CS4125
SYSTEMS ANALYSIS
SPRING SEMESTER 2010-2011

J.J. Collins
Dept of CSIS
University of Limerick

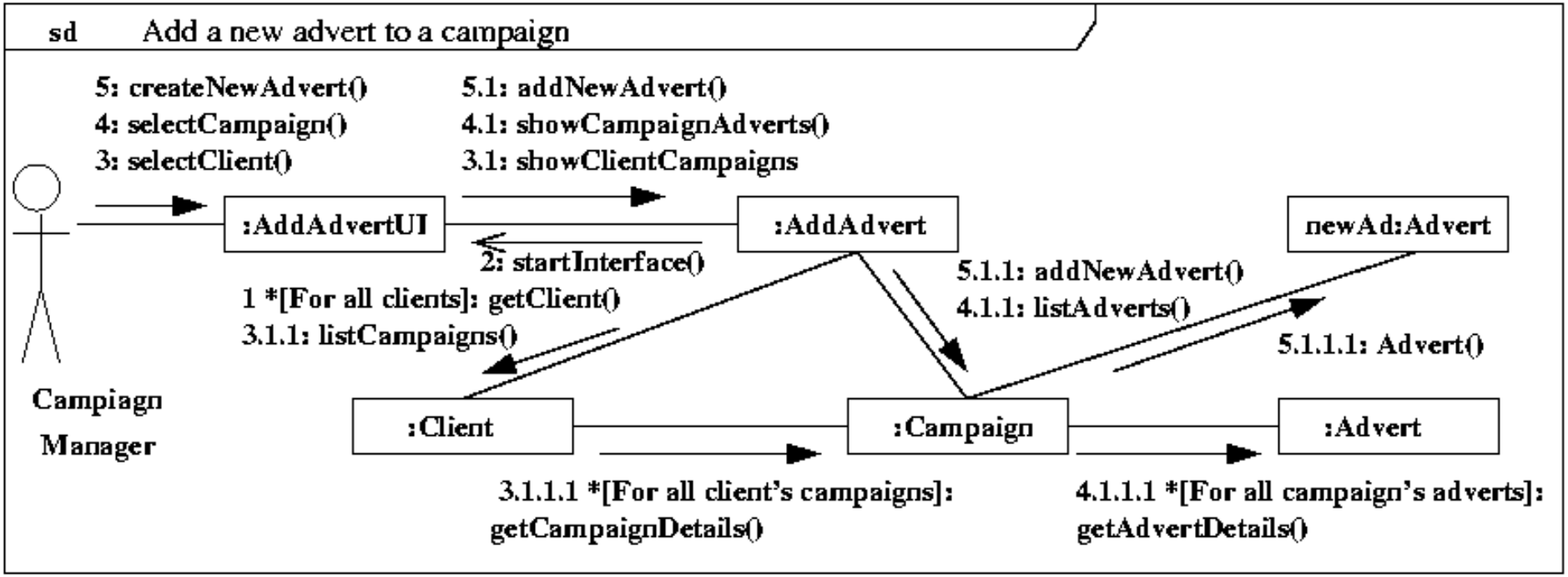
Lecture K (W7L1)

Analysis: Interaction Modelling – Communication Diagrams

Communication Diagrams

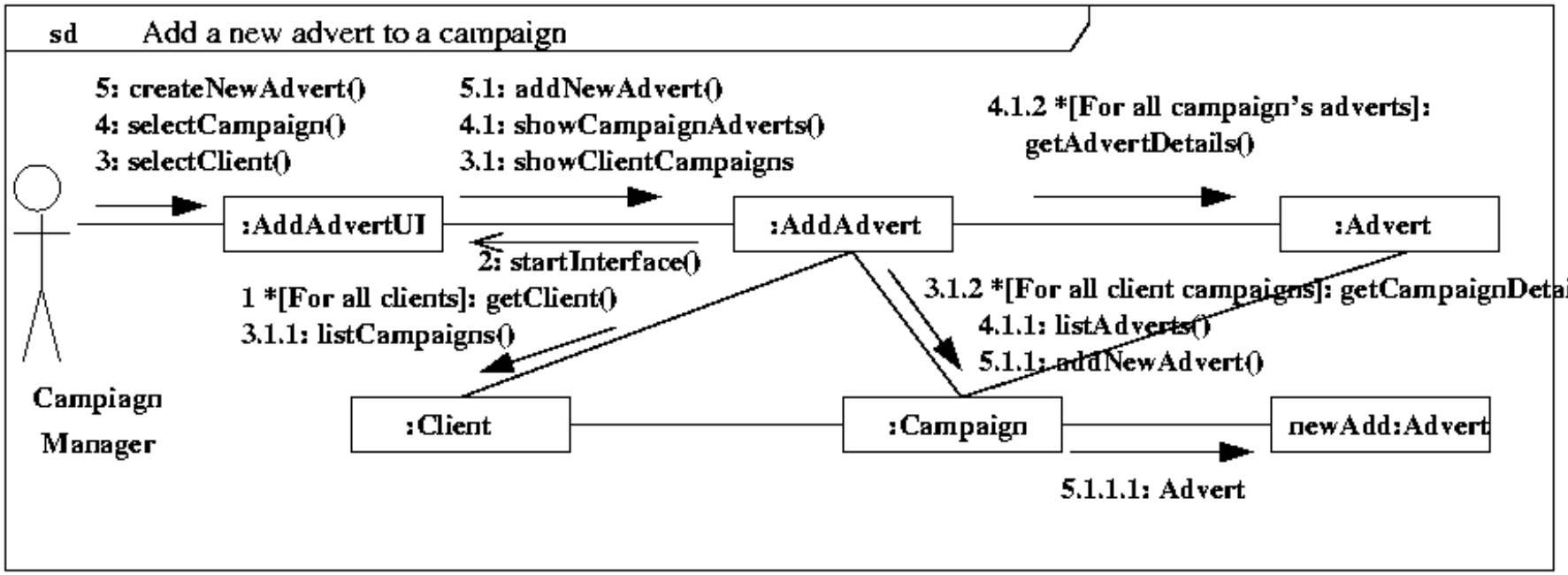
- Provide an alternative notation for interactions that also shows links between the collaborating objects.
- UML interactions can range from the performance of a single operation to the articulation of a whole use case.
- Can be used for automatic generation of sequence diagrams.
- Explicitly shows associations or links between the different classes or object roles.
- No explicit time dimension.
- Interaction is drawn on what is essentially a fragment of a class or object diagram.

Example



Communication diagram for the use case *Add a New Advert to a Campaign*.

Example



An alternative communication diagram for the use case *Add a New Advert to a Campaign*.

- Sequence numbers represents the order in which messages are exchanged.
- Some of the sequence numbers are written in nested style e.g. 2.1 and 3.1, to indicate iterative loops within the interaction that is being modelled.
- A similar style of numbering is used to indicate branching.

- What is the relationship between the following ???
- 4: m1()
- 4.1:m2()
- 4.1.1: m3()

Message Format

- The formal message label syntax is:

  sequence-expression {attribute=} signal-or-operation-name
  ['('argument-list')'] [<return-value>]

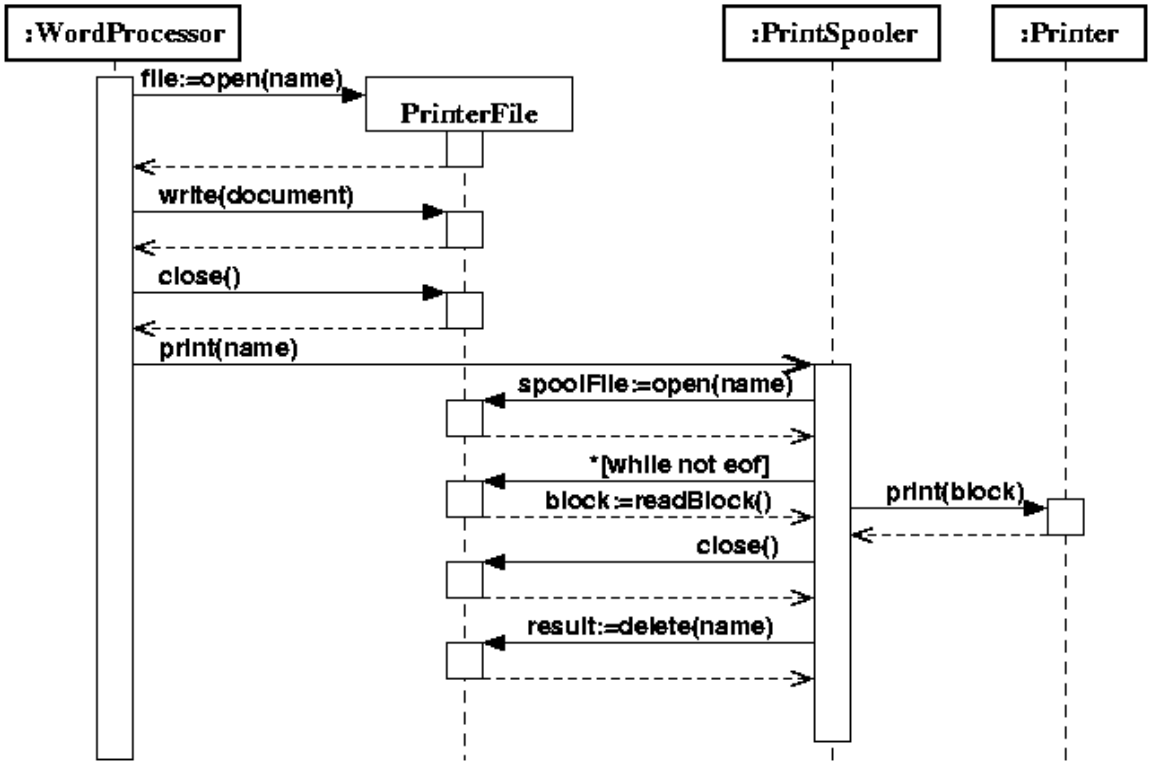
- A sequence expression is a list of integers separated by dots (`.'), and terminated by a colon.
- The message with the immediately preceding sequence number is assumed to be the predecessor by default.
- Recurrence reflects either iterative or conditional execution and is of the syntactical form:
  - Branching: `[' condition-clause `]'
  - Iteration: `*' `[' iteration-clause `]'
- Guard conditions are written in OCL. A guard condition may be used to represent the synchronisation of different threads of control.

Message Types

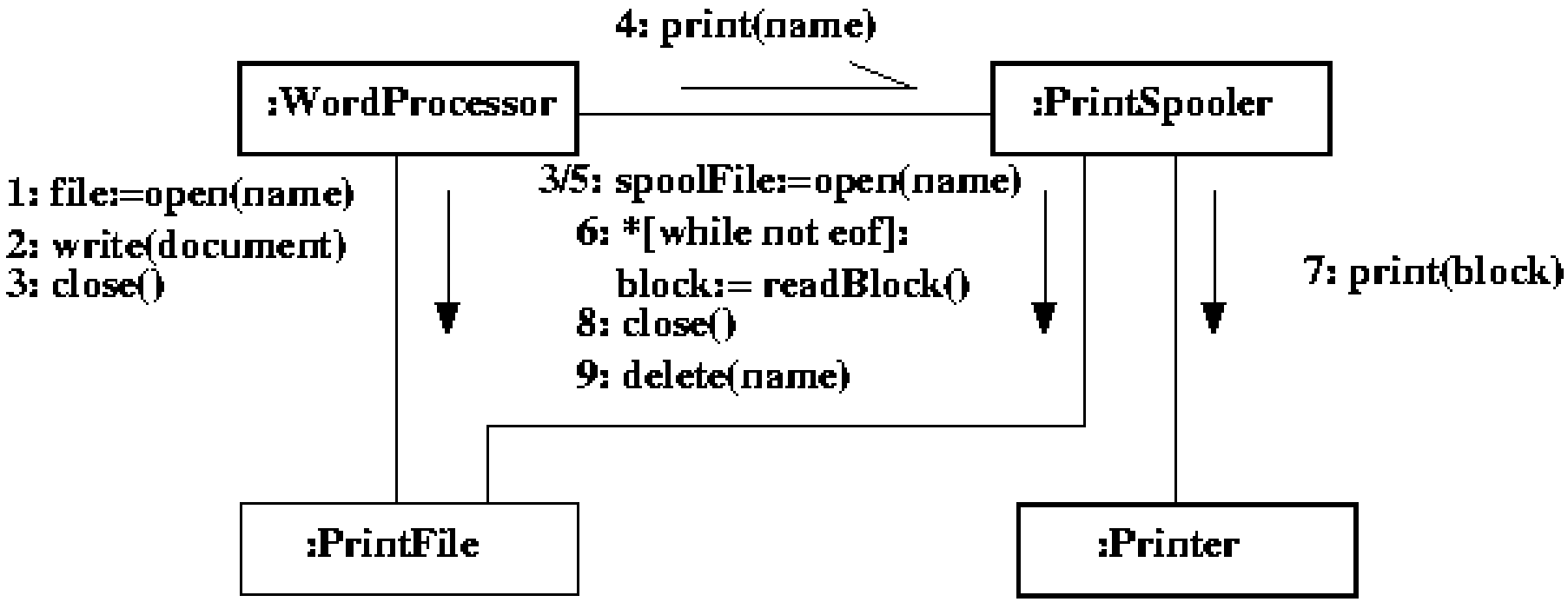
Types of Messages	Syntax Example
Simple message	4 : addNewAdvert()
Nested call with return value The return value is placed in the variable name	3.1.2: name := getName()
Conditional message This message is only sent if the condition [balance > 0] is true	[balance > 0] 5: debit(amount)
Synchronisation with other threads Message 4: playvideo() is invoked only once the two concurrent messages 3.1a and 3.1b are completed	3.1a, 3.1b/ 4: playVideo()

Active Objects

- Active Objects: object that owns a concurrent thread of control. May be it is an independent object running on its own processor, i.e. database running on a server, or running in a separate thread or process on a uniprocessor machine.
- UML 1.x

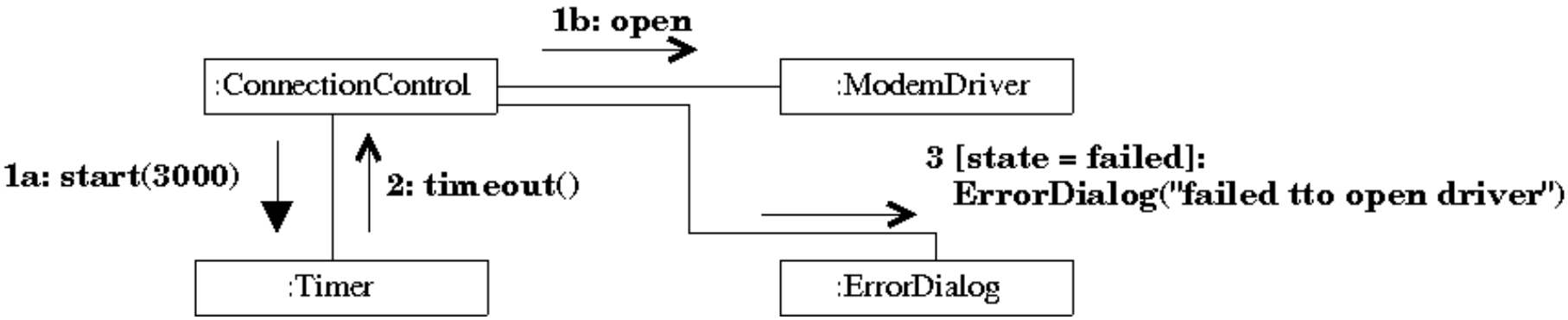


Active Objects



Active Objects shown using UML 1.x notation.

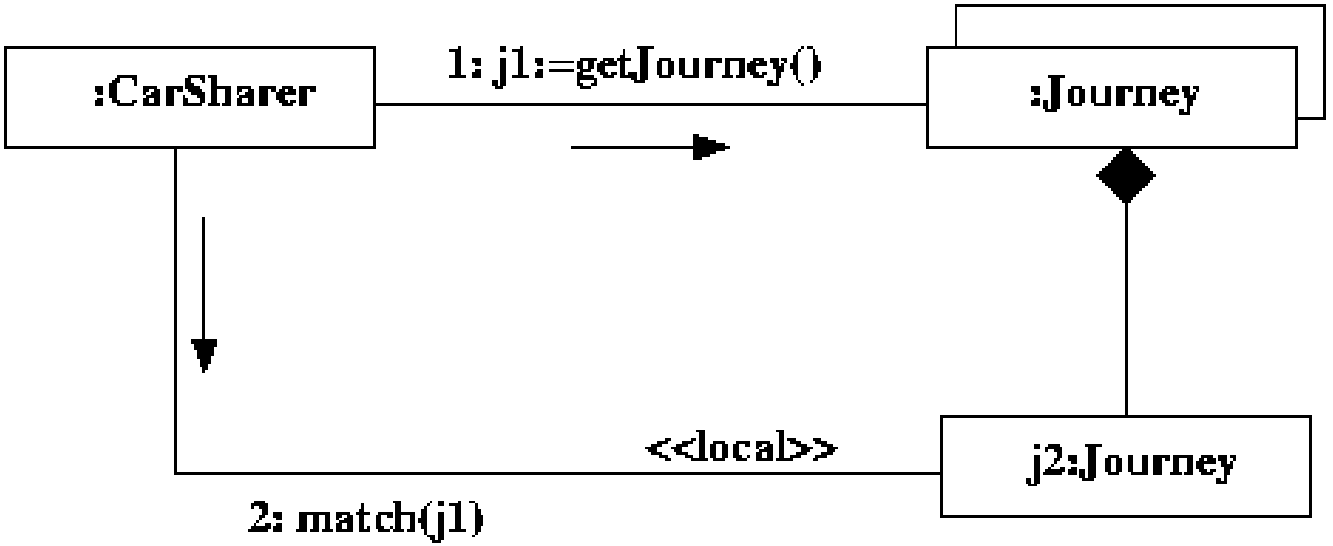
Another example



Adapted from: Simon Bennett, John Skelton, and Ken Lunn. Schaum's Outlines – UML. McGraw-Hill. 2005.

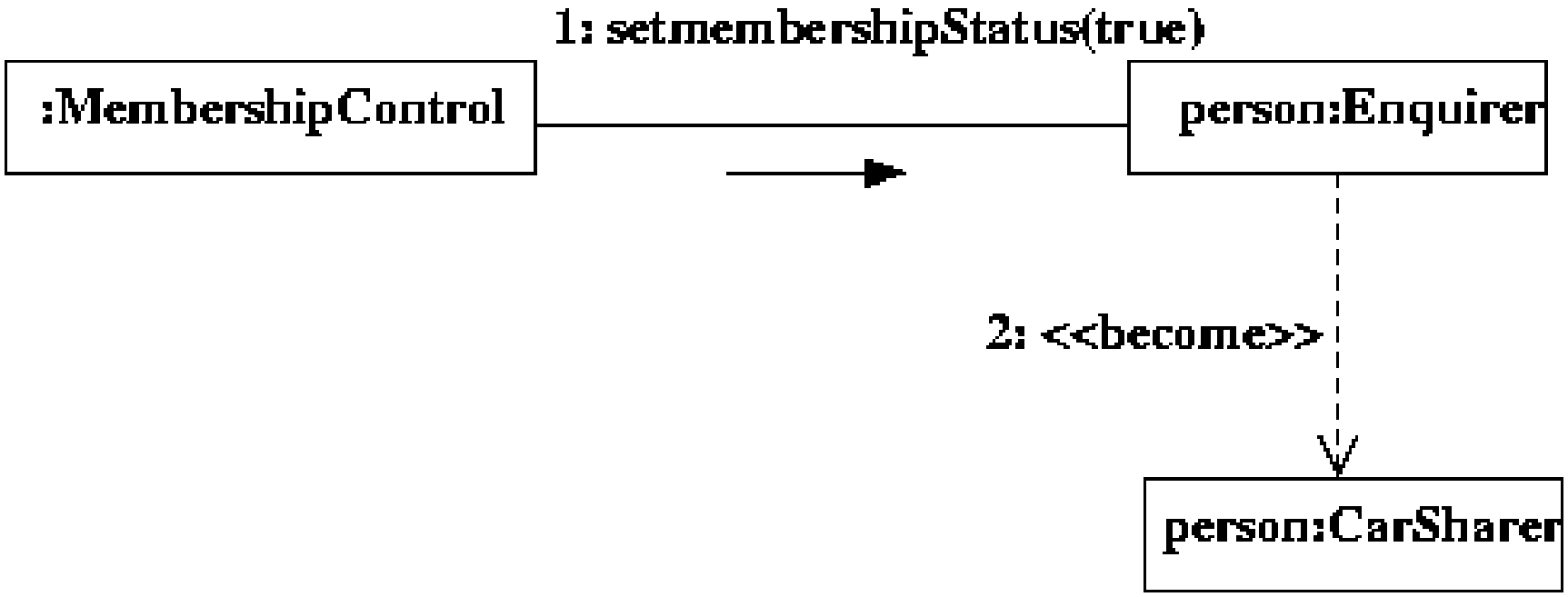
Multi-Objects

- Multiobjects: shown as two rectangles superimposed on one another.
- Represents a set of objects at the ``many'' end of an association, and to show that a message is sent to the set in order to obtain a link to each object in turn so that a further message can be sent to each.
- The individual object can be shown joined to the multiobject by a composition association.



Flow Relationships

- Show dependencies between two versions of an object at different points in time - when an object has significantly changed state <<become>>, or when an object has been copied <<copy>>



Model Consistency

- The interaction diagrams and the class diagrams should be mutually consistent.
- A good CASE tool will enforce this consistency at a syntactic level.
- Every sending object is required to know the identity or object reference of the destination object.
- Either this is know by the sending object through a direct link i.e. an association exists in the class diagram.
- Or the sending object obtains the reference indirectly from another object that has a link with the destination object.
- The representation and placement of the object references that represent associations is a design issue.
- Analysis: there is some possible pathway via object links from the sending object to the destination object, deduced from associations in the class diagram.
- Check for consistency: there may be some possible associations missing from the class diagram.
- The existence of an association does not guarantee the existence of a link.
- It is normal to revise the class model as you develop interaction diagrams.

Semi-formal Specification of Interaction

- The intent or purpose of the operation.
- The operation signature including the return type.
- An appropriate description of the logic (non-algorithmic vs. and algorithmic).
- Other operations called, whether in the same object or in other objects.
- Events transmitted to other objects.
- Attributes set during the operations execution.
- The response to exceptions.
- Any non-functional requirements that apply.
- Adapted from Larman, 1998; and Allen & Frost, 1998.

Reading

- Chapter 9 in Bennett et al. or
- Chapters 9 and 10 in Stevens and Pooley