Using Caché RoseLink

Version 5.0.17
30 June 2005

InterSystems Corporation 1 Memorial Drive Cambridge MA 02142 www.intersystems.com

Using Caché RoseLink
Caché Version 5.0.17 30 June 2005

This book was assembled and formatted in Adobe Page Description Format (PDF) using tools and information from the following sources: Sun Microsystems, RenderX, Inc., Adobe Systems, and the World Wide Web Consortium at www.w3c.org. The primary document development tools were special-purpose XML-processing applications built by InterSystems using Caché and Java.

For Support questions about any InterSystems products, contact:

InterSystems Worldwide Customer Support

Tel:	+1 617 621-0700
Fax:	+1 617 374-9391
Email:	support@InterSystems.com

Table of Contents

List of Figures

1

Introduction

Caché RoseLink provides a connection between Caché and the Rational Rose object modeling tool. Specifically, RoseLink is an add-in application that you can invoke from within the Rational Rose application that communicates with a Caché database.

Using RoseLink you can:

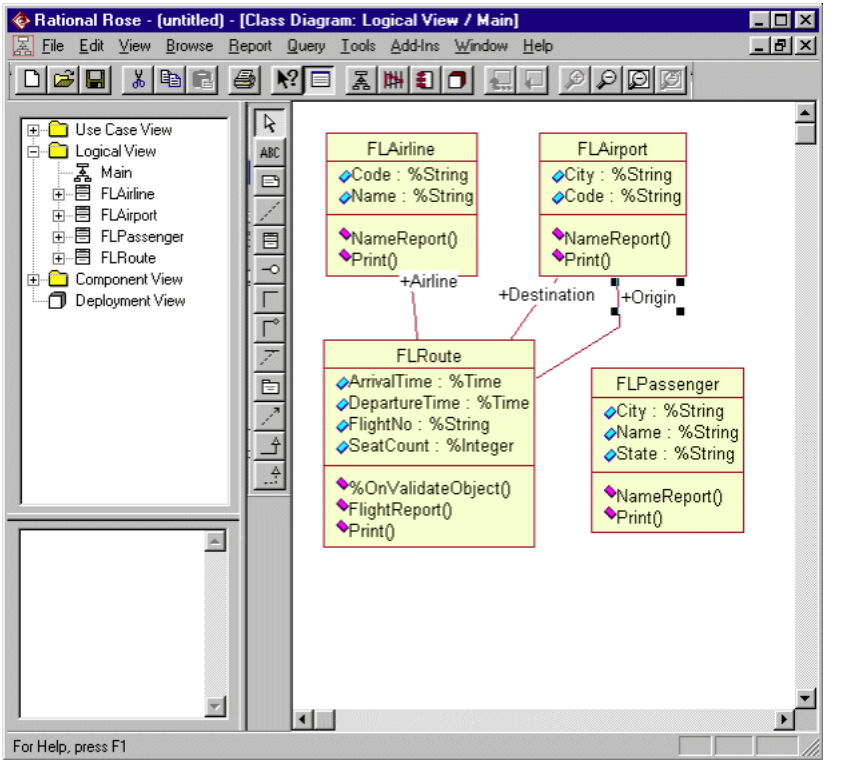
- Create an object model in Rational Rose and then export it to Caché, creating an equivalent set of class definitions in the Caché dictionary. Through this process, you can turn the specification developed in Rose into a working application in Caché. For instance, while Rose enables you to describe a method (defining its signature, for instance) Caché enables you to implement that method by providing its code.
- Import a set of Caché class definitions into Rose. This enables you to graphically document an application.

These steps may be combined, in a process called "round trip engineering" . For instance, you might create an initial definition in Rose, export it to Caché, enhance the definition in Caché, and then re-import it into Rose.

1.1 A Quick Tour

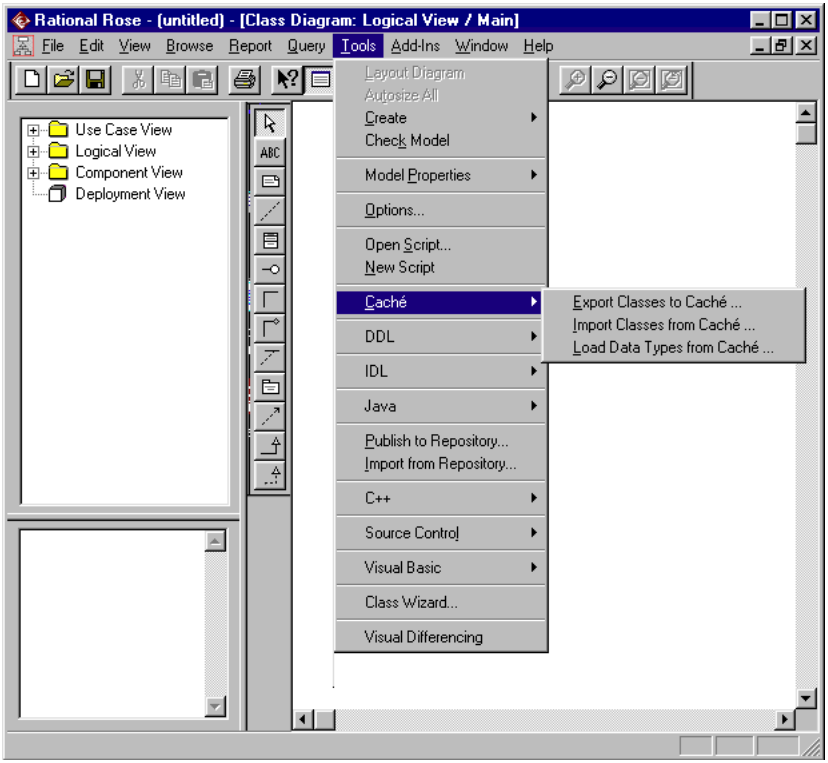
Suppose that we want to export to Caché the simple Rose model shown below. It shows four classes as well as some relationships among them:

Sample Rational Rose Model



You can invoke Caché RoseLink using the **Tools** menu in Rose:

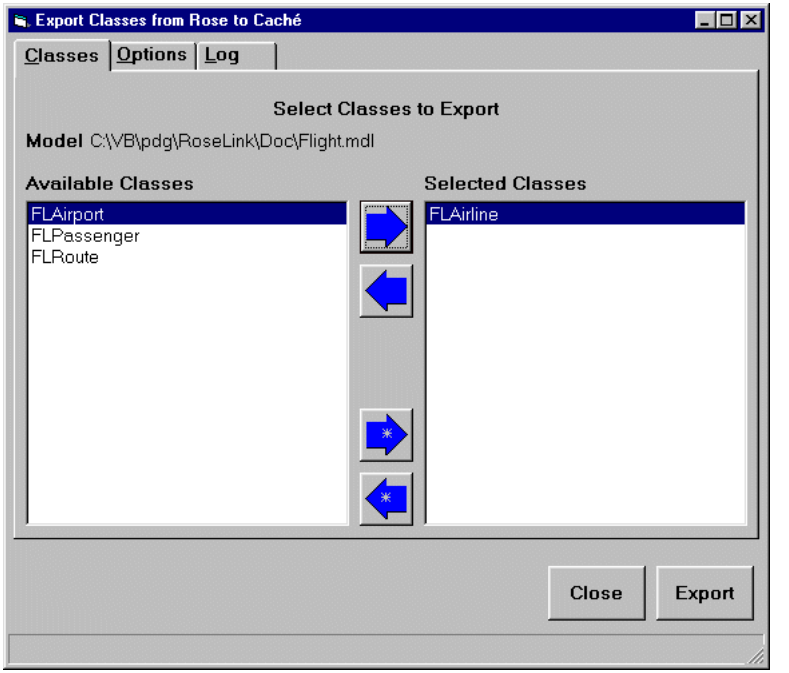
Rational Rose Tools Menu



The **Export Class to Caché** command displays a window that enables you to select the Rose classes (from the currently active diagram) that you want to export. (See figure below.) The **Available Classes** list shows the Rose classes that can be (but have not yet been selected to be) exported to Caché. You can select a class by highlighting it and then clicking on the right arrow button, which causes the class name to move from the **Available Classes** list to the **Selected Classes** list.

In the following figure, one class (FLAirline) has been selected for export and three more are available. When you press the **Export** button, RoseLink enables you to select a Caché server and namespace to connect to and then exports the selected classes.

Export Classes Dialog



When this window appears, any classes that were selected in the Rose class diagram are automatically shown in the **Selected Classes** list. If you do not want to export these classes, they can be moved back to the **Available Classes** list with the left-facing arrows.

The import process is similar: from a list of available classes stored within Caché you select which classes you wish to import.

2

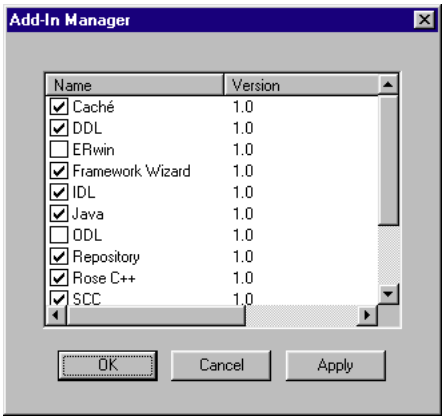
Installing RoseLink

You must have a copy of Rational Rose in order to use RoseLink.

To install RoseLink, perform the following steps:

1. Stop Rational Rose if it is running.
2. Copy the CacheRoseLink.exe file to a directory (any will do) and then execute it. This causes the required registry entries to be created and a menu file (CacheRoseLink.mnu) and property file (CacheRoseLink.pty) to be generated.
3. Start Rose and select **Add-In Manager** from the **Add-Ins** menu. Find Caché in the list, select it, and click on **OK**. (See figure below.) The Caché item will now appear in the **Tools** menu.

Rational Rose Add-in Manager



3

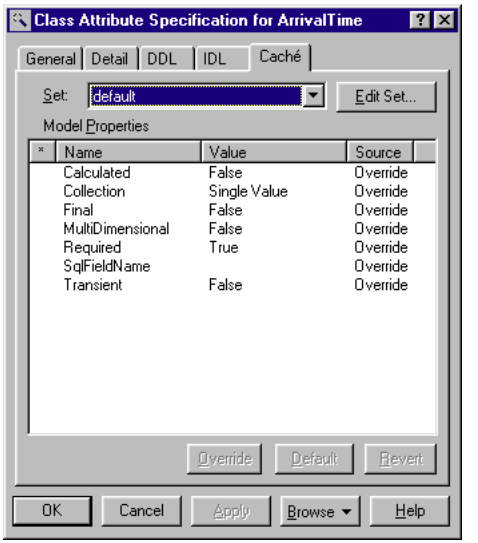
Using RoseLink

Once you have installed RoseLink, you can use it from Rational Rose.

3.1 Caché Property Pages

When Caché RoseLink is active (i.e. the Caché entry is checked in the Rose Add-In Manager), Caché-specific property pages are added to the Rose environment. For instance, the following figure shows the Caché property page for attributes. It enables, for example, specification of the collection type (single value, list, and array) of an attribute.

Caché Property Page



In addition to attributes, Caché property pages are provided for Rose classes, operations, and associations.

3.2 Exporting Classes to Caché

When you export a Rose class to Caché:

- A Caché class is created and its Name, Description, Persistent, Super, and Abstract keywords are set from the equivalent Rose characteristics.
- A Caché property is created for each Rose attribute. The property's Name, Description, Static, and Private keywords are set from the equivalent Rose characteristics. The property's type is set as follows:
  - If the type set in Rose is a valid Caché type (i.e., there is a class with that name in the target Caché namespace), then the Caché type for the new property is the same as the one specified in Rose.
  - If the type set in Rose is one of the fundamental Rose types, then it is converted to the equivalent Caché type, as shown in the following table:

Rose Type	Caché Type
Boolean	%Boolean
Byte	%Integer
Currency	%Currency
Date	%Date
Double	%Float
Integer	%Integer
Long	%Integer
Single	%Float
String	%String

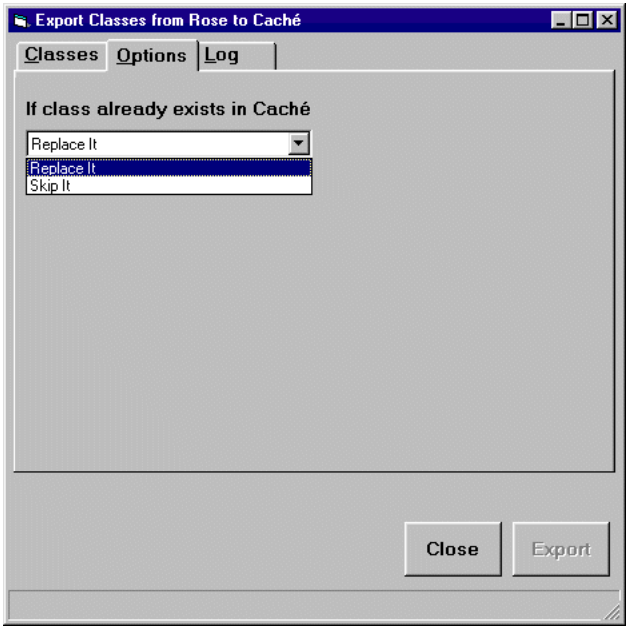
  - Otherwise, the Rose type is used for the property, although this will prevent the class from being compiled.

- A Caché method is created for each Rose operation. The method's Name, Description, ReturnType, FormalSpec, and Private keywords are set from the equivalent Rose characteristics. Note that type specifications within the ReturnType and FormalSpec are handled as described above for properties.
- One or two Caché reference properties are created for each Rose association. The role name in Rose is used as the property name in Caché. For instance, Figure 1 shows two associations connecting the FLRoute and FLAirport classes. The role names (at the FLAirport end) are Destination and Origin. This gives rise to two properties, with these names, in the FLRoute class. If these associations had role names at the other end, other reference properties would be created in the FLRoute class.
- The class hierarchy defined in Rose is created in Caché. If the class is persistent, %Persistent is automatically added to the super class list.

If errors occur when creating Caché properties or methods, the Caché class is created without the erroneous items, which are noted in the export log.

What happens when a class exists in both Caché and Rose? For both importing and exporting, the result depends on the setting chosen on the Options tab. (See Figure below.) The "Replace It" option, which is the default, causes the existing class definition to be overwritten in its entirety. If "Skip It" is selected, the existing class definition is untouched.

Options Tab



3.3 Importing Classes from Caché

When classes are imported from Caché into Rose, the entire class definition is transferred. That is, if you import a Caché class into Rose, delete it from Caché, and then export the class to Caché, the *entire* Caché definition is restored.

Characteristics of the Caché class definition fall into three categories:

- Caché characteristics that have a Rose equivalent. This is the same list of characteristics that is exported.
- Caché characteristics that have no Rose equivalent but can nonetheless be edited via Rose using Caché property pages that are added by the Caché RoseLink to the Rose modeling environment.
- Caché characteristics that have no Rose equivalent and cannot be edited via Rose. These characteristics are "hidden" within the Rose model, so that they can be restored as part of the Caché class definition during an export from Rose. Method code is an example of such a hidden characteristic.

3.4 Loading Caché Data Type Classes

If you you are building a Rose model that will be used with Caché, you may want to begin working right away with Caché data types, rather than using the basic types provided with Rose.

The **Load Data Types from Caché** menu item makes this possible by adding to your model a list of data type classes from a Caché namespace. (The data types appear in a special Caché Data Types category within the main Rose tree.) After that, these data types will appear in the appropriate Rose drop down lists for type specification in attributes, operation return types, and so on.