



Quick Start Guide

PathMATE in Rational Rose

Version 2.0
April 1, 2005

PathMATE™ Series

Pathfinder Solutions LLC
33 Commercial Drive, Suite 2
Foxboro, MA 02035 USA
www.PathfinderMDA.com
508-543-7222

Table of Contents

Preface	iii
PathMATE Overview	v
PathMATE Toolset	v
How PathMATE Works	vi
1. Download PathMATE	1
2. Model a Simple Oven	2
Create the Model	3
Load Realized Domains	4
Create the Application Domain	4
Create Three Classes in MicrowaveCooking	5
Create the Class Diagram	7
Create the Door State Chart	9
Create the Light State Chart	13
Add Entry Actions to the State Charts	16
3. Transform Your Model	20
Prepare the Model for Transformation	20
Generate Code from the Model	23
Build an Executable System	24
4. Introduction to Spotlight	27
Run SimpleOven with Spotlight	27
Browse the Door Class	28
Action Step	31
Send Event	32
State Animation	33
5. Summary	35
Next Steps	35
About Pathfinder Solutions	35
A. PathMATE Installation	36
B. Acronyms and File Types	39
C. Document the System	40
D. SimpleOven Directory Structure	41

Preface

Audience

The *Quick Start Guide* is for software engineers who want to learn how to design embedded systems with PathMATE. It's helpful but not essential to have some familiarity with IBM's Rational Rose modeler.

Related Documents

These PathMATE documents are available at www.PathfinderMDA.com, or from your Pathfinder account manager:

- *PathMATE Installation Guide*
- *Accelerating Embedded Software Development with a Model Driven Architecture* (white paper)
- *PathMATE: Model Automation and Transformation Environment for Embedded Systems* (online brochure)

Conventions

The *Quick Start Guide* uses these conventions:

- **Bold** is for clickable buttons and menu selections.
- *Italics* is for screen text, path and file names, and other text that needs special emphasis.
- Courier denotes code, or text in a log or a batch file.
- A **Note** contains important information, or a tip that saves you time.

What You'll Need

To complete the steps in this guide, you'll need the following software on your computer:

- Microsoft XP Professional Edition
- Rational Rose
- PathMATE software development toolkit
- Microsoft Visual C++ 6.0 or 7.0
- Plain text editor

How to Use this Guide

If you are not already familiar with the PathMATE toolset, read the overview in Section 1. If you have not installed the PathMATE toolset on your computer, turn to Section 2 to download the software from www.PathfinderMDA.com, then follow the installation instructions in Appendix A. After installation, carefully follow the steps in Section 3 to learn how PathMATE works.

Contents of the Tutorial

To learn how to use the PathMATE toolset, try the simple tutorial in this guide, where you'll learn how to:

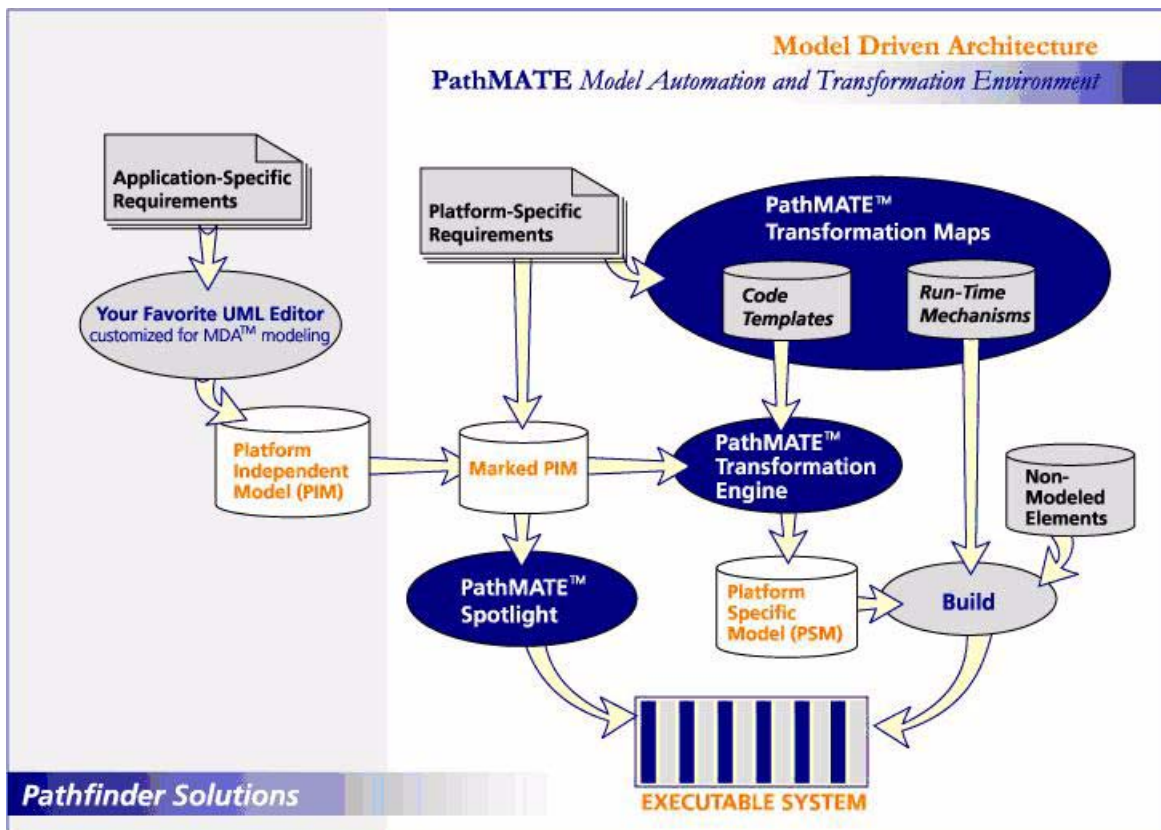
- Develop a model for a simple system.
- Generate C++ code from the model.
- Build an executable system from the code.
- Connect the application to Spotlight.

PathMATE Overview

This overview introduces Model Driven Architecture (MDA) and the PathMATE™ tools that make MDA work. MDA and PathMATE move you from writing and debugging code to developing and testing the logic of an embedded system. Over years of rigorous refinement in several industries, PathMATE tools have proven their value in rapid and effective embedded systems development.

PathMATE Toolset

The PathMATE Model Automation and Transformation Environment includes all the tools required to transform your MDA models into high-performance embedded systems (Figure).



PathMATE Model Automation and Transformation Environment

The three parts of the PathMATE toolset cooperate to turn your models into executable embedded systems:

- *Transformation Engine* – The Engine transforms platform-independent models into working, embedded software applications.
- *Transformation Maps* – Generate C, C++, or Java software with off-the-shelf Transformation Maps, or create custom maps to drive output for other languages or specific platforms.
- *Spotlight* – Verify and debug your application logic with Spotlight, the most advanced model testing environment available.

No other MDA transformation environment offers a more open or configurable set of development tools, designed to meet the requirements of embedded systems engineers.

How PathMATE Works

Use Model Driven Architecture to build complex embedded systems that meet rigorous standards for speed and reliability. MDA works because it separates what the system does from its deployment on a particular platform. PathMATE adds these advantages:

- *Greatest architectural control* – A highly configurable Transformation Engine enables you to optimize output for resource-constrained platforms.
- *Clean separation of model and code* – Conforming to the MDA paradigm, PathMATE models contain no implementation code. That gives you fast and flexible deployment and migration capabilities.
- *Configurable, target-based model execution and testing* – Preemptively eliminate platform-specific bugs, minimize quality assurance resources, and accelerate development.
- *Lowest cost of ownership* – Integrate PathMATE with your existing UML editor. Build on your previous investment in training and software.
- *Speed* – Even large transformations take just seconds with PathMATE. That enables highly iterative model development, and rapid transformation and test cycles.

This *Guide* gets you started quickly and easily.

1. Download PathMATE

Go to PathTECH at the Pathfinder website to download the files you need to evaluate PathMATE:

1. Point your browser to www.PathfinderMDA.com.
The Pathfinder Solutions home page opens.
2. Click PathTECH Login in the horizontal menu bar.
3. Log into PathTECH with User ID demo. Please contact your account manager to obtain your password.

Figure 1-1 shows the Welcome demo page that appears after you log into PathTECH.



Figure 1-1. Download PathMATE from the Pathfinder Website

4. Follow the brief instructions on the Welcome demo page to download Pathfinder Solutions PathMATE Tools and Rose integration files.

Please see *PathMATE Installation* on page 36 for detailed installation instructions. The *PathMATE Installation Guide* also explains how to license and install the toolset.

2. Model a Simple Oven

Whether you have created hundreds of models and systems or none at all, you can follow this tutorial. Learn quickly how to use the PathMATE toolset with IBM/Rational Rose and Microsoft Visual C++ to develop an executable system. The model-building part of the tutorial includes these steps:

- Create the model.
- Load reused domains.
- Create the application domain, MicrowaveCooking.
- Create three classes in MicrowaveCooking.
- Create a class diagram.
- Create two state charts.
- Add entry actions to the state charts.

To preview the sample, click **File > Open** in Rational Rose and browse to *C:\pathmate\samples\SimpleOven\rose\analysis\SimpleOven.mdl*. Double-click *SimpleOven.mdl*. The Logical View for the model opens in Rational Rose.

NOTE

The sample model shows you how the model should look when you are finished. As you build your own SimpleOven, the instructions occasionally ask you to use some building blocks supplied in the sample files.

Create the Model

To create your model, start with these steps:

1. In Windows Explorer, create this working directory for your model and all of its support files:
C:\pathmate\samples\QuickStart\rose
2. Create a subdirectory named *...\QuickStart\rose\analysis*.
3. Start Rational Rose.
4. Click **Cancel** in the Create New Model dialog, should it appear.

Rose opens a new model with a blank class diagram called *Main* contained in the Logical View (Figure 2-1). The Logical View contains your domain chart, the top level diagram in your model.

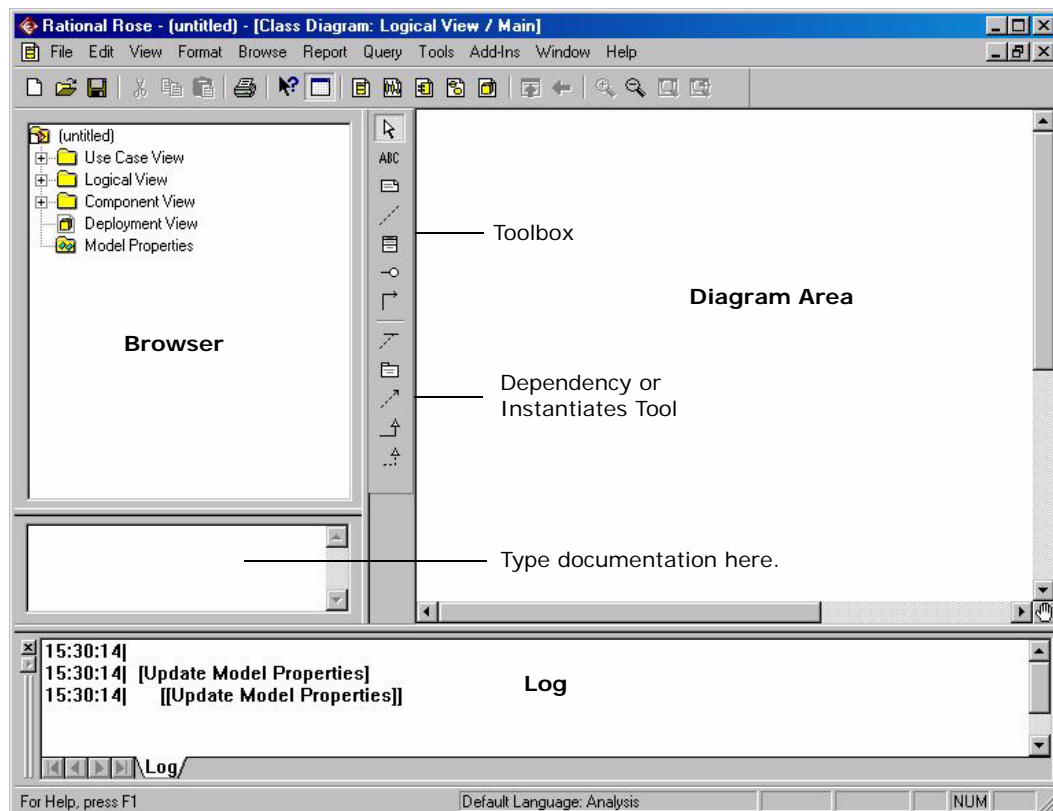


Figure 2-1. Initial Display in Rational Rose

5. In the Rational Rose browser, right-click *Main* under Logical View and rename it *SimpleOven*.
6. Click **File > Save As...** and save the new model in *C:\pathmate\samples\QuickStart\rose\analysis*. Name the .mdl file *SimpleOven.mdl*.

Load Realized Domains

SimpleOven reuses two realized domains to provide support services: SoftwareMechanisms and ExternalDeviceControl. Copy these domains to ...\\QuickStart\\rose\\analysis, and then import them into your model:

1. Locate *sw.cat* in *C:\\pathmate\\design\\rose* and copy it to *C:\\pathmate\\samples\\QuickStart\\rose\\analysis*.
2. In Rational Rose, click **File > Units > Load...** and select *C:\\pathmate\\samples\\QuickStart\\rose\\analysis\\sw.cat*. Click **Open**.
Software Mechanisms appears in the domain chart.
3. Drag SoftwareMechanisms down and to the left, away from the center of the domain chart. Click in the domain chart's open space to deselect the package (Figure 4).
4. Locate *ExternalDeviceControl.cat* in *C:\\pathmate\\samples\\SimpleOven\\rose\\analysis* and copy it to *C:\\pathmate\\samples\\QuickStart\\rose\\analysis*.
5. Click **File > Units > Load...** and select *C:\\pathmate\\samples\\QuickStart\\rose\\analysis\\ExternalDeviceControl.cat*. Click **Open**.

ExternalDeviceControl appears in the domain chart.

Figure 2-2 on page 5 shows SoftwareMechanisms and ExternalDeviceControl properly placed on the domain chart.

Create the Application Domain

SimpleOven's application domain is called MicrowaveCooking. To create MicrowaveCooking:

1. Right-click Logical View in the browser and select **New > Package** in the pop-up menu.
<<>> *NewPackage* appears in the browser.
2. Name the new package <<domain>> *MicrowaveCooking* and press Enter.
3. Drag *MicrowaveCooking* from the browser to the domain chart.
The domain *MicrowaveCooking* appears in the domain chart.
4. Use the *Dependency or instantiates* tool in the toolbox to draw a line from MicrowaveCooking to SoftwareMechanisms.
5. In the same way, connect MicrowaveCooking to ExternalDeviceControl, and ExternalDeviceControl to SoftwareMechanisms.

Figure 2-2 shows the domain chart after you complete these steps.

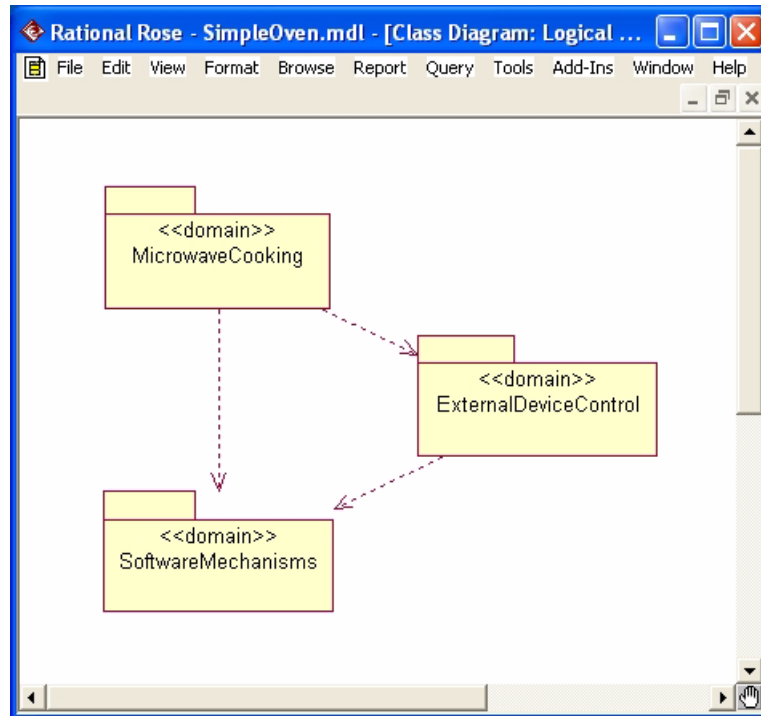


Figure 2-2. Domain Chart for SimpleOven

Create Three Classes in MicrowaveCooking

To create the classes in MicrowaveCooking:

1. Right-click *<<domain>> MicrowaveCooking* in the browser and select **New > Class** in the pop-up menu.
2. Type *Oven* in place of *<<>> NewClass* in the browser and press Enter.
3. Right-click *<<domain>> MicrowaveCooking* in the browser and select **New > Class** in the menu. Name the new class *Door*.
4. Right-click *<<domain>> MicrowaveCooking* a third time and select **New > Class** in the menu. Name the new class *Light*.

Add the attribute *dateOfManufacture* to the Oven class:

1. Right-click *Oven* in the browser and select **New > Attribute** in the pop-up menu.
2. Name the new attribute *dateOfManufacture* in the browser.
3. Right-click *dateOfManufacture* and select **Open Specification...** in the pop-up menu.
4. Select *String* in the *Type* drop-down list.
5. Click **OK** to close the Specification box.

Follow the same steps to add the attribute *serialNumber* to the Oven class (the data type for *serialNumber* is also *String*).

Add the attribute *wattage* to the Light class:

1. Right-click *Light* in the browser and select **New > Attribute** in the pop-up menu.
2. Name the new attribute *wattage* in the browser.
3. Right-click *wattage* and select **Open Specification...** in the pop-up menu.
4. Select *Integer* in the *Type* drop-down list.
5. Type *60* in the *Initial value* field.
6. Click **OK** to close the Specification box.

Figure 2-3 shows the Rational Rose browser after you create the Oven, Door and Light classes, and after you add attributes.

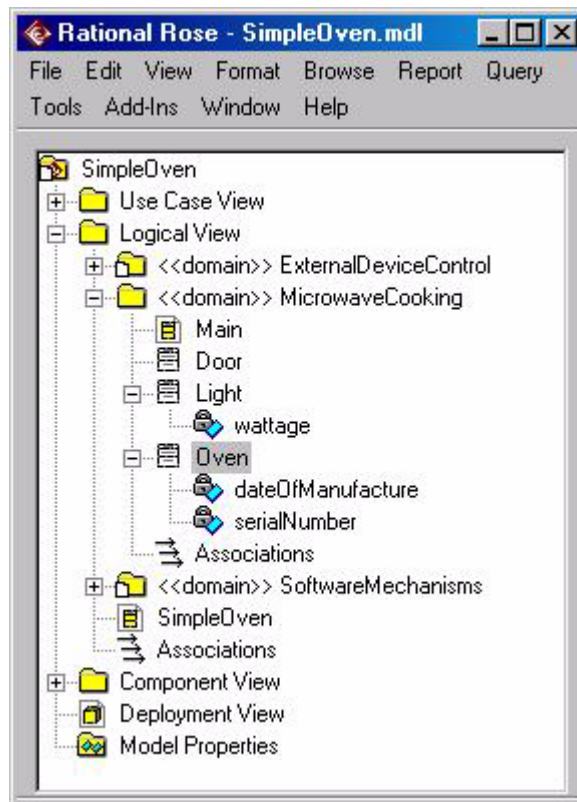


Figure 2-3. Classes with Attributes Created in MicrowaveCooking

Create the Class Diagram

To create the class diagram in the MicrowaveCooking domain:

1. Right-click *MicrowaveCooking* in the browser and select **New > Class Diagram** in the pop-up menu.
2. Name the class diagram *MicrowaveCooking* in the browser.
3. Double-click the MicrowaveCooking class diagram in the MicrowaveCooking domain.
A blank diagram for MicrowaveCooking appears.
4. Select the Oven class in the browser and drag it to the class diagram.
5. Likewise select the Door class and drag it to the open space in the diagram.
6. Repeat the drag and drop procedure for the Light class.

Figure 2-4 shows the class diagram after you complete these steps.

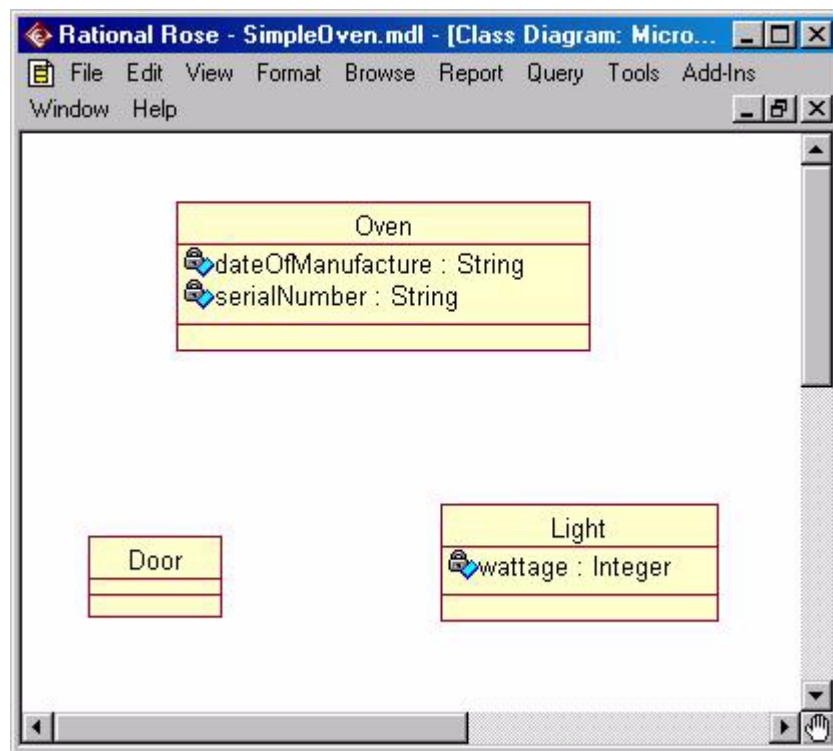


Figure 2-4. New Class Diagram for MicrowaveCooking

Now associate the Oven class with the Door class:

1. Click **Tools > Create > Association** in the top menu bar.
2. Click *Oven* in the class diagram and draw a rectilinear line to the Door class.
3. Right-click the line and select **Open Specification...** from the pop-up menu.

The Association Specification box opens.

4. Enter *A1* in the *Name* field of the Specification box.
5. Enter *is_part_of* in the *Role A* field of the Specification box.
6. Click **OK** to close the Specification box.

7. Right-click the upper part of the association line and select **Multiplicity** in the pop-up menu. Select **1** in the sub-menu.

A multiplicity of 1 appears near the upper part of association A1.

8. Right-click the lower part of the association line and select **Multiplicity** in the pop-up menu. Then select **1** in the sub-menu.

A multiplicity of 1 appears near the lower part of association A1.

Follow the same steps to associate the Oven class with the Light class in the diagram:

1. Draw the association line.
2. Open the Specification box for the association.
3. Type *A2* in the *Name* field of the specification box. Type *illuminates_interior* in the *Role A* field.
4. Click **OK** to close the Specification box.
5. Specify a multiplicity of *1* for each end of association A2.

Figure 2-5 shows the class diagram after you complete these steps.

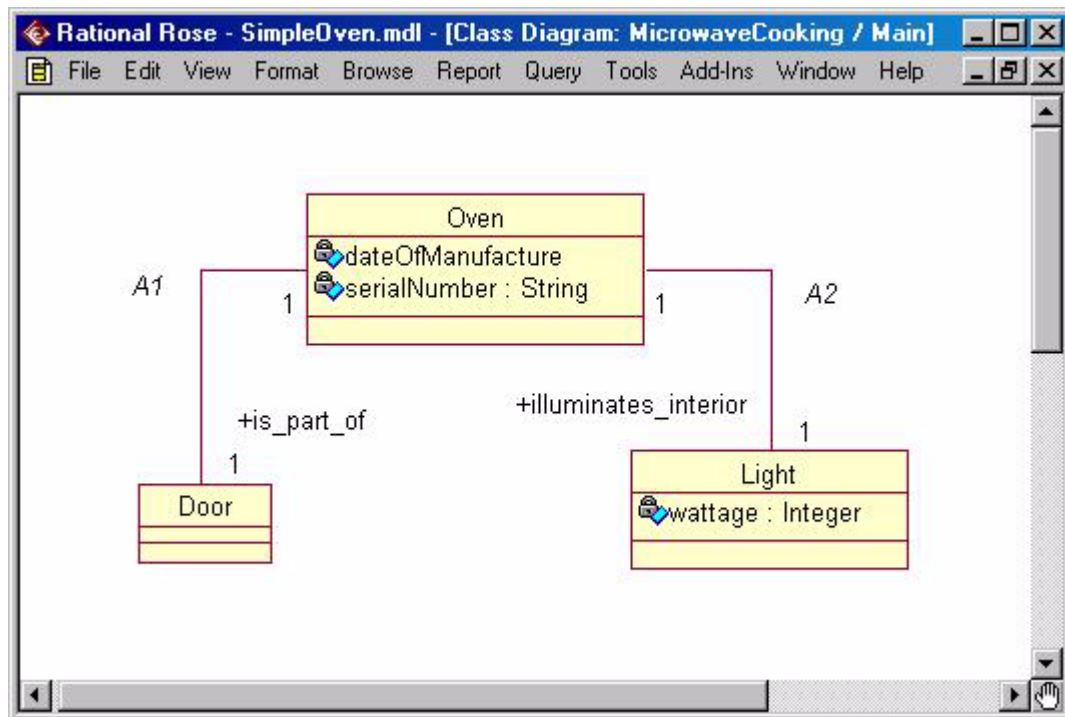


Figure 2-5. Class Diagram with Associations

Create the Door State Chart

Create a state chart for the oven door to show transitions between the closed state and the open state.

1. In the Rose browser, right-click the Door class in the MicrowaveCooking domain. Select **New > Statechart Diagram** in the pop-up menu.

A new State/Activity Model appears under the Door class in the browser.

2. Name the new diagram in the State/Activity Model *Door* and press Enter.
3. Double-click the Door state chart in the browser to open the blank state chart.

To place the state symbols in the new state chart, follow these steps:

1. Click **Tools > Create > Start State** and place the Start State in the state chart.
2. Click **Tools > Create > State** and place the New State symbol in the diagram below the Start State symbol.
3. Type *Closed* in the state symbol and click the open space to create the new state.
4. Click **Tools > Create > State** in the top menu bar and place the New State symbol in the diagram below the Closed State symbol.
5. Type *Open* in the state symbol and click the open space to create the new state.

Figure 2-6 shows the Door state chart after you create the Start state, the Closed state, and the Open state.

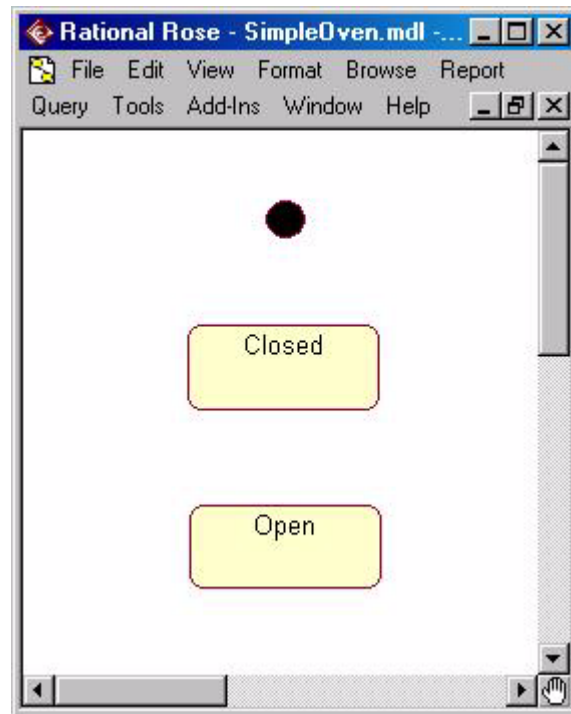


Figure 2-6. New Door State Chart

To create the transitions:

1. Click **Tools > Create > Transition** in the top menu bar.
2. Click the Start State symbol and drag a transition line to *Closed*.
3. Click **Tools > Create > Transition** and drag a new transition line from *Closed* to *Open*.
4. With the new transition line selected, click **Format > Line Style > Rectilinear** in the top menu bar.
5. Drag the line to the left side of the states.
6. Click **Tools > Create > Transition** and draw a second transition line from *Open* to *Closed*.
7. Click **Format > Line Style > Rectilinear** and drag the line to the right side of the states.

Figure 2-7 shows the Door state chart after you complete these steps.

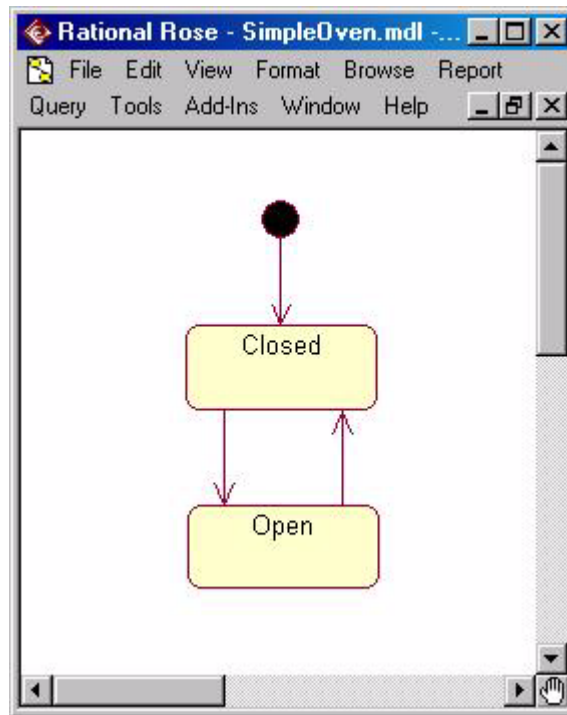


Figure 2-7. Door State Chart with Transitions

To specify the event associated with each transition:

1. Right-click the line from *Closed* to *Open* and select **Open Specification...** in the pop-up menu.
2. Type *IsOpen* in the *Event* field of the Specification box and click **OK** to close the box.

The event name *IsOpen* appears in the Door state chart.

3. Right-click the line from *Open* to *Closed* and select **Open Specification...** in the pop-up menu.
4. Type *IsClosed* in the *Event* field of the Specification box and click **OK** to close the box.

The event name *IsClosed* appears in the Door state chart.

To add the PathMATE events *IsOpen* and *IsClosed* to the Door class:

1. Open the Door state chart.
2. Right-click the transition line for *IsOpen* and select **PathMATE Open Event Spec** in the pop-up menu.

The Operation Specification for *IsOpen* appears. Click **OK** to close the box.

3. Right-click the transition line for *IsClosed* and select **PathMATE Open Event Spec** in the pop-up menu.

The Operation Specification for *IsClosed* appears. Click **OK** to close the box.

NOTE

Select the transition line, not the label, when you right-click to open the pop-up menu that contains PathMATE Open Event Spec.

Create the Light State Chart

To create the Light state chart, repeat the same procedure you used to create the Door state chart:

1. Right-click the Light class in the browser and select **New > Statechart Diagram** in the pop-up menu.
2. Name the diagram *Light* in the browser.
3. Double-click *Light* in the browser to open the new state chart.

Next create the states in the diagram:

1. Click **Tools > Create > Start State**, and place the Start State in the state chart.
2. Click **Tools > Create > State**. Place the new state in the diagram and name it *Off*.
3. Click **Tools > Create > State**. Place the new state in the diagram and name it *On*.

Then create the transitions:

1. Click **Tools > Create > Transition**. Draw a line from the Start State to *Off*.
2. Click **Tools > Create > Transition**, and draw a line from *Off* to *On*. Click **Format > Line Style > Rectilinear**, and drag the line to the left side of the diagram.
3. Right-click the line from *Off* to *On* and select **Open Specification...** in the pop-up menu.
4. Type *TurnOn* in the *Event* field of the Specification box and click **OK** to close the box.
5. Click **Tools > Create > Transition**, and draw a line from *On* to *Off*. Click **Format > Line Style > Rectilinear**, and drag the line to the right side of the diagram.
6. Right-click the line from *On* to *Off* and select **Open Specification...** in the pop-up menu.
7. Type *TurnOff* in the *Event* field of the Specification box and click **OK** to close the box.

Figure 2-8 shows the Light state chart after you complete these steps.

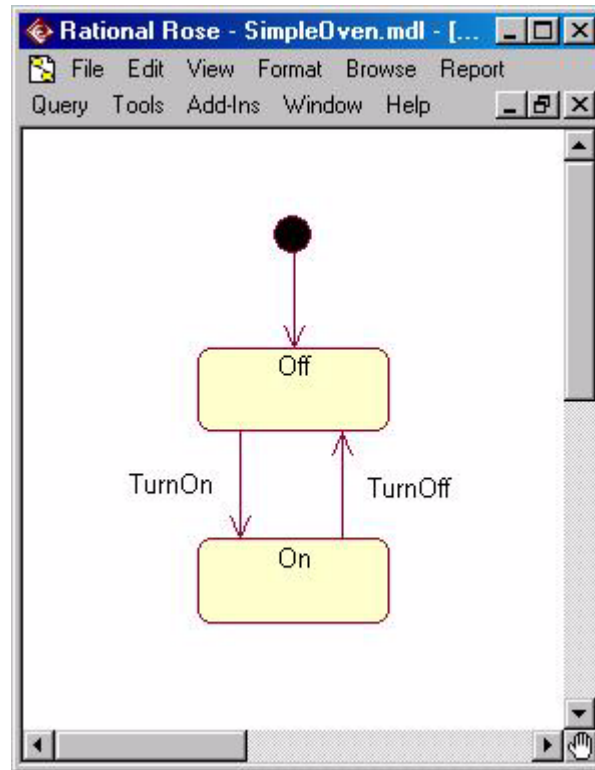


Figure 2-8. Light State Chart

To add the PathMATE events *TurnOn* and *TurnOff* to the Light class:

1. Open the state chart for the Light class.
2. Right-click the transition line for *TurnOn* and select **PathMATE Open Event Spec** in the pop-up menu.

The Operation Specification for *TurnOn* appears. Click **OK** to close the box.

3. Right-click the transition line for *TurnOff* and select **PathMATE Open Event Spec** in the pop-up menu.

The Operation Specification for *TurnOff* appears. Click **OK** to close the box.

Figure 2-9 shows the class diagram after you complete these steps.

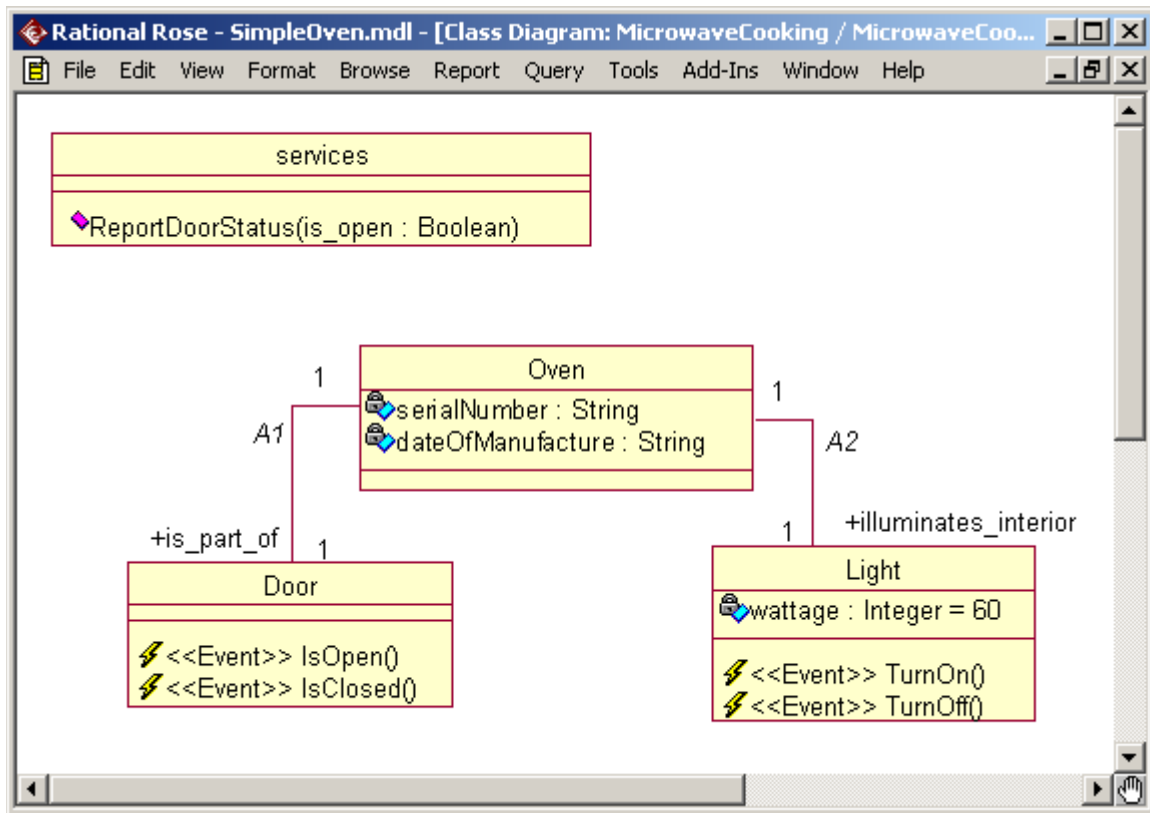


Figure 2-9. Class Diagram for MicrowaveCooking

Add Entry Actions to the State Charts

Entry actions establish a relationship between the Door state and the Light state (see Table 2-10).

Table 2-10. Interacting States in SimpleOven

Door State	Light State
Closed	Off
Open	On

To create the entry action for *Closed* in the Door state chart:

1. Right-click *Closed* in the Door state chart and select **PathMATE Open > Entry Action**.

A confirmation dialog shows the directory and the name of the support file (Figure 2-11).

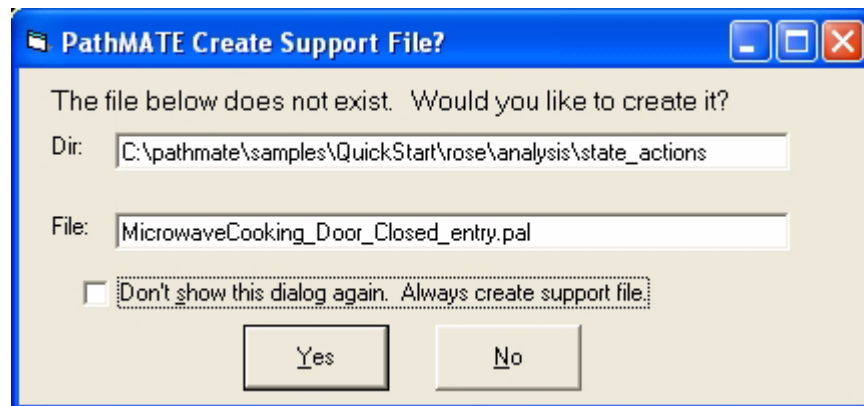


Figure 2-11. PathMATE Create Support File?

To bypass the dialog, click *Don't show this dialog again. Always create support file*.

2. Click **Yes** to create *MicrowaveCooking_Door_Closed_entry.pal*.
The .pal file opens in your text editor.
3. Copy the text below and paste it at the end of the open .pal file.

```
Ref<Light> interior_light = FIND this->A1->A2;
GENERATE Light:TurnOff() TO (interior_light);
```
4. Save and close the .pal file.
5. Right-click **Closed** in the Door state chart and select **PathMATE Update Actions** in the pop-up menu.

The entry action appears in the *Closed* state.

NOTE

The procedures below require that you clip text from this document and paste it in a text editor. To do so, click the Select Text tool in your Acrobat toolbar. Then select the text and copy it to your clipboard.

To create the entry action for *Open* in the Door state chart:

1. Right-click *Open* in the Door state chart and select **PathMATE Open > Entry Action**.

A confirmation dialog shows the directory and the name of the support file.

2. Click **Yes** to create *MicrowaveCooking_Door_Open_entry.pal*.

The .pal file opens in your text editor.

3. Copy the text below and paste it at the end of the open .pal file.

```
Ref<Light> interior_light = FIND this->A1->A2;  
GENERATE Light:TurnOn() TO (interior_light);
```

4. Save and close the .pal file.
5. Right-click *Open* in the diagram and select **PathMATE Update Actions** in the pop-up menu.

The entry action appears in the Open state.

Readjust the lines and labels in the Door state chart. Figure 2-12 shows the Door state chart after you create the entry actions.

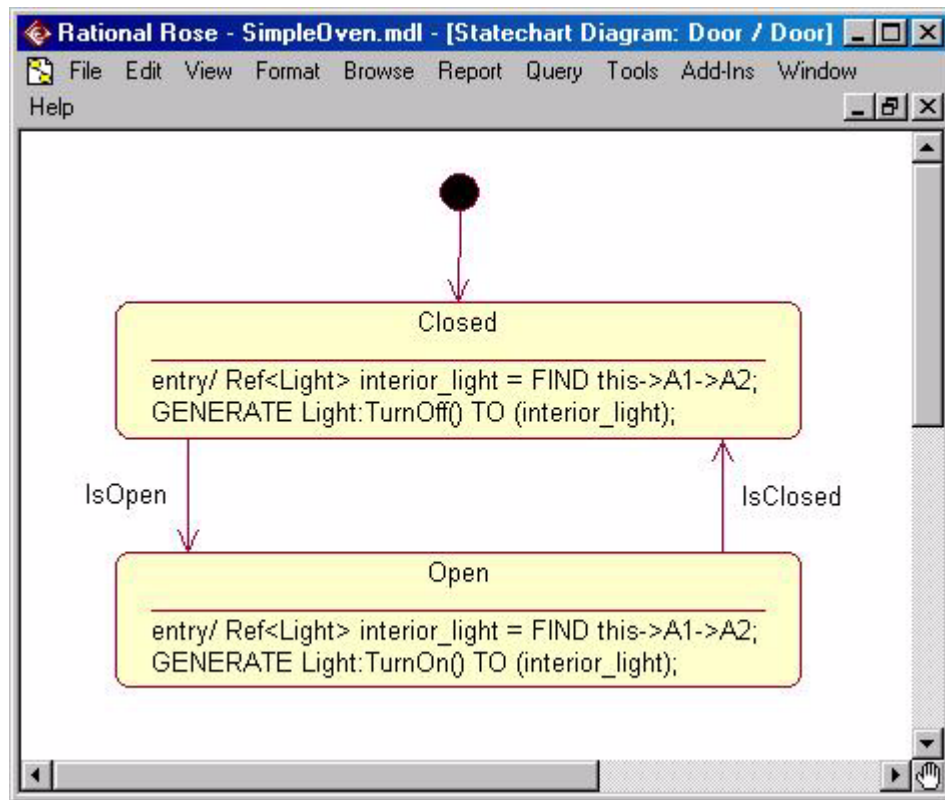


Figure 2-12. Completed Door State Chart

To create the entry action for *Off* in the Light state chart:

1. Right-click *Off* in the Light state chart and select **PathMATE Open > Entry Action**.

A confirmation dialog shows the directory and the name of the support file.

2. Click **Yes** to create *MicrowaveCooking_Light_Off_entry.pal*.

The .pal file opens in your text editor.

3. Copy the text below and paste it at the end of the open .pal file.

```
ExternalDeviceControl:DeactivateDevice(SYS_DEVICE_LIGHT);
```

4. Save and close the .pal file.
5. Right-click *Off* in the diagram and select **PathMATE Update Actions** in the pop-up menu.

The entry action appears in the *Off* state.

To create the entry action for *On* in the Light state chart:

1. Right-click *On* in the Light state chart and select **PathMATE Open > Entry Action**.

A confirmation dialog shows the directory and the name of the support file.

2. Click **Yes** to create *MicrowaveCooking_Light_On_entry.pal*.

The .pal file opens in your text editor.

3. Copy the text below and paste it at the end of the open .pal file.

```
ExternalDeviceControl:ActivateDevice(SYS_DEVICE_LIGHT);
```

4. Save and close the .pal file.

5. Right-click *On* in the diagram and select **PathMATE Update Actions** in the pop-up menu.

The entry action appears in the *On* state.

Readjust the lines and labels in the Light state chart. Figure 2-13 shows the Light state chart after you create the entry actions.

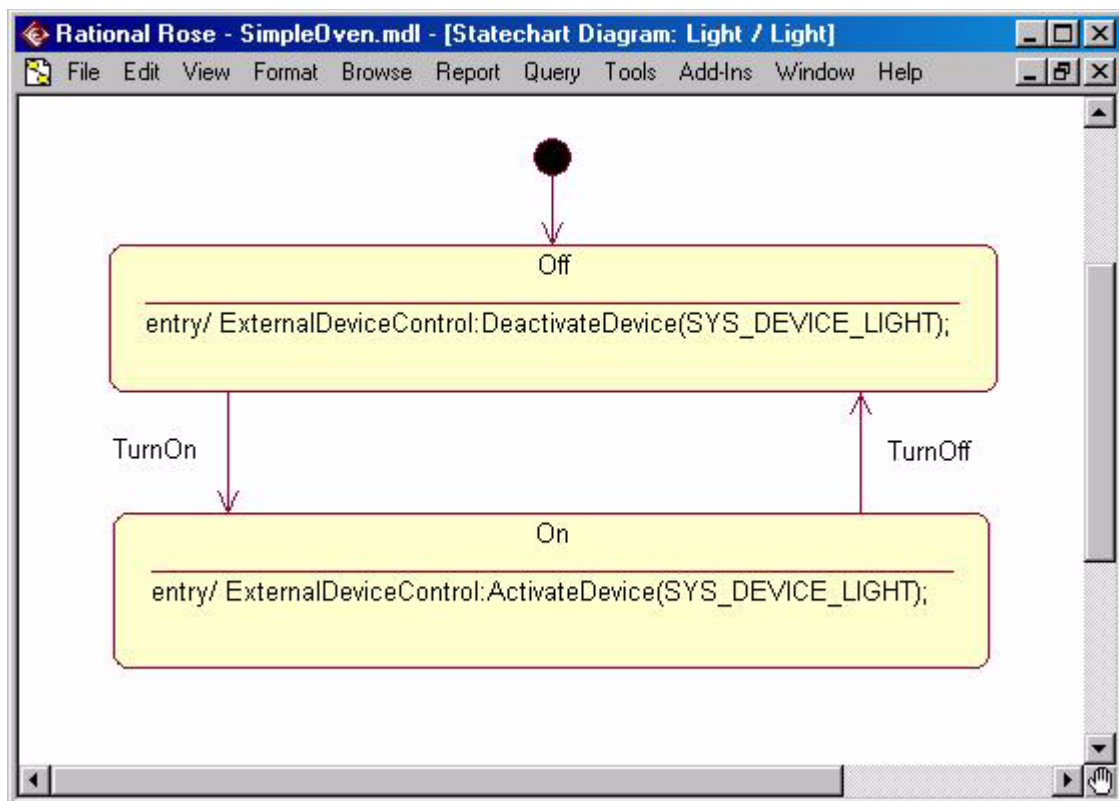


Figure 2-13. Completed Light State Chart

3. Transform Your Model

Now you are ready to transform your model into compilable code. This section takes you through the following steps:

- Prepare the model for transformation.
- Generate code from the model.
- Build an executable system.

Prepare the Model for Transformation

To prepare the model for transformation, complete these tasks:

- Initialize MicrowaveCooking.
- Create user defined types.
- Enable Spotlight.

Initialize MicrowaveCooking

To initialize the MicrowaveCooking domain:

1. Right-click MicrowaveCooking in the domain chart.
2. Select **PathMATE Open > Domain Init** in the pop-up menu.
A confirmation dialog shows the directory and the name of the support file.
3. Click **Yes** to create *MicrowaveCooking.pal* in ...*analysis\init*.
The .pal file opens in your text editor.
4. Copy the text below and paste it at the end of *MicrowaveCooking.pal*:

```
// Set up instances for MicrowaveCooking

Ref<Oven> mw_oven = CREATE Oven (serialNumber =
"G023-4ZZ-8811", dateOfManufacture =
"2004/02/22; 10:41");

Ref<Light> interior_light = CREATE Light();

Ref<Door> door = CREATE Door();

LINK mw_oven Aldoor;

LINK mw_oven A2interior_light;

// Now just to start things off, open the door
GENERATE Door:IsOpen() TO (door);
```

5. Save and close the .pal file.

Create a User Defined Type

To create user a defined type:

1. Right-click in the open space of the domain chart, or in the open space of any model diagram.
2. Select **PathMATE Open > System > Types** in the pop-up menu.
A confirmation dialog shows the directory and the name of the support file.
3. Click **Yes** to create *SimpleOven.typ* in ... \analysis\types.

The .typ file opens in your text editor.

4. Copy the text below and paste it at the end of *SimpleOven.typ*:

```
ENUM sys_device_e  
{  
    SYS_DEVICE_LIGHT  
}
```

5. Save and close the .typ file.

Enable Spotlight

To enable Spotlight, PathMATE's model-level debugger:

1. Right-click MicrowaveCooking in the domain chart and select **Open Specification...** in the pop-up menu.

The Package Specification for MicrowaveCooking opens.

2. Select the PathMATE tab in the Specification box.
3. Select *SpotlightEnabled* in the *Model Properties* field.
4. Select *SpotlightEnabled* a second time.

A drop-down list appears.

5. Select *True* from the drop-down list. Then click outside the list to close it.

The *SpotlightEnabled* setting in Model Properties is *True*. The Source is *Override* (Figure 3-1)

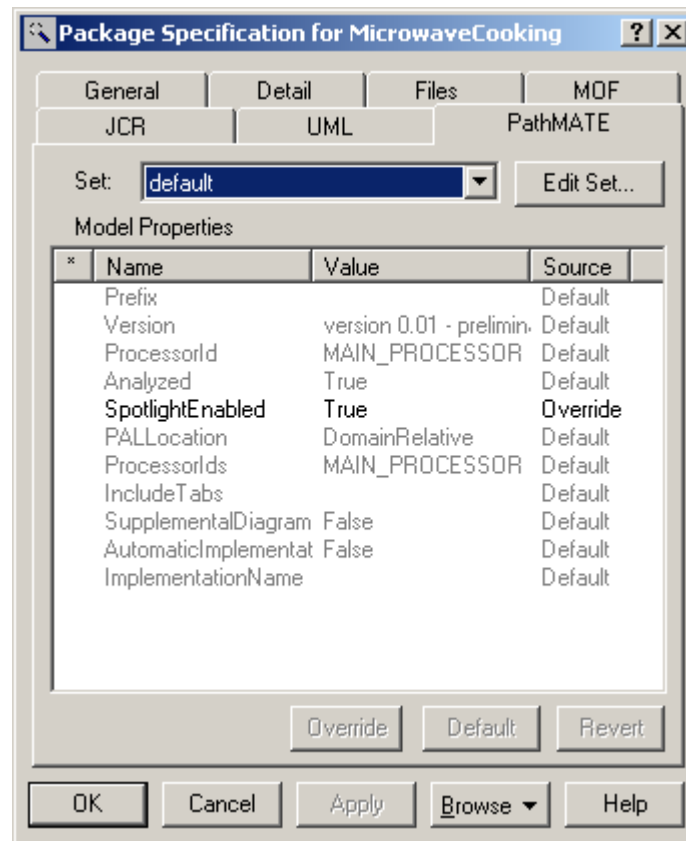


Figure 3-1. Enable Spotlight in the Package Specification for MicrowaveCooking

6. Click **OK** to close the Package Specification box.

Generate Code from the Model

To generate C++ code the first time:

1. Click **Tools > PathMATE Generate > C++** in Rational Rose.

The PathMATE Extract box opens briefly as the models are exported via XML. Then Code Generation Options opens (Figure 3-2).

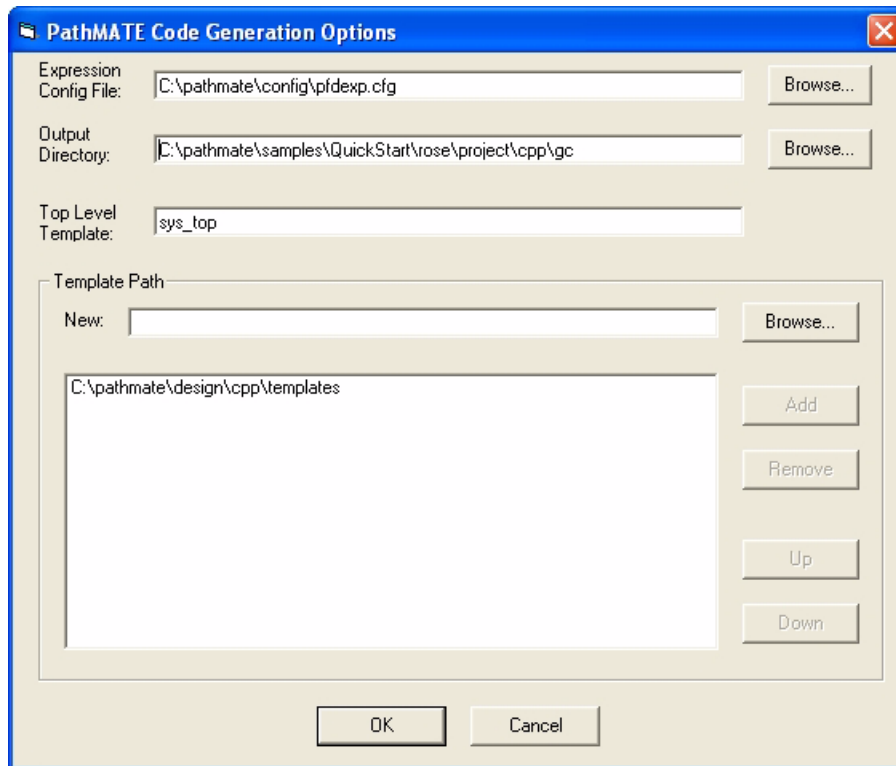


Figure 3-2. Code Generation Options

NOTE

Code Generation Options appears the first time you generate code for a model. PathMATE stores these options in ...\\QuickStart\\rose\\project\\cpp\\gencpp.bat. You may customize gencpp.bat after PathMATE has created it.

2. Verify that the *Output Directory* path is correct.
3. Click **OK** to close Code Generation Options and start transformation.

The Transformation Engine runs in a command window and informs you of progress in the code generation process.

If PathMATE encounters errors during transformation, the error log *sprngbrd.err* in *...\QuickStart\rose\project\cpp* opens automatically. If you have one or more errors:

1. Correct the problem identified in the error log.
2. Close the error log.
3. Return to Rational Rose and click **Tools > PathMATE Generate > C++** in the top menu bar.

Build an Executable System

To build an executable system, first import required files from the SimpleOven sample system, then compile the code in Visual C++.

Copy Files from the Sample System

Before you build SimpleOven, copy three files from the SimpleOven sample system to the QuickStart directory:

- *properties.txt* – contains markings used to set up the Visual C++ project files
- *genproject.bat* – the build script required to create project files for the Visual C++ compiler
- *ExternalDeviceControl_services_realized.cpp* – provides domain services to the application domain, MicrowaveCooking

Table 3-3 shows where to find each file in the SimpleOven sample system, and where to place the file in your own QuickStart directory structure. To see an overview of the entire directory structure, see *SimpleOven Directory Structure* on page 41.

Table 3-3. Files to Import from the SimpleOven Sample System

Filename	File Location in the SimpleOven Sample System	Copy the File to This Location in the QuickStart System
properties.txt	C:\pathmate\samples\SimpleOven\rose\project\cpp\properties.txt	C:\pathmate\samples\QuickStart\rose\project\cpp\properties.txt
genproject.bat	C:\pathmate\samples\SimpleOven\rose\project\cpp\genproject.bat	C:\pathmate\samples\QuickStart\rose\project\cpp\properties.txt
ExternalDeviceControl_services_realized.cpp	C:\pathmate\samples\SimpleOven\rose\project\cpp\realized_cpp\ExternalDeviceControl_services_realized.cpp	C:\pathmate\samples\QuickStart\rose\project\cpp\realized_cpp\ExternalDeviceControl_services_realized.cpp

PathMATE created the directory *...\project\cpp* when you generated code the first time. Simply place *properties.txt* and *genproject.bat* in the existing directory. To import the realized code in *ExternalDeviceControl_services_realized.cpp*, copy both the file and the directory *realized_cpp* to *...\project\cpp*, as indicated in the table.

Run the Build Script

To create the project files, run *genproject.bat*:

1. Browse to ...\\QuickStart\\rose\\project\\cpp and double-click *genproject.bat*.
The command window informs you that *genproject.bat* ran successfully.
2. Press any key to close the command window.
3. Verify that *SimpleOven.dsw* and the other project files exist in ...\\QuickStart\\rose\\project\\cpp.

Build SimpleOven in Visual C++

Use Visual C++ to build *SimpleOven.exe*. The QuickStart project files are compatible with versions 6.0 and 7.0 of the Microsoft Visual C++ compiler. Locate the project files for SimpleOven in ...\\QuickStart\\rose\\project\\cpp. Among them is a project workspace file named *SimpleOven.dsw*.

To launch the Visual C++ compiler and build *SimpleOven.exe*:

1. Double-click *SimpleOven.dsw* in ...\\project\\cpp.
Visual C++ opens.

NOTE

*When you double-click SimpleOven.dsw, a prompt may ask if you want to convert your Visual C++ 6.0 project files to version 7.0. Click **Yes To All** to convert your files and open the project in Visual C++ 7.0.*

2. Right-click SimpleOven in the Visual C++ Solution Explorer and select **Set as StartUp Project** in the pop-up menu.

3. Click **Build > Build SimpleOven** in the top menu bar.

Observe the build log at the bottom of the display as Visual C++ builds the system. Figure 3-4 shows the log at the end of a successful build.

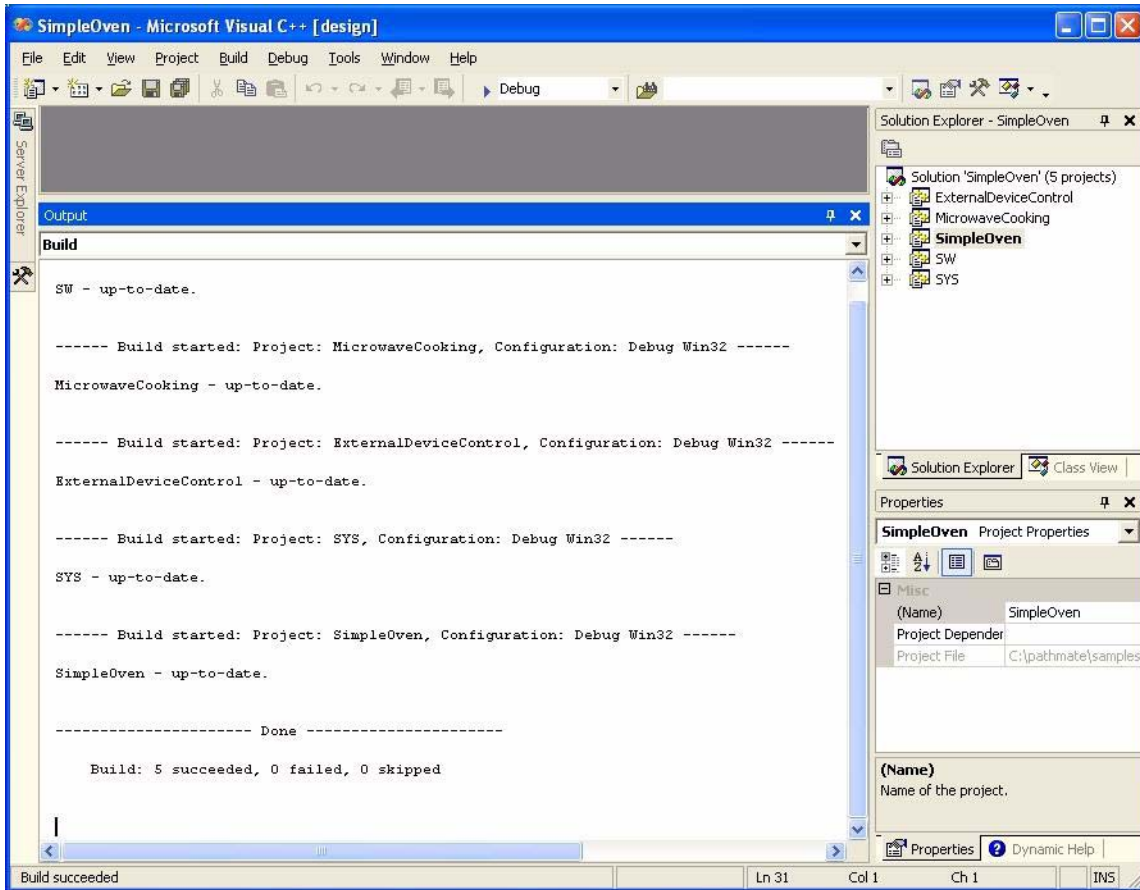


Figure 3-4. Visual C++ After a Successful Build

4. Introduction to Spotlight

Spotlight is PathMATE's model-level debugger. This section explains how to use Spotlight to browse the system, step through its action language, and send an event.

Run SimpleOven with Spotlight

After you have a successful build, launch the application and connect it to Spotlight:

1. To run the application, browse to *C:\pathmate\samples\QuickStart\rose\project\cpp\Debug* and double-click *SimpleOven.exe*.

A command window opens to say that the application is running and waiting for a connection to Spotlight on port 5150.

2. To launch Spotlight, click **Start > Pathfinder Solutions > Spotlight** on the Windows desktop.
3. Click the Connect icon at the left end of the toolbar to connect Spotlight to the target application (Figure 4-1).

When Spotlight is successfully connected, the three domains in SimpleOven appear in the browser on the left, and the status bar indicates *System: Connected*.

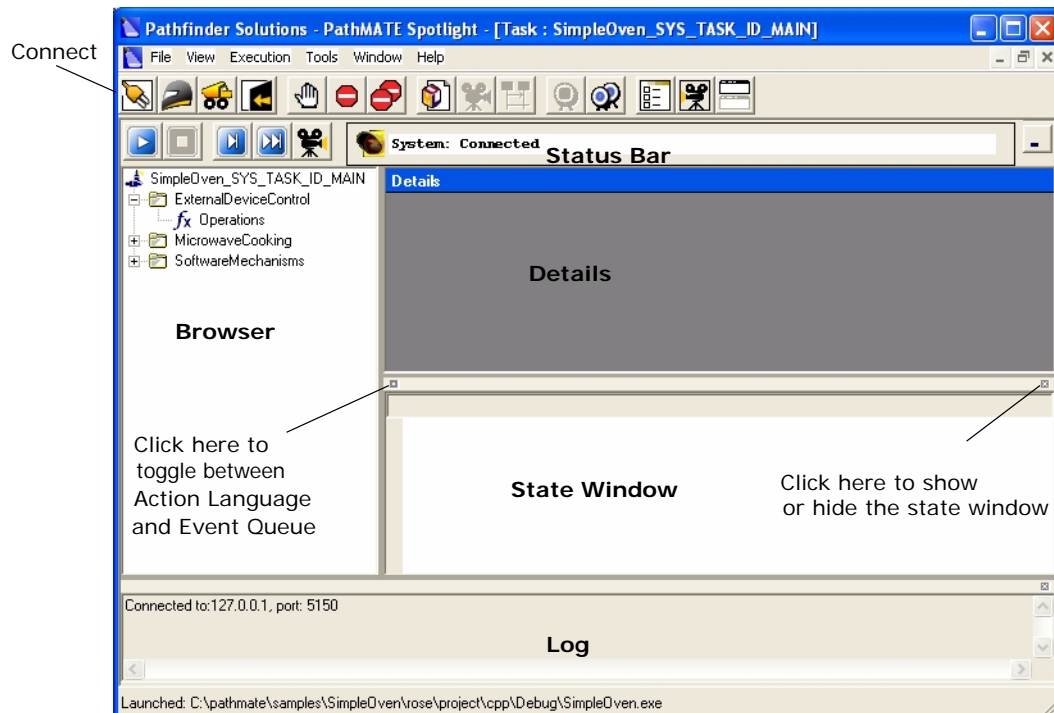


Figure 4-1. Spotlight Successfully Connected

Browse the Door Class

To browse the Door class from the Spotlight browser:

1. Click the Incident Step button above the browser (Figure 4-2).

The status indicator changes from *Connected* to *Object Transition*.

2. Expand the MicrowaveCooking domain in the browser.
3. Select the Door class in the browser (Figure 4-2).

The details pane shows one instance of the Door class. The door object is in the Open state, and it is part of one oven (association A1).

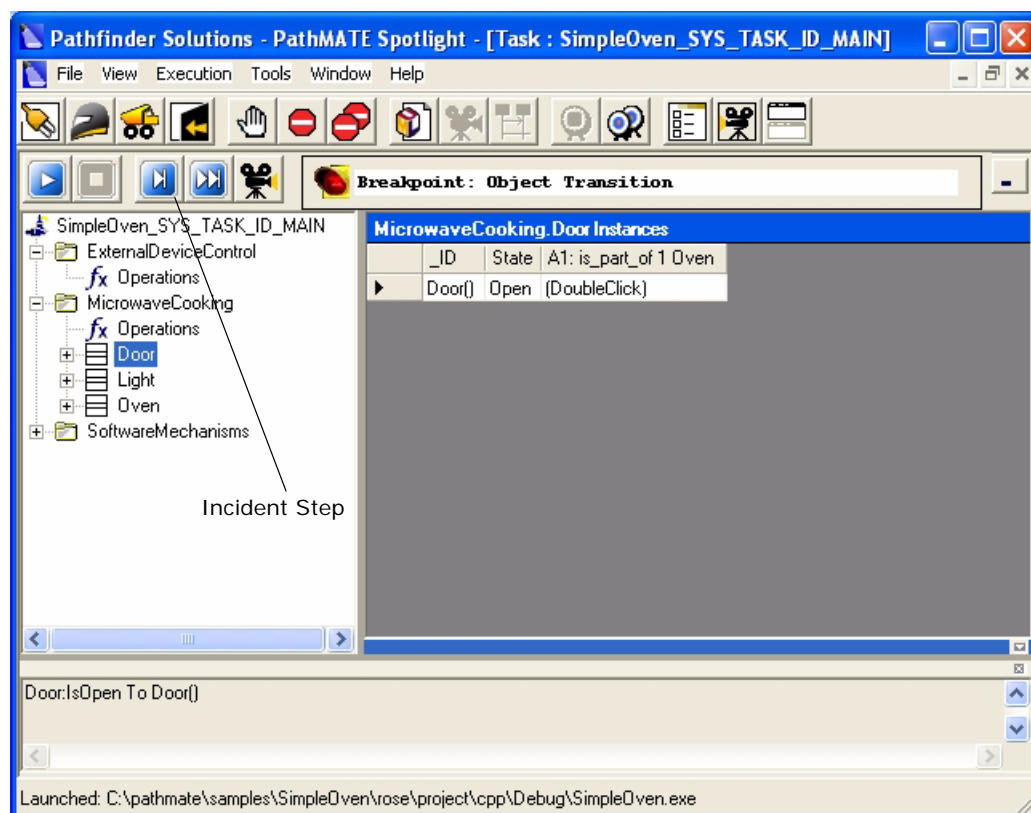


Figure 4-2. One Instance of the Door Class

4. Expand the Door class in the browser.

5. Select *States* in the Door class (Figure 4-3).

The details pane updates to show the permitted states for the Door class.

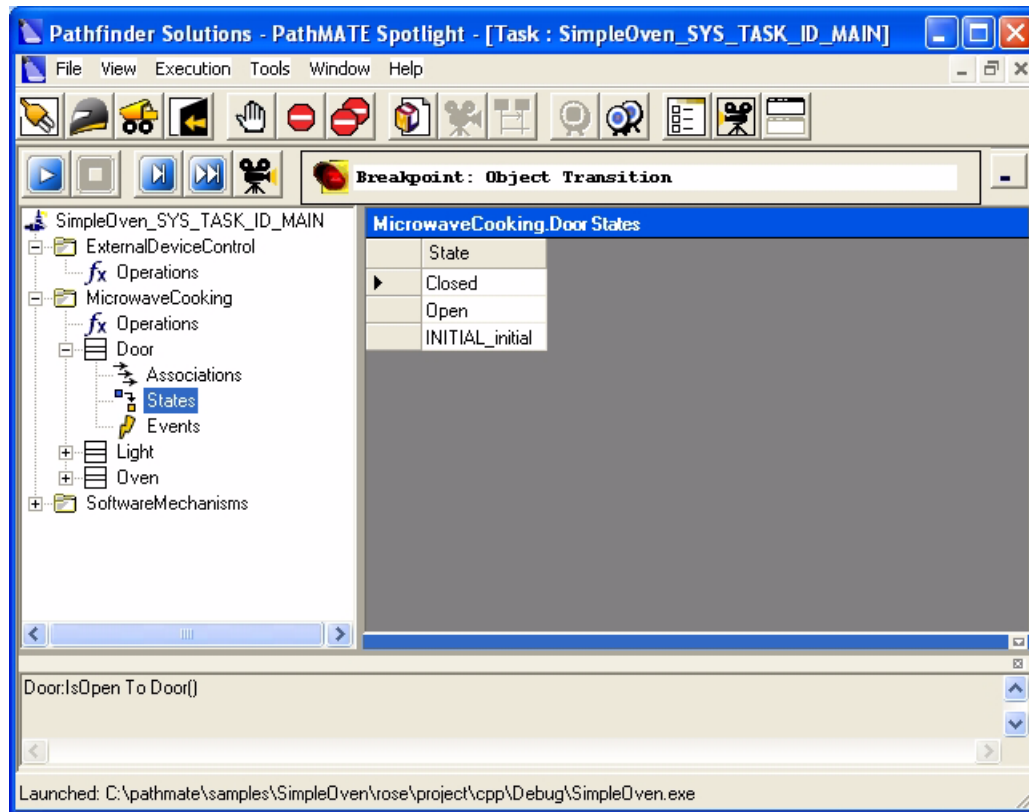


Figure 4-3. States in the Door Class

6. Select *Events* in the browser (Figure 4-4).

The details pane shows the events that you can send to the Door class.

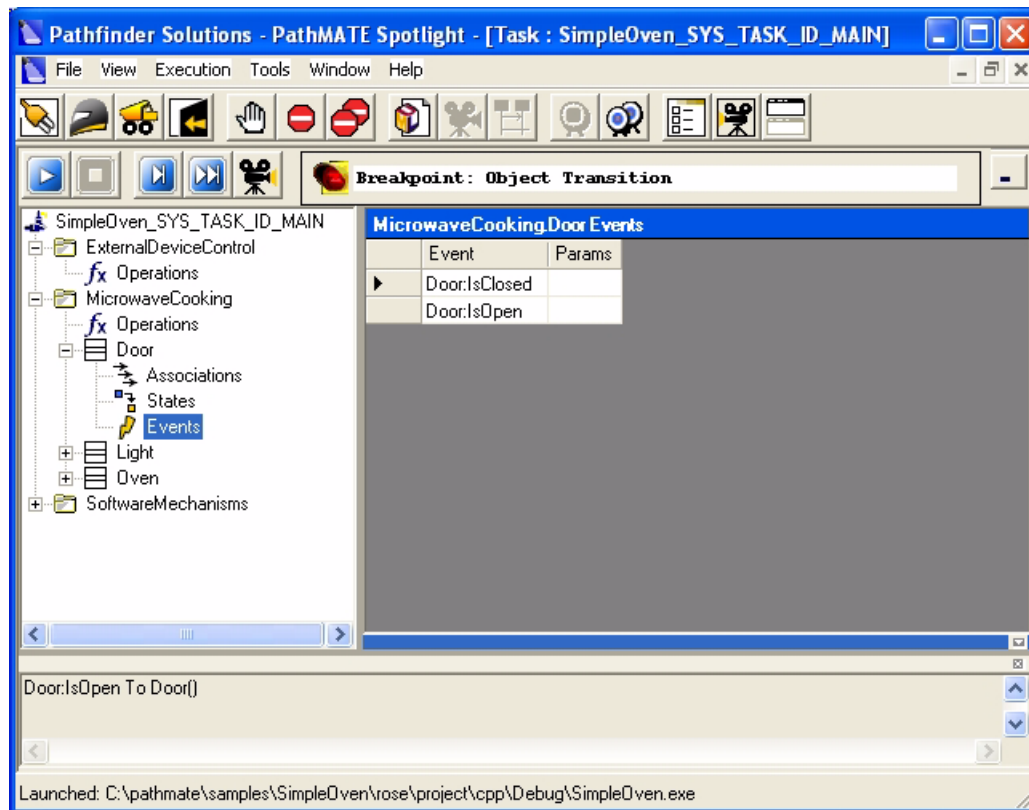


Figure 4-4. Events in the Door Class

Action Step

To step through the action language in the Simple Oven .pal files:

1. Click the Action Step button above the browser (Figure 4-5).
MicrowaveCooking_Door_Open.pal appears in the state window.

NOTE

Figure 4-5 shows how to open and close the state window. When the state window is open, use Ctrl+Q to toggle between action language and the event queue.

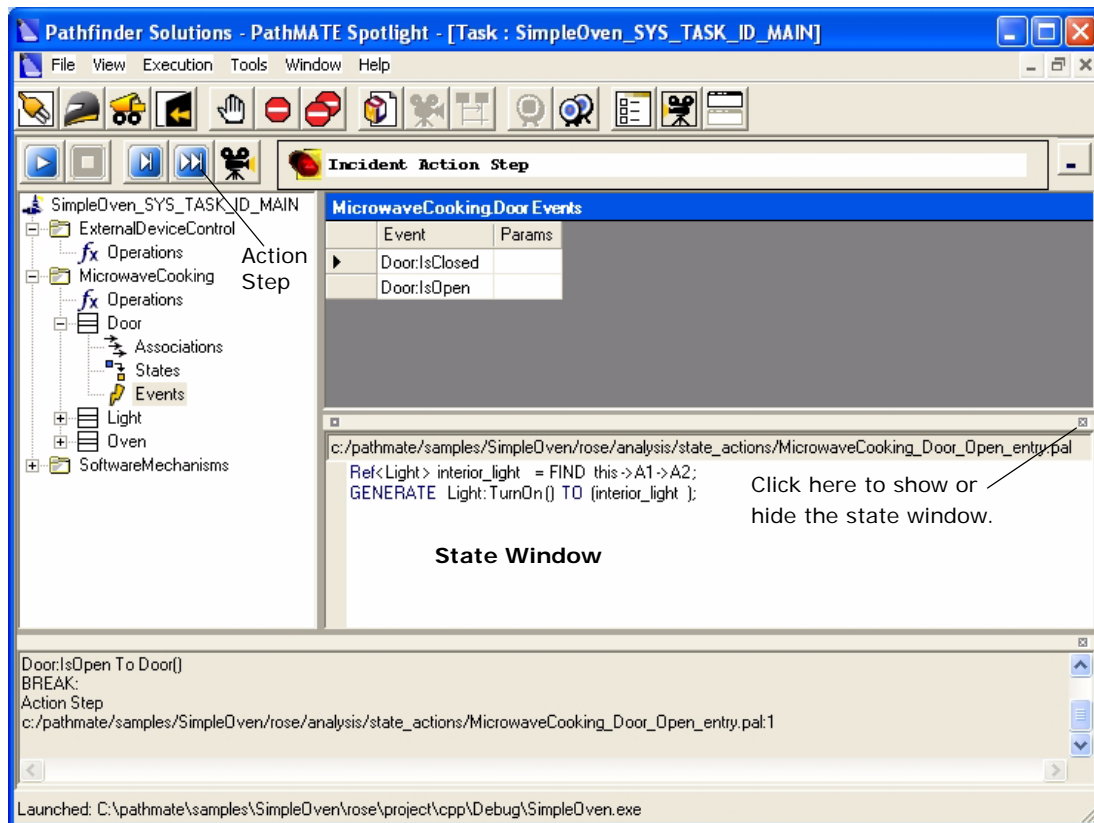


Figure 4-5. Step Through the Action Language in SimpleOven

2. Click Action Step again.
The active step arrow advances to the next PAL statement.
3. Continue to click Action Step until
MicrowaveCooking_Light_On_entry.pal appears in the PAL viewer.

Send Event

To send an event from Spotlight to SimpleOven:

1. Select *Door* in the browser.
2. Right-click *Door()* in the details pane and select **Generate event for Door()** in the pop-up menu.

The Send Event dialog opens.

3. Select *Door:IsClosed* in the drop-down list (Figure 4-6).

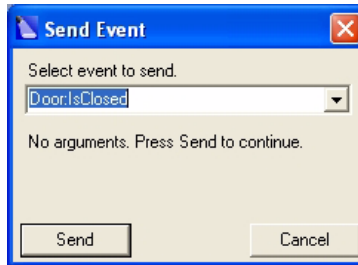


Figure 4-6. Send the Event Door:IsClosed

4. Click **Send**.
5. Press Ctrl+Q to display the event queue in the lower pane.

The event *Door:IsClosed* is queued to be sent to *Door()*. The event appears in the queue under Events in the lower pane (Figure 4-7).

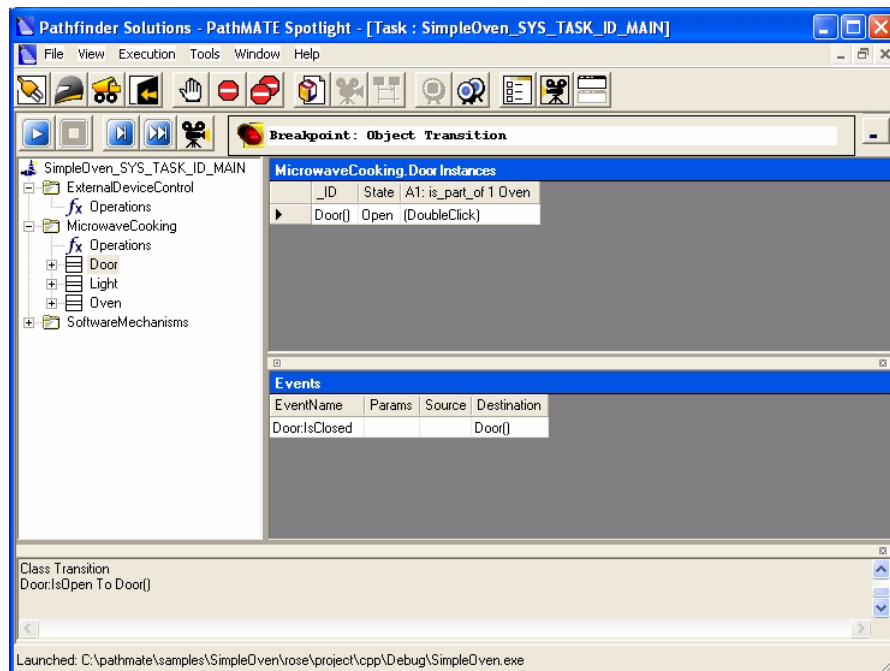


Figure 4-7. Event Queue Contains Door:IsClosed

State Animation

To animate SimpleOven's state transitions in Rational Rose:

1. Select **File > Open UML Model...** in the top menu bar to connect Spotlight to Rose.

The Open UML Model dialog appears with ...\\SimpleOven.mdl in the *Model to open* field (Figure 4-8).

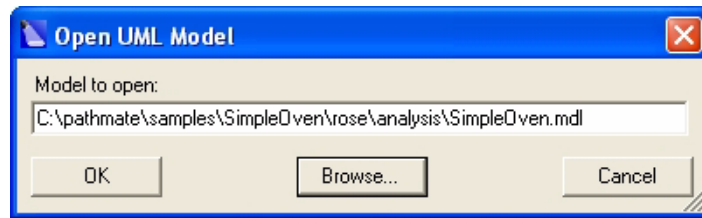


Figure 4-8. Open UML Model

2. Click **OK** to close the Open UML Model dialog.
Rational Rose opens and displays the SimpleOven domain chart.
3. Select **Execution > Animation On/Off...** in the top menu bar to enable state animation.
4. Position Spotlight and Rational Rose so both are visible on your desktop.

Make sure the Rose window is large enough to accommodate an entire statechart.

5. Click the Go button above the browser in Spotlight (Figure 4-9).

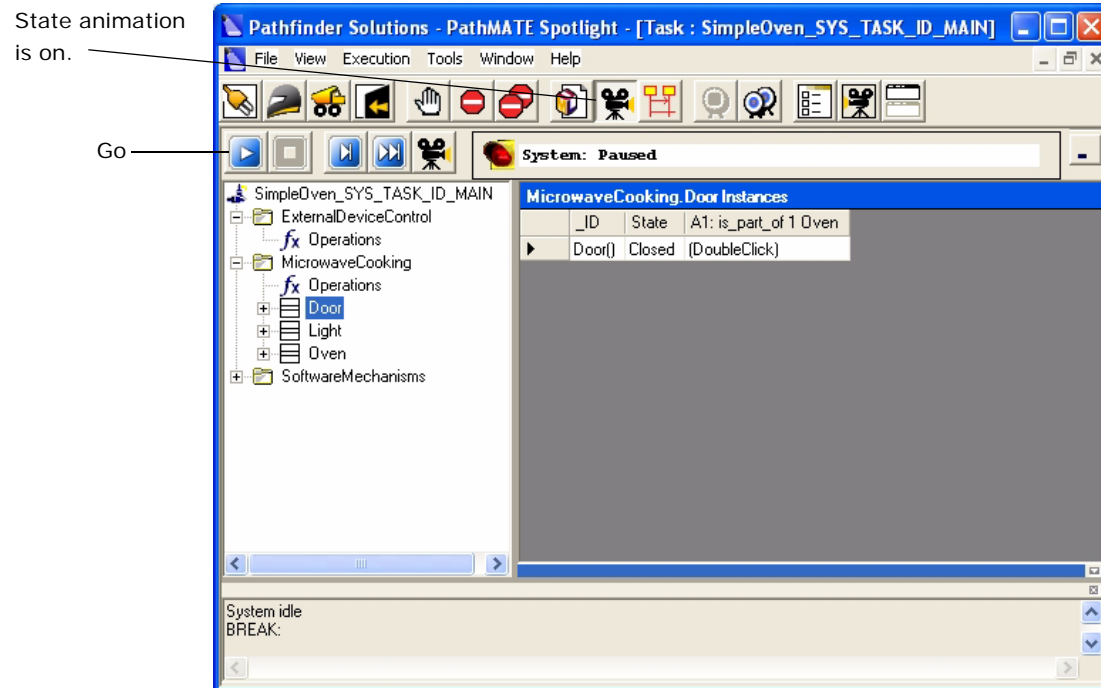


Figure 4-9. State Animation

Immediately after you click the Go button in Spotlight, the Door statechart opens in Rose to show the transition from Open to Closed. The Light statechart follows, to show the transition from On to Off.

5. Summary

Embedded MDA with PathMATE separates the logic of a system from its deployment on a specific platform. That simplifies the design of system components. Simple design yields substantial benefits all across the development cycle:

- Improve productivity in your initial development effort.
- React quickly to changing software and hardware requirements.
- Test integration of all system components at the model level, much earlier in the development cycle.
- Substantially reduce the time required for debugging.
- Improve reliability and performance.
- Consistently meet your deadlines.

Pathfinder's tools help you transform your models into executable code, predictably and accurately. Deploy faster, highly reliable embedded systems much more quickly than you thought possible.

Next Steps

If you would like to learn more about MDA and PathMATE, or learn if Pathfinder Solutions has helped an organization like yours benefit from this technology, please call or write. The white papers at www.PathfinderMDA.com contain much additional information. Most importantly, try the PathMATE toolset with real code, as outlined in this *Guide*. We can help you get started.

About Pathfinder Solutions

Headquartered outside of Boston, Massachusetts, Pathfinder Solutions provides embedded software engineers with the tools, methods and services needed to reduce development costs and improve quality. Pathfinder Solutions is an active member of the Object Management Group (OMG), and is helping to shape the future of MDA.

If you would like to learn more about MDA or PathMATE, please contact us at:

Pathfinder Solutions
33 Commercial Drive, Suite 2
Foxboro, MA 02035 USA
Phone: 508-543-7222
E-mail: info@PathfinderMDA.com

A. PathMATE Installation

To ensure a successful installation of the PathMATE toolset, please follow the steps below

Step 1. Uninstall Previous PathMATE Version

If you are installing PathMATE for the first time, skip to Step 2. If you have a previous version of PathMATE installed, uninstall it:

1. Click **Start > Settings > Control Panel** and open *Add/Remove Programs*.
2. Select *Pathfinder Solutions PathMATE for Rose* and remove it.
3. Select *PathMATE* and remove it.

Check the install directory and make sure that all files except the .lic file have been removed. For example, if you built any samples, the uninstaller will not remove the files generated from the samples.

Step 2. Install Rational Rose

Follow the prompts in the installation wizard to install Rational Rose.

Step 3. Set the Rational Rose CURDIR Symbol

The sample models and models that you create using Rose .cat files require the CURDIR symbol to be defined. To define the symbol:

1. Start Rational Rose.
2. Click **File > Edit Path Map...** in the top menu bar.
The Virtual Path Map box opens.
3. Enter *CURDIR* in the *Symbol* field.
4. Enter *&* in the *Actual Path* field and click **Add**.
The new mapping appears under *Virtual Symbol to Actual Path Mapping*.
5. Click **Close** to close the box.

Step 4. Install PathMATE

Install PathMATE in two parts: (1) **pathmate_rose-<version>.exe** integrates PathMATE's tools with the Rational Rose modeler; (2) **pathmate-<version>.exe** installs the PathMATE tools. For each program, double-click the executable file in Windows Explorer and follow the directions in the installation wizard.

Step 5. Obtain and Install a License

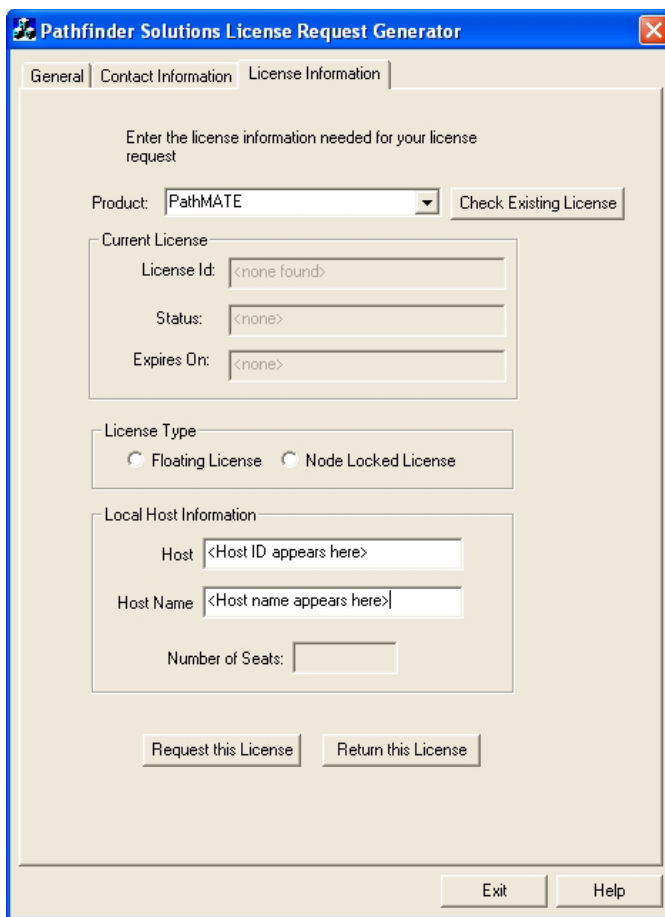
To install a PathMATE license, perform the steps below on the computer where you will use the software. If you have already received a license, simply copy the license file to *C:\pathmate\lic*.

NOTE

If you require a floating license for a license server, please contact your account manager.

To obtain and install a license:

1. Click **Start > Programs > Pathfinder Solutions > Licensing > License Request Tool** in the Windows desktop.
The Pathfinder License Request Generator opens.
2. Click the *Contact Information* tab and enter the information requested.
3. Click the *License Information* tab and select *Node Locked* License under *License Type* (Figure A-1).



The image shows a screenshot of the "Pathfinder Solutions License Request Generator" window. The window has a blue title bar and three tabs: "General", "Contact Information", and "License Information". The "License Information" tab is selected. The main area contains the following fields and controls:

- Product: A dropdown menu showing "PathMATE".
- Check Existing License: A button.
- Current License section:
 - License Id: A text box containing "<none found>".
 - Status: A text box containing "<none>".
 - Expires On: A text box containing "<none>".
- License Type section:
 - Two radio buttons: "Floating License" (selected) and "Node Locked License".
- Local Host Information section:
 - Host: A text box containing "<Host ID appears here>".
 - Host Name: A text box containing "<Host name appears here>".
 - Number of Seats: A text box.
- Request this License: A button.
- Return this License: A button.
- Exit: A button.
- Help: A button.

Figure A-1. Pathfinder License Request Generator

4. Click **Request this License**.

The Pathfinder License Generator opens.

5. Click **Save to File** in the License Generator. A Save As dialog box opens.

6. Save the license request text file in a convenient folder, such as *C:\pathmate\lic*.

7. E-mail the license request file, *pdfLicenseRequest.txt*, as an attachment to *plms@pathfindermda.com*.

You will receive a license from Pathfinder by return e-mail. When it arrives, save the .lic file in *C:\pathmate\lic*.

B. Acronyms and File Types

Table B-1 lists acronyms used in the *Quick Start Guide*:

Table B-1. List of Acronyms

Acronym	Definition
CURDIR	Current directory
MDA	Model Driven Architecture
PathMATE	Pathfinder Model Automation and Transformation Environment
PIM	Platform Independent Model
PSM	Platform Specific Model
UML	Unified Modeling Language

Table B-2 lists the file types used in the *Quick Start Guide*:

Table B-2. List of File Types

Acronym	Definition
.bat	Windows batch file
.err	PathMATE error log
.exe	Windows executable program
.lic	FlexLm license
.mdl	IBM/Rational Rose model
.pal	Platform independent action language
.typ	PathMATE type file

C. Document the System

To document your system, use the documentation pane under the browser in Rational Rose. The text you enter in the documentation pane is saved when you save your model. To create reports that include both your documentation and your UML diagrams, click **Tools > PathMATE Generate > Reports** in Rational Rose. Find the generated reports in:

C:\pathmate\samples\QuickStart\rose\reports

Table C-1 lists the main reports in the directory.

Table C-1. System Reports

Report Title	Filename
Domain Report for SimpleOven	SimpleOven_dom.rtf
Analysis Report for SimpleOven	SimpleOven_main.rtf
Class Modeling Report for SimpleOven.MicrowaveCooking	MicrowaveCooking_im.rtf
State Modeling report for SimpleOven.MicrowaveCooking	MicrowaveCooking_sm.rtf

D. SimpleOven Directory Structure

Figure D-1 shows the directory structure under *C:\pathmate\samples\QuickStart\rose* after you have created all the files related to SimpleOven.

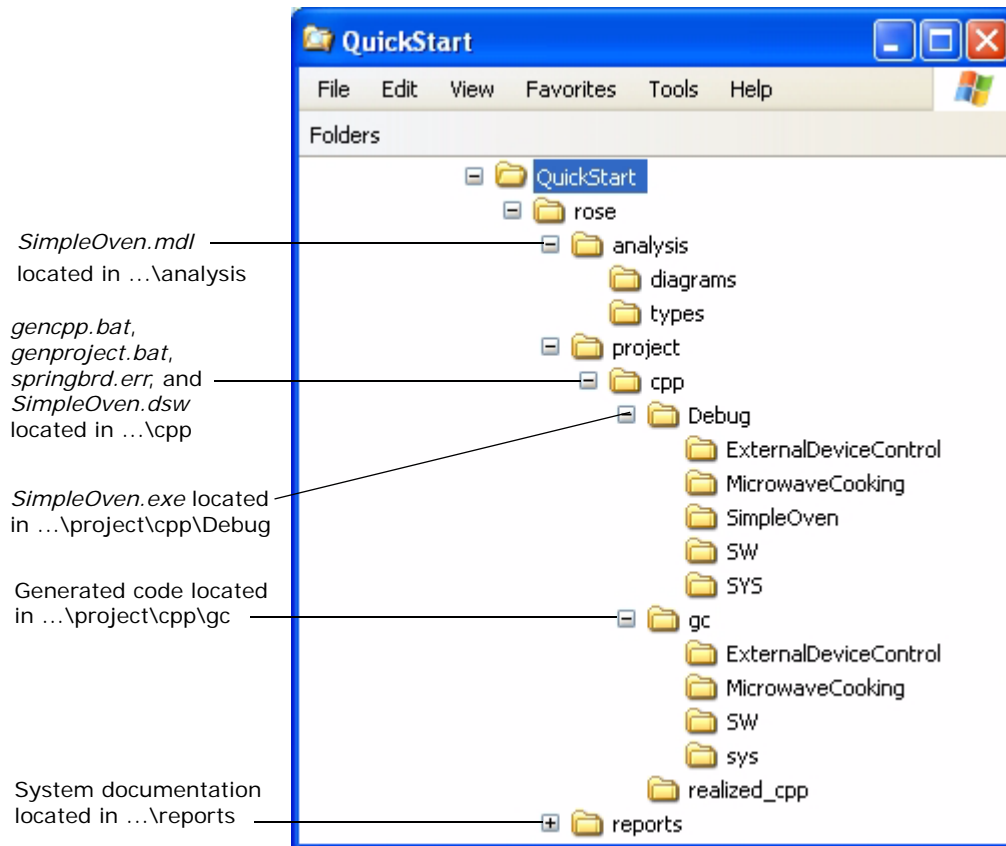


Figure D-1. QuickStart Directory Structure