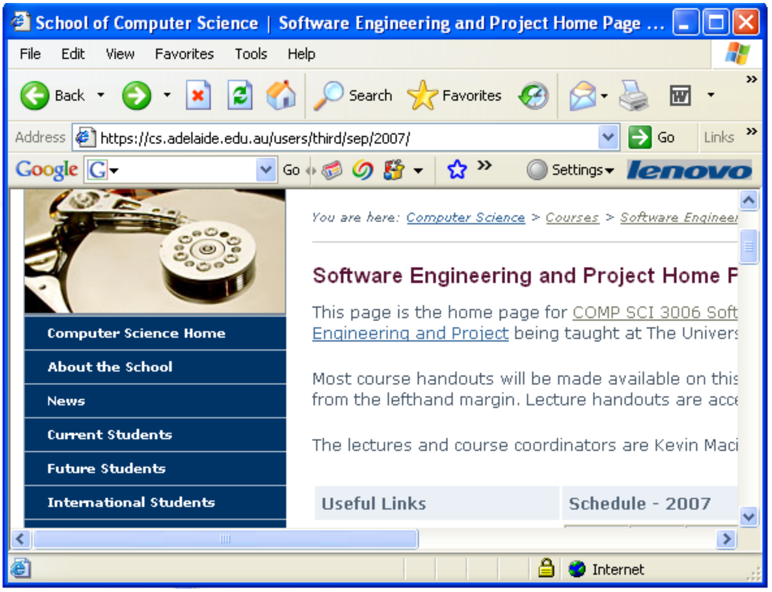


UML Example: Library user class hierarchy



Object aggregation

- Aggregation model shows how classes are composed of other classes
- An object is an aggregate of a set of other objects
  - A table is composed of a surface and three or more legs
  - A window is composed of 2 sliders, 1 header, and 1 panel
- Similar to the part-of relationship in semantic data models



You are here: Computer Science > Courses > Software Enginee

Software Engineering and Project Home P

This page is the home page for COMP SCI 3006 Soft Engineering and Project being taught at The Univers

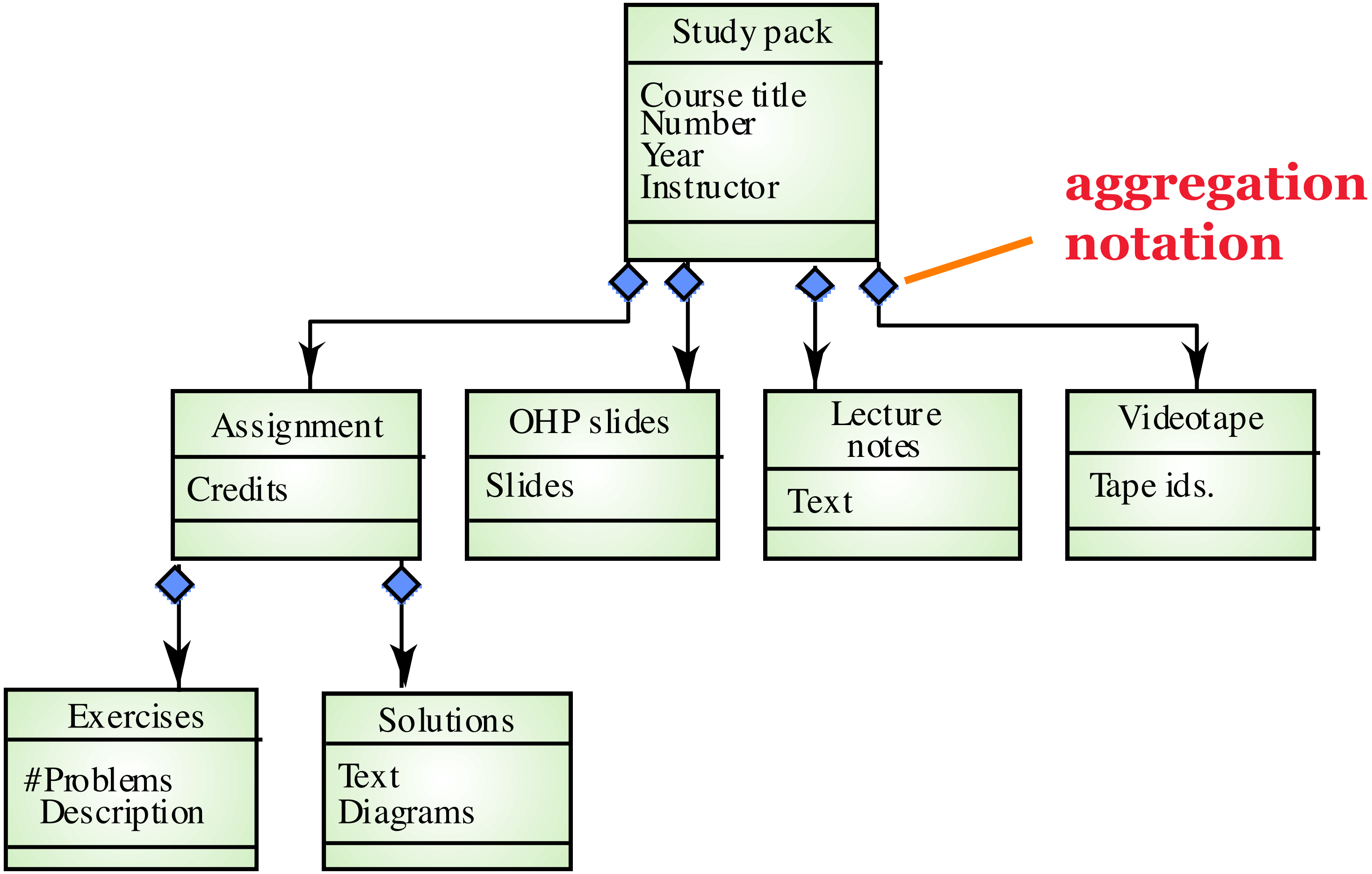
Most course handouts will be made available on this from the lefthand margin. Lecture handouts are acc

The lectures and course coordinators are Kevin Mac

Useful Links

Schedule - 2007

Object aggregation Example: Study Pack

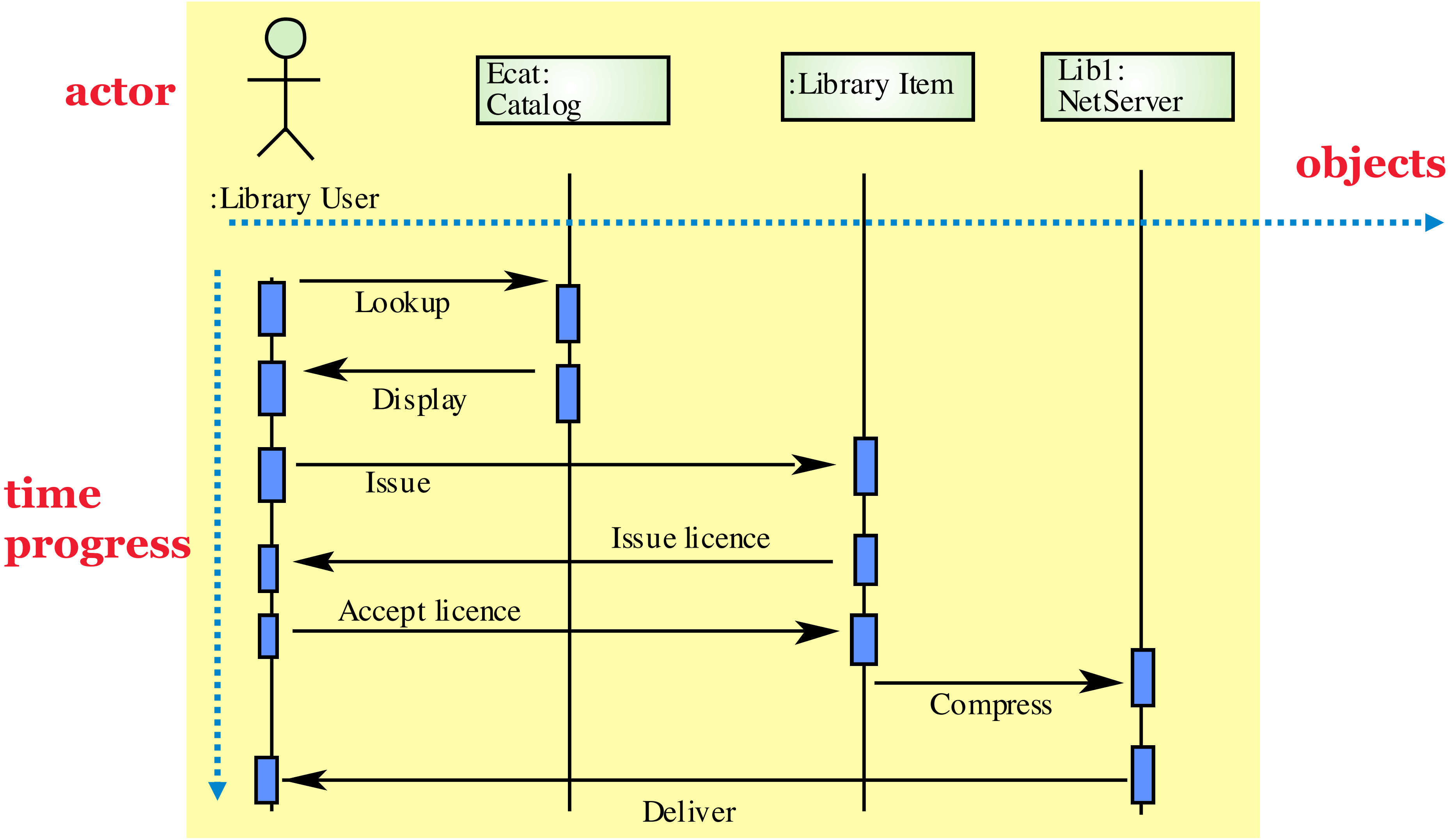


Object behaviour modelling

- A behavioural model shows the **interactions** between objects to 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

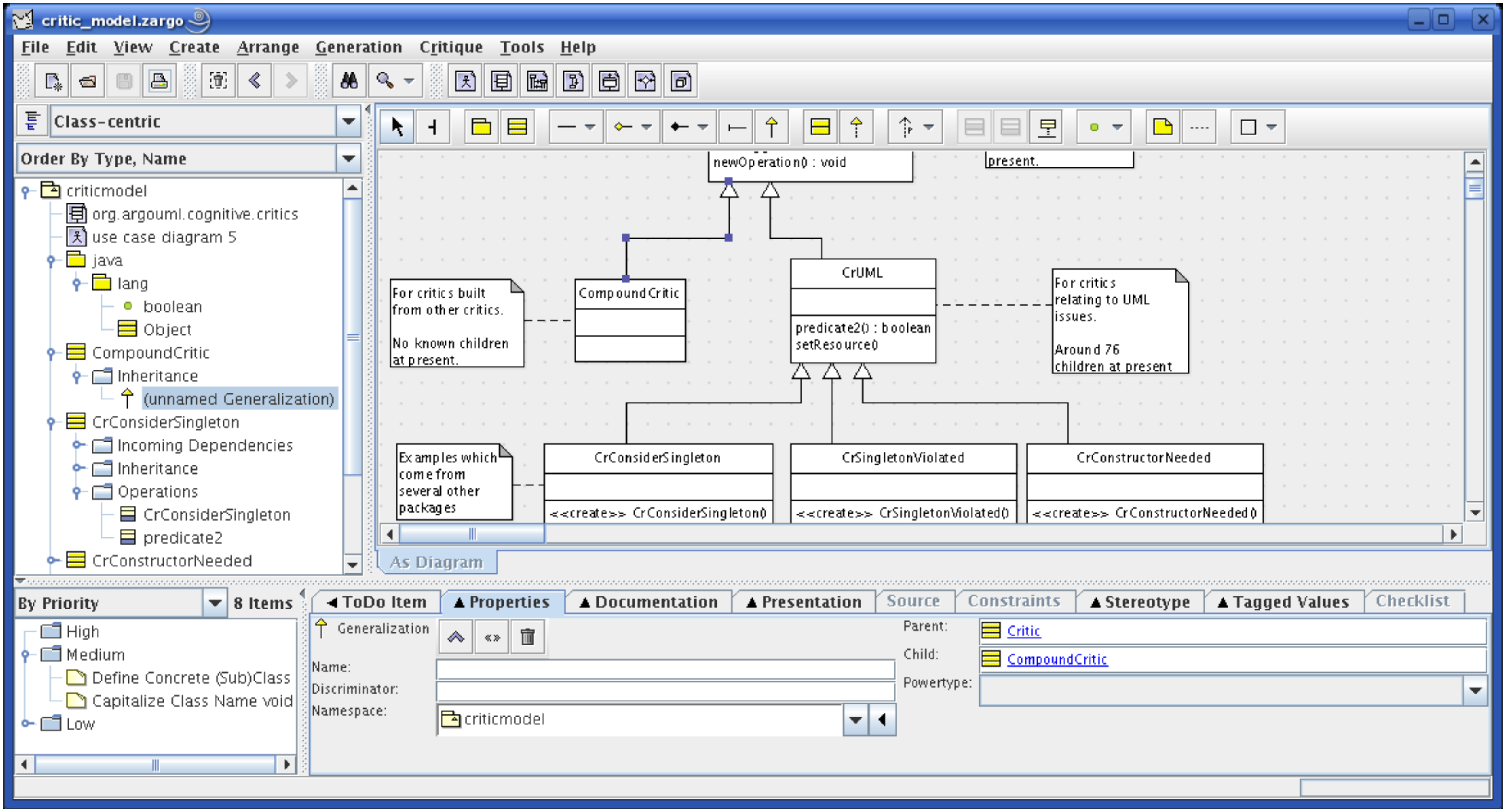
Object behaviour modelling Example: Issue of electronic items



Document your design in a professional way

- Use **standard modeling language** like UML
- **Follow strictly the notations**. If not sure, consult UML documentation, e.g.,
  - UML Notation Guide, available from course site (UML-notation.pdf)
  - *UML official web site, http://www.uml.org/*
- UML Tools
  - http://en.wikipedia.org/wiki/List_of_Unified_Modeling_Language_tools

UML Modeling Tools: ArgoUML



For critics built from other critics.

No known children at present.

For critics relating to UML issues.

Around 76 children at present

Examples which come from several other packages

Name:

Discriminator:

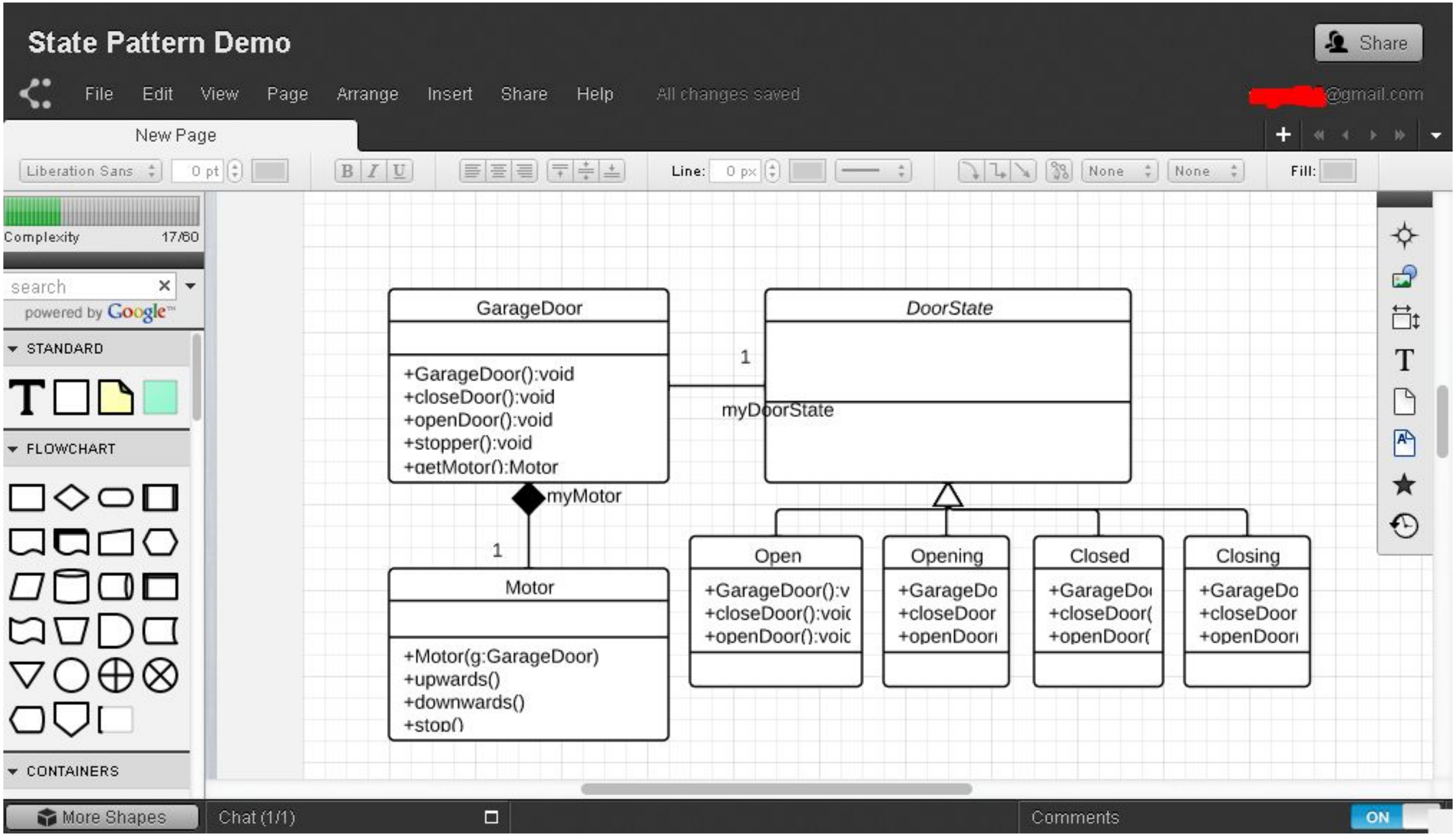
Namespace: criticmodel

Parent: Critic

Child: CompoundCritic

Powertype:

UML Modeling Tools: LucidChart



Key points revisit

- A model is an ***abstract*** system view. Complementary types of model 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 revisit

- Semantic data models describe the logical structure of data which is imported to or exported by the systems

- Object models describe logical system entities, their classification and aggregation

- A high-level design includes of a set of complementary models at various levels of abstraction

- A detailed design will typically refine the high-level design