Introduction to Rational Rose

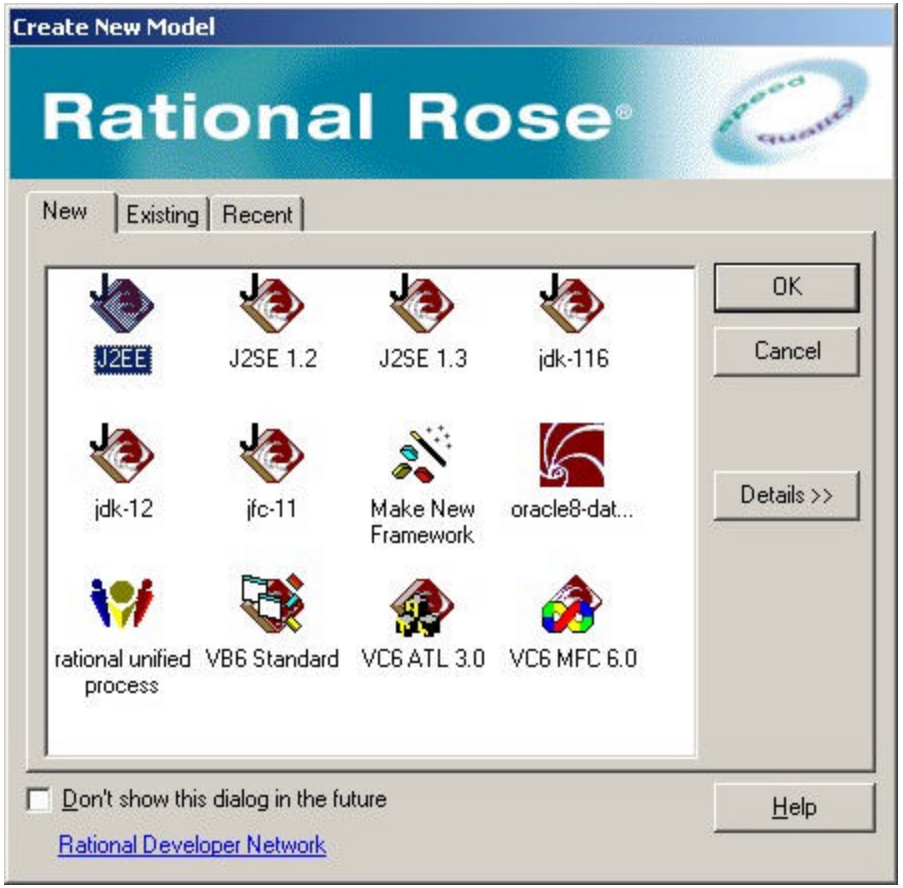
By Marc-André Laverdière

Introduction to Rational Rose

Topics Covered

- Opening the software and creating a new project
- MDL files
- Internal folders & Organization
- How to draw a Use Case Diagram
- How to draw a Class Diagram
- How to draw a Sequence Diagram
- Quirks & Perks to Know

Opening the software and creating a new project

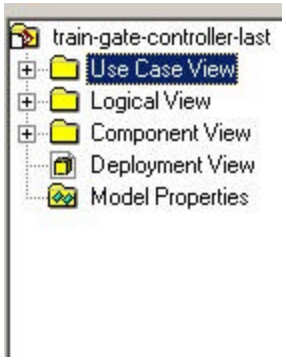


- This screen welcomes you when you open Rational Rose
- You can select the option related to the framework you wish to use. It impacts mostly for code generation
- Generally, select ‘rational unified process’

MDL Files

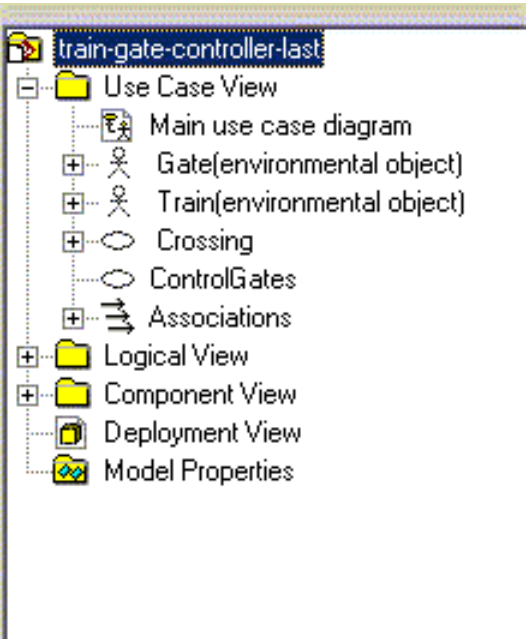
- Rational Rose’s native format uses .mdl files
- These files do not support version control nor concurrent editing
- Some advanced collaborative environment is possible, but not configured
- The important point to remember is that a changing one element will have the change cascaded to everything that uses the element.

Internal folders & Organization



- The folders are grouped based on Kruchten's 4+1 Architecture view.

Internal folders & Organization

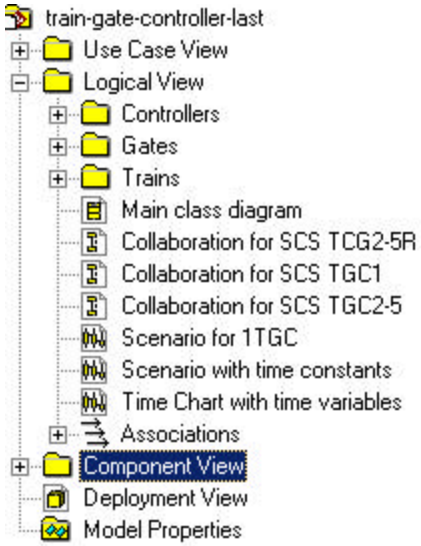


Use Case View

- **Audience:** All users of other views and Evaluators.
- **Addresses:** System consistency, validity
- **Method:** English, graphical notations, UML use cases.
- Help Architect
- Help illustrate and validate the document
- Basis for system testing

Ref: http://plg.uwaterloo.ca/~aeehassa/cs746/pres1/Arch1.html

Internal folders & Organization

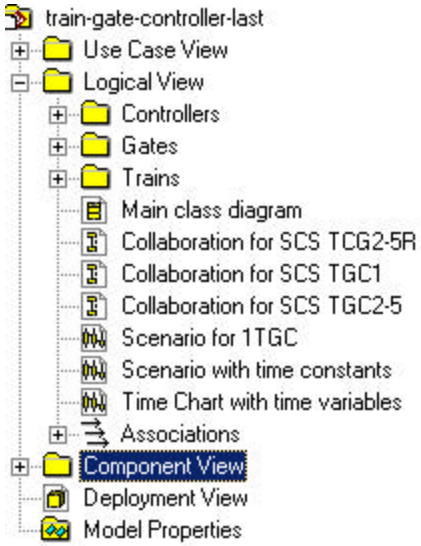


Logical View

- **Audience:** End-user
- **Addresses:** Functional requirements
- **Method:** OMT notation (object and dynamic models)

Ref: http://plg.uwaterloo.ca/~aeehassa/cs746/pres1/Arch1.html

Internal folders & Organization

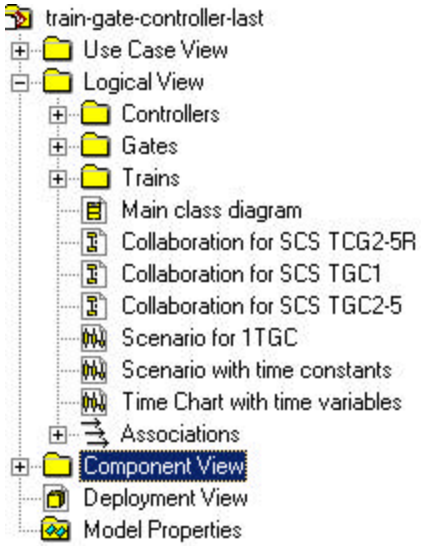


Component View

- The component view addresses the software organization of the system. This view contains information about the software, executable and library components for the system. This view contains only **component diagrams**. The component view contains a Main diagram by default. Additional diagrams can be added.
- Of little interest for small projects

http://www.comp.leeds.ac.uk/ise/rose_tutorial.htm

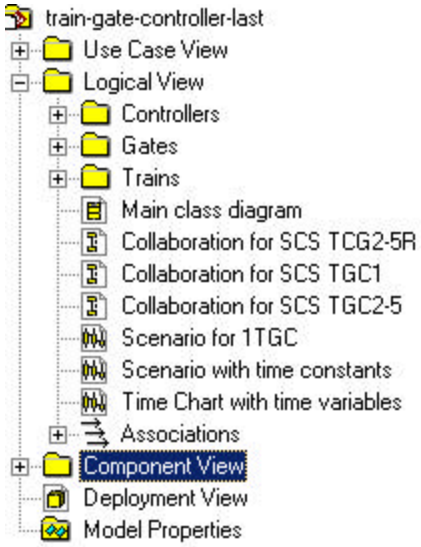
Internal folders & Organization



Deployment View

- Focuses on hardware issues
- Intended mostly for distributed computing issues
- Of no interest in most cases

Internal folders & Organization

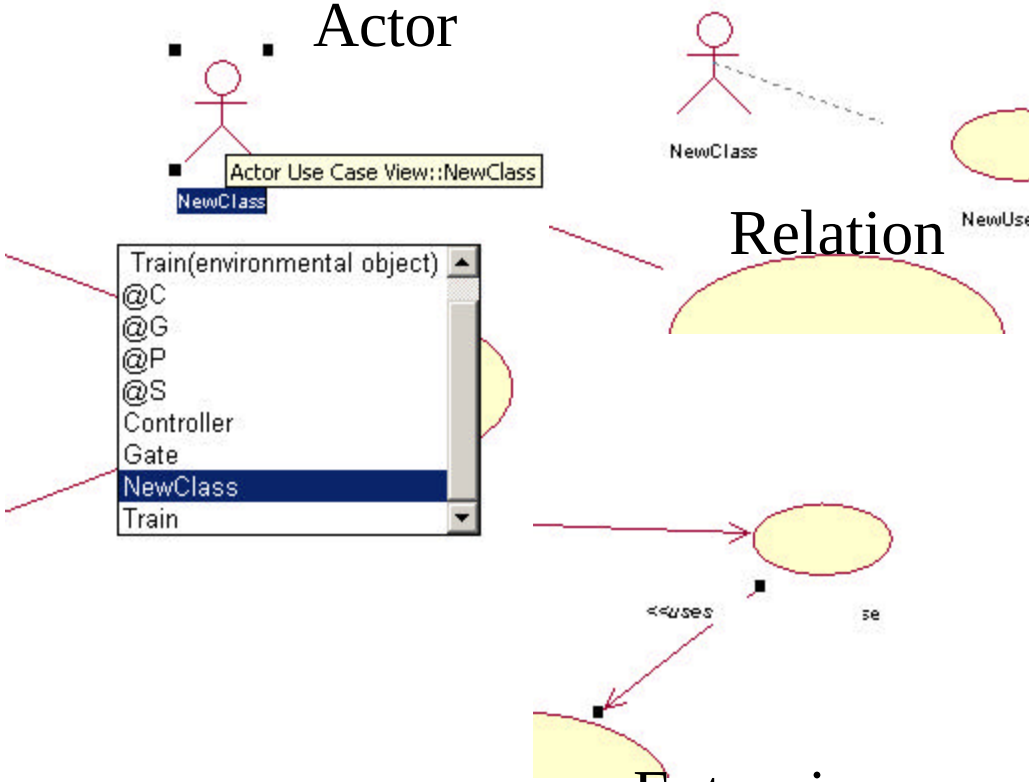


Subdirectories

- Use subdirectories to organize your models. It simplifies things when the model gains complexity.
- Allows to create a supplemental level of abstraction by grouping some objects.

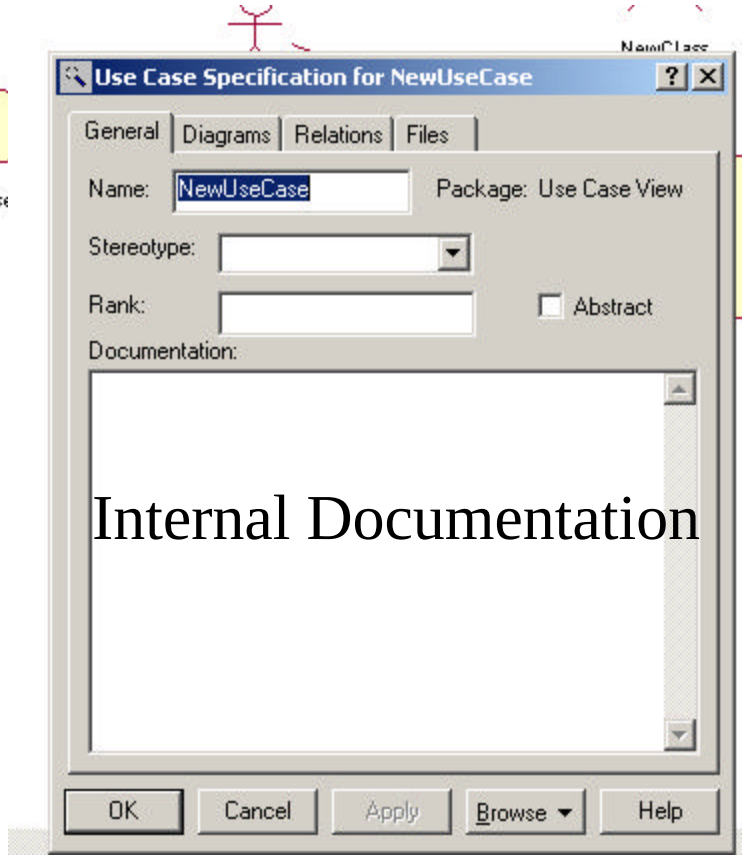
How to draw a Use Case Diagram

Actor



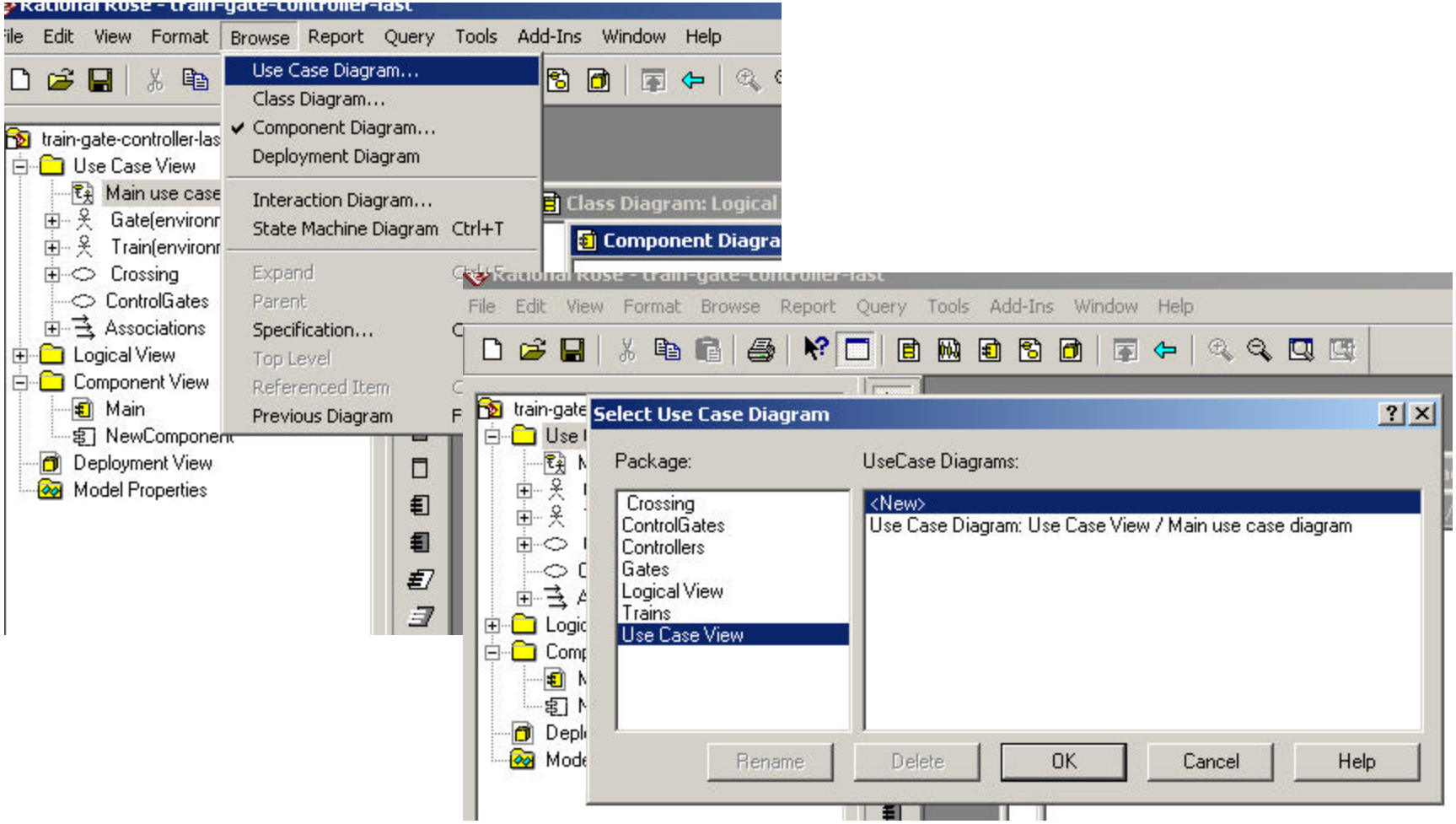
Relation

Extension

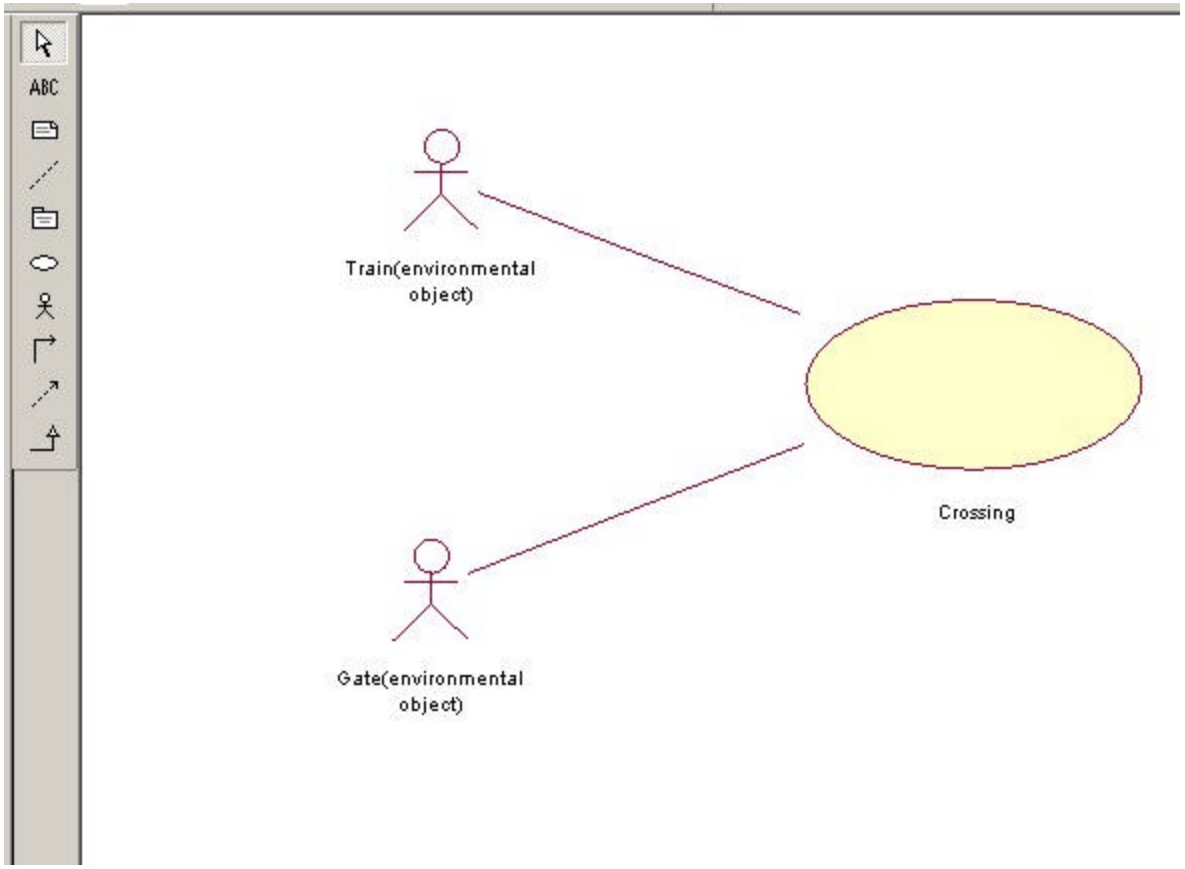


Internal Documentation

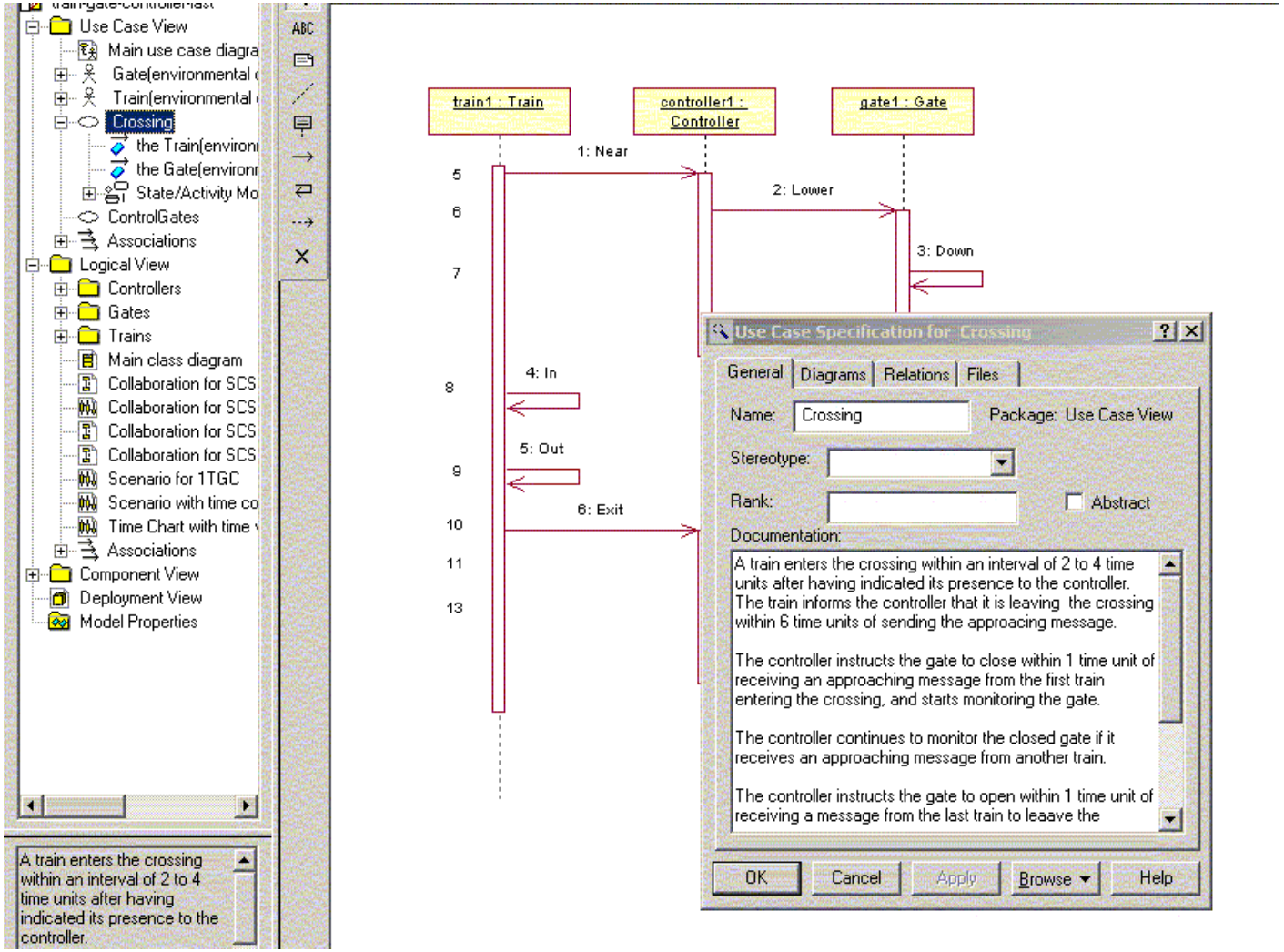
Creating a New Use Case Diagram



Use Case Editing

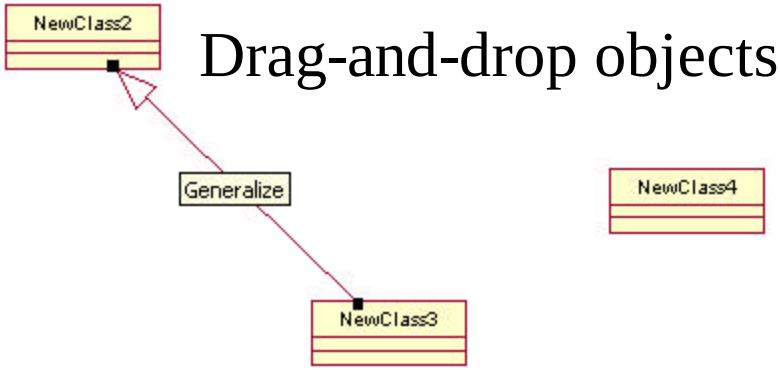


Use Case Editing



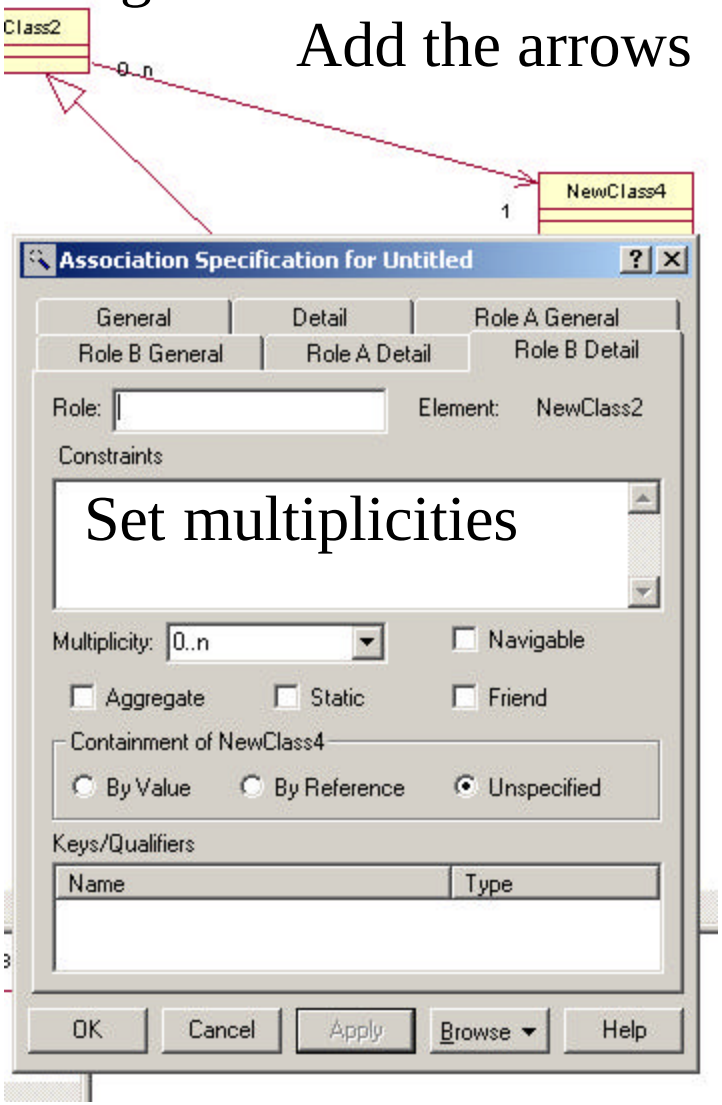
How to draw a Class Diagram

Name new classes directly



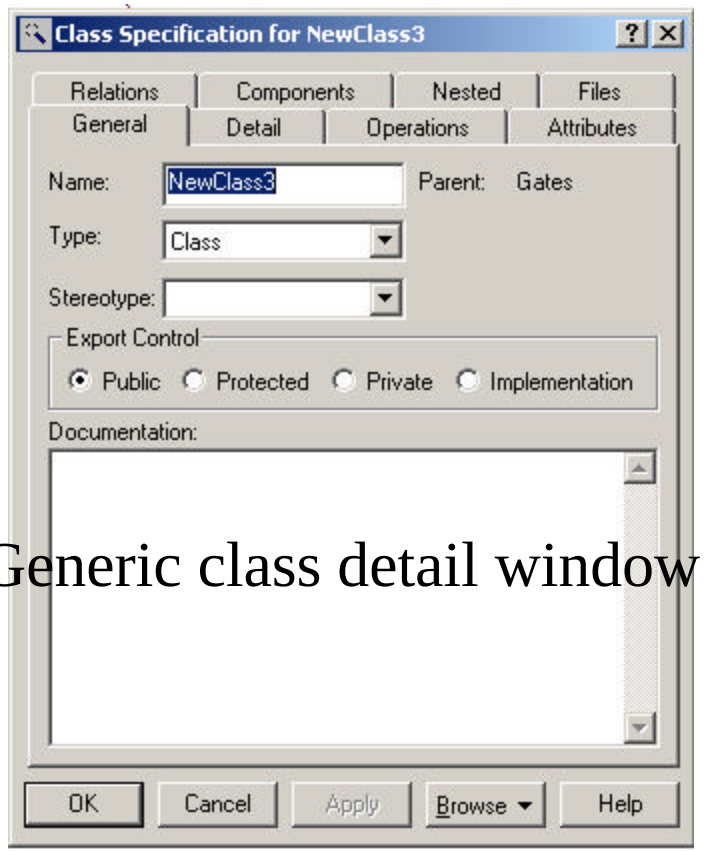
Drag-and-drop objects

Add the arrows

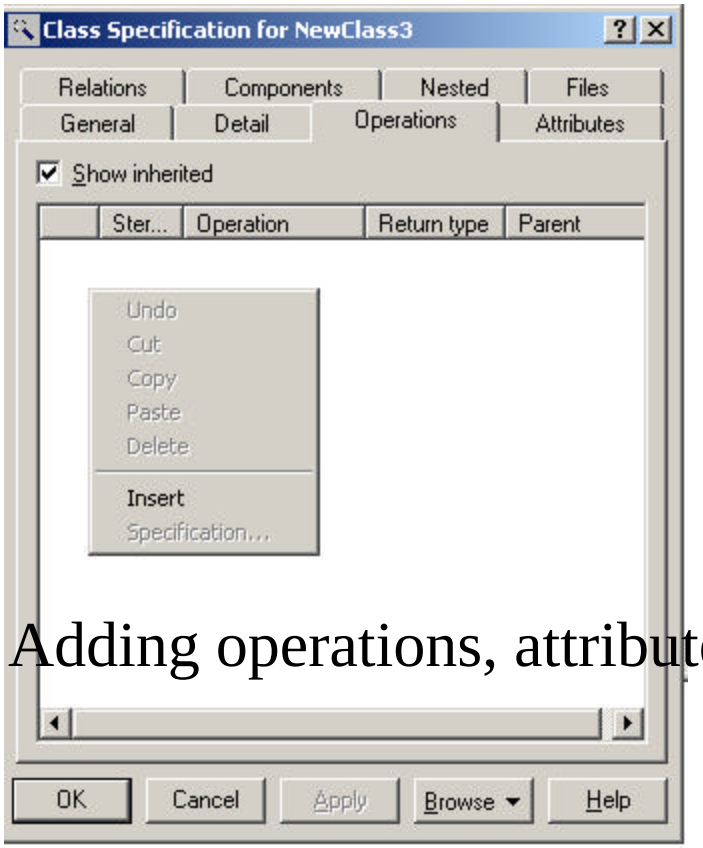


Set multiplicities

Editing a Class

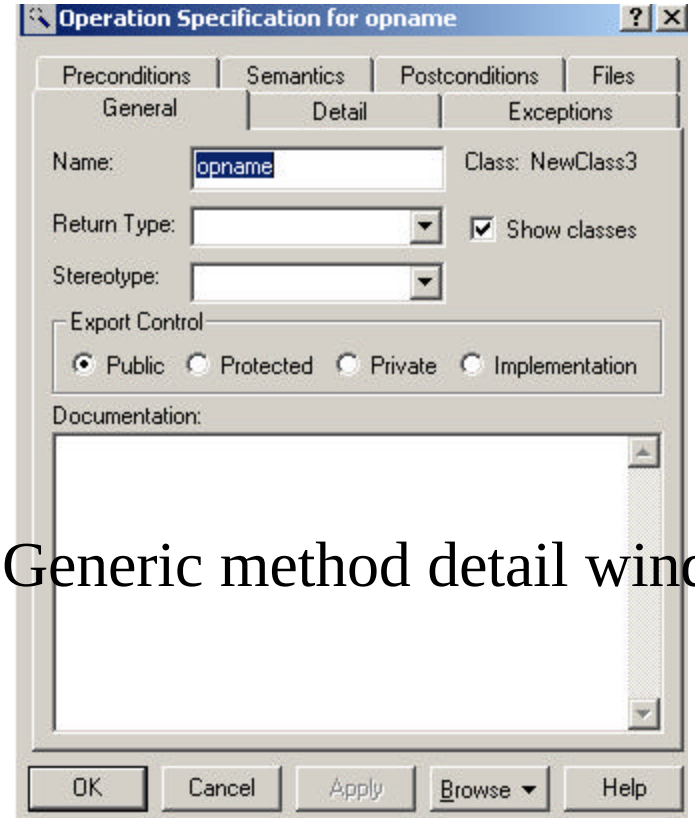


Generic class detail window

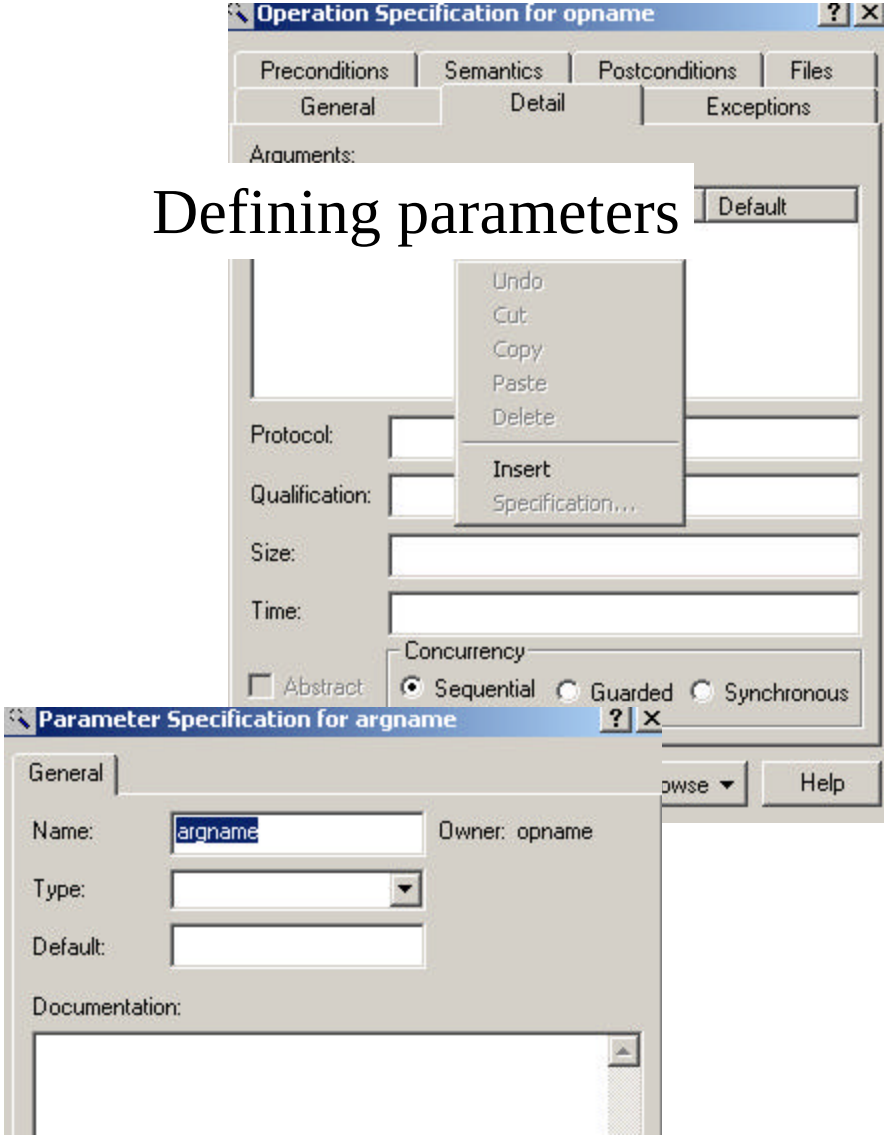


Adding operations, attributes

Specifying Methods

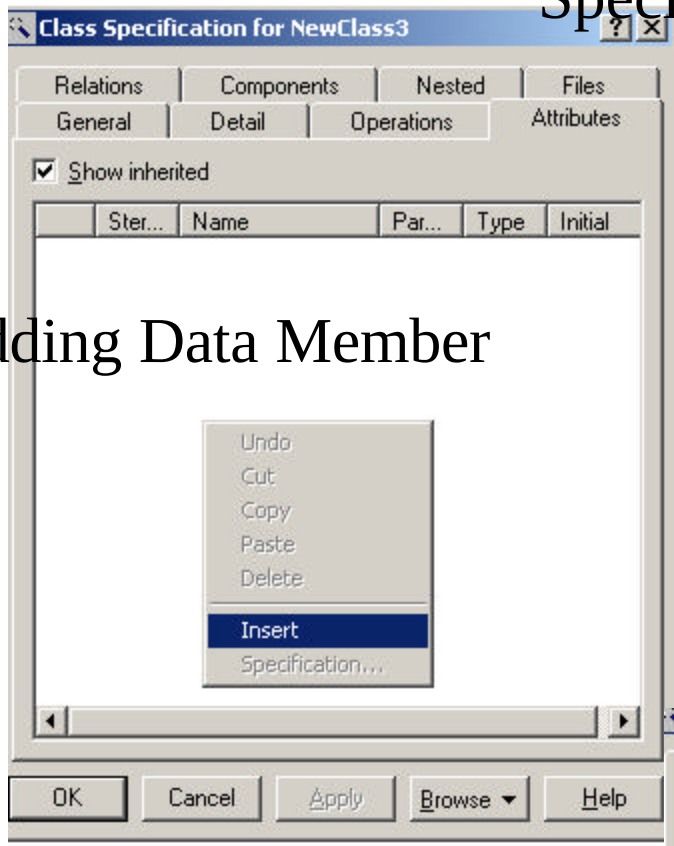


Generic method detail window

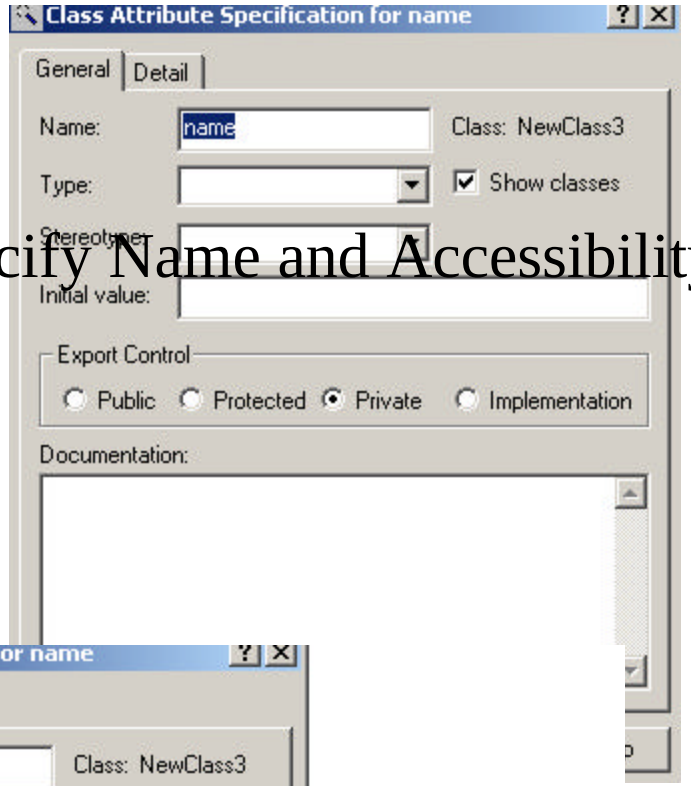


Defining parameters

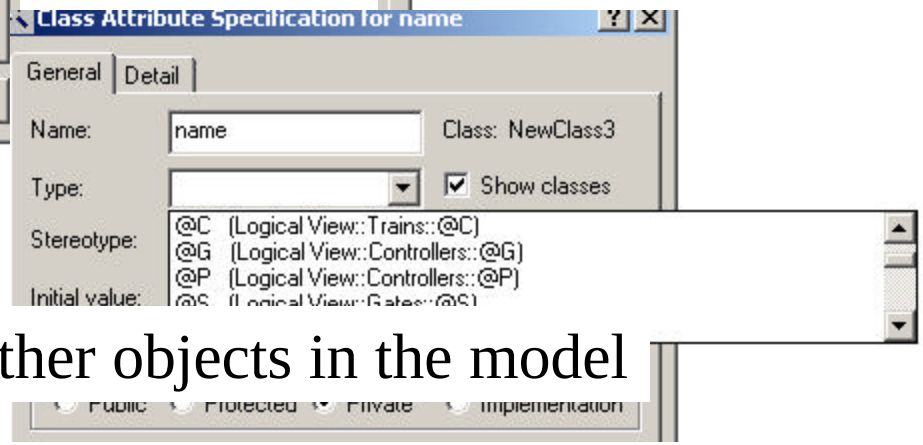
Specifying Data Members



Adding Data Member

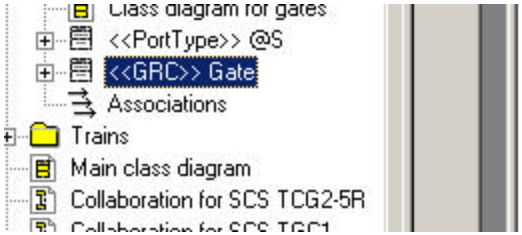


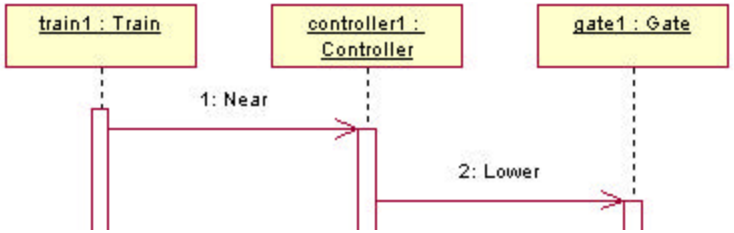
Specify Name and Accessibility



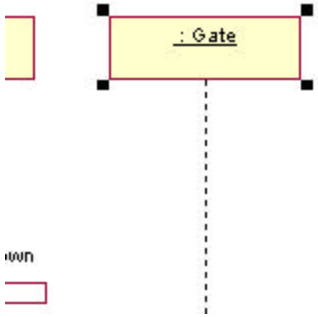
Reference Directly other objects in the model

How to draw a Sequence Diagram

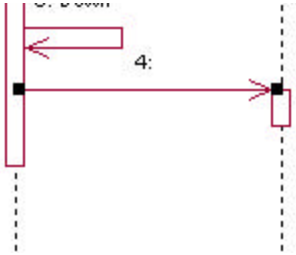




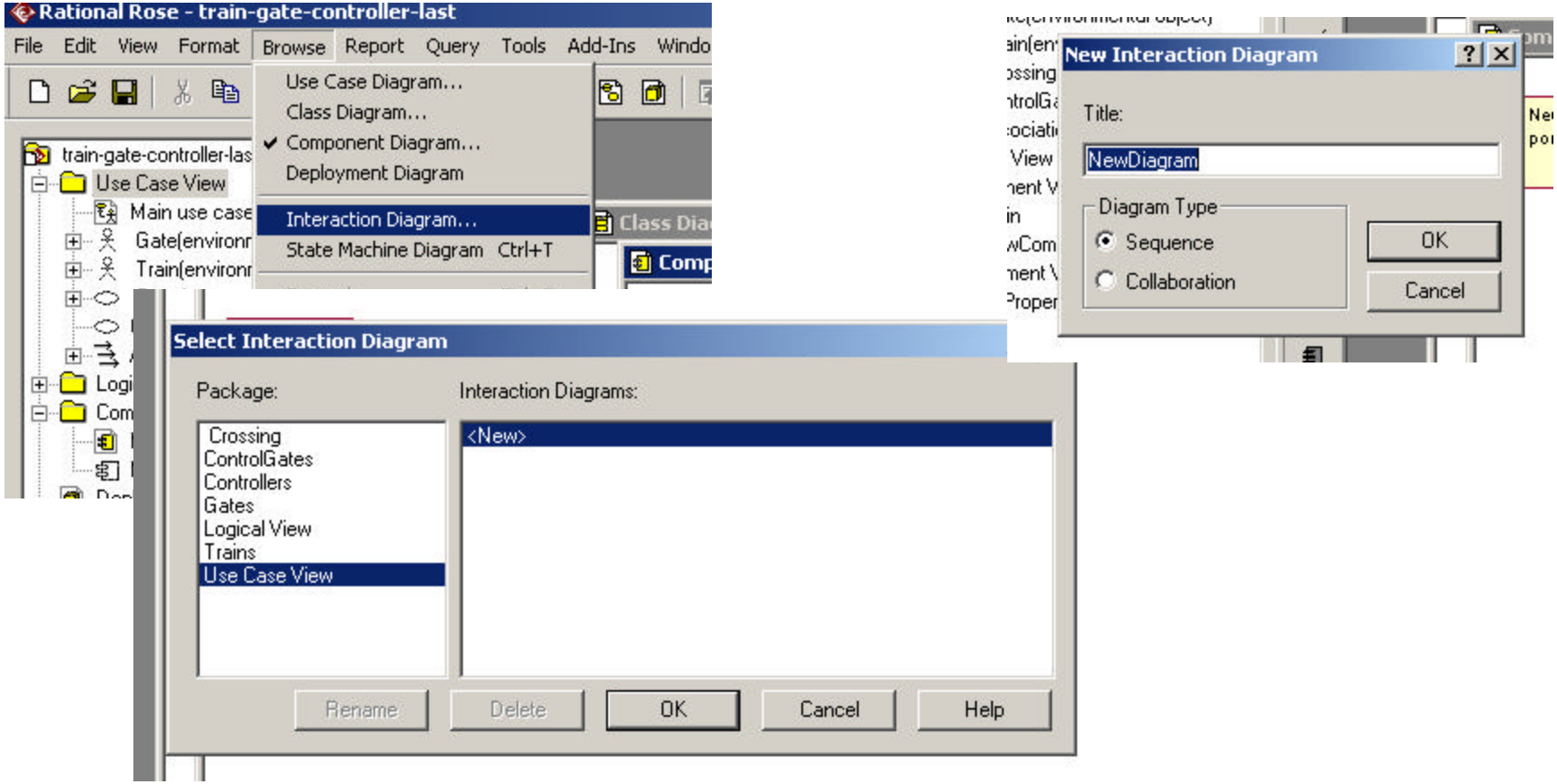
Drag-and-drop objects



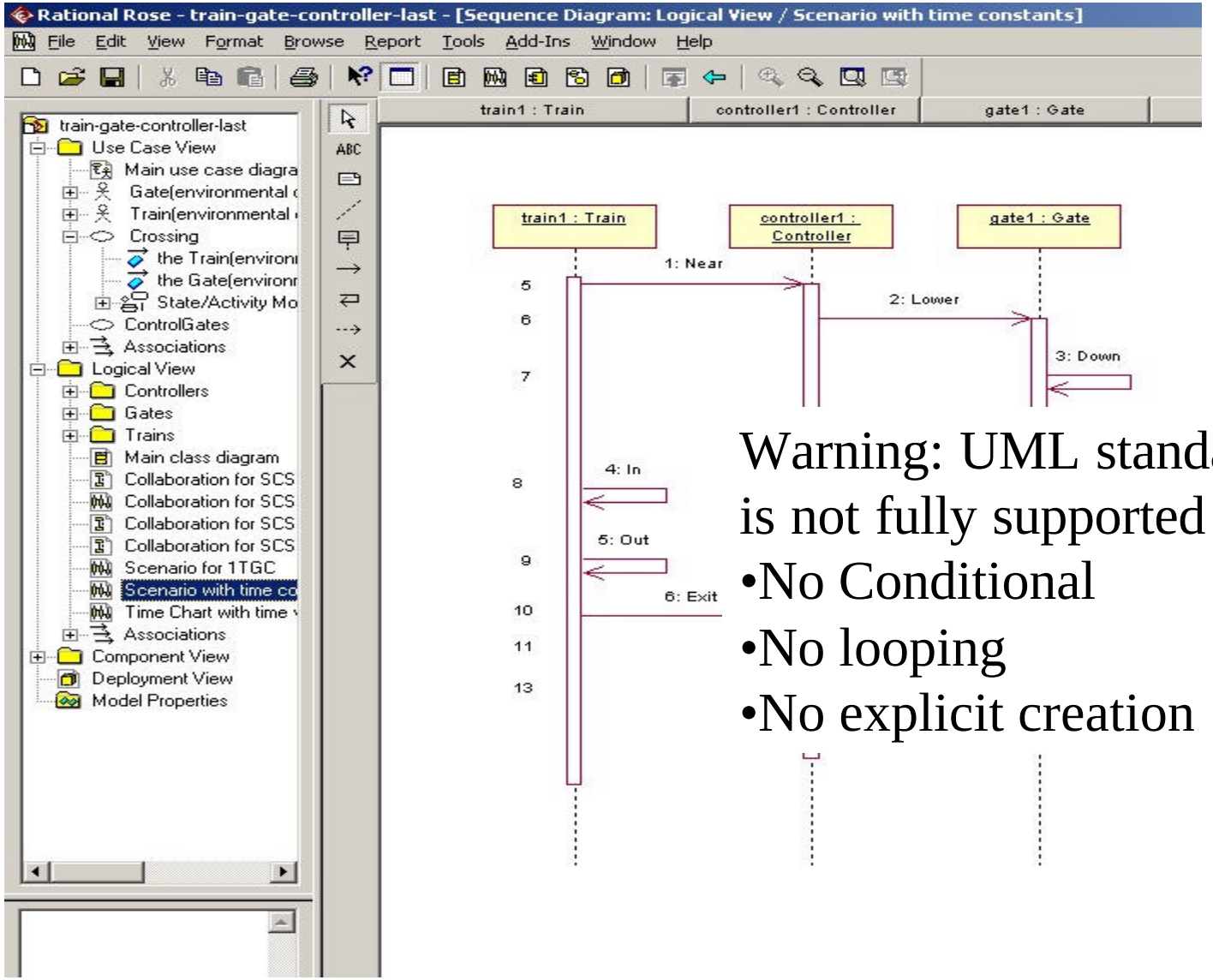
Add the arrows



Creating a New Sequence Diagrams



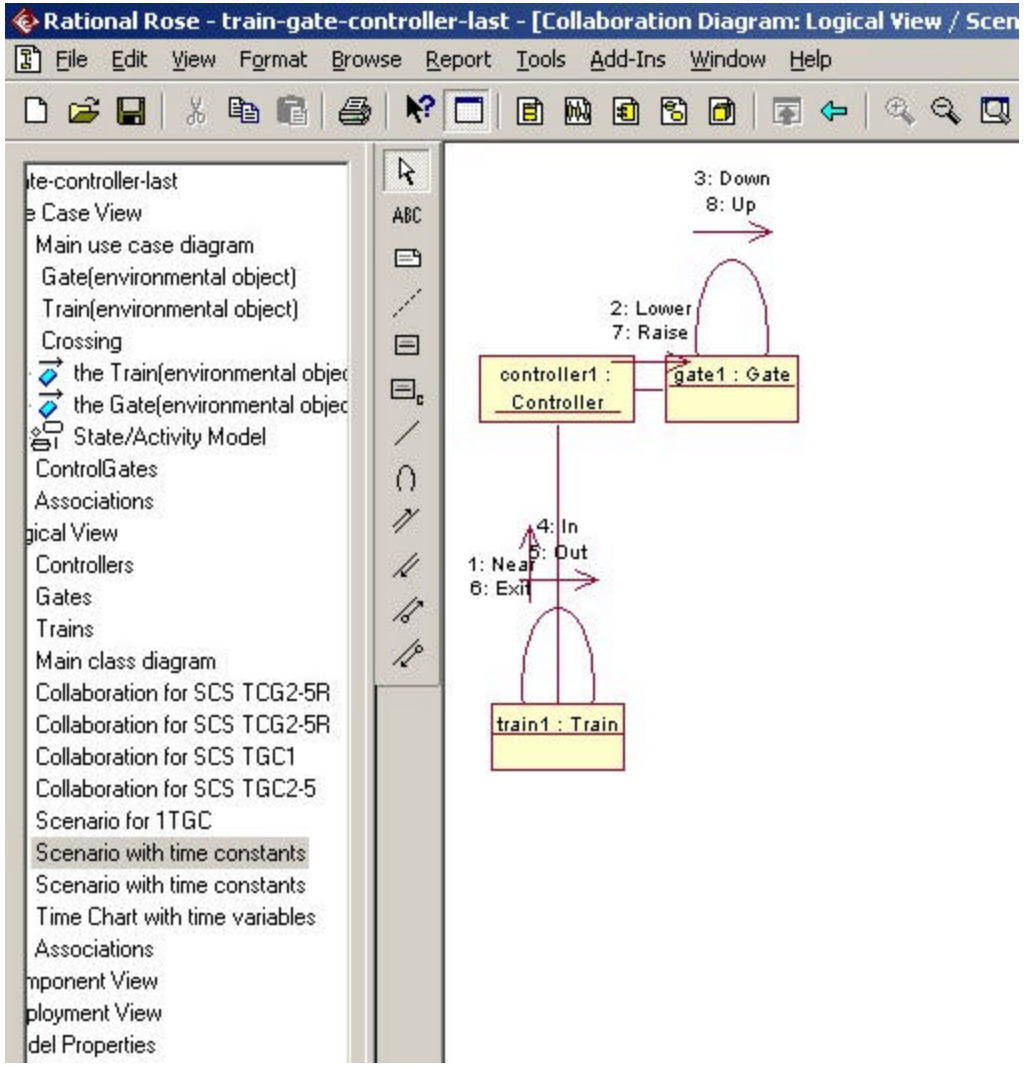
Sequence Diagrams



Warning: UML standard is not fully supported

- No Conditional
- No looping
- No explicit creation

Conversion of Collaboration Diagrams



Press F5 from the sequence diagram to obtain the corresponding diagram

Quirks & Perks to Know

Bad Things:

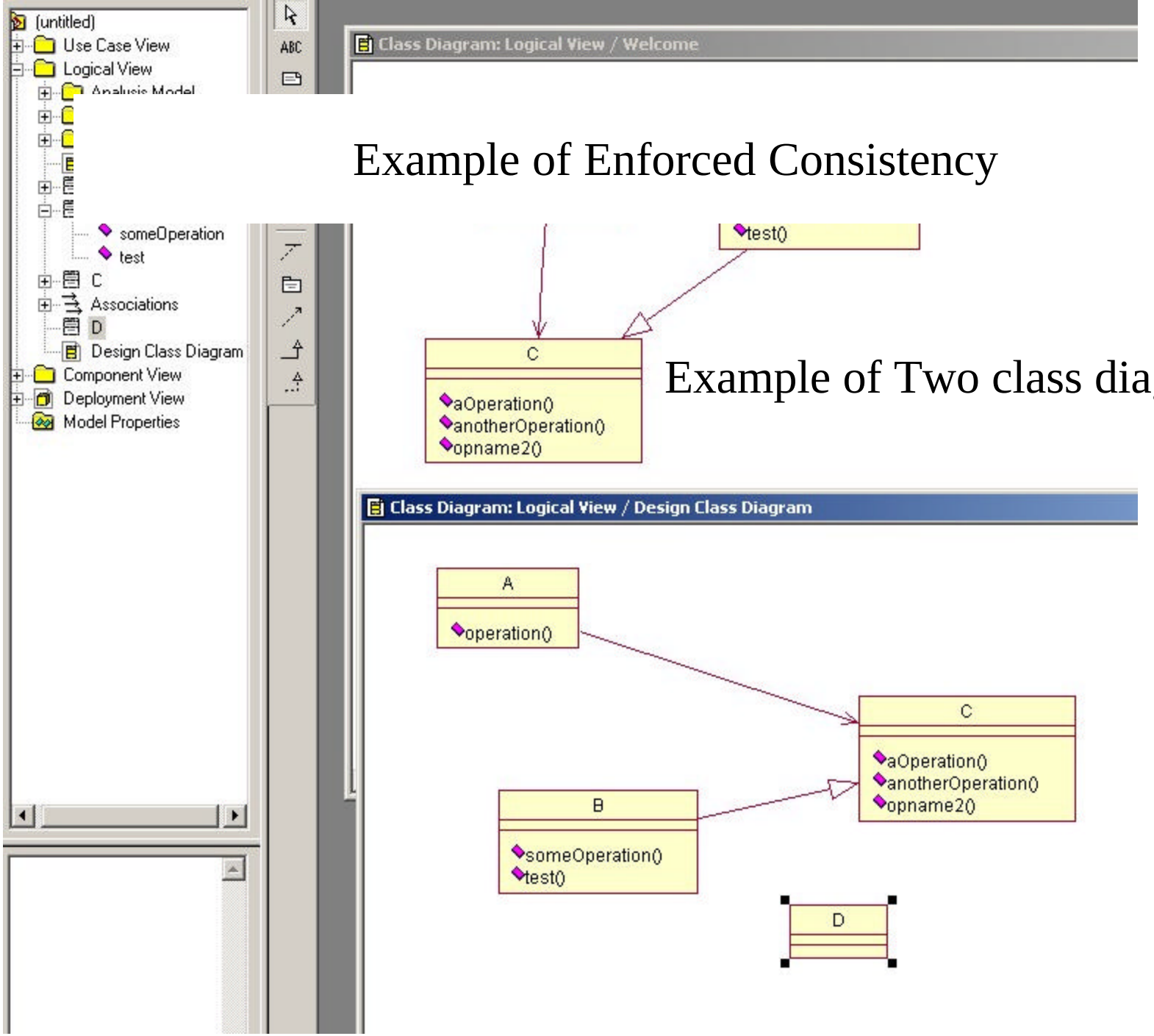
- Hitting ‘del’ only removes an object from the model. To really delete, use ctrl+d.
- You can only remove arrows with ctrl+d in sequence diagrams
- Classes are not automatically updated when adding operations in sequence diagrams
- Save often and make sure the saves are made. It does happen that rational won’t save changes…
- The labs are not set up for collaborative work, nor for report generation and other advanced features

Quirks & Perks to Know

Good things:

- You don’t have to double-click on an object all the time in order to edit its properties. You can single-click on it and you can edit textually.
- By using multiple representations of the same entity, you ensure consistency throughout the design
- Copy-paste between Rational Rose documents work well, and you can merge the work of two documents that way.

Example of Enforced Consistency



Example of Two class diagrams