Pathfinder
Solutions

Rose to RSM7 Model Conversion Guide

Version 1.2
July 6, 2007

PathMATE Technical Notes

Pathfinder Solutions LLC
33 Commercial Street, Suite 2
Foxboro, MA 02035 USA

www.PathfinderMDA.com
508-543-7222

Table Of Contents

1 Introduction

Due to limitations in the underlying RSDP infrastructure, PathMATE users converting PathMATE models from Rational Rose to Rational Software Modeler V7 with PathMATE V7 may encounter problems which currently require manual intervention. This guide identifies those problems and provides methods to correct converted models.

2 Initial Conversion Steps

To start the conversion process, use the directions provided in the PathMATE on-line help section titled “Model Conversion”. This help page links to several other pages, which explain the procedure to:

1. Prepare a Rose model to import into RSM7,
2. Import the model into a UML2 model in RSM7, and
3. Convert it to a PathMATE model.

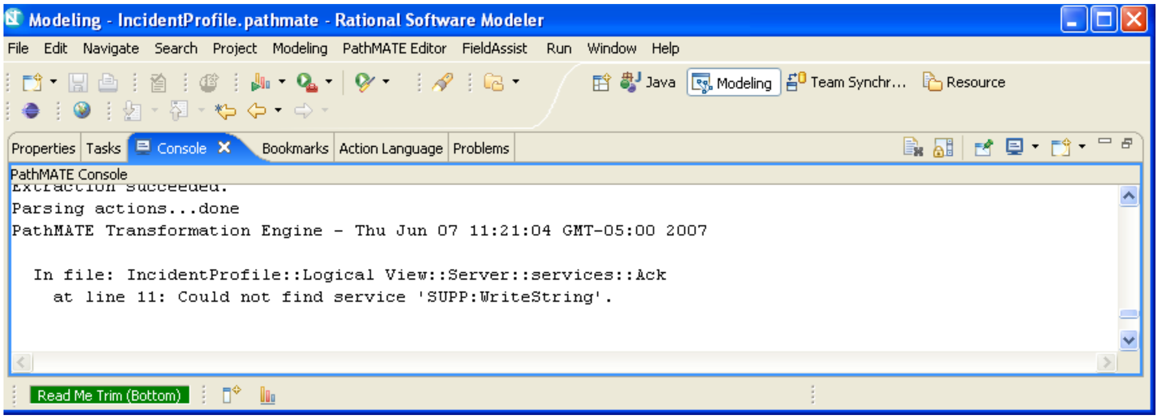
During the import process the user will be able to select which profiles are imported and associated with the imported model. It is recommended that only user created profiles and the PathMATE profile be imported.

Once the model is converted, run a transformation to detect errors in the model. The following sections of this guide outline solutions to errors caused by the conversion process.

3 Implementation Name vs. Full Domain Name in Action Language

3.1 Overview

In Rose, some models may have references to domains via the domain’s implementation name instead of the full name. This problem is found in the action language and presents it self in an error message similar to this:



```
Extraction succeeded.
Parsing actions...done
PathMATE Transformation Engine - Thu Jun 07 11:21:04 GMT-05:00 2007

  In file: IncidentProfile::Logical View::Server::services::Ack
    at line 11: Could not find service 'SUPP:WriteString'.
```

Figure 3-1 Console View

3.2 Rose Perspective

In Rose the transformation engines use implementation names as references the desired domains or classes.

3.3 RSM7 Perspective

In RSM7 7 the action language needs to reference other Domains and Classes using their full names.

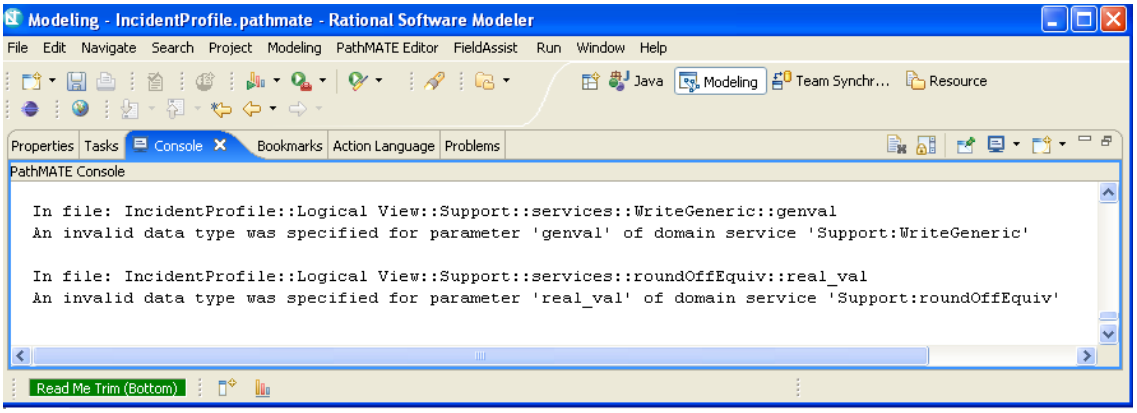
3.4 How to Fix

In order to Fix these issues, one has to go into each location in which there is an Action Language error and correct it by hand. This is done by converting the implementation name – typically an acronym – into its full name.

4 Types Lost or Corrupted

4.1 Overview

Some data items loose their types during conversion. This conversion issue appears for certain defined PathMATE define types. It is caused by a lack of transferring type names from the Rose model to the RSM7 model and presents it self in error output similar to this:



```
In file: IncidentProfile::Logical View::Support::services::WriteGeneric::genval
An invalid data type was specified for parameter 'genval' of domain service 'Support:WriteGeneric'

In file: IncidentProfile::Logical View::Support::services::roundOffEquiv::real_val
An invalid data type was specified for parameter 'real_val' of domain service 'Support:roundOffEquiv'
```

Figure 4-1 Console View

4.2 Rose Perspective

In Rose the type is simply specified in the details section of the properties of classes with their name and type.

4.3 RSM7 Perspective

In RSM7 7 these parameters are all found under their classes and the type is displayed and specified in the properties tab.

4.4 How to Fix

To Fix this problem the parameters with incorrect or empty type fields must be found in the model and corrected using the original Rose model as a reference.

4.4.1 Common Types

<Primitive Type> GenericValue

<Primitive Type> Real

<Primitive Type> Character

<Primitive Type> Handle

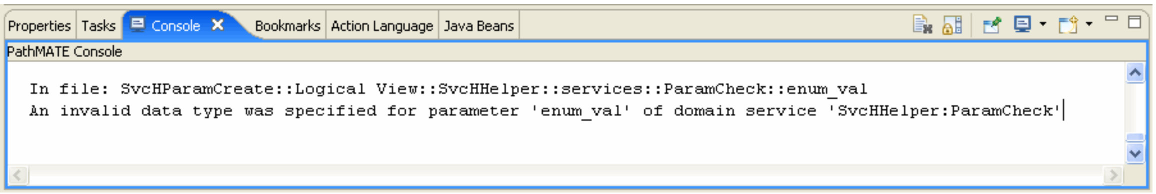
<Primitive Type> InccidentHandle (ServiceHandle in Rose)

Model Specific Files

5 Domain Types Not Visible

5.1 Overview

Some user defined data types contained within a domain that are used globally (in other domains) are not imported as public types. This issue presents itself during transformation in error output similar to this:



```
In file: SvcHParamCreate::Logical View::SvcHHelper::services::ParamCheck::enum_val
An invalid data type was specified for parameter 'enum_val' of domain service 'SvcHHelper:ParamCheck'
```

Figure 5-1 Console View

5.2 Rose Perspective

In Rose the system types were defined in separate files. Also they did not carry a private/public attribute.

5.3 RSM7 Perspective

In RSM7 these Domain types are found in the model file and can be seen in their specific Domain. The issue is encountered when these Domain Types are imported and assigned to be private, and the importer does not check to determine if the domain type is referenced else where in the model.

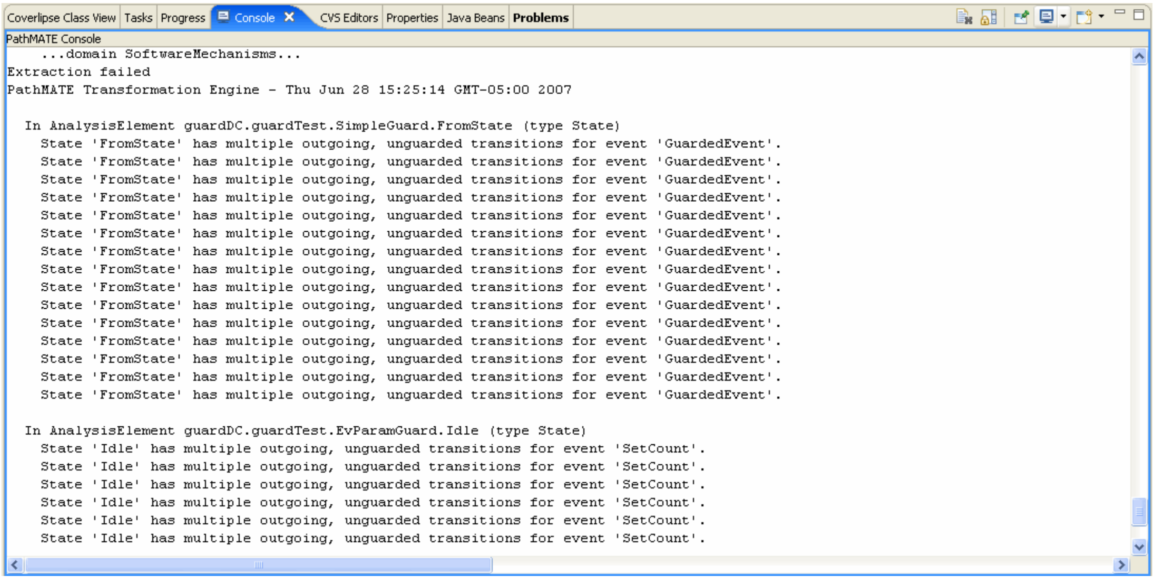
5.4 How to Fix

The simple Fix for this is to change the visibility property of the affected types to “public”.

6 Guarded Transitions Improperly Configured

6.1 Overview

Some state chart transition guards are imported, and are displayed in the project explorer but not on the diagram. Newly created guards show up in both locations. This problem presents itself in an error output similar to this:



```
    ...domain SoftwareMechanisms...
Extraction failed
PathMATE Transformation Engine - Thu Jun 28 15:25:14 GMT-05:00 2007

  In AnalysisElement guardDC.guardTest.SimpleGuard.FromState (type State)
    State 'FromState' has multiple outgoing, unguarded transitions for event 'GuardedEvent'.
    State 'FromState' has multiple outgoing, unguarded transitions for event 'GuardedEvent'.
    State 'FromState' has multiple outgoing, unguarded transitions for event 'GuardedEvent'.
    State 'FromState' has multiple outgoing, unguarded transitions for event 'GuardedEvent'.
    State 'FromState' has multiple outgoing, unguarded transitions for event 'GuardedEvent'.
    State 'FromState' has multiple outgoing, unguarded transitions for event 'GuardedEvent'.
    State 'FromState' has multiple outgoing, unguarded transitions for event 'GuardedEvent'.
    State 'FromState' has multiple outgoing, unguarded transitions for event 'GuardedEvent'.
    State 'FromState' has multiple outgoing, unguarded transitions for event 'GuardedEvent'.
    State 'FromState' has multiple outgoing, unguarded transitions for event 'GuardedEvent'.
    State 'FromState' has multiple outgoing, unguarded transitions for event 'GuardedEvent'.
    State 'FromState' has multiple outgoing, unguarded transitions for event 'GuardedEvent'.
    State 'FromState' has multiple outgoing, unguarded transitions for event 'GuardedEvent'.
    State 'FromState' has multiple outgoing, unguarded transitions for event 'GuardedEvent'.
    State 'FromState' has multiple outgoing, unguarded transitions for event 'GuardedEvent'.

  In AnalysisElement guardDC.guardTest.EvParamGuard.Idle (type State)
    State 'Idle' has multiple outgoing, unguarded transitions for event 'SetCount'.
    State 'Idle' has multiple outgoing, unguarded transitions for event 'SetCount'.
    State 'Idle' has multiple outgoing, unguarded transitions for event 'SetCount'.
    State 'Idle' has multiple outgoing, unguarded transitions for event 'SetCount'.
    State 'Idle' has multiple outgoing, unguarded transitions for event 'SetCount'.
    State 'Idle' has multiple outgoing, unguarded transitions for event 'SetCount'.
```

Figure 6.1 Transformation Errors

6.2 Symptoms in RSM7

In RSM7 these issues present itself by showing the (imported) guards in the project explorer and not on the diagram.

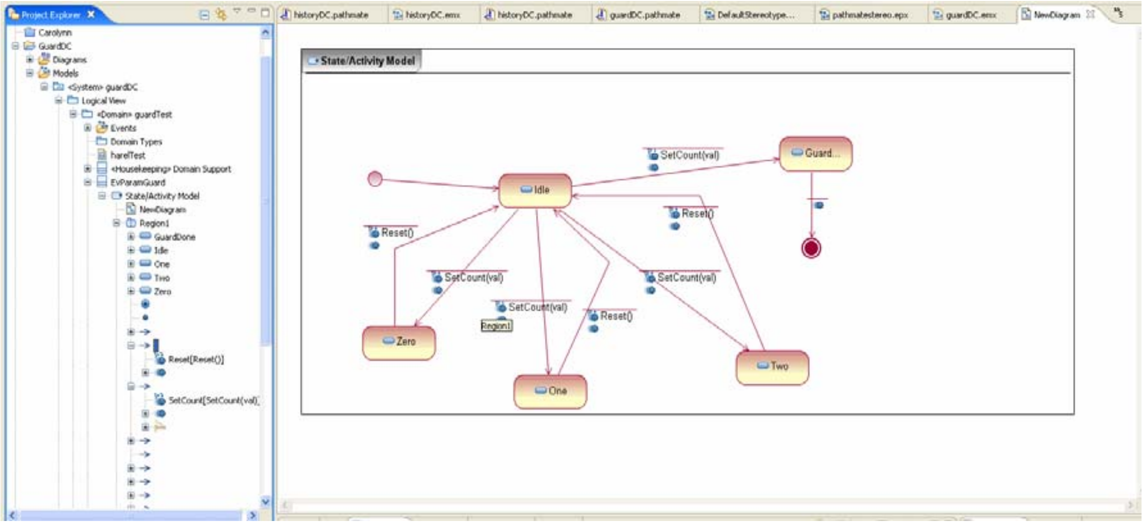


Figure 6-2 Current / Incorrect Diagram

Note that newly created guards do show up in both locations.

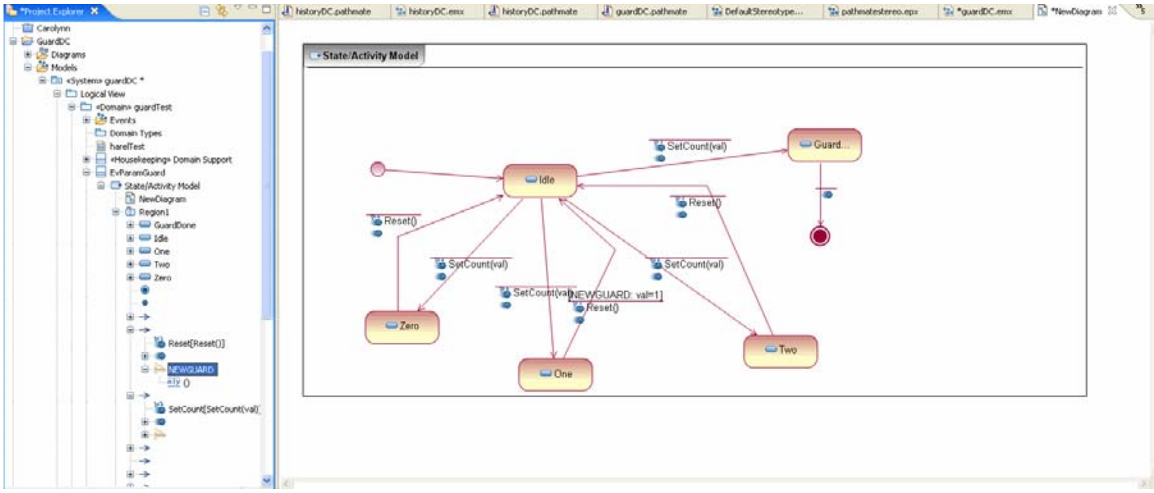
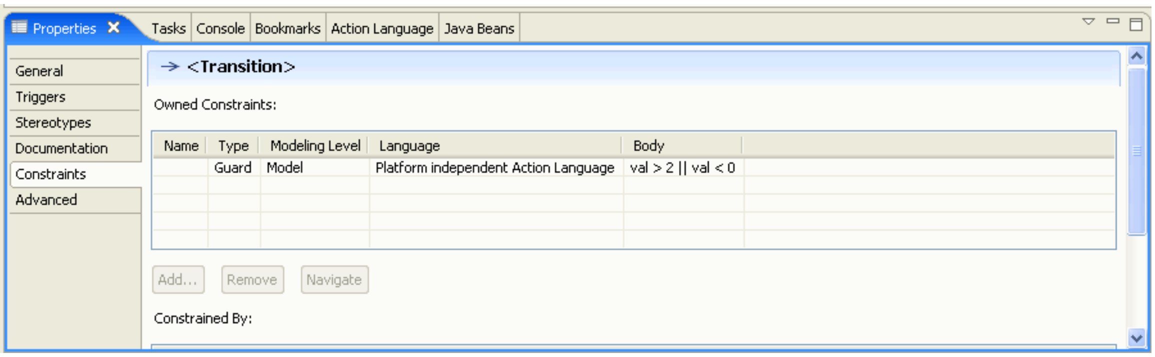


Figure 6-3 When a Guard is manually Added

6.3 How to Fix

Locate the transitions in the project explorer. Follow these steps for each transition that should be guarded.

Step 1:
Look at the properties for the transition. Select the Constraints tab, notice that a unnamed constraint will be there.



<Transition>

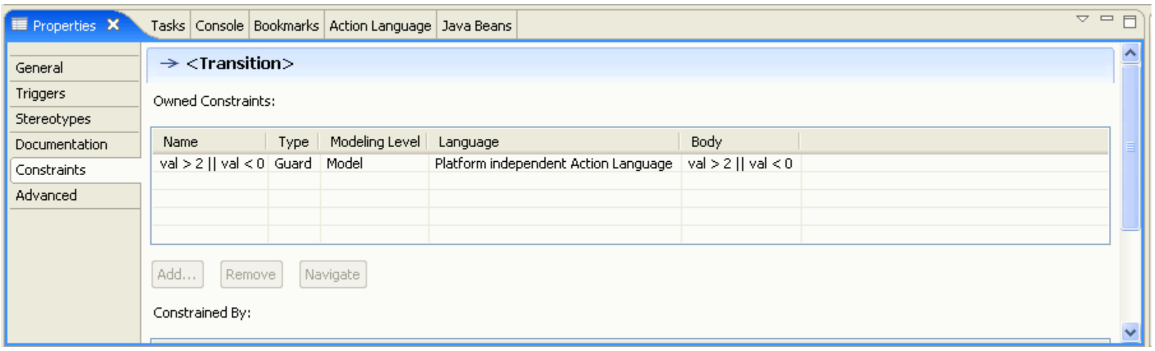
Owned Constraints:

Name	Type	Modeling Level	Language	Body
	Guard	Model	Platform independent Action Language	val > 2 || val < 0

Constrained By:

Figure 6-4 Imported Constraints

Name this constraint, after which the constraint will be recognized by the model (however it is still not displayed).



<Transition>

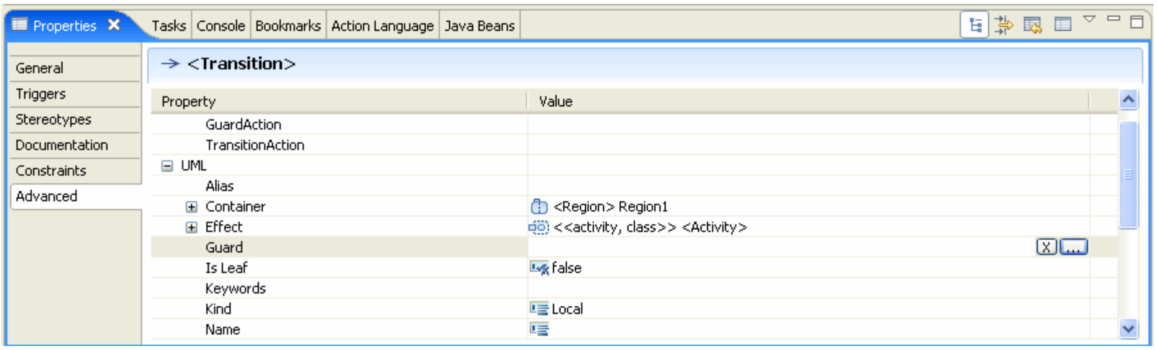
Owned Constraints:

Name	Type	Modeling Level	Language	Body
val > 2 || val < 0	Guard	Model	Platform independent Action Language	val > 2 || val < 0

Constrained By:

Figure 6-5 Correct Constraints

Step 2:
Now select the advanced tab. Under the UML properties there is a row for Guard which is empty.

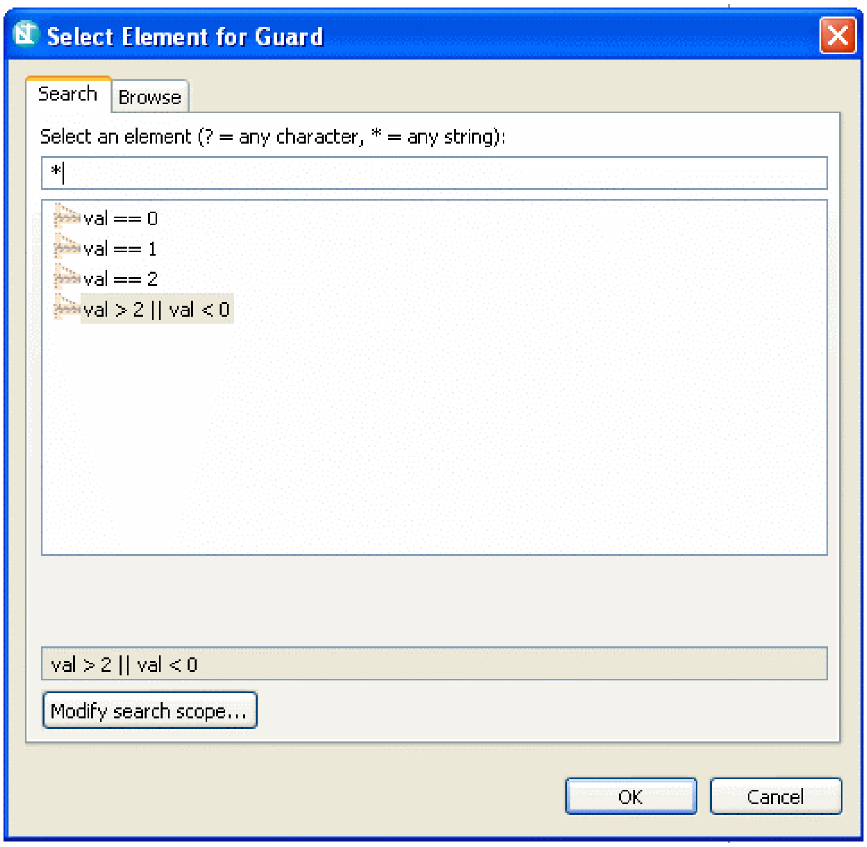


→ <Transition>

Property	Value
GuardAction	
TransitionAction	
⊟ UML	
Alias	
⊞ Container	<Region> Region1
⊞ Effect	<<activity, class>> <Activity>
Guard	
Is Leaf	false
Keywords	
Kind	Local
Name	

Figure 6-6 Current Properties>Advanced

Click on the … to select the Guard. When the prompt appears enter a * and select the appropriate Guard from the list. (NOTE: Only the Guards which you have named using step 1 will appear)



Select an element (? = any character, * = any string):

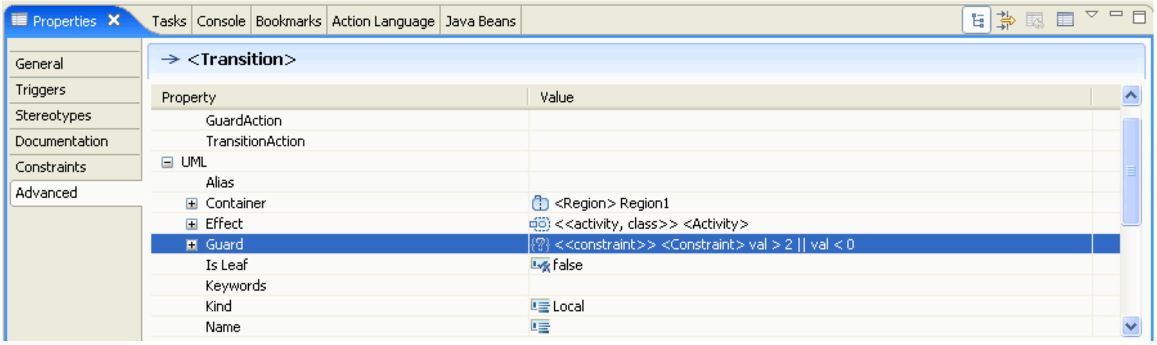
*

val == 0
val == 1
val == 2
val > 2 || val < 0

val > 2 || val < 0

Figure 6-7 Guard Selector

After this the UML>Guard Property will be filled.



<Transition>

Property	Value
GuardAction	
TransitionAction	
⊟ UML	
Alias	
⊞ Container	<Region> Region1
⊞ Effect	<<activity, class>> <Activity>
⊞ Guard	<<constraint>> <Constraint> val > 2 || val < 0
Is Leaf	false
Keywords	
Kind	Local
Name	

Figure 6-8 Correct Properties>Advanced

And the Guard will appear on the diagram.

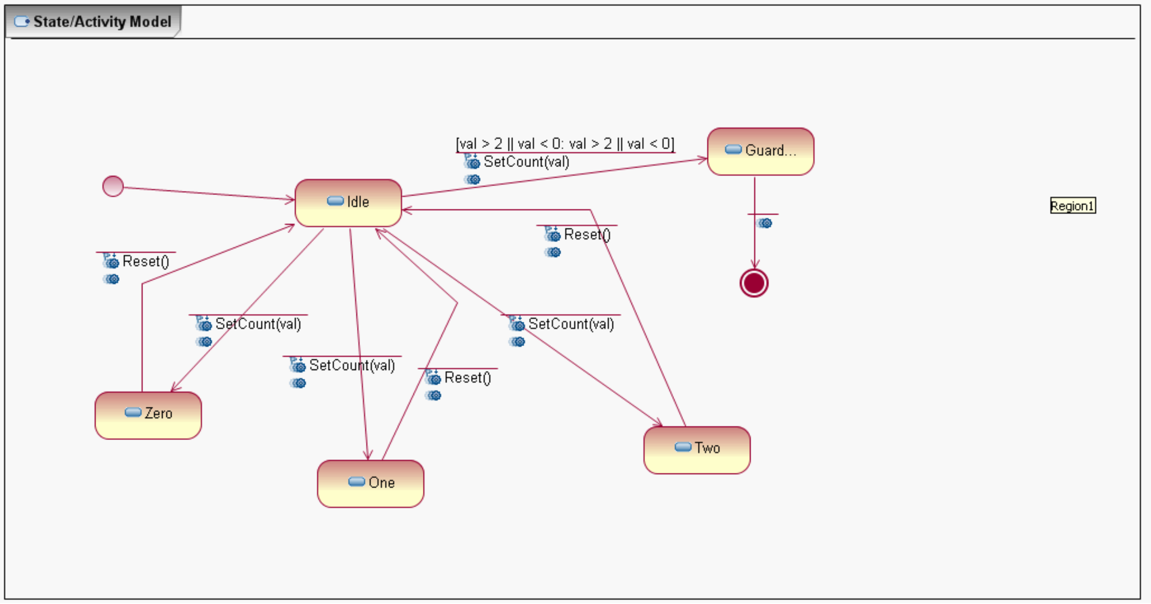


Figure 6-9 Diagram w/ Currently Corrected Guard

Remember to complete these steps for every transition, as missing one transition can result in an error for every transition.