Rational Rose Tutorial

Computer Systems Technology

BCIT

Overview

- CASE
- ROSE Concepts
- Class diagrams
- Sequence diagrams
- Managing model complexity
- Document generation
- Code generation

CASE

- How do we keep models and code consistent?
- Need for Computer Aided Software Engineering (CASE)
  - A CASE tool supports parts or all activities of the software lifecycle
- Front-end CASE tools:
  - Tools that create only object models
- Back-end CASE tools
  - Netbeans, Borland Jbuilder, IBM VisualAge are development environments
- Documentation is usually supported by separate tools
  - Microsoft Word, PowerPoint, ...
- Lifecycle CASE tool: Supports analysis, design, implementation and documentation
  - Rational Rose Software

ROSE Concepts

- ROSE enables the user to:
  - construct an object model of a system
  - draw diagrams representing different views of the model
  - generate code (example Java)
- ROSE distinguishes between an entity *(model element)* and its representation *(icon*):
  - there can be only one model element (e.g., a class) with a given name
  - a model element (e.g., class Customer) can be represented by many icons in many different diagrams
  - Deleting an icon from a diagram does not always delete the model element

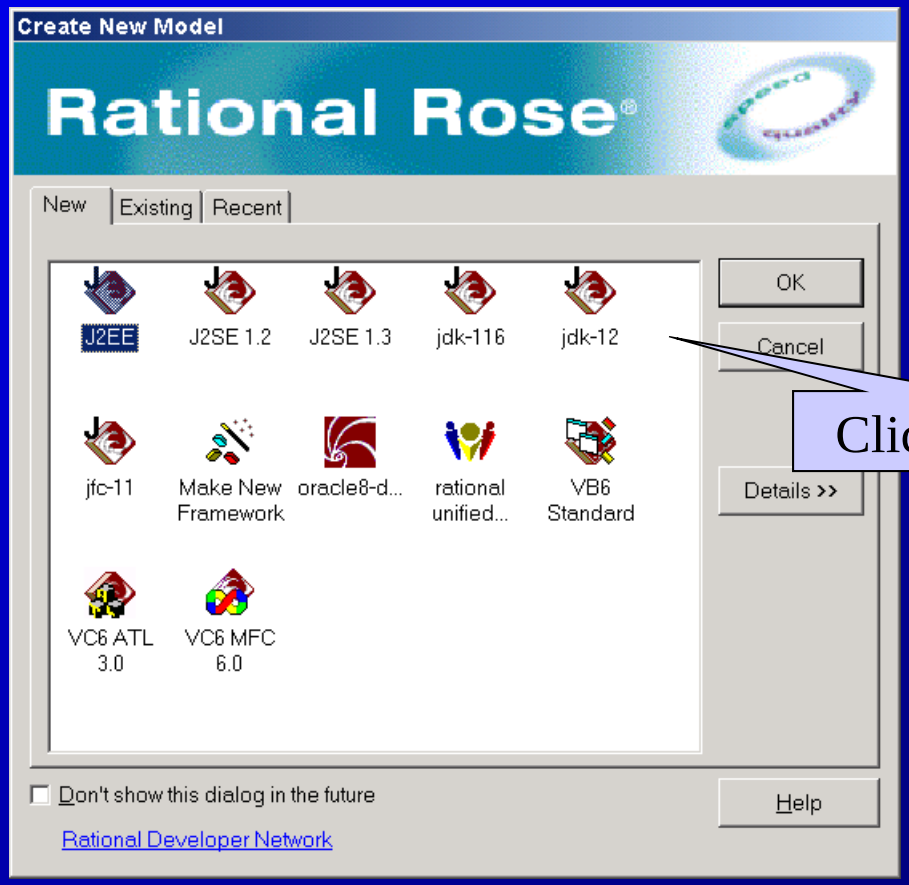
Create New Model screen

Step 1

Start Rational Rose

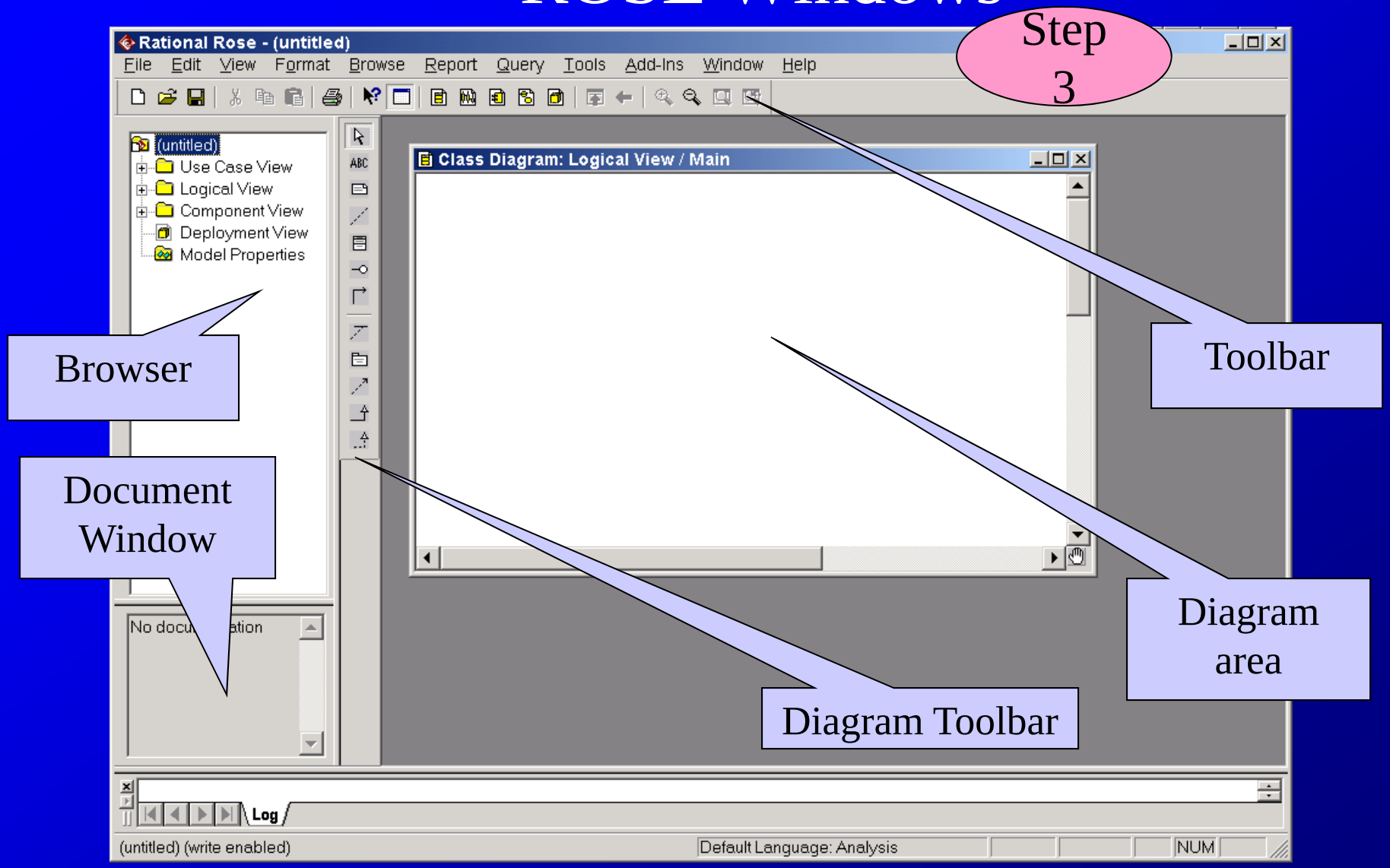
Step 2

Start a new Model



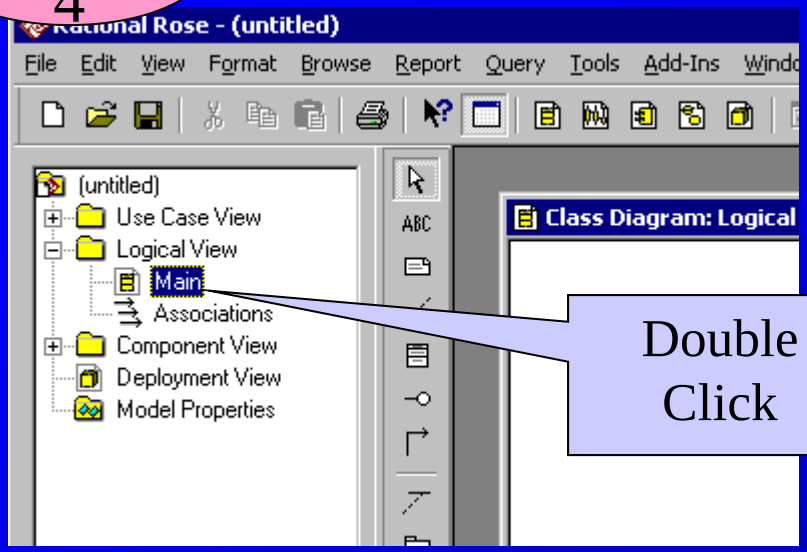
Click on cancel

ROSE Windows



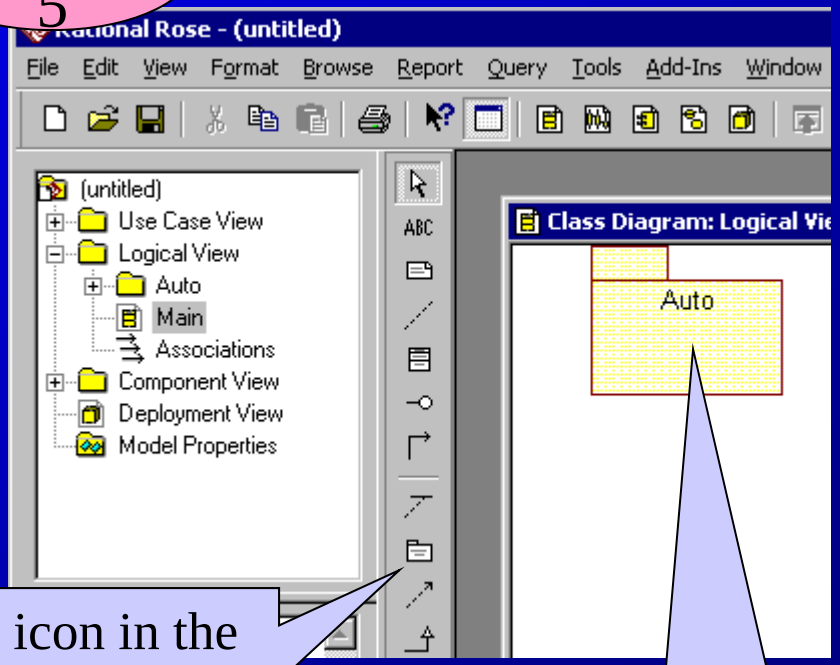
Creating a package

Step 4



Double Click

Step 5



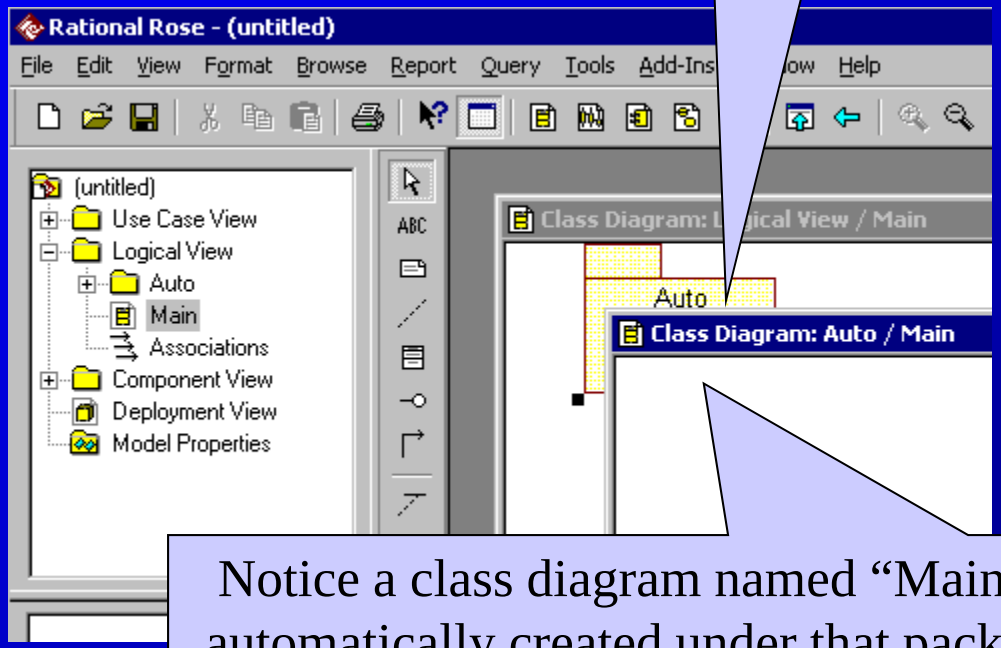
Click on the package icon in the toolbar and then click on the diagram

Rename the package to “Auto”

Creating classes belonging to a package

Step 6

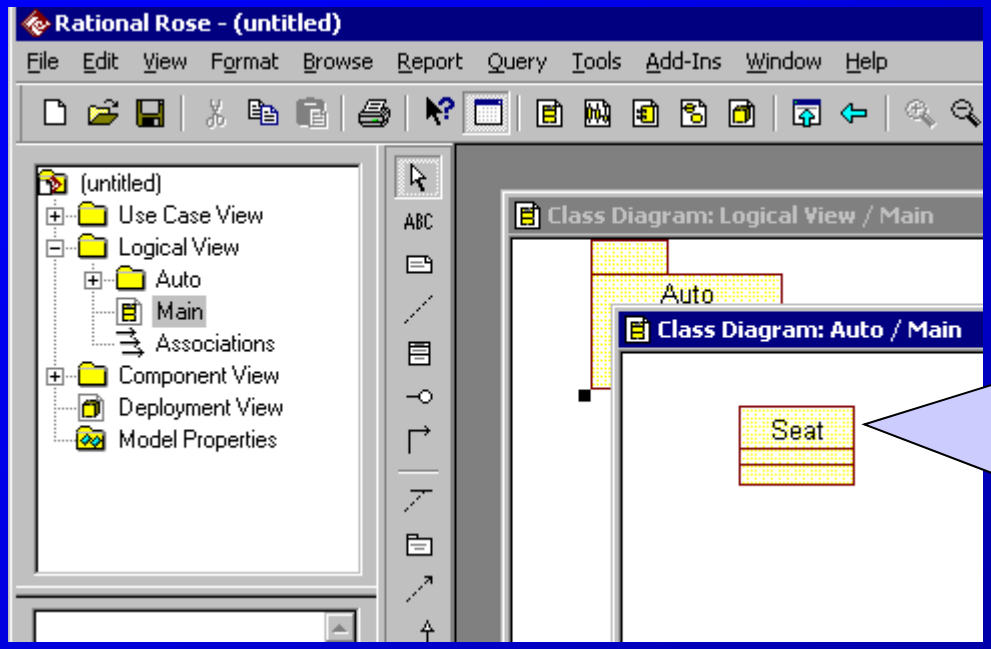
Double-click on the "Auto" package.



Notice a class diagram named "Main" is automatically created under that package.

Creating a new class

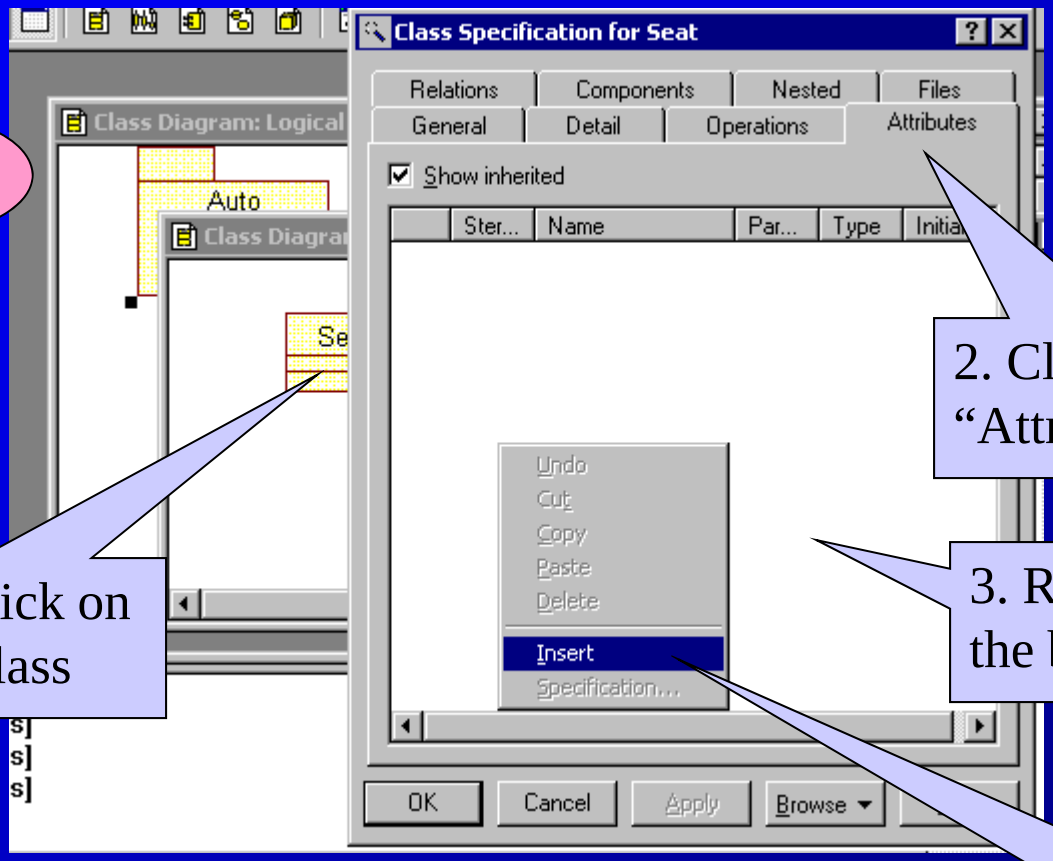
Step 7



- Click on the class tool in the diagram toolbar
- Click on the class diagram
- Type the name of the class (e.g., “Seat”)
- Resize the class symbol

Adding Attributes

Step 8



1. Double click on the “Seat” class

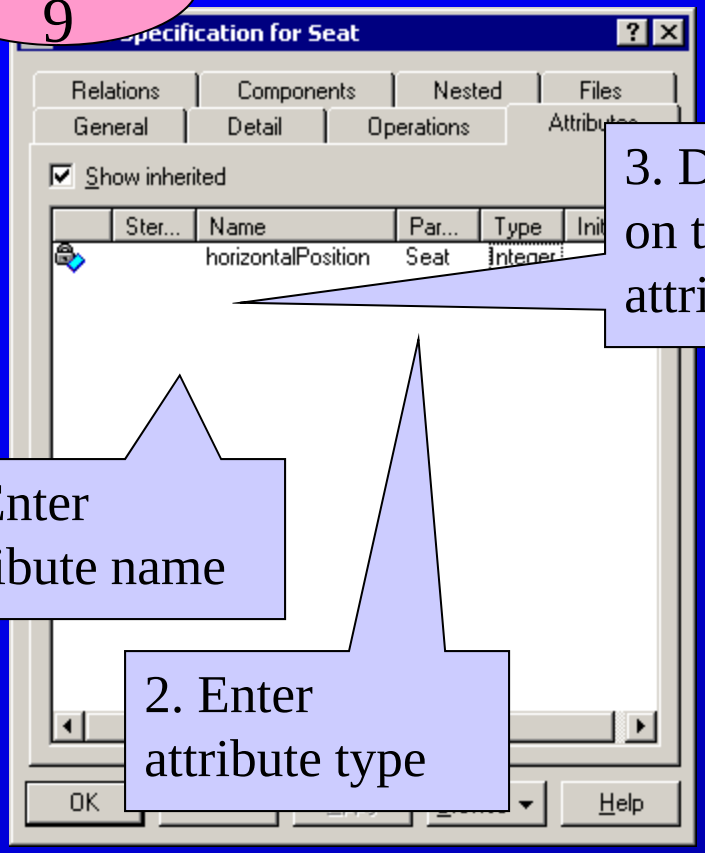
2. Click on the “Attributes” tab

3. Right click on the blank area

4. Select “Insert” menu item

Add attribute (continued)

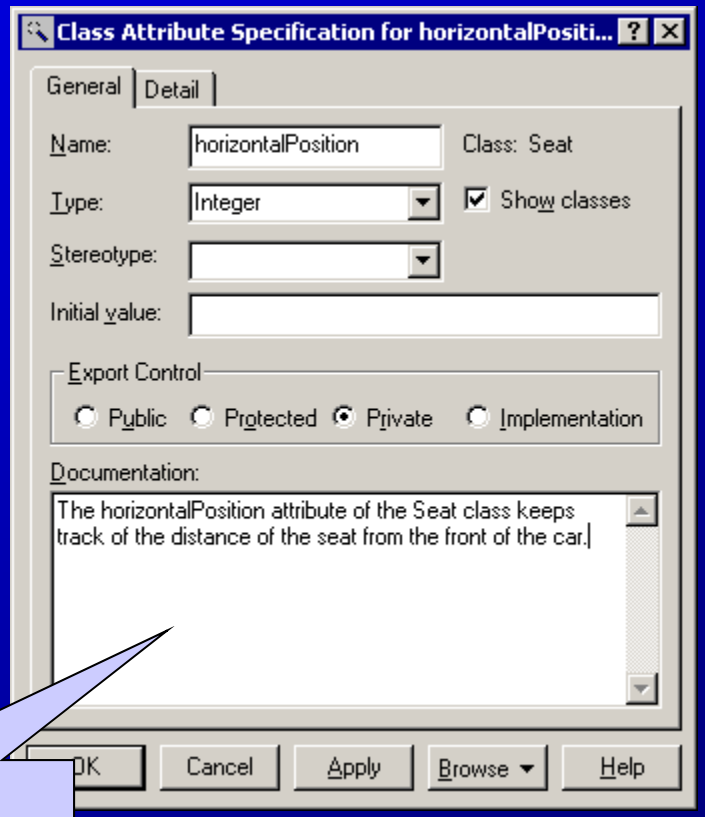
Step 9



1. Enter attribute name

2. Enter attribute type

3. Double click on the new attribute

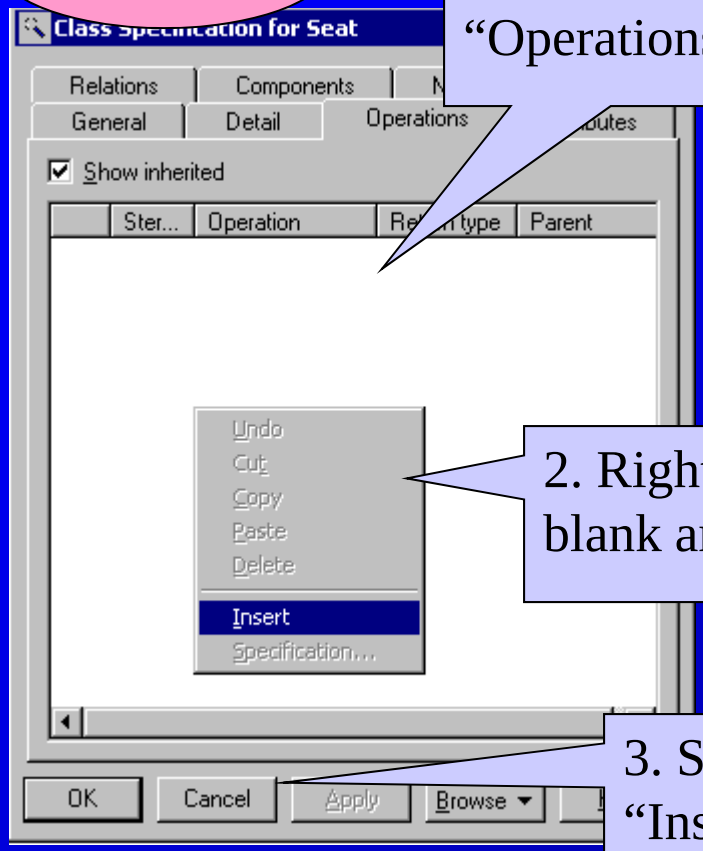


Type attribute documentation then click on "OK"

Add operation

Step 10

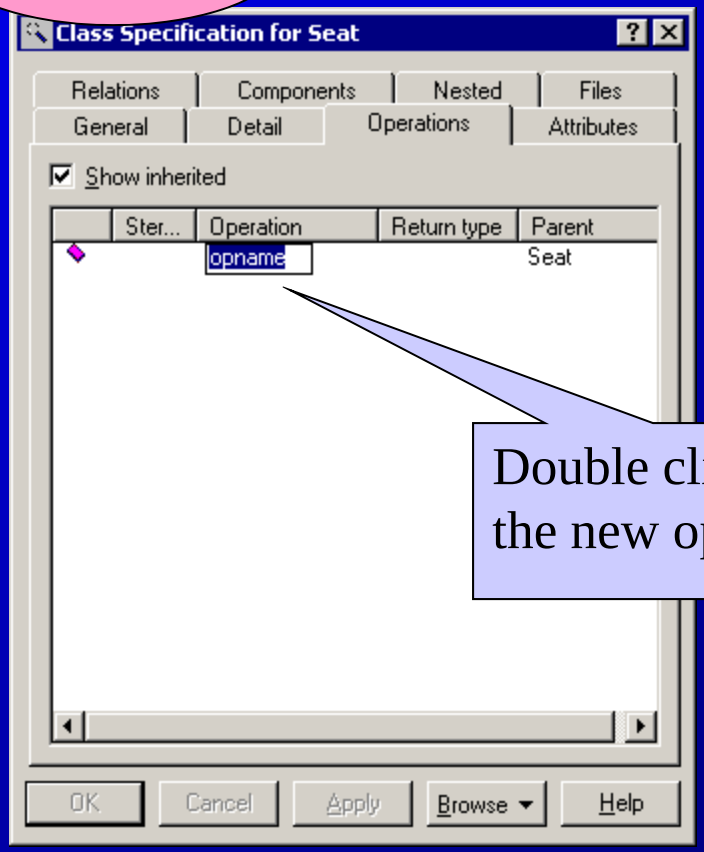
1. Click on the "Operations" tab



2. Right click on blank area

3. Select "Insert"

Step 11

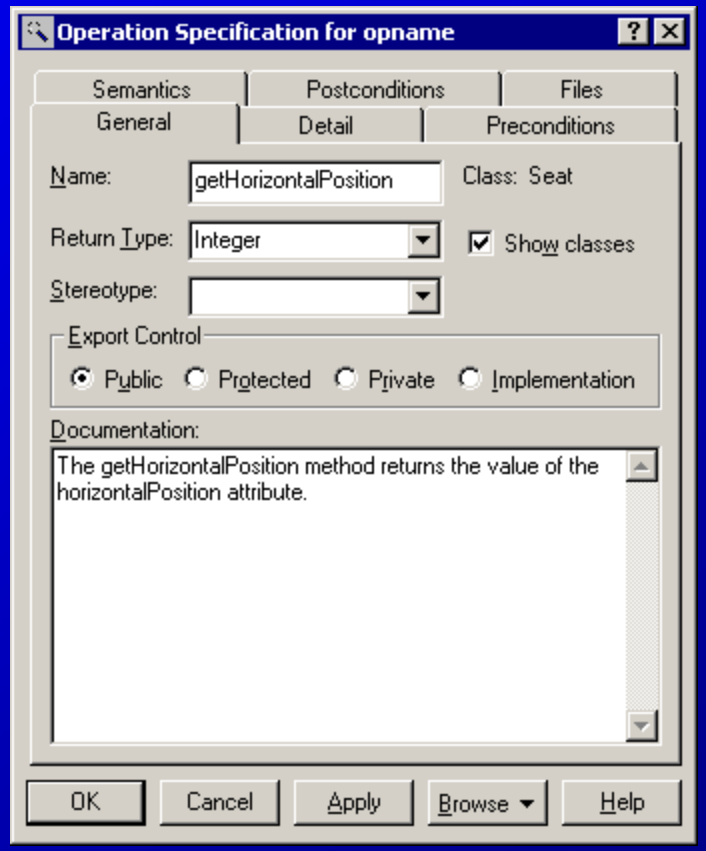


Double click on the new operation

Add operation (continued)

Step 12

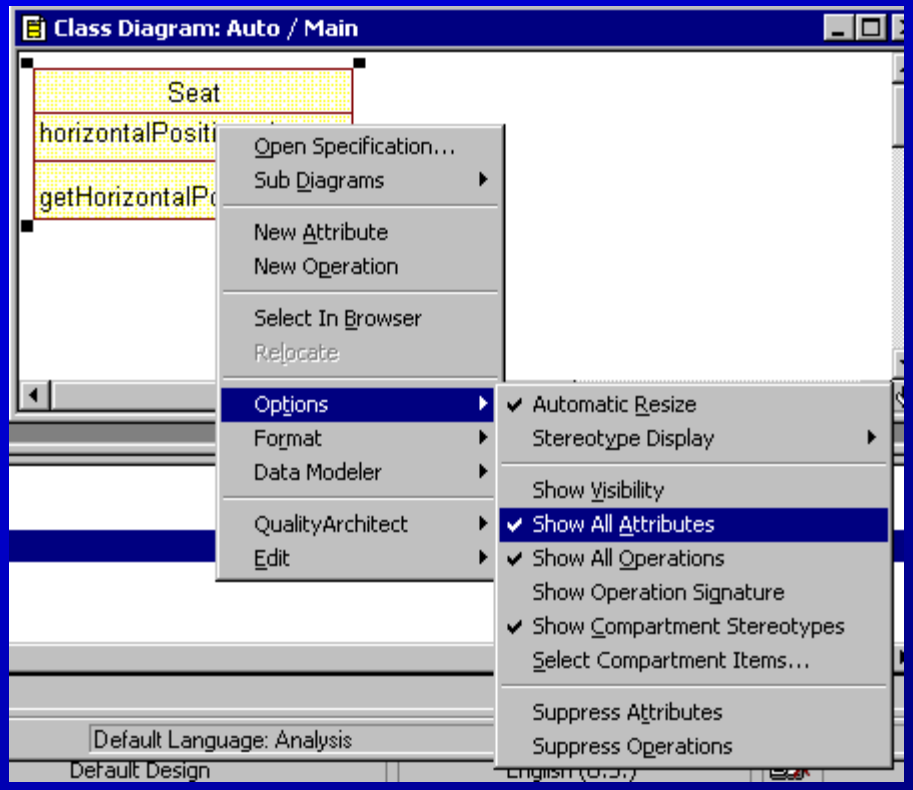
- Type its signature, return type, and documentation,
- Click “OK”
- Click “OK” one more time to return to the class diagram



Showing and hiding properties

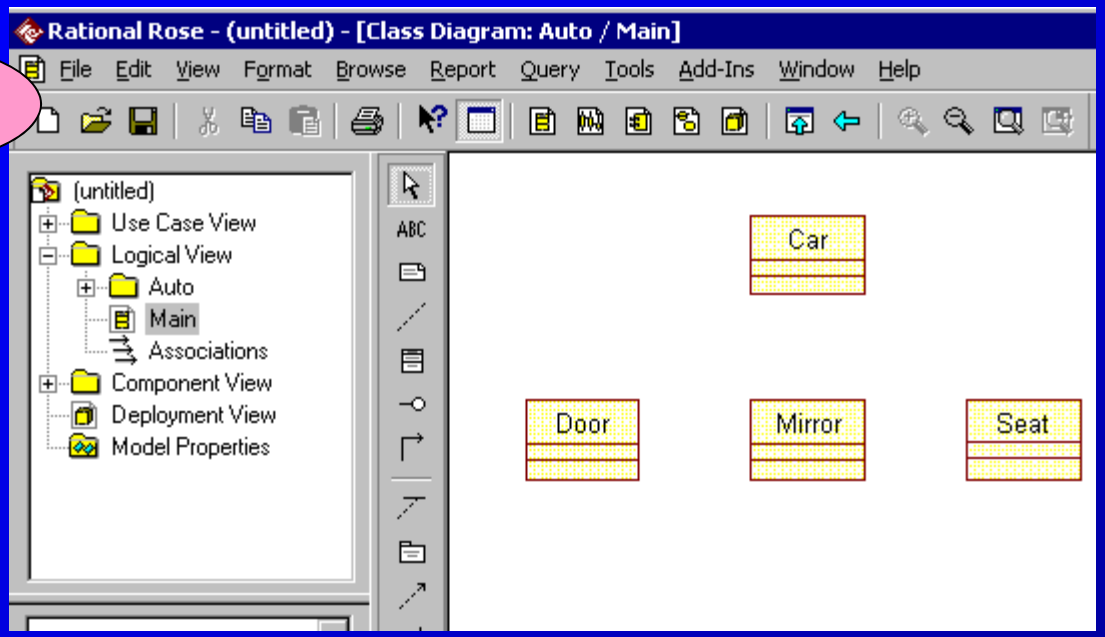
Step 13

- Right click on the “Seat” class and select “Options”
- Disable
  - “Show All Attributes”
  - “Show All Operations”
  - “Show Visibility”
- The default values for these check marks can be set on the “Diagram” tab by selecting “Options” from the “Tools” menu.



Creating More Classes

Step 14

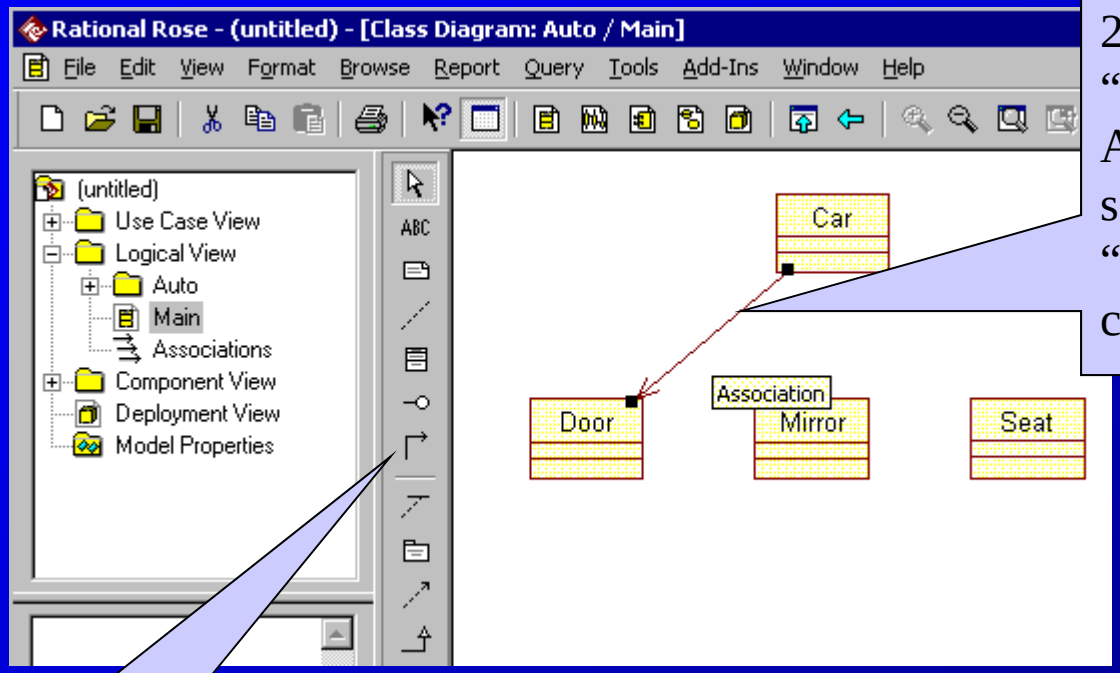


Create the additional classes:

- Car
- Door
- Mirror

Creating associations

Step 15



2. Drag the "Unidirectional Association" from source class (e.g. "Door") to the target class (e.g. "Car")

1. Click on the "Unidirectional Association" tool.

Creating associations (continued)

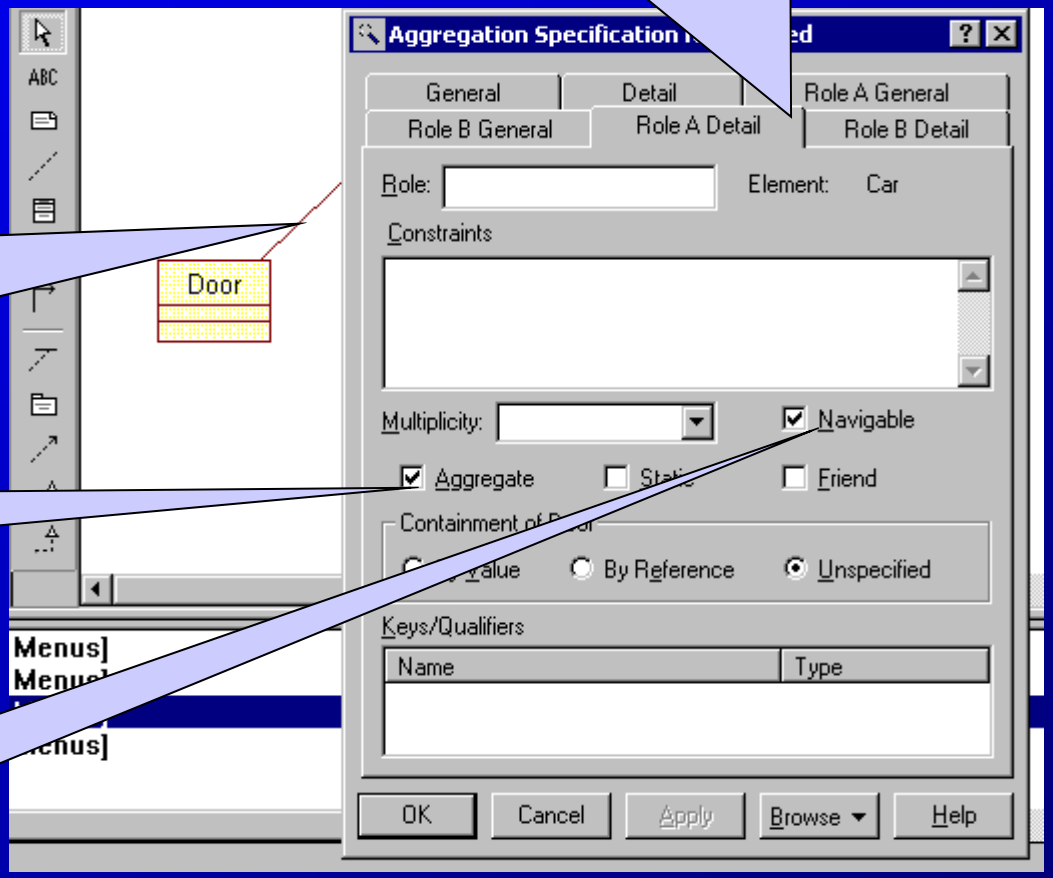
Step 16

Role A Detail and Role B Detail are important to set the shape of the arrow and the direction of it.

Double click on the association to change its attributes
- name
- roles
- cardinality

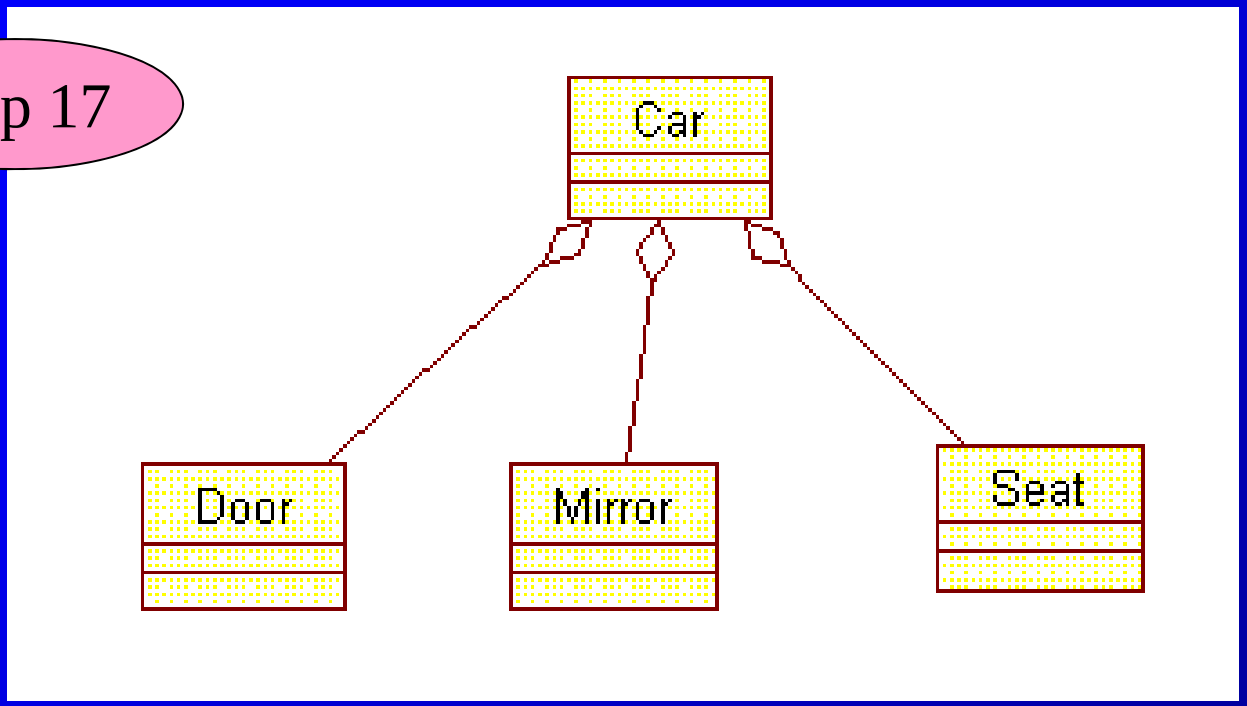
Change the head of the arrow to diamond

If it is navigable has an arrow head if it is not the arrow head is removed



Creating associations (continued)

Step 17

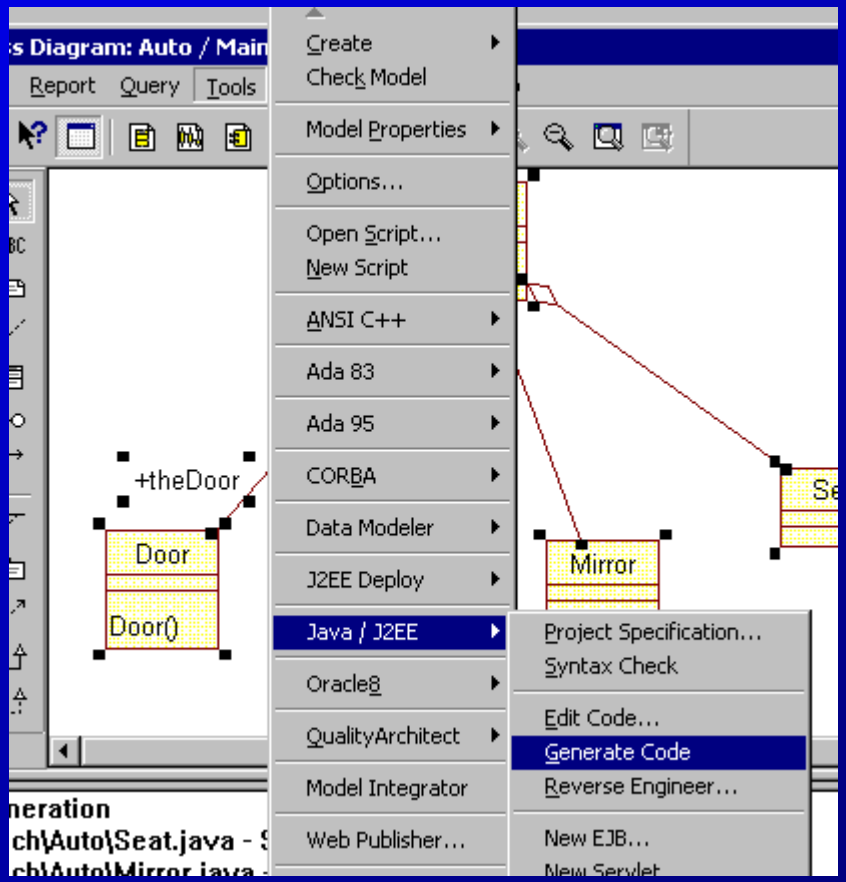


Complete the above associations by repeating steps 15 and 16.

Code Generation with ROSE: Generating Java code

Step 18

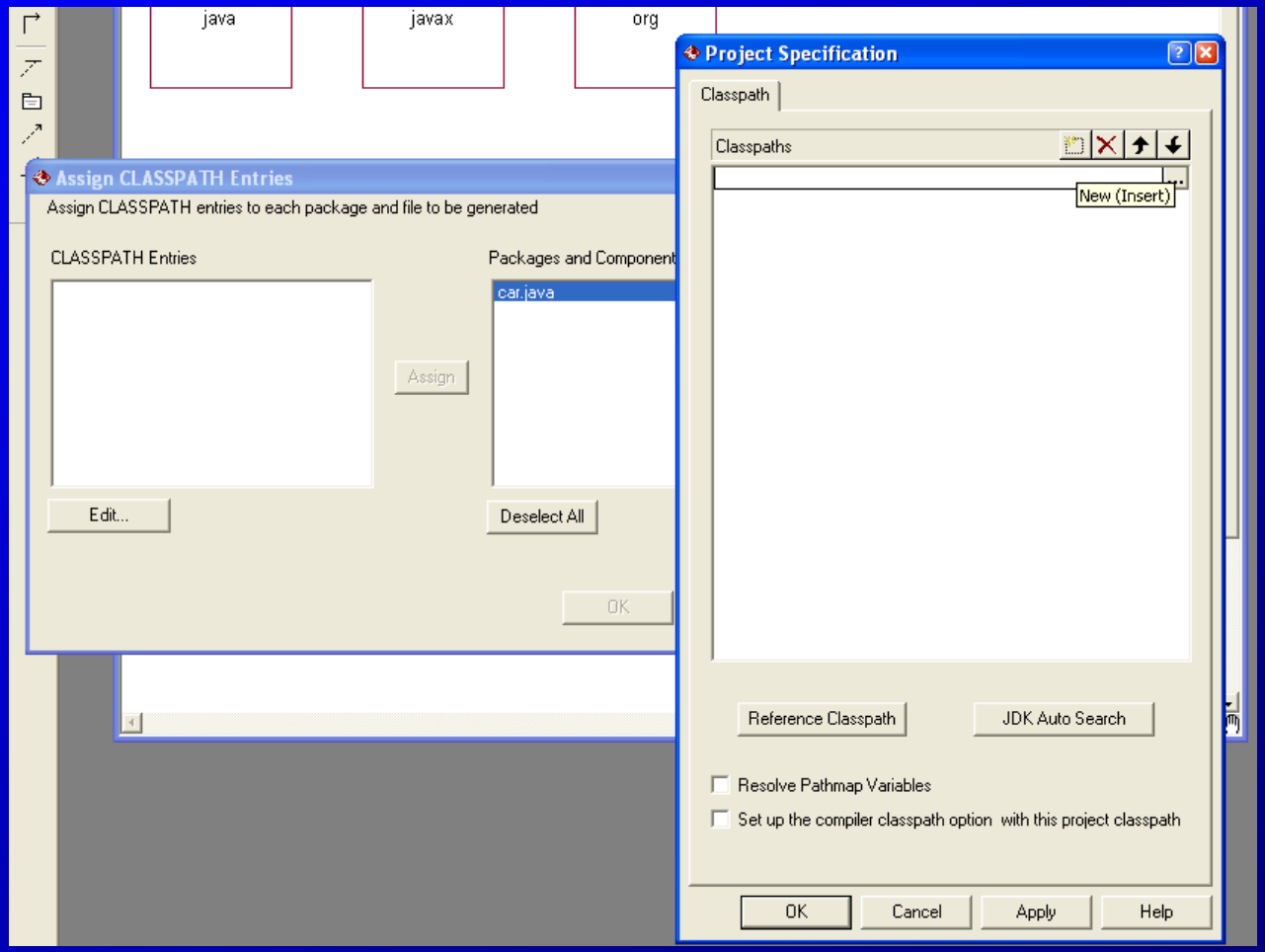
- Select the classes whose code is to be generated
- Select “Generate Code” from “Tools / Java-J2EE”



Code Generation with ROSE: Generating Java code

Step 19

- Assign a “Classpath” which essentially means a folder on your hard drive that will contain the Java classes
- Select “Edit Code” to view the Java code, after is generated.
  - Hint: do not modify the comments generated by ROSE (e.g., //##begin ...)



Typical process for using Rational Rose

- Create actors and use cases, including their brief descriptions
- Draw use case diagrams
- Create a class diagram per use case for displaying its participating objects.
  - Create or include the classes
  - Create any additional relationships
- Create a sequence diagram per use case
  - Include each participating object
  - Create messages
  - Create operations
  - Bind messages and operations
- Organize classes into packages
- Generate code and documentation

Things to remember

- In Rose, names are unique across the model, for example:
  - An actor and a class cannot have the same name
  - A use case and a class cannot have the same name
- Most text strings can be edited by double clicking on them
- Specifications (properties of class, use case, actor) can be displayed by double clicking on the corresponding icon
- Items can be inserted and removed from list by right clicking on the list. This applies to
  - Attributes and operations
  - Diagrams associated with a use case
- On-line help provides more information than the manual
- ***Backup your files, PCs crash frequently***