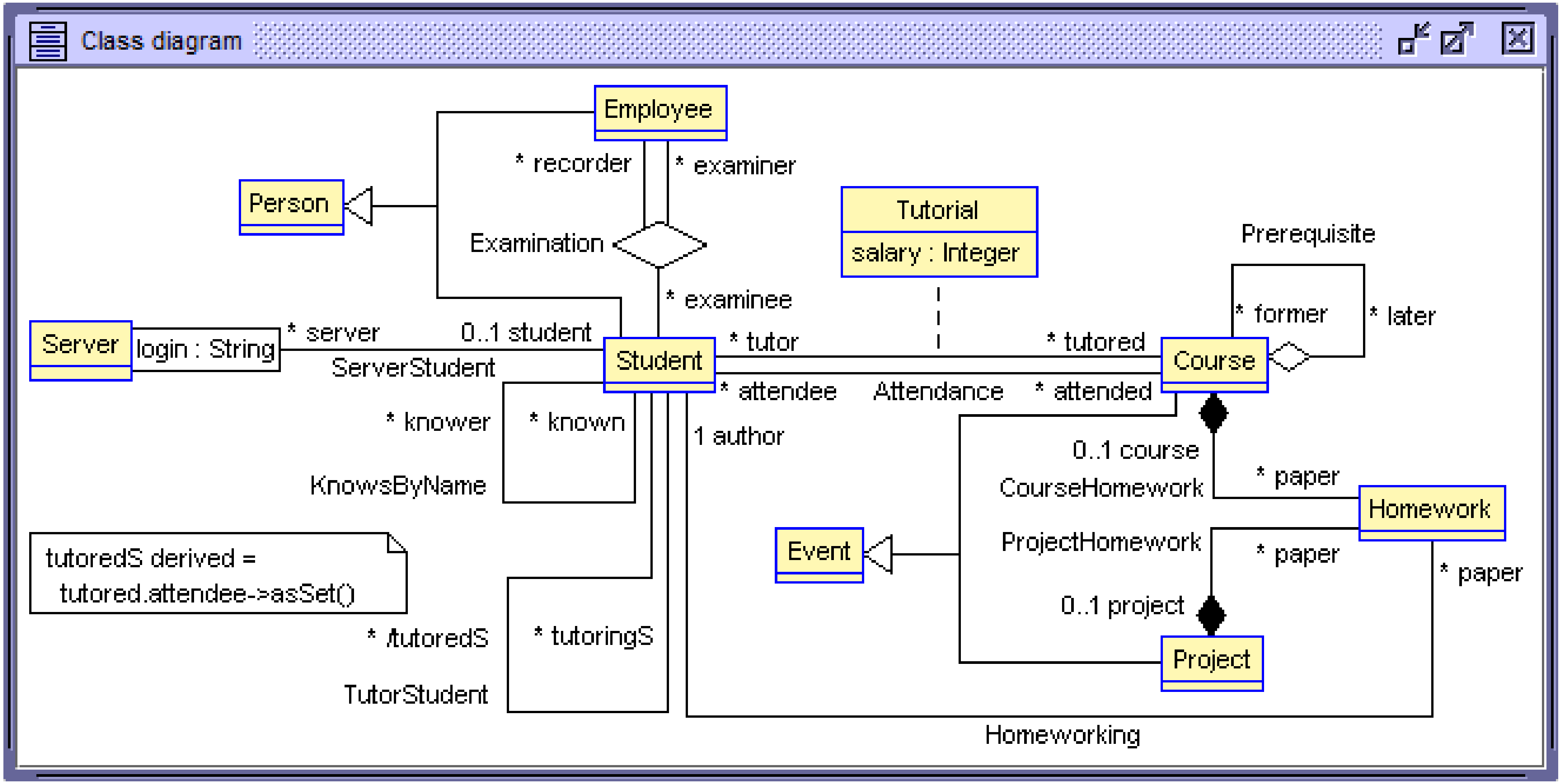
Design of Information Systems

Variety of UML Associations

Martin Gogolla
University of Bremen, Germany
Database Systems Group

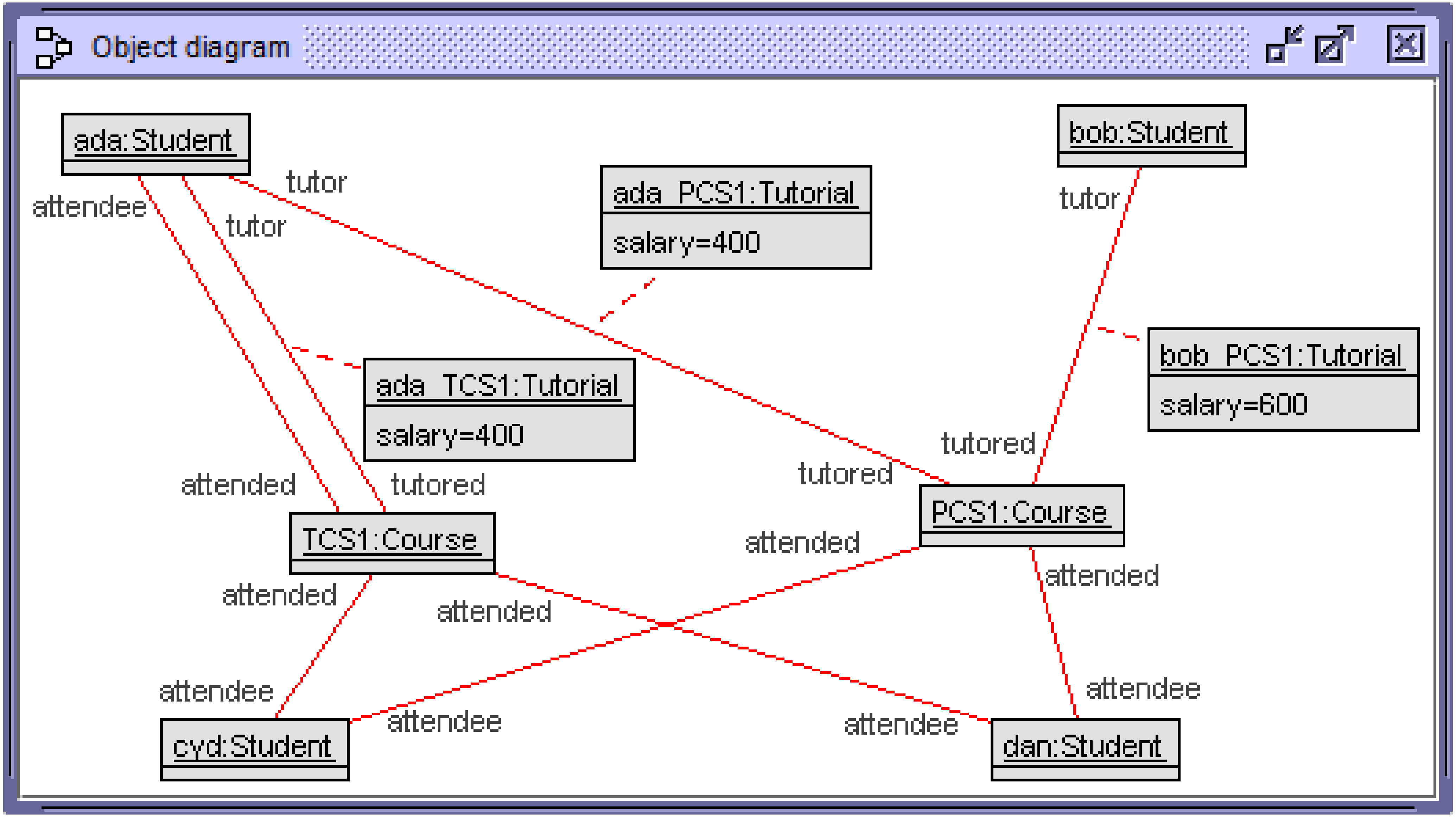
UML Associations

- Binary association; standard association with two association arms
- Association class; association with attributes
- Refexive association; association with one class participating twice
- Ternary / n-ary association; with at least three association arms
- Aggregation; weak part-whole relationship; weak binding of part to whole
- Composition: strong part-whole relationship; strong binding of part to whole
- Functional association; 0..1 or 1..1 multiplicity
- Qualified association; with qualifier attribute(s); array-like construct
- Derived association; with defining OCL term; determines path in class diagram
- *Above classification not disjoint, partly overlapping*

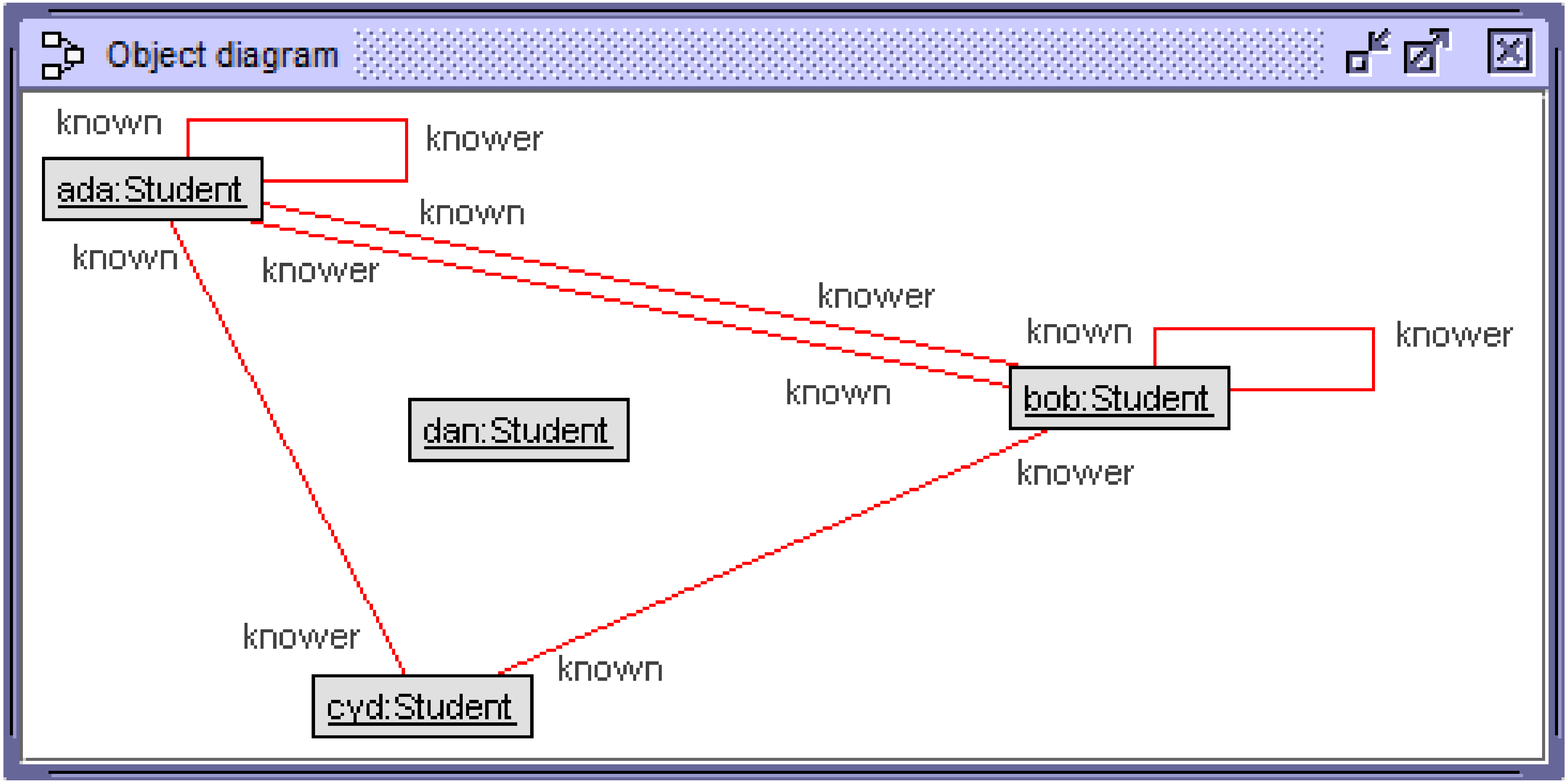


tutoredS derived =
tutored.attendee->asSet()

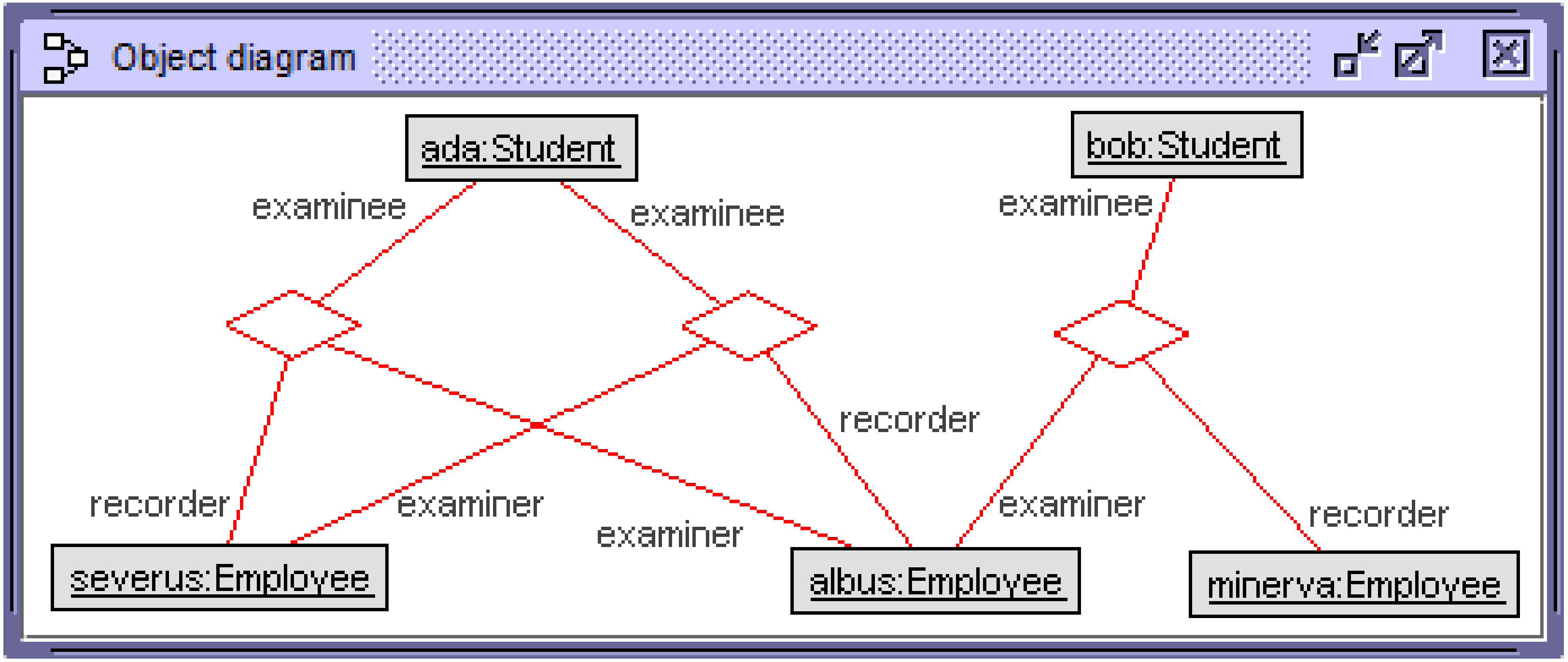
Binary association & Association class



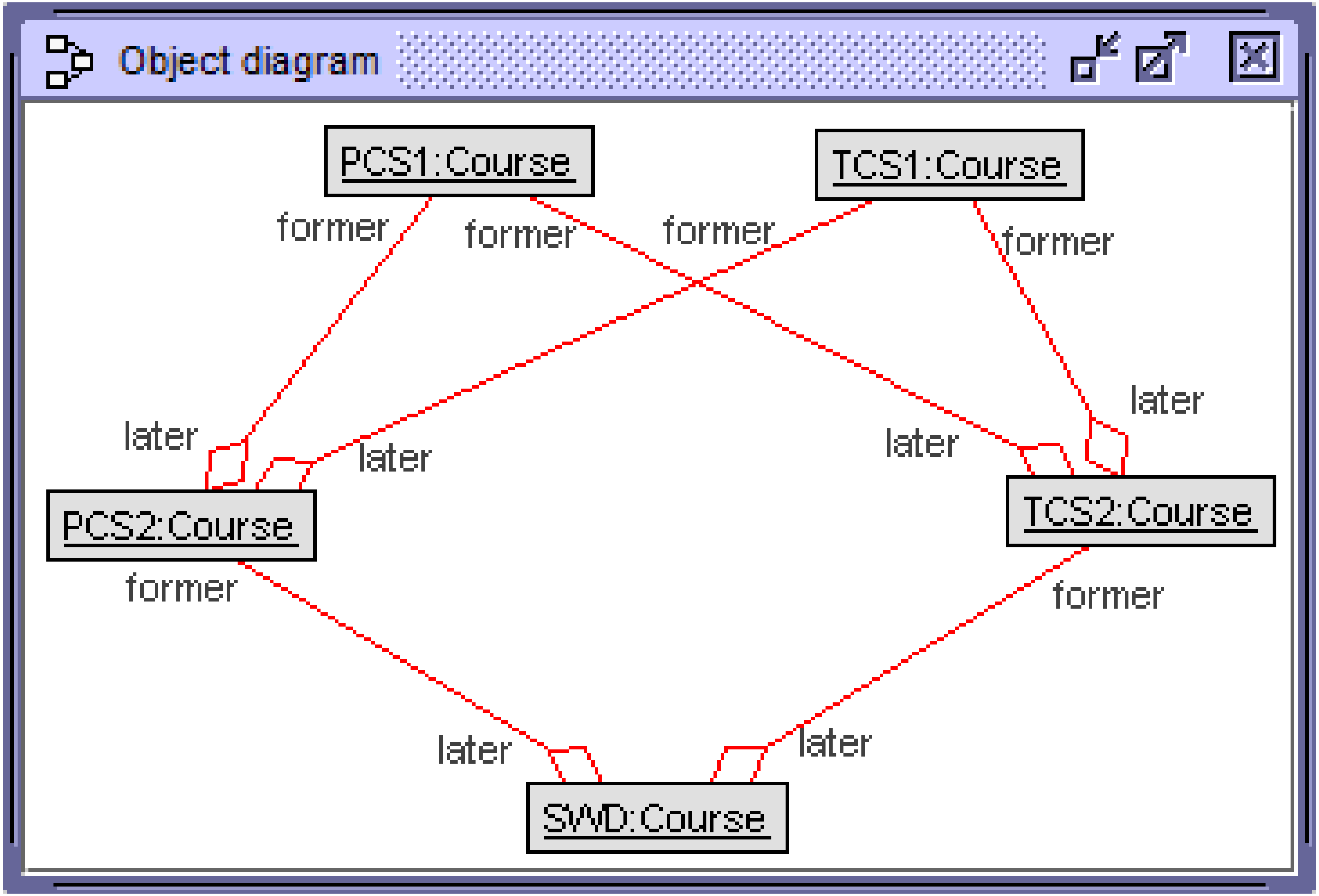
Reflexive association



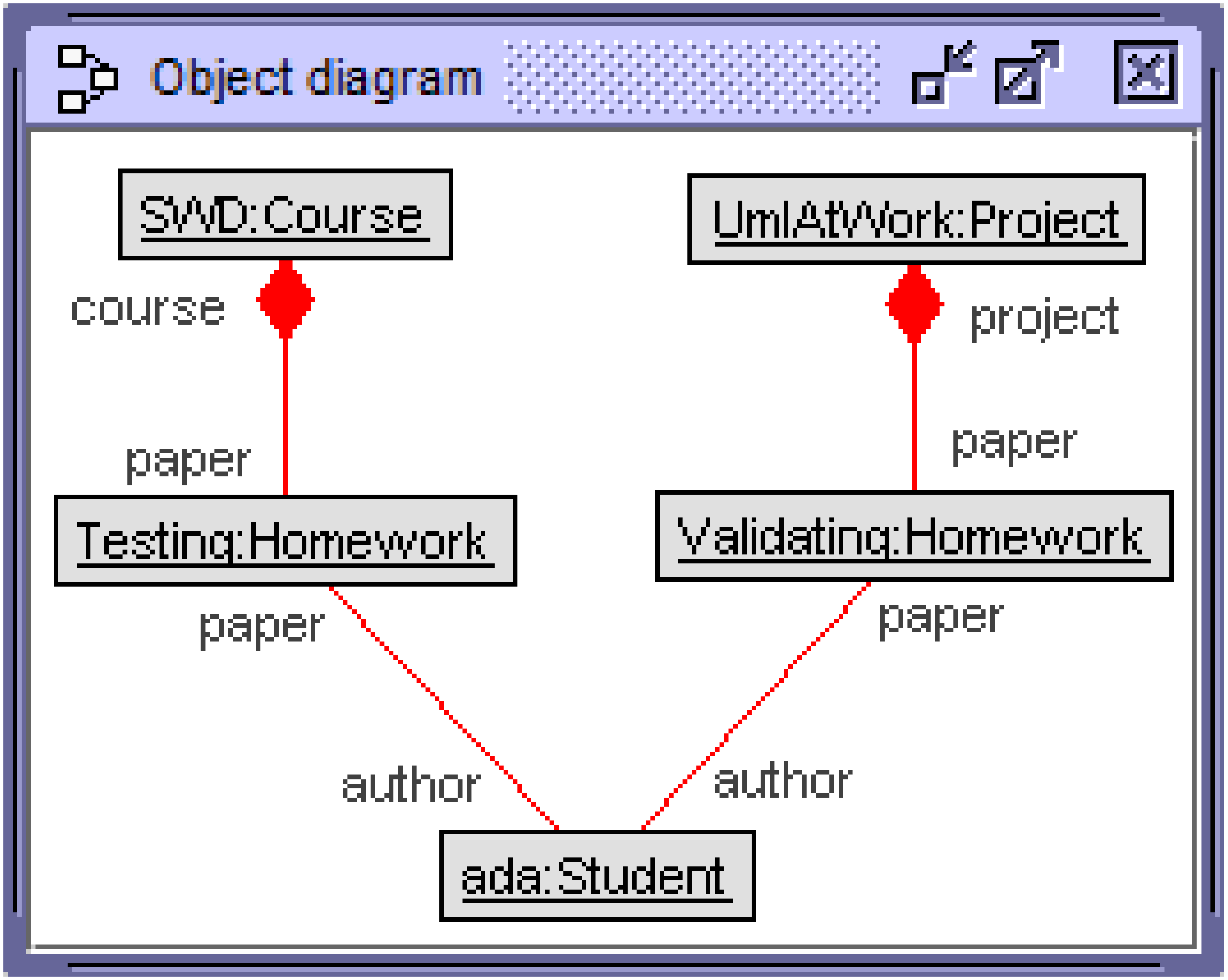
Ternary association



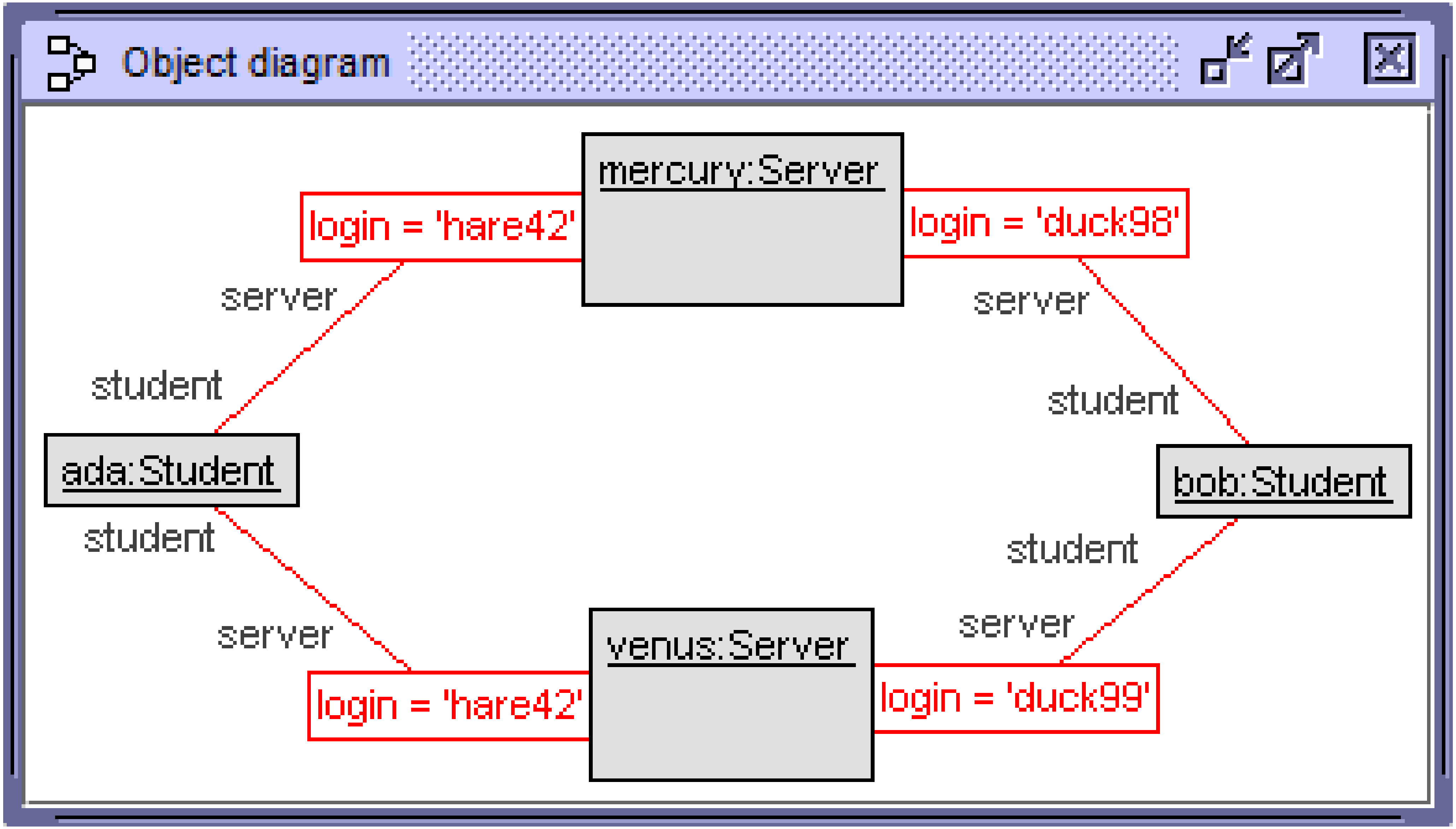
Aggregation



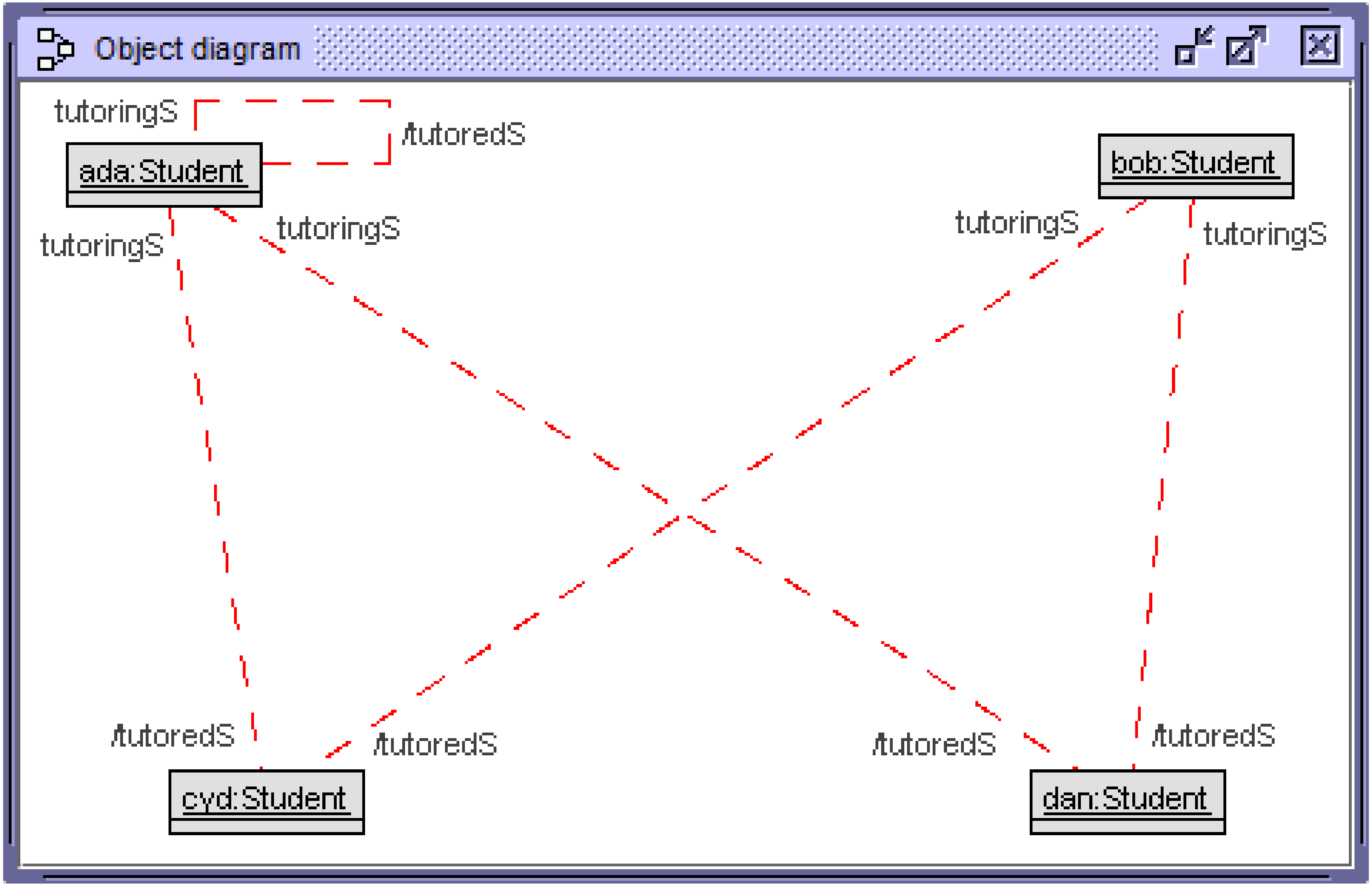
Composition & Functional association



Qualified association



Derived association



USE model

```
model StudentWorld

class Person
end

class Event
end

class Student < Person
end

class Course < Event
end

class Project < Event
end

association Attendance between
  Student[*] role attendee
  Course[*] role attended
end
```

USE model

```
associationclass Tutorial between
  Student[*] role tutor
  Course[*] role tutored
attributes
  salary:Integer
end

association KnowsByName between
  Student[*] role knower
  Student[*] role known
end

class Employee < Person
end

association Examination between
  Student[*] role examinee
  Employee[*] role examiner
  Employee[*] role recorder
end
```

USE model

```
aggregation Prerequisite between
  Course[*] role later
  Course[*] role former
end

class Homework
end

association Homeworking between
  Student[1] role author
  Homework[*] role paper
end

composition CourseHomework between
  Course[0..1] role course
  Homework[*] role paper
end
```

USE model

```
composition ProjectHomework between
  Project[0..1] role project
  Homework[*] role paper
end

class Server
end

association ServerStudent between
  Server[*] role server qualifier(login:String)
  Student[0..1] role student
end

association TutorStudent between
  Student[*] role tutoringS
  Student[*] role tutoredS derived=self.tutored.attendee->asSet
end
```