Collaboration Diagram

Types of Interaction diagrams

There are two types of interaction diagrams:

- **Sequence** diagrams:
  - emphasize the **order / Sequence** or **concurrency** of the interactions.
- **Collaboration** diagrams:
  - emphasize the interacting **objects**.

Scenarios

- Use case diagram presents an outside view of the system.
- Flow of events, captures the functionality of the use case
- Scenarios are used to describe how use cases are realized as interactions among societies of objects

Scenarios

- It is one path through the flow of events for the use case.
- Scenarios help discovering objects, classes and object interaction needed to carry out a piece of the functionality in use case
- Scenarios shows how the responsibility of a use case are distributed among the objects

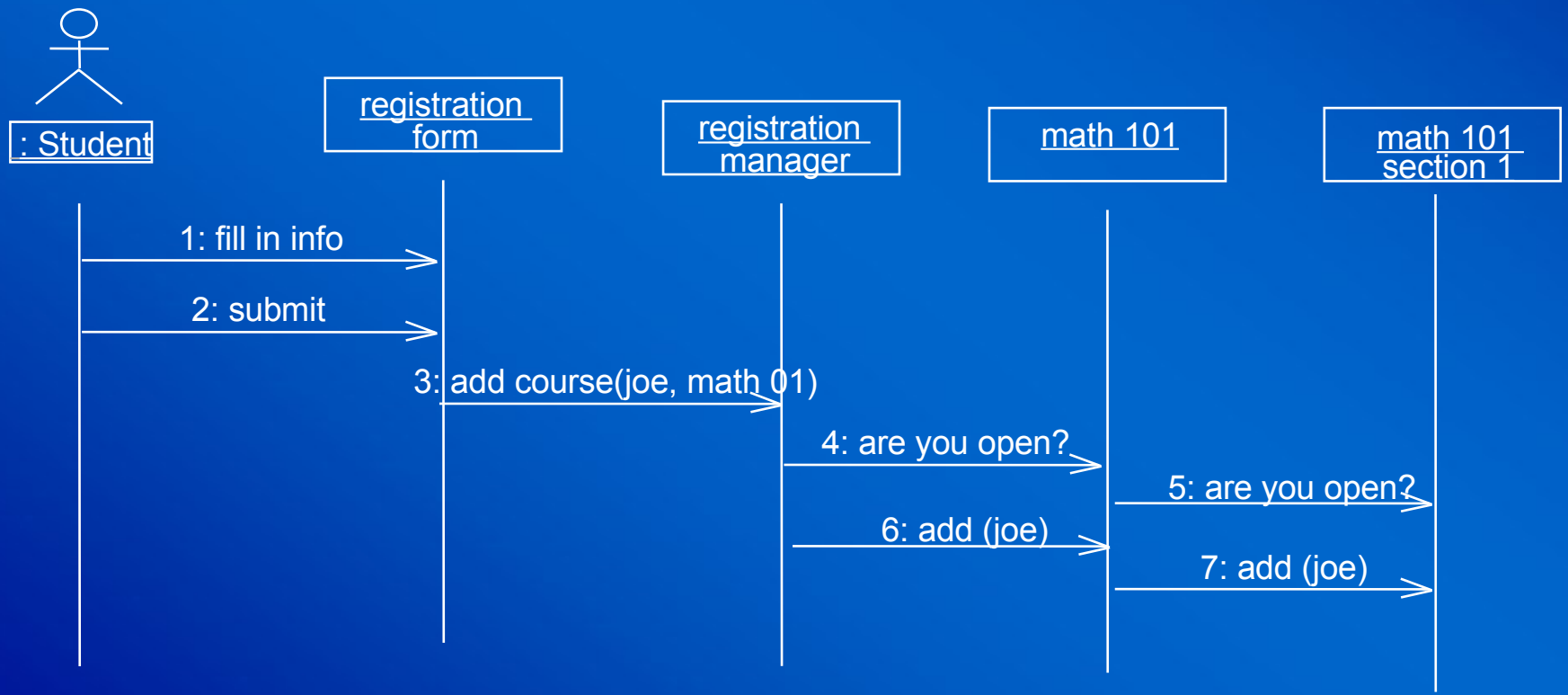
Scenarios how to capture?

- Sequence Diagram
- Collaboration Diagram

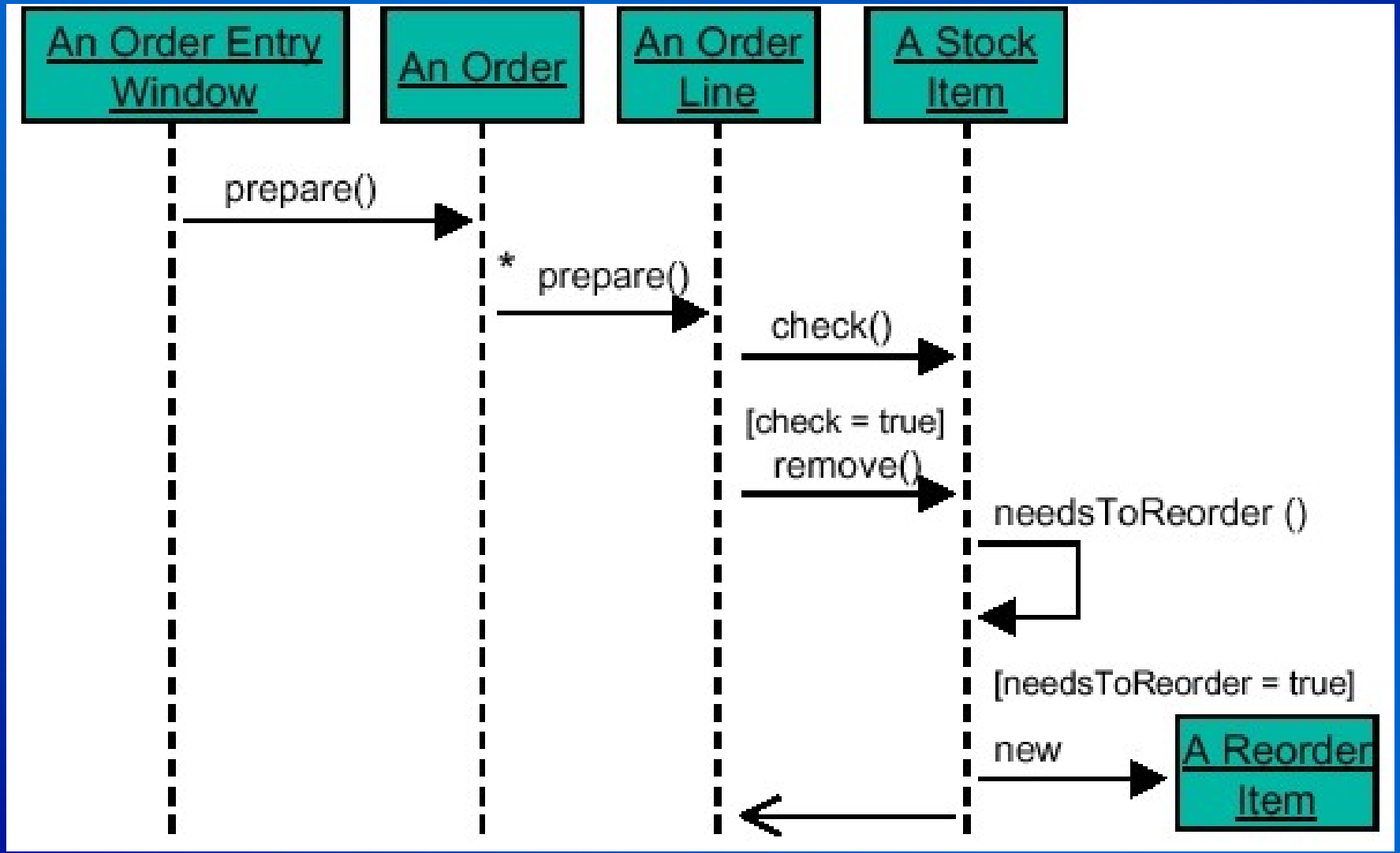
Sequence Diagram

Sequence Diagrams

- A sequence diagram displays object interactions arranged in a time sequence



Sequence Diagram (Example)

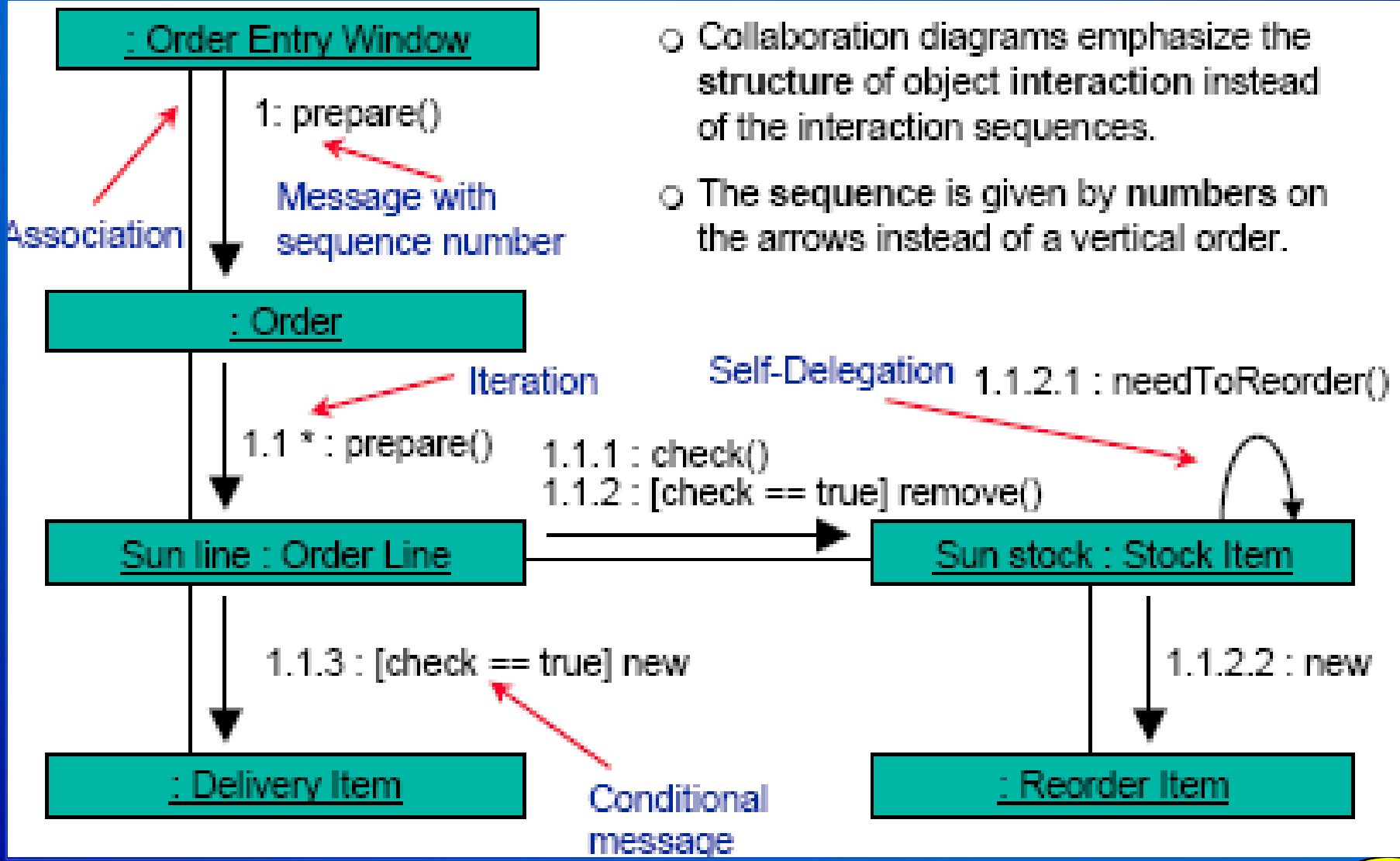


Collaboration Diagram

Collaboration Diagrams

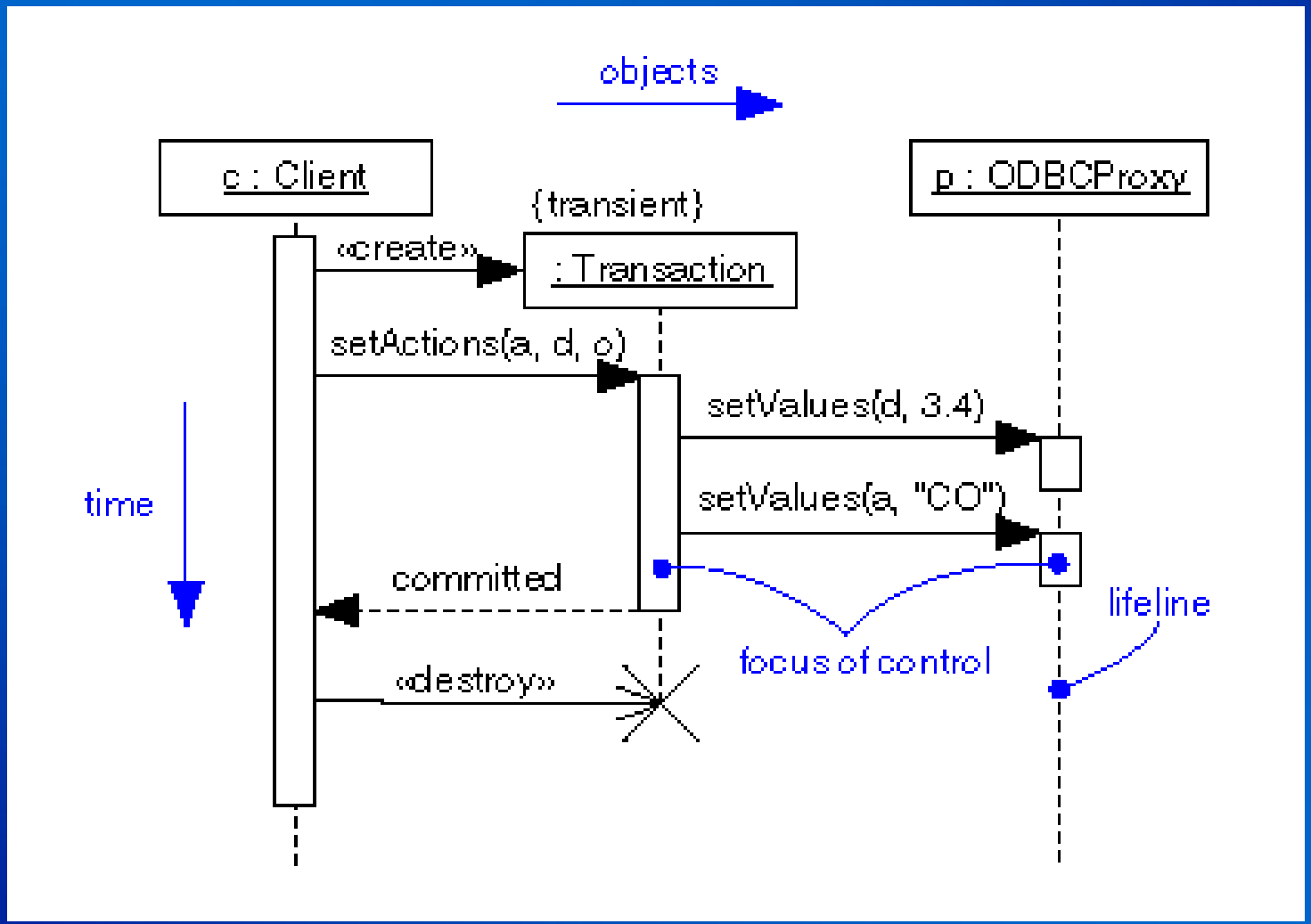
- Dynamic behavior of objects can, in addition to sequence diagrams, also be represented by **collaboration** diagrams.
- The transformation from a sequence diagram into a collaboration diagram is a bi-directional function.
- The difference between sequence diagrams and collaboration diagrams is that collaboration diagrams emphasize more the structure than the sequence of interactions.
- Within sequence diagrams the order of interactions is established by vertical positioning whereas in collaboration diagrams the sequence is given by numbering the interactions.

Collaboration Diagrams

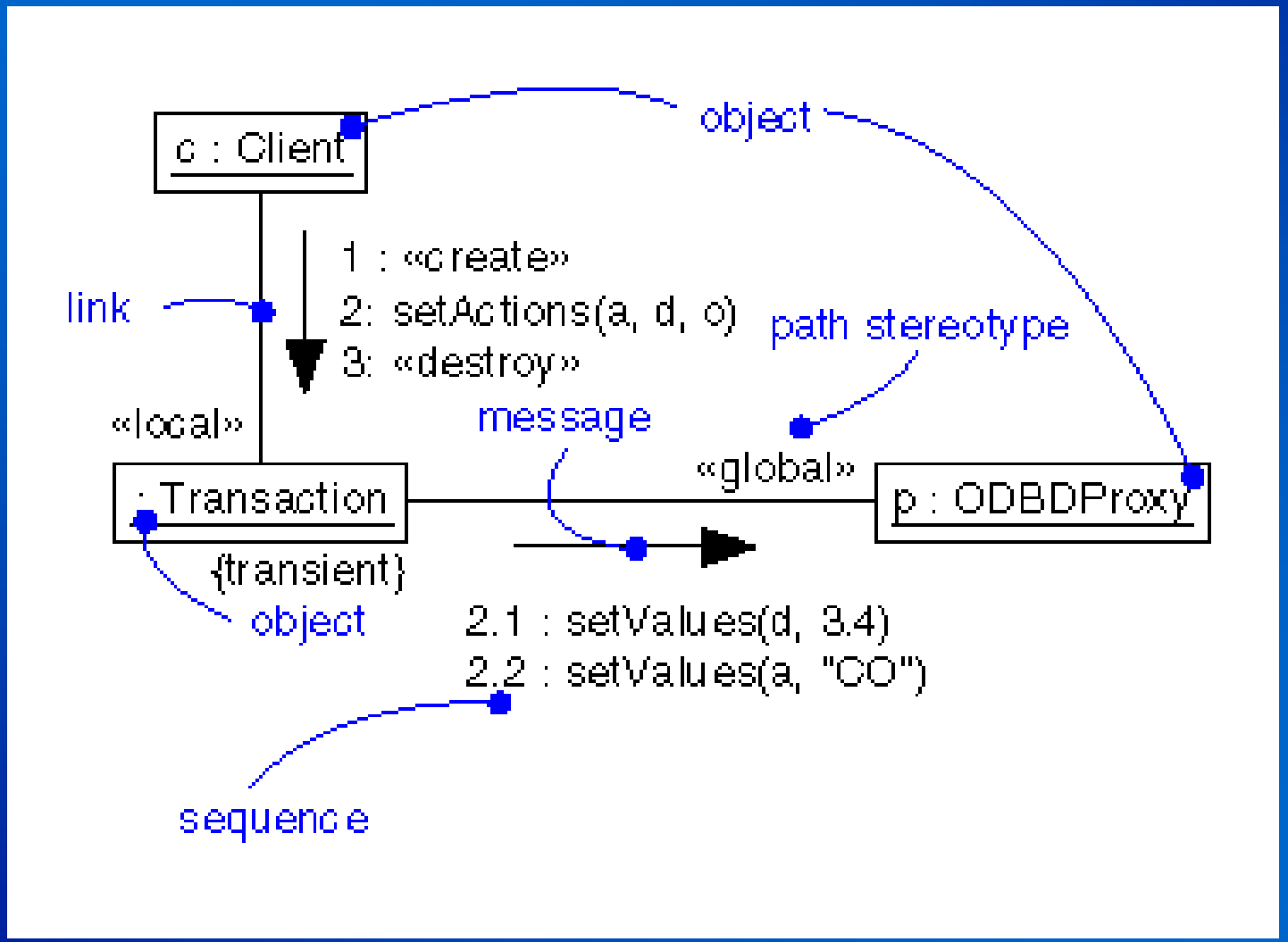


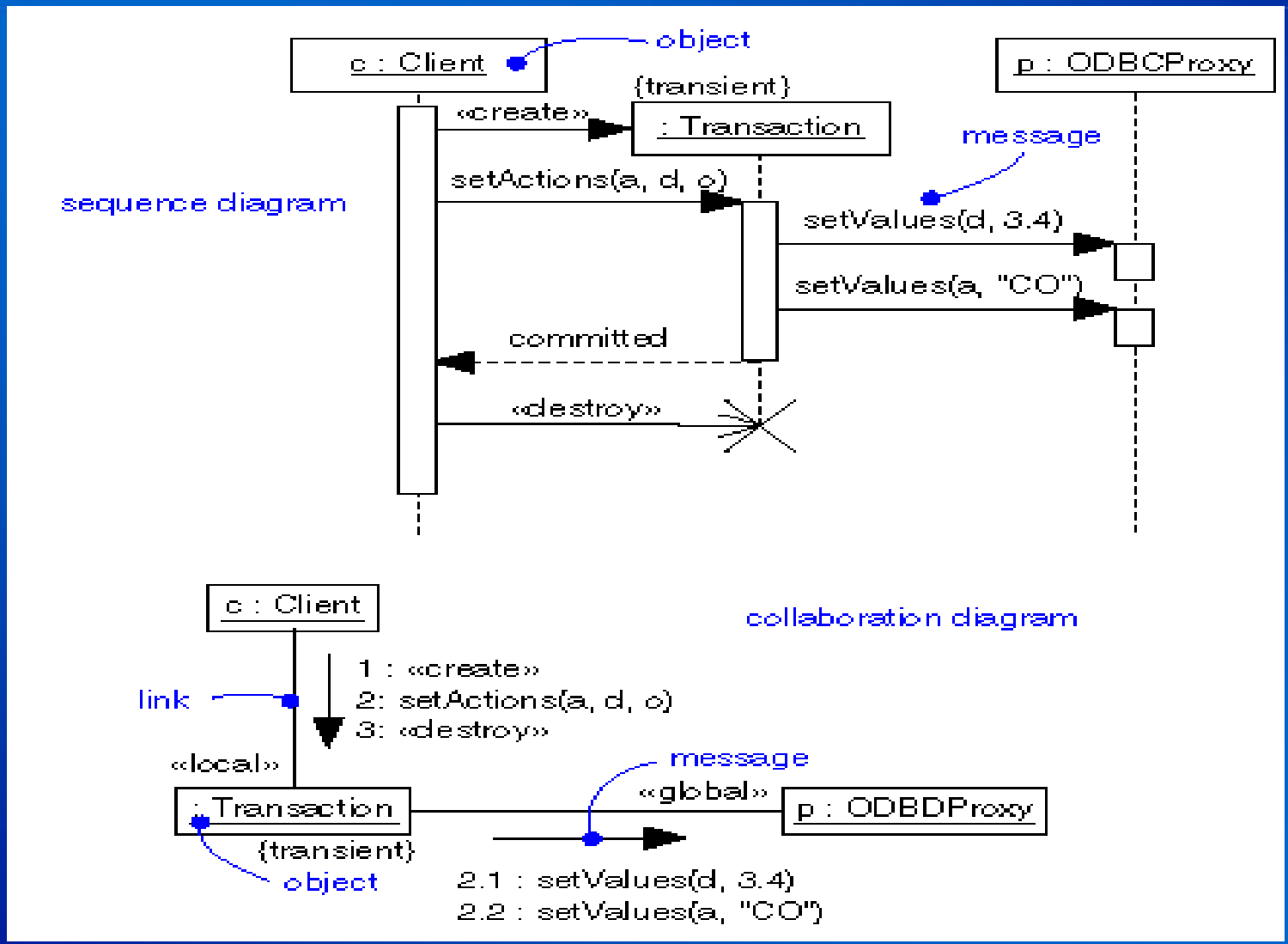
- Collaboration diagrams emphasize the **structure** of object interaction instead of the interaction sequences.
- The **sequence** is given by **numbers** on the arrows instead of a vertical order.

Another Example Sequence diagram



Another Example Collaboration diagram




c : Client
object
p : ODBCProxy
{transient}
«create»
: Transaction
message
setActions(a, d, o)
sequence diagram
setValues(d, 3.4)
setValues(a, "CO")
committed
«destroy»
c : Client
collaboration diagram
1 : «create»
link
2: setActions(a, d, o)
3: «destroy»
«local»
message
«global»
: Transaction
p : ODBDProxy
{transient}
object
2.1 : setValues(d, 3.4)
2.2 : setValues(a, "CO")

Fun Example
Objects

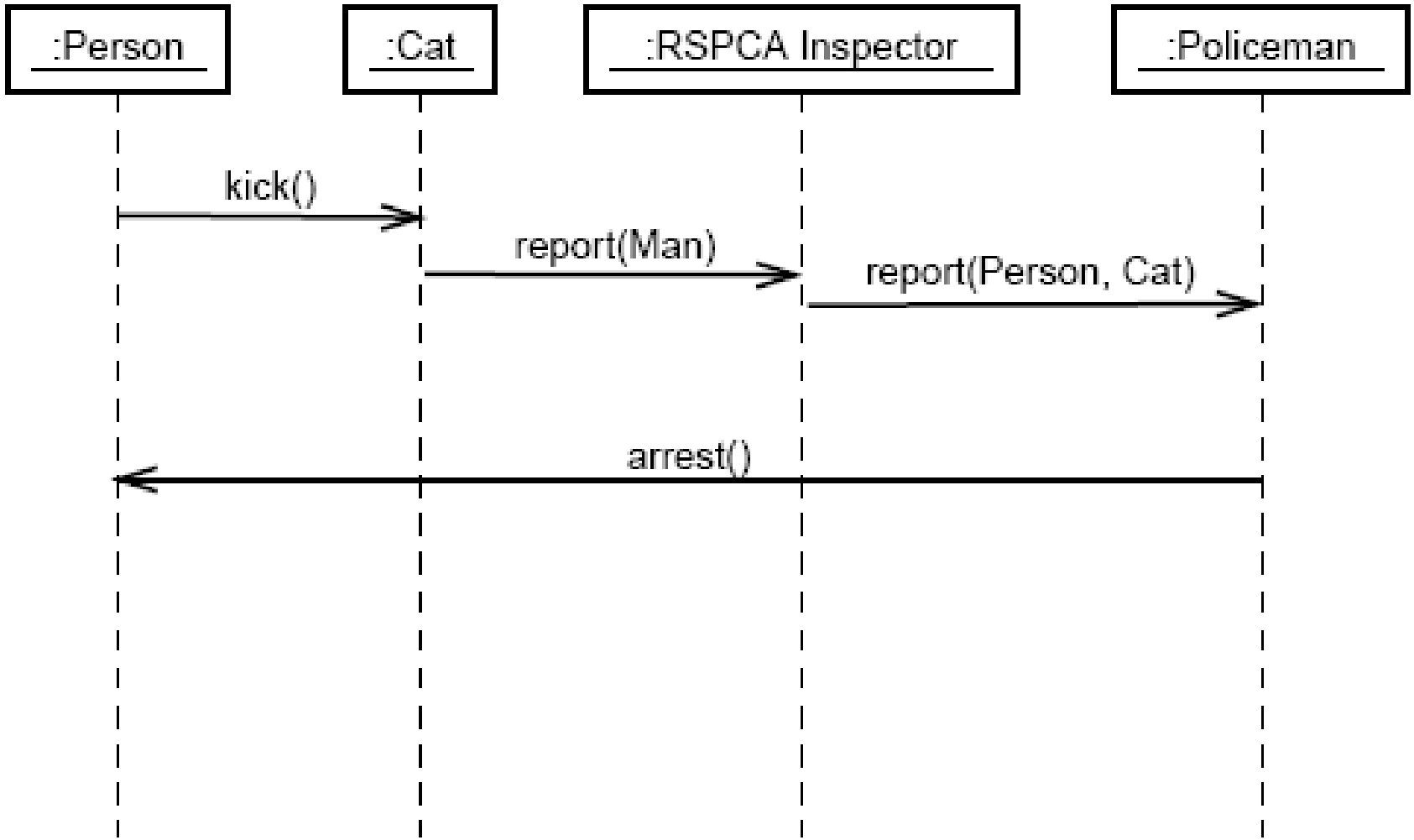
:Cat

:Policeman

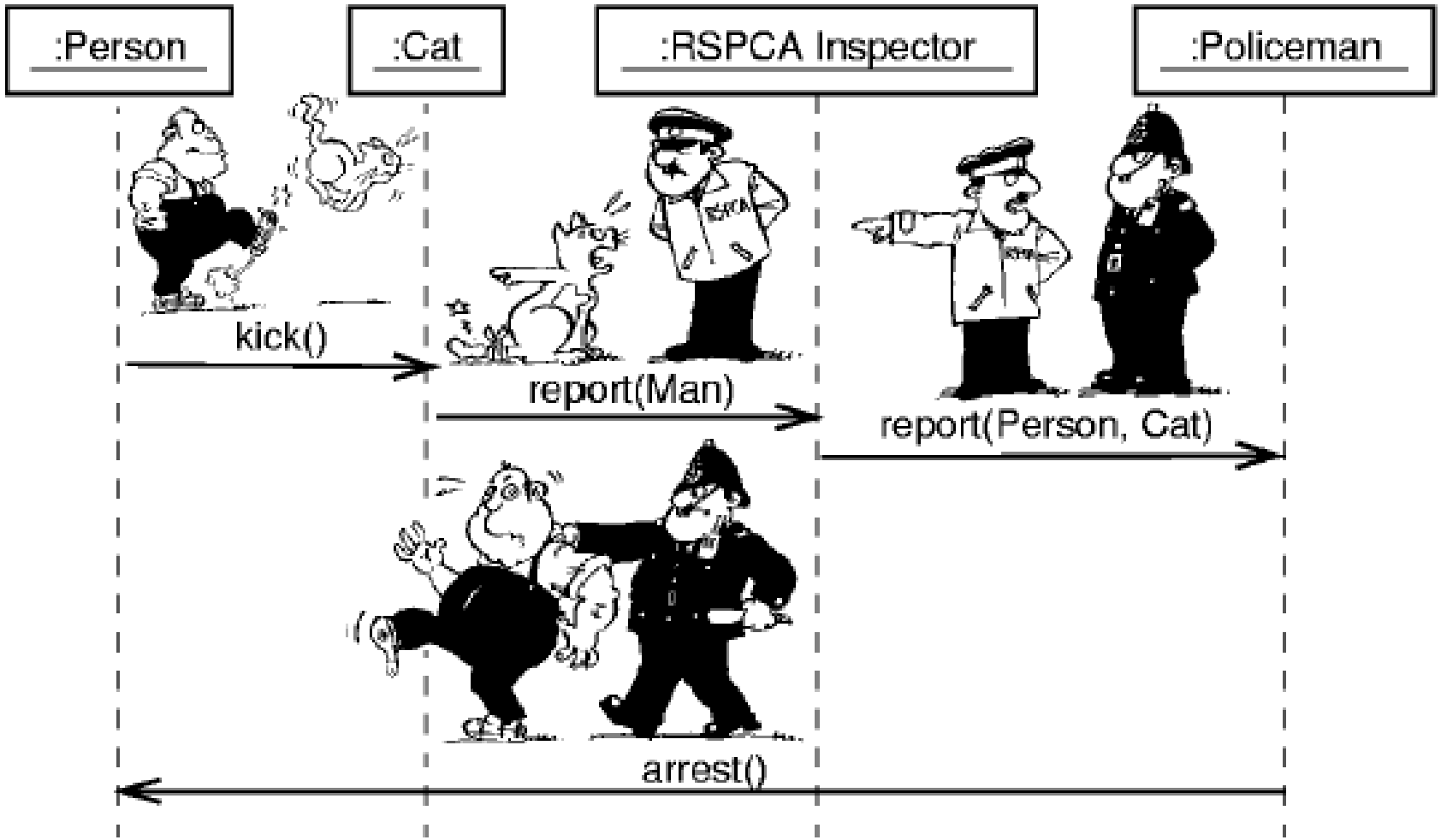
:Person

:RSPCA Inspector

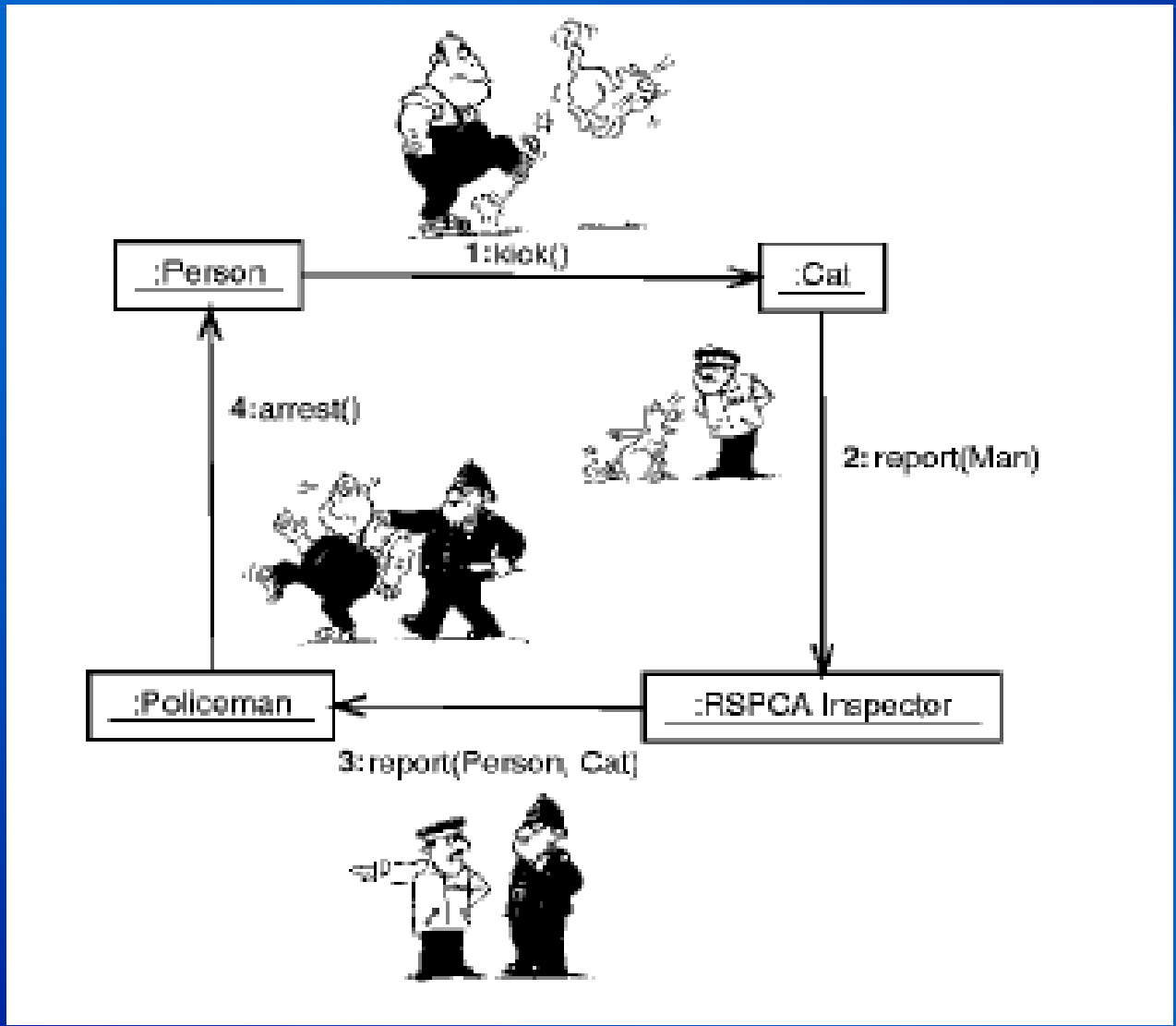
Fun Example
Sequence diagram



Fun Example
Sequence diagram



Fun Example
Collaboration diagram



When to use Interaction Diagrams

Use Interaction Diagrams

- When catching user requirements:
  - Describe the behavior of several objects within a single use case.
  - Show collaborations among objects.
- After having described the object behavior completely with state and activity diagrams:
  - Test the state and activity diagrams against the scenarios.

Do not Use Interaction Diagrams

- For precise definition of a single class behavior (use state diagrams).
- If you want to describe the behavior across many use cases or many threads (consider an activity diagram).

Rational Rose
Live Demo

Your Turn - Lab Activity
Practicing Interaction Diagrams

Use Rational Rose 2002 to create Sequence and Collaboration diagrams for the Scenario provided in Lab 09 of lab manual

What Next…?

- For your project, you need to:
  - Discover Classes from flow of events
  - Identify Boundary, Control and Entity Classes
  - Capture dynamic behavior of use case using Sequence and Collaboration diagrams