Enterprise Architect Import Plugin Documentation 19.0 beta

User Guide

Contents

ear

1 User Guide

ear

1.1 Getting started

This user guide will serve as an introduction to the Enterprise Architect Import plugin features and functionality. It also shows how to use it to work with your modeling tools.

Related pages

1.1.1 Introduction

MagicDraw has the functionality to import UML models that conform to various XMI versions (including XMI 2.1) from Sparx Systems Enterprise Architect (EA), a modeling and visualization tool based on the UML 2.3 standard. EA has the ability to import and export XMI compliant models; therefore, you can use EA to import UML2.1(XMI2.1). However, the XMI models exported from EA contain some XMI conflicts and EA-specific data that do not conform to the UML standards.

Enterprise Architect Import Plugin for MagicDraw allows you to migrate the XMI models from EA to MagicDraw flawlessly by using an additional transformation process with a set of mapping rules.

The main purpose of Enterprise Architect Import Plugin is to help MagicDraw users who need to import models from EA manage the conflicts that can cause problems during loading the XMI models to MagicDraw, as well as transforming some EA-specific data into UML elements with stereotypes.

In addition to the ability to import model elements, the plugin also allows for the import of diagrams. The current plugin version supports:

- Class diagrams
- Package diagrams
- Object diagrams
- Component diagrams
- Deployment diagrams
- Use Case diagrams
- Activity diagrams
- Sequence diagrams
- Communication diagrams
- StateMachine diagrams
- CompositeStructure diagrams
- InteractionOverview diagrams

Related pages

Plugin information (see page 5)

Installation (see page 6)

1.1.2 Plugin information

Enterprise Architect Import Plugin for MagicDraw supports Enterprise Architect Versions 7.1, 7.5, and 8.0 (most of the testing procedures performed on EA 7.1.833 and EA 7.5.847). The plugin helps you import and transform an EA exported XMI using the UML2.1(XMI2.1) option into a MagicDraw file (*.mdxml).

Your imported models will include the following details:

- UML models
- Profiles
- Stereotype usage information
- EA-specific data:
    - Constraints
    - Requirements
    - Scenarios
    - Files
    - Requirements (External)
    - Changes
    - Issues
- Diagram information
    - Class diagrams
    - Package diagrams
    - Object diagrams
    - Component diagrams
    - Deployment diagrams
    - Use Case diagrams
    - Activity diagrams
    - Sequence diagrams
    - Communication diagrams
    - StateMachine diagrams
    - CompositeStructure diagrams
    - InteractionOverview diagrams
- SysML (SysML 1.1 model from EA will be transformed to MagicDraw SysML.)

Related pages

Introduction (see page 4)

Installation (see page 6)

1.1.3 Installation

The Enterprise Architect Import plugin comes bundled with your modeling tool. When you install the tool, the plugin is automatically installed as well.

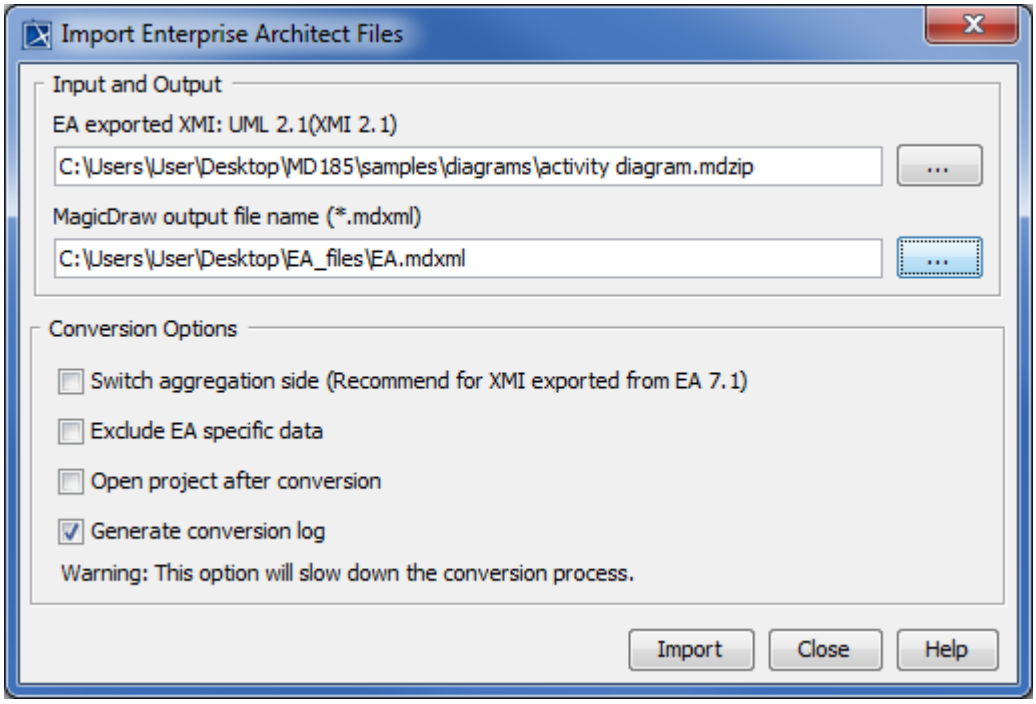
Related pages

Introduction (see page 4)

Plugin information (see page 5)

1.2 Working with Enterprise Architect Import Plugin

The Enterprise Architect Import (EA) plugin will automatically load when you start MagicDraw. You can use it to import UML 2.1 XMI 2.1 to your modeling tool.



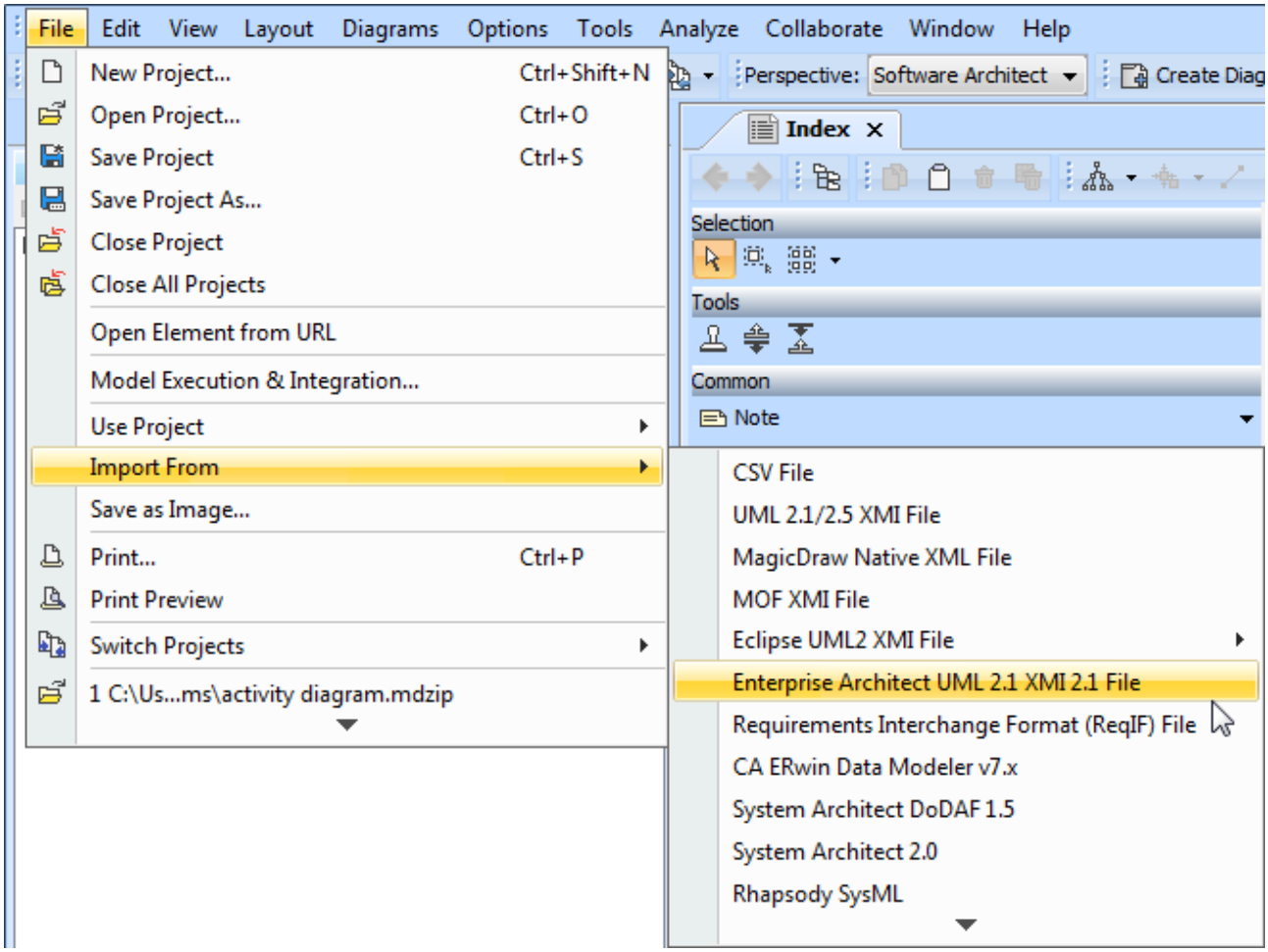
1.2.1 Using the Enterprise Architect Import plugin to import files.

The following table describes the UI components of the **Import Enterprise Architect Files** dialog.

UI component	Function
EA exported XMI: UML 2.1 (XMI2.1)	To specify an EA exported XMI file. You can click ... to select the file.
MagicDraw output filename (*.mdxml)	To specify a MagicDraw output filename. You can click ... to select the file.
Switch aggregation side	To configure the **aggregation switch-side**. This option is recommended for XMI files exported from EA 7.1.
Exclude EA specific data	To exclude all EA-specific data from being imported (see Transforming EA Specific Data (see page 11) to see a list of EA specific-data that can be transformed into UML elements with stereotypes).
Open project after conversion	To load the output project file once the conversion process has been completed.
Generate conversion log	To generate a conversion log and save it in the same folder as the MagicDraw output file. The same conversion information will also be displayed on the MagicDraw messages window.

To open the **Import Enterprise Architect Files** dialog

- On the MagicDraw main menu, click **File** > **Import From** > **Enterprise Architect UML2.1 XMI2.1 File** to open the **Import Enterprise Architect Files** dialog.



1.2.2 The Enterprise Architect Import plugin menu.

To import an EA project

1. Either type an EA exported XMI file or click ... to select the file.

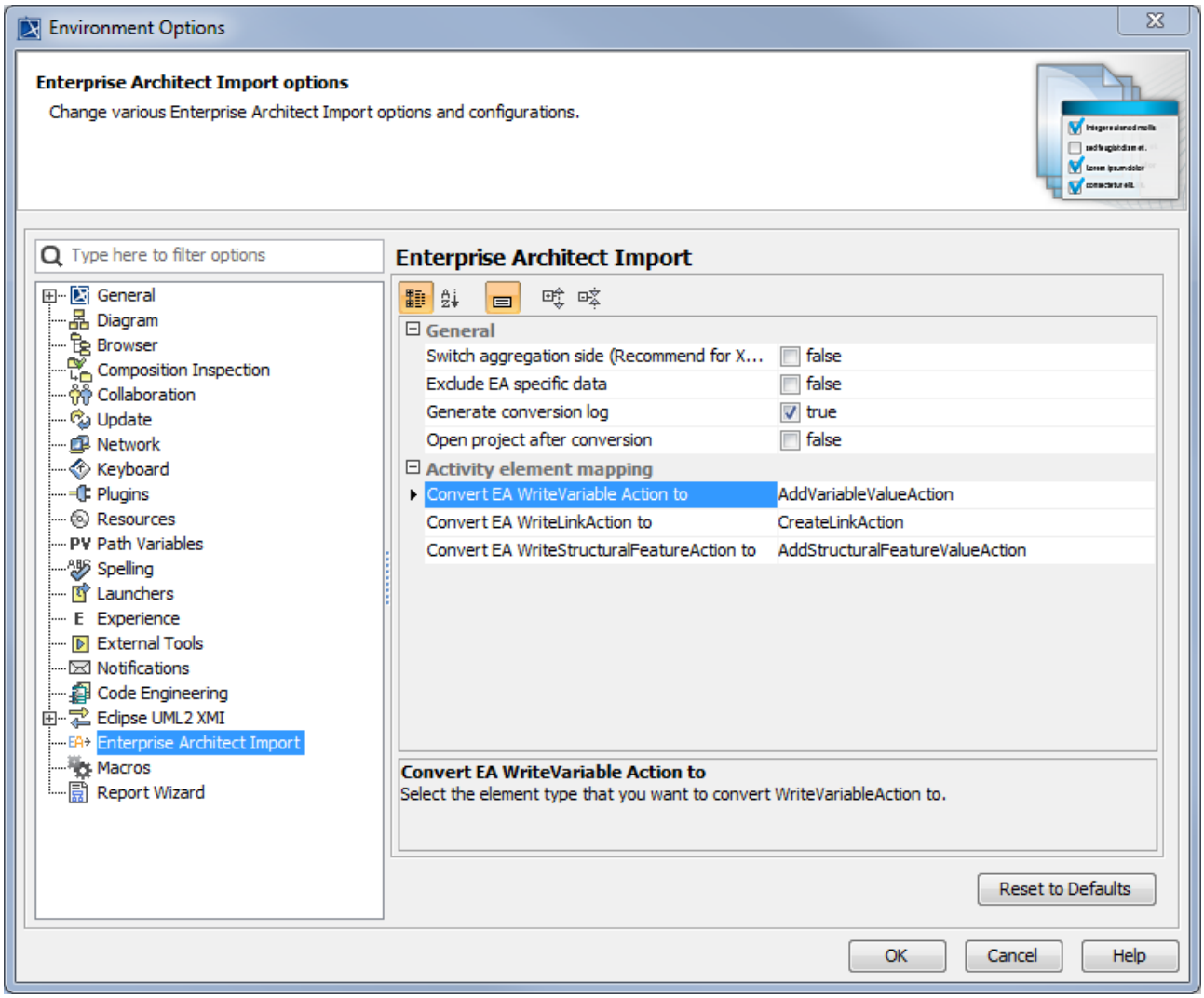
> **Note**
>
> Enterprise Architect Import Plugin supports the EA XMI files exported with the option specified as: XMI Type = **UML2.1(XMI 2.1).**

2. Either type a MagicDraw output filename or click ... to select the file.
3. Select the conversion options.
4. Click Import to start importing the file.

On this page

1.2.3 Conversion options

In addition to the options that you have in the **Import Enterprise Architect File** dialog, several other conversion options exist in the MagicDraw **Environment Options** dialog.



The EA options in the Environment Options dialog.

The transformation options are classified into two groups: General and Activity element mapping.

General

The options available in the **General** group are the same as those in the **Import Enterprise Architect Files** dialog.

Activity element mapping option

The Activity element mapping group provides options to convert the EA elements to other element types. The following table shows the Activity Element Mapping options.

Property name	Function
Convert EA WriteVariableAction to	To convert EA WriteVariableAction to either AddVariableValueAction or RemoveVariableValueAction.
Convert EA WriteLinkAction to	To convert EA WriteLinkAction to either CreateLinkAction or DestroyLinkAction.
Convert EA WriteStructuralFeatureAction to	To convert EA WriteStructuralFeatureAction to either AddStructuralFeatureValueAction or RemoveStructuralFeatureValueAction.

To view the other Enterprise Architect Import options in the **Environment Options** dialog

1. Click **Options** > **Environment** on the MagicDraw main menu to open the **Environment Options** dialog.
2. Select **Enterprise Architect Import** from the tree menu on the left-hand side of the dialog.

On this page

Related page

Conversion options (see page 9)

1.2.4 Conversion messages

The Enterprise Architect Import Plugin consists of a series of XMI conversions. Each conversion will be reported to the MagicDraw **Messages** window and also saved to a log file if the **Generate conversion log** option is selected.

The conversion log will be saved in the same directory as the MagicDraw output file using the same name, but with a different .log extension.

Note

You can open the MagicDraw **Messages** window by pressing **Ctrl** + **M**.

Related page

C (see page 10)onversion options (see page 9)

1.3 Transforming EA Specific Data

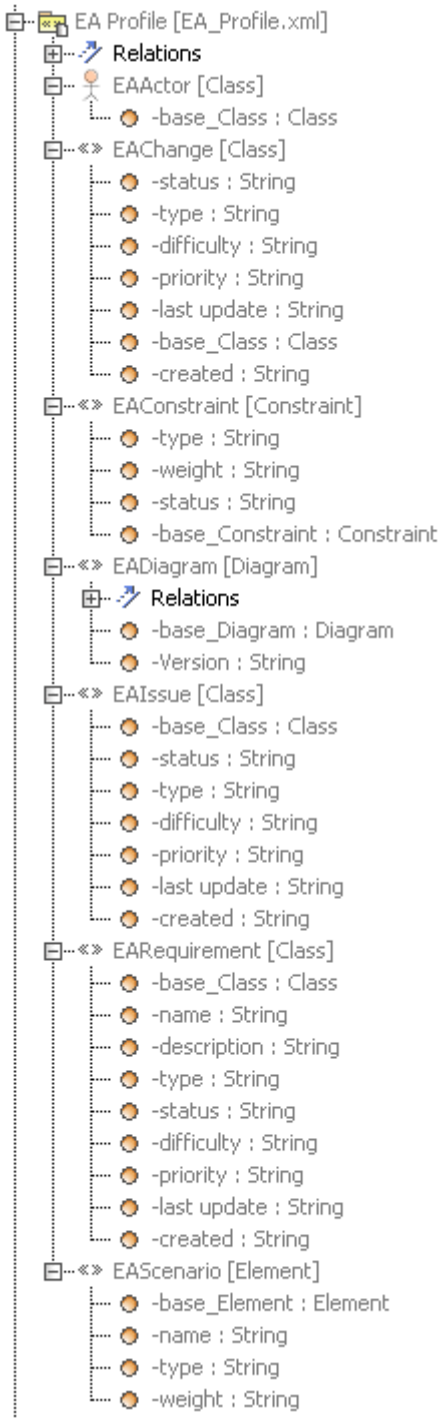
In addition to UML data, each EA-exported XMI contains EA-specific information. The Enterprise Architect Import Plugin can transform this particular information into UML elements with the stereotypes applied if you include EA-specific data before importing the XMI file. This data includes:

- Constraints: name, description, type, weight, and status.
- Requirements: name, description, type, status, difficulty, priority, and last update.
- Scenarios: name, description, type, and weight.
- Files: file path type.
- Requirements (External): type, status, difficulty, priority, last update, created, and note.
- Changes: type, status, difficulty, priority, last update, created, and note.
- Issues: type, status, difficulty, priority, last update, created, and note.

Note

You can access and specify the EA information in the **Property** dialog in EA.

To include EA-specific data in the transformation process, the plugin creates a set of stereotypes and tag definitions as the EA Profile.



1.3.1 The EA Profile in a treeview.

1.3.2 Constraints

Each EA constraint will be transformed into a UML constraint and <<EAConstraint>> will be applied to the constraint. The properties of an EA constraint will be mapped either to the properties of a UML constraint or to the tag values of <<EAConstraint>>. The following table shows the constraint mapping details.

EA	MagicDraw
name	The name property of a UML constraint.
description	EAConstraint::type tag value.
type	EAConstraint::weight tag value.
weight	EAConstraint::status tag value.
constraint owner	Constrained Element property point to the constraint owner.

1.3.3 Requirements

Each EA requirement will be transformed into a UML Class. Because a requirement cannot be created in an element that is the owner of a Class in EA, the transformed requirement will be kept in a separate Package, named **EA Requirement**. A Realization will then be created from the owner of the requirement into a transformed requirement. See the following table for details.

EA	MagicDraw
name	EARequirement :: name tag value
description	EARequirement :: description tag value
type	EARequirement :: type tag value
status	EARequirement :: status tag value
difficulty	EARequirement :: difficulty tag value
priority	EARequirement :: priority tag value
last update	EARequirement :: name update value

1.3.4 Scenarios

Each EA scenario will be transformed into a UML Comment and <<EAScenario>> will be applied to the comment. The properties of a scenario will be mapped either to the properties of each UML Comment or to the tag values of <<EAScenario>>. See the following table for details.

EA	MagicDraw
name	EAScenario::name tag value
description	The Body property of a UML Comment.
type	EAScenario :: type tag value
weight	EARequirement :: weight tag value

subject	An annotated Element property pointing to an EA subject element.

1.3.5 Files

EA can add files to a UML element. The information will be transformed into a Hyperlink in MagicDraw.

EA	MagicDraw
Local file	File
Web address	Webpage.

1.3.6 Requirements (external)

An EA-created Requirement differs from the one you create as an internal element for each element. EA requirements will appear in the Project Browser and can be pasted on a diagram. Each EA Requirement will be transformed into a Class and <<EARequirement>> will be applied to the requirement.

EA	MagicDraw
type	EARequirement :: type tag value
status	EARequirement :: status tag value
difficulty	EARequirement :: difficulty tag value
priority	EARequirement :: priority tag value
last update	EARequirement :: last update value
created	EARequirement :: created tag value
note	Documentation

1.3.7 Changes

EA can create a Change and will export it as a Class. The Class information will be transformed into the <<EAChange>> tag values. See the following table for details.

EA	MagicDraw
type	EAChange :: type tag value
status	EAChange :: status tag value
difficulty	EAChange :: difficulty tag value
priority	EAChange :: priority tag value
last update	EAChange :: last update value

created	EAChange :: created tag value
note	Documentation

1.3.8 Issues

EA can create an Issue and will export it as a Class. The Issue information will be transformed into the <<EAIssue>> tag values. See the following table for details.

EA	MagicDraw
type	EAIssue :: type tag value
status	EAIssue :: status tag value
difficulty	EAIssue :: difficulty tag value
priority	EAIssue :: priority tag value
last update	EAIssue :: last update value
created	EAIssue :: created tag value
note	Documentation

On this page

1.4 Importing Diagrams

Enterprise Architect Import Plugin allows you to import diagrams. The imported diagram information includes:

1.4.1 Geometry Properties

The geometry properties imported to MagicDraw are:

- Positions on a diagram (for shape elements)
- Width and height (for shape elements)
- Path break points (for link elements)

> **Note**
>
> Other display properties can override geometry information.For example, if an imported element width is shorter than the required width to display text on the element, the width will be adjusted automatically.

1.4.2 Color Properties

Color properties will be imported along with the diagrams to MagicDraw. Each color property has a different name in MagicDraw. The following table shows color properties mapping.

EA	MagicDraw
Background	Fill color
Border color (for shape element)	Pen color
Font color	Text color
Line color (for link element)	Pen color

1.4.3 Display Properties

The display properties in EA can be categorized into three groups: shape, link, and diagram. Only those properties that correspond to MagicDraw will be imported. For example, the **Show Diagram Details** property in EA will be imported as the **Show Diagram Info** property in MagicDraw.

On this page

1.5 Special Transformation

An EA-exported XMI contains both non-standard UML elements and elements that can break the XMI schema. To retain standard UML elements and keep the XMI schema intact, the Enterprise Architect Import Plugin applies specific transformation rules. The following sections describe how the plugin transforms each model element, enabling you to import a complete XMI model that conforms to UML standards.

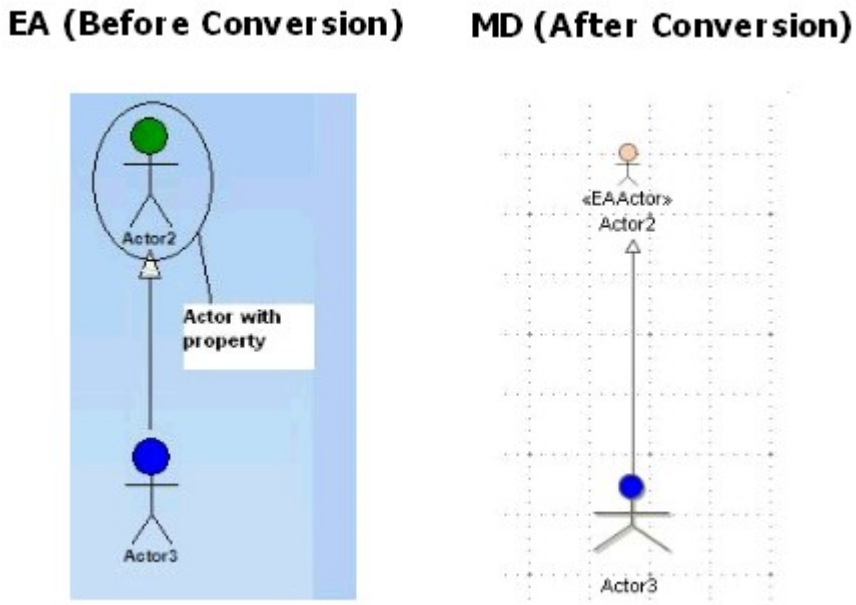
On this page

1.5.1 Use Case diagram elements

This page describes all Use Case diagram elements.

Actor with Properties

An Actor with properties will be transformed into a Class with the **EAActor** stereotype.



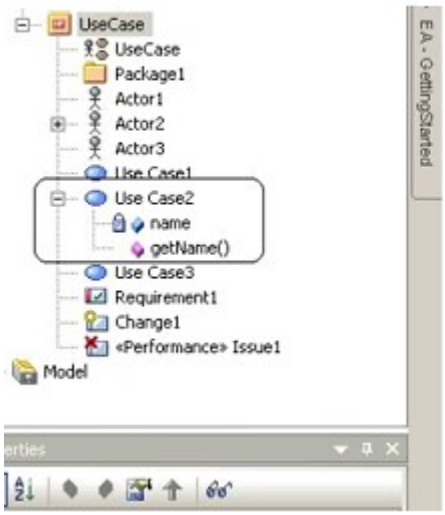
An actor with properties.

> **Note**
>
> An Actor that has been converted to a Class with <<EAActor>> will not display some properties (such as Fill Color) because the stereotype image will be shown instead.

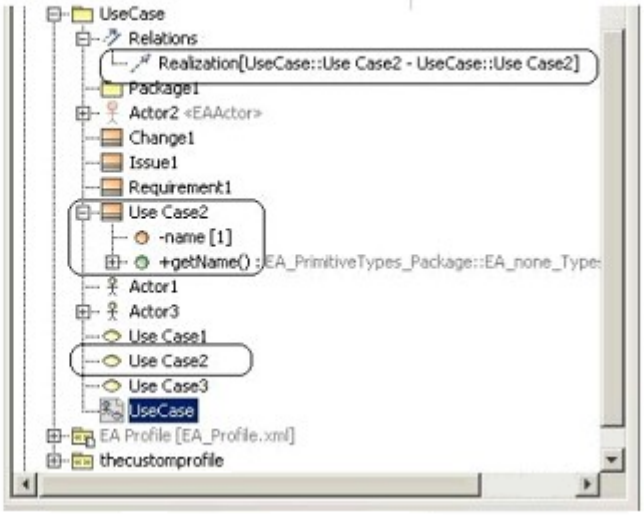
Use Case with Invalid Inner Elements

A NestedClassifier, ownedComment, ownedRule, ownedAttribute, or ownedOperation cannot be an inner element of a uml:UseCase. It will be moved to a new created realized Class.

EA (Before Conversion)



MD (After Conversion)

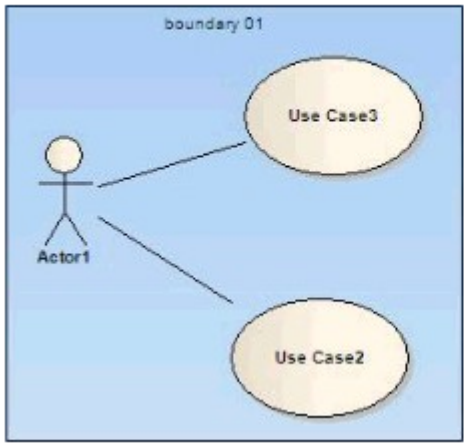


A Use Case with invalid inner elements.

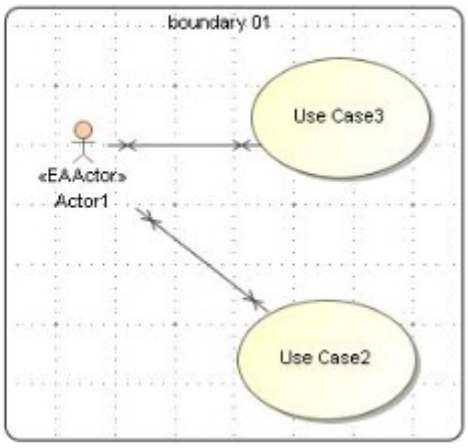
Boundary

A boundary in EA will be converted into a rectangle with rounded corners in MagicDraw. The boundary can contain inner elements. Unlike the rectangular boundary in MagicDraw, the boundary in EA will take all inner elements with it whenever it is moved.

EA (Before Conversion)



MD (After Conversion)



On this page

1.5.2 Activity diagram elements

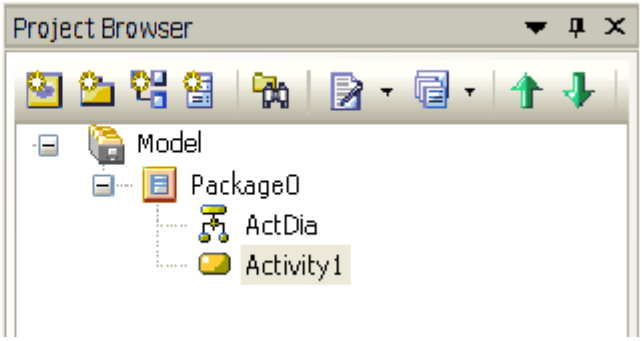
Activity

You can directly place An Activity element in EA as an element view on an Activity diagram. However, this behavior conflicts with MagicDraw and UML notation. In MagicDraw, if you drag an Activity from the containment tree to an Activity diagram, a new CallBehaviorAction view will be created and the 'Behavior' property of the CallBehaviorAction will be set to the Activity. This same behavior will be used in the import process.

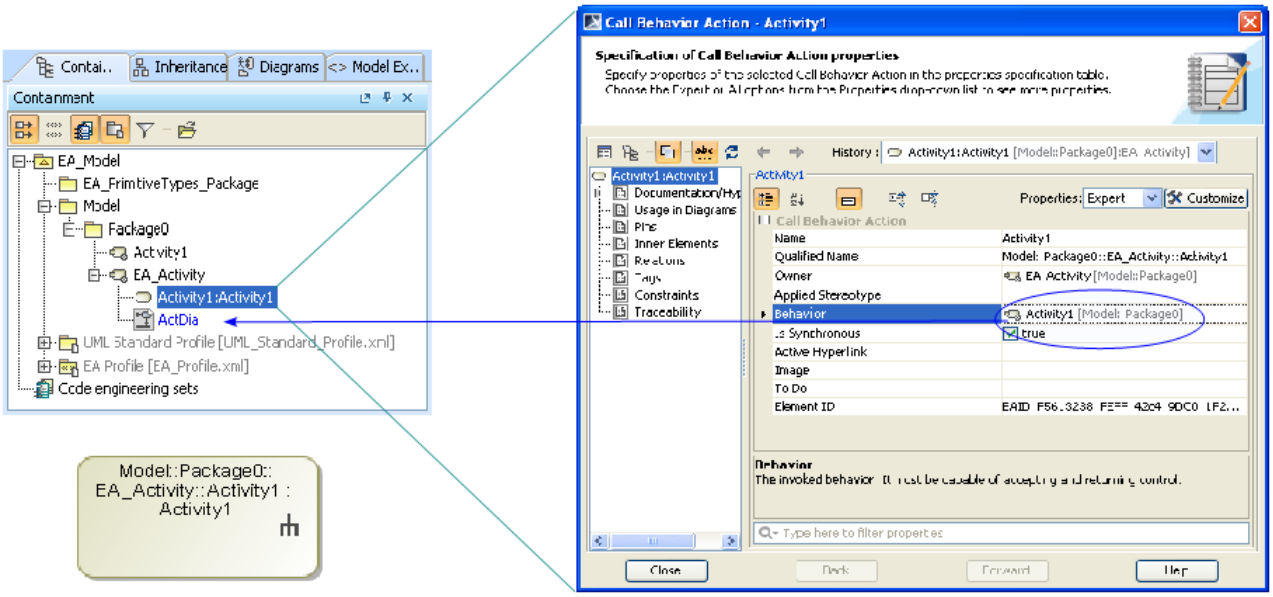
An Activity element created in EA and placed on an Activity diagram will be transformed into two elements: Activity, and CallBehaviorAction elements. Both elements will have the same name and will be linked through the property of a CallBehaviorAction element called 'Behavior'.

After transforming the element, the following transformation message will open:

```
Updated element <xmi:id>: A new CallBehaviorAction was created and its
Behavior was set to the element.
```



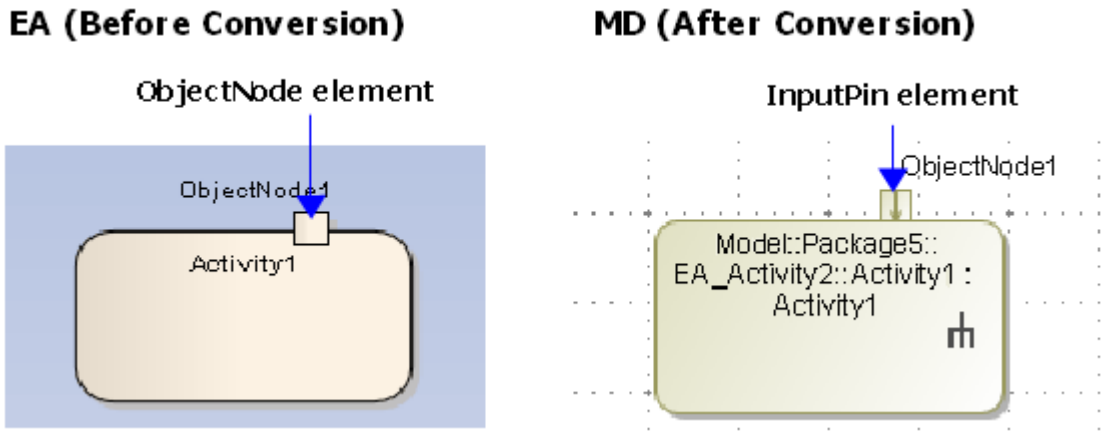
EA activity.



Activity with a New CallBehavior in MagicDraw.

> **Note**
>
> The EA Activity and CallBehaviorAction elements have similar characteristics in that you can attach a control flow to it and others. EA has its own CallBehaviorAction element.

Additionally, any ObjectNode elements attached to the Activity element will be transformed into InputPin elements and attached to the newly created CallBehaviorAction element.



An ObjectNode converted into an InputPin.

Activity diagram

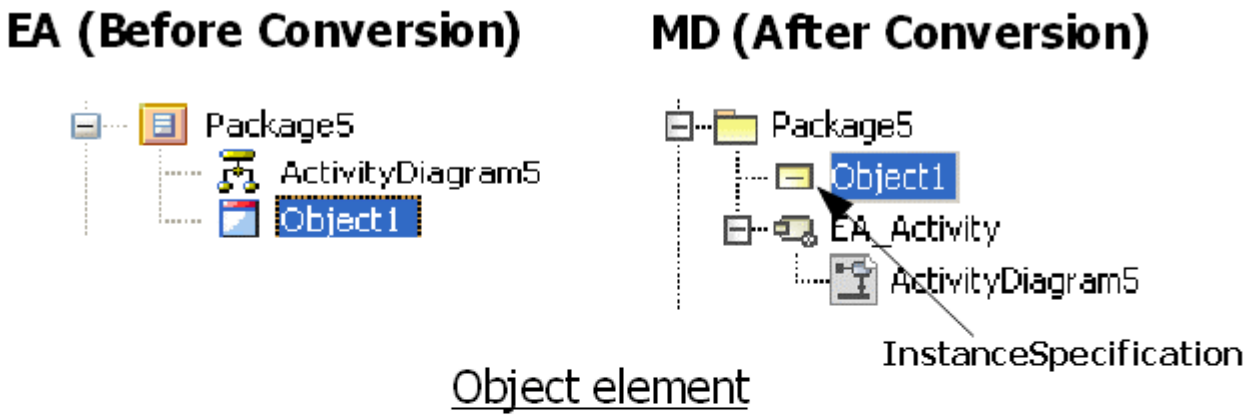
Every Activity Diagram element from EA will be placed inside an Activity element that has the same name.

Activity diagram in the MagicDraw Containment tree.

> **Note**
>
> In MagicDraw, every Activity diagram element must be placed inside an Activity element that has the same name. However, this is not the case in EA.

Object as the inner element of an Activity

Object elements inside an Activity element in EA will be removed.



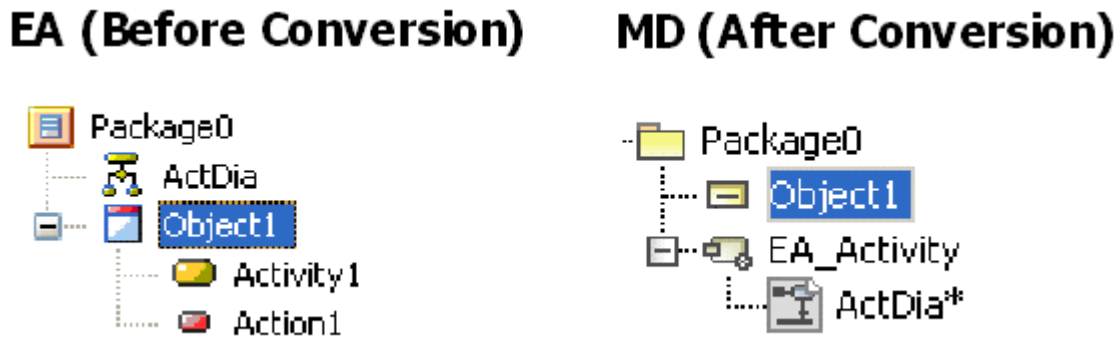
Object element

Object element in Activity element

Object elements transformation.

> **Note**
>
> Object elements in MagicDraw have their XMI types defined as 'uml:Central- BufferNode'. However, those in EA have their XMI types defined as 'uml:InstanceSpecification', which do not belong to an Activity diagram.

An Object element containing any ActivityDiagram-related elements will be removed.



Object Containing Activity-related Elements.

> **Note**
>
> In MagicDraw, an Object element (CentralBufferNode) is not allowed to contain elements other than comments and hyperlinks.

Synch Node

A Synch element in EA will be transformed into a Join element in MagicDraw. It will look exactly like a Fork/Join element.

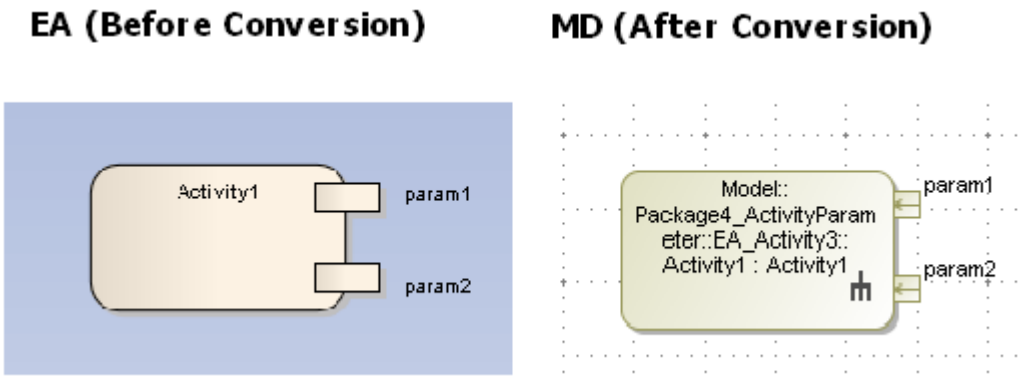
EA (Before Conversion) **MD (After Conversion)**

Synch element transformation.

> **Note**
>
> A MagicDraw Fork/Join element (whose type is uml:ForkNode) can be used to construct either a Fork and Join node in an Activity diagram. The JoinNode element (whose type is uml:JoinNode) is allowed to be placed in the Activity diagram, but the element's image will be displayed as the Fork/Join element's default image.

Activity Parameter

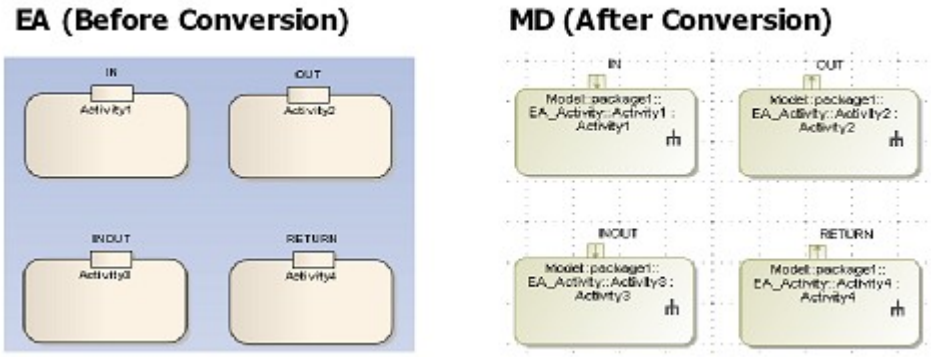
If you create an ActivityParameter element, MagicDraw will automatically create an ActivityParameterNode element to represent it. Every ActivityParameterNode element in EA will be transformed into a Pin element.



Activity parameter node.

You can specify four parameter types for each Activity Parameter element: in, out, inout, and return.

The ActivityParameterNode element of an ActivityParameter element whose parameter type is either 'in' or 'inout' will be transformed into an InputPin element. The ActivityParameterNode element of an ActivityParameter element whose parameter type is either 'out' or 'return' will be transformed into an OutputPin element.



Activity parameter type.

> **Note**
>
> Usually, if you specify the parameter type of an ActivityParameter element as 'inout', two Pin elements (InputPin and OutputPin elements) will be created for the element. Since EA will only create one ActivityParameterNode element, this element will be transformed into an InputPin element.

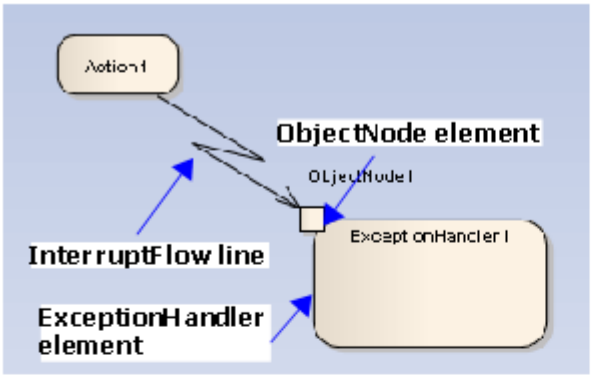
Exception Handler

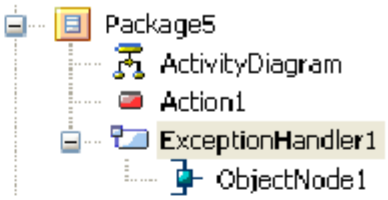
The Exception Handler element in EA differs from the UML's ExceptionHandler. This EA element will be transformed into a CallBehaviorAction element. Any ObjectNode element attached to it will be transformed into an InputPin element and any InterruptFlow line will be transformed into an ExceptionHandler line in MagicDraw (Figure 17).

After completing the transformation, the following transformation messages will open:

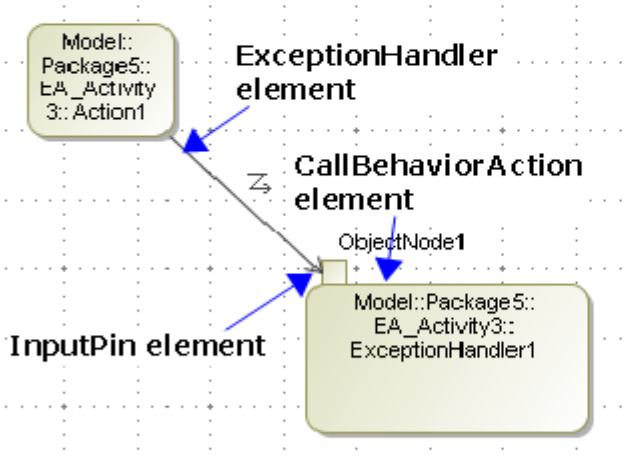
> `Updated element <xmi:id>: EA ExceptionHandler is transformed to an CallBehaviorAction with and input pin.`
>
> `Updated element <xmi:id>: EA InterruptFlow was transformed to an ExceptionHandler.`

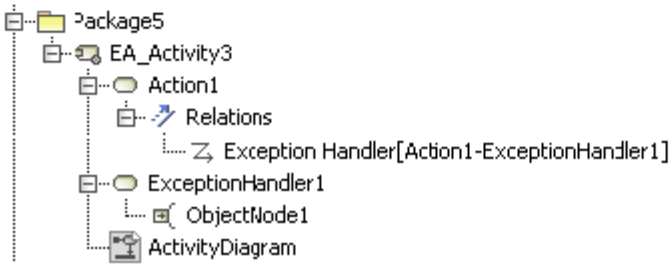
EA (Before Conversion)





MD (After Conversion)





ExceptionHandler.

ObjectFlow

An ObjectFlow line whose ends are not attached to any of the following elements will be transformed into a ControlFlow.

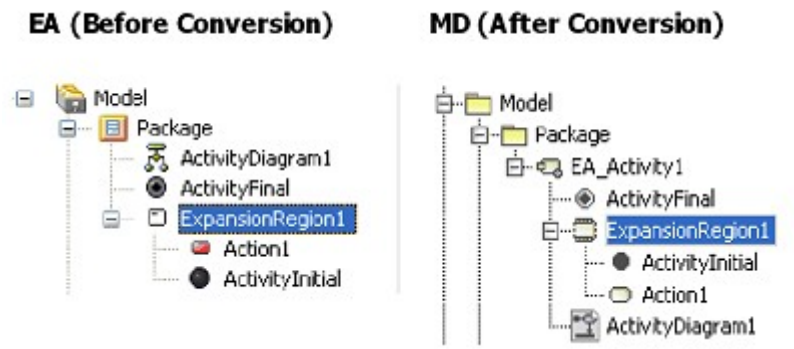
- InputPin
- OutputPin
- ObjectNode
- CentralBufferNode
- DataStoreNode

After completing the transformation, the following transformation message will open:

```
Updated element <xmi:id>: uml:ObjectFlow updated to uml:ControlFlow.
```

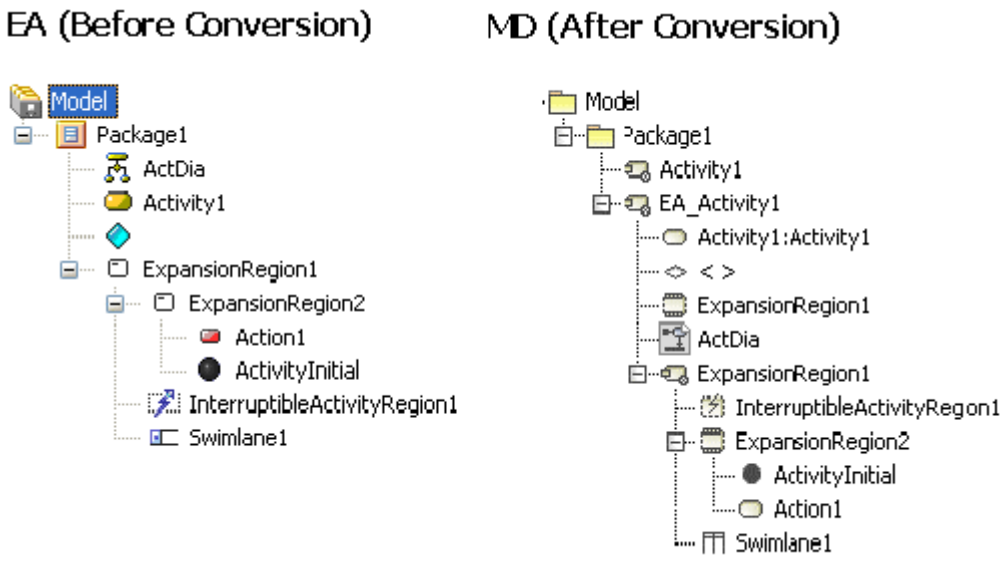
ExpansionRegion

Most of the elements placed inside any ExpansionRegion elements in EA will stay in their original place.



ExpansionRegion tree view.

However, if there is any Activity, Swimlane, InterruptibleActivityRegion, StructuredActivityNode, LooNode, SequenceNode, ConditionalNode, or other ExpansionRegion contained within an ExpansionRegion, it will be placed within a dummy Activity element. The created dummy will have the same name and will be placed at the same level as the ExpansionRegion element.



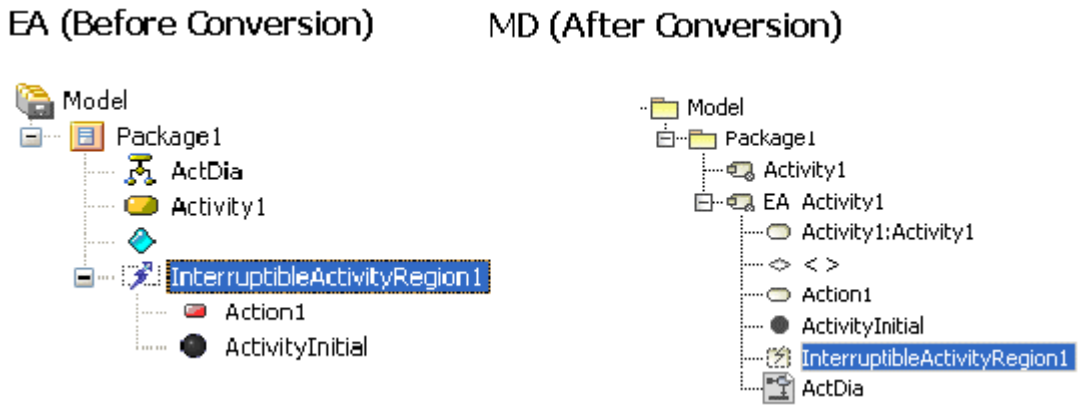
Nested ExpansionRegion tree view.

After completing the transformation, the following transformation message will open:

> Updated element <xmi:id>: ExpansionRegion cannot contain some inner elements.
>
> An Activity with the same name as the ExpansionRegion was created to contain inner elements.

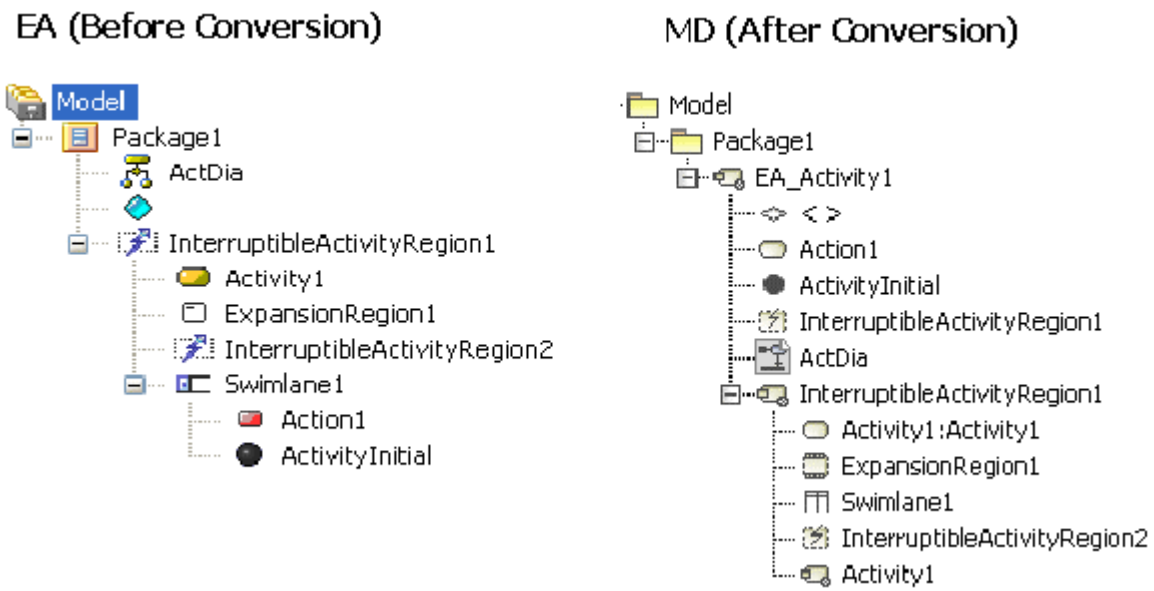
InterruptibleActivityRegion

Most of the elements placed inside an InterruptibleActivityRegion element in EA will be placed at the same level as the InterruptibleActivityRegion element.



InterruptibleActivityRegion tree view.

However, if there is any Activity, Swimlane, ExpansionRegion, StructuredActivityNode, LoopNode, SequenceNode, ConditionalNode, or other InterruptibleActivityRegion contained within an InterruptibleActivityRegion, it will be placed within a dummy Activity. The created dummy will have the same name and will be placed at the same level as the InterruptibleActivityRegion.



Nested InterruptibleActivityRegion tree view.

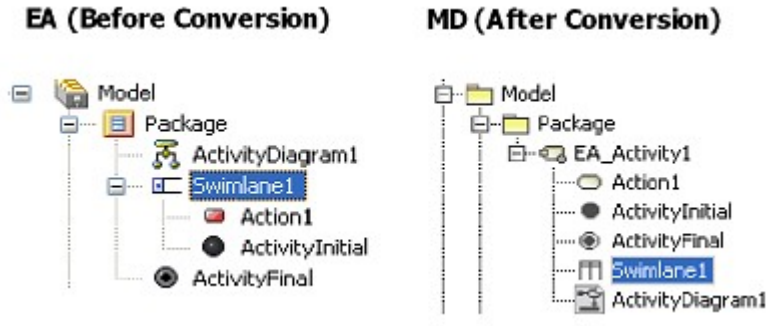
After completing the process, the following transformation message will open:

```
Updated element <xmi:id>: InterruptibleActivityRegion cannot contain
some inner elements.

An Activity with the same name as the InterruptibleActivityRegion was
created to contain inner elements.
```

Swimlane

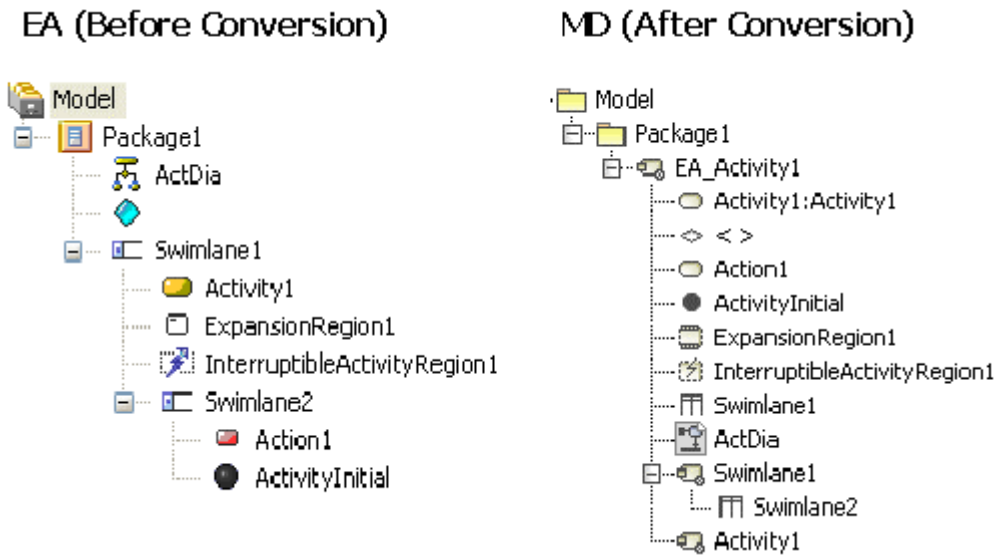
Most of the elements placed inside any Swimlane element in EA will be placed at the same level as the Swimlane element.



Swimlane tree view.

A dummy Activity will also be created to hold any other Swimlanes that it may contain. The dummy activity will have the same name and will be placed at the same level as the Swimlane.

If two or more Swimlanes are nested together, then every element (except Swimlane element) contained within either of them will be placed at the same level as the Swimlane topping the nested-Swimlane-elements hierarchy.



Nested Swimlane tree view.

After completing the process, the following transformation message will open:

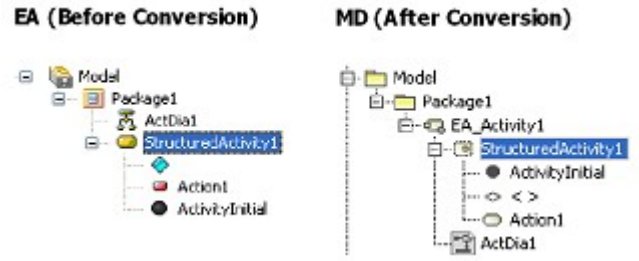
```
Updated element <xmi:id>: Swimlane cannot contain some inner elements.
The XMI structure was fixed.
```

StructuredActivity

Four elements are classified as Structured Activity elements in EA:

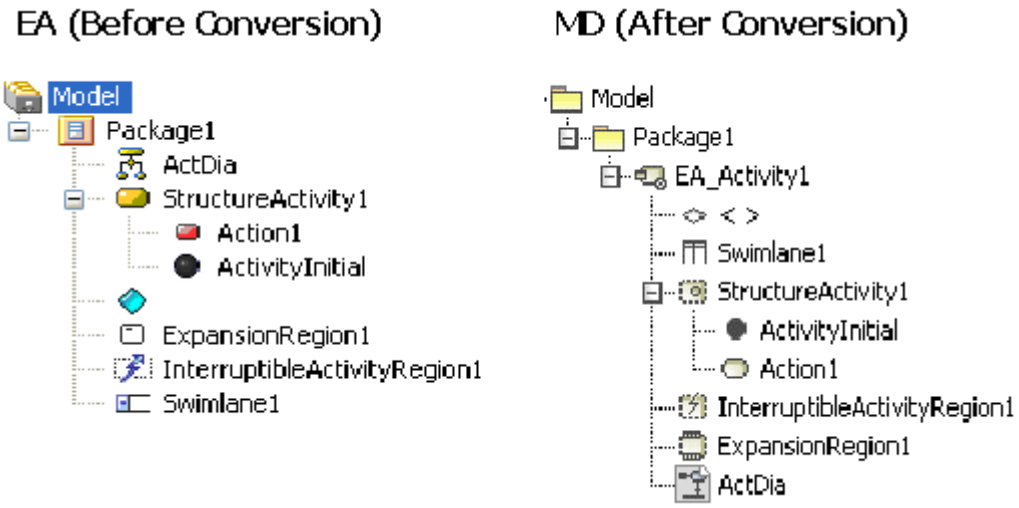
- StructuredActivityNode element
- LoopNode element
- SequenceNode element
- ConditionalNode element

Most of the elements placed inside any StructuredActivityNode, LoopNode, SequenceNode, or ConditionalNode elements in EA will stay in their original place.



StructuredActivityNode tree view.

However, if there is any Activity, Swimlane, InterruptibleActivityRegion, ExpansionRegion, or another StructuredActivity contained within a StructuredActivity, it will be placed within a dummy Activity. The created dummy will have the same name, and will be placed at the same level as the StructuredActivity.



Nested StructuredActivityNode tree view.

After completing the process, the following transformation messages will open, depending on the Structured Activity elements involved:

- Updated element <xmi:id>: StructuredActivityNode cannot contain some inner elements. An Activity with the same name as the StructuredActivityNode was created to contain inner elements.
- Updated element <xmi:id>: ConditionalNode cannot contain some inner elements. An Activity with the same name as the ConditionalNode was created to contain inner elements.
- Updated element <xmi:id>: LoopNode cannot contain some inner elements. An Activity with the same name as the LoopNode was created to contain inner elements.
- Updated element <xmi:id>: SequenceNode cannot contain some inner elements. An Activity with the same name as the SequenceNode was created to contain inner elements.

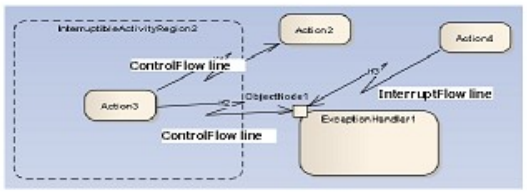
InterruptFlow

In some cases, EA InterruptFlows are ControlFlow lines. Their image will be displayed as the InterruptFlow line in the Activity diagram. An InterruptFlow is not a ControlFlow line if the InterruptFlow line is drawn from one element in an InterruptibleActivityRegion to another outside the InterruptibleActivityRegion. In an XMI file, this line will be imported as a ControlFlow line, and its image will be changed to that of the ControlFlow line.

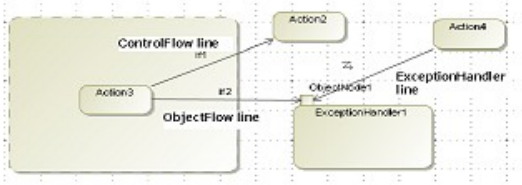
However, if either end of the line is any of the following elements, it will be transformed into an ObjectFlow line.

- InputPin element
- OutputPin element
- ObjectNode element
- CentralBufferNode element
- DataStoreNode element

EA (Before Conversion)



MD (After Conversion)

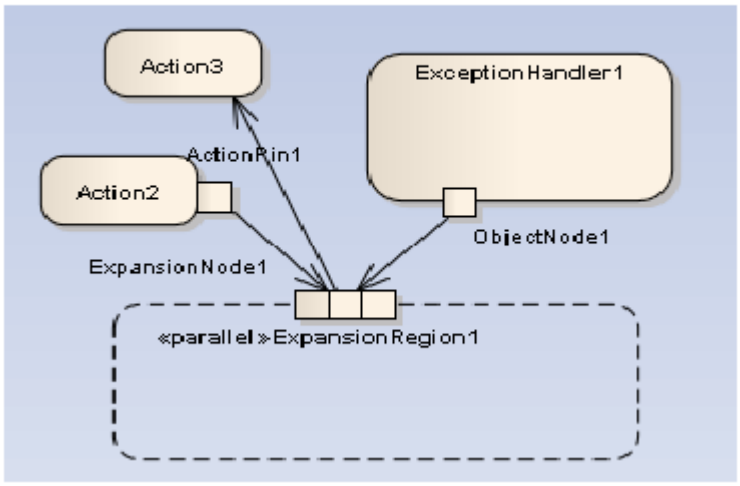


InterruptFlow.

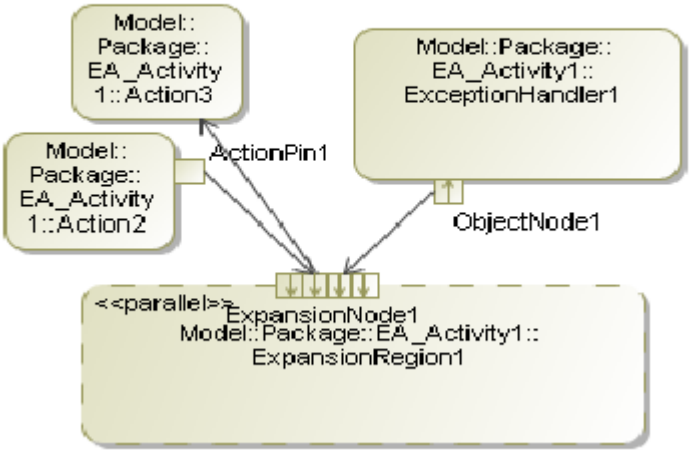
ExpansionNode

An ExpansionNode is a Pin which can only be contained within an ExpansionRegion and will be imported like any other Pin elements. However, if an ExpansionNode in EA is created inside another element rather than an ExpansionRegion, that particular ExpansionNode will not be imported.

EA (Before Conversion)



MD (After Conversion)



ExpansionNode.

On this page

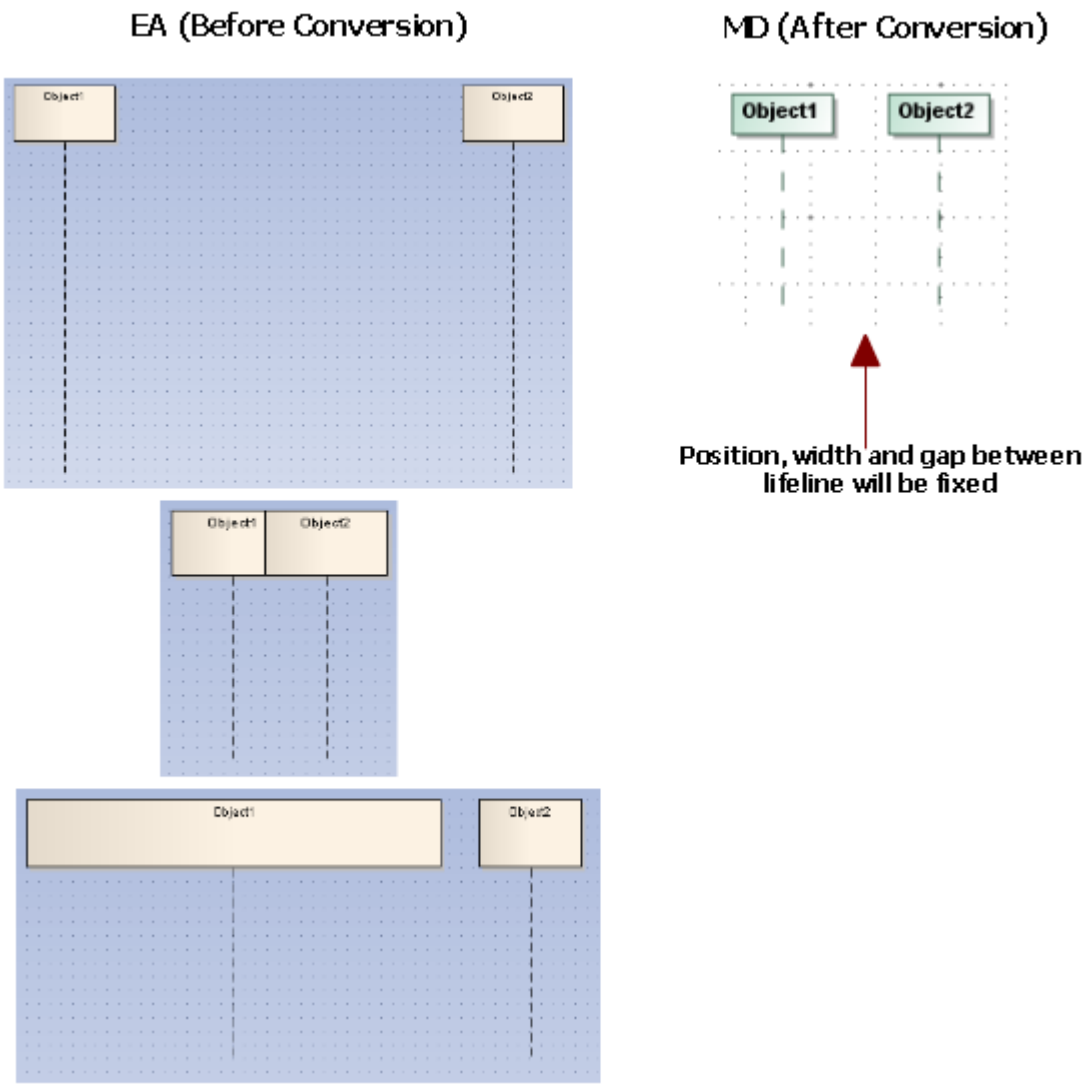
1.5.3 Sequence diagram elements

Lifelines

All of the EA Lifelines will be imported, but a part or port within a Lifeline will be transformed into a new separate Lifeline.

Gaps between Lifelines

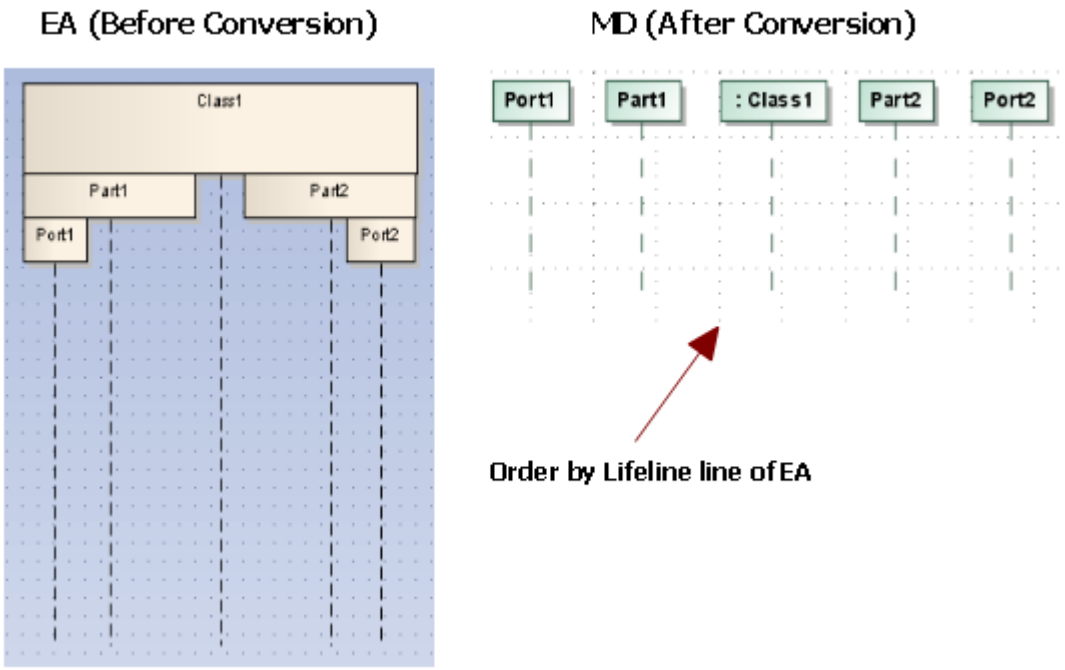
The position and width of any Lifeline created in EA will not be imported. Every Lifeline will be given a fixed value and position in MagicDraw. MagicDraw will place the first Lifeline on the left-hand side of the diagram and the second Lifeline on the right-hand side next to the first one. The length of the gap between the Lifelines will be fixed.



Gaps between Lifelines.

Lifelines Arrangement

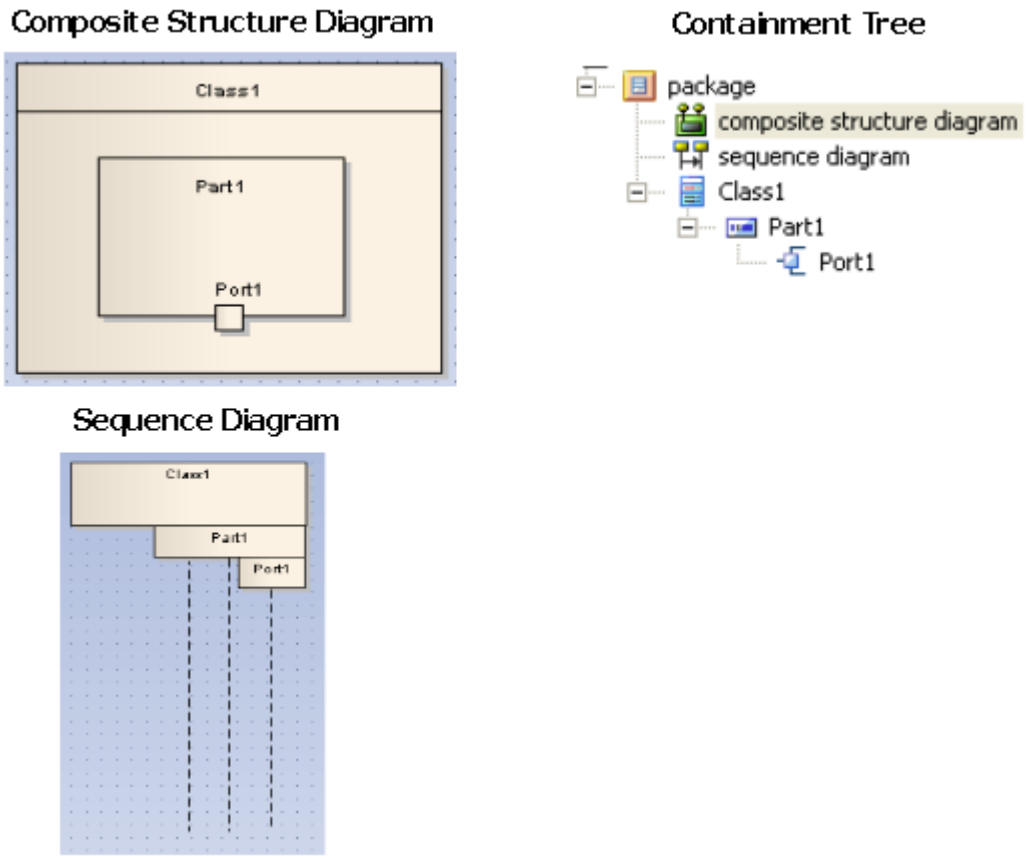
A Lifeline can be nested within another component, such as a Part or Port. If this is the case, every component nested within the Lifeline and the Lifeline itself will be drawn separately in MagicDraw. They will be arranged in order depending on the position of their Lifeline lines.



Lifelines arrangement.

Class, Part, and Port

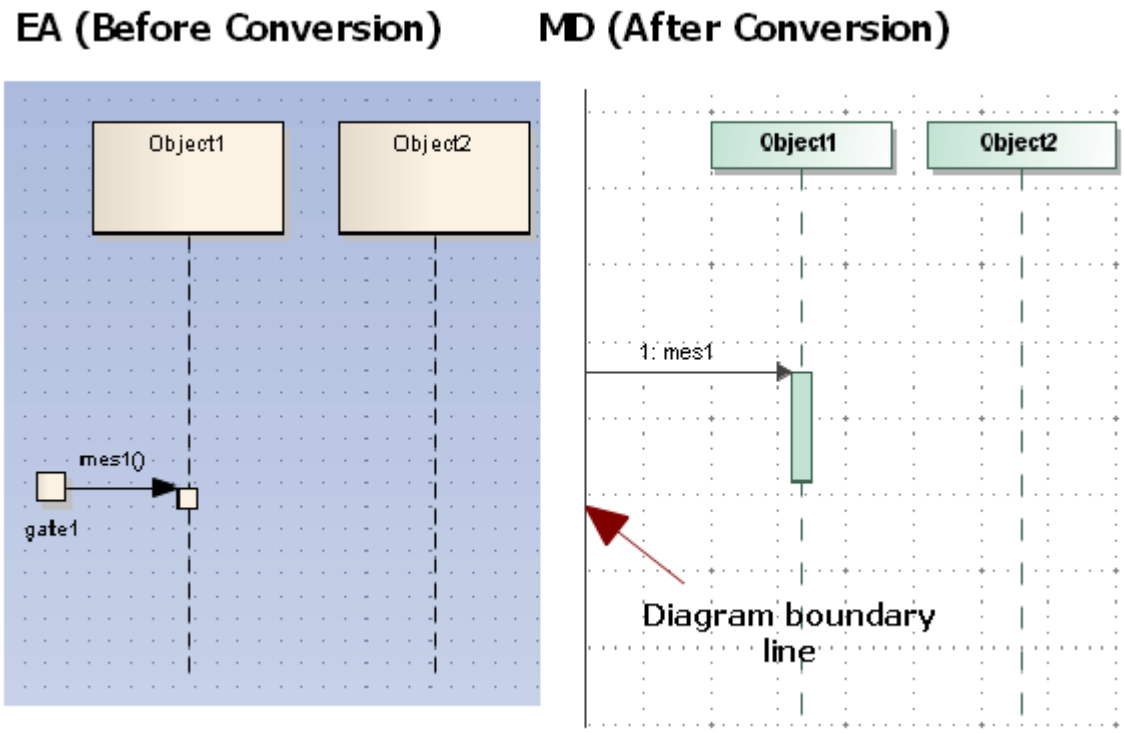
When represented as Lifelines, Classes, Parts, and Ports have different characteristics from the others. They will be bundled according to their relationships. A Composite Structure diagram provides one convenient way to create a Class, Part, or Port.



Class, Part, and Port.

Gate

A Sequence Message whose tail is connected to a Gate and head connected to a Lifeline in EA will be transformed into a Sequence Message with its tail connected to one of the boundary lines of the diagram in which it is contained. The Gate itself will be removed.

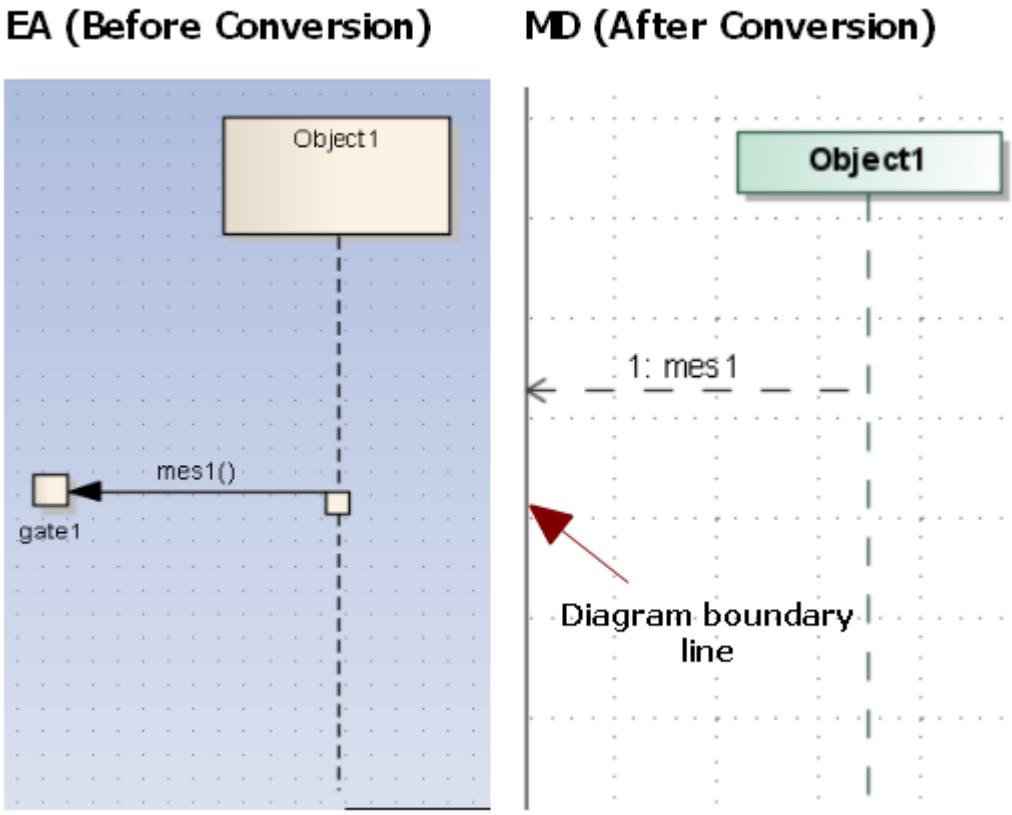


Gate.

Once the Gate has been removed and the transformation process has been completed, the following transformation message will open:

> `Removed element <xmi:id>: uml:Gate.`

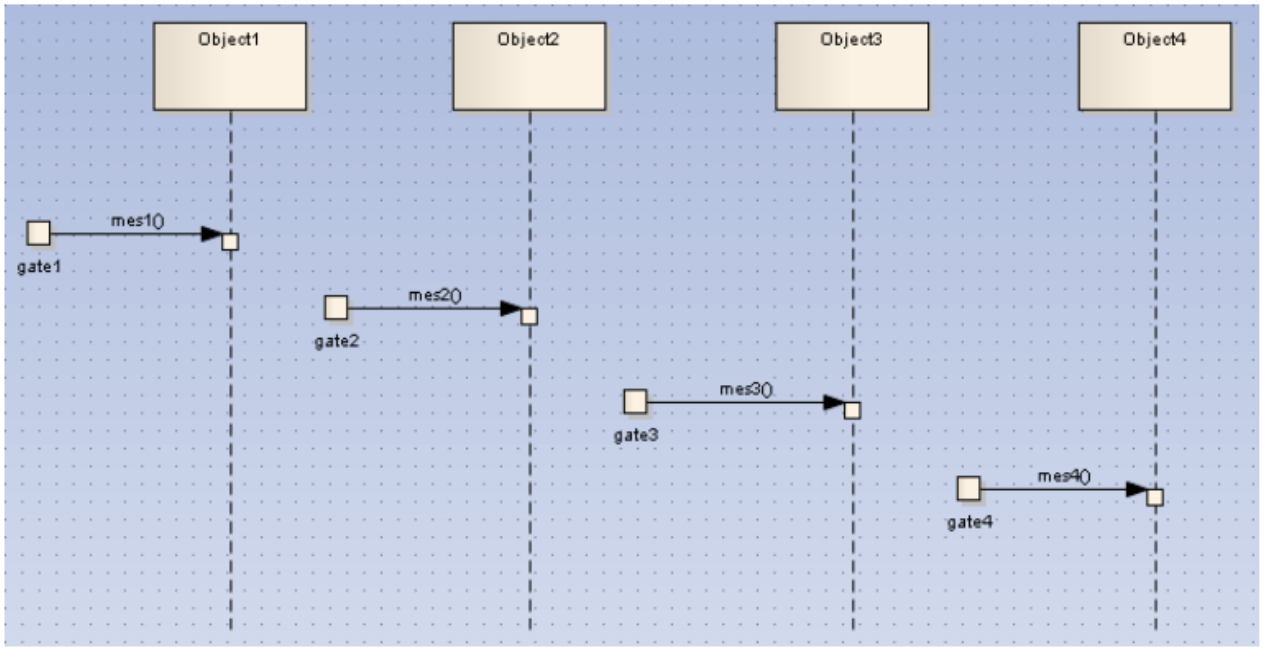
A Sequence Message whose head is connected to a Gate and tail connected to a Lifeline line in EA will be transformed into a Reply Message with its tail connected to one of the boundary lines of the diagram in which it is contained. The Gate itself will be removed



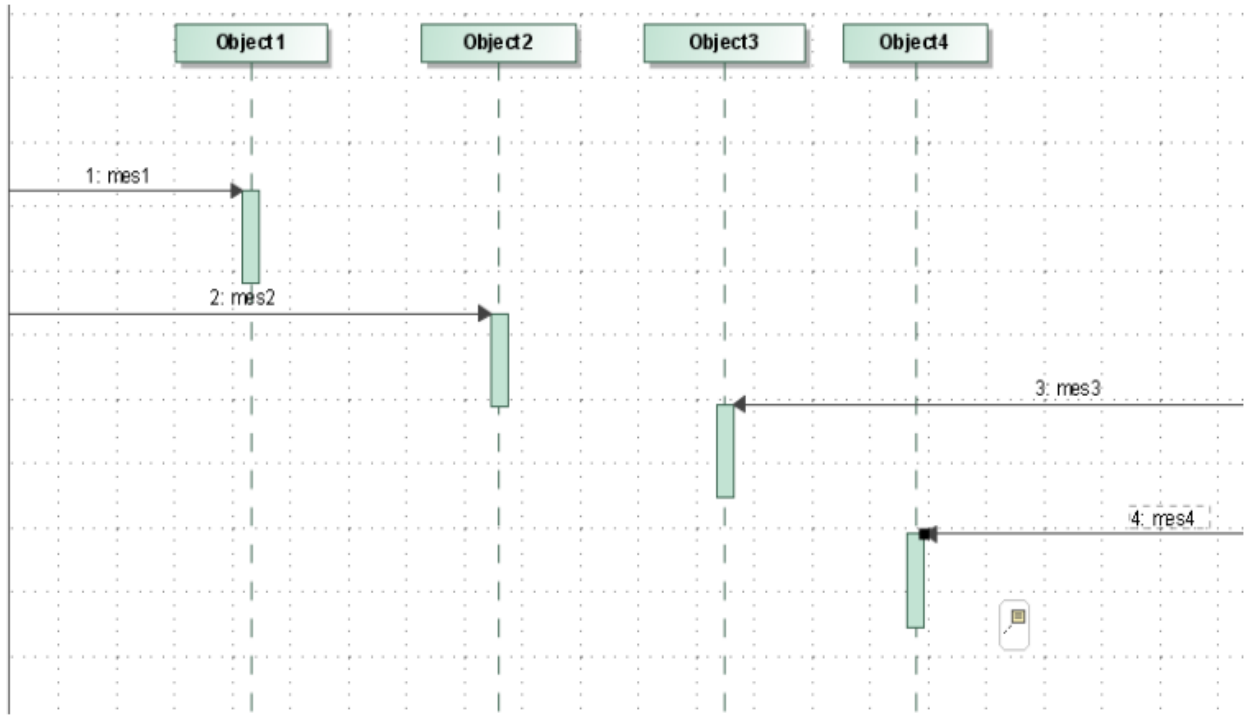
Gate (EA) and Reply Message (MagicDraw).

The Diagonal Message and Reply Message will be connected to the nearest diagram boundary.

EA (Before Conversion)



MD (After Conversion)



Gate (EA) and Direction of Diagonal Message (MagicDraw).

Anything else connected to a Gate will also be removed, except the tail of a Sequence Message. For example, if the head of a Sequence Message is connected to a Gate, the Message will be removed. If the

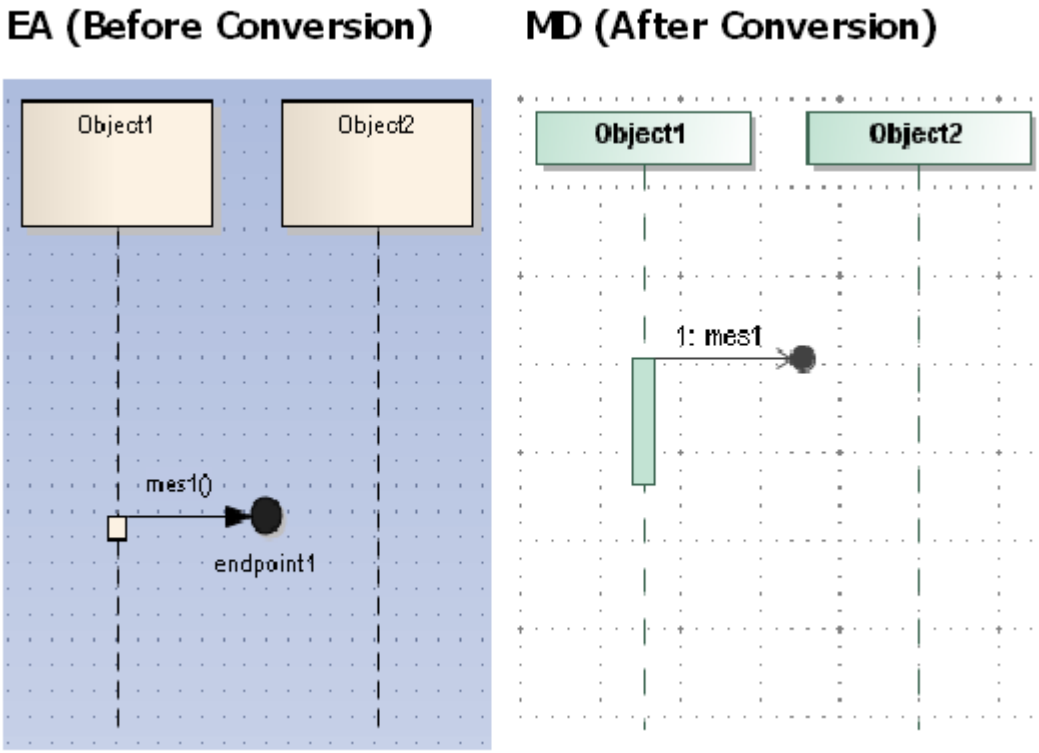
tail of a Message is connected to a Gate, but the head is connected to anything other than a Lifeline line, the Message will be removed.

Endpoints

A Sequence Message whose head connected to an Endpoint and tail connected to a Lifeline line in EA will be transformed into a Lost Message. The Endpoint element itself will be removed. Once the process has been completed, the following transformation message will open:

```
Removed element <xmi:id>: EndPoint.
```

A Message whose head connected to a Lifeline line and tail connected to an Endpoint will be transformed into a Found Message.



Endpoint.

Anything else connected to an endpoint will also be removed, except the head of a Sequence Message. For example, if the tail of a Sequence Message is connected to an endpoint, the Message will be removed.

If the head of a Message is connected to an endpoint, but the tail is connected to anything other than a Lifeline line, the Message will be removed. Once the process has been completed, the following transformation message will open:

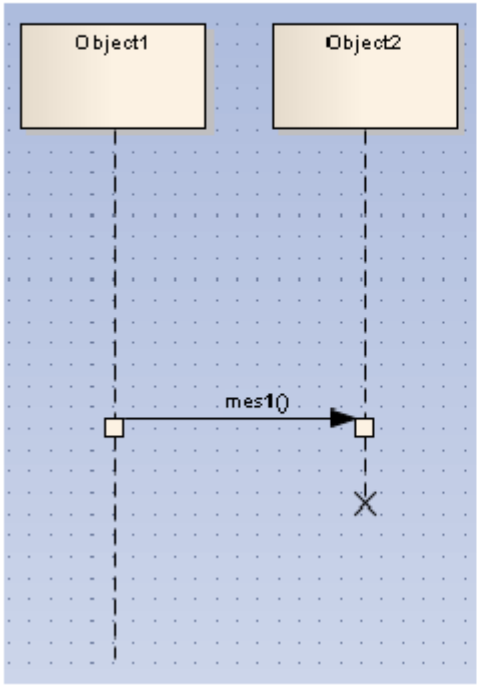
```
Removed element <xmi:id>: Invalid Message. Source and Target of the Mes-
sage are not connected to any Lifeline.
```

Delete Messages

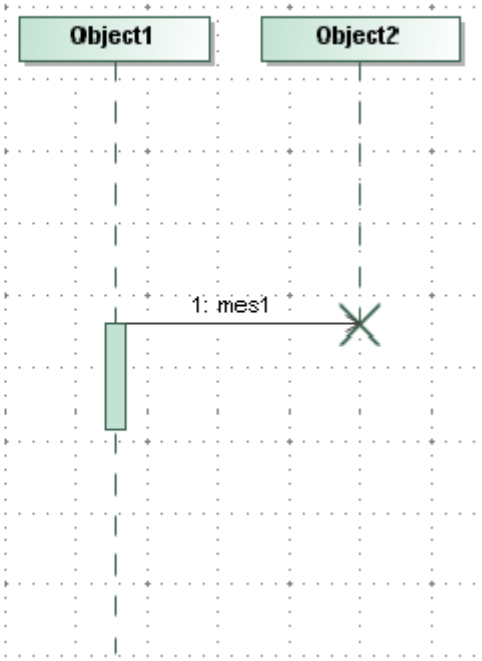
A Sequence Message whose property 'Lifecycle' is set to 'Delete' (that causes the Lifeline targeted by the Message to end at some range after the point of contact), will be transformed into a Delete Message. The Lifeline connected to its head will end at the point of contact, and all Messages will be removed after that point of contact of the Lifeline. MagicDraw will report the following transformation message once each process has been completed:

```
Removed element <xmi:id>: Message is under Delete Message.
```

EA (Before Conversion)

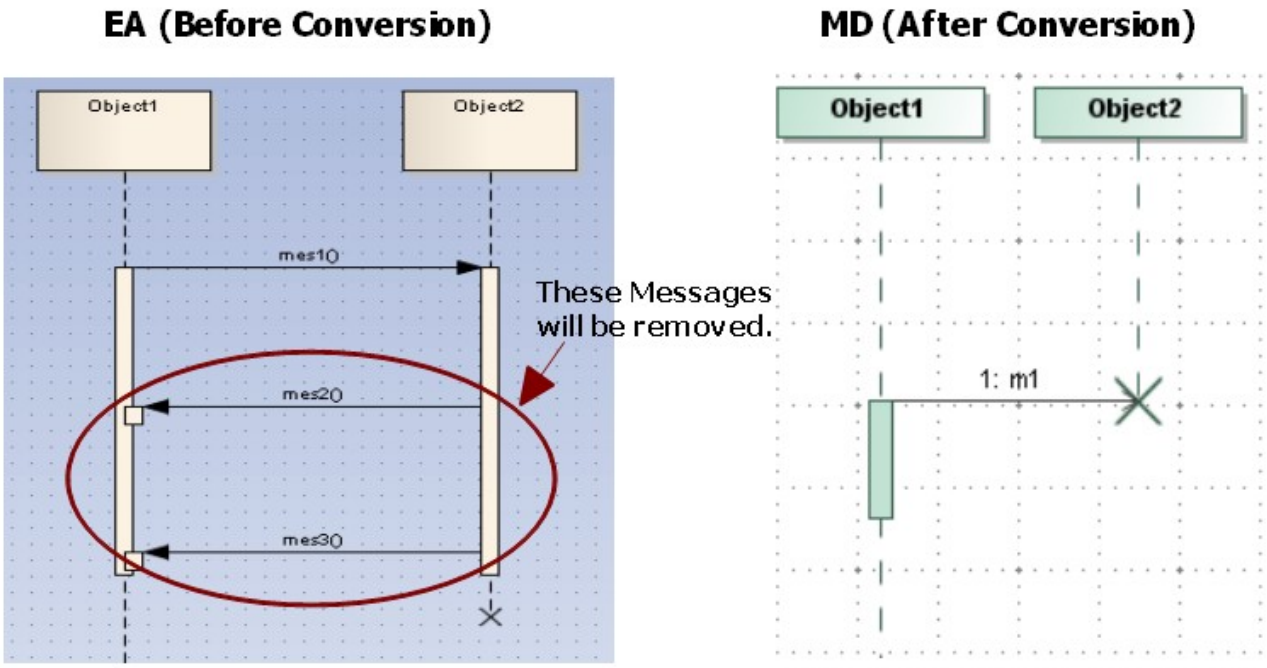


MD (After Conversion)



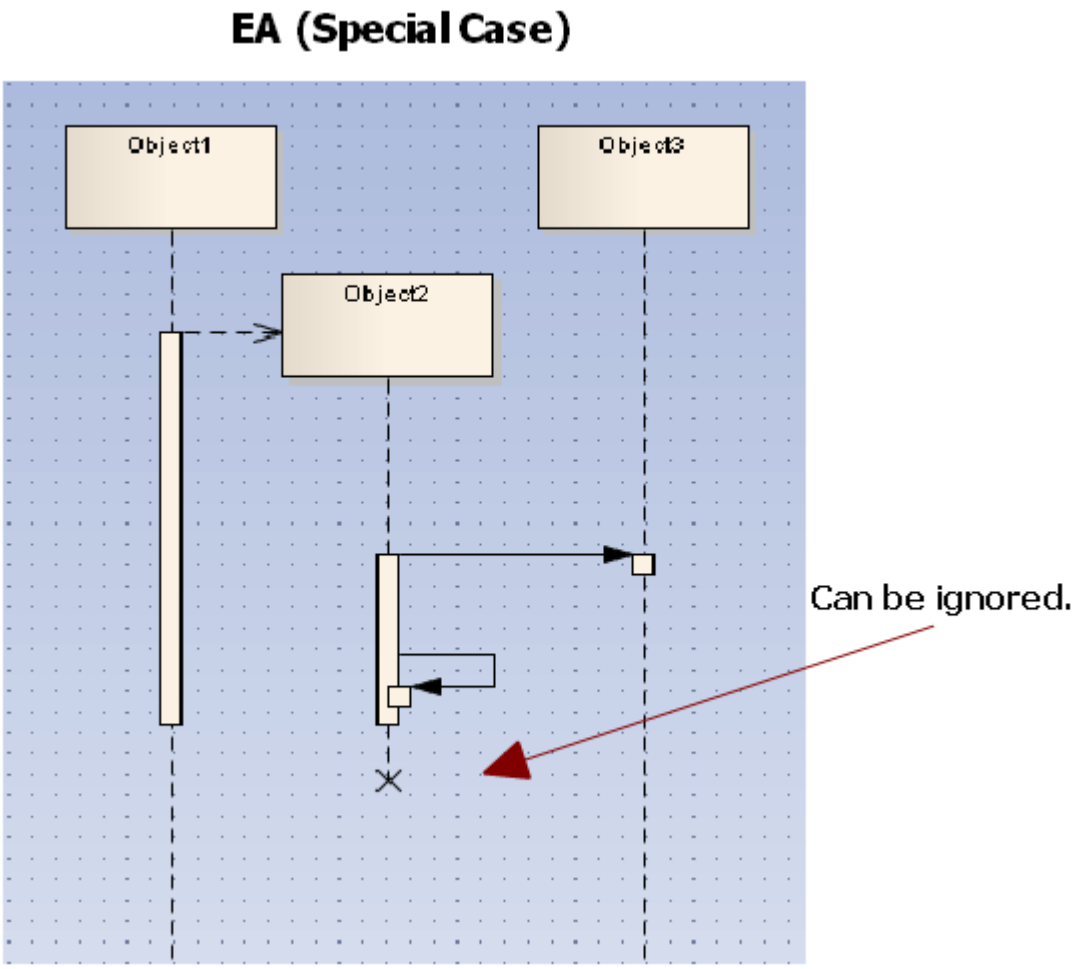
Delete message.

Under certain circumstances, a Lifeline connected to the head of a Delete Message does not end at the point of contact. Instead, an Activation will be created and started from the point of contact. The Lifeline will then end at the end of the Activation. However, that Activation should not interact with any Message.



Special case of delete message.

Under certain circumstances, an 'X' sign (normally drawn after a Delete Message) in EA will be drawn on a Lifeline line whose Lifeline is being pointed by a Create Message. If this is the case, the **X** sign has no significant meaning and can be ignored.

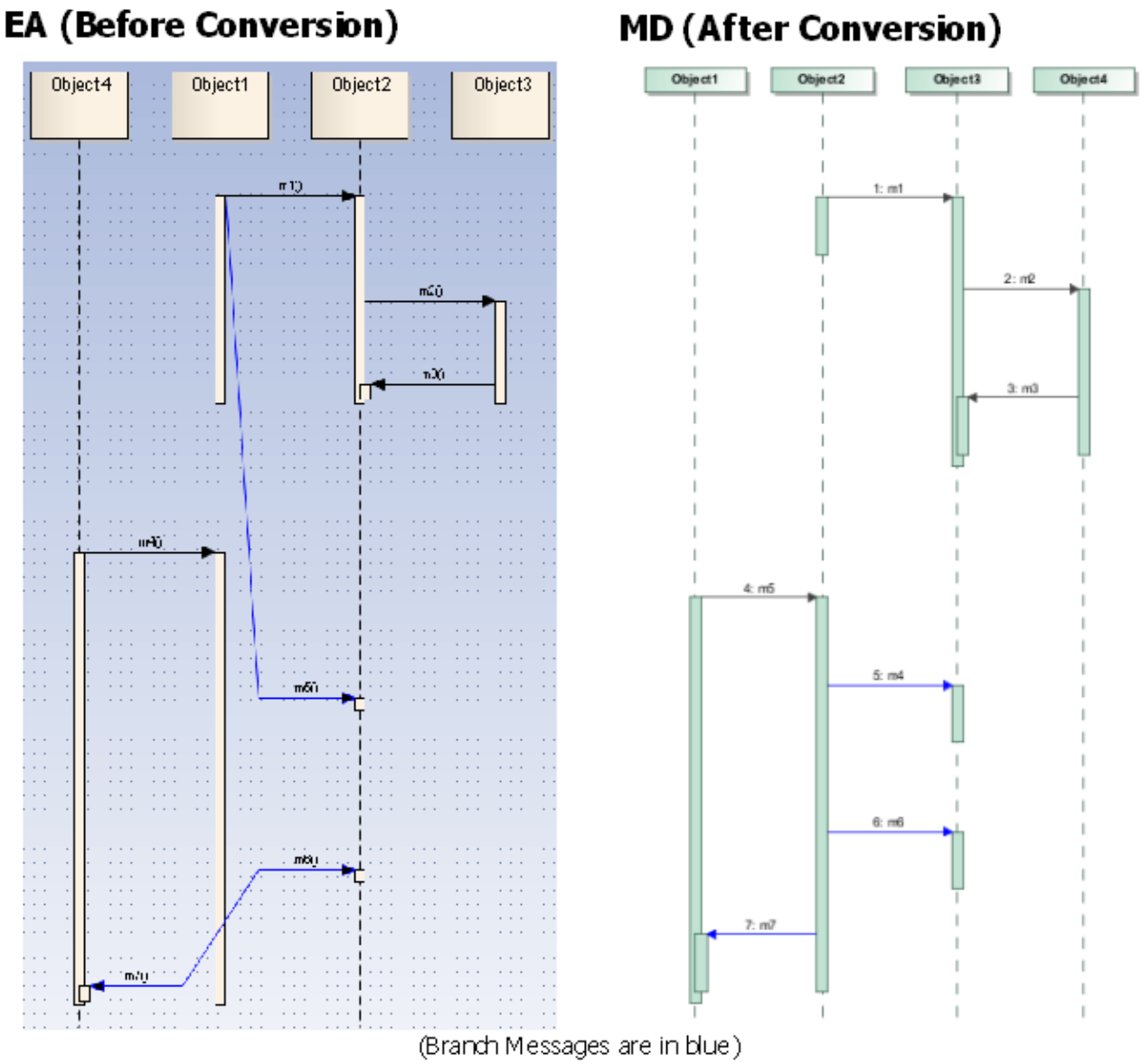


The X sign.

Branch Messages

If the **Branch with previous Message** option of a Sequence Message is enabled, the tail of the message connect to the tail of its previous Message on the same Lifeline line. The message will be called 'Branch Message' from that time on.

Every Branch Message in EA will be transformed into a normal Message in MagicDraw.



Branch Message.

After the transformation process has been completed, the following transformation message will open:

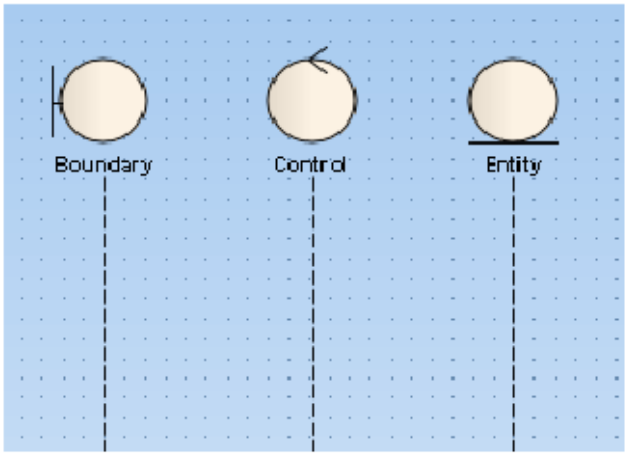
```
Updated element <xmi:id>: Branch Message updated to regular Message.
```

Boundaries, Controls, and Entities

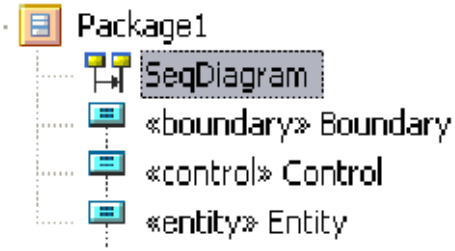
The Boundary, Control, and Entity elements in a Sequence diagram context in EA are Lifeline elements whose stereotype property types are set to *Boundary*, *Control*, or *Entity*. They will be imported like any other normal Lifelines. However, an XMI exported from EA has an <<entity>> stereotype problem. This means the Lifeline with the <<*entity*>> *stereotype* applied is unable to show the stereotype icon. You can solve this problem by placing an <<*entity*>> *stereotype* in *EA_Profile.xml.* The converted model can then use it instead of the one from EA.

EA (Before Conversion)

Display picture

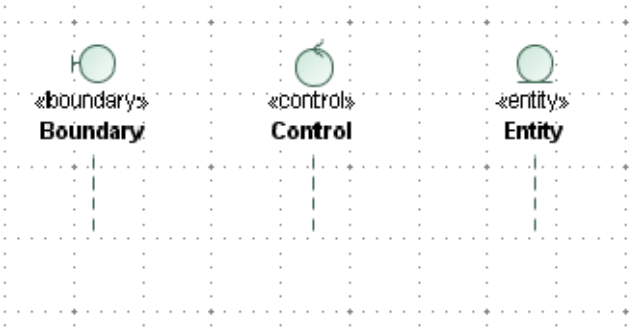


Containment Tree

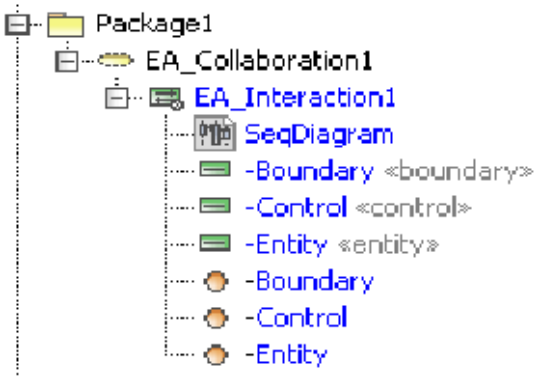


MD (After Conversion)

Display picture



Containment Tree



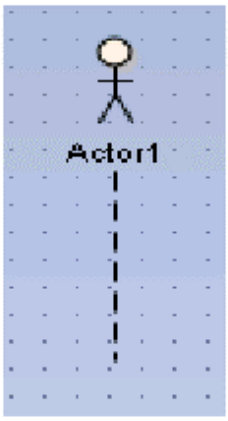
Boundary, Control, and Entity.

Actors

An Actor element in the Sequence diagram context in EA is not a regular Lifeline element. It is a special Lifeline element whose property type is *uml:Actor*. It will be transformed into a regular Actor and a Lifeline element will be created to represent it. The figure below shows an Actor element in the MagicDraw Containment tree.

EA (Before Conversion)

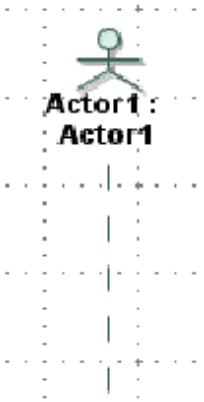
Display picture



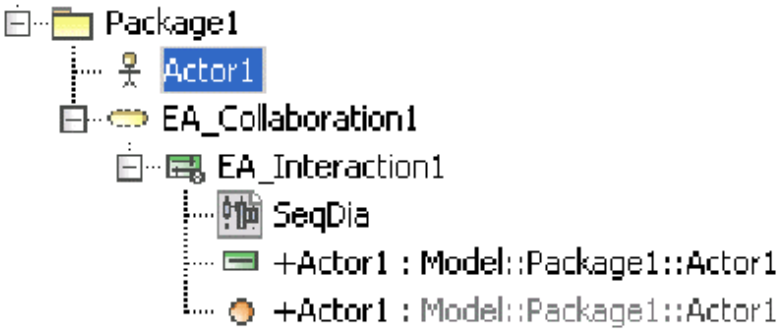
Containment Tree

MD (After Conversion)

Display picture



Containment Tree

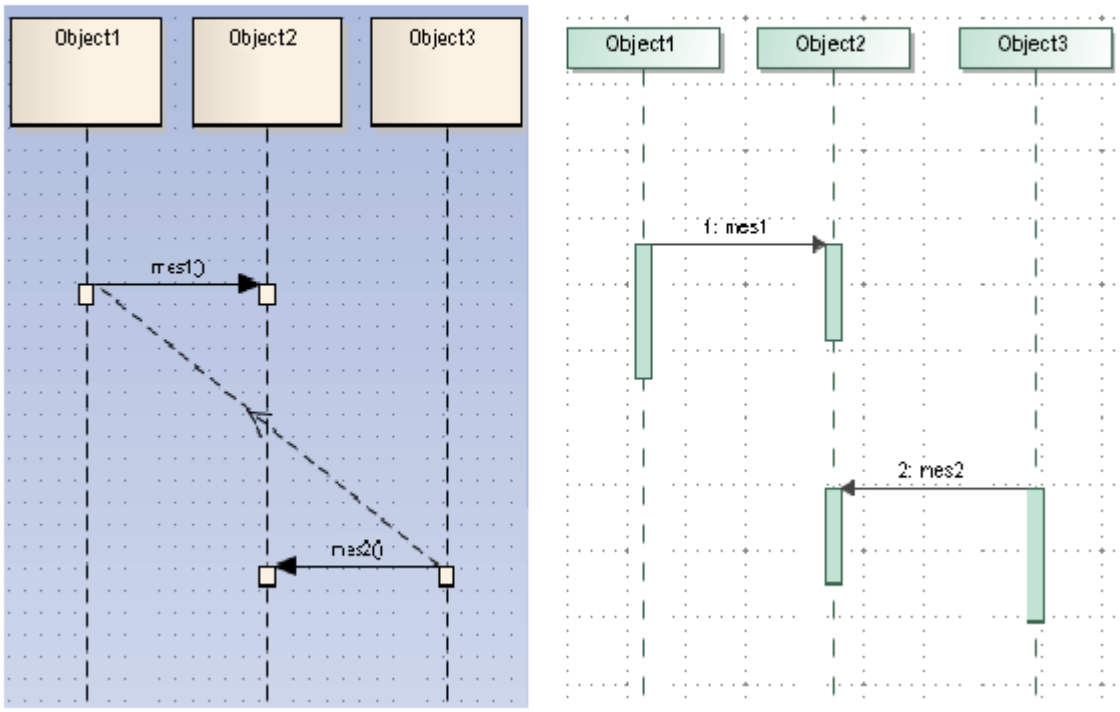


Actor.

General Ordering

Not every General Ordering element in EA will be imported, as MagicDraw does not support it in the current release.

EA (Before Conversion) **MD (After Conversion)**



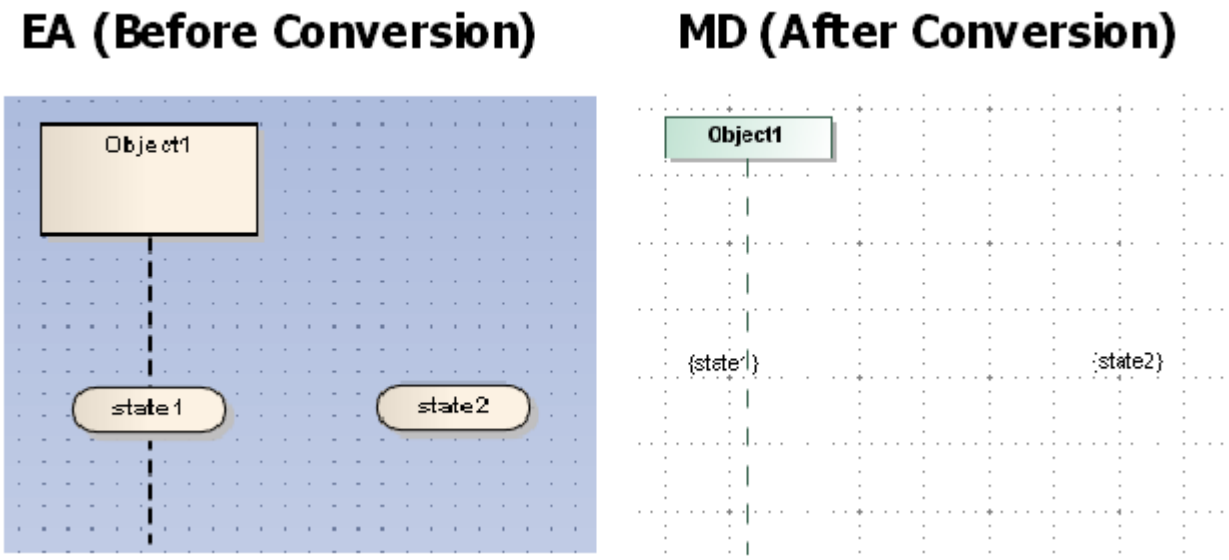
General ordering.

After completing the transformation process, the following transformation message will open:

> `Updated element <xmi:id>: General Ordering will not be imported.`

State Invariant

A State Invariant in EA will be transformed as is. MagicDraw does not support State Invariants in the current release.

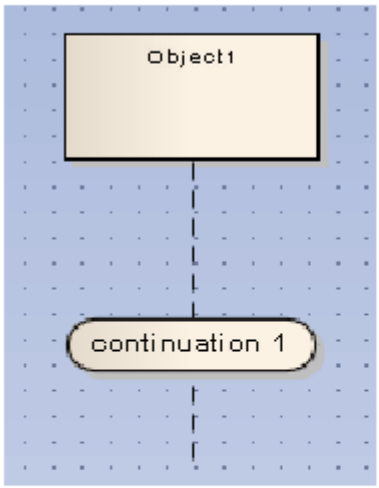


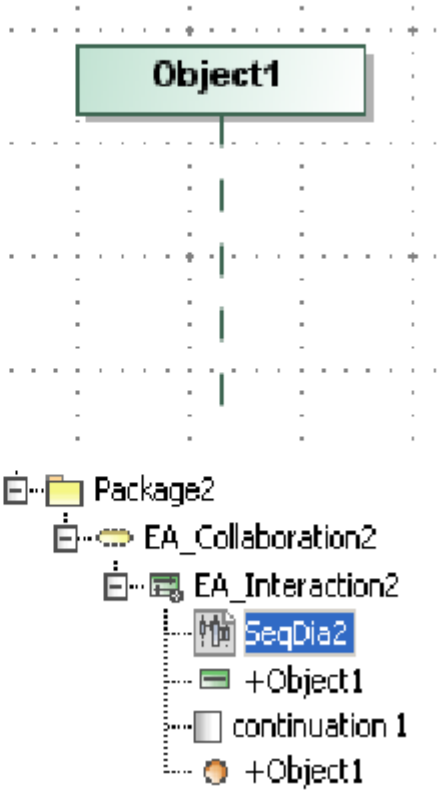
State Invariant.

Continuation

Continuations from EA can be imported to MagicDraw. They are viewable in the containment tree in MagicDraw, but without a picture displayed in the diagram. MagicDraw does not support Continuations in the current release.

EA (Before Conversion) **MD (After Conversion)**





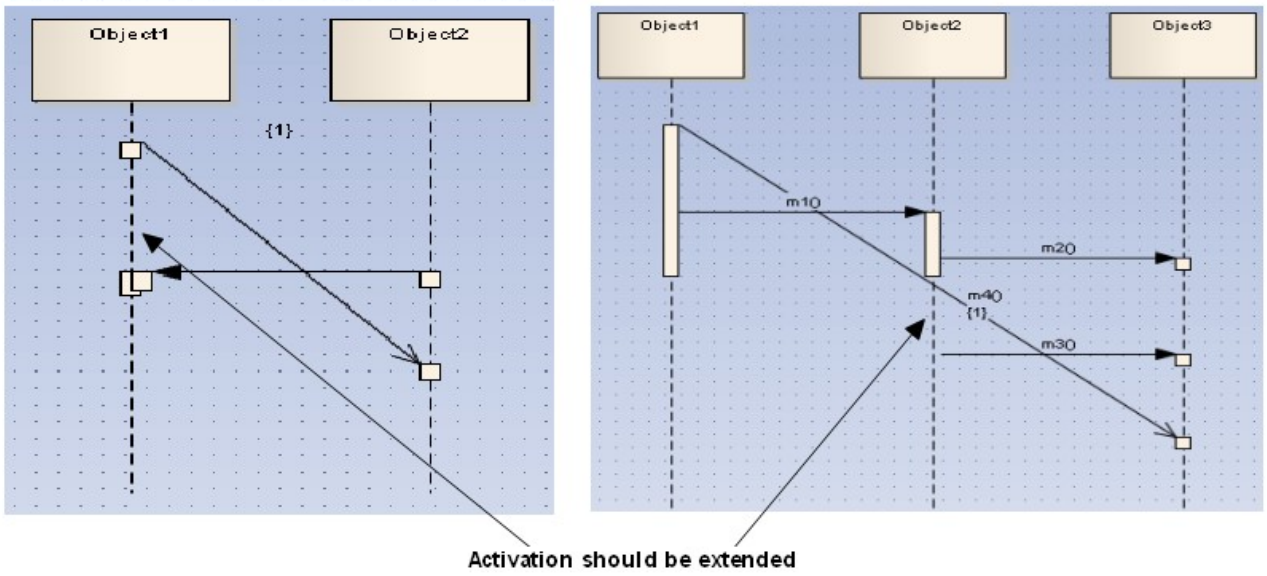
Continuation.

After completing the transformation process, the following transformation message will open:

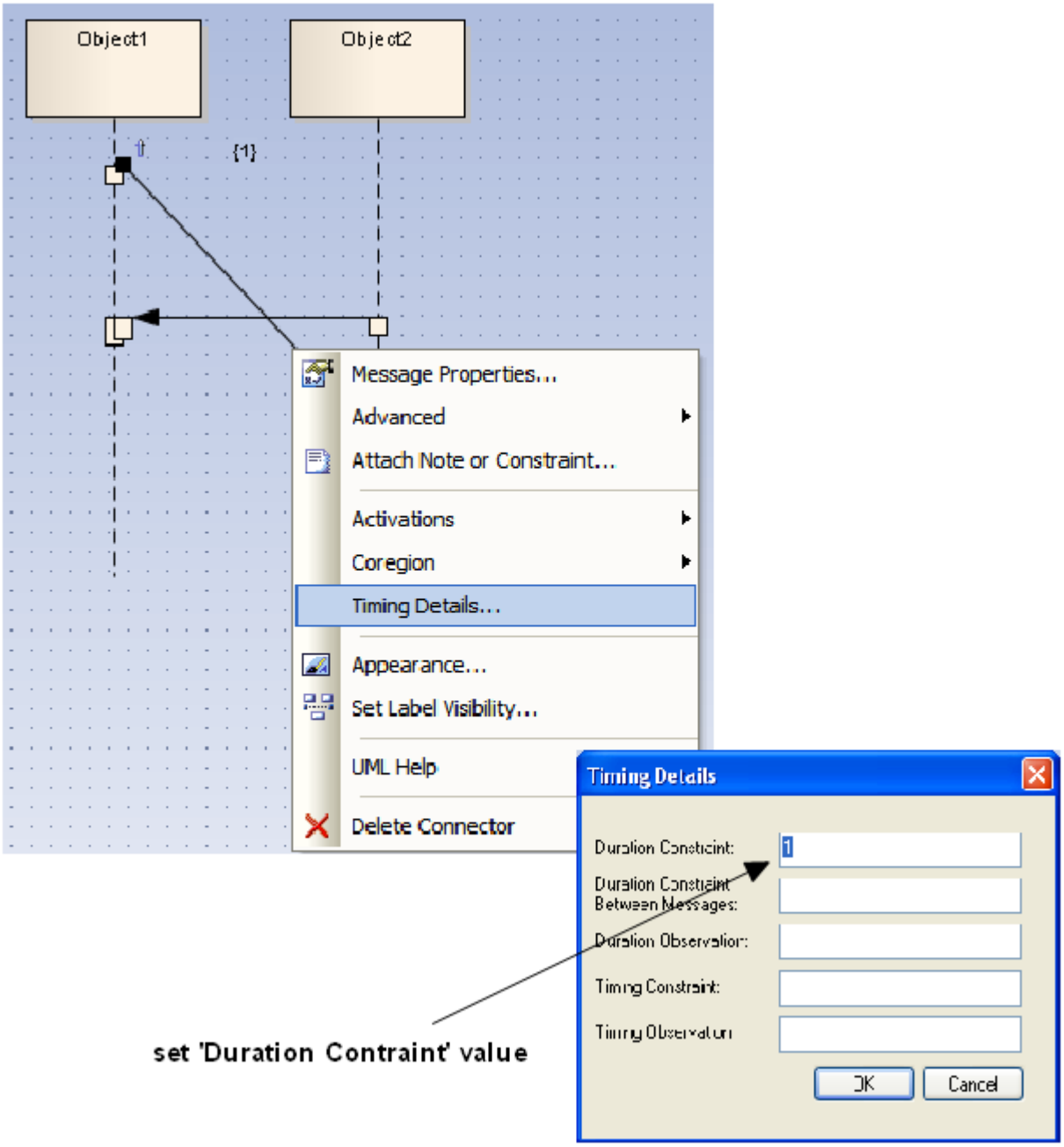
```
Updated element <xmi:id>: Continuation will not be displayed.
```

Diagonal Message

A Diagonal Message is a Message whose destination's height is adjustable. A diagonal Message may change Activations.



To create a Diagonal Message in EA, you must specify the Timing Details property of the Message and add a numeric value to the Duration Constraint input field.



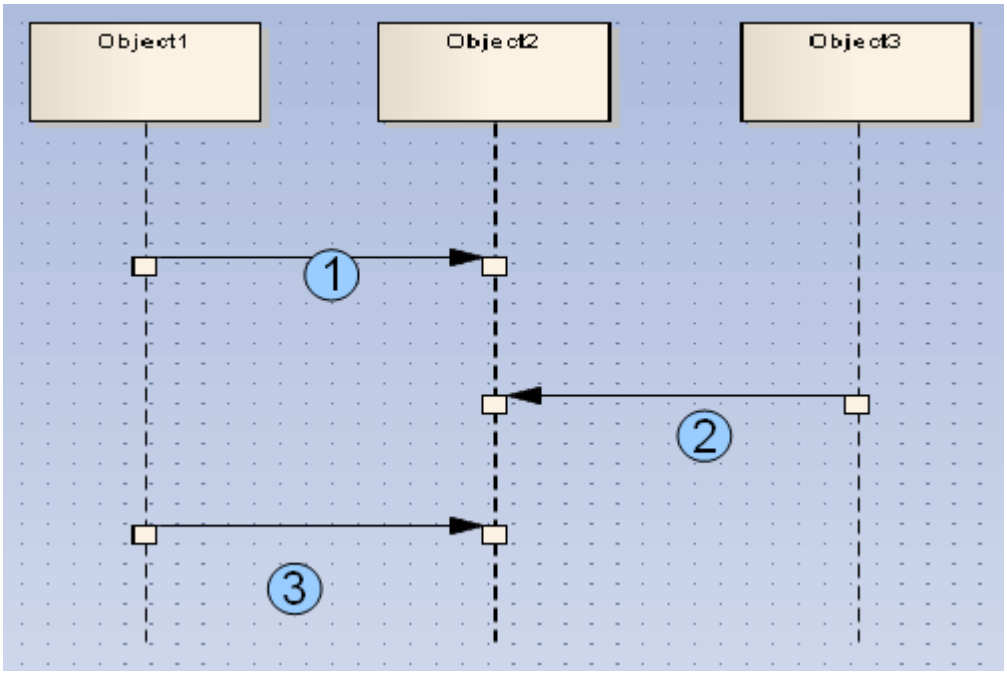
Creating a Diagonal Message.

Synchronous Message behavior

The placement of Synchronous Messages in EA affects the way Activations are created.

Order

A Message in EA has a 'Sequence Number' that indicates the order of the message in the diagram. This information can be found in the exported XMI file. Knowing it helps predict how Activations will be created. The order starts from the top and goes downward, so the first Message in the diagram is the one drawn at the top of the diagram. Its Sequence Number will be set to '1'.

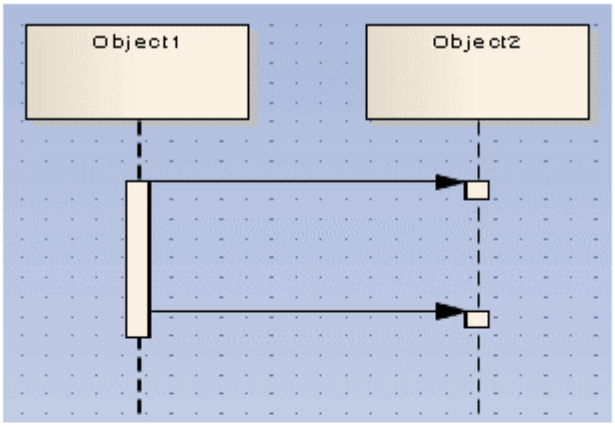


Message sequence number.

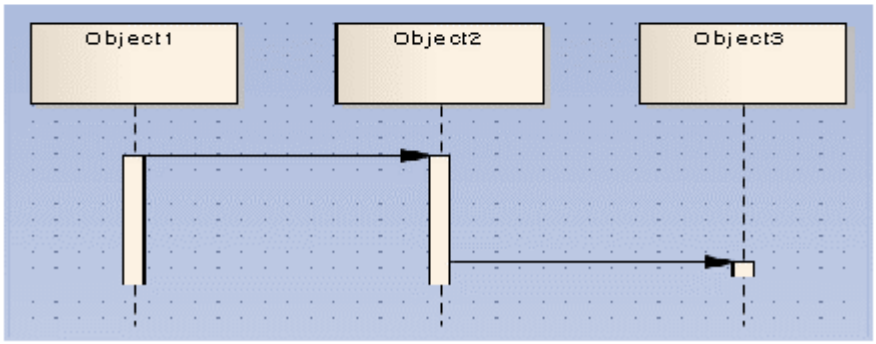
Process

Generally, in EA, any two Synchronous Messages will be in the same process if they meet the following conditions:

- Both of their tails are placed on the same Lifeline.
- The head of the upper Message and the tail of the lower Message are on the same Lifeline.



condition 1

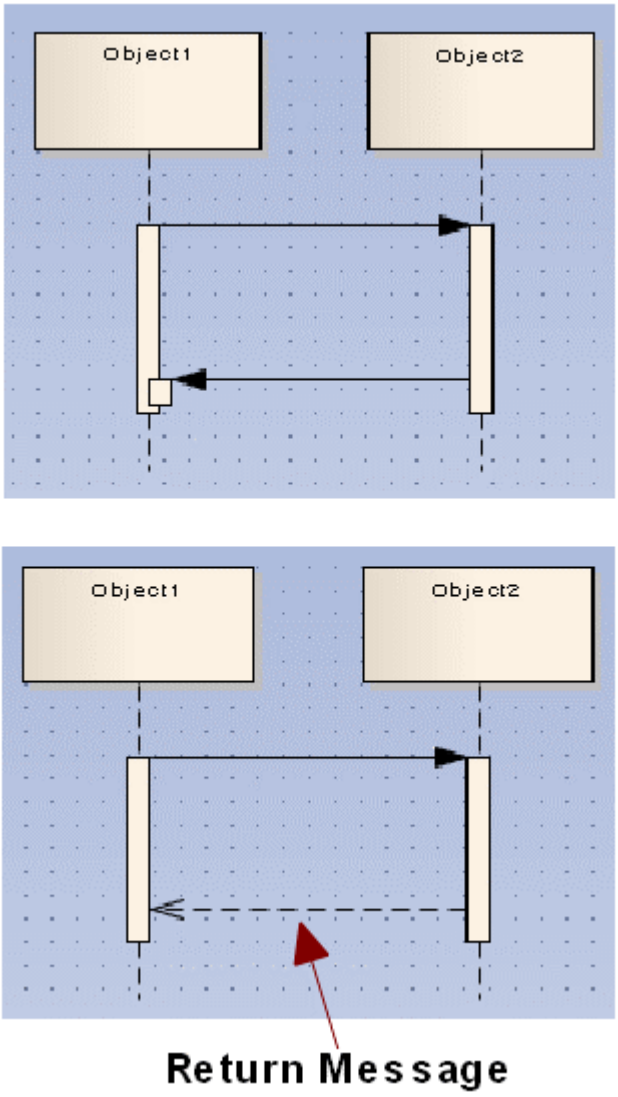


condition 2

Process.

Activation level

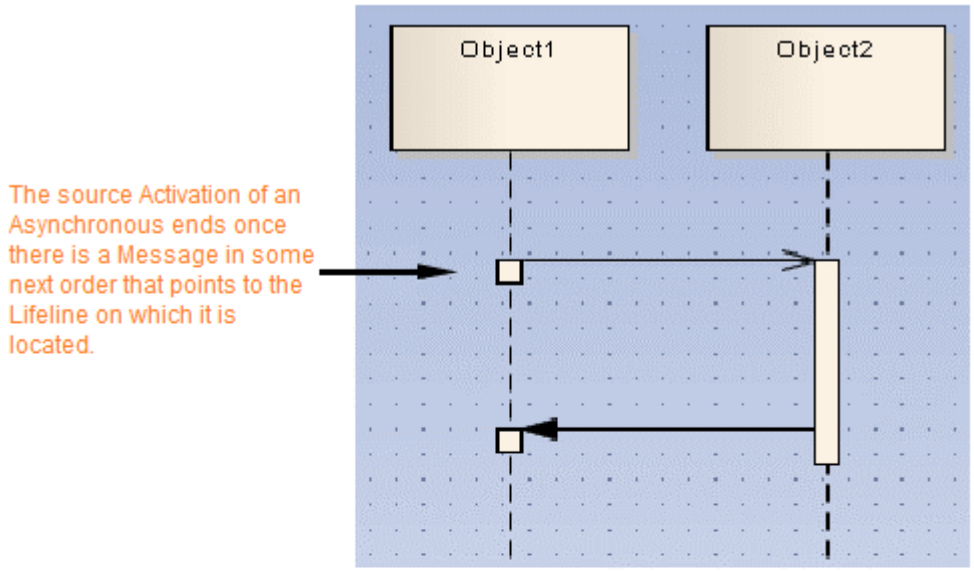
The Activation Level starts from level 0. It will increase in increments of 1 as an ongoing process that receives a Message that is not a Return Message (Reply Message in MagicDraw).

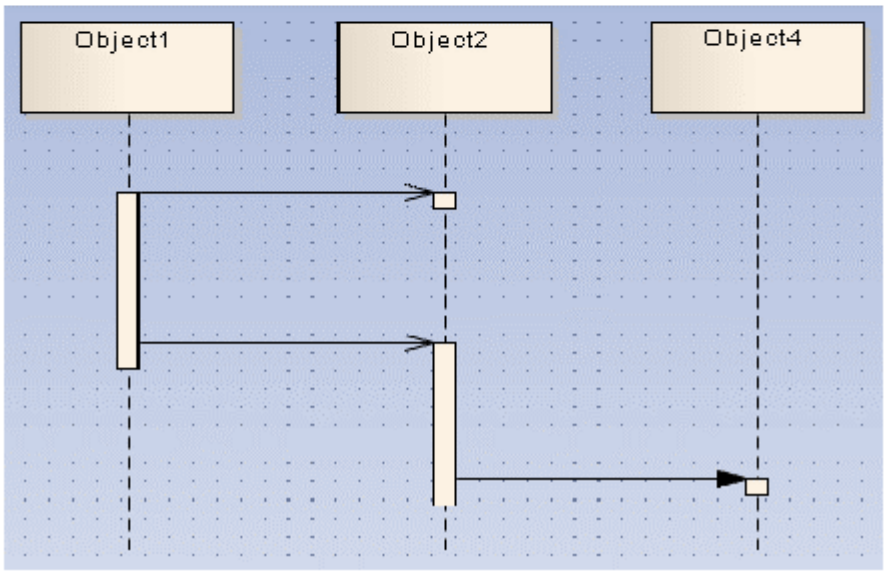


Activation level.

Asynchronous Message behavior

If a Message is an Asynchronous Message in EA, its source Activation will end if there is no other Message in the same group with a higher Sequence Number. Its source will be attached to the same Lifeline.

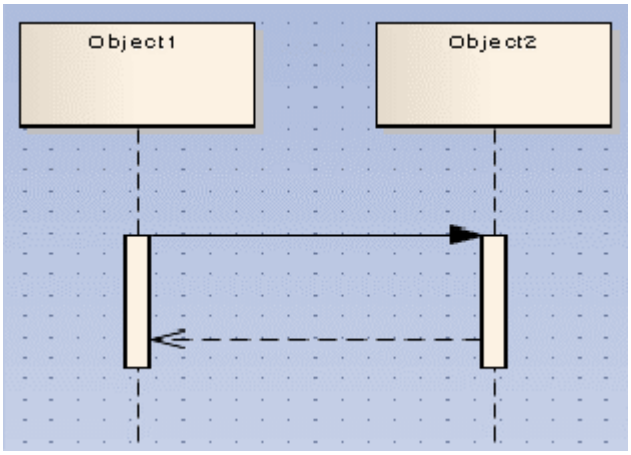




Asynchronous Message.

Return Message behavior

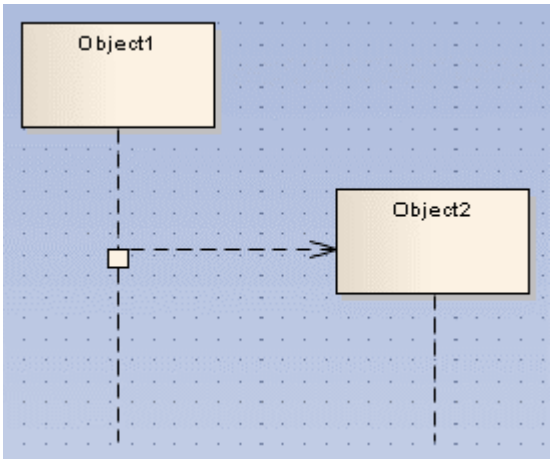
A Return Message in EA is called a Reply Message in MagicDraw. When it is pointed to an Activation, it will not create an Activation at the top of the existing Activation, unlike other normal Synchronous Messages.



Return message.

Activation options

You can control how Activations behave at some level through the Message options. You can access Message options by right-clicking any Message, and then selecting the **Activations** option. However, MagicDraw does not support Activation options in the current release.



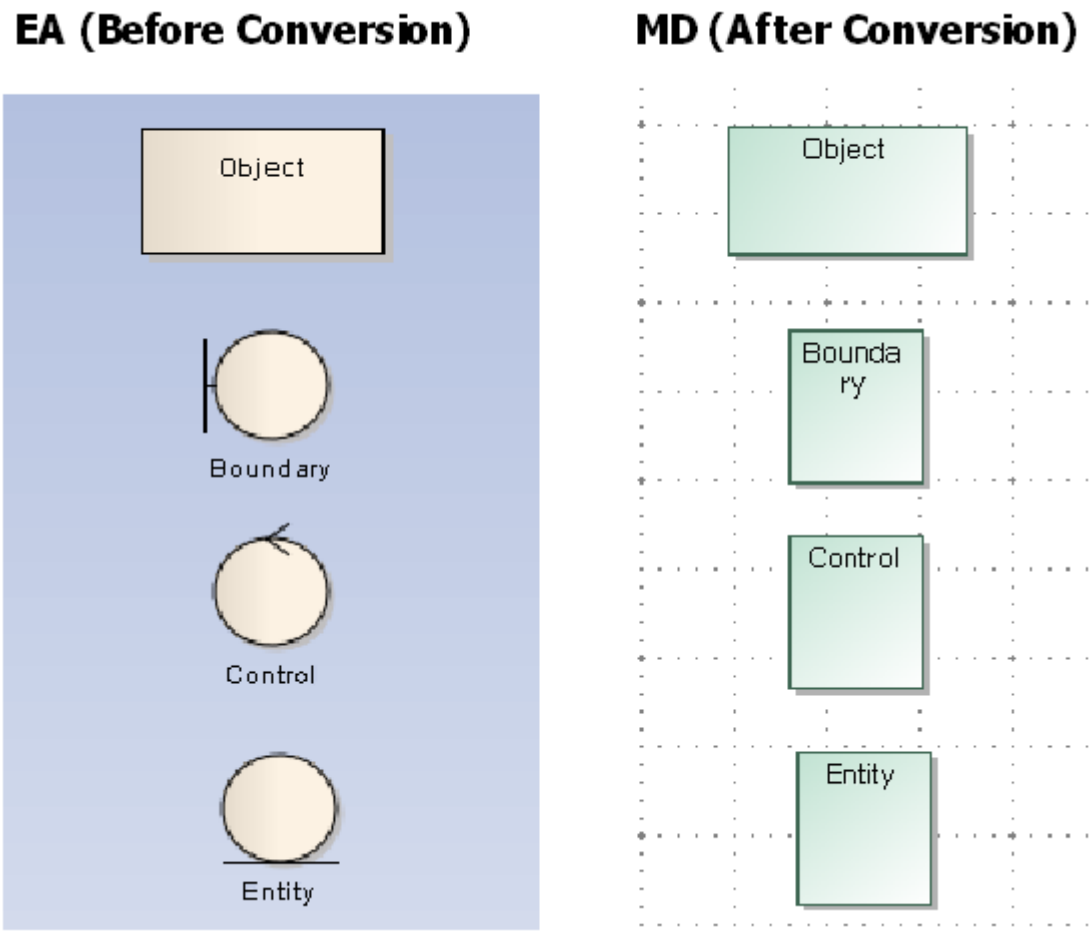
Activation options.

On this page

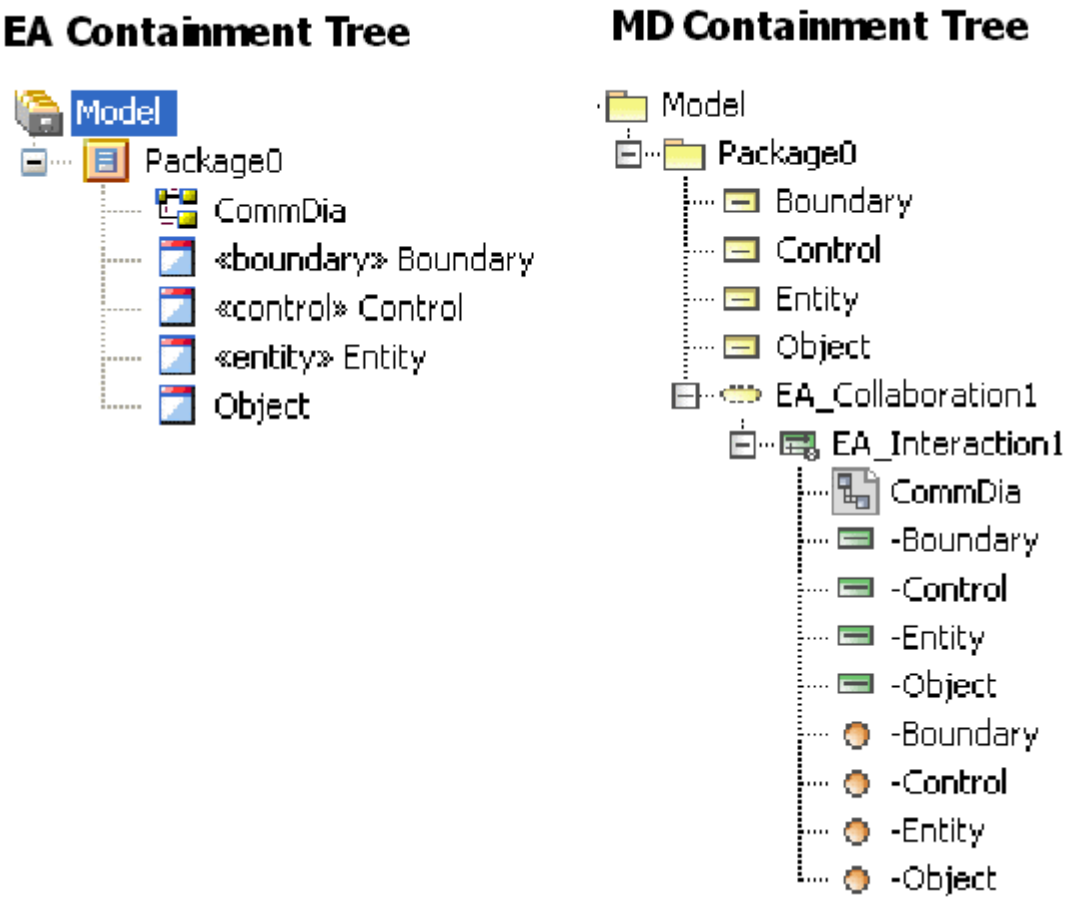
1.5.4 Communication diagram elements

Object, Boundary, Entity, and Control

All Object, Boundary, Entity, Control elements in EA are InstanceSpecifications. After conversion, their UML element types will remain, and a Lifeline and an OwnedAttribute element will be created to represent each of them in the Communication diagram in which they occur.



Object, Boundary, Entity, and Control (diagram view).

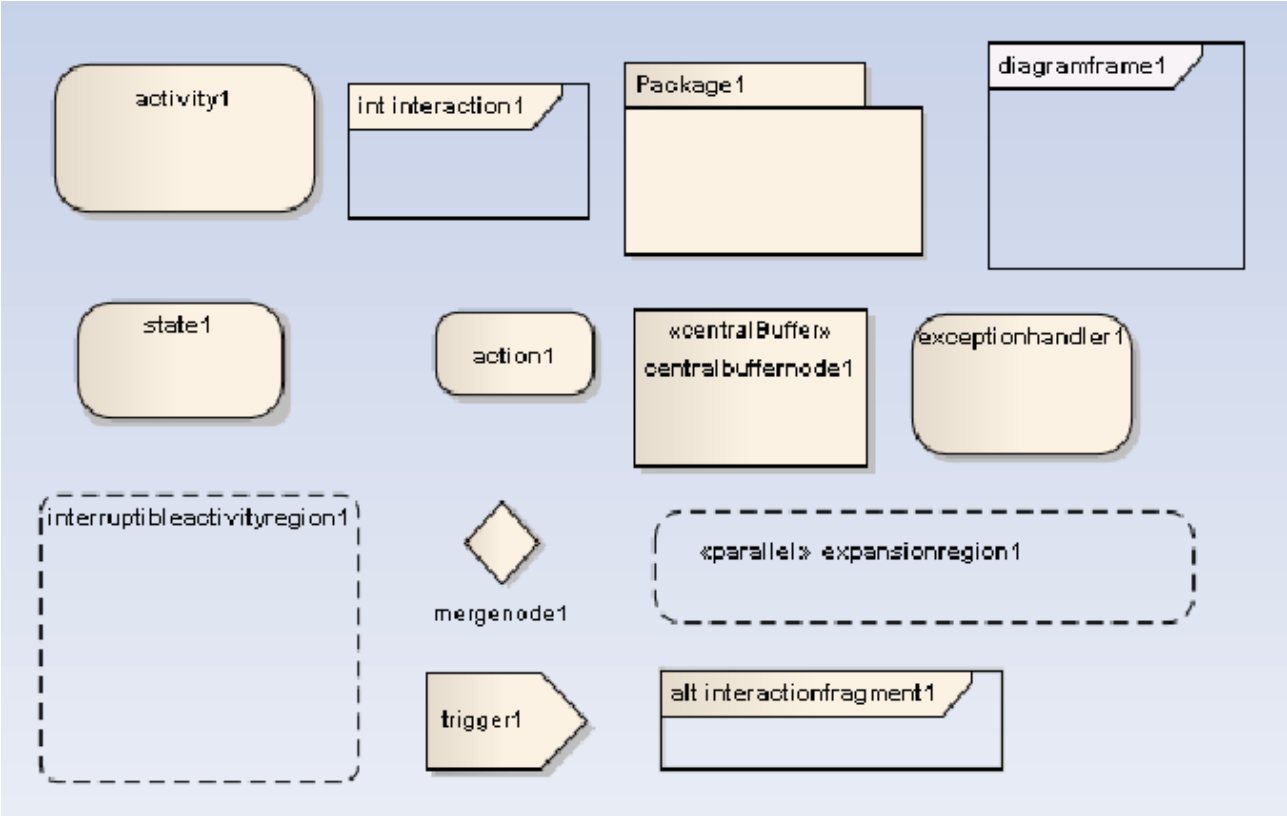


Object, Boundary, Entity, and Control (Containment tree).

Other elements that can be drawn in a Communication diagram will be handled in a similar manner.

Exception Elements

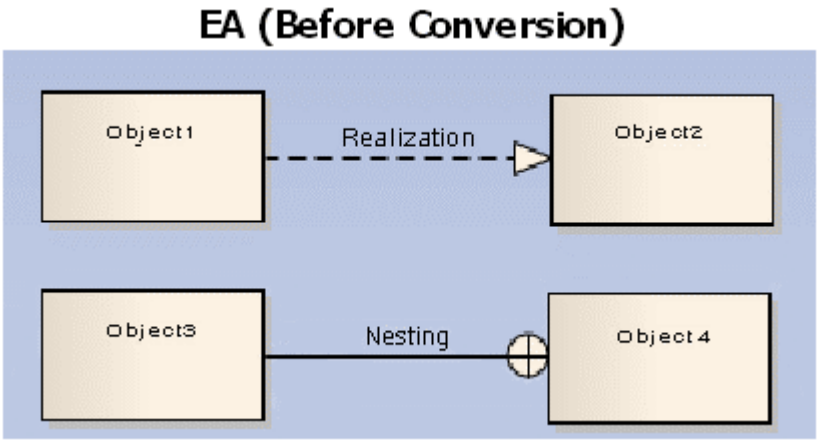
Some of the elements that can be drawn in a Communication diagram using EA are not supported by MagicDraw. Consequently, their displaying parts will not be imported. These elements include Package, Activity, Action, DiagramFrame, State, Interaction, ExceptionHandler, CentralBufferNode, InterruptibleActivityRegion, MergeNode, Trigger, ExpansionRegion, and InteractionFragment.



Exception elements.

Realization and Nesting

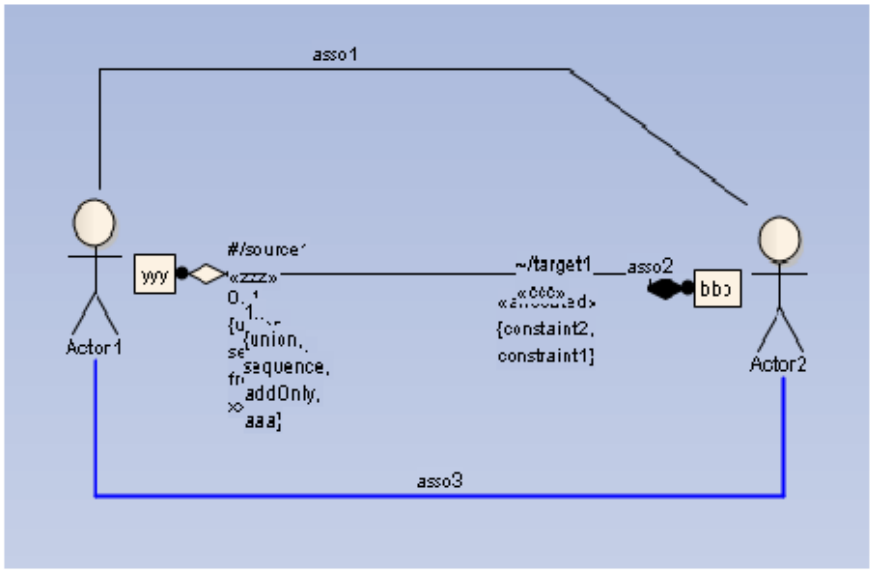
The Realization element type in EA is 'uml:Realization'. The Nesting element type in EA is exported in XMI as *uml:Dependency*. The Realization and Nesting lines in EA are not supported in the Communication diagram; therefore, only their model data will be imported, not the displaying parts.

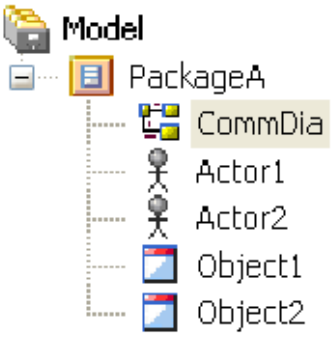


Association

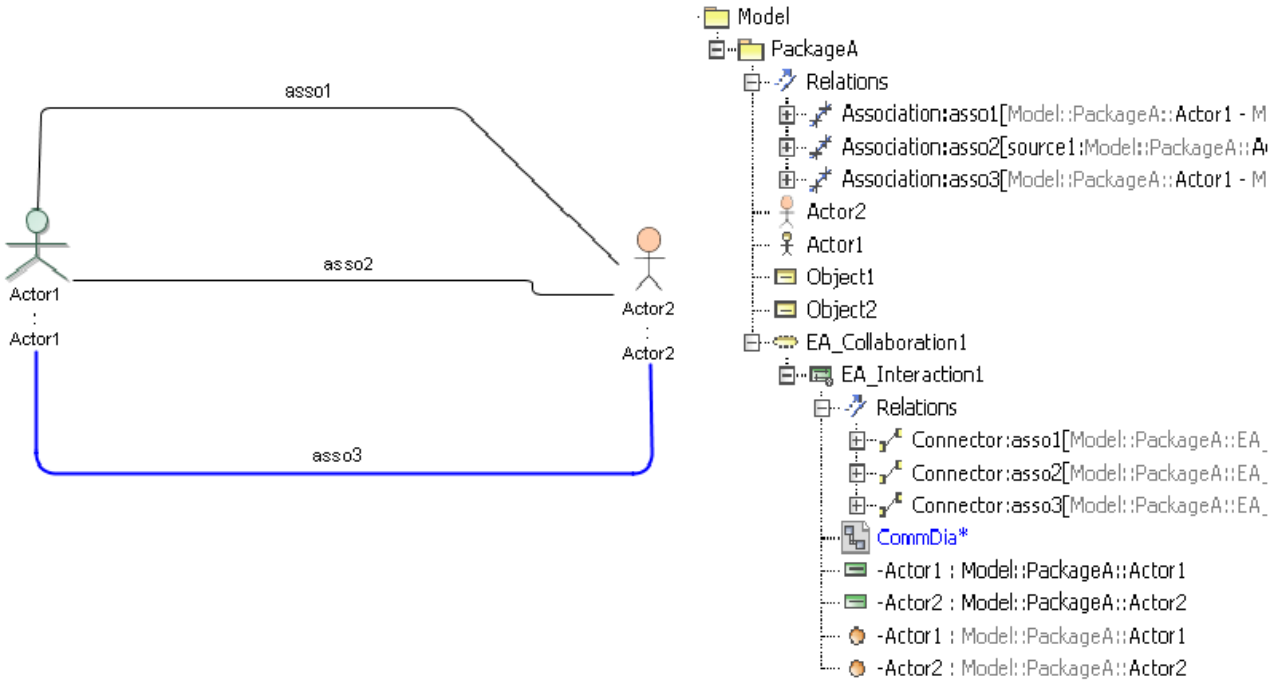
Every Association relationship drawn in a Communication diagram in EA will have a Connector line created for each of them. The elements attached to both ends of the Association line will have a Lifeline element created to represent each of them. The Association lines and the elements attached to them will not be removed. However, the elements that will be shown in the MagicDraw's diagram frame will be the Lifeline elements and the Connector lines created to represent them.

EA (Before Conversion)





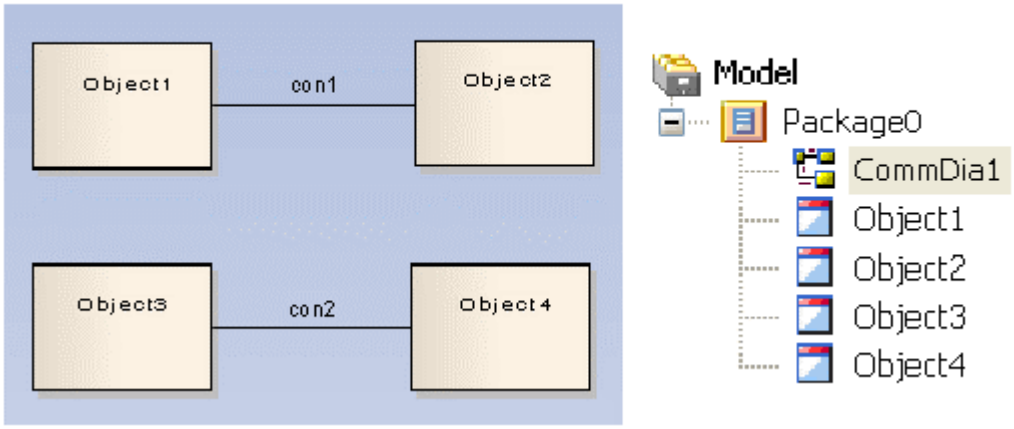
MD (After Conversion)



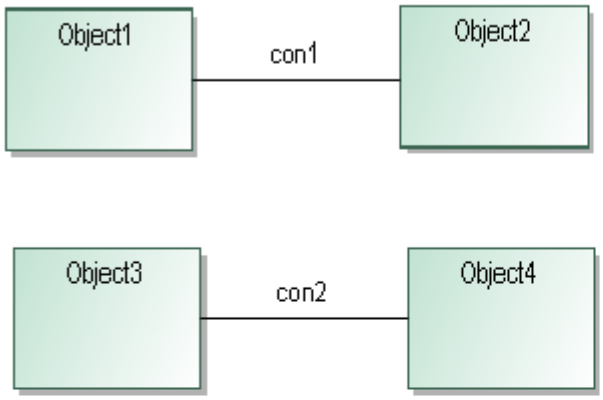
Association.

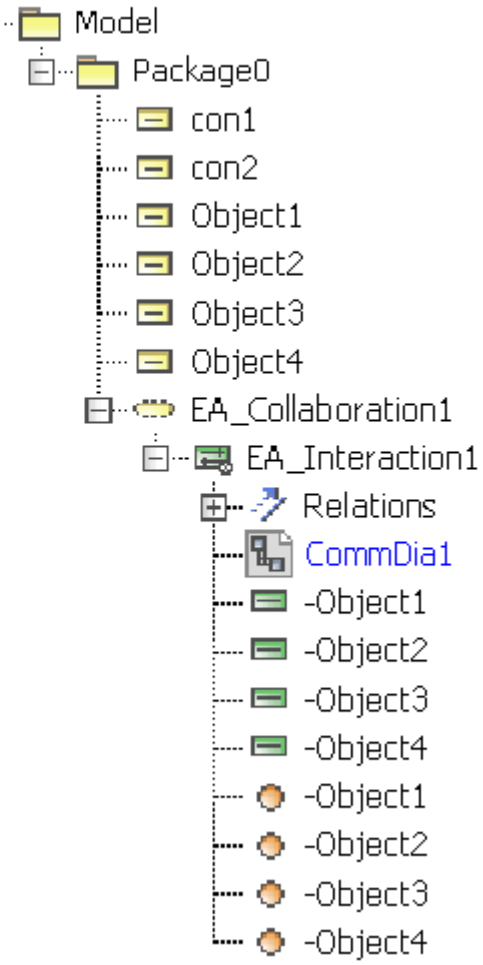
If an Association line is connected to the InstanceSpecification elements, its data will be removed. If an Association line is linked between two InstanceSpecification elements, it will be transformed into an InstanceSpecification element. This is one of the constraints belonging to the Communication diagram.

EA (Before Conversion)



MD (After Conversion)



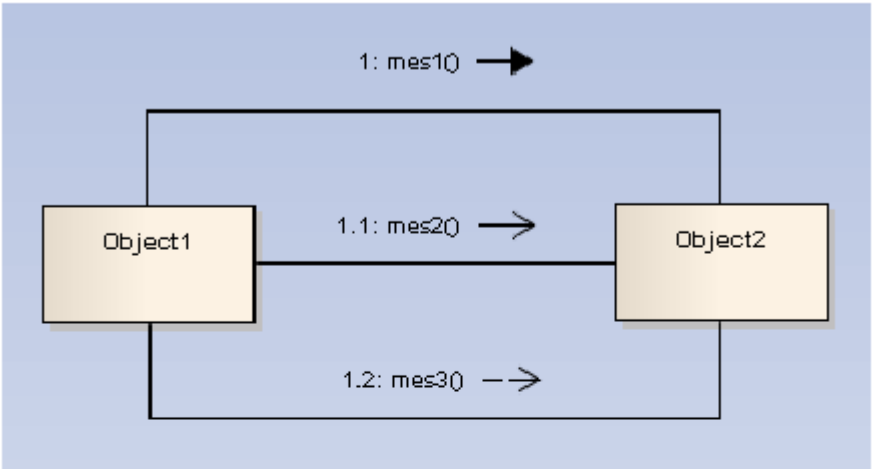


Association Line between InstanceSpecifications.

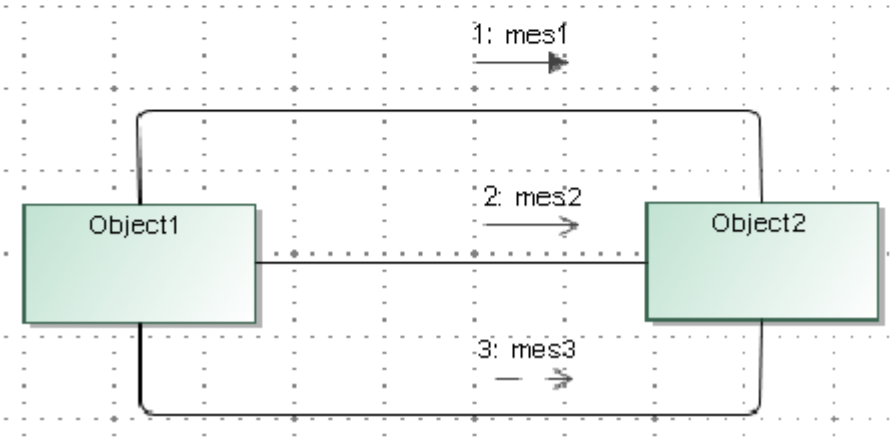
Message

Messages can be created on Connectors and will be imported to MagicDraw.

EA (Before Conversion)



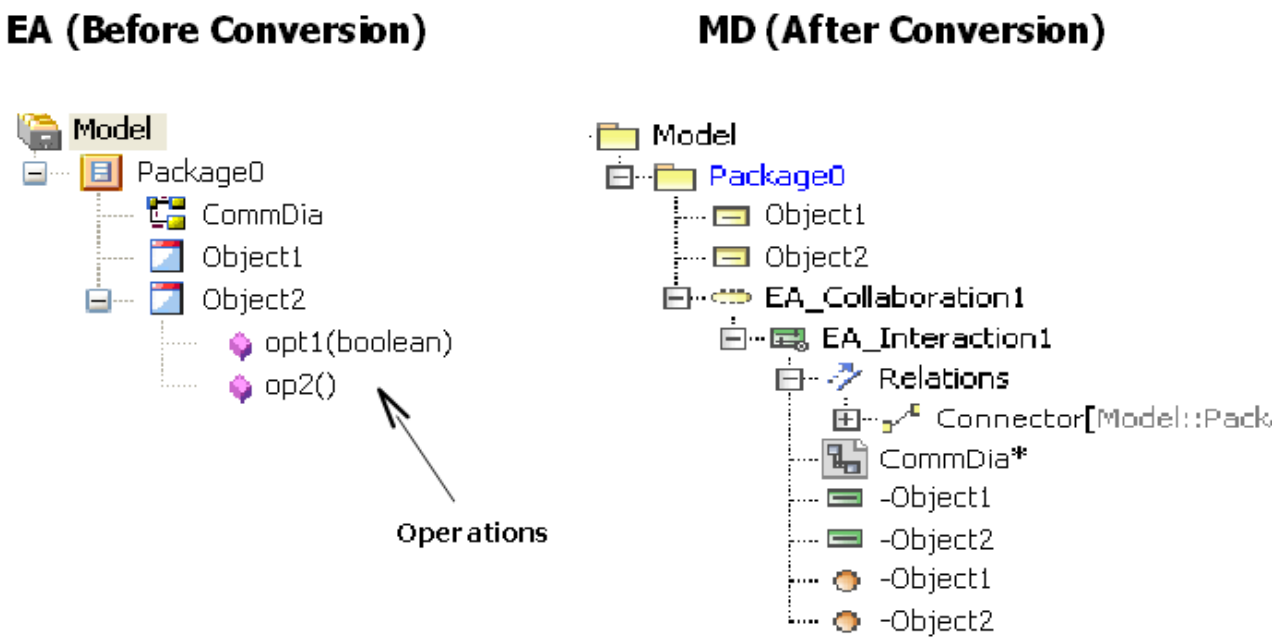
MD (After Conversion)



Message.

Operations

An InstanceSpecification cannot contain Operation elements. If the XMI file from EA has some InstanceSpecification elements containing Operation elements, those Operations will be removed.



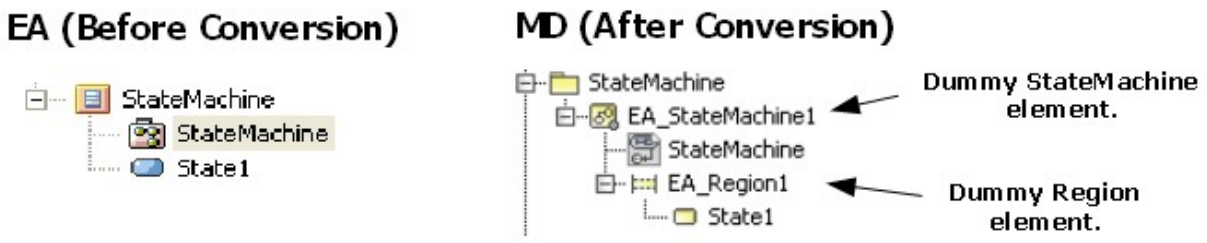
Operations.

On this page

1.5.5 State Machine diagram

StateMachine

After conversion, a dummy StateMachine element will be created. The dummy StateMachine is either created by the XMI exporter from EA or by EA Import plugin. A StateMachine diagram will be placed inside the dummy StateMachine and a dummy Region element will be created to contain all of the StateMachine elements.

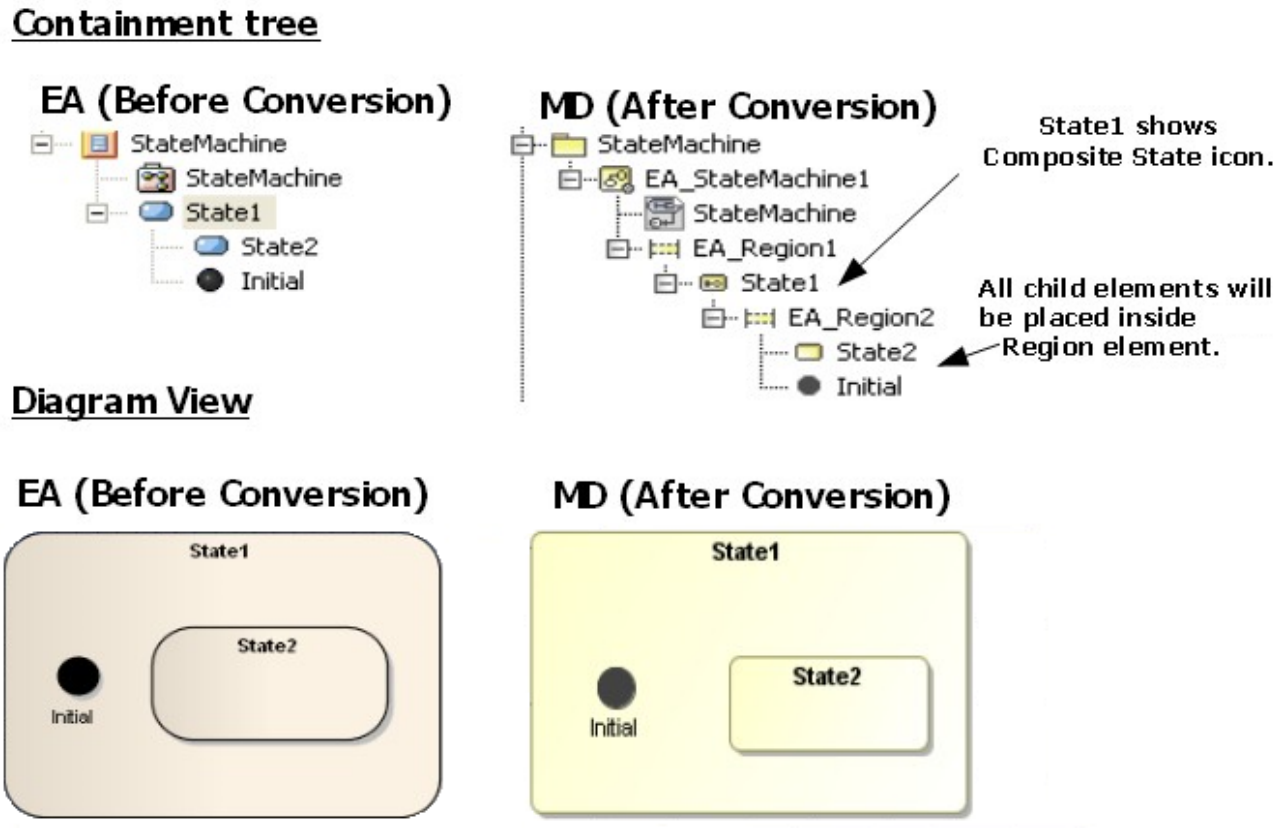


A dummy StateMachine element and a dummy Region element.

State

State containing other elements

A State element containing other elements will be transformed to a Composite State. All of the contained elements will be placed inside the Region element of the State element.



State Containing Other Elements.

State Containing StateMachine

If a State element contains a StateMachine element, the StateMachine element will be brought out and placed at the same level of the Region element of the State element.

Containment tree

EA (Before Conversion)

MD (After Conversion)

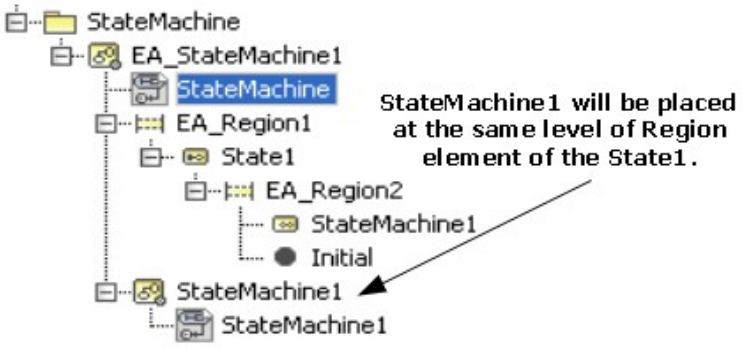
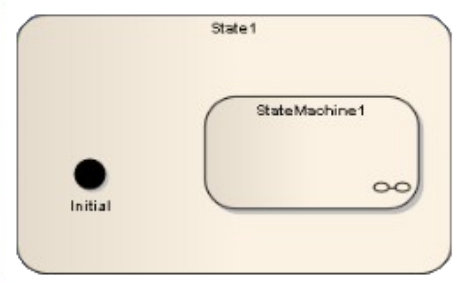
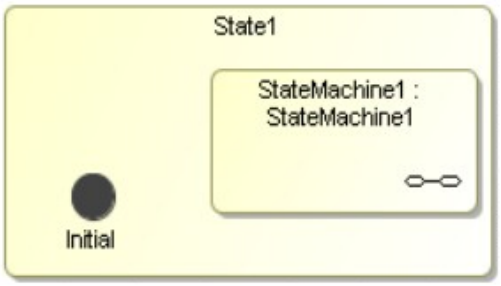


Diagram View

EA (Before Conversion)



MD (After Conversion)



State Containing StateMachine.

State containing Attribute and Operation

If a State element contains Attribute and Operation elements, the Attribute and Operation elements will be removed from both the Diagram view and Containment tree.

Containment tree

EA (Before Conversion)

MD (After Conversion)

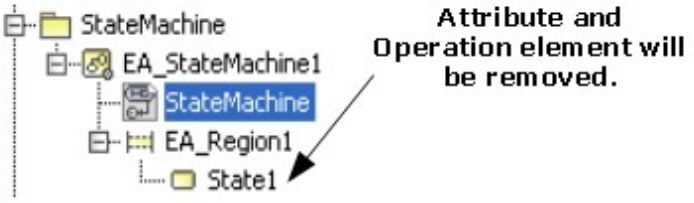
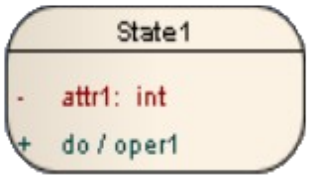
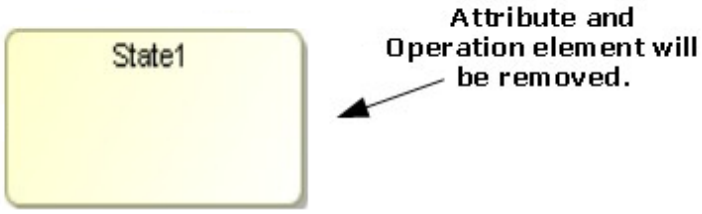


Diagram View

EA (Before Conversion)



MD (After Conversion)



State Containing Attribute and Operation.

The following transformation messages will open:

- Removed element <xmi:id>: State cannot contain Attribute.
- Removed element <xmi:id>: State cannot contain Operation.

State containing Diagram Element

If a diagram element is placed inside a State element, it will be removed.

Containment tree

EA (Before Conversion)

MD (After Conversion)

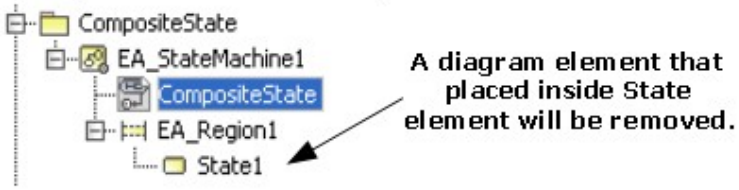
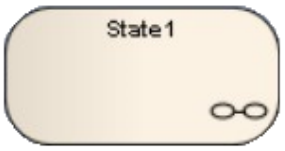


Diagram View

EA (Before Conversion)



MD (After Conversion)

State Containing Diagram Element.

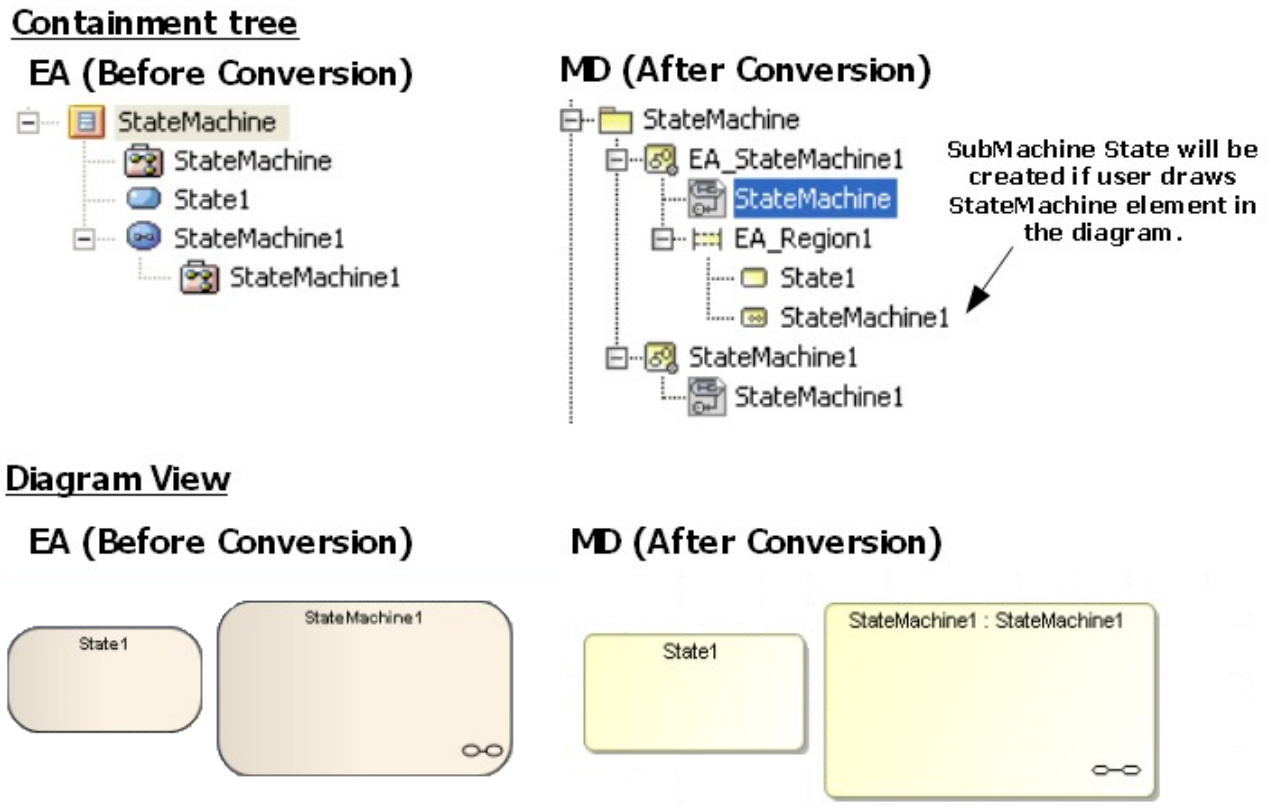
The following transformation message will open:

```
Removed element <xmi:id>: State cannot contain diagram element.
```

StateMachine placed on a diagram

If a StateMachine element is drawn in a StateMachine diagram, an additional SubMachineState will be created to represent the StateMachine.

> **Note**
>
> A SubMachine State is a State whose SubMachine property is set to a StateMachine.



StateMachine Represented by a SubMachine State.

Object

An Object element placed in a State Machine diagram will be removed from the Diagram view. However, its data will be preserved in the Containment tree as an InstanceSpecification.

Containment tree

EA (Before Conversion)

MD (After Conversion)

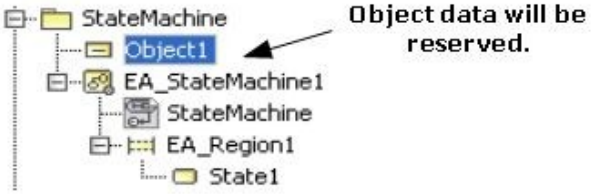
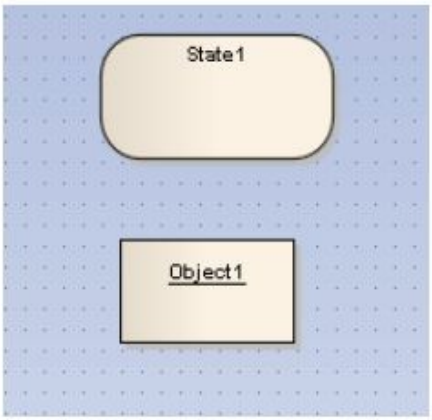
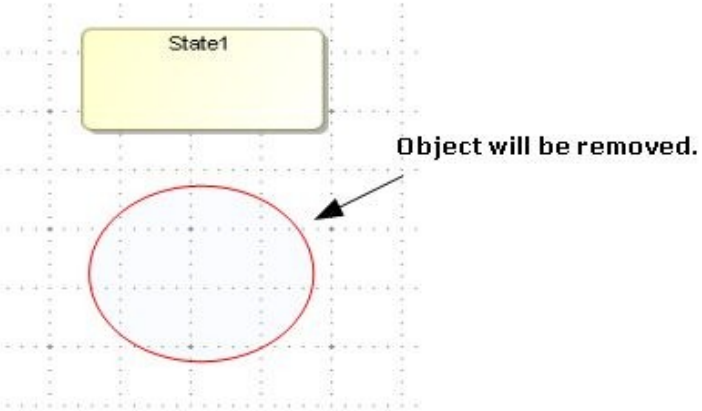


Diagram View

EA (Before Conversion)



MD (After Conversion)



Object.

The following transformation message will open:

```
Removed view <xmi:id>: The view represents element that does not support
in State Machine Diagram.
```

Object containing a State Machine element

In EA, an Object element can contain a State Machine element. After it has been imported to MagicDraw, the State Machine element placed inside an Object element will be removed both from the Diagram view and Containment tree. However, the Object element data will be preserved. All of the Object element data will be placed at the closest owner package of the Object element.

Containment tree

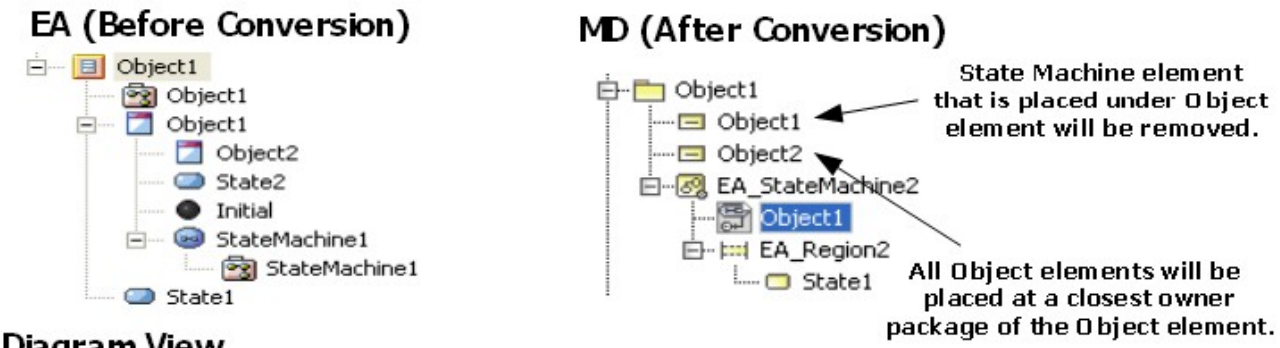
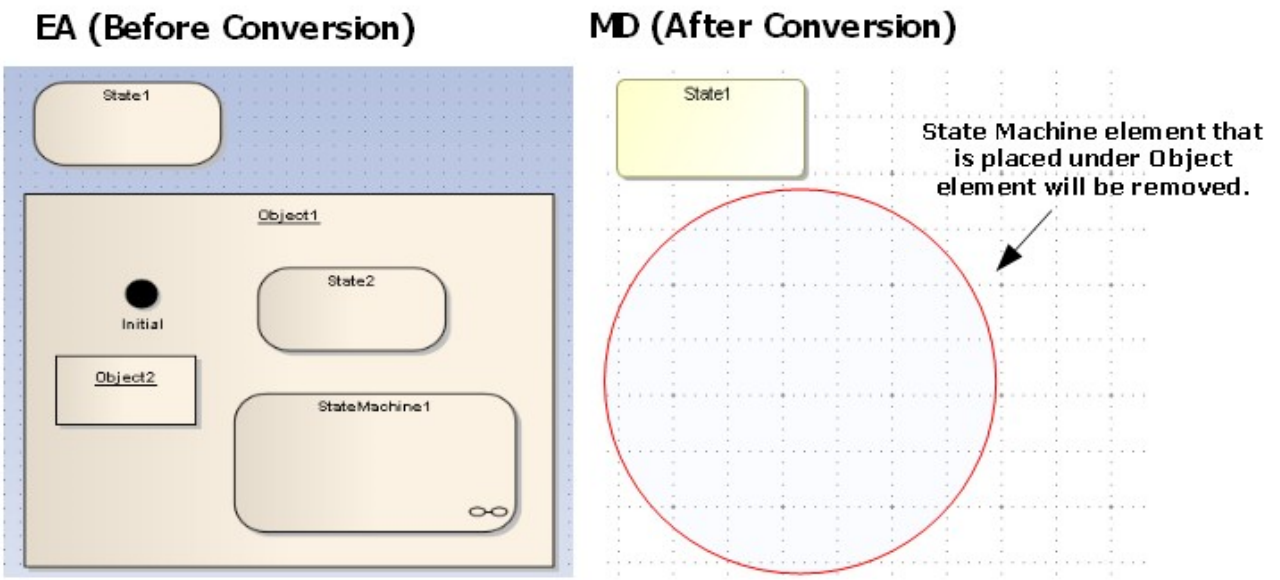


Diagram View



Object Containing State Machine Element.

The following transformation message will open:

```
Removed element <xmi:id>: Invalid element. Instancespecification can not
contain element from State Machine.
```

Synch

A Synch element will be transformed to a Junction element.

Containment tree

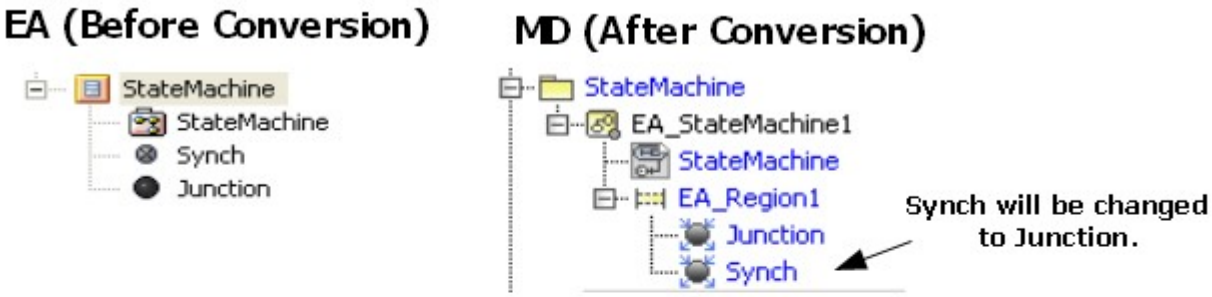
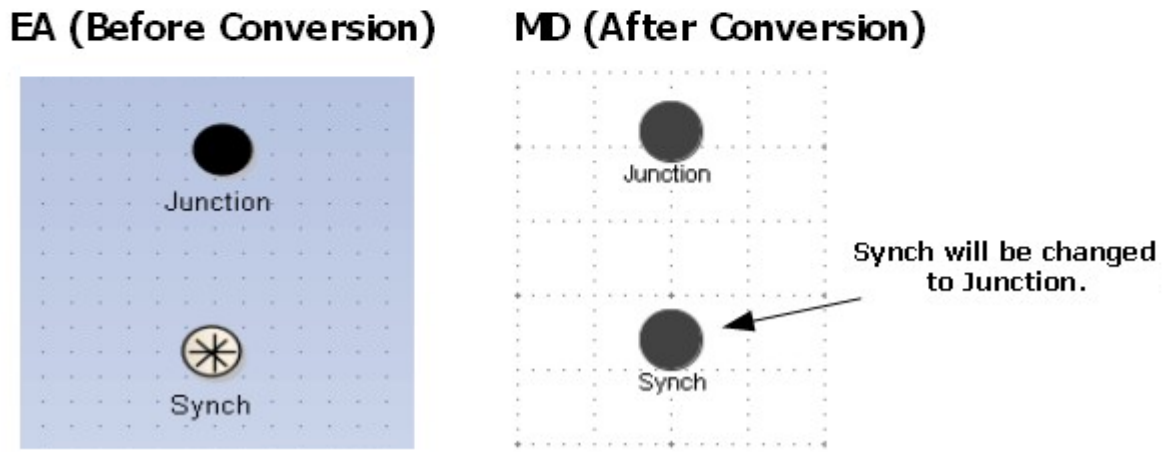


Diagram View



Synch.

The following transformation message will open:

```
Updated element <xmi:id>: Synch updated to Junction.
```

EntryPoint / ExitPoint

An Entry or Exit point placed inside a State (or StateMachine) element will be placed outside a Region element of its parent element in the Containment tree. It will be relocated to the nearest boundary of its parent element in the Diagram view.

Containment tree

EA (Before Conversion)

MD (After Conversion)

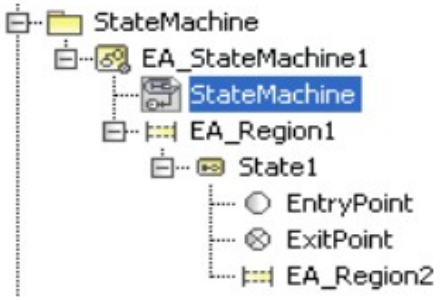
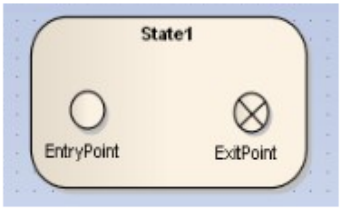
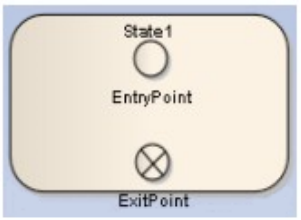


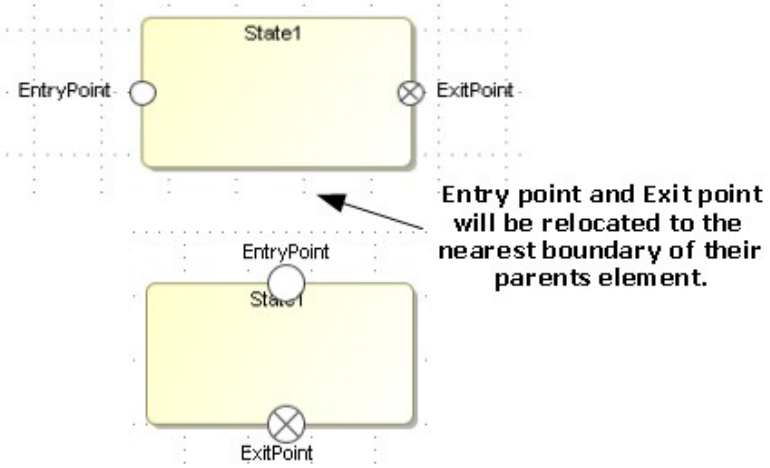
Diagram View

EA (Before Conversion)





MD (After Conversion)



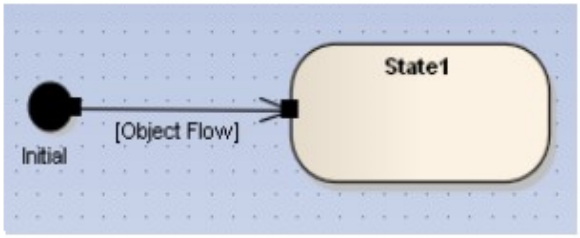
Entry/Exit Point.

Object Flow connecting State Machine elements

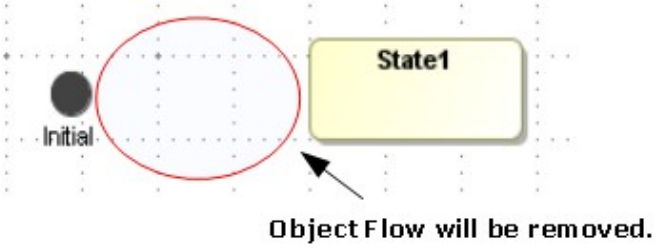
An Object Flow connecting the elements in a State Machine will be removed from both the Diagram view and Containment tree.

Diagram View

EA (Before Conversion)



MD (After Conversion)



ObjectFlow Connecting State Machine Elements.

The following transformation message will open:

```
Removed element <xmi:id>: Invalid ObjectFlow. Source or Target of
theObjectFlow are connected to element from State Machine.
```

Information Flow connecting State Machine elements

An Information Flow connecting the elements in a State Machine will be removed from the Diagram view; however, its data will be preserved.

Containment tree

EA (Before Conversion)

MD (After Conversion)

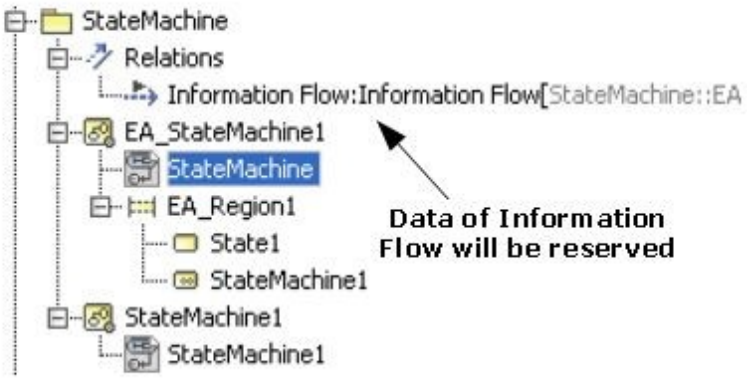
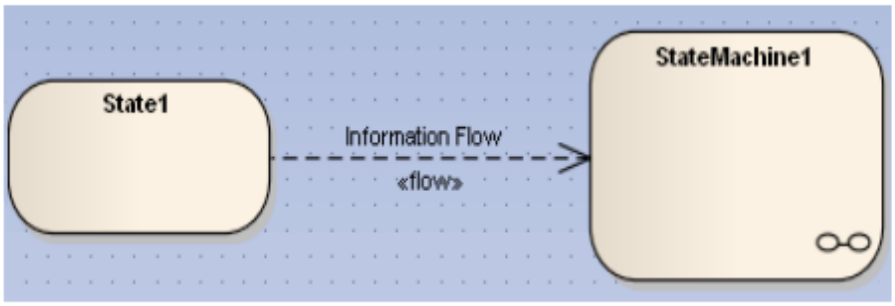
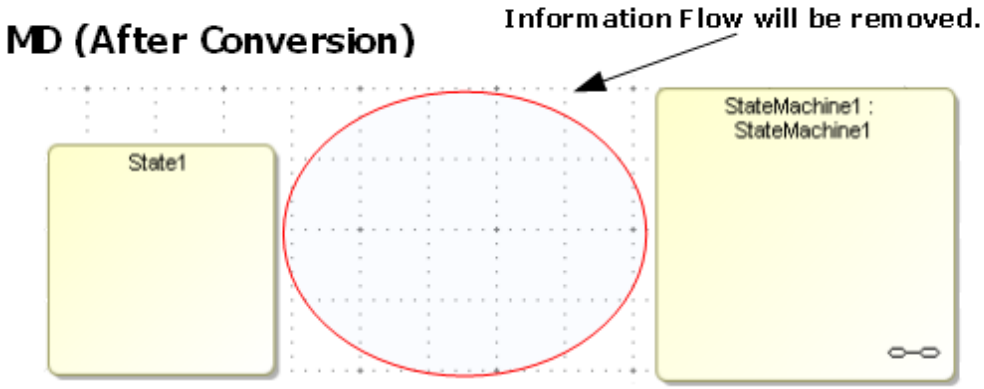


Diagram View

EA (Before Conversion)



MD (After Conversion)



Information Flow, Trace, and Dependency.

The following transformation message will open:

```
Removed element <xmi:id>: The view represents element that does not sup-
port in State Machine Diagram.
```

Trigger

If there is a Trigger element unrelated to any Transition line, a dummy StateMachine will be created to hold the Trigger element. The dummy StateMachine will be named after the parent package name concatenating with 'trigger'.

In the case of a Trigger element related to Transition, the data of the Trigger element will be placed inside its parent Transition. Each Trigger element will be given an event type, represented by an Event element. After conversion, the Event will be placed at the closest owner package of the Event.

Containment tree

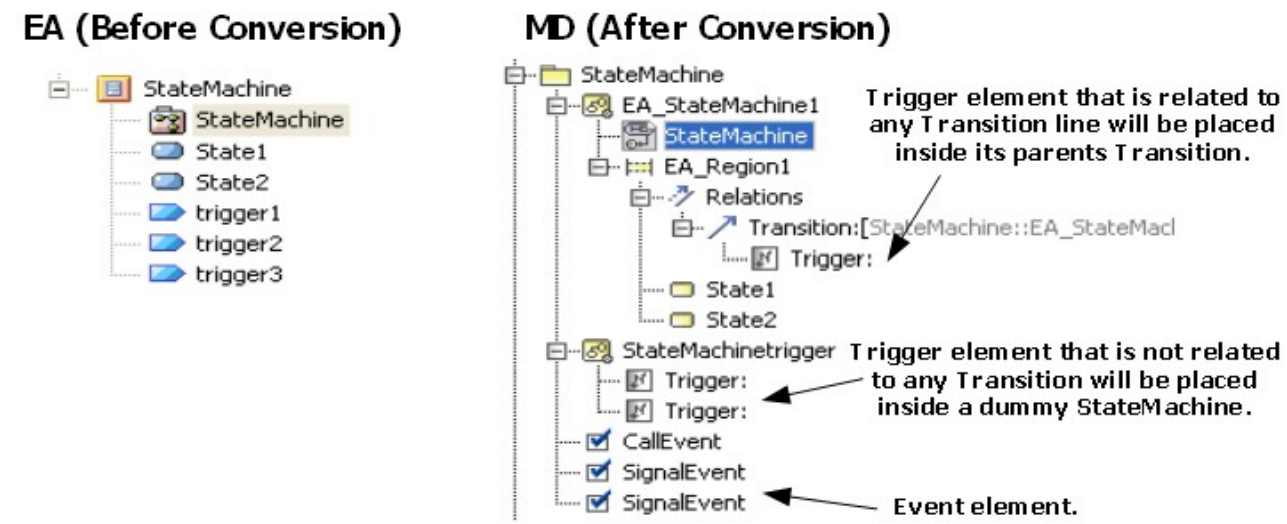
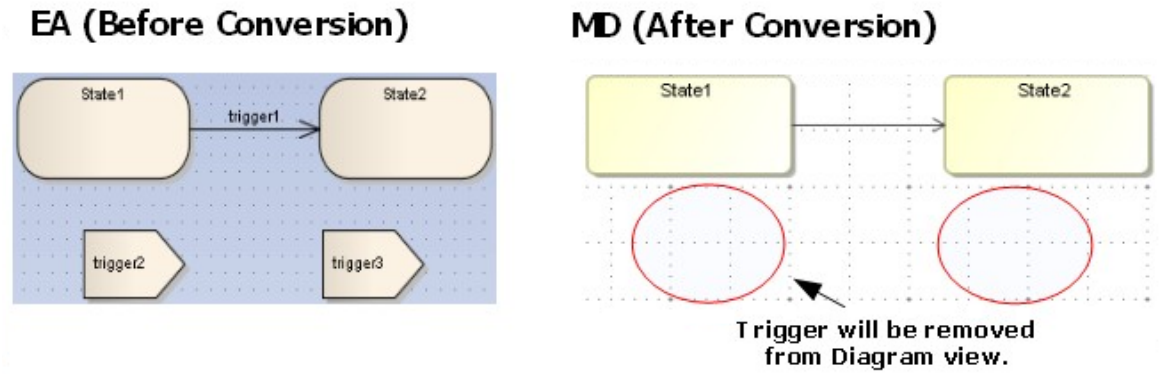


Diagram View



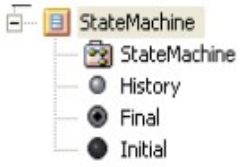
Trigger.

Self Transition

Most of the elements in a StateMachine diagram can have a self transition, except the Initial, Final, and History elements. The self transition in these three elements will be removed from the Diagram view, but the data of the self transition will be preserved in the Containment tree.

Containment tree

EA (Before Conversion)



MD (After Conversion)

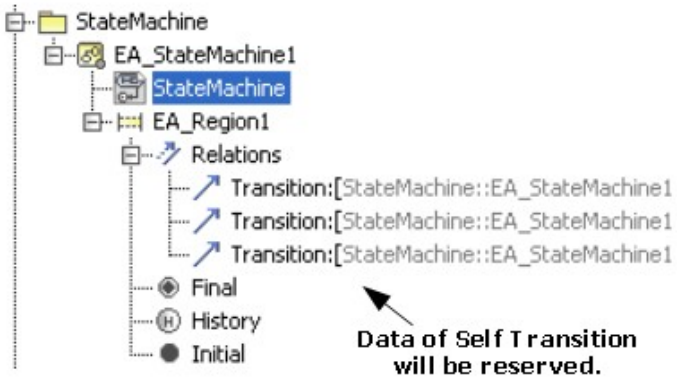
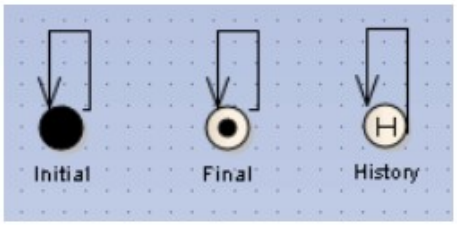
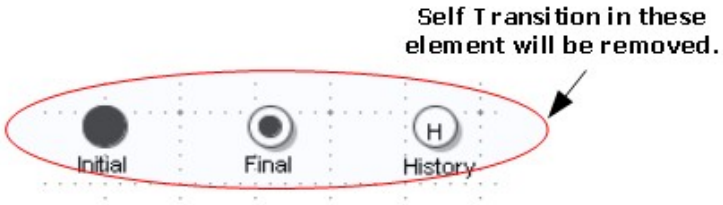


Diagram View

EA (Before Conversion)



MD (After Conversion)



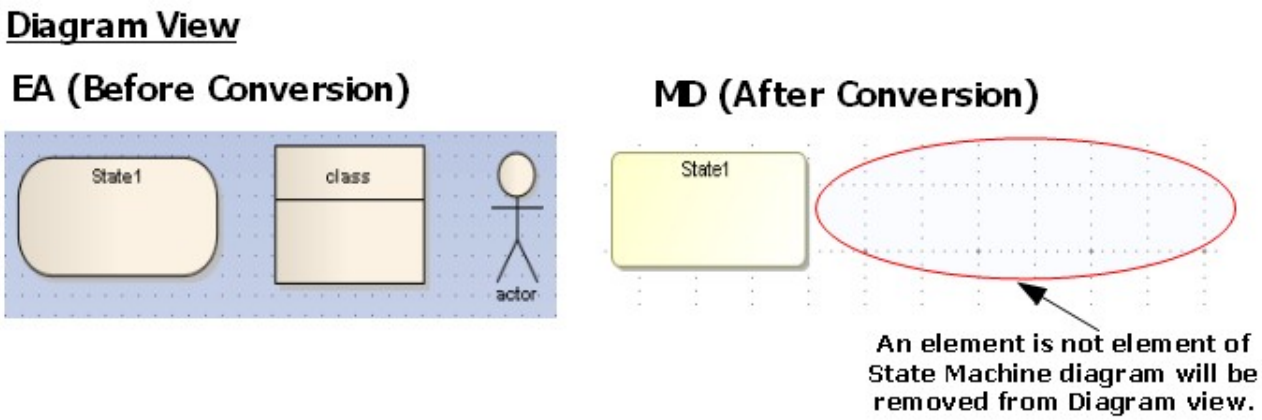
Self Transition.

The following transformation messages will open:

- `Removed view <xmi:id>: Self Transition does not support forInitial.`
- `Removed view <xmi:id>: Self Transition does not support forFinal.`
- `Removed view <xmi:id>: Self Transition does not support forHistory.`

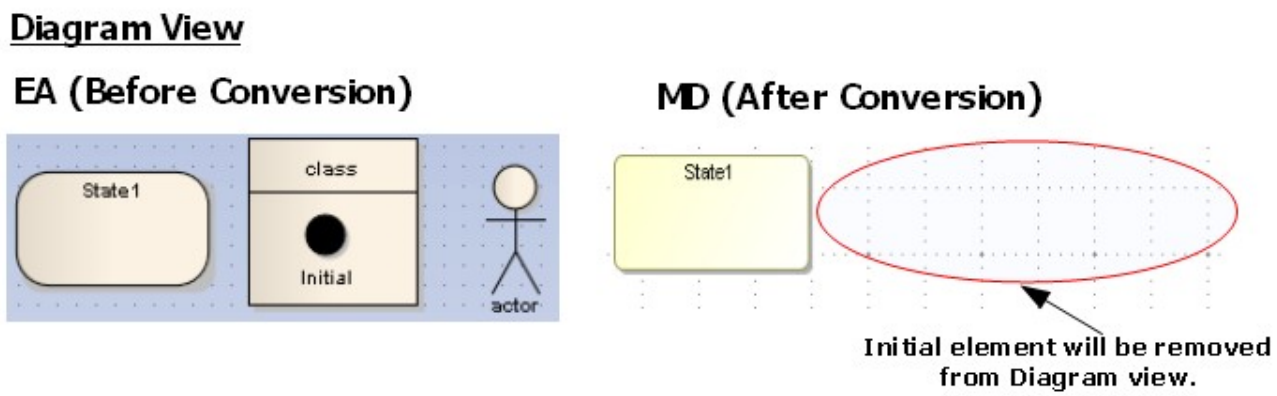
Removed element

Any element which is not the element of a State Machine diagram (such as Class, Actor, Use-case, or Action) drawn in the State Machine diagram will be removed from the Diagram view. However, its data will be preserved in the Containment tree.



Removed element.

If an element which is not the element of a State Machine diagram has a child element and is drawn in the State Machine diagram, the element will be removed from the Diagram view. However, its data will be preserved in the Containment tree.



Removed Element with a Child Element

The following transformation message will open:

```
Removed view <xmi:id>:The view represents element that does not support
in State Machine Diagram.
```

On this page

1.5.6 Composite structure diagrams

This section presents additional EA specific Composite Structure diagram information.

Import elements

EA differs from MagicDraw in the Composite Structure diagram's content elements design. The table below shows the differences by focusing on the EA's elements and how they will be transformed or converted into a format MagicDraw can correctly load and display. The element names shown in the table are the same in both EA and MagicDraw's GUI. The contents in the brackets ([...]) are the XMI element type references.

Enterprise Architect	MagicDraw
Interaction	
Class [uml:Class]	Class [uml:Class]
Interface [uml:Interface]	Interface [uml:Interface]
Part [uml:Class] Part [uml:Property]	Part [uml:Property]
Port [uml:Port]	Port [uml:Port]
Collaboration [uml:Collaboration]	Collaboration Use [uml:Collaboration]
Expose Interface [not exist]	

Connector [element type is not exist]	Connector [uml:Connector]
Assembly [element type is not exist]	
Delegate [element type is not exist]	Connector [uml:Connector]
Role Binding [uml:Dependency]	Role Binding [uml:Dependency]
Represents [uml:Dependency]	Dependency [uml:Dependency]
Occurrence [uml:Dependency]	Dependency [uml:Dependency]

Conversion details

Dummy class

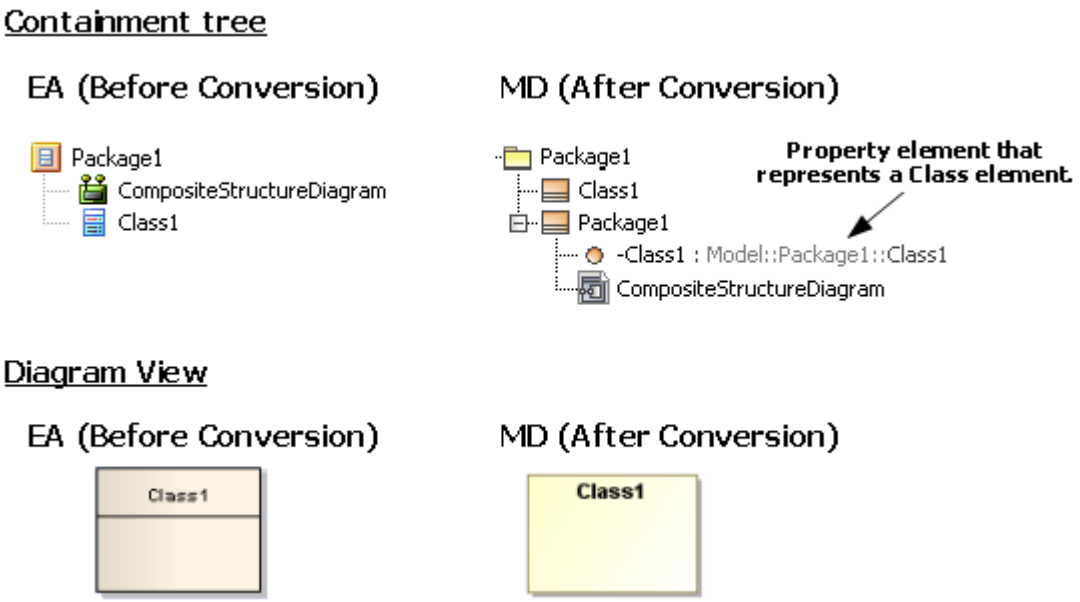
In MagicDraw, the Composite Structure diagram needs a Context element to contain itself. However, in EA there is no Context element. After conversion, a dummy Class element will be created to represent the Context element of Composite Structure diagram.

A dummy Class will be named after the closest owner package of the Composite Structure diagram. The Composite Structure diagram and all of the Composite Structure elements will be placed inside the dummy Class element.

Dummy class.

Class

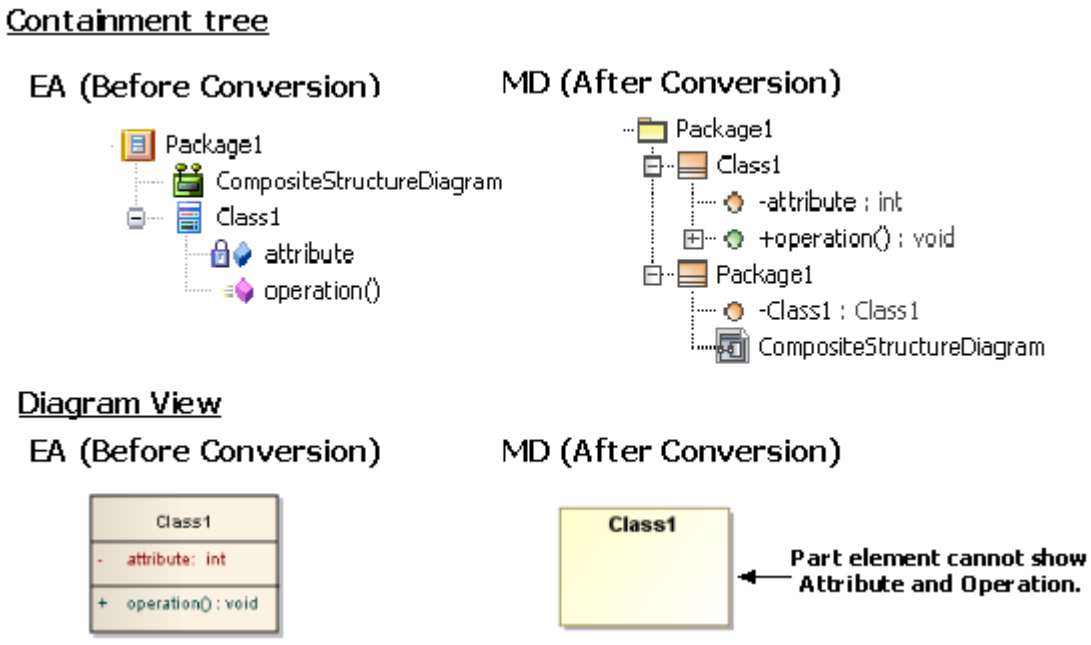
The Class element data will be normally copied even if the Class element is drawn in a Composite Structure diagram. A Property element will be created to represent the Class element. The Property element will be named after the Class element.



Property Representing a Class Element.

Class with Attributes and Operations

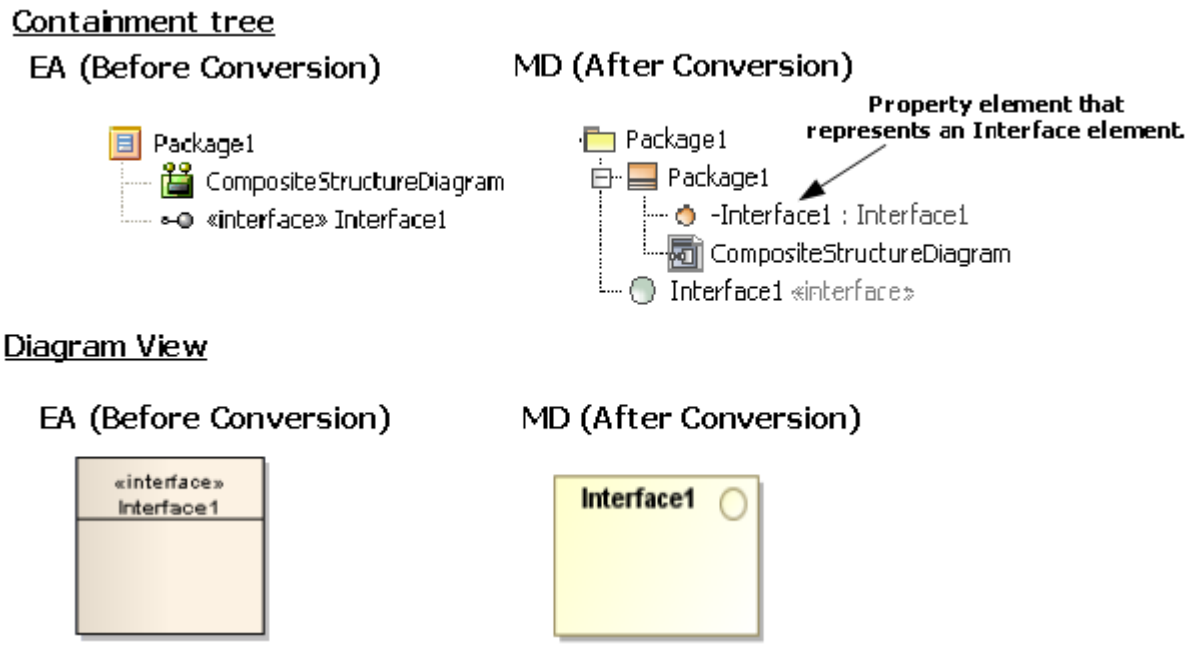
The Class view in EA will be converted to a Part in MagicDraw. Attributes and operations will not be shown in the Composite Structure diagram.



Part Element after Conversion.

Interface

The Interface element data will be normally copied even if the Interface element is drawn in a Composite Structure diagram. A Property element will be created to represent the Interface element. The Property element will be named after the Interface element.



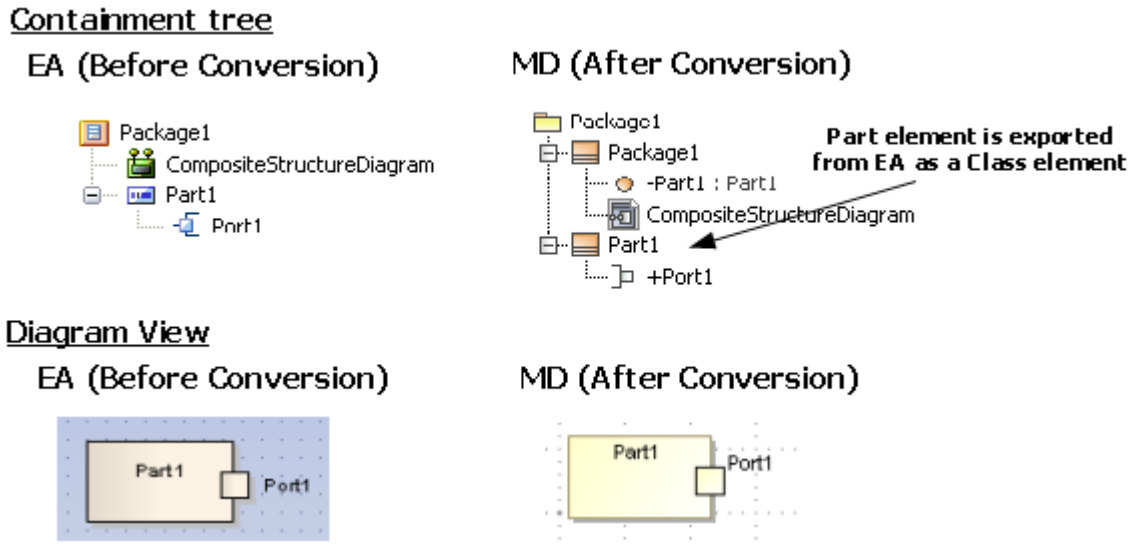
Property Representing Interface Element.

Part

There are two types of Parts: (i) Part with port; and (ii) Part that sets type to other elements.

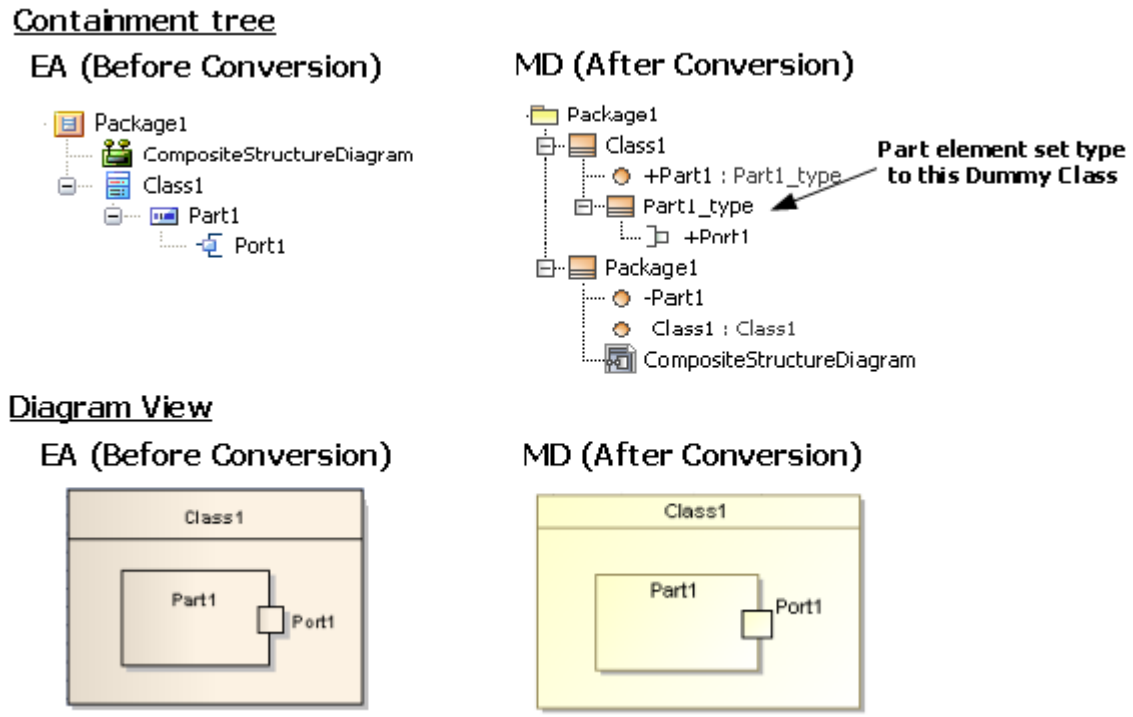
Part with Port

If you assign the type of a Part element to a Class, Component, or Node, the Part element can have a Port (see Port (see page 82) for more information). If a Part is not nested to any element, the Part element will be exported from EA as a Class. Therefore, this Part element can have a Port.



Part that does not Nest Element(s) with Posts.
If a Part is nested to an element and the type is not set, then the Part type will be used to set type to a dummy Class. This particular Part can have a Port. A dummy Class will be created at the same level of

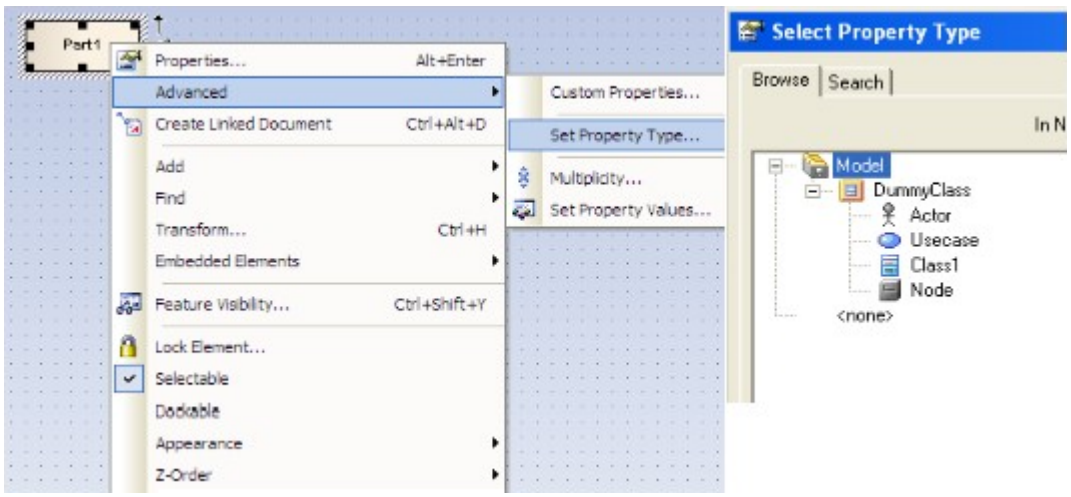
the Part element that sets type to it. The dummy Class will be named after the Part concatenating with '_type'.



Part that Nests Element(s) with Post.

Part that sets type to other elements

The type of a Part can be set to another element (such as a Class, Actor, Component, or Usecase) by right-clicking the Part element and selecting **Advanced** > **Set Property Type** > **Select Property Type.**



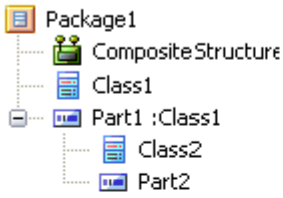
Setting Part Type.
If the Part type is set to a type that cannot be the owner of a port, the port will be removed.

In MagicDraw, the Property element data cannot have any elements. However, in EA, it can have elements. Therefore, if you draw a Part element and it has an element inside it, that particular element will be relocated to an element, which is the Part type.

If a Part is not nested to any element, the Part element will be exported from EA as a Class. The element inside the Part will not be relocated. If that particular Part assigns a type to other elements, its type will always be set to Class.

Containment tree

EA (Before Conversion)



MD (After Conversion)

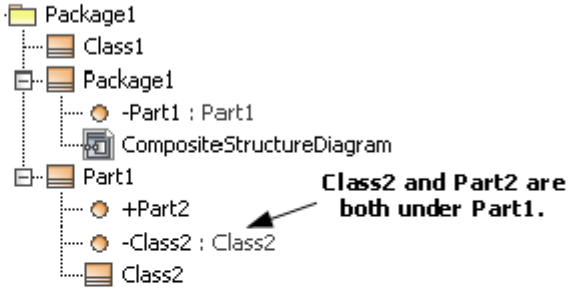
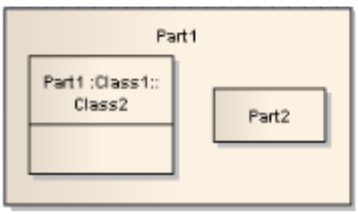
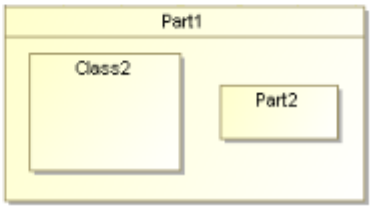


Diagram View

EA (Before Conversion)



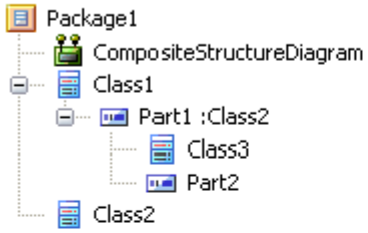
MD (After Conversion)



Part that Sets Type to Other Elements.
If a Nested Part whose type is set to another element (such as a Class, Actor, or Usecase) and this Part contains Nested elements, all of the Nested elements of the Part will be relocated to the element that is the type of this Part.

Containment tree

EA (Before Conversion)



MD (After Conversion)

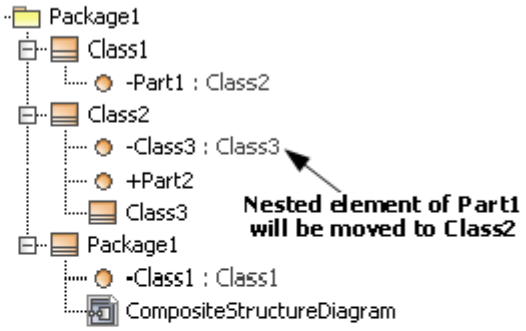
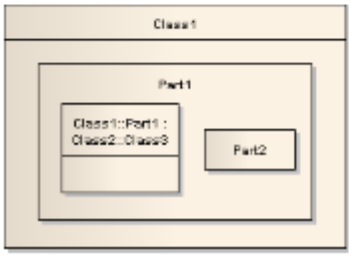
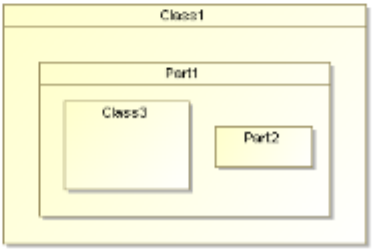


Diagram View

EA (Before Conversion)



MD (After Conversion)

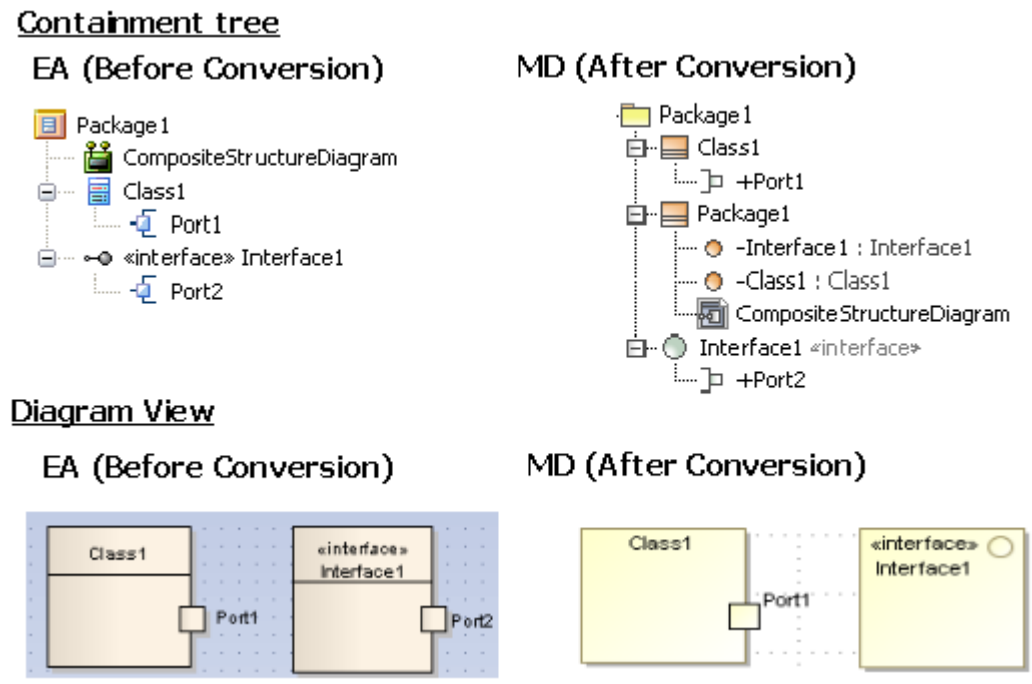


Part that Sets Type to Other Elements has Nested Element Inside Itself.

Port

In MagicDraw, a Property element that can have a Port is the Property that sets type to a Class, Component, or Node, and other elements cannot have a Port. However, in EA, most of the elements can have a Port.

If a Port is created with an element that is not a Class, Component or Node, the Port will be removed from the diagram view. However, its data will be preserved.



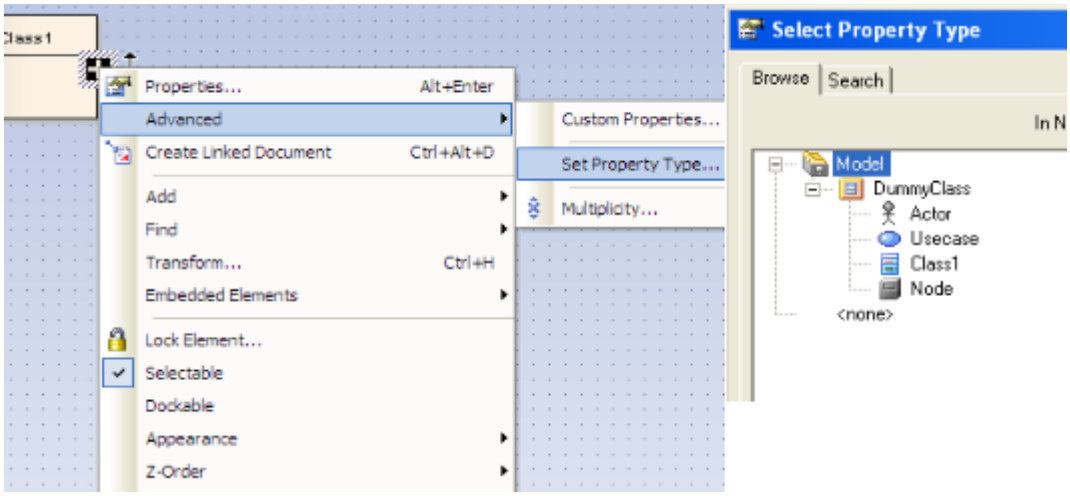
Only Class, Component, and Node Can Have a Port.

The following transformation message will open:

```
Removed view <xmi:id>: Invalid Port. Port can be added to Part that its
type is set to Class, Component and Node only.
```

Port whose type is set to other elements

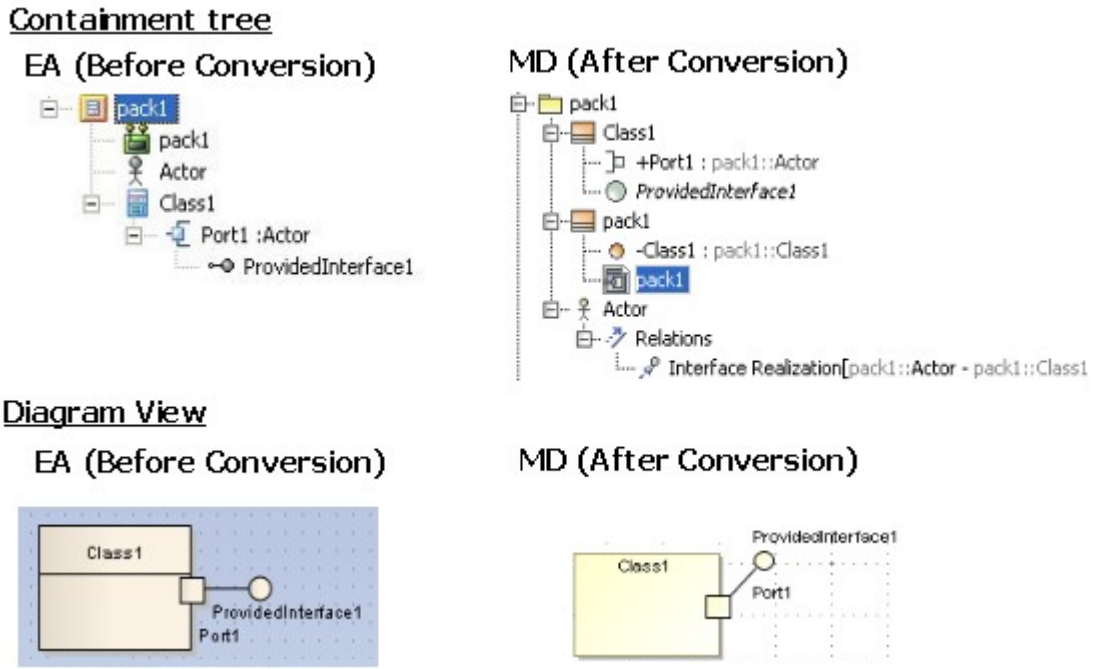
You can set the type of Port element to another element (a Class, Actor, Component, or Usecase) by right-clicking the Port element and selecting **Advanced** > **Set Property Type** > **Select Property Type.**



Setting Port Type.

Port whose type is set to another element and containing Expose Interfaces

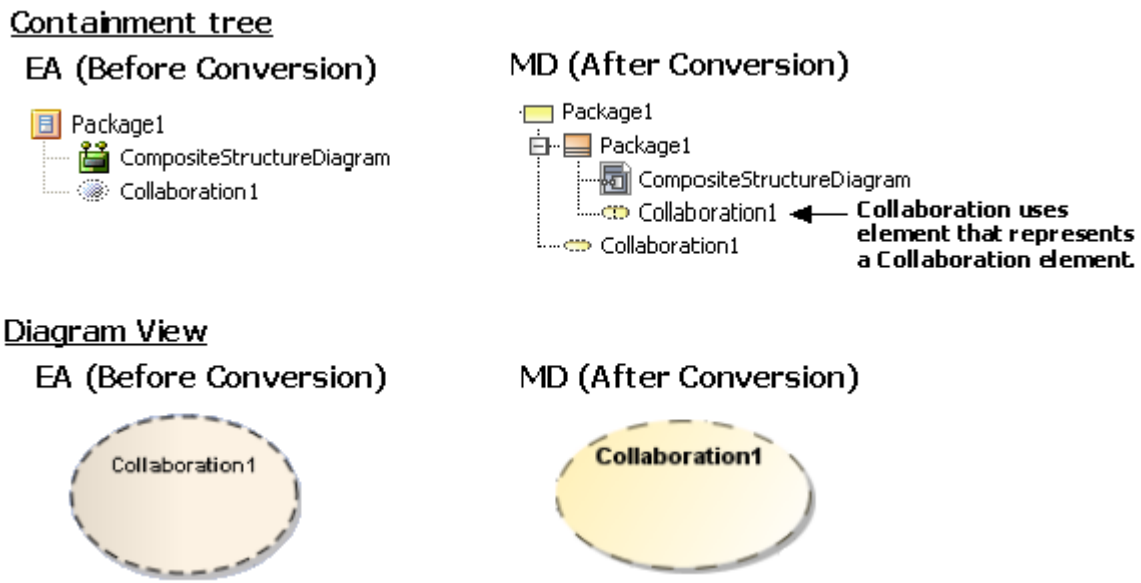
If an Expose Interface is created on a Port and the Port type is set to another element, the Expose Interface data will be relocated to the Port's type element. For example, if the Port1 type is set to Actor, an Interface Realization (the Expose Interface data) will be placed inside the Actor after conversion.



Port that Sets Type to Other Element Contains Expose Interface.

Collaboration

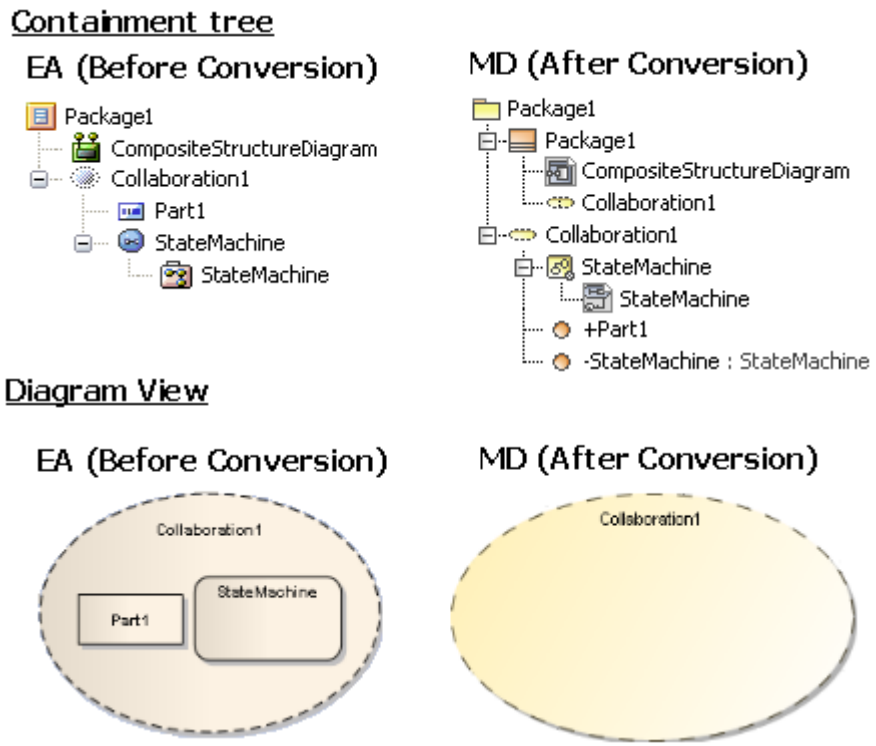
The Collaboration element data and its nested data will normally be copied. However, if the Collaboration element is drawn in a Composite Structure diagram, a Collaboration use element will be created to represent the Collaboration element. The Collaboration use element will be named after the Collaboration element.



Collaboration use.

Collaboration containing other elements

A Collaboration can contain only Property, Activity, State Machine, and Interaction. Elements other than these will be removed. If any element is drawn inside the Collaboration, it will be removed from the diagram view.



Collaboration Containing Other Elements.

The following transformation messages will open:

> - `Removed element <xmi:id>: Invalid Element. Collaboration can contain Property, Activity, State Machine and Interaction only.`
> - `Removed view <xmi:id>: Invalid Element view. Collaboration cannot be contained in any element view.`

Expose Interface

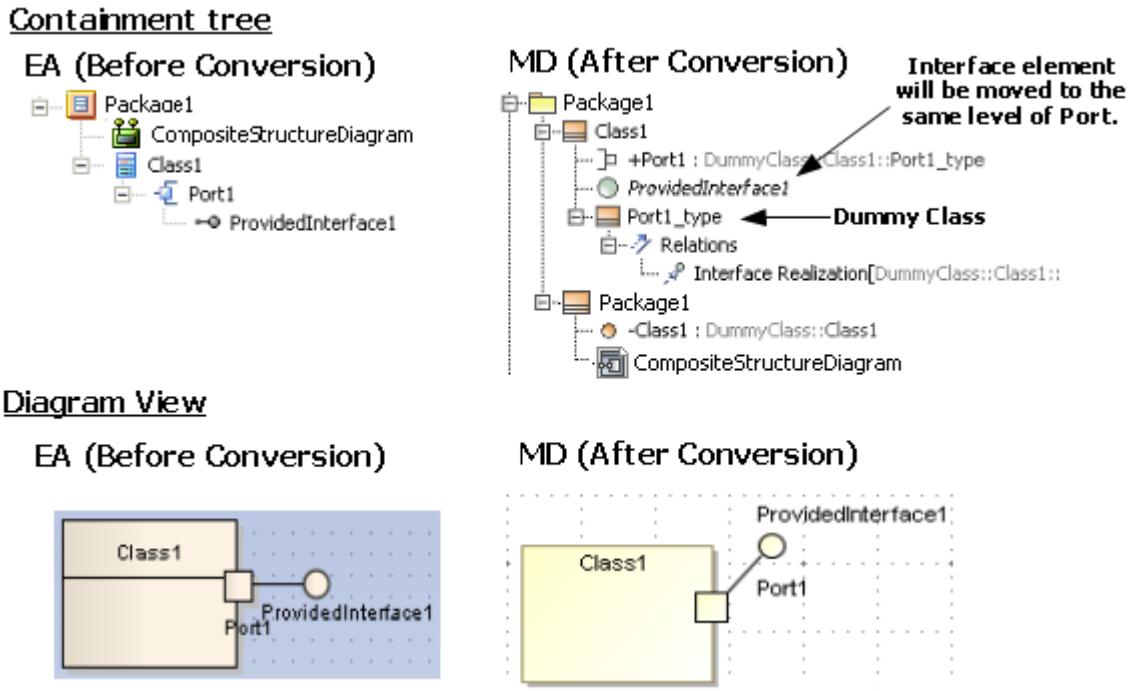
Expose Interfaces are the Provided and Required interfaces in MagicDraw.

In MagicDraw, you can draw a Provided Interface or Required Interface in a Port only and set the type of the Port to another element (an Actor, Usecase, or Class). You can create that particular Port on a Part element whose type is set to a Class, Component, or Node only.

Expose Interface with Port

If you draw an Expose Interface in a Port and the Port does not assign a type to any elements, that particular Port will be used to set type to a Dummy Class. The Dummy Class will be created and named

after the Port concatenating with '_type'. The Interface element will be moved to the same level of the Port.



Expose Interface with Port.
If an Expose Interface is created on a Port and the type of the Port is set to another element, the Expose Interface data will be relocated to the Port's type element (see (ii) Port whose type is set to another element and containing Expose Interfaces for more information).

In MagicDraw, some elements can be used as a Port type. These elements can have either a Provided Interface or Required Interface, or both.

The following is a list of the Port's type elements with specific conditions:

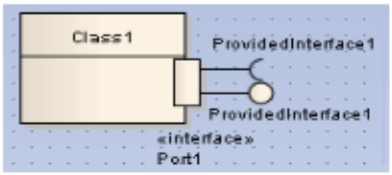
- Interfaces can only have a Provided Interface.
- Artifacts cannot have a Provided Interface.
- Information Items cannot have a Provided Interface.
- Signals cannot have a Provided Interface.
- Components always have both Provided and Required Interfaces.
- The other elements can have both Provided and Required Interfaces.
- After conversion, if the type of a Port is set to an Interface and the Port has an Expose Interface, only the Provided Interface will be shown and the Interface Realization data will be removed.
- Artifacts, Information Items, and Signals cannot hold Interface Realization. If the type of a Port is set to one of them, the Provided Interface will be removed. However, this does not affect the Required Interface.
- If the type of a Port is set to a Component and the Port has an Expose Interface, it will always show both the Provided and Required Interfaces.

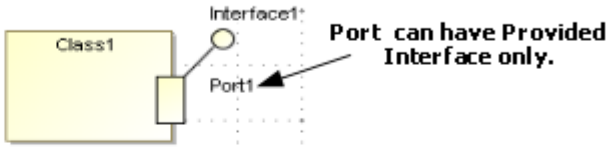
Diagram View

EA (Before Conversion)

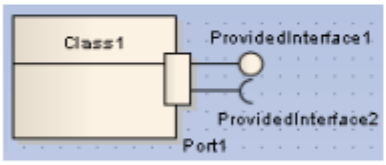
MD (After Conversion)

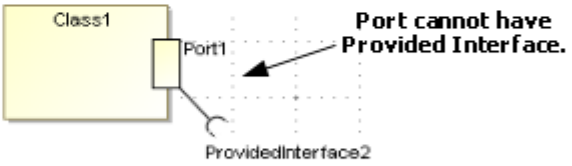
Set Port type to Interface



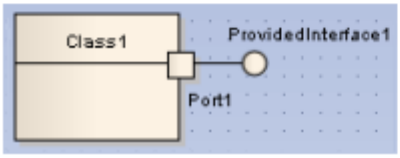


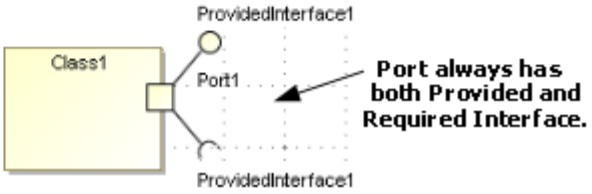
Set Port type to Artifact, Information Item and Signal





Set Port type to Component





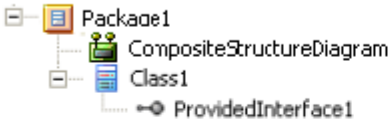
Port's Type Element with Specific Conditions.
The following transformation message will open:

```
Updated element <xmi:id>: Provided Interface conflicts with Port type.
The Port type is updated to an owner of Interface Realization.
```

If an Expose Interface is created with an element that is not a Port, the element will be removed both from the Containment tree and diagram view.

Containment tree

EA (Before Conversion)



MD (After Conversion)

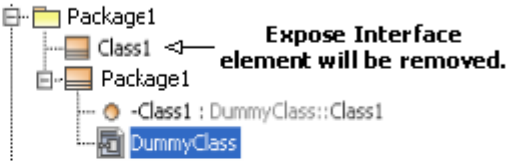
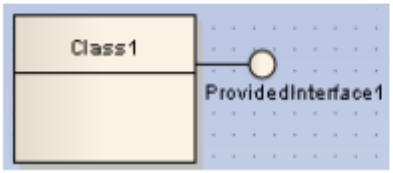
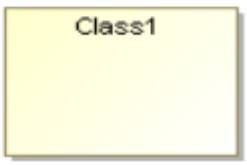


Diagram View

EA (Before Conversion)



MD (After Conversion)



Expose Interface with Element that is not Port.
The following transformation message will open:

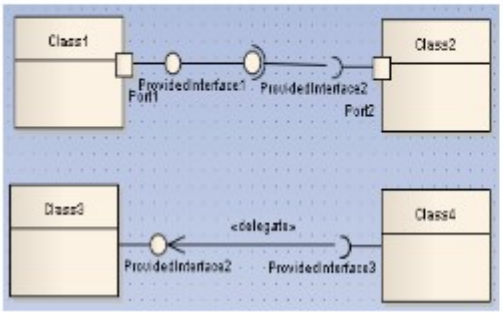
```
Removed element <xmi:id>: Invalid Element. Expose Interface can be added
to Port only.
```

Expose Interface with relationship

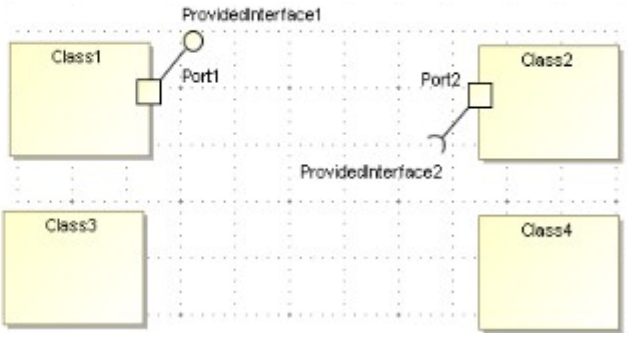
In MagicDraw, the Provided and Required Interfaces cannot be connected with any relationship, as shown in Figure 93. If there is any relationship connecting to the Expose Interface, it will be removed from both the Containment tree and diagram view.

Diagram View

EA (Before Conversion)



MD (After Conversion)



Expose Interface with Lines.
The following transformation message will open:

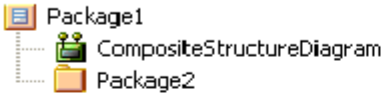
```
Removed element <xmi:id>: Invalid Element. Expose Interface cannot be
connected with the element.
```

Package

In MagicDraw, a Package cannot be drawn in a Composite Structure diagram. If it is drawn in EA, it will be removed from the diagram once it has been converted to MagicDraw.

Containment tree

EA (Before Conversion)



MD (After Conversion)

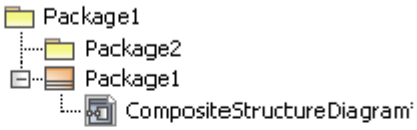
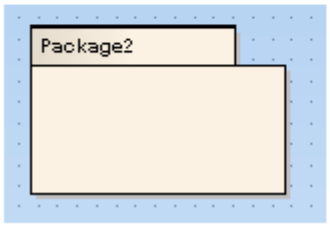
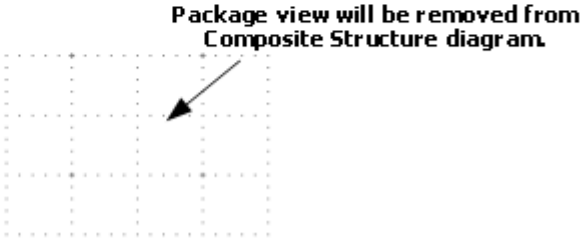


Diagram View

EA (Before Conversion)



MD (After Conversion)



Package.

Assembly

In MagicDraw, there is no Assembly line. It will be updated to a Connector.

The following transformation message will open:

```
Updated element <xmi:id>: Assembly updated to Connector.
```

Updating the Assembly Line to the Connector Line.

Dependency

There are many relationships that EA exporter exports to Dependency as the following elements:

- Delegate
- Role Binding
- Represents
- Occurrence
- Nest
- Derive
- Import
- Instantiate
- Usage
- Realize
- Trace

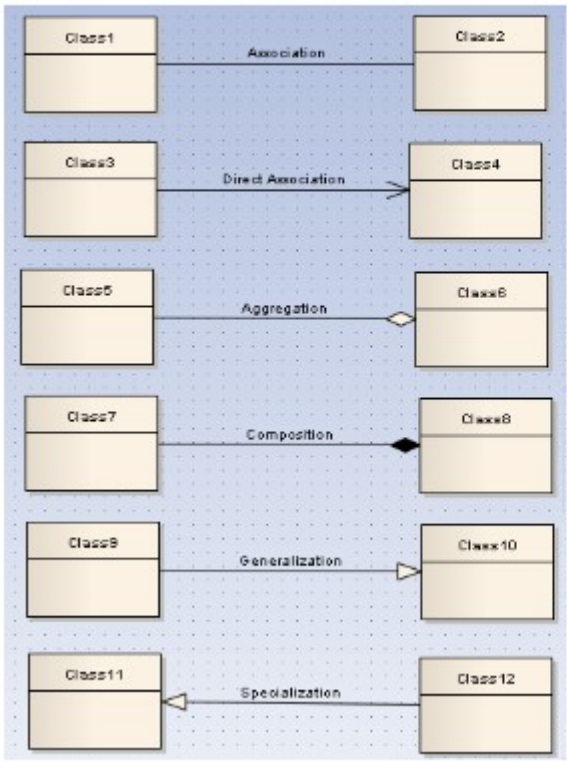
The above relationships will be shown in the diagram view as Dependencies with stereotype.

Removed relationships

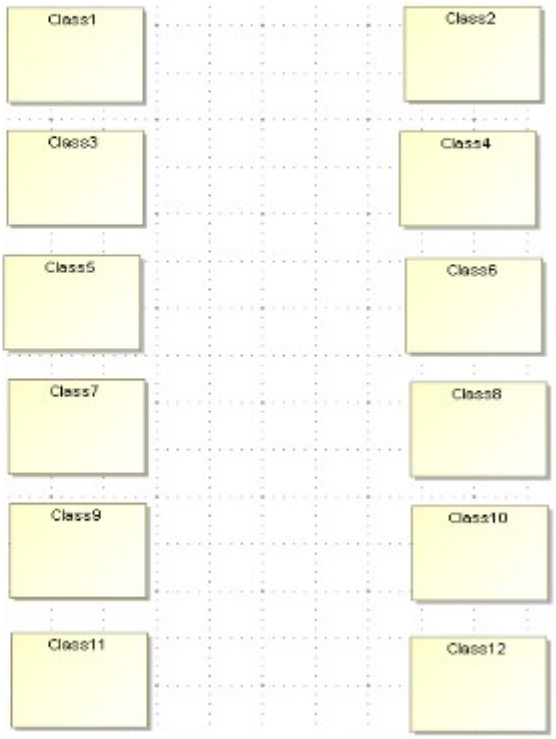
If an Association, Direct Association, Aggregation, Composition, Generalization, or Specialization is created in a Composite Structure diagram, it will be removed from the diagram view, but its data will be preserved.

Diagram View

EA (Before Conversion)



MD (After Conversion)



Removing lines.

The following transformation messages will open:

- `Removed view <xmi:id>: Association cannot be shown in Composite Structure diagram.`
- `Removed view <xmi:id>: Direct Association cannot be shown in Composite Structure diagram.`
- `Removed view <xmi:id>: Aggregation cannot be shown in Composite Structure diagram.`
- `Removed view <xmi:id>: Composition cannot be shown in Composite Structure diagram.`
- `Removed view <xmi:id>: Generalization cannot be shown in Composite Structure diagram.`

Transformation report

A report containing the conflicts solved during transformation, along with other transformation information (such as special mapping and removal of some irrelevant data), is required to be provided to the users.

The following is a list of the transformation messages:

- Removed view <xmi:id>: Invalid Port. Port can be added to Part that set type to Class, Component, and Node only.
- Removed element <xmi:id>: Invalid Element. Collaboration can contain Property, Activity, State Machine, and Interaction only.
- Removed view <xmi:id>: Collaboration cannot be contained in any element view.
- Removed view <xmi:id>: UseCase cannot be contained in any element view.
- Removed element <xmi:id>: Invalid Element. Expose Interface can be added to Port only.
- Removed element <xmi:id>: Invalid Element. Expose Interface cannot be connected with the element.
- Updated element <xmi:id>: Assembly updated to Connector.
- Updated element <xmi:id>: Provided Interface conflicts with Port type. The Port type is updated to an owner of Interface Realization.
- Removed view <xmi:id>: Association cannot be shown in CompositeStructure diagram.
- Removed view <xmi:id>: Direct Association cannot be shown in Composite Structure diagram.
- Removed view <xmi:id>: Aggregation cannot be shown in Composite Structure diagram.
- Removed view <xmi:id>: Composition cannot be shown in Composite Structure diagram.
- Removed view <xmi:id>: Generalization cannot be shown in Composite Structure diagram.

On this page

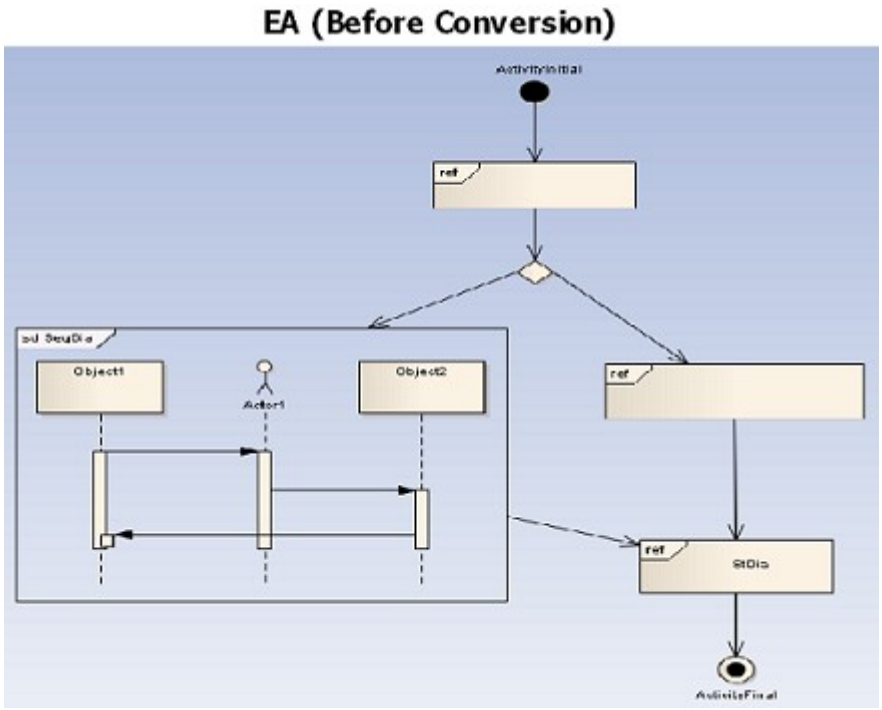
1.5.7 Interaction Overview Diagrams

The following sections describe additional EA specific Interaction Overview diagram information.

Conversion Details

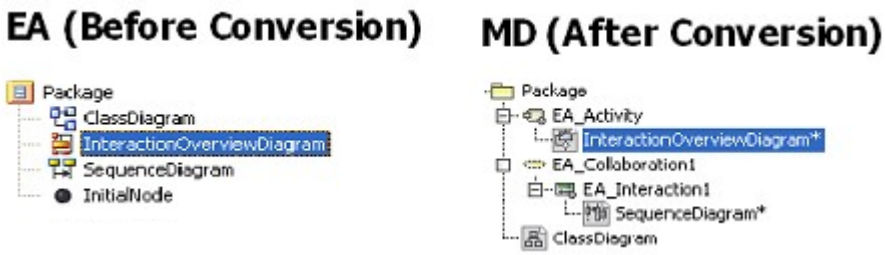
Interaction Overview Diagram

An Interaction Overview diagram is one of four types of Interaction diagrams. The other three are Timing, Sequence, and Communication diagrams. An Interaction Overview diagram, just like an Activity diagram, visualizes a sequence of activities. Most of the notation elements for Interaction Overview diagrams are the same as those for Activity diagrams (such as initial, decision, fork, join, and final nodes). Interaction Occurrences and Interaction elements are two new elements in the interaction overview diagrams.



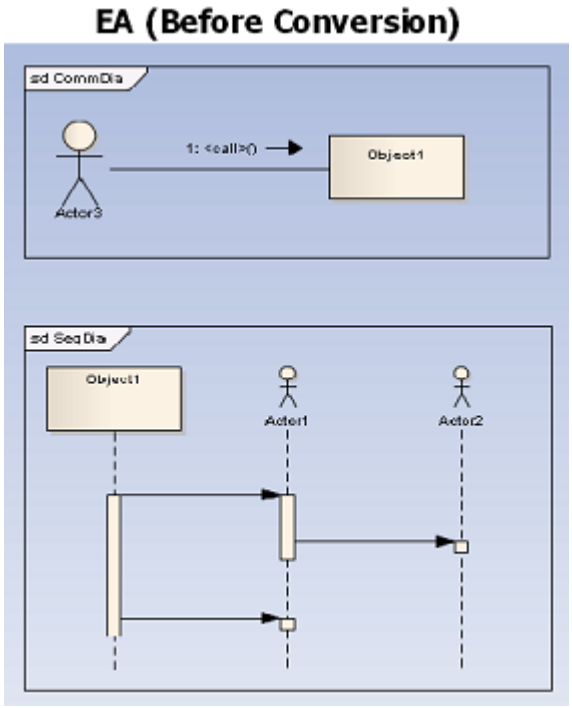
Interaction Overview Diagram.

After conversion, the Interaction Overview diagram will be placed within an Activity element.

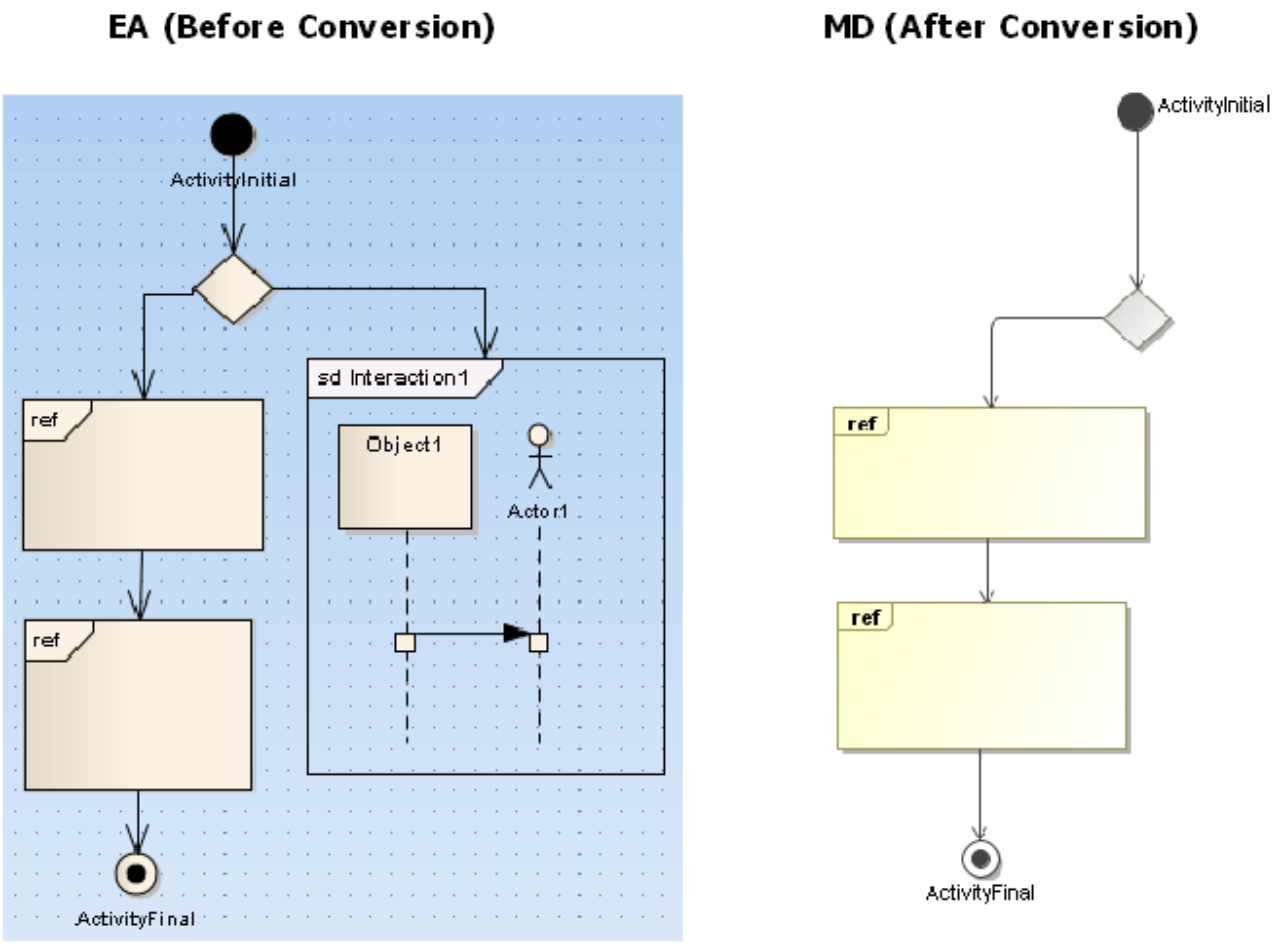


Interaction Element

Interaction elements display an inline Interaction diagram (Interaction Overview, Timing, Sequence, or Communication diagram). In EA, Interaction elements can be created to display other diagrams than those classified as Interaction diagrams.



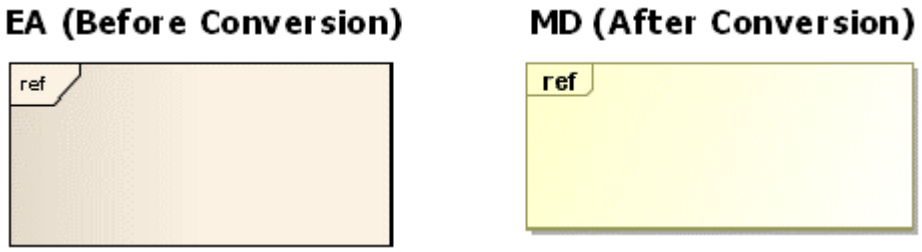
Interaction element.



Interaction Element in Interaction Overview Diagram.

Interaction Occurrence

Interaction Occurrence elements refer to an existing Interaction diagram. They are visually represented by a rectangular frame, which can be created by dragging the Interaction diagram from the project Browser to an Interaction Overview diagram and selecting the 'Interaction Occurrence' option from the pop-up dialog.



On this page

1.6 Known Limitations and Constraints

The following table shows the conversion limitations and constraints in EA exported XMI and EA Import Plugin.

No.	Constraint name	Description
1	N-ary Association	Due to the difference between the EA exported XMI and MagicDraw XMI, the view of N-ary association will not be imported.
2	Diagram Legend	MagicDraw does not have a similar element. Diagram legends will not be imported.
3	Diagram Note	The element most similar to Diagram Note is the MagicDraw diagram information table (**Option** > **Show diagram info**). However, in EA it will be mapped to the **Option** > **show diagram details**. So Diagram Notes will not be imported to MagicDraw.
4	Text size	The size of text displayed in EA and MagicDraw may vary. The text size in EA is mostly larger.
5	Word wrap	Due to size constraints in text displayed in element blocks, the result of word wrap in EA and MagicDraw is likely to be different.
6	Contact point between element and link	The position of both ends of a link element cannot be mapped to MagicDraw because the last segments of the link element in MagicDraw always point to the center of connected elements.
7	Display alternative image	In EA, you can display an element in a diagram by using an alternative image. MagicDraw will not import that particular image and will use a default shape instead.

8	Link label position	A Link element such as an association has many text labels, such as multiplicity and role name labels. MagicDraw will not import the position of these labels and will use a default position instead.
9	Pin position	A pin position in MagicDraw may have been slightly moved from its original position in EA. Especially if the pin is placed at the corner of its containing element, it will be shifted a little away from the corner (mostly in a clockwise direction).
10	Nested CallBehavior-Action	A CallBehaviorAction element nested with another CallBehaviorAction element. The outermost part of the element will remain. The other will be removed.
11	DataStoreNode inner element	A DataStoreNode element that contains Activity-Diagram-related-elements. Every element inside that particular DataStoreNode element will be removed. Only the DataStoreNode element will remain.
12	Object inner element	An Object element that contains Activity-Diagram-related-elements. Every element inside that particular Object element will be removed. Only the Object element will remain.
13	Lifeline position	The Lifeline position in MagicDraw will not correspond to the original position in EA. The position is fixed.
14	Diagonal Sequence Message	MagicDraw does not support Diagonal Sequence Messages.
15	Sequence Activation Options	MagicDraw does not support manipulating Sequence Activation through Sequence Activation Options.
16	State contains diagram	A diagram element placed inside a State element will be removed.
17	Region in Orthogonal State	EA exported XMI contains incorrect information when: • more than one region has identical names. • region was created and then removed before exporting to XMI. This will result in an unexpected result after importing it into MagicDraw.
18	Assembly Relationship in Composite Structure diagram	Assembly relationships in EA are exported to XMI as Connectors. They will then be imported to MagicDraw as Connectors.

19	Interaction elements as diagram frame	MagicDraw does not import Interaction elements displayed as diagram frame in an Interaction Overview diagram.
20	Message timing details	Duration Observation, and Timing Observation are not imported to MagicDraw.
21	Concurrent State Regions in StateMachine	Adding and removing multiple Concurrent State Regions to and from StateMachine in EA can cause the EA XMI to be incorrectly exported. If the EA exported XMI is in this state, the result of the StateMachine imported to MagicDraw cannot be determined. The same problem also occurs if there are multiple Concurrent State Regions with the same name.
22	Combined Fragment	Adding and removing multiple Interaction Operands to and from Combined Fragment in EA can cause the EA XMI to be incorrectly exported. The problem can be fixed by importing the EA XMI back to a new project in EA and exporting it back before importing it to MagicDraw.
23	Problem occurs during conversion of Sequence diagram.	Importing EA XMI to MagicDraw sometimes results in the following error message: “Problem occurred during conversion of Sequence diagram”. If this occurs, please try to import the XMI back to a new project in EA and export it back again before importing it to MagicDraw.

Index