Starting Help of StarUML

StarUML is an open source project to develop fast, flexible, extensible, featureful, and freely-available UML/MDA platform running on Win32 platform. If you use StarUML(tm), you can easily and quickly design exact software models which is based on UML standard. It will guarantee to maximize the productivity and quality because of generating numerous results automatically from it.

1. StarUML Overview
2. Basic Concepts
3. Managing Project
4. Modeling with StarUML
5. Configuring StarUML
6. Managing Modules
7. Generating Codes and Documents
8. Verifying Model
9. Printing
10. User-Interface Reference

StarUML Overview

This chapter contains a general overview of StarUML™. Included are brief overviews of StarUML™ and UML, and outlines of StarUML™ ¡s new features and overall organization.

- What is StarUML
- Key Features
- System Requirements

What is StarUML

StarUML™ is a software modeling platform that supports UML (Unified Modeling Language). It is based on UML version 1.4 and provides eleven different types of diagram, and it accepts UML 2.0 notation. It actively supports the MDA (Model Driven Architecture) approach by supporting the UML profile concept. StarUML™ excels in customizability to the user's environment and has a high extensibility in its functionality. Using StarUML™, one of the top leading software modeling tools, will guarantee to maximize the productivity and quality of your software projects.

UML Tool that Adapts to the User

StarUML™ provides maximum customization to the user's environment by offering customizing variables that can be applied in the user's software development methodology, project platform, and language.

True MDA Support

Software architecture is a critical process that can reach 10 years or more into the future. The intention of the OMG (Object Management Group) is to use MDA (Model Driven Architecture) technology to create platform independent models and allow automatic acquisition of platform dependent models or codes from platform independent models. StarUML™ truly complies with UML 1.4 standards, UML 2.0 notation and provides the UML Profile concept, allowing creation of platform independent models. Users can easily obtain their end products through simple template document.

Excellent Extensibility and Flexibility

StarUML™ provides excellent extensibility and flexibility. It provides Add-In frameworks for extending the functionality of the tool. It is designed to allow access to all functions of the model/meta-model and tool through COM Automation, and it provides extension of menu and option items. Also, users can create their own approaches and frameworks according to their methodologies. The tool can also be integrated with any external tools.

Key Features

StarUML™ has the following new features.

Feature	Description
Accurate UML standard model	StarUML™ strictly adheres to the UML standard specification specified by the OMG for software modeling. Considering the fact that the results of design information can reach 10 years or more into the future, dependence on vendor-specific irregular UML syntax and semantics can be quite risky. StarUML™ maximizes itself to order UML 1.4 standard and meaning, and it accepts UML 2.0 notation on the basis of robust meta model**.**
Open software model format	Unlike many existing products that manage their own legacy format models inefficiently, StarUML™ manages all files in the standard XML format. Codes written in easy-to-read structures and their formats can be changed conveniently by using the XML parser. Given the fact that XML is a world standard, this is certainly a great advantage, ensuring that the software models remain useful for more than a decade.
True MDA support	StarUML™ truly supports UML Profile. This maximizes extensibility of UML, making modeling of applications possible even in areas like finance, defense, e-business, insurance, and aeronautics. Truly Platform Independent Models (PIM) can be created, and Platform Specific Model (PSM) and executable codes can be automatically generated in any way.
Applicability of methodologies and platforms	StarUML™ manipulates the approach concept, creating environments that adapt to any methodologies/processes. Not only the application framework models for platforms like .NET and J2EE, but also basic structures of software models (e.g. 4+1 view-model, etc.) can be defined easily

Excellent extensibility	All functions of the StarUML™ tools are automated according to Microsoft COM. Any language which supports COM (Visual Basic Script, Java Script, VB, Delphi, C++, C#, VB.NET, Python, etc.) can be used to control StarUML™ or develop integrated Add-In elements.
Software model verification function	Users can make many mistakes during software modeling. Such mistakes can be very costly if left uncorrected until the final coding stage. In order to prevent this problem, StarUML™ automatically verifies the software model developed by the user, facilitating early discovery of errors, and allowing more faultless and complete software development.
Useful Add-Ins	StarUML™ includes many useful Add-Ins with various functionalities: it generates source codes in programming languages and converts source codes into models, imports Rational Rose files, exchanges model ing information with other tools using XMI, and supports design patterns. These Add-Ins offer additional reusability, productivity, flexibility and interoperability for the modeling information.

System Requirements

The following are the minimum system requirements for running StarUML(tm).

- Intel(R) Pentium(R) 233MHz or higher
- Windows(R) 2000, Windows XP™, or higher
- Microsoft(R) Internet Explorer 5.0 or higher
- 128 MB RAM (256MB recommended)
- 110 MB hard disc space (150MB space recommended)
- CD-ROM drive
- SVGA or higher resolution monitor (1024x768 recommended)
- Mouse or other pointing device

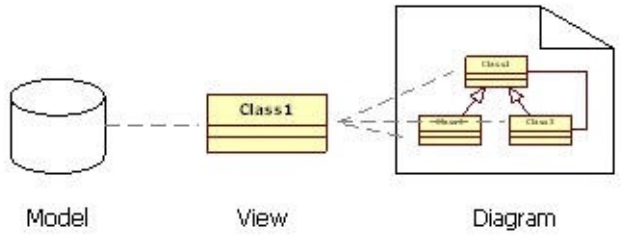
Basic Concepts

This chapter introduces the fundamental concepts required for effective use of StarUML™. Included are descriptions of models, views and diagrams, projects, units, approaches, frameworks, model fragments and their differences as well as UML profile.

- Model, View and Diagram
- Project and Unit
- Module

Model, View and Diagram

StarUML™ makes a clear conceptual distinction between models, views and diagrams. A **Model** is an element that contains information for a software model. A **View** is a visual expression of the information contained in a model, and a **Diagram** is a collection of view elements that represent the user's specific design thoughts.



Project and Unit

Project

A project is the basic management unit in StarUML™. A project can manage one or more software models. It is the top-level package that always exists in any software model. In general, one project is saved in one file.

Project Structure

A project contains and manages the following sub-elements.

Project Sub-Element	Description
Model	Element that manages one software model.
Subsystem	Element that manages models that express one subsystem.
Package	The most general element for managing elements.

Project File

Project files are saved in the XML format with the extension name ".UML". All models, views and diagrams created in StarUML™ are saved in one project file. A project may also be divided up and saved in multiple units. A project file contains the following information.

- UML profiles used in the project
- Unit files referenced by the project
- Information for all models contained in the project
- Information for all diagrams and views contained in the project

Units

While a project is generally saved in one file, there are cases where one project needs to be saved in many smaller files so that a number of developers can work

on the project together. In this case, the project can be managed as multiple units. A unit can have a hierarchical structure; it may contain many sub-units under it. Units are saved as .UML files and are referenced by project files (.UML) or other unit files (.UNT).

Unit Composition

Only package, subsystem and model elements can constitute one unit. All elements under these package type elements are saved in the respective unit file (.UNT).

Unit Hierarchical Structure

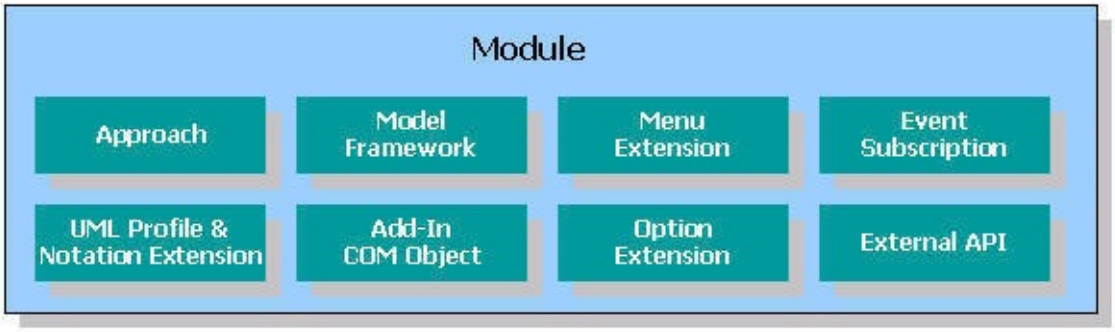
Just as a project can manage many units under it, a unit also can manage many sub-units. Since a parent unit has reference to its child units, all units have a hierarchical structure.

Model Fragments

A model fragment is a part of a project saved as a separate file. Only model, subsystem or package items can constitute model fragments. Model fragment files are saved with the extension name “.MFG”. Model fragment files can be easily included in any project at any time. Model fragments are essentially different from units in that once included in a project, they merge completely with the rest of the project.

Module

The module is a package to provide new functions and features as extending StarUML™. The module can be created as combinations of several extension elements. Also, you can not only configure only extension element to an independent module for purpose, but also create same typed extension elements in a module.



Module of StarUML™ provide the following functions.

- Expansion of the main menu or popup menu.
- Addition of new approach
- Addition of new profile
- Addition of new profile
- Addition of new element through stereotype or expansion of notation
- Implementation of new function (through COM Server or simple script file)
- Integration with other applications
- Other Add-In functions

Approaches

There are countless methodologies for software development, and each company or organization has its own, or uses an existing one that is modified to

meet the requirements of its development team or projects. Application domains, programming languages, and platforms are also different for each piece of software developed. Consequently, many items have to be configured in the initial phase of software modeling. StarUML™ provides the concept of approaches to facilitate easier configuration of such items.

Approach Structure

An approach consists of the following items.

Approach Component	Description
Project Structure	Specifies the basic structure of the project. The basic structure can be designed with package, subsystem and model elements. The diagram can also be given a default layout.
Import Profiles	Automatically includes the default UML profiles in the project.
Import Frameworks	Automatically loads and includes the default frameworks in the project.
Import Model fragments	Automatically loads and i nclude the default model fragments in the project.

Frameworks

Frameworks in StarUML™ refer to software models that express class libraries or application frameworks like MFC, VCL, and JFC. Including and using frameworks in projects makes it much easier for the user to model software that depends on specific class libraries or application frameworks.

Framework Structure

A framework consists of one framework file (.FRW) and one or more unit files (.UNT).

Component	Description
Framework File(.FRW)	Framework files contain information for the units included and the UML profiles used.
	Unit files contain actual model information for the

Unit File(.UNT)	framework.

UML Profile

UML (Unified Modeling Language) is so general that it can be used to express any thoughts or concepts. This can also be the source of its weakness, as concepts of specific domains cannot be expressed in fine detail. To overcome such weakness, StarUML™ provides UML profiles that expand UML. StarUML™ supports easy expansion of UML by directly accommodating the concepts in UML profiles.

UML Profile Structure

A UML profile consists of the following components.

Component	Description
Stereotype	The Stereotypes are attached to specific UML elements to further clarify their semantics and provide extension attributes, making more accurate modeling possible.The stereotype specifies not only icon file to express graphic notation but also defines notation schema method as using extension notation defined file(.PNX).For more detail about extension notation, refer to developer’s guide.
TagDefinition	When the default UML element properties are inadequate for accurate modeling, tag definition provides additional information for the elements. In StarUML™, tag definitions can either be included in specific stereotypes or exist independently.
DataType	The datatype that is contained in the profile by default.
DiagramType	The DiagramType is extension element suggesting by StarUML™ so that user can define new diagram.
ElementPrototype	The element prototype is extension element suggesting by StarUML™ so that user can define a sample for creating element as configuring attributes in the present defined element. These defined element prototypes can create

	elements as linking to palette or create elements through external API.
ModelPrototype	The model prototype is an extension element which is suggested by StarUML™ so that is similar to element prototype, but it's only applied for the model. The defined element as model prototype is expressed on model addition menu.
Palette	The Palette is extension element suggesting by StarUML™ so that user can addition palette.

For detailed descriptions on writing profiles, see the StarUML™ Developer guide.

Application of UML Profile

UML profiles can be used for the following purposes. The OMG (Object Management Group) also specifies UML profile standards for specific purposes.

- Profiles for specific programming languages (C/C++, Java, C#, Python, etc.)
- Profiles for specific development methodologies (RUP, Catalysis, UML Components, etc.)
- Profiles for specific domains (EAI, CRM, SCM, ERP, etc.)

Addition of Module

If you install modules which developing by users or distributing by third party vendors, you can use extension functions in StarUML™. In order to install new additional modules in a system, complicated authentication is not needed. If you want to install modules, copy files which consist of modules after making sub directory under <install-dir>\modules\.

Addition of Module in StarUML™

StarUML™ contains server modules on the platform.

StarUML™ basically provides UML standard profile, a few of approaches

- and standard module to provide transformation between sequence & collaboration diagram.
- Provides Generator module to generation for document and code.
- Provides Java module to support Java profile, J2SE/J2EE Framework, code generation, reverse engineering.
- Provides C++ module to support C++ profile, MFC Framework, code generation, reverse engineering.
- Provides C# module to support C# profile, .NET BCL framework, code generation, reverse engineering.
- Provides XMI module to support XMI import & export for model exchange.
- Provides Rose module to read Rational Rose File.
- Provides Pattern module to support design pattern.

Managing Project

This chapter describes in detail the procedures for project management: creating a new project, making a part of the project into a unit, creating and importing model fragments, importing frameworks, and including and excluding UML profiles.

- Managing a Project
- Managing Units
- Working with Model Fragments
- Importing a Framewordk
- Working with UML Profiles

Managing a Project

This chapter describes in detail the procedures for project management.

- Creating New Project
- Opening Project
- Saving Project
- Closing Project
- Element Management by Models, Subsystems and Packages

Creating New Project

In order to work on a new software development, a new project must be created. You may start with a completely empty project or with a new project that has been initialized according to a specific approach.

Procedure for Creating New Project #1 – New Project:

1. Select the **[File] -> [New Project]** menu.

   A new project is created with the default approach selected by the user.
2. Depending on the approach, profiles and/or frameworks may be included/loaded

Procedure for Creating New Project #2 – Select New Project Dialog Box:

1. Select the **[File] -> [Select New Project…]** menu.

2. A list of the available approaches will be displayed in the Select New Project dialog box. Select one from the list and click the **[OK]** button.

   A new project is created and initialized according to the selected approach.
3. Depending on the approach, profiles and/or frameworks may be

included/loaded.

Note

- The list of the available approaches may differ depending on the user’s installation environment.
- To change the default approach, open the Select New Project dialog box, select an approach, and then check the option “**Set As Default Approach**”

Opening Project

In order to work on a saved project, the project file must be opened. If the project includes more than one unit, all the related units will also be loaded with the project.

Procedure for Opening Project:

1. Select the **[File]** -> **[Open…]** menu.
2. At the Open Project dialog box, select a project file (.UML) and click the **[Open]** button.

3. The selected project file will be opened.

Note

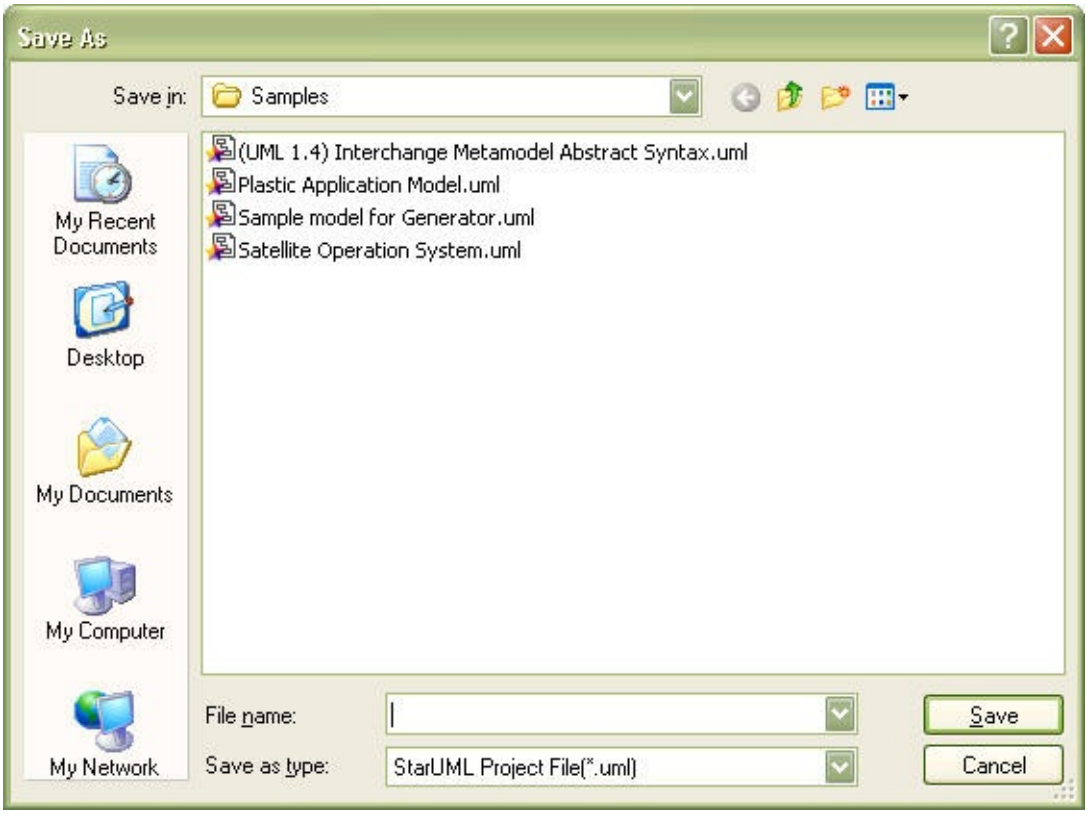
- Projects can also be opened through the Select New Project dialog box.

Saving Project

In order to preserve any changes made to a project, the project file must be saved properly. Your work can be saved over the existing project file or saved as a new project file. When a project file is saved, information on the related units is saved together with it.

Procedure for Saving Project:

1. Select the **[File]** -> **[Save]** menu.
2. If the project file name has not been specified, the Save Project dialog box appears. Enter the file name and click the **[Save]** button.



3. The project file is saved.

Procedure for Saving Project as Another File:

1. Select the **[File]** -> **[Save As…]** menu.

2. At the Save As dialog box, enter the new file name and click the **[Save]** button.

3. The project is saved as another file.

Note

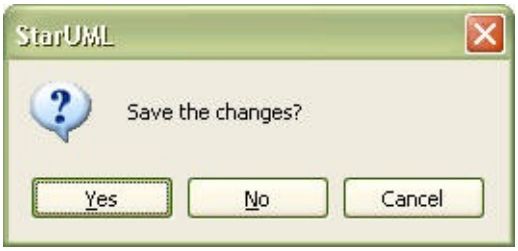
- If the project contains one or more units and the units have been changed, a dialog box will appear asking whether you want to save the changed units. Select **[Yes]** to save all changed units with the project.

Closing Project

The project can be closed if it no longer requires editing.

Procedure for Closing Project:

1. Select the **[File]** -> **[Close]** menu.
2. If the project has not been saved after changes were made, the user will be prompted to save the changes. The user can select yes, no, or cancel.



3. The project is closed and becomes no longer available for editing.

Element Management by Models, Subsystems and Packages

A software model consists of many elements and diagrams. Grouping these elements and diagrams for efficient management is very important. StarUML™ supports three types of grouping elements (models, subsystems and packages), which the user can use appropriately according to each purpose.

Grouping Elements Provided in StarUML™

Grouping Element	Description
Model	Model expresses the physical system for specific purposes (aspects). For example, it can express a specific aspect of the system (e.g. analysis aspect, design aspect, user aspect, etc.).
Subsystem	Subsystem groups the elements that specify the entire physical system or parts of it.
Package	Package logically groups and manages model elements. It is an extremely generalized element that can be used in any way for organizing elements.

Managing Units

Although a project can be managed as one file, it may be convenient to divide it into many units and manage them separately if many developers are working on it together. This section describes procedures for creating and managing units.

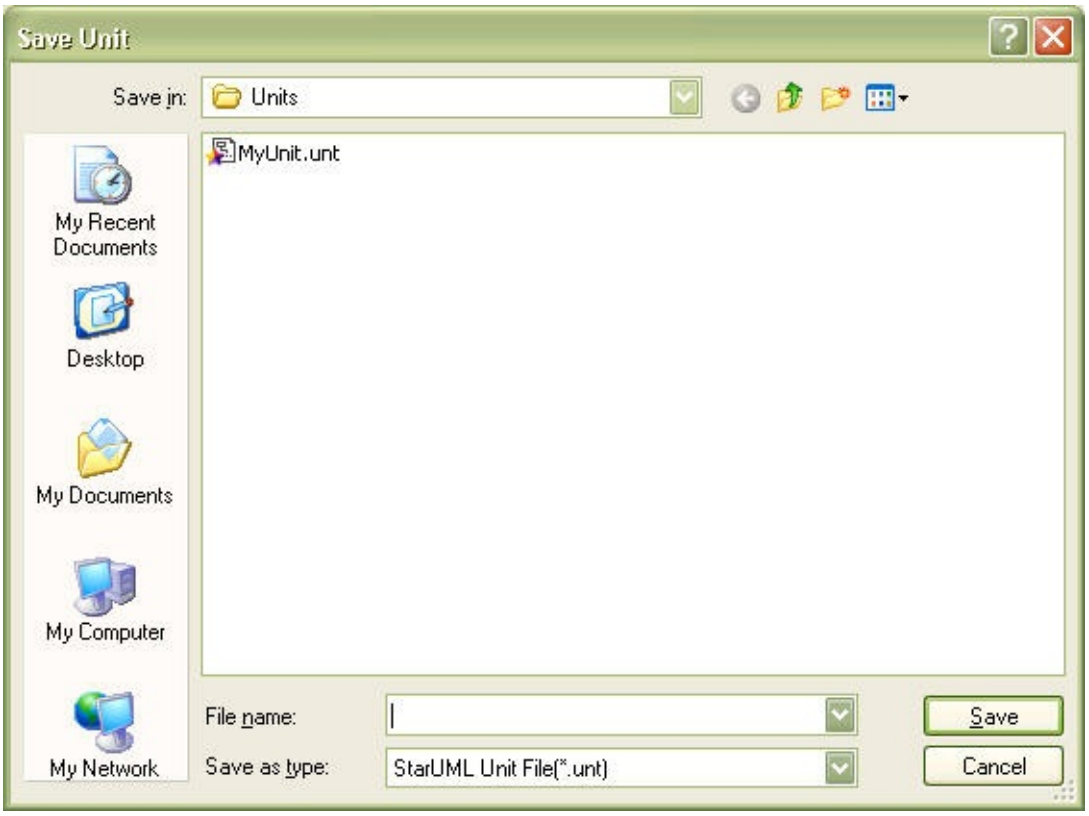
- Creating Unit
- Merging Unit
- Saving Unit
- Removing Unit

Creating Unit

It may be necessary to save a part of a project or unit as a separate unit. For instance, when many developers are working on the project together, the project can be divided into many units and managed by tools like Microsoft Visual SourceSafe or CVS. Only Package, Model and Subsystem elements can be saved as units.

Procedure for Creating New Unit:

1. Select an element (package, model or subsystem) to make into a unit.
2. Right-click and select the **[Unit] -> [Separate Unit]** menu.
3. At the Save dialog box, enter the unit file name and click the **[Save]** button.



4. The selected element is saved as a unit.

Merging Unit

If the elements in a unit no longer need to be managed as a separate unit, the unit file can be merged with the project.

Procedure for Merging Unit:

1. Select from the model explorer an element (project, model, package or subsystem) that will contain the unit to import.
2. Right-click and select the **[Unit]** -> **[Uncontrol Unit…]** menu.
3. The unit is merged with the selected project or parent unit.

Note

- Merging a unit does not automatically delete the unit file (.UNT). Please delete it manually if no longer required.

Saving Unit

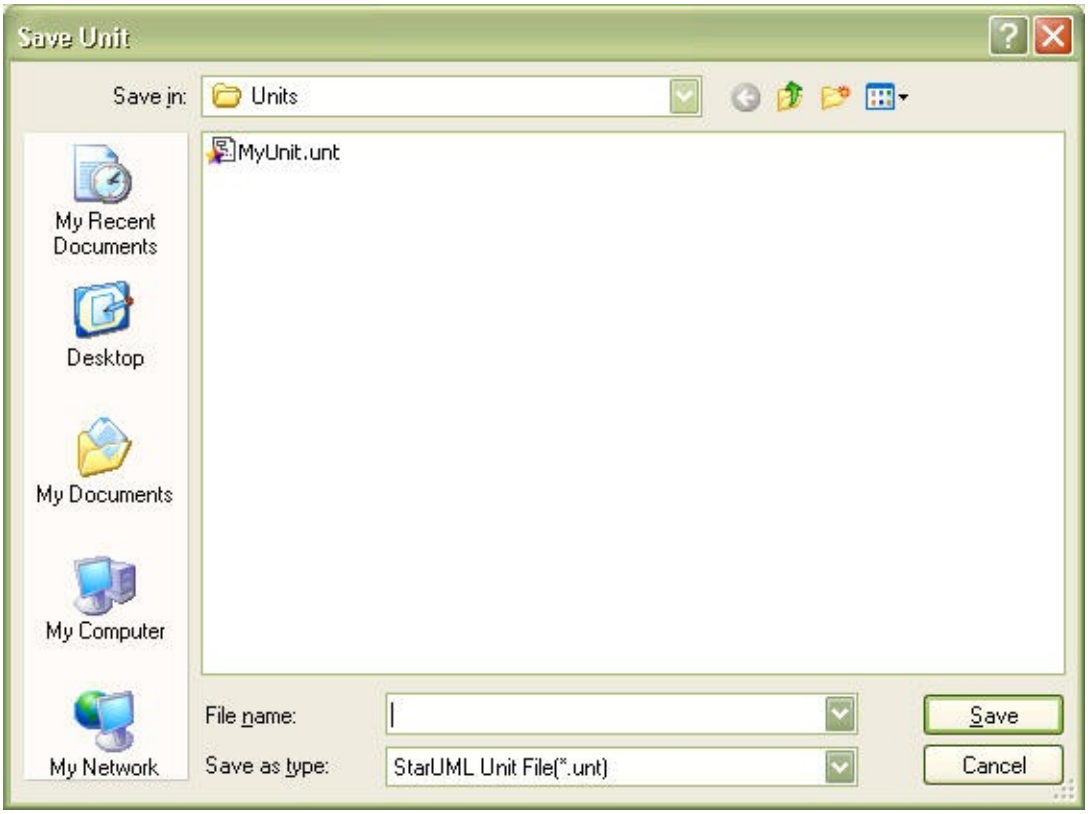
If changes are made to a unit, they needs to be saved properly. The changes can be saved over the existing unit file or saved as another unit file.

Procedure for Saving Unit:

1. Select the unit to save from the model explorer.
2. Right-click and select the **[Unit] -> [Save Unit]** menu.
3. The unit file is saved.

Procedure for Saving Unit as Another File:

1. Select the unit to save from the model explorer.
2. Right-click and select the **[Unit] -> [Save Unit As…]** menu.
3. At the Save Unit As dialog box, enter the new unit file name and click the **[Save]** button.



4. The new unit file is saved.

Note

- Saving a unit as another file does not delete the original unit file. Please delete it manually if no longer required.

Removing Unit

If a unit is no longer required in a project, the unit can be removed. Removing a unit deletes all the elements contained in it and the unit is no longer loaded in the project automatically. Please take note that you should use “Merge Unit” instead of “Remove Unit” if you intend to merge a unit with a project and no longer manage it as a separate unit.

Procedure for Removing Unit:

1. To remove a unit, select from the model explorer the element (package, model or subsystem) that contains the unit.
2. Right-click and select the **[Unit] -> [Delete Unit]** menu.
3. A dialog box appears confirming whether you want to remove the unit. Click **[Yes]**.

4. The unit is completely removed from the project.

Note

- Selecting the element that contains a unit and selecting the **[Edit] -> [Delete From Model]** menu has the same effect.
- You need to decide whether to completely remove the unit from the project or merge the unit with the project.
- Removing a unit does not delete the unit file (.UNT). Please delete it manually if no longer required.

Working with Model Fragments

Model fragments can be used for saving parts of a project.

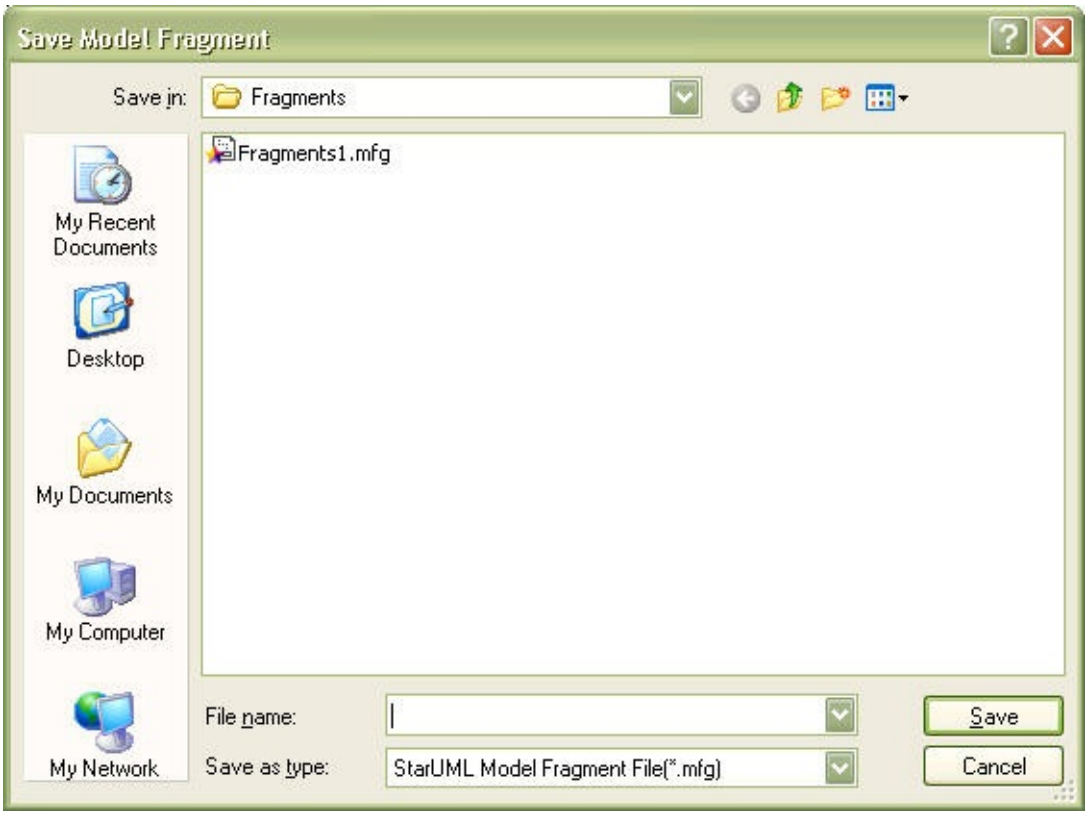
- Creating Model Fragment
- Importing Model Fragment

Creating Model Fragment

Parts of a project can be saved as separate model fragment files for access by other users or future reuse. Unlike units, model fragments are not referenced by other files and do not reference other files. They are independent entities. Model fragments can be included in a project at any time.

Procedure for Creating Model Fragment:

1. Select from the model explorer a package, subsystem or model to make a model fragment.
2. Select the **[File] -> [Export] -> [Model Fragment…]** menu.
3. At the Save Model Fragment dialog box, enter the model fragment file name and click the **[Save]** button.

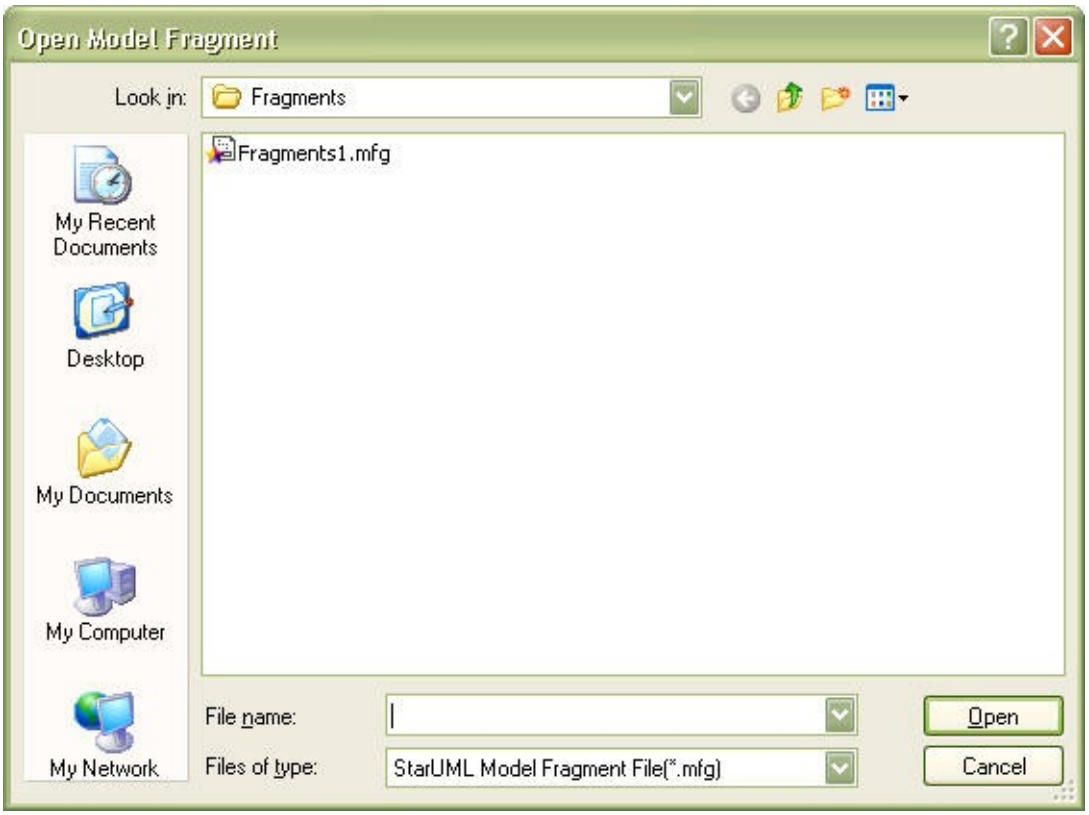


Importing Model Frament

Elements saved in a model fragment file (.MFG) can be imported into a project. Importing a model fragment copies and includes the elements contained in the model fragment into the project. No references are used.

Procedure for Importing Model Fragment:

1. Select the **[File] -> [Import] -> [Model Fragment…]** menu.
2. At the Open Model Fragment dialog box, select a model fragment file (.MFG) to read and click the **[Open]** button.



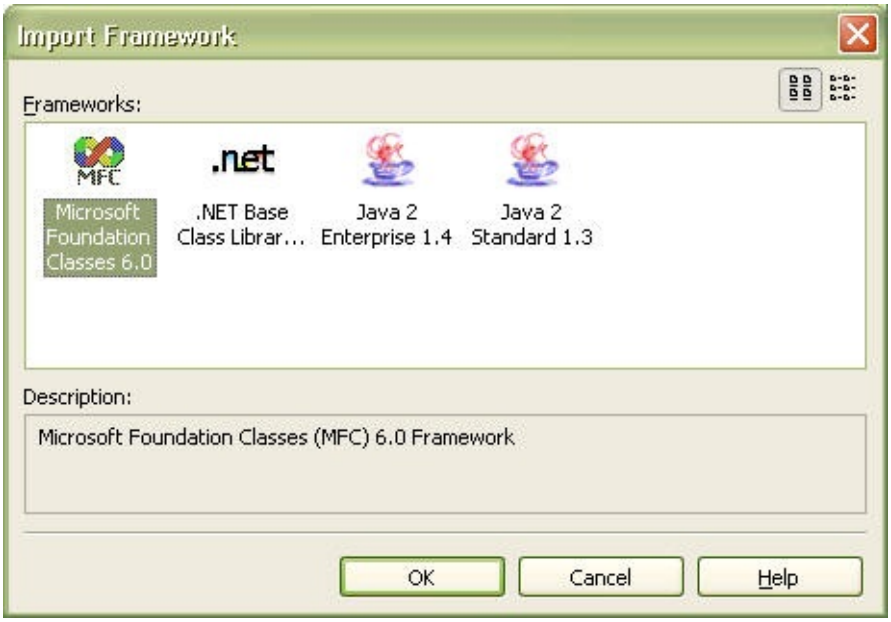
3. The Select Element dialog box appears, to determine which element will contain the model fragment to import. Select an element (package, model, subsystem, or project) to contain the model fragment and click the **[OK]** button.
4. The model fragment is added to the selected element.

Importing a Framewordk

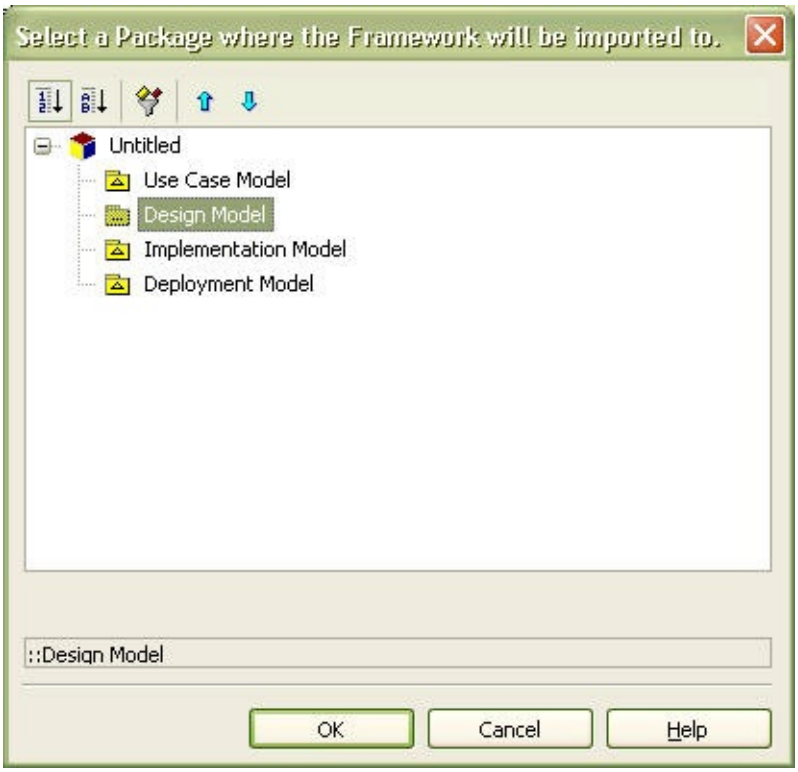
In order to use a framework in a project, the framework must be loaded. Once a framework is loaded, all the elements contained in the framework can be used. Note that the units in frameworks are usually read-only files and the framework elements cannot be modified directly.

Procedure for Importing Framework:

1. Select the **[File] -> [Import] -> [Framework…]** menu.
2. At the Import Framework dialog box, select a framework to import and click the **[OK]** button.



3. The Select Element dialog box appears, to determine which element will contain the framework to import. Select an element (package, model, subsystem, or project) to contain the framework and click the **[OK]** button.



4. The framework is added to the selected element.

Note

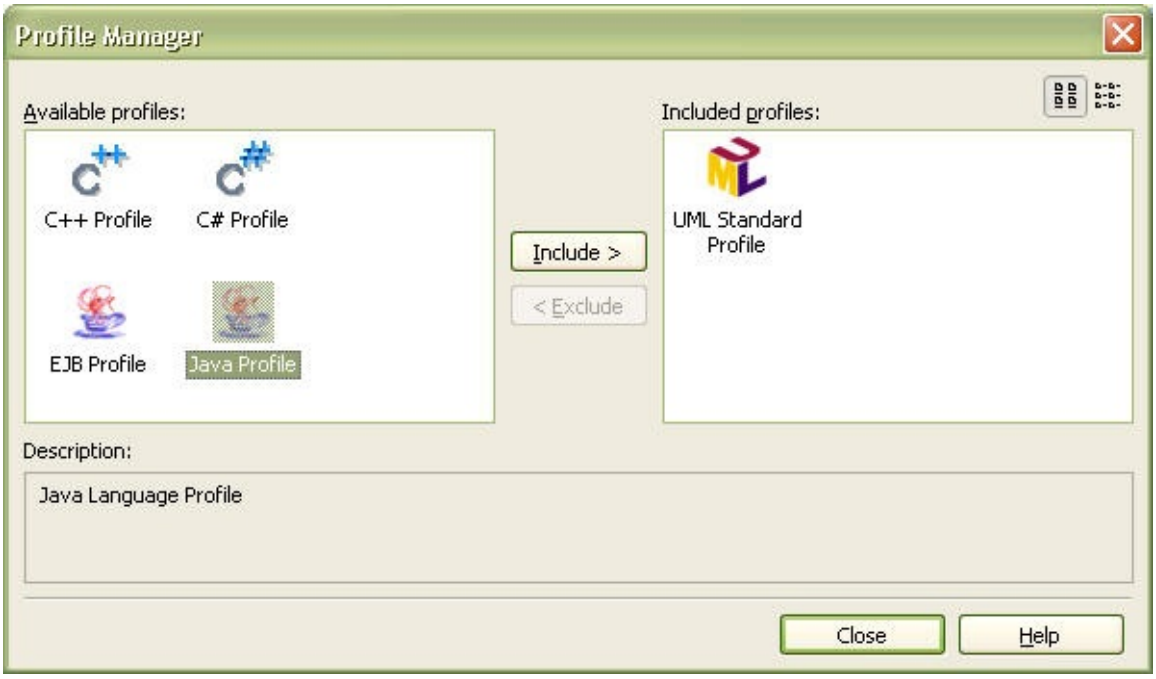
- Importing a framework does not save the framework elements in the project. The framework units are referenced in the project, and they must always be present whenever the project is opened.
- In order to delete an imported framework, you have to delete all the related units manually.

Including UML Profile

Predefined UML profiles can be included for use with the current project. Once a UML profile is included in a project, the stereotypes, tag definitions and data types defined in the profile can be used in the project.

Procedure for Including UML Profile:

1. Select the **[Model] -> [Profiles…]** menu.
2. At the Profile Manager window, select a profile from the available profile list on the left, click the **[Include]** button and then click the **[Close]** button.



3. The selected profile is included in the current project.

Note

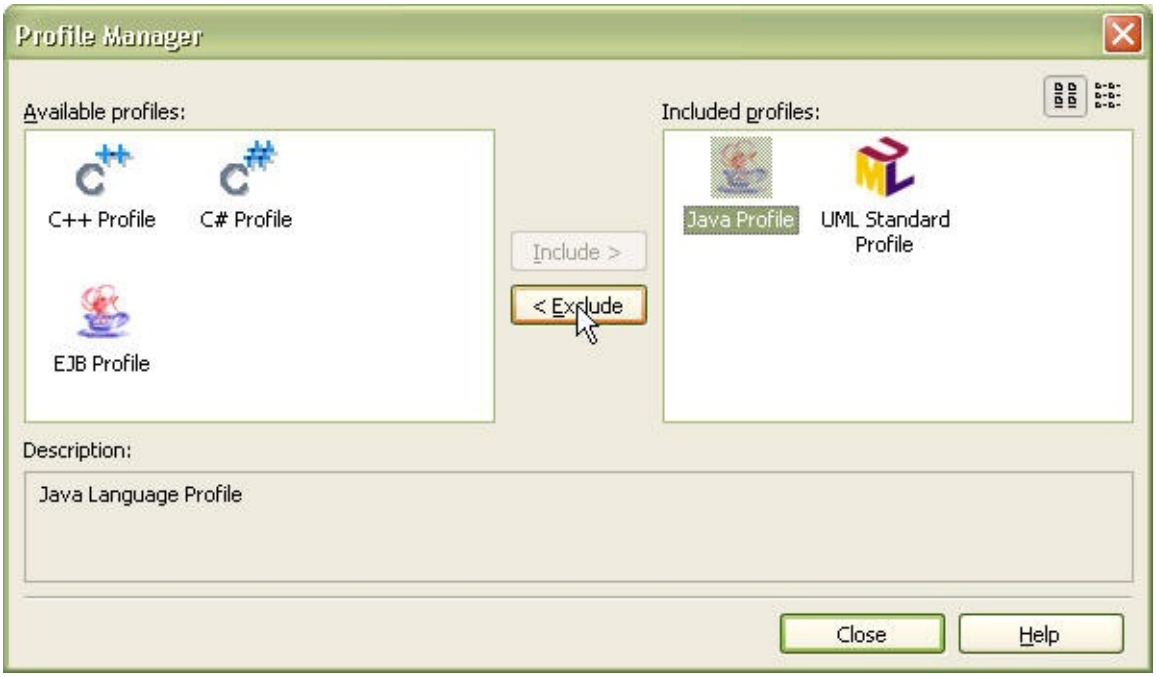
- The profile list in the Profile Manager may vary according to the user's installation environment.

Excluding UML Profile

The UML profiles included in the current project can be excluded. Once a UML profile is excluded from a project, the stereotypes, tag definitions and data types defined in the profile cannot be used in the project.

Procedure for Excluding UML Profile:

1. Select the **[Model]** -> **[Profiles…]** menu.
2. At the Profile Manager window, select a profile from the included profile list on the right, click the **[Exclude]** button and then click the **[Close]** button.



3. The selected profile is excluded from the current project.

Note

- Excluding a profile while its stereotypes and tag definitions are in use may result in loss of information for the related elements. Please exercise caution when excluding profiles.
- The profile list in the Profile Manager may vary according to the user's installation environment.

Modeling with StarUML

This chapter describes in detail the procedures for creating diagram element and editing. Included are to organize model structure using model explorer.

- Editing Elements and Diagrams
- Organizing Model Structure
- Modeling with UseCase Diagram
- Modeling with Class Diagram
- Modeling with Sequence Diagram
- Modeling with Collaboration Diagram
- Modeling with Statechart Diagram
- Modeling with Activity Diagram
- Modeling with Component Diagram

Editing Elements and Diagrams

This chapter outlines the procedures for creating diagrams. Included are procedures for creating and deleting diagram and elements, editing model and view elements, and description of the additional functions provided for effective management of diagrams.

- Creating New Project
- Creating Element in Diagram
- Creating View Element in Diagram
- Activates Diagram
- Editing Element in Diagram
- Copy and Paste
- Resize and Move
- Creating Element by using ShortCut Generation Syntax
- Editing Model Element
- Editing View Element
- Diagram Management
- Finding Element
- Aligning Element
- Layout Diagram
- Configuring Zoom-in/Zoom-Out
- Saving Diagram as Image File
- Copying Diagram as bitmap
- Navigating Diagram

- Configuring Default Diagram

Creating New Diagram

StarUML™ supports 11 UML diagram types. The user can freely create and manage different diagrams as needed.

Procedure for Creating New Diagram:

1. Select from the model explorer or diagram area an element to contain the new diagram.
2. Right-click and select the **[Add Diagram]** menu. A new diagram will be created when selection is made for the diagram type.

Types of Diagrams Available

Diagram Type	Description
 Class Diagram	Class Diagram is a visual expression of various static relations of class-related elements. Class Diagram can contain not only classes but also interfaces, enumerations, packages, various relations, instances, and their links.
 Use Case Diagram	Use Case Diagram is an expression of relations between the use cases in a specific system or object and the external actors. Use Case expresses the functions of the system and how the system functions interact with the external actors.
 Sequence Diagram	Sequence Diagram expresses the interactions of instances. It is a direct expression of the InteractionInstanceSet, which is a set of the stimuli exchanged between the instances within a CollaborationInstanceSet. While Sequence Role Diagram is a ClassifierRole-oriented expression, Sequence Diagram is an Instance-oriented expression.
	Sequence Role Diagram expresses the interactions of the role concepts. It is a direct expression of the Interaction, which is a set of the messages

 Sequence Diagram (Role)	exchanged between the ClassifierRoles within a Collaboration. While Sequence Diagram is an Instance-oriented expression, Sequence Role Diagram is a ClassifierRole-oriented expression.
 Collaboration Diagram	Collaboration Diagram expresses the collaboration between instances. It is a direct expression of the collaboration model of the instances within a CollaborationInstanceSet. While Collaboration Role Diagram is a ClassifierRole-oriented expression, Collaboration Diagram is an Instance-oriented expression.
 Collaboration Diagram (Role)	Collaboration Role Diagram expresses the collaboration between the role concepts. It is a direct expression of the collaboration model of the ClassifierRoles within a Collaboration. While Collaboration Diagram is an Instance-oriented expression, Collaboration Role Diagram is a ClassifierRole-oriented expression.
 Statechart Diagram	Statechart Diagram expresses the static behaviors of a specific object through states and their transitions. Although Statechart Diagram is generally used to express the behaviors for instances of classes, it can also be used to express behaviors of other elements.
 Activity Diagram	Activity Diagram is a special form of Statechart Diagram that is suitable for expressing the activity execution flow. Activity Diagram is commonly used for expressing workflow, and it is frequently used for objects like classes, packages, and operations.
 Component Diagram	Component Diagram expresses the dependency between the software components. The elements that constitute software components and the elements that implement those components can all be expressed by Component Diagram.
 Deployment Diagram	Deployment Diagram expresses the hardware elements of the physical computer and devices and the software components, processes and objects that

	are assigned to them.
 Composite Structure Diagram	Composite Structure Diagram is a diagram to express internal structure of Classifier. It is included in interaction point with other parts of system.

Note

- The types of diagrams available vary from one element type to another.

Creating Element in Diagram

In order to create a new element in a diagram, a diagram must be opened first. The pallet contains the different types of elements available for creation depending on the diagram type. The list of available elements varies from one diagram type to another.

Procedure for Creating Element from Pallet:

1. Select an element type to create from the pallet.
2. Click a location in the diagram area to create the element. (Drag the mouse to select an area to specify the size of the new element. If creating an element that connects two elements together, ensure that the connection is made accurately.)

Procedure for Creating Multiple Elements in One Go:

1. Select an element type to create from the pallet.
2. Click the **[Lock]** item in the pallet or click the element to create once again.
3. Create multiple elements.
4. Click the 

item in the pallet when creating elements is complete.

Note

- Creating an element in the diagram from the pallet actually involves creating a model element and its view element.

Creating View Element in Diagram

Besides creating a new element in the diagram from the pallet, view elements can also be created for existing model elements.

Procedure for Creating New View Element (Drag-and-Drop Method):

1. Select from the model explorer a model to be represented by the new view element.
2. Drag the model element and drop it in the diagram area to create a view element (In this case, the connections to all the related elements are automatically displayed).

Note

- This drag-and-drop method may not work when creating view elements for certain model element types and diagram types.
- Model elements can also be created for not existing view elements. For detailed descriptions on creating model element, see the creating model element.

Activates Diagram

In order to edit the specific diagram, you have to activate the diagram when you open several diagrams. If you want to activate the opened diagram, click the diagram on tab. In the case of having a lot of opened diagrams, you can activate the diagram as you selecting it in diagram list on pop-up menu.

Procedure for the diagram activity with selected in menu:

1. Right-click on the diagram tab and select the **[Pages]** menu.
2. Selects a diagram name to activate among diagram lists as submenu.

Editing Element in Diagram

Elements can directly be edited in the diagram area.

Procedure for Editing Elements:

1. Double-click a view element to click in the diagram.
2. At the quick dialog, edit the element name, visibility, etc., or click the button to create elements under the selected element.
3. Hit **[Enter]** or click another location in the diagram to apply the changes.

Note

- For detailed descriptions on element to Quick dialogs, see the Quick dialogs.

Copy and Paste

When copying or cutting elements for pasting, a clear distinction has to be made between model elements and view elements. If a model element is copied, it has to be pasted under a model element. In this case, all the sub-elements contained in the selected element are copied together. View elements can be copied within the same diagram or to different diagrams. Copied view elements can be pasted in diagrams only; they cannot be pasted to model elements. Copying and pasting may also be restricted depending on the view element types and diagram types.

Procedure for Copying and Pasting Model Elements:

1. Select a model element to copy from the model explorer.
2. Right-click and select the **[Copy]** menu. The model element is copied to the clipboard.
3. Select from the model explorer a model element where the copied element will be pasted.
4. Right-click and select the **[Paste]** menu. The copied model element will be recalled from the clipboard and pasted under the selected element.

Copied model elements can be pasted only to the elements that can contain them.

Procedure for Copying and Pasting View Elements in Diagram:

1. Select from the diagram area the view elements to copy. (You may select multiple elements by dragging the mouse over an area. Click the view elements while holding down the **[Shift]** key to add the elements to the selection.)
2. Right-click and select the **[Copy]** menu. The view elements are copied to the clipboard.
3. Open the diagram where the copied view elements will be pasted. (Double-click a view element from the model explorer or the diagram explorer, or select a view element from the diagram tab.)

4. Right-click and select the **[Paste]** menu. The copied view elements will be pasted to the active diagram.

Copy/Paste for Different Diagram Types

Diagram Type	Copy/Paste
Class Diagram	Elements can be copied or pasted freely between Class, UseCase, Component, CompositeStructure, and Deployment diagrams.
UseCase Diagram	Elements can be copied or pasted freely between Class, UseCase, Component, CompositeStructure, and Deployment diagrams.
Sequence Diagrams	Elements cannot be copied or pasted
Collaboration Diagrams	Elements cannot be copied or pasted
Statechart Diagram	Elements can be copied or pasted only between diagrams within the same StateMachine
Activity Diagram	Elements can be copied or pasted only between diagrams within the same ActivityGraph
Component Diagram	Elements can be copied or pasted freely between Class, UseCase, Component, CompositeStructure, and Deployment diagrams
Deployment Diagram	Elements can be copied or pasted freely between Class, UseCase, Component, CompositeStructure, and Deployment diagrams.
CompositeStructure Diagram	Elements can be copied or pasted freely between Class, UseCase, Component, CompositeStructure and Deployment diagrams.

Resize and Move

You can optimize the view size or position from the diagram area, and you can modify view position or size little by little by Special+Cursor Key.

Procedure for Resizing View:

1. Click a view to click in the diagram.
2. Modifies a size as dragging the point for direction where you want among points on select mark after selecting a view.

Procedure for Resizing View by using the keyboard:

1. Click a view to click in the diagram.
2. The user can specify for view resizing by using Shift+Cusor key. The Shift+Cursor Key can move to the present configured gird unit, and you can modify view position little by little by Shift+Alt+Cursor Key.

Procedure for moving View:

1. Selects the view to move in diagram as clicking mouse. If there are several views, select the views by Ctrl+Click or an area for including views as dragging.
2. Move views to where you want to go by using mouse.

Procedure for moving View by using the keyboard:

1. Selects the view to move in diagram as clicking mouse. If there are several views, select the views by Ctrl+Click or an area for including views as dragging.
2. Move views to where you want to go by using Ctrl+Cursor Key. The Ctrl+Cursor Key can move to the present configured gird unit, and you can modify view position little by little by Ctrl+Alt+Cursor Key.

Creating Element by using ShortCut Generation Syntax

Elements can also be created without being mouse by using the shortcut Generation Syntax.

Procedure creating element by using the ShortCut Generation Syntax:

1. Select from the diagram area the view.
2. Run Quick Dialog as selecting **[Enter]**.
3. Enter a syntax that is element in the quick dialog.

ShortCut Generation Syntax

Shortcut generation syntax can generate a target model and relationship with it by writing simple text. The basic rule of the shortcut generation syntax is as follows. Describe the target model names to make a relationship with notations to generate relationship. If there is no target model name, generate new appropriate model elements and the relationship. The relationship-notation of shortcut generation syntax to be used in each diagram is as follows:

Diagram Type	Notation	Current Element	Description
	<=	Classifier	The target element linking with the current element makes a link of specialization.
	=>	Classifier	The target element linking with the current element makes a link of generalization.
	--	Classifier	The target element linking with the current element makes a link of association.
	<-	Classifier	Makes navigable association relationship from target element to the current element.

Class Diagram Component Diagram Deployment Diagram Composite Structure Diagram	->	Classifier	The target element linking with the current element makes a link of navigable association.
	<>-	Classifier	The target element linking with the current element makes a link of aggregate.
	-<>	Classifier	Makes aggregate relationship from target element to the current element.
	<*>-	Classifier	The target element linking with the current element makes a link of compose.
	-<*>	Classifier	Makes compose relationship from target element to the current element.
	<--	Classifier	Makes dependency relationship from target element to the current element.
	-->	Classifier	The target element linking with the current element makes a link of dependency.
	)-	Classifier	Makes requirement relationship from target element to the current element.
	-(	Classifier	The target element linking with the current element makes a link of requirement.
	@-	Classifier	Makes realization relationship from target element to the current element.
	-@	Classifier	The target element linking with the current element makes a link of realization.
	()-	UseCase	The target model(Actor) linking with the current element makes a link of communication.

Usecase Diagram	-()	Actor	The target model(UseCase) linking with the current element makes a link of communication.
	<i-	UseCase	Makes include relationship from target element to the current element.
	-i>	UseCase	The target element linking with the current element makes a link of include.
	<e-	UseCase	Makes include relationship from target element to the current extend.
	-e>	UseCase	The target element linking with the current element makes a link of extend.
Sequence Diagram Seqeunce Diagram(Role)	<-	Object, ClassifierRole	The target element linking with the current element makes a link of stimulus.
	->	Object, ClassifierRole	Makes include relationship from target element to the current stimulus.
	<->	Object, ClassifierRole	Makes stimulus that has a return relationship from target element to the current element.
	<-	Stimulus, Message	Makes sub-stimulus(comes from target element) in current stimulus.
	->	Stimulus, Message	Makes sub-stimulus(goes from target element) in current stimulus.
	<->	Stimulus, Message	Makes sub-stimulus(with return goes from target element) in current stimulus.
	<~	Stimulus, Message	Makes stimulus(comes from target element) in front of current stimulus.
		Stimulus,	Makes stimulus(goes from target element) in front of current

	~>	Message	stimulus.
	<_	Stimulus, Message	Makes stimulus(comes from target element) in the rear of current stimulus.
	_>	Stimulus, Message	Makes stimulus(goes from target element) in the rear of current stimulus.
Collaboration Diagram Collaboration Diagram(Role)	<-	Object, ClassifierRole	The target element linking with the current element makes a link of stimulus.
	->	Object, ClassifierRole	Makes stimulus relationship from target element to the current element.
	<->	Object, ClassifierRole	Makes stimulus that has a return relationship from target element to the current element.
Statechart Diagram/ Activity Diagram	<-	State, ActionState	Makes transition relationship from target element to the current element.
	->	State, ActionState	The target element linking with the current element makes a link of transition.
	-*	State, ActionState	Makes transition relationship from target element(Initial State) to the current element.
	-@	State, ActionState	The target element(Final State) linking with the current element makes a link of transition.
	<-<>	State, ActionState	Makes transition relationship from target element(Decision) to the current element.
	-><>	State, ActionState	The target element(Decision) linking with the current element makes a link of transition.

	-(H) -(h)	State, ActionState	The target element(History) linking with the current element makes a link of transition.
	-(H*) -(h*)	State, ActionState	The target element(Deep History) linking with the current element makes a link of transition.
	<-|	State, ActionState	Makes transition relationship from target element to the current element(with Join).
	|->	State, ActionState	The target element(with Fork) linking with the current element makes a link of transition.

Editing Model Element

This section describes in detail the procedures for editing model elements.

- Configuring Property
- Documenting Model Element
- Attaching File or URL
- Recording Constraints
- Editing Tagged Values

Configuring Property

Model elements contain various properties. The user can change models in various ways by editing these property values. The following properties are available.

Property Types

Property Type	Description
Name	Indicates the name of the model element.
Stereotype	Indicates the stereotype for the model element.
TypeExpression	Indicates the expression for special type.
String	Indicates string.
Boolean	Indicates True or False.
Enumeration	Selects one of the various literals.
Reference	Indicates a specific element.
Collection	Indicates multiple elements (editable through the collection editor).

Editing the Name Property

Enter the element name in the **Name** item in the property editor. Names cannot contain these special characters "**:**". Names must also be unique within the namespace. For example, names of the classes within a package must all be unique. A warning message will appear if the name conflicts with another element.

Editing the Stereotype Property

Enter the stereotype name in the **Stereotype** item in the property editor. The stereotype name can be a stereotype defined in the UML profile or can be a simple name that is not pre-defined. The following methods can be used to edit the stereotype property.

- **Entering Defined Stereotype**: Enter a stereotype name that is defined in a profile included in the current project. The stereotype is directly referenced.

- **Entering Undefined Stereotype**: Enter a stereotype name that is not defined in the profiles included in the current project. This value is just a simple string value.
- **Selecting from the Stereotype Dialog Box**: Open the Stereotype dialog box and select a stereotype from the defined stereotype list.

Editing the TypeExpression Property

The TypeExpression property is included in Attribute, Parameter, etc. Enter the type expression in the **Type** item in the property editor. The following methods can be used to edit the type expression property.

- **Entering Defined Type Name**: Enter the name of a classifier element (classes, interfaces, signals, exceptions, components, nodes, subsystems, etc.) included in the current project. Elements are directly referenced.
- **Entering Defined Type Pathname**: Directly enter the pathname of a classifier element included in the current project (e.g. ::Logical View::Package1::Class1)
- **Entering Undefined Type Name**: Enter a name that is not related to any of the classifiers included in the current project. This value is just a simple string value.
- **Selecting from the Select Element Dialog Box**: Open the Select Element dialog box and directly select a defined type or select a data type defined in the profile.

Documenting Model Element

Detailed descriptions can be recorded for model elements.

Procedure for Documenting Model Element:

1. Select from the model explorer or the diagram area an element to include a description.
2. At the inspector area in the main window, select the **[Documentation]** tab.
3. Enter description in the editable area.

Attaching File or URL

Related files or web page URLs can be attached to elements. The attached files or web pages can be easily accessed through the associated applications or the web browser.

Procedure for Attaching File or URL:

1. Select an element from the model explorer or the diagram area.
2. At the inspector area in the main window, select the **[Attachments]** tab.
3. Right-click and select the **[Add]** menu or click the **[Add]** button on the toolbar.
4. At the Attachment dialog box, enter the full pathname and filename of the attachment file or the web page URL (or click the browse button on the right to select from the browse window), and click the **[OK]** button.

Procedure for Removing Attached Item:

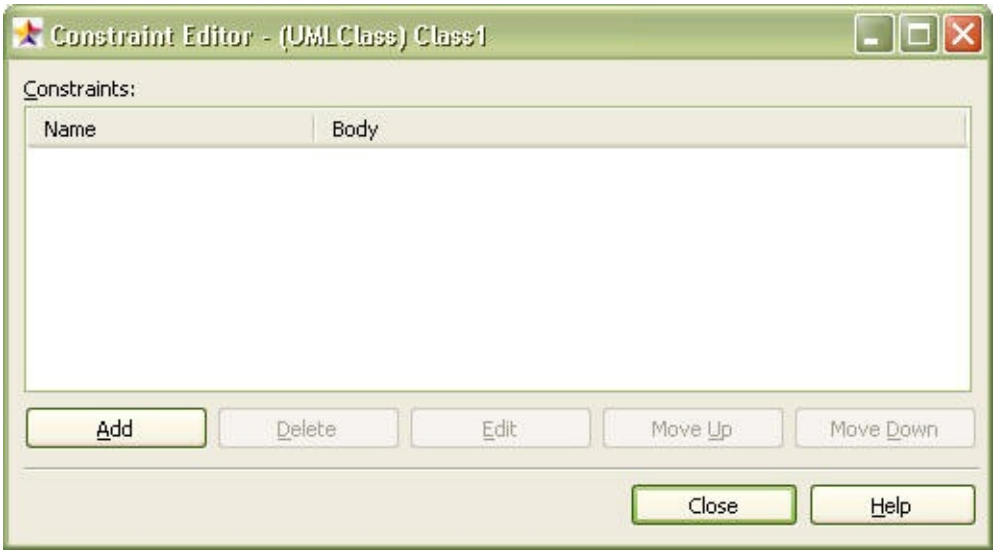
1. Select an element from the model explorer or the diagram area.
2. At the inspector area in the main window, select the **[Attachments]** tab.
3. Select an attached item to delete from the list. Right-click and select the **[Delete]** menu or click the button on the toolbar.

Recording Constraints

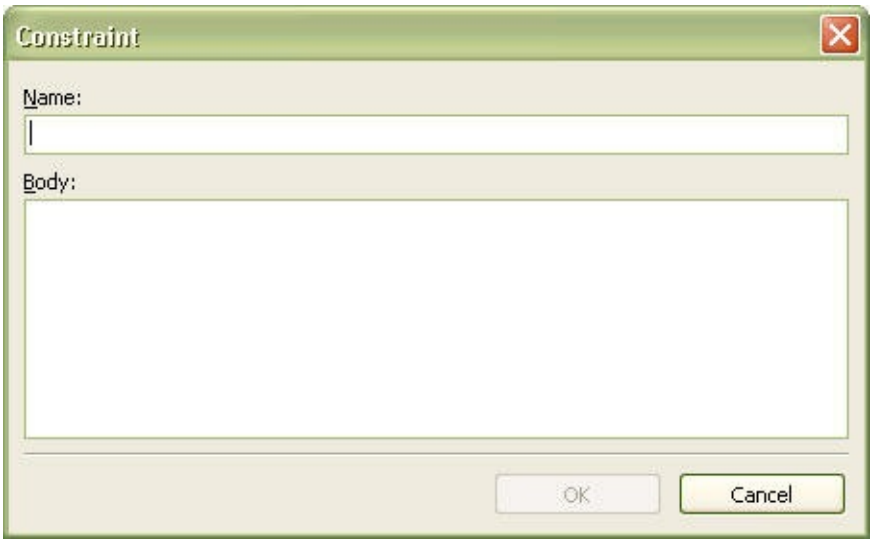
Multiple constraints can be recorded for elements. Constraints are regulations applied to elements. They can be written in easy-to-understand normal language, or be can be written to comply with the OCL (Object Constraint Language) grammar defined by UML.

Procedure for Adding Constraints:

1. Select an element to add constraints to.
2. Right-click and select the **[Constraint Editor]** menu.
3. At the Constraint Editor, click the **[Add]** button.



4. At the Constraint dialog box, enter the name and contents and then click the **[OK]** button.



Procedure for Deleting Constraints:

1. Select an element to delete constraints from.
2. Right-click and select the **[Constraint Editor]** menu.
3. At the Constraint Editor, select constraints to delete from the list and then click the **[Delete]** button.

Procedure for Editing Constraints:

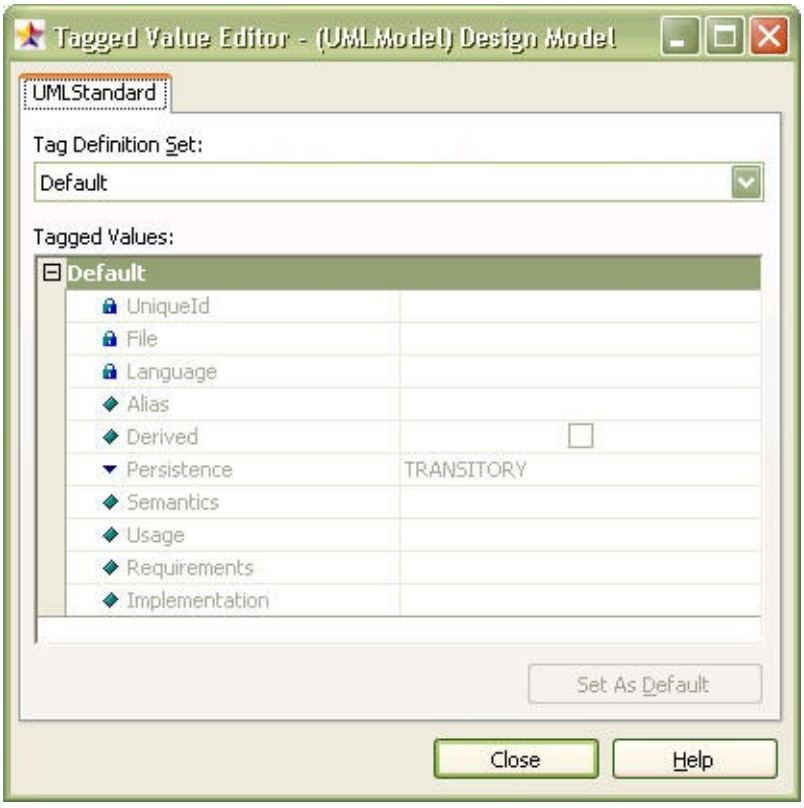
1. Select an element to edit constraints for.
2. Right-click and select the **[Constraint Editor]** menu.
3. At the Constraint Editor, select constraints to edit from the list and then click the **[Edit]** button.
4. At the Constraint dialog box, edit the name and contents. Click the **[OK]** button.

Editing Tagged Values

Besides the basic properties, the tagged values of elements, which are added by UML profiles, can be edited.

Procedure for Editing Tagged Value:

1. Select from the model explorer or the diagram area an element for which to edit the tagged value.
2. Right-click and select the **[Tagged Values]** menu.
3. At the Tagged Value Editor, select the tab that corresponds to the profile that contains the tagged value to edit.



4. Select from the **[Tag Definition Set]** combo box the set that contains the tagged value. Select a tagged value from the **[Tagged Values]** list and edit the value.

Procedure for Reverting Edited Tagged Values to Default Values:

1. Select from the model explorer or the diagram area the element that contains the tagged value.
2. Right-click and select the **[Tagged Values]** menu.
3. At the tagged value editor, select the tab that corresponds to the profile that contains the tagged value.
4. Select from the **[Tag Definition Set]** combo box the set that contains the tagged value. Select a tagged value from the **[Tagged Values]** list and click the **[Set to Default]** button.

Editing View Element

This section describes the functions for editing view elements: deleting views, applying formats, show/hide options, etc.

- Deleting View Element
- Showing Compartment Stereotype
- Showing Compartment Visibility
- Showing Operation Signature
- Showing Property
- Showing Parent Name
- Showing Stereotype
- Suppressing Literal
- Suppressing Operation
- Suppressing Attribute
- Applying Word Wrap
- Applying Font
- Applying Fill Color
- Applying Line Color
- Configuring Line Style
- Configuring Automatic Resize

Deleting View Element

Deleting a view element means deleting only the view element that represents a model element on the screen, without deleting the model element itself.

Procedure for Deleting View Element:

1. In order to delete a view element, select the view element shown in the diagram.
2. Hit the **[Del]** key or select the **[Edit] -> [Delete]** menu.

Note

- Deleting a view element does not delete its model element.

Showing Compartment Stereotype

Elements like classes, usecases, and subsystems that contain attributes, operations, literals, etc. have compartments to show their attributes and operations in diagram. Class has attribute and operation compartments, subsystem has an operation compartment, and enumeration has literal and operation compartments. Stereotypes of the elements (attributes, operations, etc.) displayed in these compartments can be configured to be shown or hidden.

Procedure for Showing Compartment Stereotype:

1. Select from the diagram area an element for which to show the compartment stereotype.
2. Right-click and select the **[Format] -> [Show Compartment Stereotypes]** menu.
3. Perform the steps above once again to hide the compartment stereotype.

Showing Compartment Visibility

Elements like classes, usecases, and subsystems that contain attributes, operations, literals, etc. have compartments to show their attributes and operations in diagram. Class has attribute and operation compartments, subsystem has an operation compartment, and enumeration has literal and operation compartments. Visibility of the elements displayed in these compartments can be configured to be shown or hidden.

Procedure for Showing Compartment Visibility:

1. Select from the diagram area an element for which to show the compartment visibility.
2. Right-click and select the **[Format] -> [Show Compartment Visibility]** menu.
3. Perform the steps above once again to hide the compartment visibility.

Showing Operation Signature

When displaying elements that contain operations such as class and subsystem, the parameter names and types for operations can be configured to be shown or hidden.

Procedure for Showing Operation Signature:

1. Select from the diagram area an element to show the operation signature.
2. Right-click and select the **[Format] -> [Show Operation Signature]** menu.

Perform the steps above once again to hide the operation signature.

Showing Property

Among the element tag definitions, element tagged values and changeability attributes are shown in the view elements property section. The user can configure this property section to be shown or hidden.

Procedure for Showing Properties:

1. Select from the diagram area an element for which to show the properties.
2. Right-click and select the **[Format] -> [Show Properties]** menu.

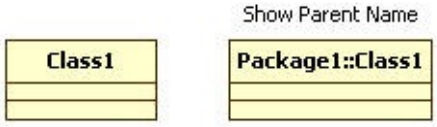
Perform the steps above once again to hide the properties.

Note

- In the case of Changeability property value of AssociationEnd element is changeable or Ordering property value is UNORDERED, the relative property value is not be showed in the property part of diagram view element.

Showing Parent Name

In general, view elements show their own names only. However, a project containing multiple packages may have elements with the same names in different packages, and there may be cases where these elements need to be displayed in the same diagram. In such a case, the elements need to show their parent names in order to be distinguished from one another. The names are in the format ParentName::OwnName.



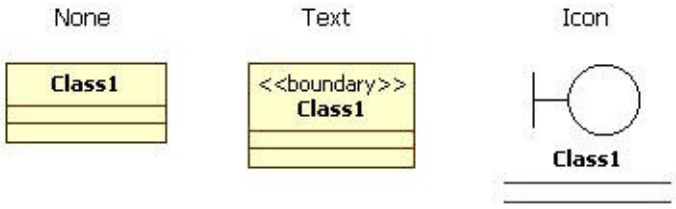
Procedure for Showing Parent Name:

1. Select from the diagram area an element for which to show the parent name.
2. Right-click and select the **[Format] -> [Show Parent Name]** menu.

Perform the steps above once again to hide the parent name.

Showing Stereotype

View elements can be expressed as different shapes depending on the stereotypes. The following expression formats are available.



- **Hide [Shift+Ctrl+N]**: Hides the stereotype.
- **Show with Text [Shift+Ctrl+T]**: Stereotype name is shown inside << and >>.
- **Show with Icon [Shift+Ctrl+I]**: View element is expressed with the stereotype icon. The stereotype must be registered with an icon to use this option. Otherwise the stereotype is shown in text.
- **Show with Decoration[Shift+Ctrl+I]** : View elements is described as text and small-sized stereotype icon. In this case, icons in the stereotype have to be registered, and it is described as text if it is not. The some elements like Actor, Interface, Component, Node and Artifact are showed as decoration type as the default icon if they are not registered in stereotype.

Suppressing Literal

Enumerations have literals, which are shown in the literal compartment areas of enumerations in the diagram. The user can configure these literals to be shown or suppressed.

Procedure for Suppressing Literals:

1. Select from the diagram area an enumeration type element for which to hide the literals.
2. Right-click and select the **[Format] -> [Suppress Literals]** menu.

Perform the steps above once again to show the literals.

Suppressing Operation

Elements that contain operations such as class, exception, usecase and subsystem show these operations in their operation compartment areas. The user can configure these operations to be shown or suppressed.

Procedure for Suppressing Operations:

1. Select from the diagram area an element for which to hide the operations.
2. Right-click and select the **[Format] -> [Suppress Operations]** menu.

Perform the steps above once again to show the operations.

Suppressing Attribute

Elements that contain attributes such as Class, Exception and UseCase show these attributes in their attribute compartment areas. The user can configure these attributes to be shown or suppressed.

Procedure for Suppressing Attributes:

1. Select from the diagram area an element for which to hide the attributes.
2. Right-click and select the **[Format] -> [Suppress Attributes]** menu.

Perform the steps above once again to show the attributes.

Applying Word Wrap

When an element name is defined as more over a word, visibility of diagram is decreased since the size of the view is being over extension. If you use Word Wrap, you can optimize the view size as expressing the long name of elements to several lines.

Procedure for applying Word Wrap:

1. Select from the diagram area an element for which to apply Word Wrap.
2. Right-click and select the **[Format] -> [Word Wrap Name]** menu.

Perform the steps above once again to removed Word Wrap.

Note

- Some elements such as relative elements, unexpressed elements on a diagram and Swimlane cannot apply Word Wrap.

Applying Font

Text font shape, color, size, etc. for view elements can be changed.

Procedure for Applying Font:

1. Select from the diagram area an element for which to change font.
2. Right-click and select the **[Format] -> [Font]** menu.
3. At the Font dialog box, select font shape, size, color, etc. and click the **[OK]** button.

Note

- **[Font style]** for some UML-related view elements are not editable. This is because the font styles are defined by the UML conventions and cannot be changed.

Applying Fill Color

Fill colors for view elements can be changed.

Procedure for Applying Fill Color:

1. Select from the diagram area an element for which to change the fill color.
2. Right-click and select the **[Format] -> [Fill Color]** menu.
3. At the Color dialog box, select a color to apply and click the **[OK]** button.

Applying Line Color

Colors for the view element outlines or connecting lines can be changed.

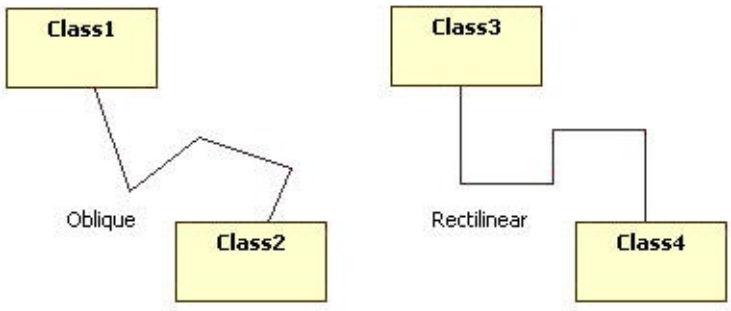
Procedure for Applying Line Color:

1. Select from the diagram area an element for which to change the line color.
2. Right-click and select the **[Format] -> [Line Color]** menu.
3. At the Color dialog box, select a color to apply and click the **[OK]** button.

Configuring Line Style

Line type view elements such as Association, Dependency and Generalization are expressed by either of the following two line styles.

- **Rectilinear**: Line always changes in 90 degree angles.
- **Oblique**: Line changes at any angle.



Procedure for Changing Line Style:

1. Select from the diagram area a view element that has a Line Style.
2. Right-click and select the **[Format] -> [Line Style]** menu. Select rectilinear or oblique.

Configuring Automatic Resize

Although the user can change the view element sizes at any time, view elements can also be configured to resize automatically.

Procedure for Configuring Automatic Resize for View Element:

1. Select from the diagram area a view element to configure automatic resize.
2. Right-click and check the **[Format] -> [Auto Resize]** menu.
3. To remove the automatic resize setting, select the checked menu item once again to uncheck it.

Diagram Management

This section describes the procedures for managing diagrams: opening, closing and deleting diagrams.

- Opening Diagram
- Closing Diagram
- Deleting Diagram

Opening Diagram

In order to edit a diagram, the diagram must be opened. Once a diagram is opened, the tabs for the diagram are displayed. Select a tab to make the diagram active for editing.

Procedure for Opening Diagram:

1. Search for the diagram to open in the model explorer or the diagram explorer.
2. Double-click the diagram to open it. The diagram automatically becomes active.

Closing Diagram

Close a diagram if it no longer needs to be edited. Closing a diagram does not delete it. A closed diagram can be opened again at any time.

Procedure for Closing Diagram:

1. Select the tab of the diagram to close to make the diagram active.
2. Right-click on the tab and select the **[Close Diagram]** menu.

Procedure for Closing All Open Diagrams:

1. Select the **[View] -> [Close All Diagrams]** menu.

Deleting Diagram

A diagram can be deleted if it is no longer needed. Please be careful, because deleting a diagram also deletes all information related to the diagram.

Procedure for Deleting Diagram:

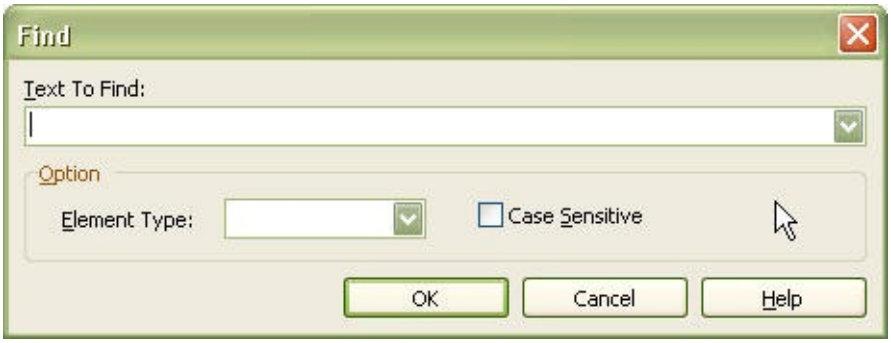
1. Select a diagram to delete, from the model explorer or the diagram explorer.
2. Right-click and select the **[Delete Model]** menu.

Finding Element

Software models usually contain a large number of elements. Sometimes it becomes very difficult to locate wanted elements from among the many elements in a software model. The Find Element function can be used to search the wanted elements quickly.

Procedure for Finding Element:

1. Select the **[Edit] -> [Find]** menu.
2. At the Find dialog box, enter in the **[Find what]** field the full or partial name of the element to find. To limit the element types to find, select the element type from the **[Options-Element type]** menu. To match cases, check the **[Options-Match case]** item. Click the **[OK]** button.



3. The find results are added in the **[Messages]** section of the information area. Double-click a message to find the related element.

Aligning Element

Elements laid out in diagram can be aligned in certain directions or with certain spacing.

Align Element Function

Align Function	Description
Align Left	Align the selected elements to the left.
Align Right	Align the selected elements to the right.
Align Middle	Center the selected elements horizontally.
Align Top	Align the selected elements to the top.
Align Bottom	Align the selected elements to the bottom.
Align Center	Center the selected elements vertically.
Space Equally, Horizontally	Evenly distribute the selected elements horizontally.
Space Equally, Vertically	Evenly distribute the selected elements vertically.
Bring to Front	Bring the selected elements to the front.
Send to Back	Send the selected elements to the back.

Procedure for Aligning Elements:

1. Select the elements to align in the diagram area (two or more elements must be selected for aligning, except for Bring to Front and Send to Back).
2. Right-click and select the **[Format] -> [Align]** menu. Select the menu for the aligning method wanted.

Layout Diagram

In cases where the diagram elements are laid out in a disordered way, the elements can be automatically laid out for tidier display.

Procedure for Laying Out Diagram Elements:

1. Make a diagram to layout the active diagram.
2. Right-click and select the **[Format] -> [Layout Diagram]** menu.

Note

- The layout diagram function is not available for Sequence Diagram.

Configuring Zoom-In/Zoom-Out

If there are too many elements in the diagram area or if the element texts are too small, the diagram can be zoomed in or zoomed out for better view.

Procedure for Zooming In/Zooming Out Diagram:

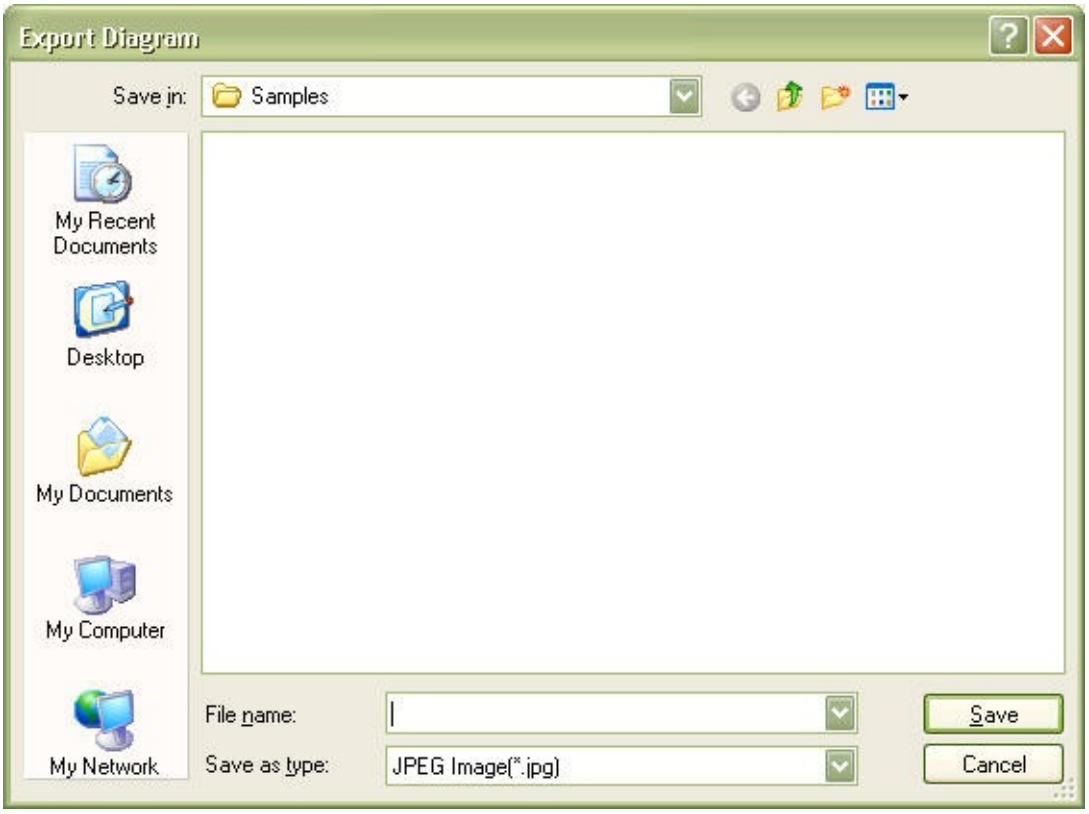
1. Select the **[View]** -> **[Zoom]** menu.
2. Select the **[Zoom-In]** menu to zoom-in the diagram by one level (5%), or select the **[Zoom-Out]** menu to zoom-out by one level. To display the whole diagram in one screen, select the **[Fit to Window]** menu. You may also select a zooming ratio (50%, 75%, 100%, 125%, 150%, 175%, and 200%).

Saving Diagram as Image File

Diagrams can be saved as image files. StarUML™ supports these image formats: JPEG (.jpg, .jpeg), bitmap (.bmp), metafile (.wmf), and extended metafile (.emf).

Procedure for Saving Diagram as Image:

1. Make a diagram to save as image the active diagram.
2. Select **[File] -> [Export Diagram]** from the main menu.
3. At the Save dialog box, enter the file name, select the file format, and then click the **[Save]** button.



Note

- In the case of metafile(.wmf) images, some viewer may not display. It is recommended to used to the extended metafile(.emf).

Copying Diagram as bitmap

In order to insert a editing diagram to other document, the diagram image can be copied as bitmap. The diagram can be inserted to a document as editing image itself if copying it as bitmap, but it could have image distortion in the case of zoom in/out.

Procedure for copying diagram as bitmap:

1. Make a diagram to copy as bitmap the active diagram.
2. Select **[Edit] -> [Copy Diagram As Bitmap]** from the main menu.

Note

- Diagram information is copied to meta image if copying by Ctrl+C after selecting View. The meta image has no image distortion as zoon in/out in a document, but it could have difference with real image of the diagram in text editor program.

Navigating Diagram

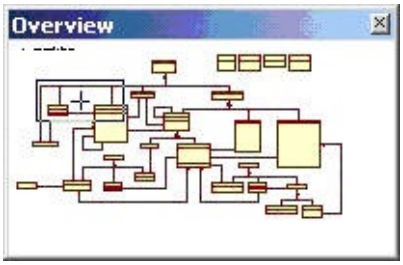
If a diagram contains a lot of information, the diagram may become very large. In this case, only a limited section of the diagram can be shown on the screen. StarUML™ provides various methods to effectively navigate the diagram area, allowing the user to move to specific diagram locations quickly. The following methods can be used for navigating diagram.

Navigating with ScrollBar and Wheel

Moves for diagram domain what you want as using scroll bar. If you use wheel mouse, you can move to up and down by using mouse wheel.

Navigating with Birs Eye View

There is a small icon ⊹ at the lower right-hand corner of the diagram area. Click this icon to see the entire diagram in a small area. Move to a diagram location while holding down the mouse button and then release the mouse button. This function is useful for navigating over a long distance.



Navigating with Ctrl + Mouse

Hold down the Ctrl key and move the mouse to move the diagram. This function is useful for navigating over a short distance.

Configuring Default Diagram

A project can contain many diagrams. Among the many diagrams, there can be more than one default diagram, which is the most basic diagram of all. For instance, a diagram that expresses the overall structure of the project can be configured as the default diagram. Only Class Diagram, UseCase Diagram, Component Diagram or Deployment Diagram can be set as the default diagram. The default diagram is automatically opened when opening the project.

Procedure for Configuring Default Diagram:

1. Select from the model explorer or the diagram explorer a diagram to configure as the default diagram.
2. Select the **[Properties]** tab in the inspector area.
3. At the property editor, check the **DefaultDiagram** property.

Organizing Model Structure

This section describes in detail the procedures for editing model elements.

- Creating Model Element
- Deleting Model Element
- Moving Model Element
- Modify Model Element Order
- Model Alignment

Creating Model Element

Model elements can also be created without being displayed in the diagram. Such a model is not displayed in any diagrams, and more than one view can be made later to represent it in diagrams.

Procedure for Creating Model Element:

1. Select from the model explorer an element to contain the new model element.
2. Right-click and select the **[Add]** menu and select an element type from the menu. Or, select the **[Model] -> [Add]** menu from the main menu.
3. The new model element will be created under the selected model.

Deleting Model Element

If you delete a model element, many related elements are deleted together. Please exercise caution because deleting a model element results in deletion of the following elements.

- **Included Model Elements**: All model elements included in the model being deleted are also deleted.
- **Related Model Elements**: All relations such as Generalization, Association and Dependency related to the model element being deleted are also deleted.
- **View Elements**: All view elements that represent the model being deleted are also deleted.

Procedure for Deleting Model Element:

1. Select from the model explorer a model element to delete, or select a view element from the diagram area to delete the model element represented by it.
2. Hit **[Ctrl+Del]** or select the **[Edit] -> [Delete Model]** menu.
3. The selected model element is deleted.

Moving Model Element

Model elements can be moved so as to be placed under other elements, such as by moving a class to be placed under another package or moving an attribute to be placed under another class. Model elements can be moved to be placed only under elements that can contain model elements. They cannot be moved to be placed under other types of elements.

Procedure for Moving Model Element:

1. Select from the model explorer an element to move.
2. Drag the element and drop it at the element that is to contain it.

Modify Model Element Order

The order between model elements can be modified to show intuitively configurations of software model. The order modification between model elements can be only among same kinds of elements. Also, it can be if sort of model navigator is only Storage Order.

Procedure for modify order of model element :

1. Select an element to modify the order in model navigator.
2. Move model element to a line as push **[Move UP]** or **[Move Down]** button.

Elements such as Attribute, Operation, Enumeration Literal which is expressing in Collection editor can be modified their order in Collection editor.

Procedure to modify order of model element in collection editor :

1. Select upper element of an element to modify its order.
2. Run collection editor as selecting **[Model]->[Collection Editor...]**.
3. Select tap which is relative in collection including element.
4. Select element to modify the order.
5. Modify the order of the model element as push [Move Up] or [Move Down] button. You can modify the order by using Ctrl+Cursor key.

Model Alignment

The structure of models in model navigator can be aligned as saving order or alphabet order. The aligned model is only shown by model navigator, the order among real models are not modified. In order to sort models, click [Align as saving order] or [Align as Alphabetical order]. If change model alignment way, the spreading statue of each node in model navigator is cancelled, and the top leveled node is spreaded.

Modeling with UseCase Diagram

The following elements are available in a usecase diagram.

- Actor
- UseCase
- Association / Directed Association
- Generalization
- Dependency
- Include
- Extend
- System Boundary
- Package

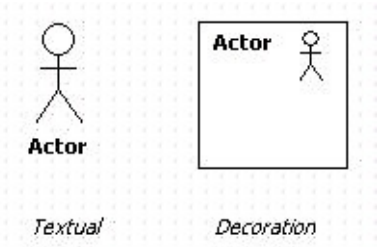
Actor

Semantics

An actor defines a coherent set of roles that users of an entity can play when interacting with the entity. An actor may be considered to play a separate role with regard to each use case with which it communicates.

Procedure for creating Actor

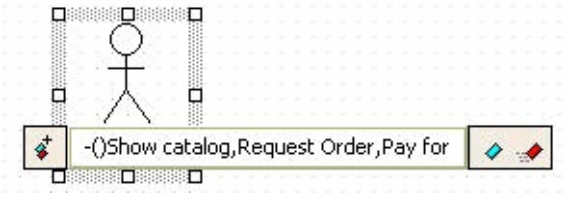
In order to create Actor, click **[Toolbox] -> [UseCase] -> [Actor]** button and click the position where to place Actor. Actor is shown in the form of stick man or rectangle with icon, that is decoration view. To display actor in decoration view, select **[Format] -> [Stereotype Display] -> [Decoration]** menu item or select **[Decoration]** item in [ ] combo button on toolbar.



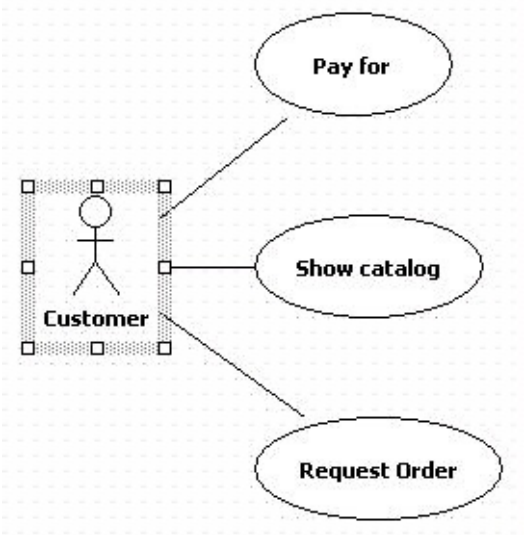
Procedure for creating multiple UseCases used by Actor at once

In order to create multiple UseCases related to Actor at once, use shortcut creation syntax of Actor.

1. At the Actor's quick dialog, enter UseCase's name after "-()" string. To create multiple UseCases, enter same but separate UseCase's name by "," character.



2. And press **[Enter]** key. Several UseCases associated with the Actor are created and arranged vertically.


Pay for
Show catalog
Customer
Request Order

UseCase

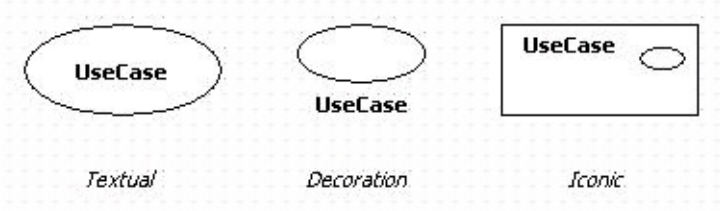
Semantics

The use case construct is used to define the behavior of a system or other semantic entity without revealing the entitys internal structure. Each use case specifies a sequence of actions, including variants, that the entity can perform, interacting with actors of the entity.

Procedure for creating UseCase

In order to create UseCase, click **[Toolbox]** -> **[UseCase]** button and click the position where to place UseCase on the **[main window]**.

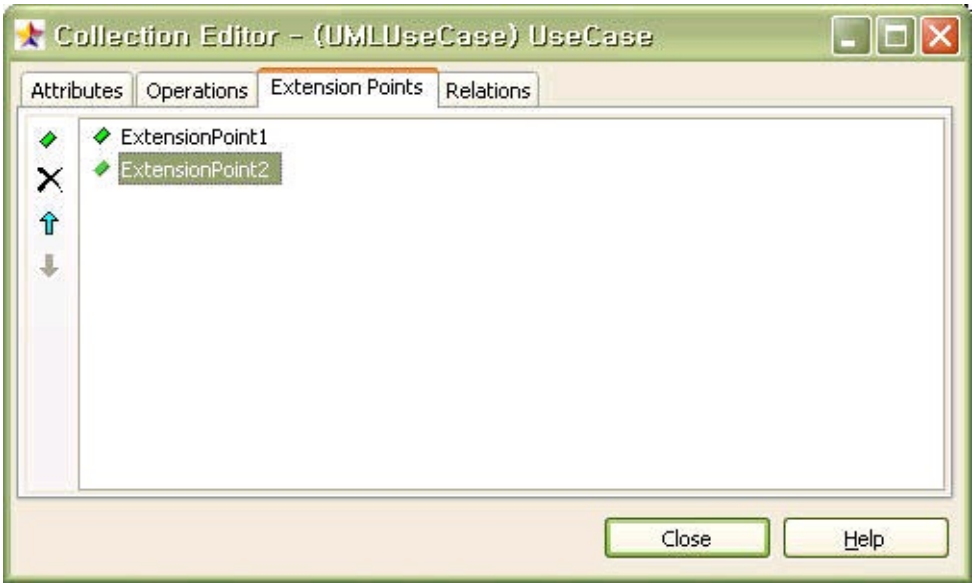
UseCase is expressed in the forms of textual, decoration, iconic. To change UseCase's view style, select menu item under **[Format] -> [Stereotype Display]** or select [ ]button's combo item.



Procedure for adding Extension

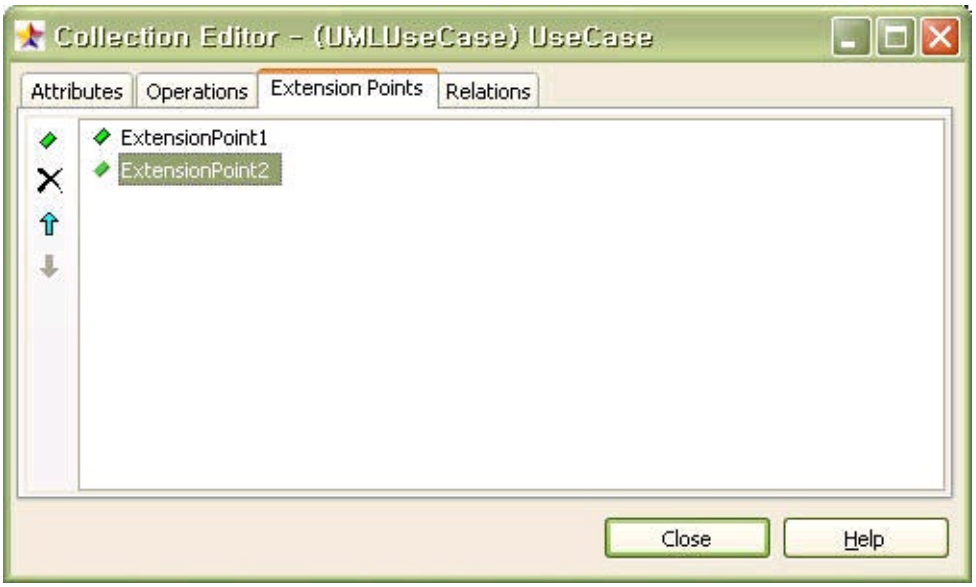
An extension point references one or a collection of locations in a use case where the use case may be extended.

To edit ExtensionPoints of UseCase, click UseCase's **[Collection Editor...]** popup menu or click button of **[ExtensionPoints]** collection property.



Procedure for entering UseCase specification

To enter basic flow, alternative flow properties of usecase, select **[Tagged Values...]** popup menu or click **[Ctrl+F7]** button. At tagged value editor, select **[UseCaseSpecification]** item and enter the properties.



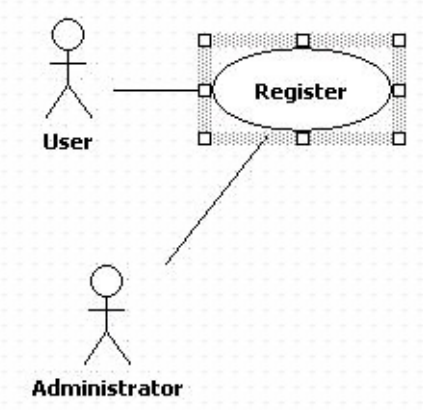
Procedure for creating Actor from UseCase

In order to create multiple Actors related to UseCase at once, use shortcut creation syntax.

Double-click UseCase, or select UseCase and press **[Enter]** key. At quick

1. dialog, enter Actor's name after "()-" string and separate Actor names by "," character.

2. And press **[Enter]** key. Several Actors associated with the UseCase are created and arranged vertically.



Association / Directed Association

Semantics

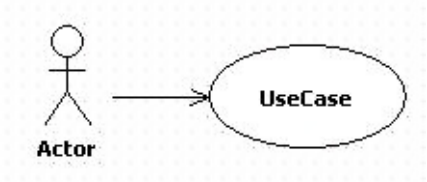
A association is an association among exactly two classifiers (including the possibility of an association from a classifier to itself).

Procedure for creating association

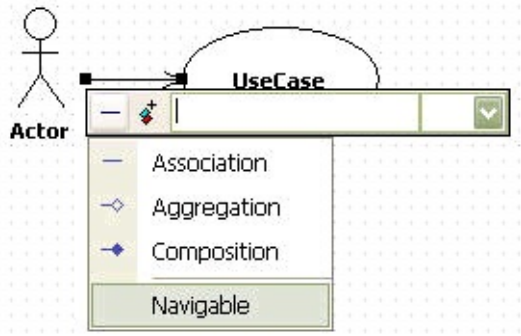
In order to create association, click **[Toolbox] -> [UseCase] -> [Association]** button, drag from first element, and drop to second element in the **[main window]**.

Procedure for creating directed association

The procedure is equal to the association's, drag and drop in the arrow direction.



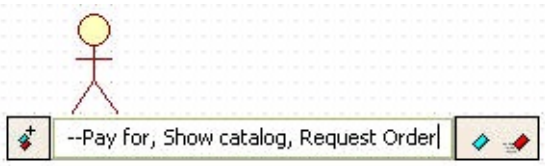
Or create association, click the actor-side association end. At the quick dialog, uncheck navigable and association becomes directed.



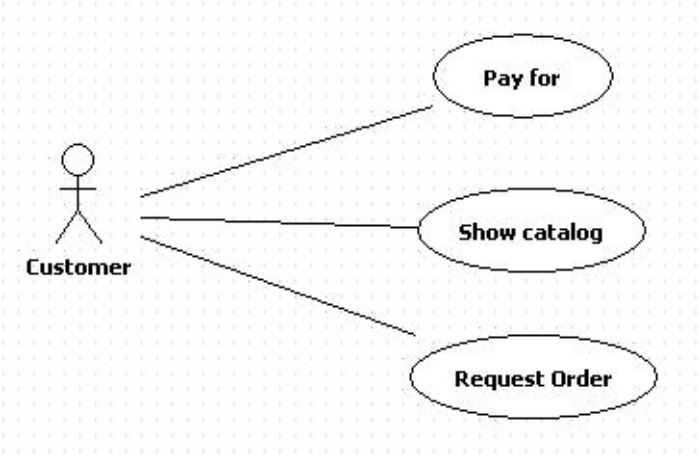
Procedure for creating element related to association/directed assocition

In order to create element associated with current element, use shortcut creation syntax.

1. Double-click element and enter element's names associated after "--" or "->" string at the quick dialog. Separate element names with "," character to relate multiple elements.



2. Press **[Enter]** key and several elements associated with selected element are created and arranged automatically.



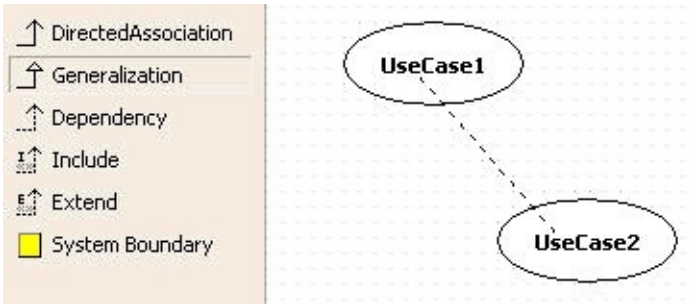
Generalization

Semantics

Generalization is the taxonomic relationship between a more general element (the parent) and a more specific element (the child) that is fully consistent with the first element and that adds additional information.

Procedure for creating generalization

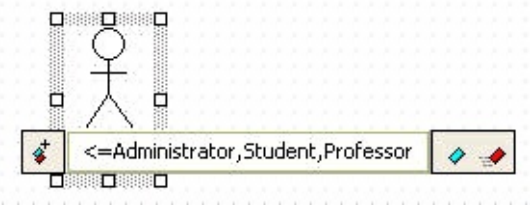
In order to make generalization, click **[Toolbox] -> [UseCase] -> [Generalization]** button, drag from child element and drop to parent element in the **[main window]**.



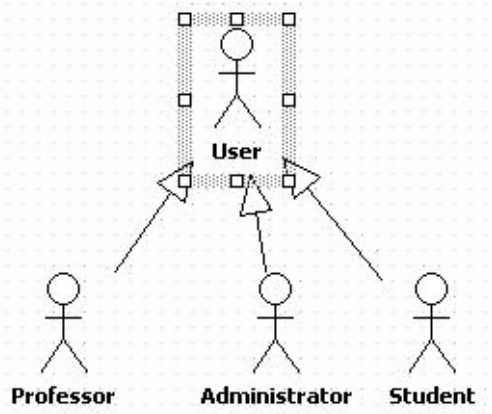
Procedure for creating multiple child actors inherited from actor

To create multiple elements inherited from some element,

1. Enter with "<=" string as following at the quick dialog, and several elements inherited from selected element are created at once.



2. Child elements are generated below selected element and arranged automatically.



If you want to create multiple parent element at once, enter "=>" string instead of "<=" in the quick dialog.

Dependency

Semantics

A dependency is a type of relationship that signifies that one element, or group of elements, acting as the client depends on another element or group of elements that act as a supplier. It is a weak relationship that denotes that if the supplier is changed the client may be affected. It is a unidirectional relationship.

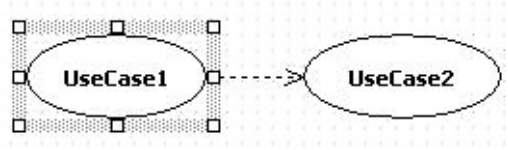
Procedure for creating dependency

In order to create dependency, click **[Toolbox] -> [UseCase] -> [Dependency]** button, drag element and drop to other element depended.

Procedure for creating other usecase depended by current usecase

Enter with "-->" string at the quick dialog as following.

So dependency relationship is created between two elements.



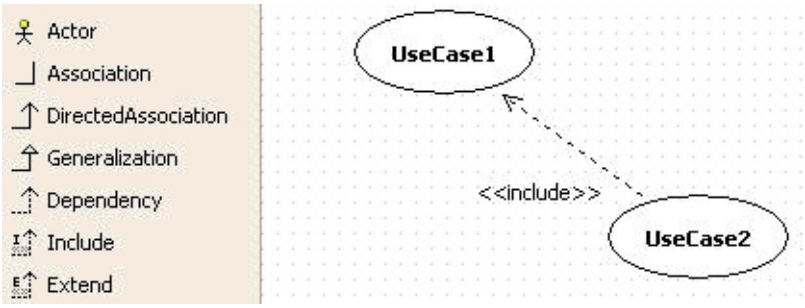
Include

Semantics

An include relationship defines that a use case contains the behavior defined in another use case.

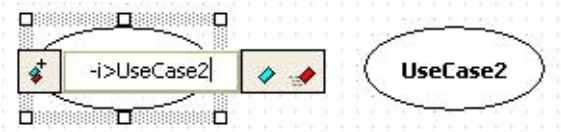
Procedure for creating include

In order to create include relationship, click **[Toolbox] -> [UseCase] -> [Include]** button, drag from element including and drop to element included in the **[main window]**.

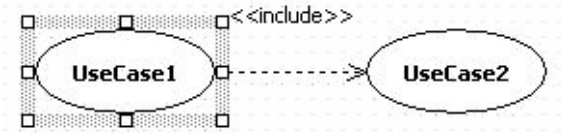


Procedure for creating other usecase included by current usecase

Enter with "-i>" string at the quick dialog as following.



So include relationship is created between two elements.



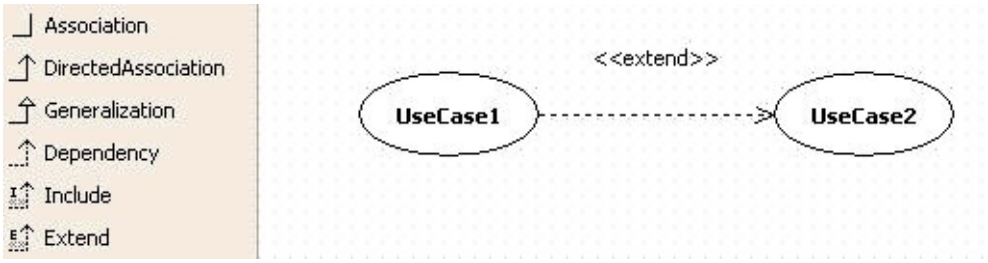
Extend

Semantics

An extend relationship defines that instances of a use case may be augmented with some additional behavior defined in an extending use case.

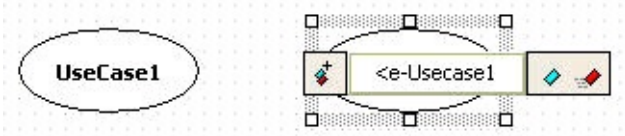
Procedure for creating extend

In order to create extend, click **[Toolbox] -> [UseCase] -> [Extend]** button, drag from element extending and drop to element extended in the **[main window]**.

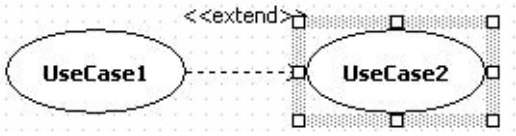


Procedure for creating other usecase extending current usecase

Enter with "<e-" string at the quick dialog as following.



So extend relationship is created between two elements.



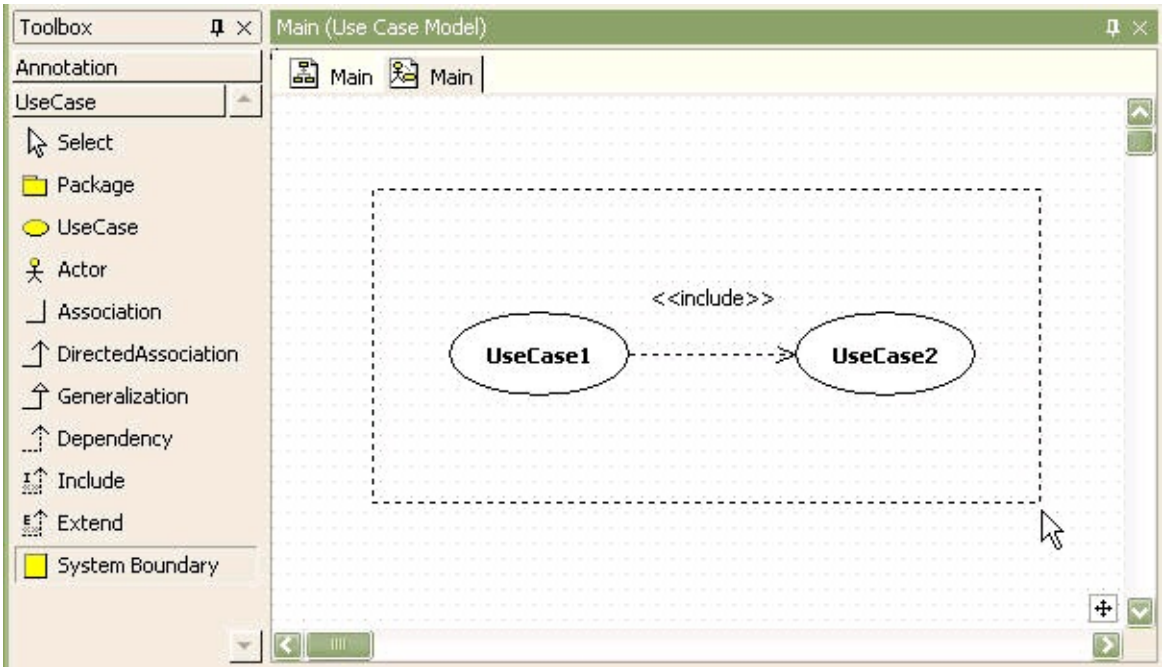
System Boundary

Semantics

A *System Boundary* is a type of partition that represents the boundary between the thing you are representing with the use cases (inside the boundary) and the actors (outside the boundary). Its most typical usage is the boundary of an entire system. Use cases can be used to represent subsystems and classes and so the boundary may be more specific than an entire system. A package with a stereotype *topLevel* can be used as a boundary and name space within the use case model to denote the same thing as the *use case boundary*.

Procedure for creating system boundary

In order to create system boundary, click **[Toolbox] -> [UseCase] -> [System Boundary]** button, drag from the starting point of system boundary and drag to right-bottom point of system boundary.



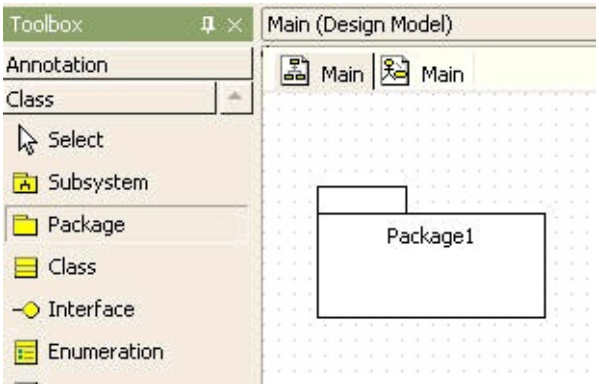
Package

Semantics

A package is a grouping of model elements. Packages themselves may be nested within other packages. A package may contain subordinate packages as well as other kinds of model elements. All kinds of UML model elements can be organized into packages.

Procedure for creating package

In order to create package, click **[Toolbox] -> [UseCase] -> [Package]** button and click at the location where package will be placed in the **[main window]**.



Modeling with Class Diagram

The following elements are available in the class diagram.

- Subsystem
- Package
- Class
- Interface
- Enumeration
- Signal
- Exception
- Port
- Part
- Association
- DirectedAssociation
- Aggregation
- Composition
- Generalization
- Dependency
- Realization
- AssociationClass
- Connector
- Object
- Link

Subsystem

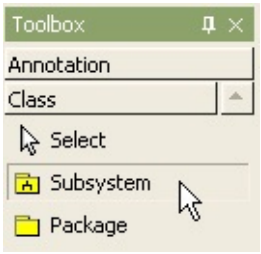
Semantics

Whereas a package is a generic mechanism for organizing model elements, a subsystem represents a behavioral unit in the physical system, and hence in the model.

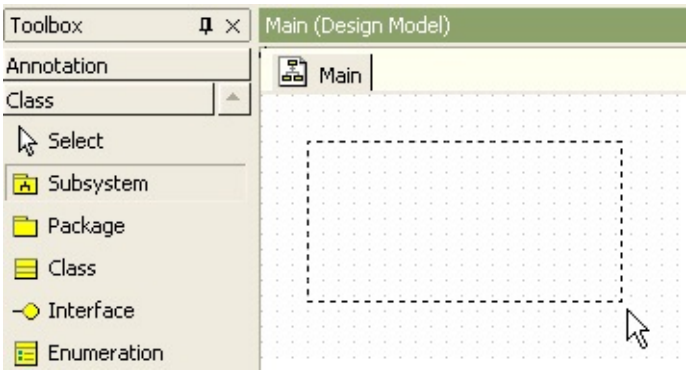
Procedure for creating subsystem

In order to create subsystem,

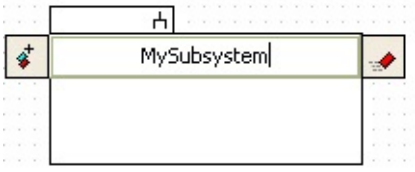
1. Click **[Toolbox]** -> **[Class]** -> **[Subsystem]** button.



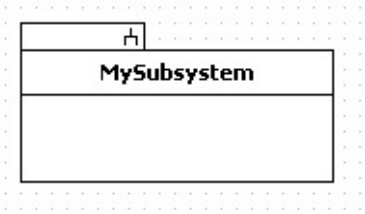
2. And click at the location or boundary where subsystem will be placed in the **[main window]**.



3. Then a subsystem is created on the class diagram and subsystem quick dialog is opened. At the quick dialog, enter the subsystem name.



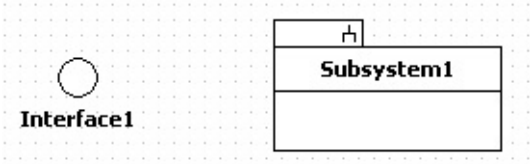
4. And press **[Enter]** key to have done this procedure.



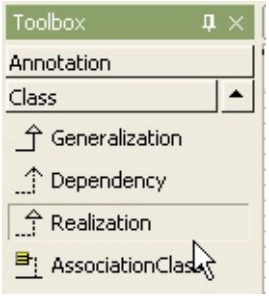
Procedure for creating providing interface of subsystem.

In order to providing inteface of subsystem,

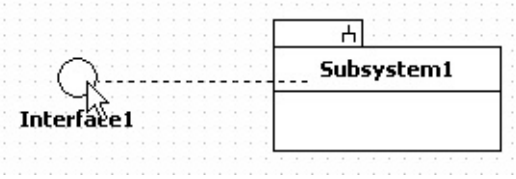
1. Create interface and susbystem.



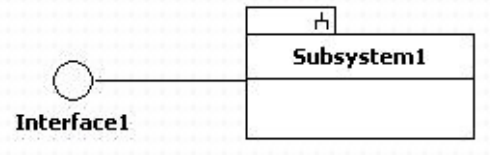
2. Click **[Toolbox]** -> **[Realization]** button.



3. Drag from subsystem and drop to interface.

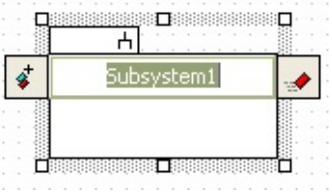


4. Between interface and subsystem, providing interface relationship is created finally.

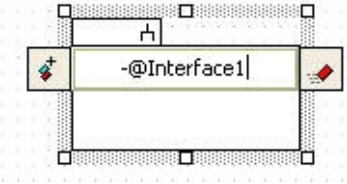


In order to create interface and realization at once,

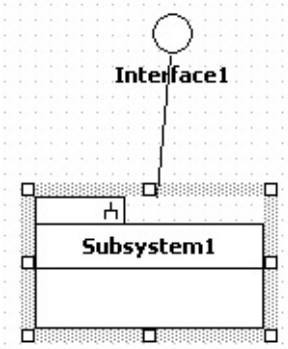
1. Double-click subsystem and subsystem quick dialog is opened.



2. Enter text in the quick dialog as following



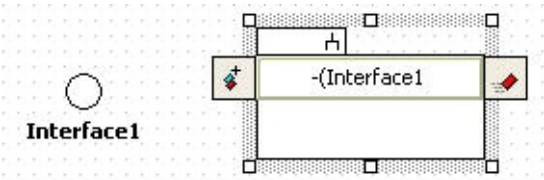
3. Press **[Enter]** key and interface provided by subsystem is created .



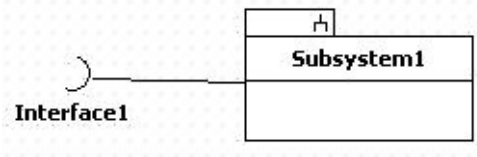
Procedure for creating requiring interface

In order to create requiring interface, use shortcut creation syntax.

1. Double-click subsystem. At the quick dialog, enter text as follows.



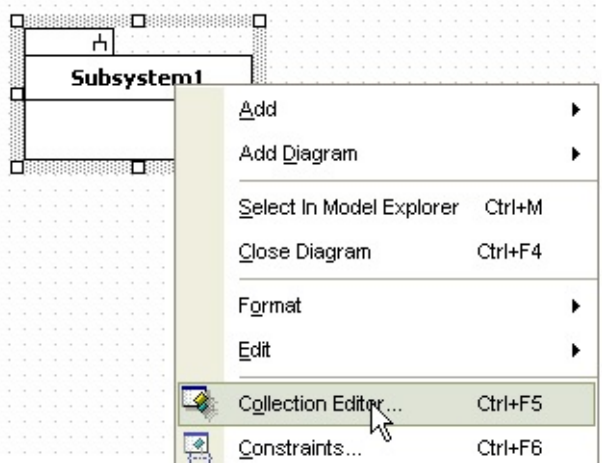
2. Then subsystem connects to interface as requiring relationship.



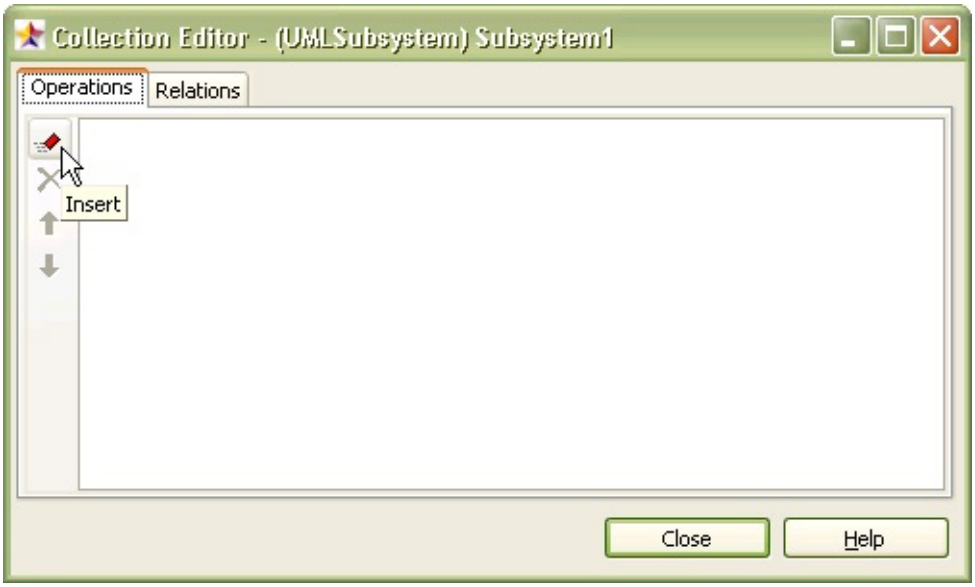
Procedure for adding operation to subsystem

Subsystem can have operation. In order to add operation to subsystem,

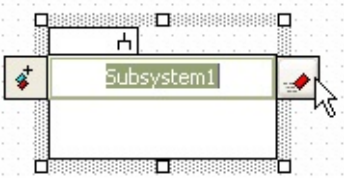
1. Select **[Collection Editor...]** popup menu.



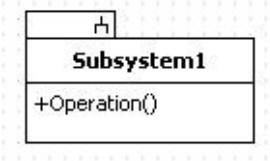
2. At the **[collection editor]**, add operation on the **[operations]** tab.



3. Or click  button at the quick dialog of subsystem.



4. Then a new operation is created.


Subsystem1
+Operation()

Class

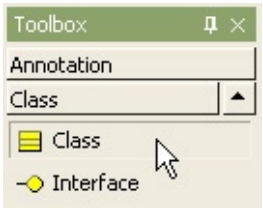
Semantics

A class is the descriptor for a set of objects with similar structure, behavior, and relationships.

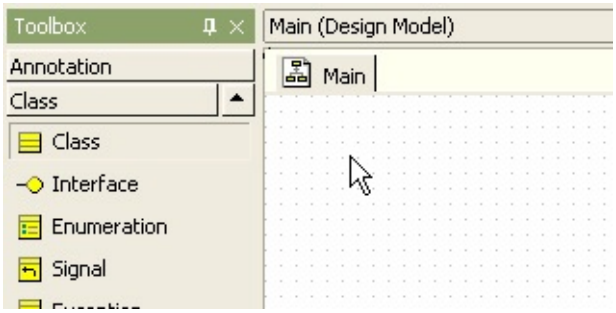
Procedure for creating class

In order to create class,

1. Click **[Toolbox]** -> **[Class]** -> **[Class]** button.

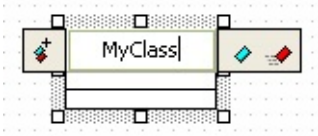


2. And click at the position where class will be placed in the [main window].



3. A new class is created on the diagram and class quick dialog is opened.

4. At the quick dialog, enter the class name and press **[Enter]** key.



Procedure for adding attribute

There are three method to add attribute to class.

- using quick dialog

  using model in the **[main window]** or the **[model explorer]**

-
- using **[collection editor]**

In the case of using quick dialog,

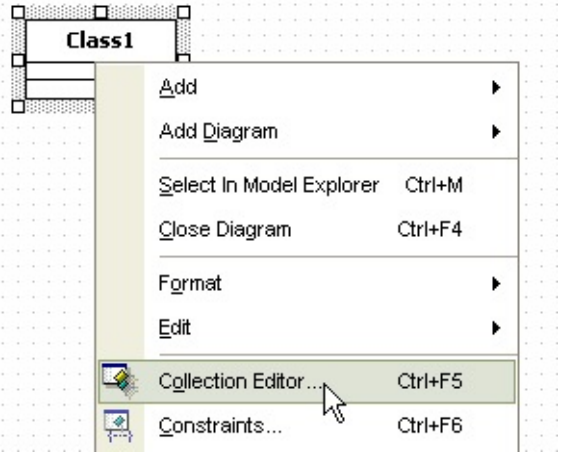
1. Double-click class.
2. Press **[Add Attribute]** button at the quick dialog, and you can add attribute.

In the case of using model,

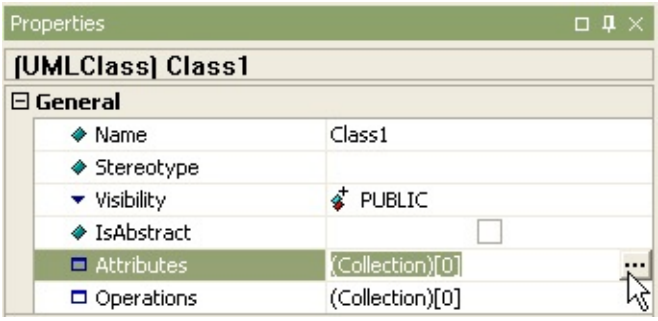
1. Select class in the **[main window]** or in the **[model explorer]**.
2. Right-click the selected class, select **[Add] -> [Attribute]** popup menu, and you can do.

In the last case,

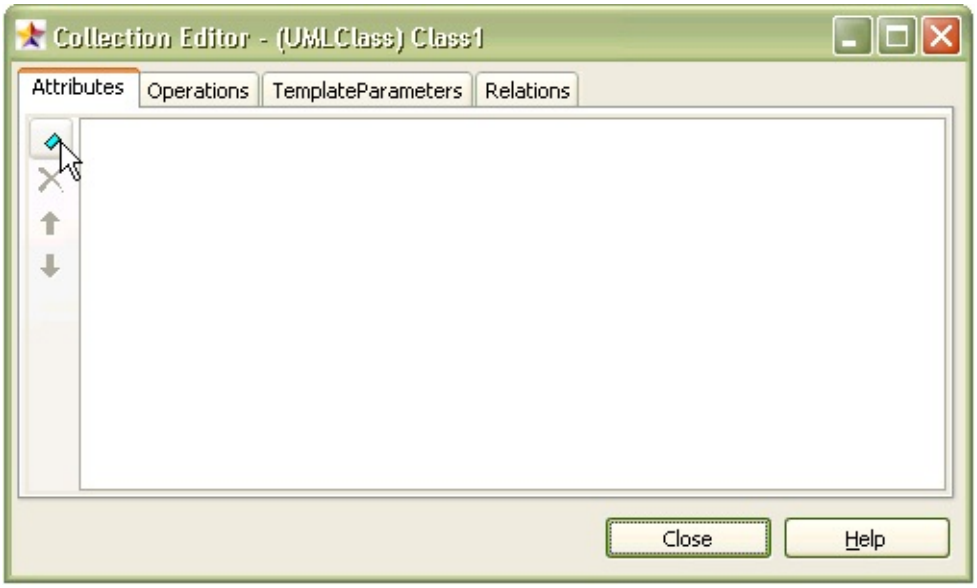
1. Select **[Collection Editor...]** popup menu.



2. Or click ··· button in **[attributes]** property on properties window.



3. At **[attribute]** tab of the **[collection editor]**, you can add attribute by using button.



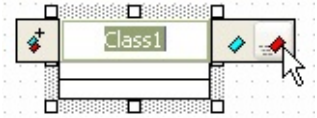
Procedure for adding operation

There are three method to add attribute to class.

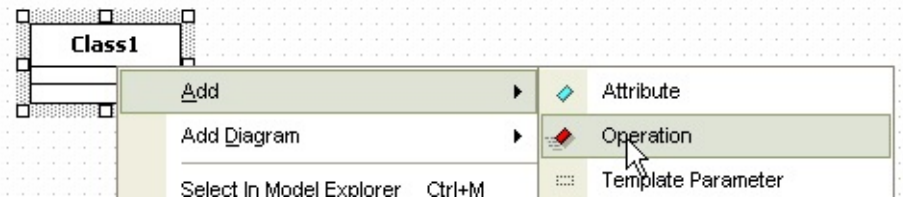
- using quick dialog
- using model in the **[main window]** or the **[model explorer]**
- using **[collection editor]**

In the case of using quick dialog,

1. Double-click class and class quick dialog is shown.
2. Press **[Add Operation]** button at the quick dialog, and you can add operation.

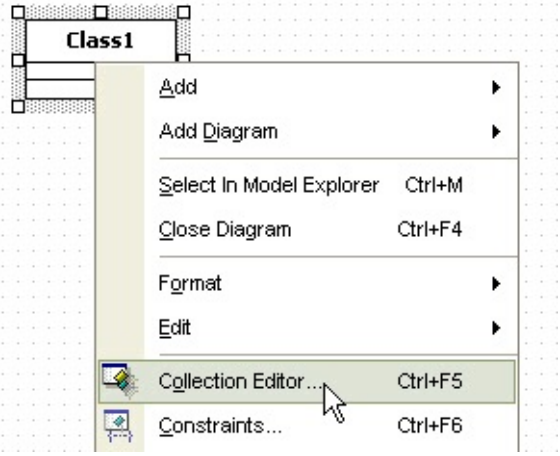


In the case of using model, select class in the **[main window]** or in the **[model explorer]**, right-click the selected class, select **[Add] -> [Operation]** popup menu, and you can do.

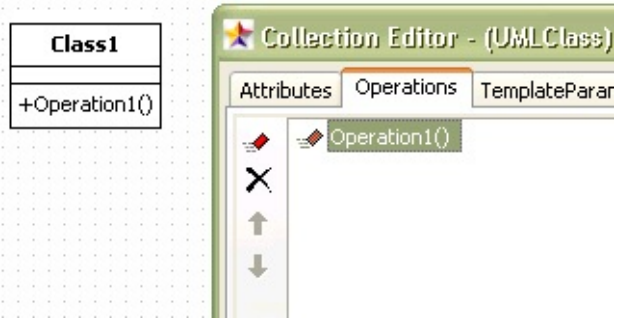


In the last case,

1. Select **[Collection Editor...]** popup menu.



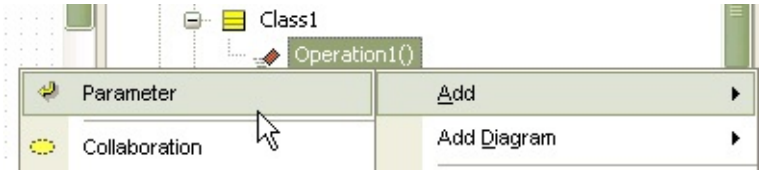
2. At **[operations]** tab of the **[collection editor]**, you can add operation by using  button.



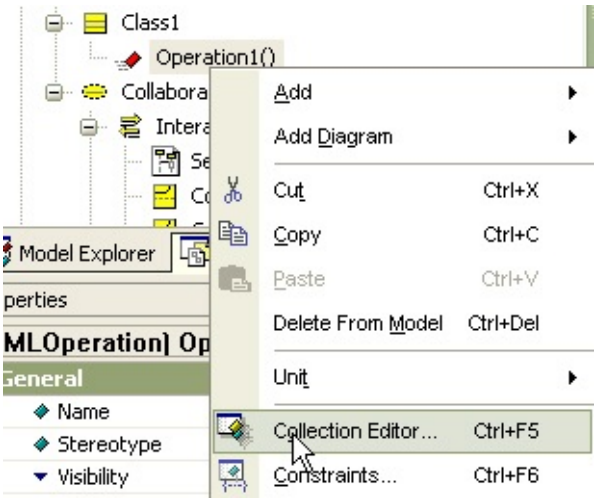
Procedure for adding parameter to operation

In order to add parameter to operation,

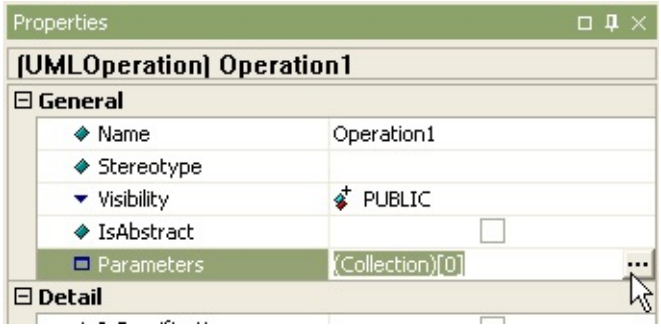
1. Select operation in the **[model explorer]**, select **[Add] -> [Parameter]** popup menu, and new parameter will be added.



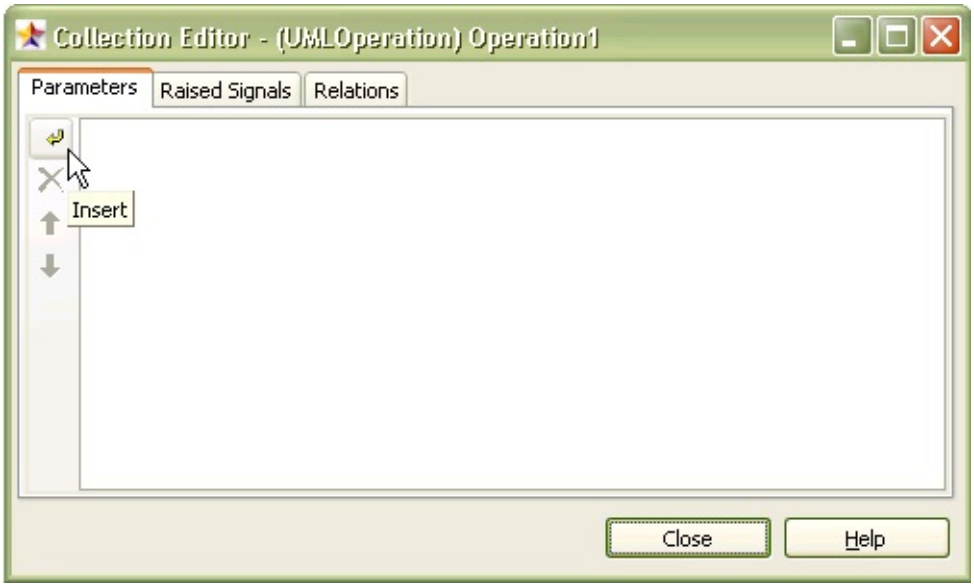
2. Or select operation in the **[model explorer]**, select **[Collection Editor...]** popup menu.



3. Or click ... button in **[Parameters]** property on properties window.



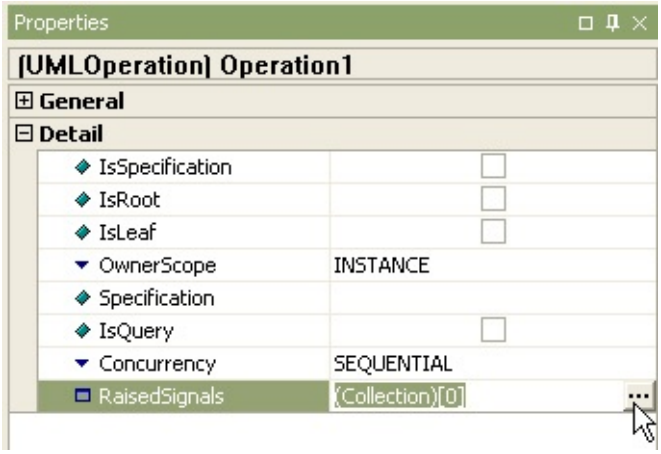
4. At the **[Parameters]** tab of the **[collection editor]**, you can add parameter by using button.



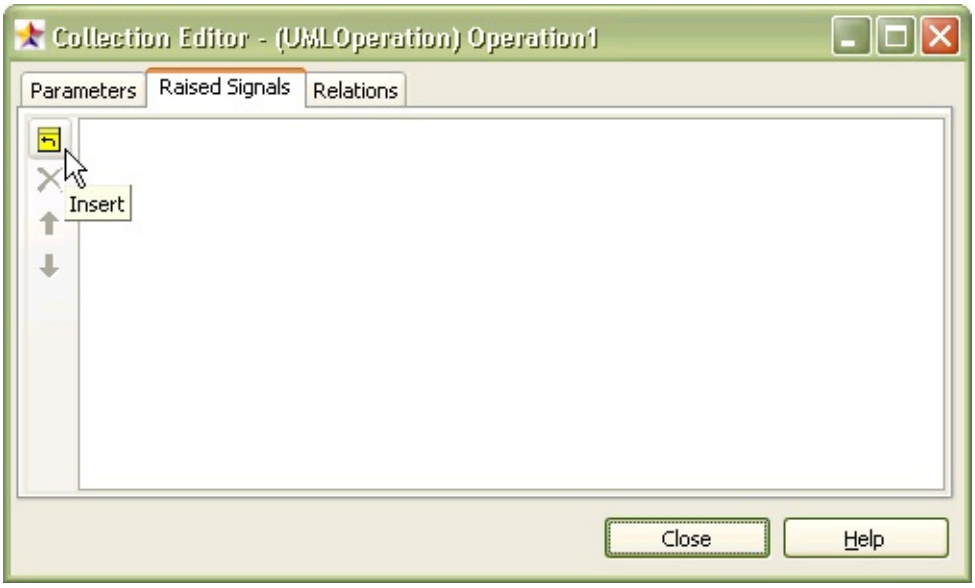
Procedure for adding exception to operation:

Before this procedure, there must exist a exception or more. To do this, see **"Procedure for creating signal" or "Procedure for creating exception"**.

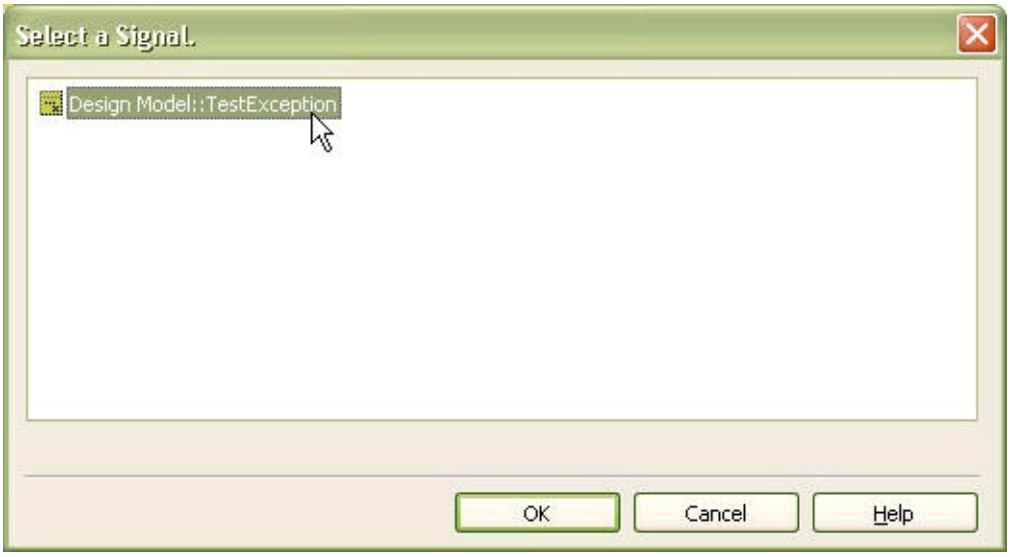
1. Click ··· button in **[RaisedSignals]** property on properties window.



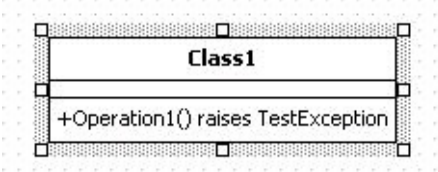
2. At **[Raised Signals]** tab of the **[collection editor]**, you can add exception to the operation by using button.



3. At **[Select a Signal]** dialog, select signal or exception raised by operation and click **[OK]** button.



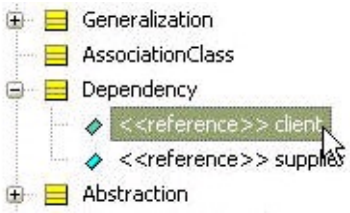
4. The result is as follows.



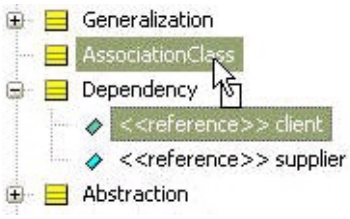
Procedure for moving attribute/operation into other class

In order to move attribute or operation into the other class,

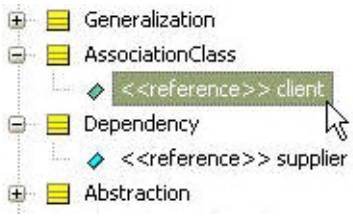
1. Click a attribute(or operation).



2. Drag it.



3. Drop it into another class.

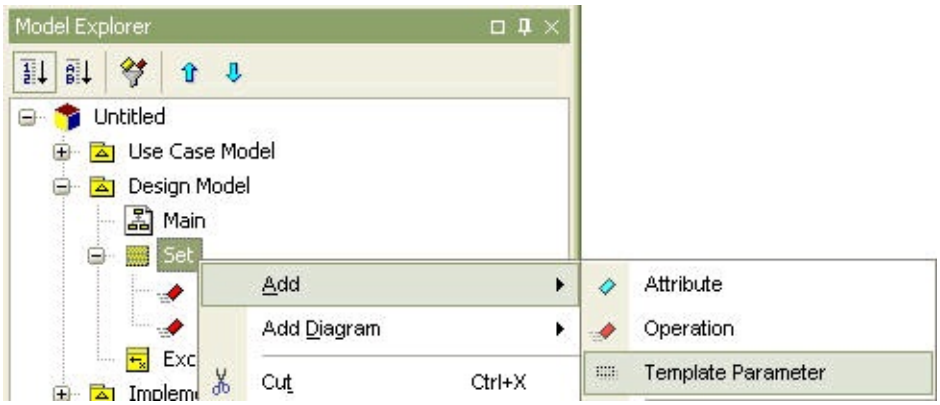


Procedure for adding template parameter to class

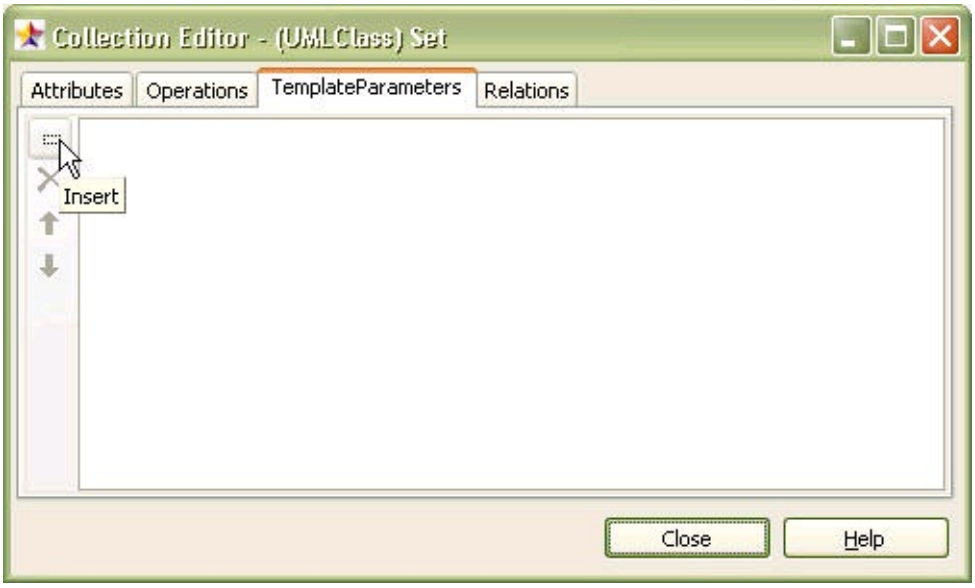
There are two way to add template parameter to class

- using class model in the **[main window]** or the **[model explorer]**
- using **[collection editor]**

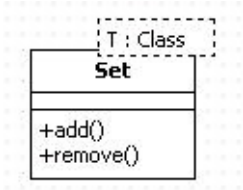
Select class in the **[main window]** or the **[model explorer]**, right-click, and select **[Add] -> [TemplateParameter]** popup menu. Then you can add template to class.



Select **[Collection Editor...]** popup menu or click ··· button in **[TemplateParameter]** property on properties window. At the **[TemplateParameters]** tab of the **[collection editor]**, you can add template parameter to class by using button.



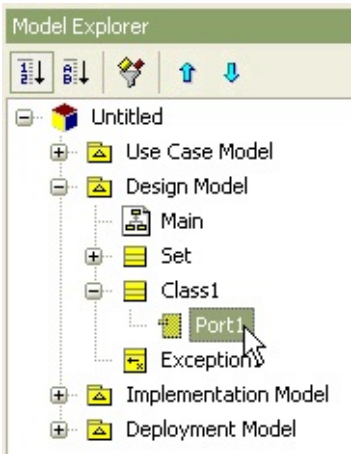
A new template parameter is added to class. The result is as follows.



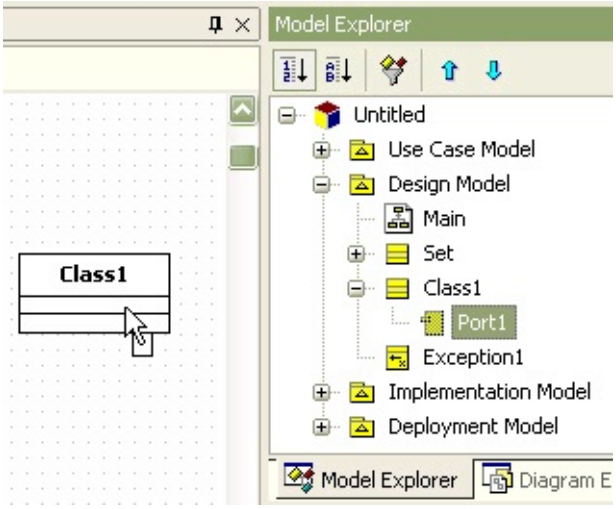
Procedure for creating view by dragging port

You can create port by dragging port from **[model explorer]** to main diagram.

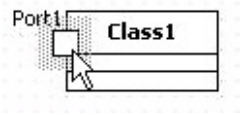
1. Drag port in the **[model explorer]**.



2. Drop on the class in the main diagram. If it is not dropped on the class but on the other area of the diagram, Class with port will be created.



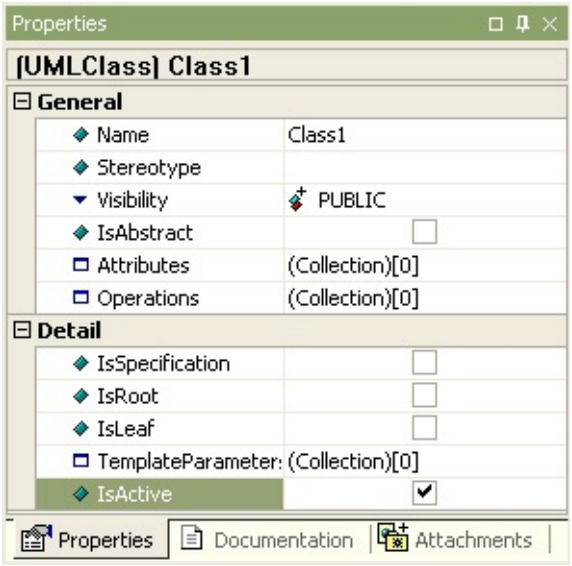
3. The class has a port as follows.



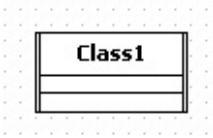
Procedure for setting active class

In order to set class to active class,

1. Set class's **[IsActive]** property to true.



2. The result class is shown as follows.



Interface

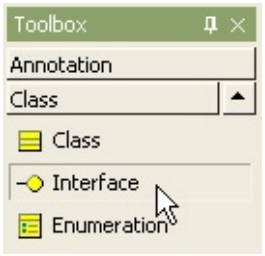
Semantics

An interface is a specifier for the externally-visible operations of a class, component, or other classifier (including subsystems) without specification of internal structure.

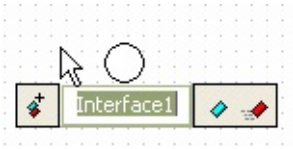
Procedure for creating interface

In order to create class,

1. Click **[Toolbox] -> [Class] -> [Interface]** button.



2. And click at the position where interface will be placed in the **[main window]**. Then interface quick dialog is opened. Enter the interface name at the quick dialog.

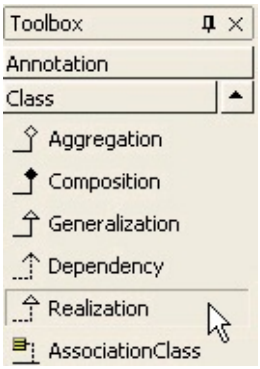


3. Press [Enter] key. Then the result is as follows.

Procedure for creating providing relationship

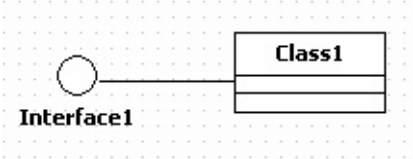
In order to create providing relationship,

1. Click **[Toolbox] -> [Class] -> [Realization]** button.



2. Drag from one(Class, Port, Part, Package, Subsytem) and drop to interface in the **[main window]**.

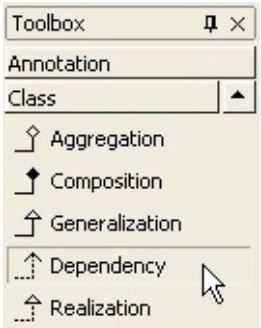
3. Then providing interface relationship is created as follows.



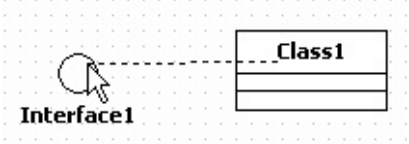
Procedure for creating requiring relationship

In order to create requiring relationship,

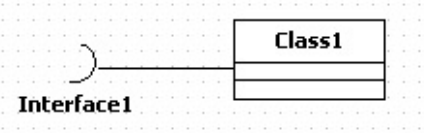
1. Click **[Toolbox]** -> **[Class]** -> **[Dependency]** button.



2. Drag from one(Class, Port, Part, Package, Subsytem) and drop to interface in the **[main window]**.



3. Then requiring interface relationship is created as follows.



Enumeration

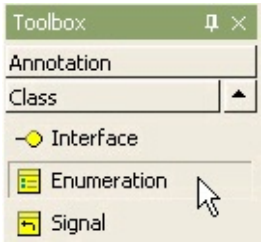
Semantics

An Enumeration is a user-defined data type whose instances are a set of user-specified named enumeration literals. The literals have a relative order but no algebra is defined on them.

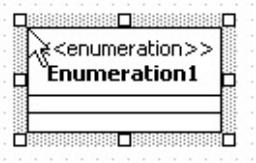
Procedure for creating enumeration

In order to create enumeration,

1. Click **[Toolbox] -> [Class] -> [Enumeration]** button.



2. And click at the position where enumeration will be placed in the **[main window]**.



Signal

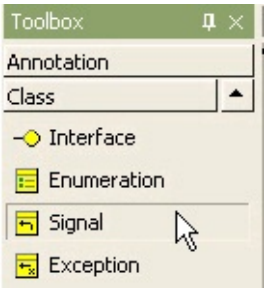
Semantics

A signal is a specification of an asynchronous stimulus communicated between instances. The signal is a child to Classifier, with the parameters expressed as Attributes. A Signal is always asynchronous. A Signal is associated with the BehavioralFeatures that raise it.

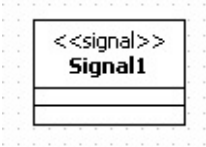
Procedure for creating signal

In order to create signal,

1. Click **[Toolbox]** -> **[Class]** -> **[Signal]** button.



2. And click at the position where signal will be placed in the **[main window]**.



Exception

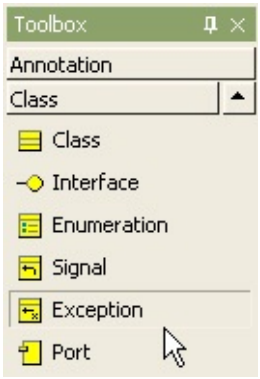
Semantics

An exception is a signal raised by behavioral features typically in case of execution faults. An Exception is associated with the BehavioralFeatures that raise it.

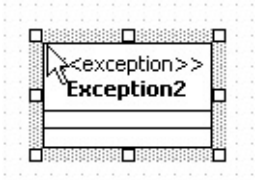
Procedure for creating exception

In order to create exception,

1. Click **[Toolbox] -> [Class] -> [Exception]** button.



2. And click at the position where exception will be placed in the **[main window]**.



Port

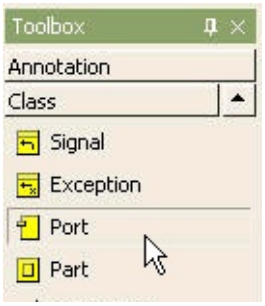
Semantics

A Port is a structural feature of a classifier that specifies a distinct interaction point between that classifier and its environment or between the (behavior of the) classifier and its internal parts..

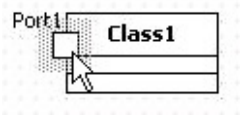
Procedure for creating port

In order to create port,

1. Click **[Toolbox]** -> **[Class]** -> **[Port]** button.



2. And click the class where the port will be contained in the **[main window]**.

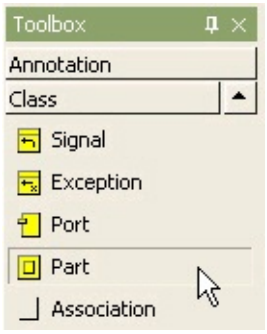


Part

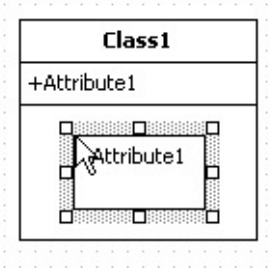
Procedure for creating part

In order to create part,

1. Click **[Toolbox]** -> **[Class]** -> **[Part]** button.



2. And click the class where the part will be contained in the **[main window]**.



Association

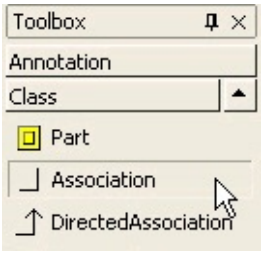
Semantics

An association is an association among exactly two classifiers (including the possibility of an association from a classifier to itself).

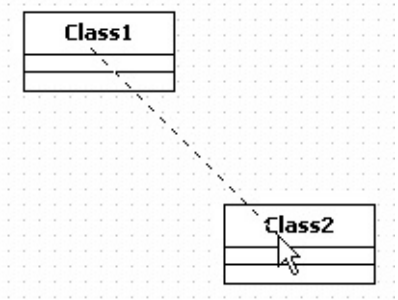
Procedure for creating association

In order to create association,

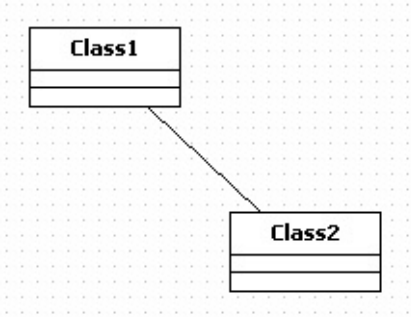
1. Click **[Toolbox]** -> **[Class]** -> **[Association]** button.



2. Drag from one associated and drop to another in the **[main window]**.



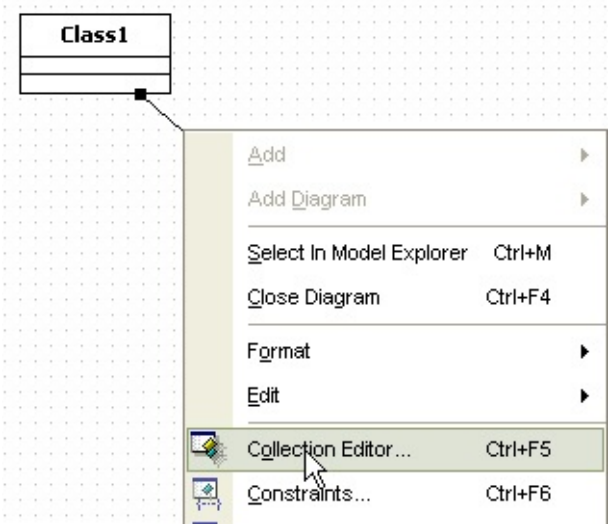
3. Between two classes, a new association is created as follows.



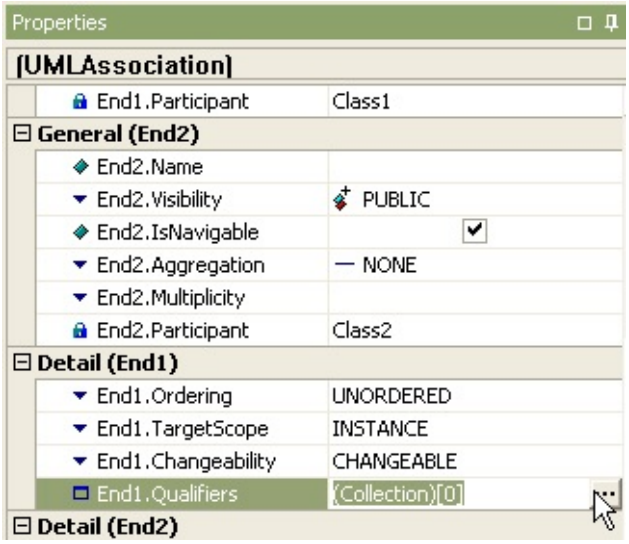
Procedure for adding qualifier to association

In order to add qualifier to association,

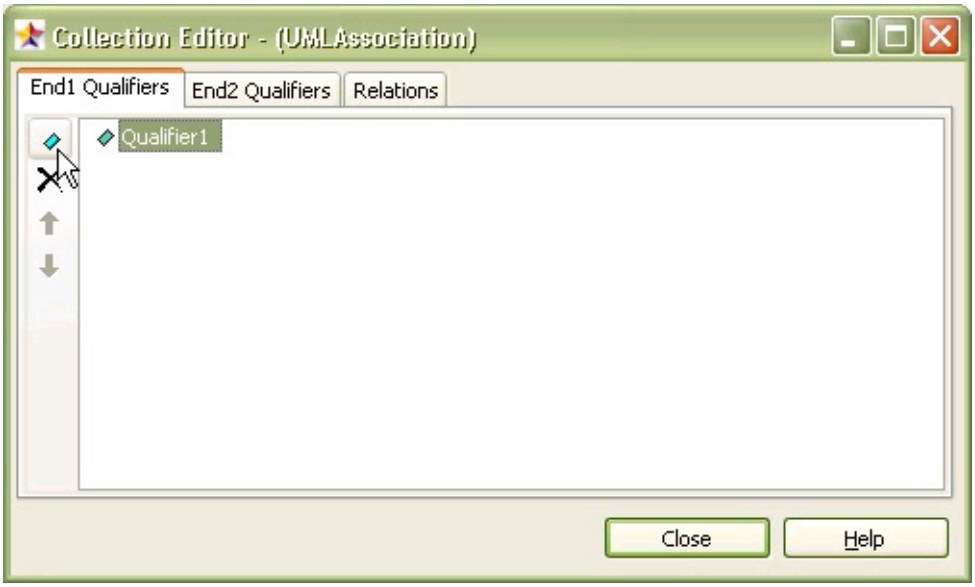
1. Select association's **[Collection Editor...]** popup menu.



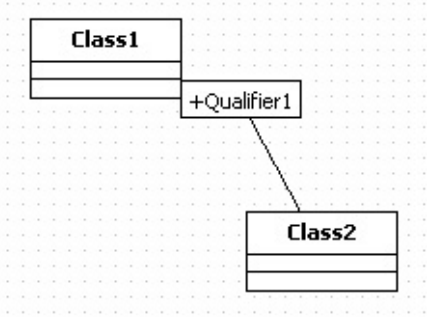
2. Or click ··· button in **[End.Qualifiers]** property on properties window.



3. At **[Qualifiers]** tab of the **[collection editor]**, you can add qualifier to the association by using button.



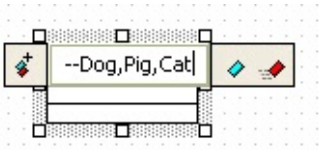
4. The result is as follows.



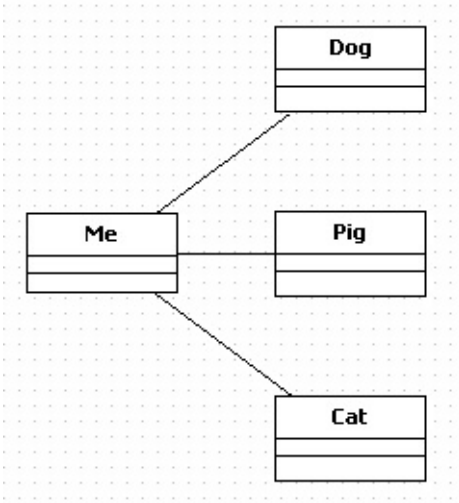
Procedure for creating multiple classes related to current class at once

If you want to create Dog, Pig, Cat classes related to Me class

1. Double-click Me class or press **[Enter]** key. At quick dialog, enter as following.



2. Then three classes with association are created as following.

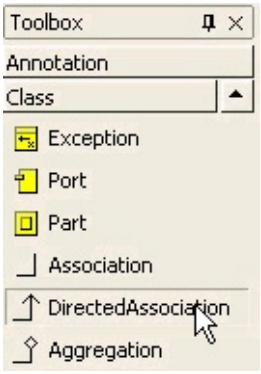

Dog
Me
Pig
Cat

DirectedAssociation

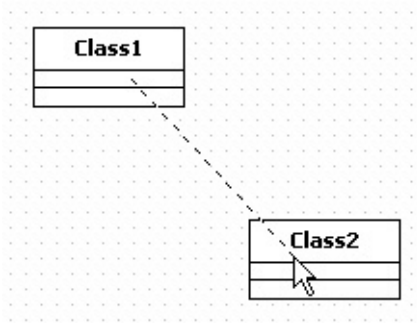
Procedure for creating directed association

Procedure for creating directed association is equal to association's.

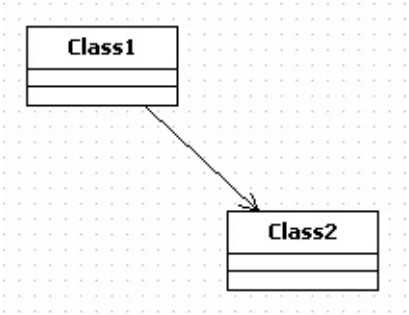
1. Click **[Toolbox] -> [Class] -> [DirectedAssociation]**.



2. Drag and drop between two elements in arrow direction.



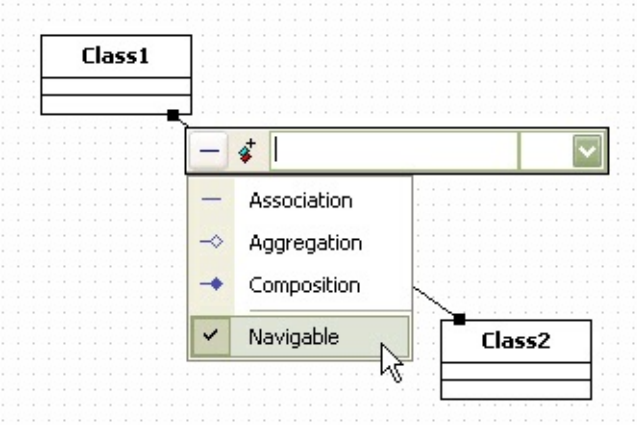
3. The result is as follows.



Procedure for changing association to directed association

To change association to directed association, click the arrow-opposite-side association end. At the quick dialog, uncheck navigable and association

becomes directed.

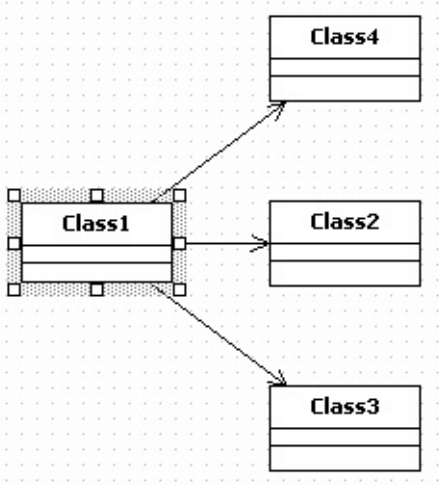


Procedure for creating element having directed association by shortcut creation syntax

In order to create element having directed association, use shortcut creation syntax,

1. Double-click element. At the quick dialog, enter name of elements that have directed association after "->" string and separate names with ",".

2. Press **[Enter]** key and multiple elements associated with selected element are created and arranged automatically.



Aggregation

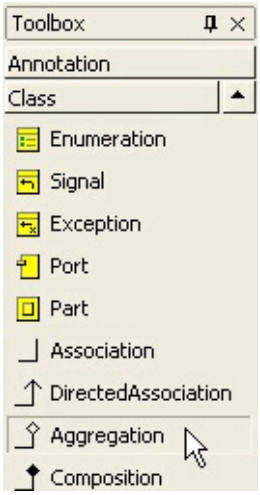
Semantics

An aggregate is a more specific type of association. The aggregate is signified by a hollow diamond on the point where the association connects with the classifier (association end). Aggregate adds the concept of whole and part to the 'vanilla' association. The classifier at the hollow diamond end is the whole.

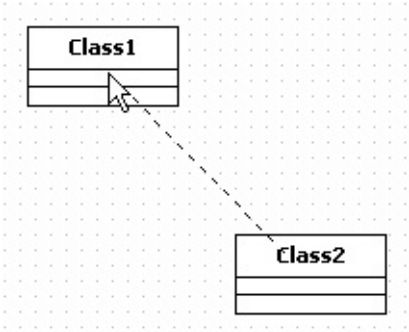
Procedure for creating aggreate

In order to create aggregation,

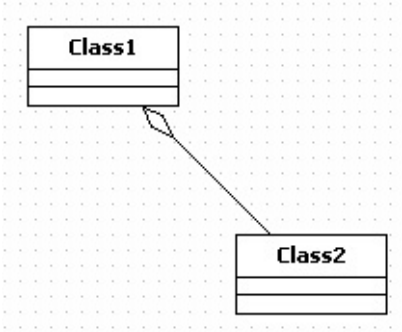
1. Click **[Toolbox]** -> **[Class]** -> **[Aggregation]** button.



2. Drag from one associated and drop to another in the **[main window]**.



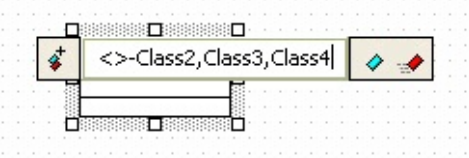
3. The result is as follows.



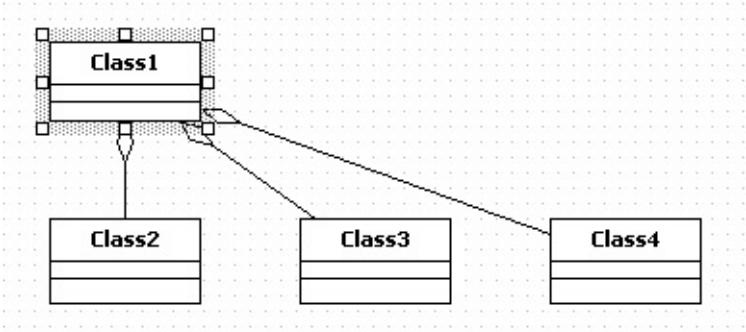
Procedure for creating aggregated class by shortcut creation syntax

In order to create class aggregated to selected class, use shortcut creation syntax.

1. Double-click to popup quick dialog. At the quick dialog, enter name of class aggregated to current class after "<>-" string and separate names with ",".



2. Press **[Enter]** key and classes aggregated to selected class are created and arranged automatically.



Composition

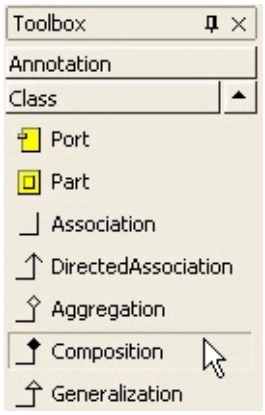
Semantics

A composite is a more specific type of association. The composite is signified by a filled diamond on the point where the association connects with the classifier (association end). Composite adds the concept of whole and part to the "vanilla" association and responsibility for the lifetime of the parts. The classifier at the filled diamond end is the whole.

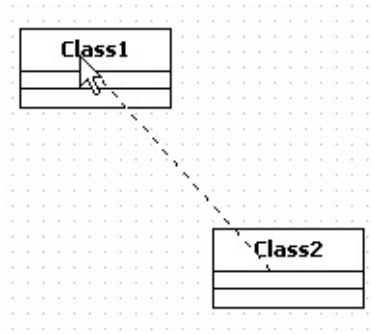
Procedure for creating composition

In order to create composition,

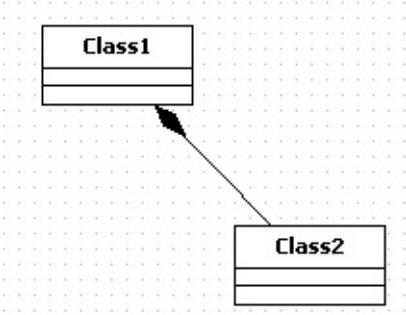
1. Click **[Toolbox]** -> **[Class]** -> **[Composition]** button.



2. Drag from one class and drop to another class composed in the **[main window]**.



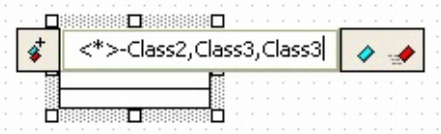
3. Between two classes, a new composition relationship is created as follows.



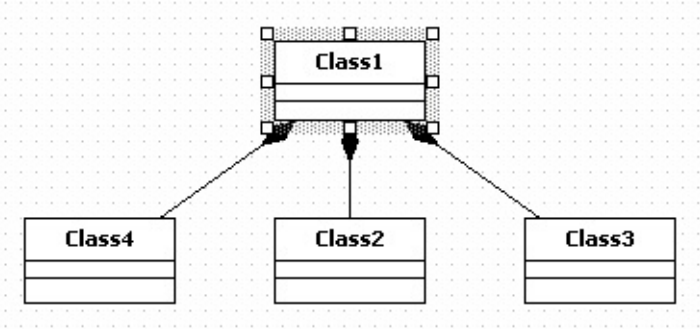
Procedure for creating composing class by shortcut creation syntax

In order to create class composing selected class, use shortcut creation syntax.

1. Double-click to popup quick dialog. At the quick dialog, enter name of class composing selected class after "<*>-" string and separate names with ",".



2. Press **[Enter]** key and classes composing selected class are created and arranged automatically.



Generalization

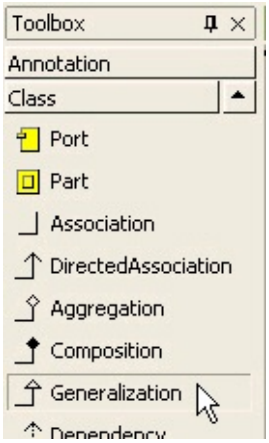
Semantics

Generalization is the taxonomic relationship between a more general element (the parent) and a more specific element (the child) that is fully consistent with the first element and that adds additional information. It is used for classes, packages, usecases, and other elements.

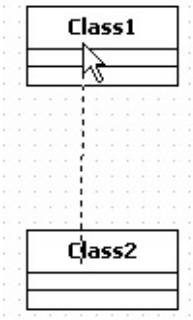
Procedure for creating generalization

In order to create generalization,

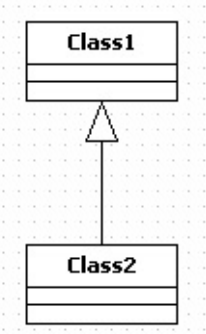
1. Click **[Toolbox]** -> **[Class]** -> **[Generalization]** button.



2. Drag from child element and drop to parent element in the **[main window]**.



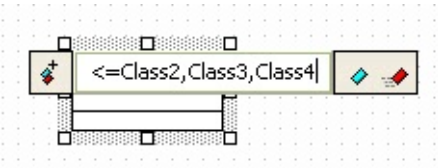
3. Then a new generalization is created.



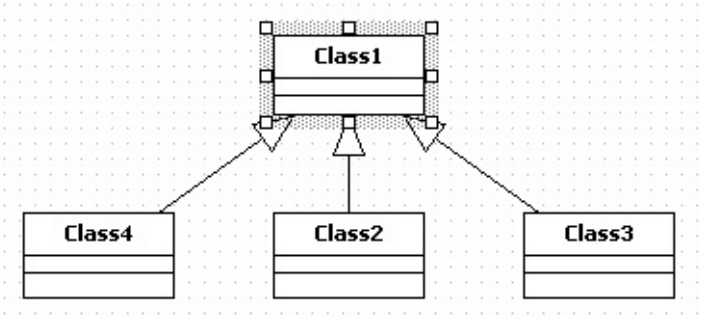
Procedure for creating multiple children classes at once.

In order to create multiple children classes inheriting selected class at once, use shortcut creation syntax.

1. Double-click to popup quick dialog. At the quick dialog, enter name of class inheriting selected class after "<=" string and separate names with ",".



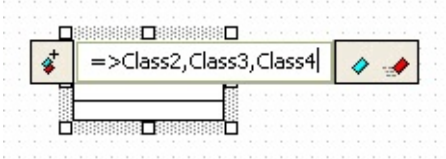
2. The children classes are created below selected class and arranged automatically.



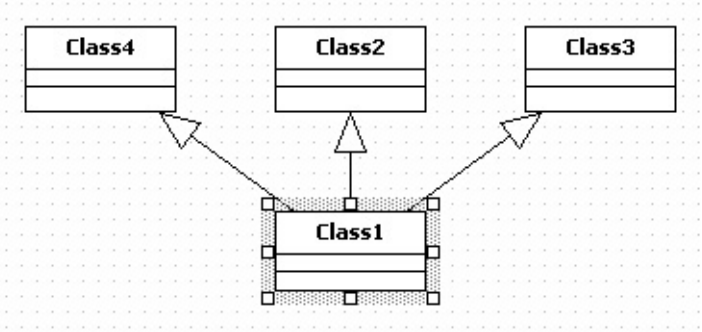
Procedure for creating multiple parent classes at once

In order to create multiple parent classes of selected class at once, use shortcut creation syntax.

1. Double-click to popup quick dialog. At the quick dialog, enter name of parent classes of selected class after "=>" string and separate names with ",".



2. The parent classes are created above selected class and arranged automatically.



Dependency

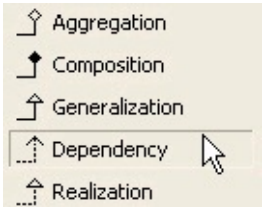
Semantics

A dependency indicates a semantic relationship between two model elements (or two sets of model elements). It relates the model elements themselves and does not require a set of instances for its meaning. It indicates a situation in which a change to the target element may require a change to the source element in the dependency.

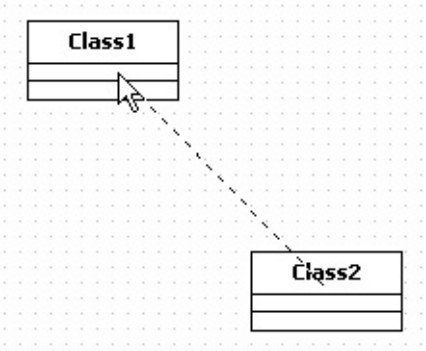
Procedure for creating dependency

In order to create dependency,

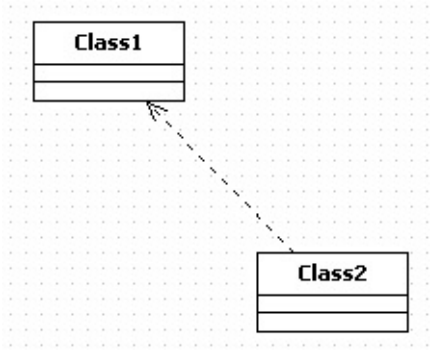
1. Click **[Toolbox]** -> **[Class]** -> **[Dependency]** button.



2. Drag and drop between elements in the **[main window]** in depending direction.



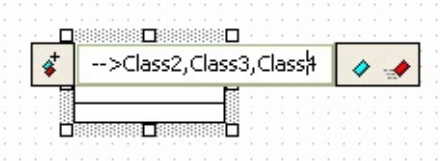
3. A new dependency between two classes is created.



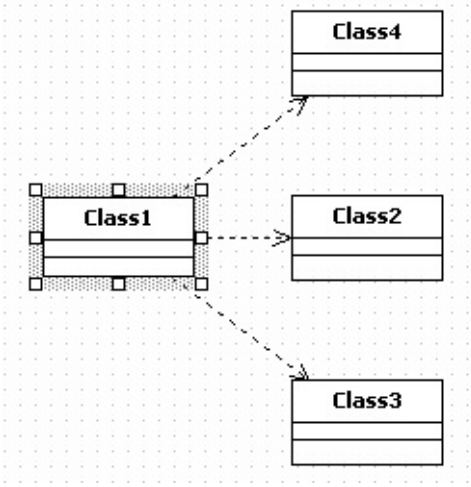
Procedure for dependent element by shortcut creation syntax

In order to create element depending by selected element, use shortcut creation syntax.

1. Double-click to popup quick dialog. At the quick dialog, enter name of dependent elements by selected element after "-->" string and separate names with ",".



2. Press **[Enter]** key and dependent elements by selected class are created and arranged automatically.



Realization

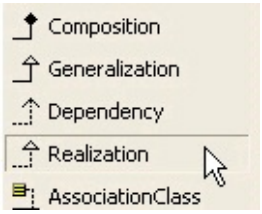
Semantics

A realization signifies that a relationship exists between a set of elements that form a specification (the client) and another set of elements that form the implementation (the supplier).

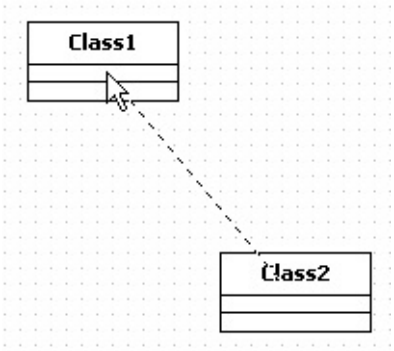
Procedure for creating realization

In order to create realization,

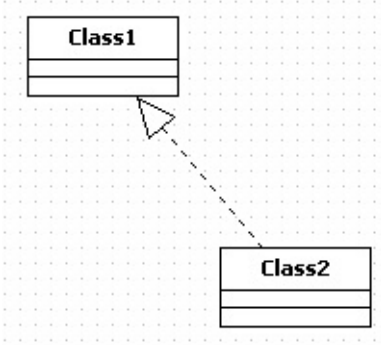
1. Click **[Toolbox]** -> **[Class]** -> **[Realization]** button.



2. Drag and drop between elements in the **[main window]** in realization direction.



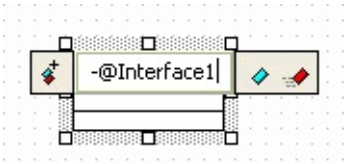
3. The result is as follows.



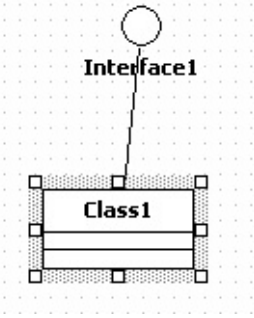
Procedure for creating realization target element of selected element

In order to create target interface element of selected element, use shortcut creation syntax.

1. Double-click to popup quick dialog. At the quick dialog, enter name of interface elements of selected element after "-@" string and separate names with ",".



2. Press **[Enter]** key and interface elements of selected element are created and arranged automatically.



AssociationClass

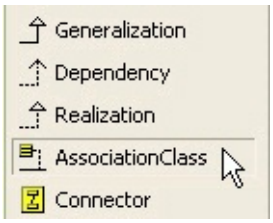
Semantics

An association class is an association that is also a class. It not only connects a set of classifiers but also defines a set of features that belong to the relationship itself and not any of the classifiers.

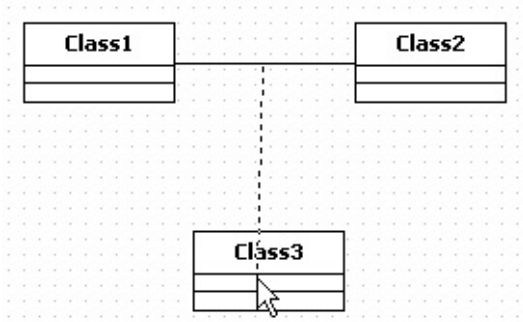
Procedure for creating association class

In order to create association class,

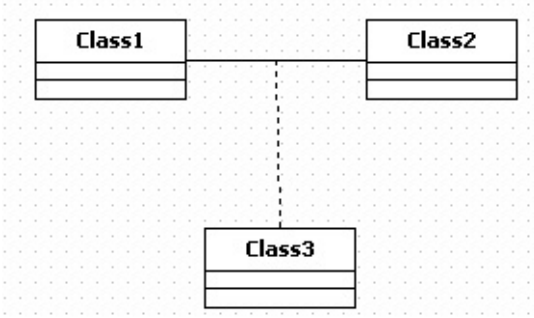
1. Click **[Toolbox] -> [Class] -> [AssociationClass]** button.



2. Drag from association and drop to the class as association class in the **[main window]**.



3. The result is as follows.

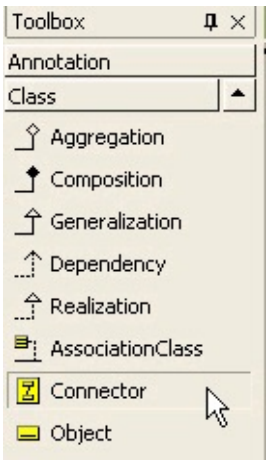


Connector

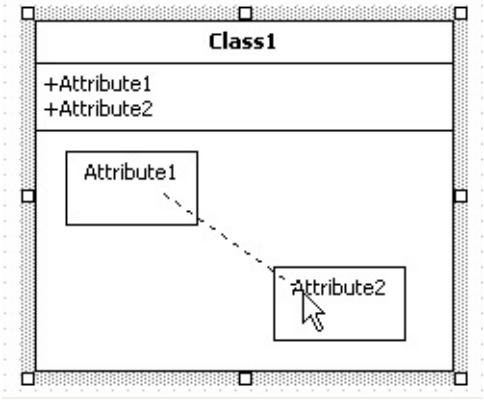
Procedure for creating connector

In order to create connector,

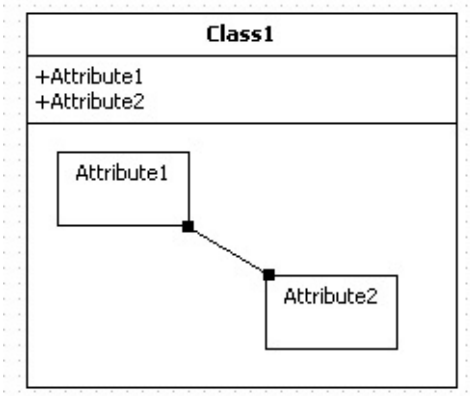
1. Click **[Toolbox]** -> **[Class]** -> **[Connector]** button.



2. Drag from one part and drop to the other part in the **[main window]**.



3. Between two parts, new connector is created finally.


Class1
+Attribute1
+Attribute2
Attribute1
Attribute2

Object

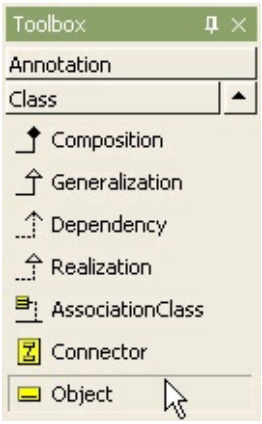
Semantics

An object represents a particular instance of a class. It has identity and attribute values. A similar notation also represents a role within a collaboration because roles have instance-like characteristics.

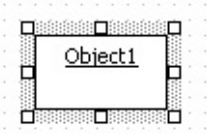
Procedure for creating object

In order to create object,

1. Click **[Toolbox]** -> **[Class]** -> **[Object]** button.



2. And click at the position where object will be placed in the **[main window]**.

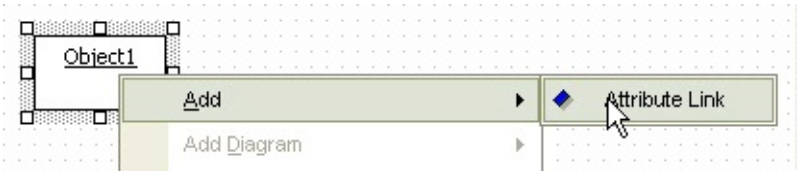


Procedure for adding AttributeLink to object

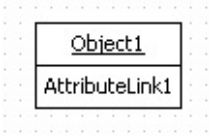
There are two way to add AttributeLink to Object.

- using object model in the **[main window]** or the **[model explorer]**
- using **[collection editor]**

In the case of using object model, select object in the **[main window]** or in the **[model explorer]**, right-click the selected object, select **[Add]** -> **[Attribute Link]** popup menu, and you can add Attribute Link.



In the other case, select **[Collection Editor...]** popup menu of object or click ... button in slots property on properties window. At **[Slots]** tab of the **[collection editor]**, you can add attribute link by using button.



Link

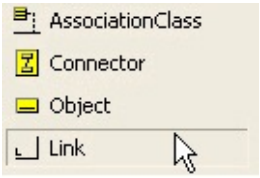
Semantics

A link is a tuple (list) of object references. Most commonly, it is a pair of object references. It is an instance of an association.

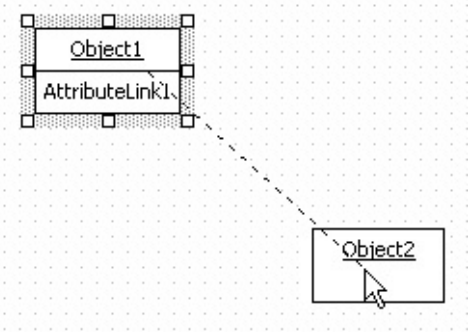
Procedure for creating link

In order to create Link,

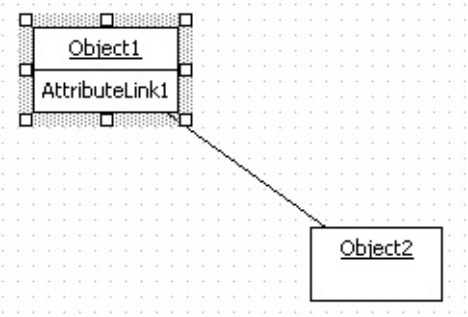
1. Click **[Toolbox]** -> **[Class]** -> **[Link]** button.



2. Drag from one Object and drop to the other Object in the **[main window]**.



3. The result is as follows.



Modeling with Statechart Diagram

The following elements are available in a statechart diagram.

- State
- SubmachineState
- InitialState
- FinalState
- JunctionPoint
- ChoicePoint
- ShallowHistory
- DeepHistory
- Synchronization
- Flow Final
- Transition
- SelfTransition

State

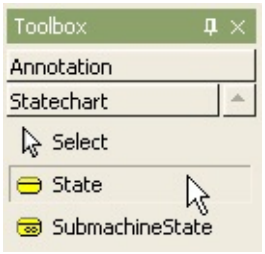
Semantics

A state is a condition during the life of an object or an interaction during which it satisfies some condition, performs some action, or waits for some event.

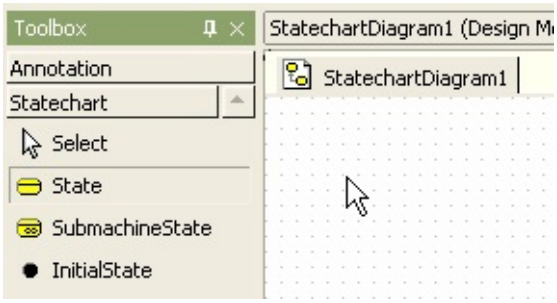
Procedure for creating state

In order to create State,

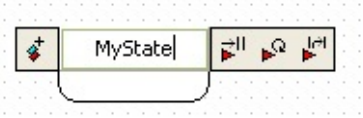
1. Click **[Toolbox]** -> **[Statechart]** -> **[State]** button.



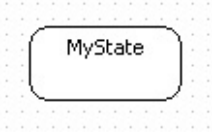
2. And click at the position where State will be placed in the **[main window]**.



3. A state is created and quick dialog appears. Enter the state name at the quick dialog .



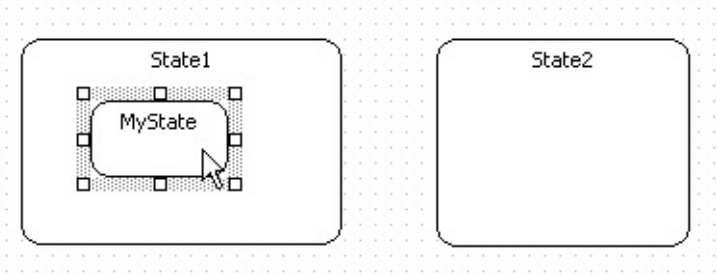
4. And press **[Enter]** key to have done this procedure.



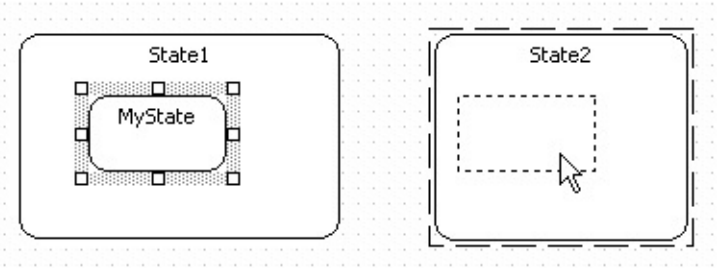
Procedure for moving state into another state

In order to move a state into another state,

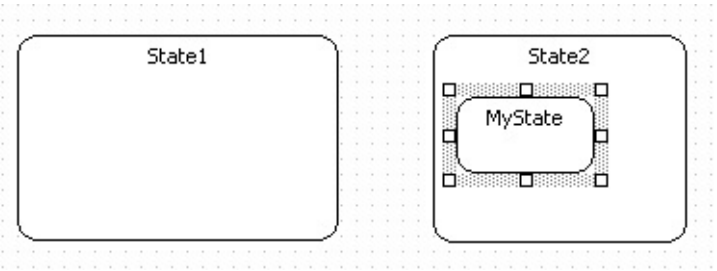
1. Click a state that is contained in some state.



2. Drag it into another state.



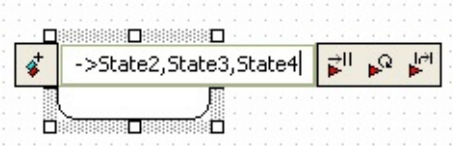
3. The selected state is move into another state.



Procedure for transiting to multiple states at once

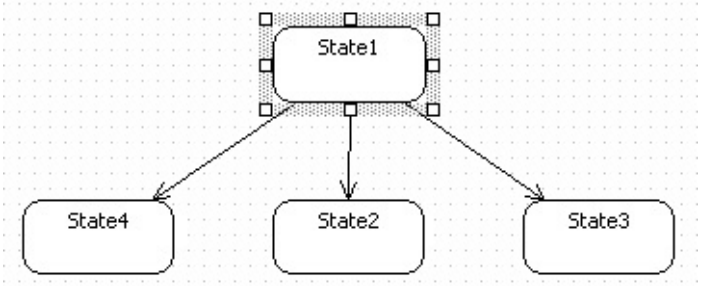
In order to create states with incoming or outgoing transition from selected state at once, use shortcut creation syntax.

1. Double-click state. Then quick dialog is shown. At the quick dialog, After "->" string(or "<-" string for incoming), enter target state names, and separate state names by "," character.



And press **[Enter]** key. Several states outgoing(incoming) from selected

2. state are created and arranged automatically.



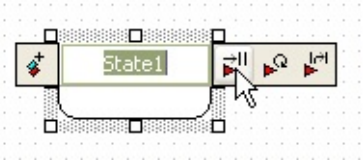
Procedure for adding entry/do/exit action

There are three way to add action to state.

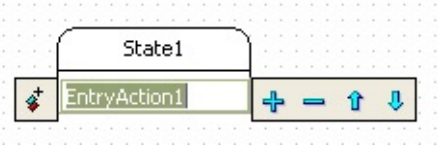
- using quick dialog
- using model in the **[main window]** or the **[model explorer]**
- using **[collection editor]**

In the case of using quick dialog,

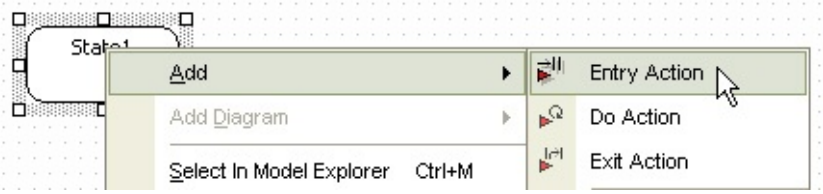
1. Double-click state.
2. At the quick dialog, press **[Add Entry/Add DoAction/Add ExitAction]** button at the quick dialog.



3. And you can add action.

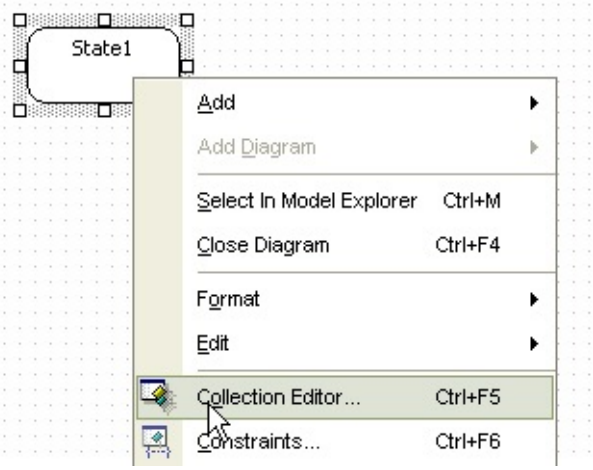


In the case of using model, select state in the **[main window]** or in the **[model explorer]**. Right-click the selected state, select **[Add] -> [Entry/Do/Exit]** popup menu. And you can do.

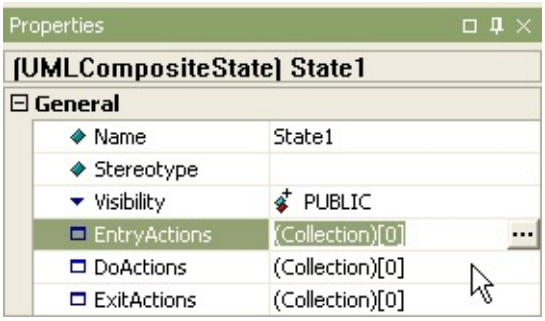


In the last case,

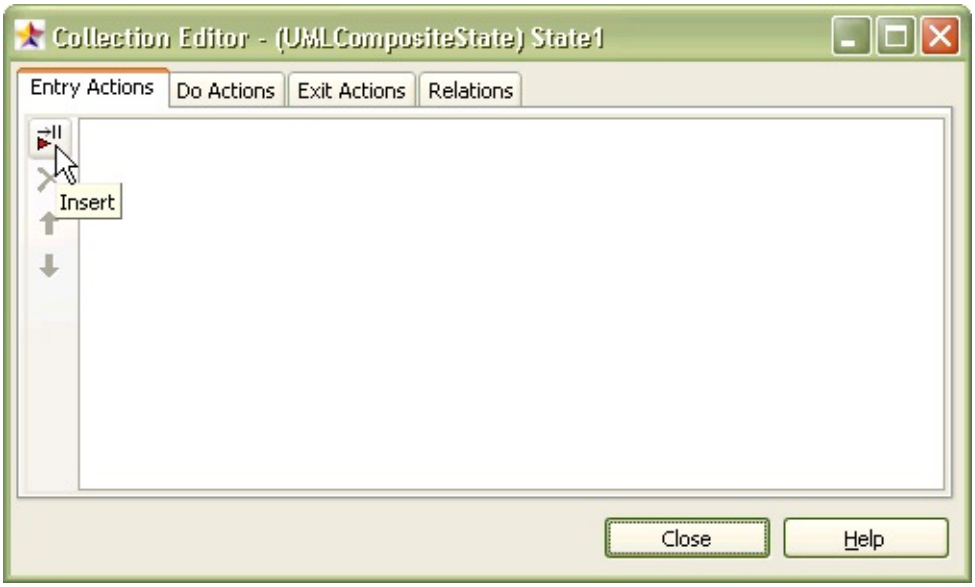
1. select **[Collection Editor...]** popup menu of state.



2. Or click ··· button in **[EntryActions/DoActions/ExitActions]** property on properties window.



3. At **[Entry Actions/Do Actions/Exit Actions]** tab of the **[collection editor]**, you can add action by using button.


Collection Editor - (UMLCompositeState) State1
Entry Actions
Do Actions
Exit Actions
Relations
Insert
Close
Help

SubmachineState

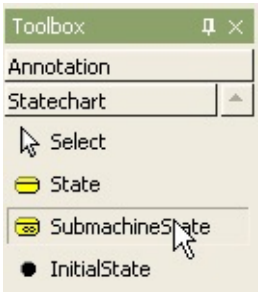
Semantics

A submachine state is a syntactical convenience that facilitates reuse and modularity. It is a shorthand that implies a macro-like expansion by another state machine and is semantically equivalent to a composite state.

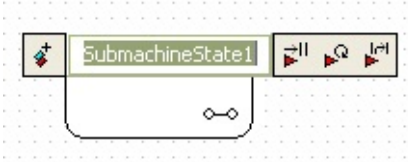
Procedure for creating submachine state

In order to create SubmachineState,

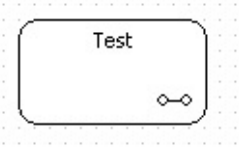
1. Click **[Toolbox] -> [Statechart] -> [SubmachineState]** button.



2. And click at the position where SubmachineState will be placed in the **[main window]**. A submachine state is created and quick dialog is opened.



3. At the quick dialog, enter the submachine state name and press **[Enter]** key.



InitalState

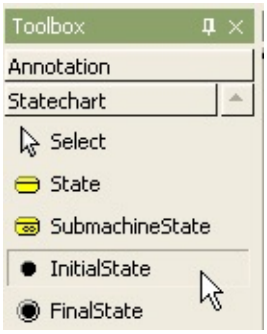
Semantics

An initial is a kind of pseudostate that represents the starting point in a region of a state machine. It has a single outgoing transition to the default state of the enclosing region, and has no incoming transitions. There can be one (and only one) initial state in any given region of a state machine. It is not itself a state but acts as a marker.

Procedure for creating initial state

In order to create InitialState,

1. Click **[Toolbox] -> [Statechart] -> [InitialState]** button.

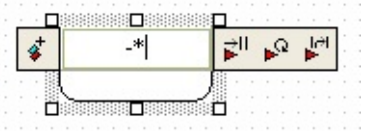


2. And click at the position where InitialState will be placed in the **[main window]**.

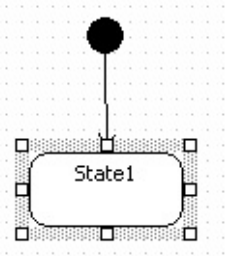
Procedure for creating initial state from state

In order to create initial state with outgoing transition to selected object, use shortcut creation syntax.

1. Double-click state. At the quick dialog, After "-*" string, enter initial state name or none.



2. Press **[Enter]** key and initial state with outgoing transition to selected state is created.



FinalState

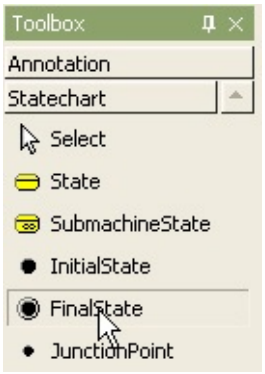
Semantics

A final state represents the last or "final" state of the enclosing composite state. There may be more than one final state at any level signifying that the composite state can end in different ways or conditions. When a final state is reached and there are no other enclosing states it means that the entire state machine has completed its transitions and no more transitions can occur.

Procedure for creating final state

In order to create FinalState,

1. Click **[Toolbox] -> [Statechart] -> [FinalState]** button.

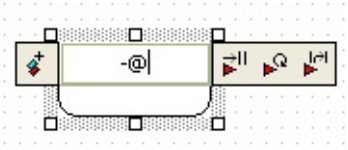


2. And click at the position where FinalState will be placed in the **[main window]**.

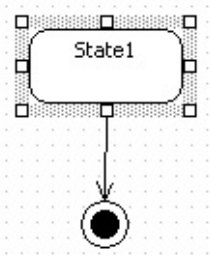
Procedure for creating final state from state

In order to create final state with outgoing transition from selected object, use shortcut creation syntax.

1. Double-click state. At the quick dialog, After "-@" string, enter final state name or none.



2. Press **[Enter]** key and final state with ingoing transition from selected state is created.



JunctionPoint

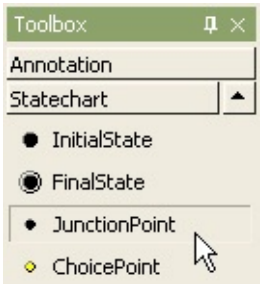
Semantics

JunctioPoint chains together transitions into a single run-to-completion path. May have multiple input and/or output transitions. Each complete path involving a junction is logically independent and
only one such path fires at one time. May be used to construct branches and merges.

Procedure for creating junciton point

In order to create JunctionPoint,

1. Click **[Toolbox]** -> **[Statechart]** -> **[JunctionPoint]** button.



2. And click at the position where JunctionPoint will be placed in the **[main window]**.

ChoicePoint

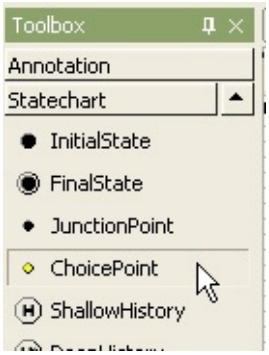
Semantics

ChoicePoint splits an incoming transition into several disjoint outgoing transitions. Each outgoing transition has a guard condition that is evaluated after prior actions on the incoming path have been completed. At least one outgoing transition must be enabled or the model is ill formed.

Procedure for creating choice point

In order to create ChoicePoint,

1. Click **[Toolbox] -> [Statechart] -> [ChoicePoint]** button.



2. And click at the position where ChoicePoint will be placed in the **[main window]**.

ShallowHistory

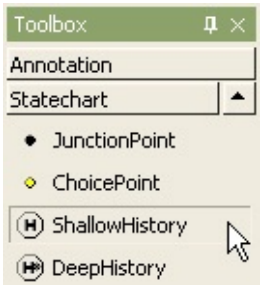
Semantics

When reached as the target of a transition, shallow history restores the state within the enclosing composite state that was active just before the enclosing state was last exited. Does not restore any substates of the last active state.

Procedure for creating shallow history

In order to create ShallowHistory,

1. Click **[Toolbox]** -> **[Statechart]** -> **[ShallowHistory]** button.



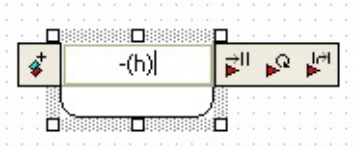
2. And click at the position where ShallowHistory will be placed in the **[main window]**.

State History State :

Procedure for creating final state from state

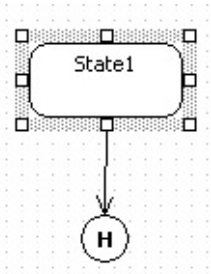
In order to create history with outgoing transition from selected object, use shortcut creation syntax.

1. Double-click state. At the quick dialog, enter one of "-(h)", "-(H)", "-(h*)", "-(H*)" string.



2. Press **[Enter]** key and history with outgoing transition from selected state is

created.



DeepHistory

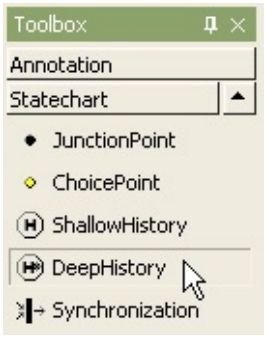
Semantics

When reached as the target of a transition, deep history restores the full state configuration that was active just before the enclosing composite state was last exited.

Procedure for creating deep history

In order to create DeepState,

1. Click **[Toolbox]** -> **[Statechart]** -> **[DeepState]** button.



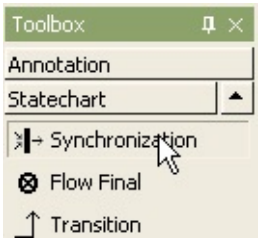
2. And click at the position where DeepState will be placed in the **[main window]**.

Synchronization

Procedure for creating synchronization bar

In order to create Synchronization,

1. Click **[Toolbox] -> [Statechart] -> [Synchronization]** button.

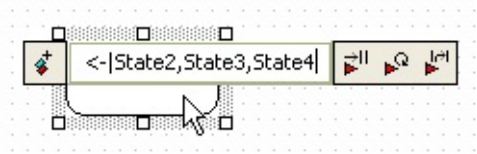


2. And click at the position where Synchronization will be placed in the **[main window]**.

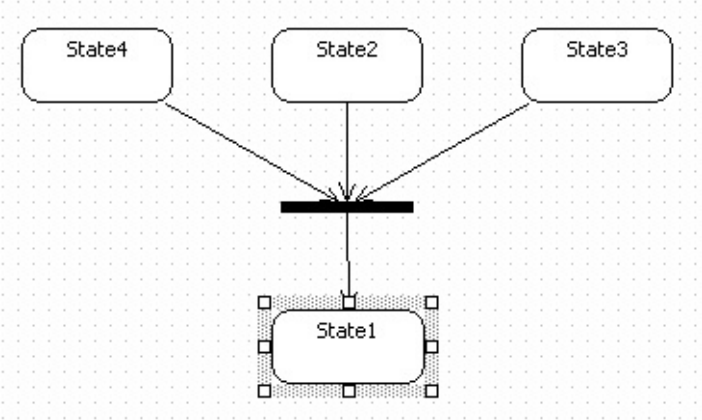
Procedure for creating join

In order to create incoming join transition to selected object, use shortcut creation syntax.

1. Double-click state. At the quick dialog, enter "<-|" and state names to be joined, and separate state names by "," character.



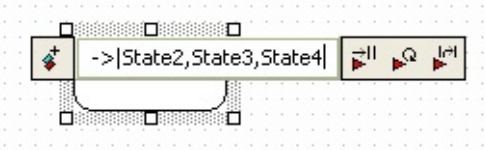
2. Press **[Enter]** key and states joined to selected state is created and arranged automatically.



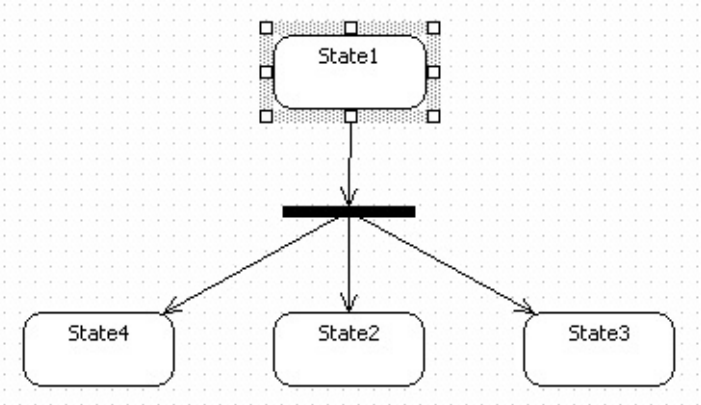
Procedure for creating join

In order to create outgoing fork transition to selected object, use shortcut creation syntax.

1. Double-click state. At the quick dialog, enter "->|" and state names to be forked, and separate state names by "," character.



2. Press **[Enter]** key and states forked from selected state is created and arranged automatically.

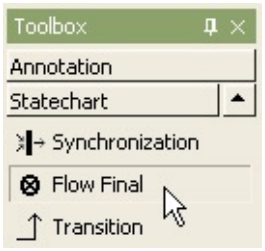


Flow Final

Procedure for creating flow final

In order to create Flow Final,

1. Click **[Toolbox] -> [Statechart] -> [Flow Final]** button.



2. And click at the position where Flow Final will be placed in the **[main window]**.

Transition

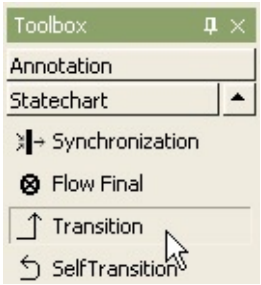
Semantics

A transition is a directed relationship between a source state vertex and a target state vertex. It may be part of a compound transition, which takes the state machine from one state configuration to another, representing the complete response of the state machine to a particular event instance.

Procedure for creating transition

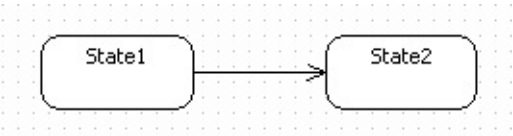
In order to create Transition,

1. Click **[Toolbox]** -> **[Statechart]** -> **[Transition]** button.



2. Drag and drop between states in transition direction in the **[main window]**.

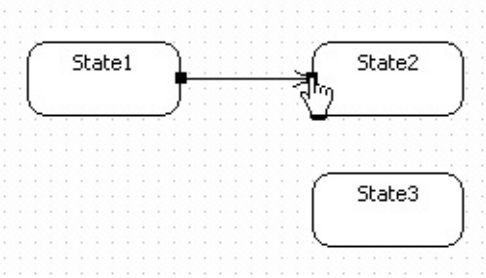
3. Between two states, a transition is created.



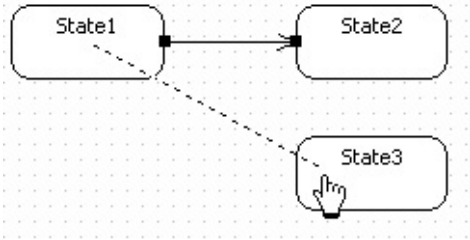
Procedure for reconnecting to another element

In order to reconnect to another state,

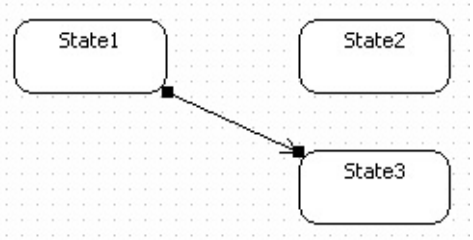
1. Click the end of transition.



2. Drag and drop it into another state.



3. Then transition's end will be changed.

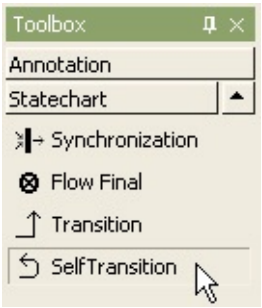


SelfTransition

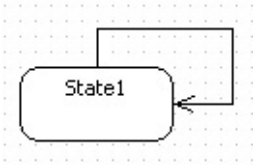
Procedure for creating self-transition

In order to create self-transition,

1. Click **[Toolbox]** -> **[Statechart]** -> **[SelfTransition]** button.



2. Click state to have self-transition in the **[main window]**.



Modeling with Sequence Diagram

The following elements are available in a sequence diagram.

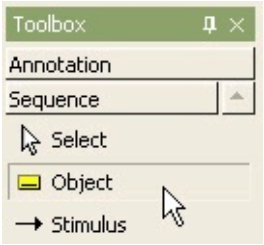
- Object
- Stimulus
- SelfStimulus
- Combined Fragment
- Frame
- Diagram

Object

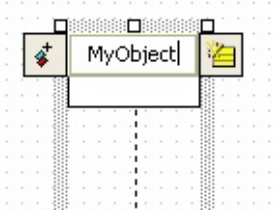
Procedure for creating object

In order to create object,

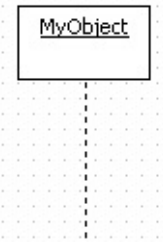
1. Click **[Toolbox]** -> **[Sequence]** -> **[Object]** button.



2. And click at the position where object will be placed in the **[main window]**.
3. Object quick dialog is shown. At the quick dialog, enter the object name.



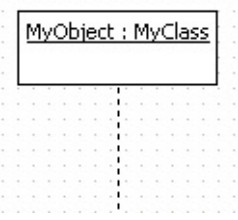
4. Press **[Enter]** key.



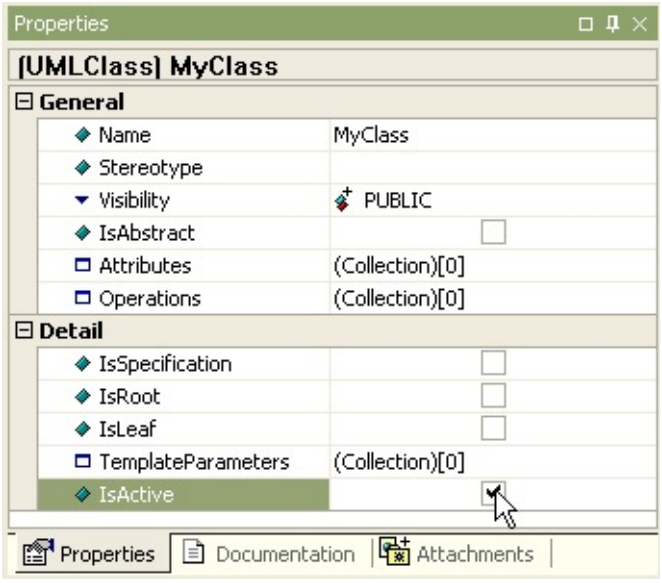
Procedure for setting active object

In order to set class to active object,

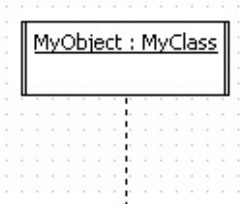
1. Set assigned class's **[IsActive]** property to true.



2. For MyObject, change MyClass's IsActive property.



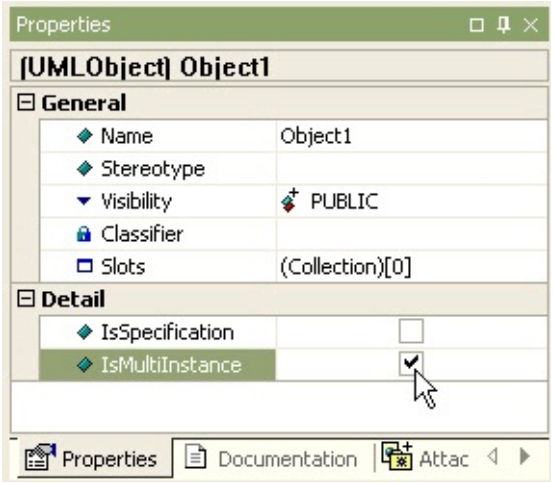
3. If class property is not assigned, you can't change object to active object. The result is as follows.



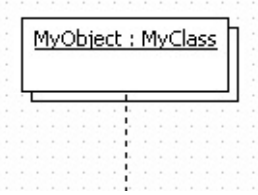
Procedure for setting to multi object

In order to set object to multi object,

1. Set object's **[IsMultiInstance]** property to true.



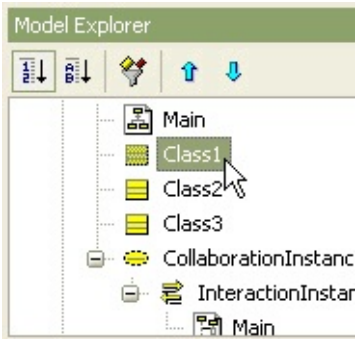
2. Then the object is changed to multi object.



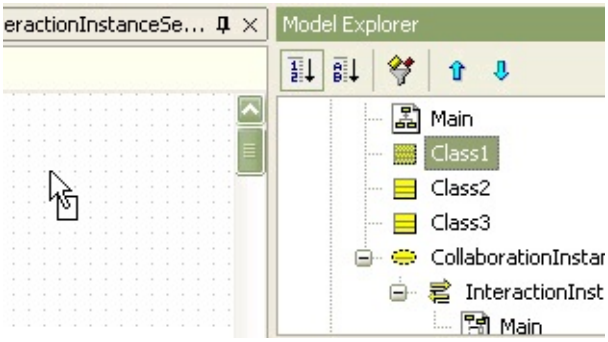
Procedure for creating object from class

In order to create object from class,

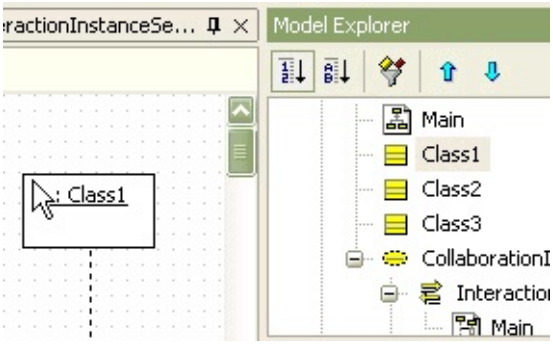
1. Select class in the **[model explorer]**.



2. Drag and drop it into **[main window]**.



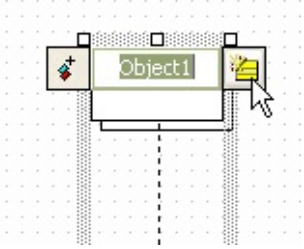
3. Finally, a object is created on the diagram.



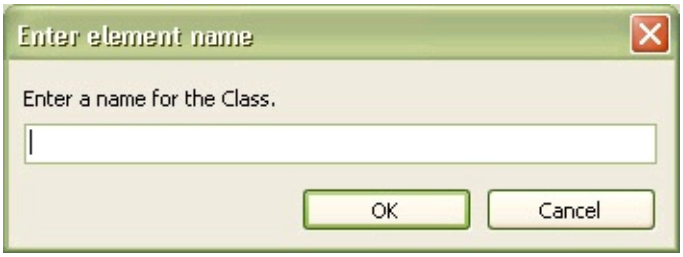
Procedure for creating class from object

If class is not assigned to object,

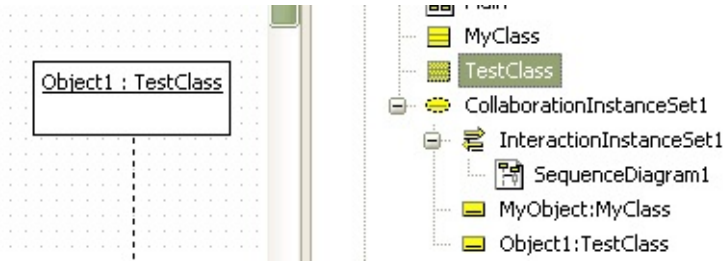
1. Double-click object to pop up quick dialog, click add class button



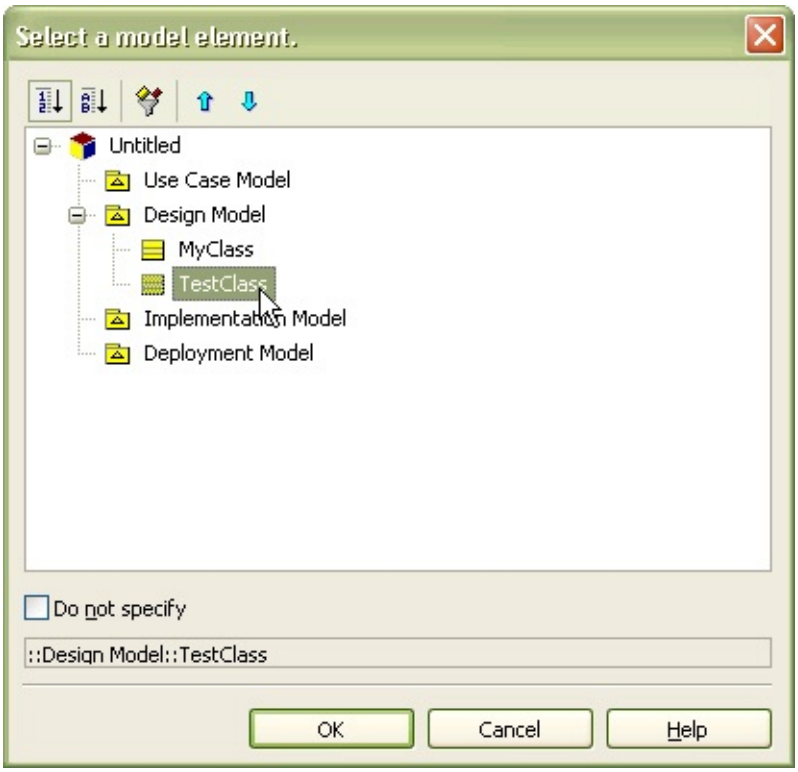
2. At the [Enter element name] dialog, enter the new class name.



3. And new class is created and assigned to object.



4. If you want existing class to be assigned to object, click ··· button in object's classifier property, and select class to be assigned to object at the **[Select a model element]** dialog.

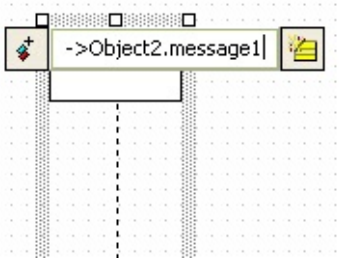


Procedure for creating outgoing from object stimulus by using shortcut creation syntax

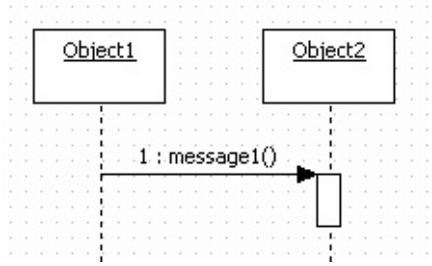
In order to create outgoing stimulus from selected object to another object,

1. Double-click from-object, or select from-object and press **[Enter]** key to pop up quick dialog.

2. At the quick dialog, enter stimulus name after "->" string ("<-" string for incoming and "<->" for outgoing with return).



3. Press **[Enter]** key and outgoing stimulus from selected object to target object is created and placed at the last order.


Object1
Object2
1 : message1()

Stimulus

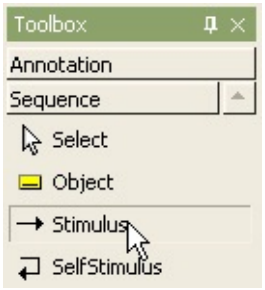
Semantics

A Stimulus is a communication between two Instances that conveys information with the expectation that action will ensue. A Stimulus will cause an Operation to be invoked, raise a Signal, or cause an Instance to be created or destroyed.

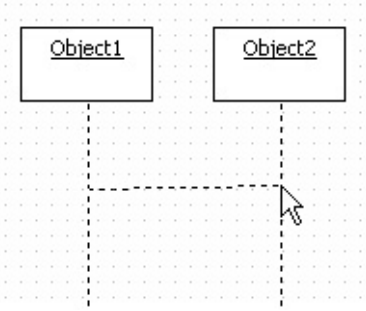
Procedure for creating stimulus

In order to create stimulus,

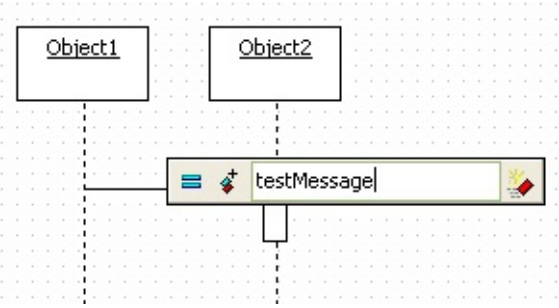
1. Click **[Toolbox] -> [Sequence] -> [Stimulus]** button.



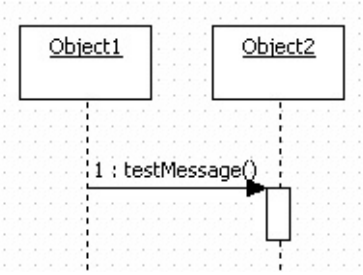
2. Drag from one object, and drop to the other(object or lifeline) in the **[main window]** in outgoing direction.



3. Stimulus quick dialog is opened. Enter the stimulus name at the quick dialog and press **[Enter]** key.



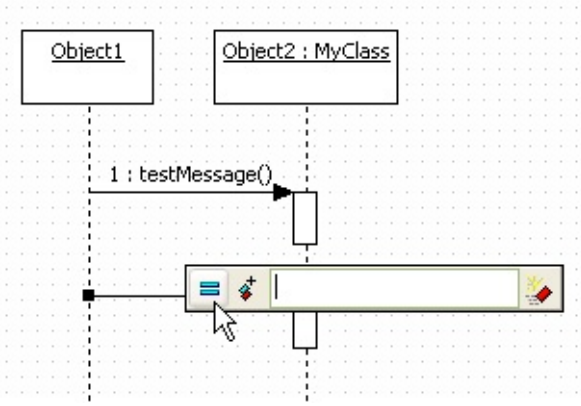
4. Finally, a stimulus is created as follows.



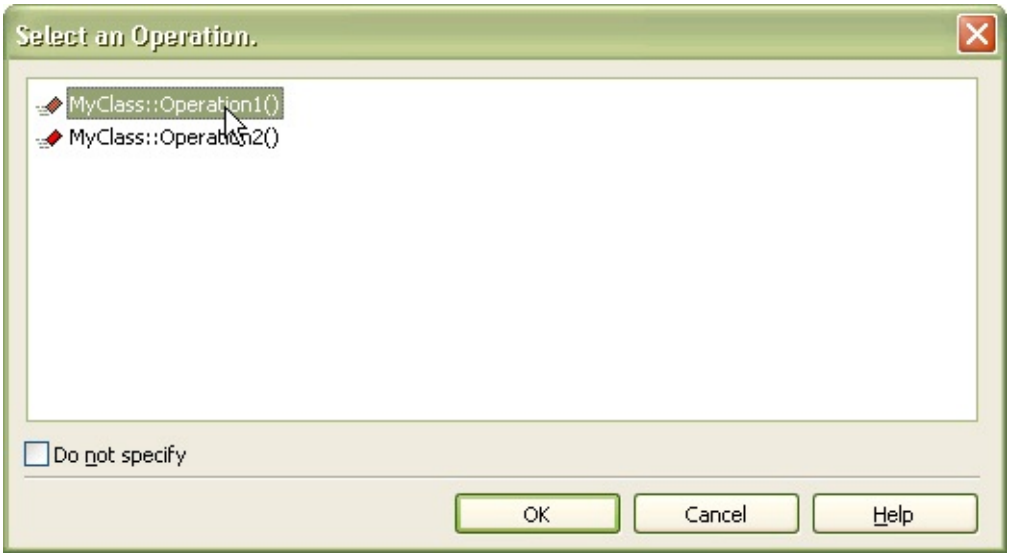
Procedure for using operation in class as stimulus

If classifier property of receiver(object) of stimulus is assigned and you want to assign operation to stimulus,

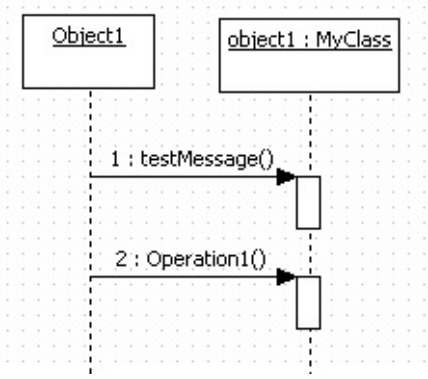
1. Double-click stimulus
2. Click button at the quick dialog.



3. Select operation on the **[Select an operation]** dialog, and click **[OK]** button.



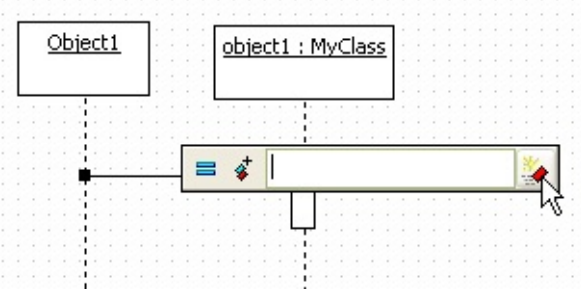
4. New stimulus mapped to class's operation is added as follows.



Procedure for creating operation of class from object

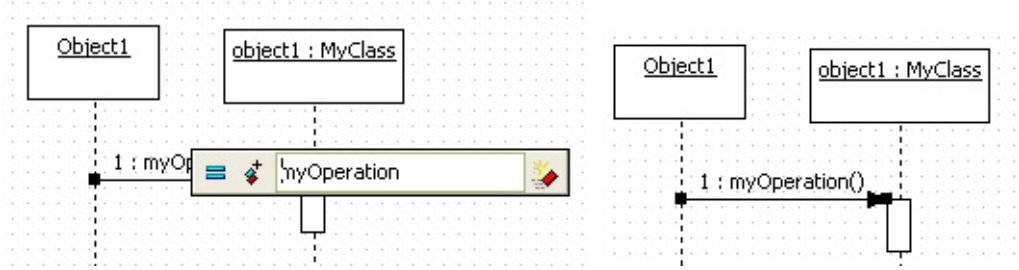
To create operation of class as stimulus's receiver from object and assign it to stimulus,

1. Double-click stimulus, click  button at the quick dialog.

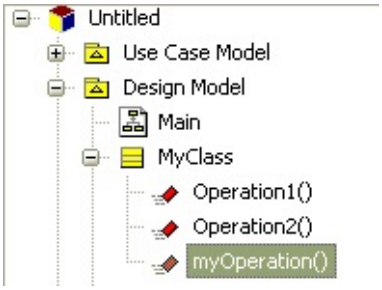


2. Enter new operation name to be created, and click **[OK]** button.

3. New operation is added to the class and text is filled at the quick dialog (This procedure is valid when there exists assigned class.). Press **[Enter]** key.



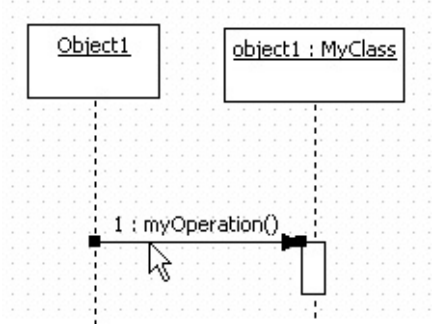
4. See **[model explorer]** to confirm creation of new operation.



Procedure for creating previous stimulus of current stimulus by using shortcut creation syntax

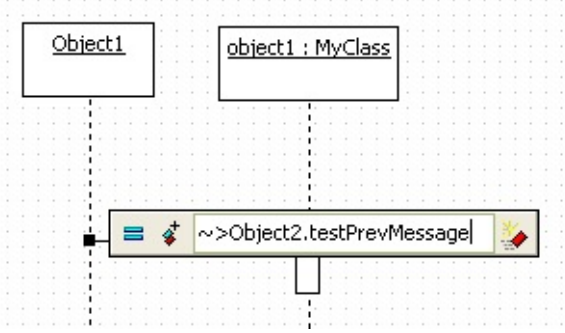
In order to create previous stimulus to current stimulus,

1. Double-click a stimulus, or select a stimulus and press **[Enter]** key.

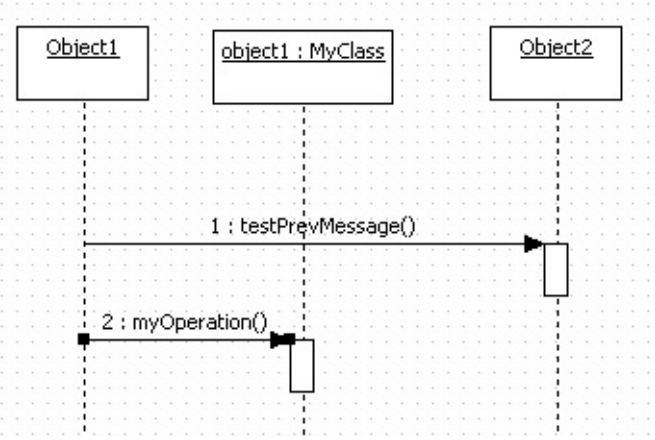


2. At the quick dialog, After "~>" string("<~" for incoming stimulus), enter

target object name and stimulus name.



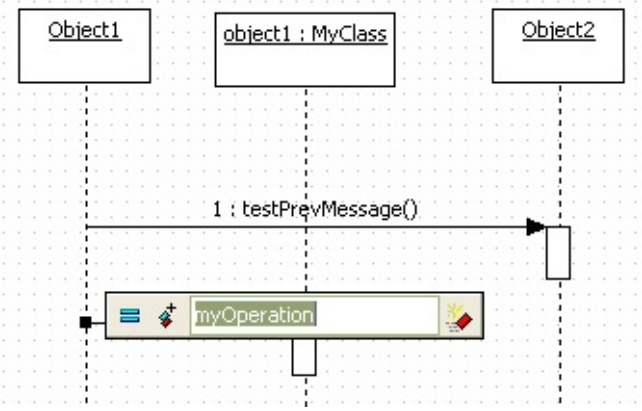
3. Press **[Enter]** key, and then new object and stimulus are created and arranged above selected stimulus.



Procedure for creating next stimulus to current stimulus by using shortcut creation syntax

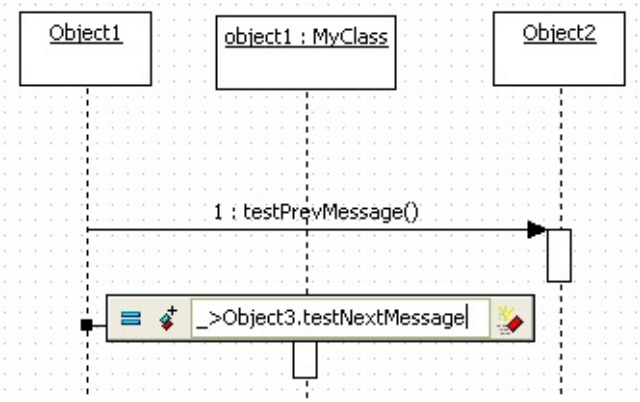
In order to create next stimulus to selected stimulus,

1. Double-click a stimulus, or select a stimulus and press **[Enter]** key.

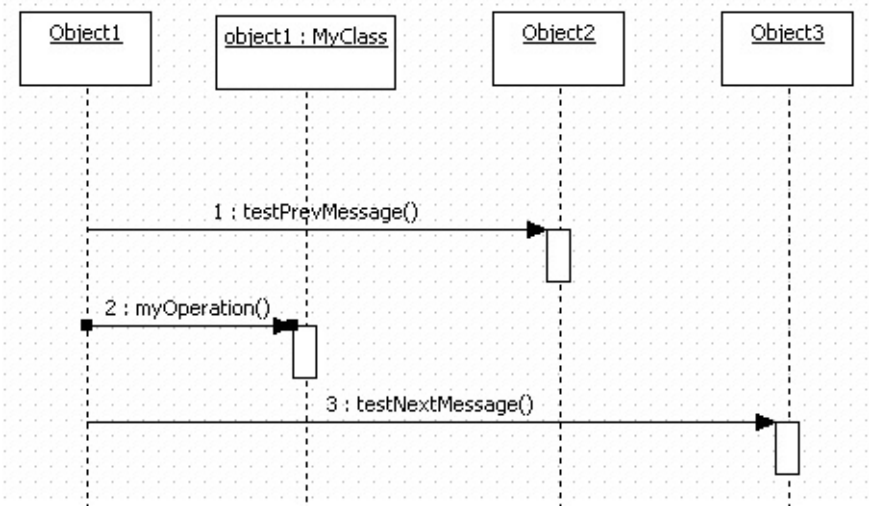


2. At the quick dialog, After "_>" string("<_" for incoming stimulus), enter

target object name and stimulus name.



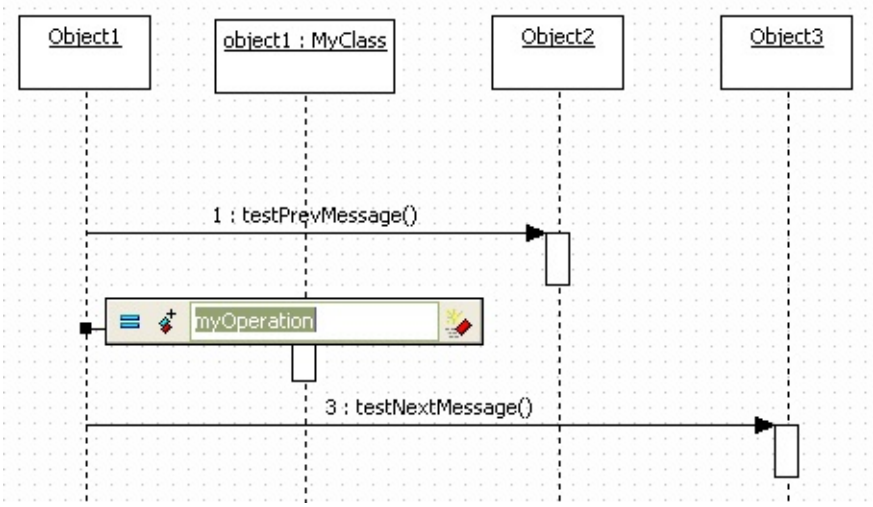
3. Press **[Enter]** key, and then new object and stimulus are created and arranged next to selected stimulus.



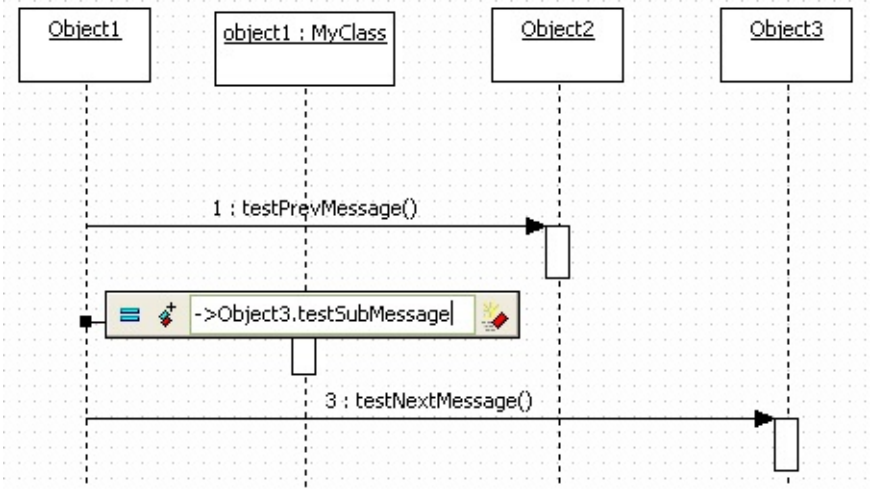
Procedure for creating sub stimulus by using shortcut creation syntax

In order to create a sub stimulus of selected stimulus,

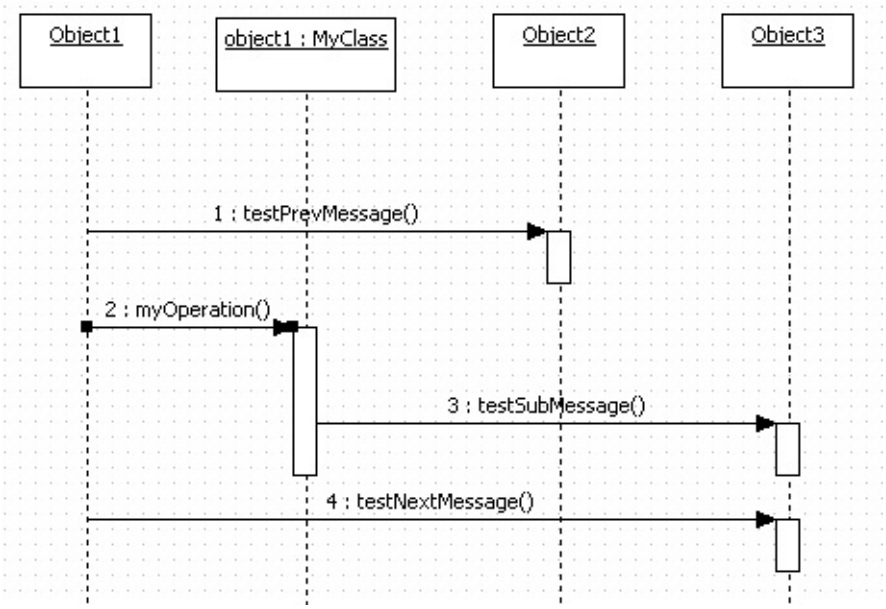
1. Double-click a stimulus, or select a stimulus and press **[Enter]** key.



2. At the quick dialog, After "->" string("<-" for incoming stimulus), enter target object name and sub stimulus name.



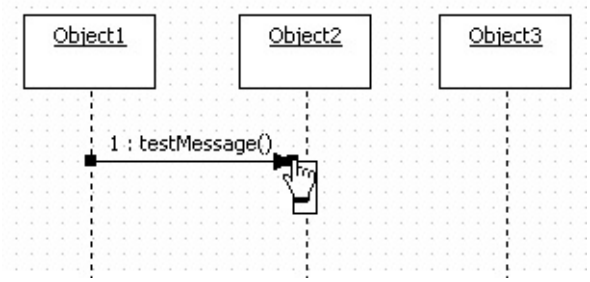
3. Press **[Enter]** key, and then new object and stimulus are created and arranged on the bottom of selected stimulus's activation.



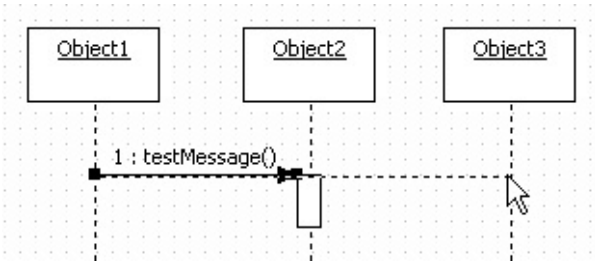
Procedure for reconnecting to another object

In order to reconnect stimulus to another object,

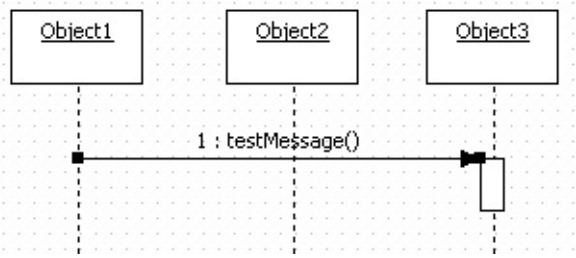
1. Click the end of stimulus.



2. Drag the end of stimulus and drop it to another object.

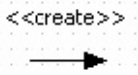
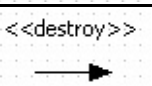


3. Then stimulus will be connected to another object.



Procedure for changing ActionKind of stimulus

The **[ActionKind]** property of stimulus should be assigned to one of five sort as following. To change **[ActionKind]** property, select stimulus and select the **[ActionKind]** property on the properties window.

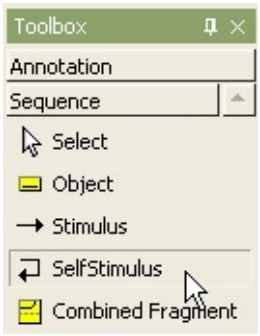
ActionKind	Shape
CALL	
SEND	
RETURN	
CREATE	
DESTROY	

SelfStimulus

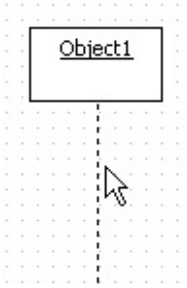
Procedure for creating self-stimulus

In order to create self-stimulus,

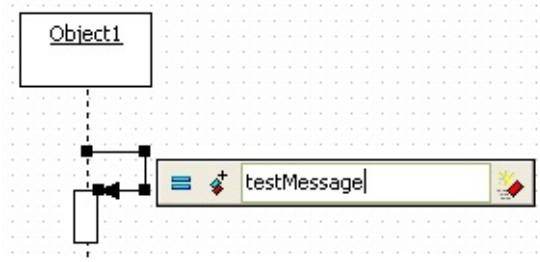
1. Click **[Toolbox] -> [Sequence] -> [SelfStimulus]** button.



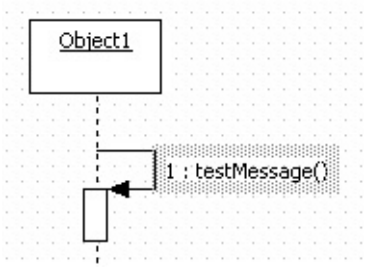
2. And click the object(or lifeline) that self-stimulus will be placed in the **[main window]**.



3. Object quick dialog is opened. At the quick dialog, enter the stimulus name and press **[Enter]** key.



4. The result of procedure is as follows. You may arrange stimulus position to reduce overlapping of text and line.

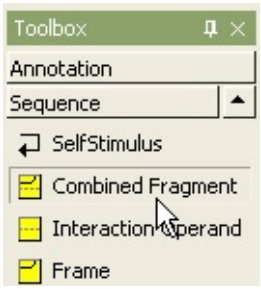

Object1
1 : testMessage()

Combined Fragment

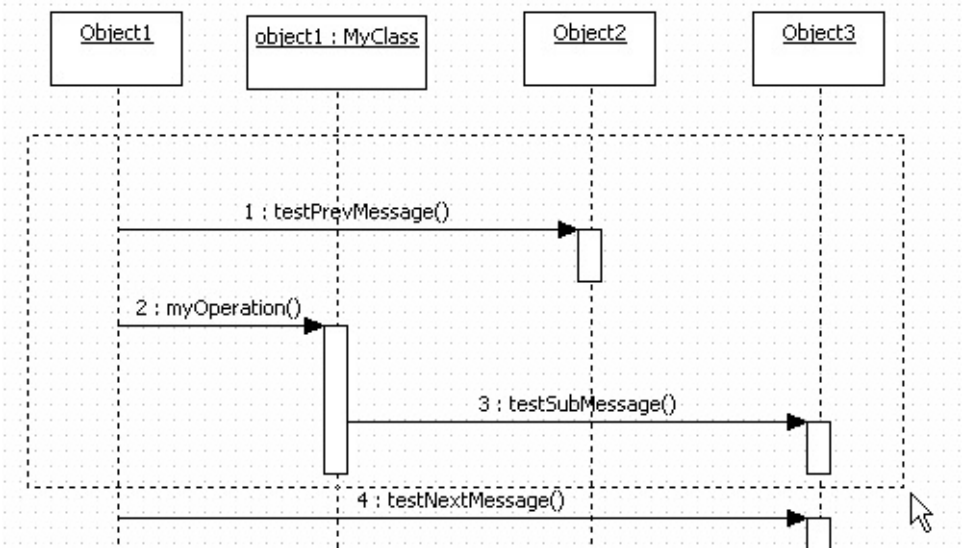
Procedure for creating combined fragment

In order to create Combined Fragment,

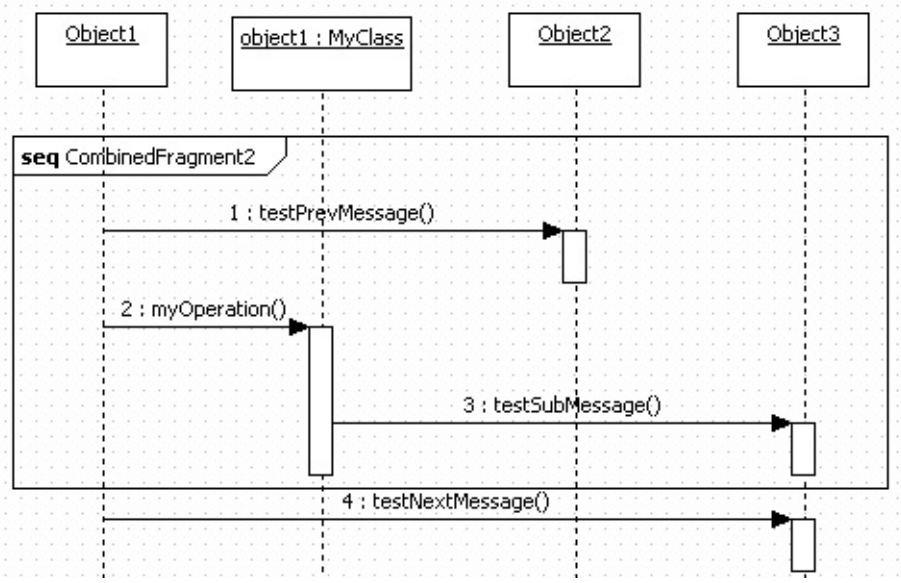
1. Click **[Toolbox] -> [Sequence] -> [Combined Fragment]** button.



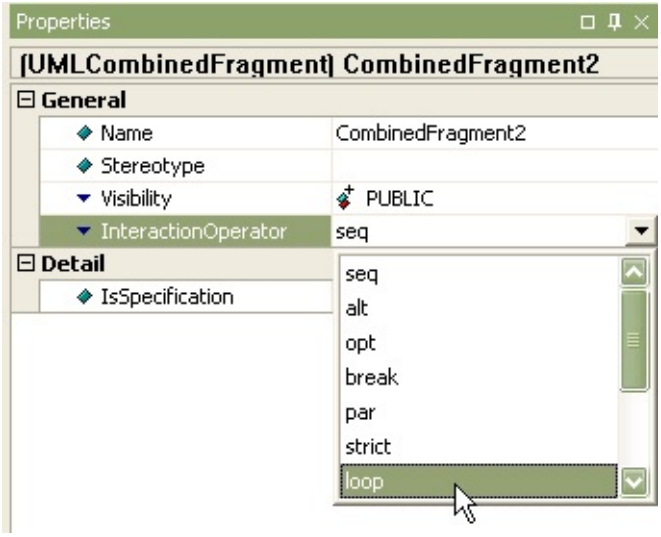
2. And click at the position where Combined Fragment will be placed in the **[main window]**.



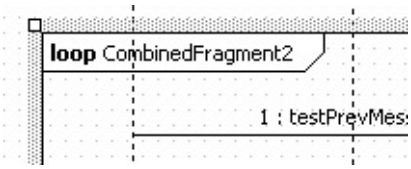
3. A combined fragment is created.



4. Change interaction operator in the properties as follows.



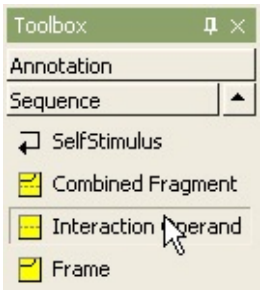
5. The combined fragment is shown as follows.



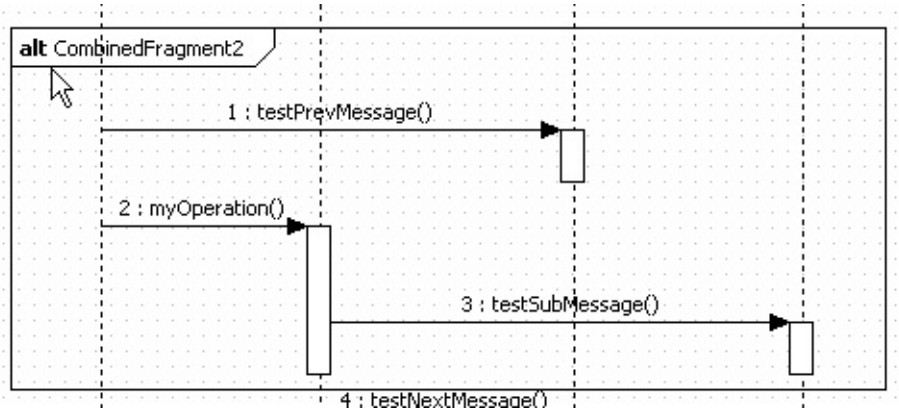
Procedure for creating interaction operand

In order to create Interaction Operand,

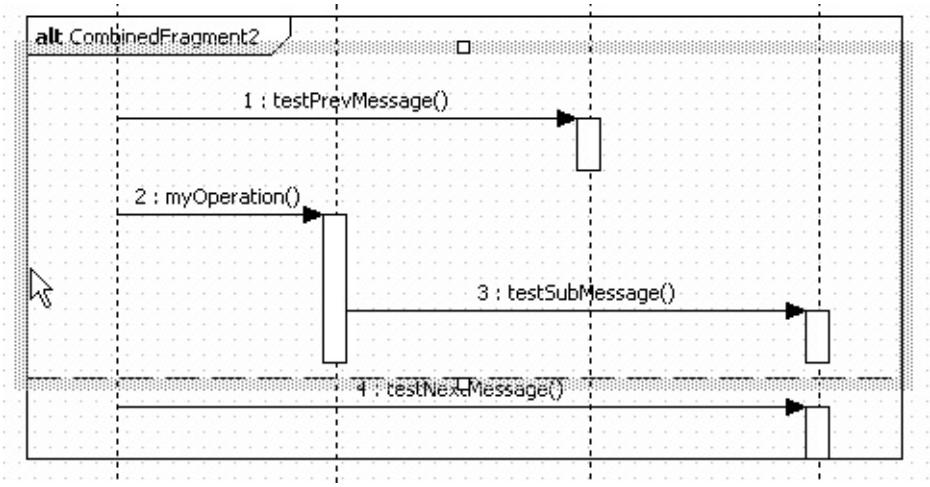
1. Click **[Toolbox] -> [Sequence] -> [Interaction Operand]** button.



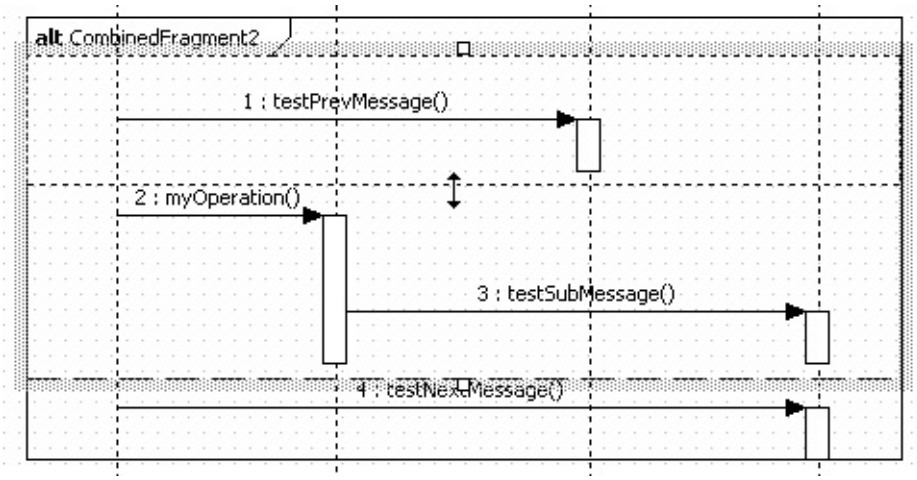
2. And click at the Combined Fragment where Interaction Operand will be placed in the **[main window]**.



3. New interaction operand is added to the combined fragment. Click the interaction operand.



4. The selection points of interaction operand are shown, drag it to arrange its boundary.

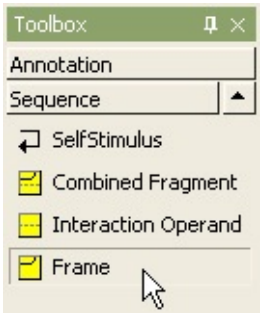

alt CombinedFragment2
1 : testPrevMessage()
2 : myOperation()
3 : testSubMessage()
4 : testNextMessage()

Frame

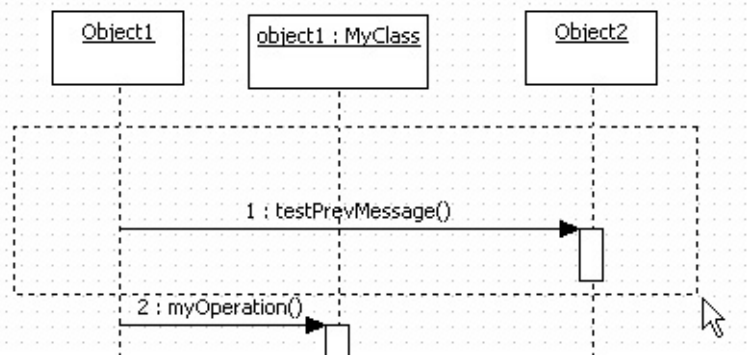
Procedure for creating frame

In order to create Frame,

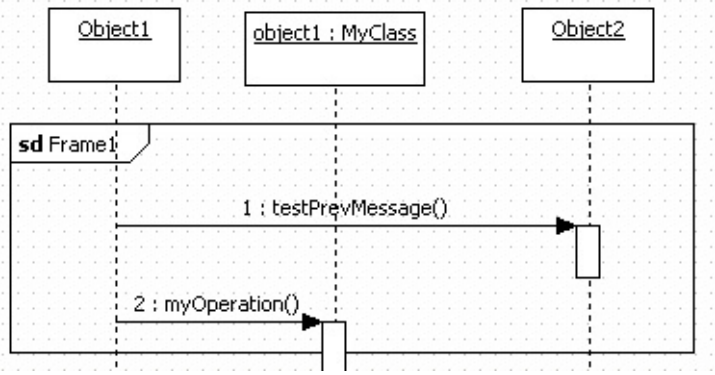
1. Click **[Toolbox] -> [Sequence] -> [Frame]** button.



2. And click at the position where Frame will be placed in the **[main window]**.



3. A new frame is created as follows.

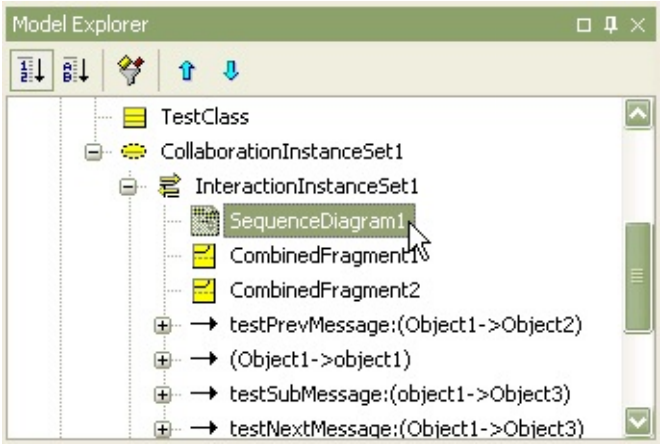


Diagram

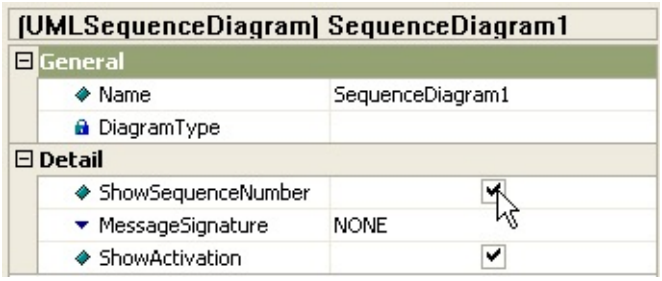
Procedure for showing sequence numbers in the diagram

In order to show or hide stimulus sequence number,

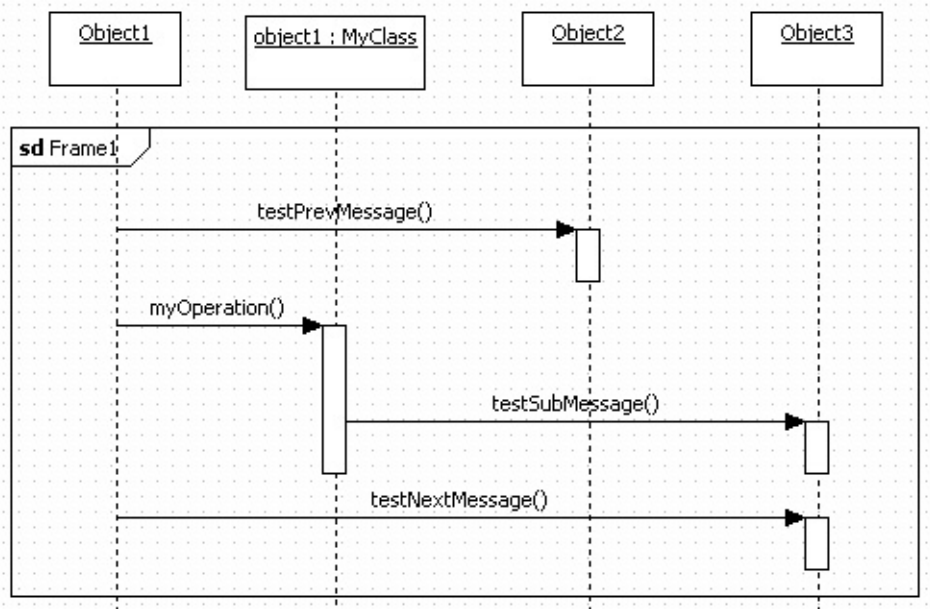
1. Select the diagram in the **[model explorer]** or in the **[main window]**



2. And configure **[ShowSequenceNumber]** property of diagram to true or false.

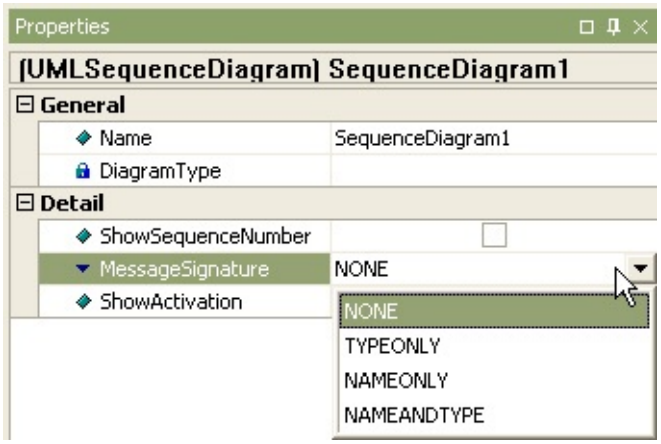


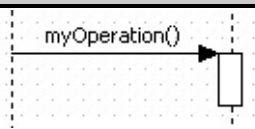
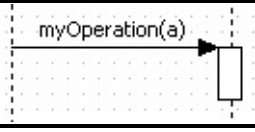
3. When **[ShowSequenceNumber]** is false, sequence diagram is shown as follows.

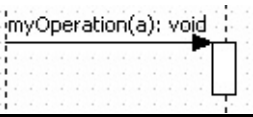
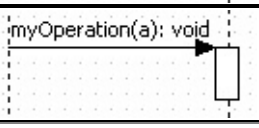


Procedure for changing signature style of message in the diagram

There are four message style. To change stimulus signature, select the diagram in the **[model explorer]** or in the **[main window]**, and configure **[MessageSignature]** property of diagram to one of the followings.

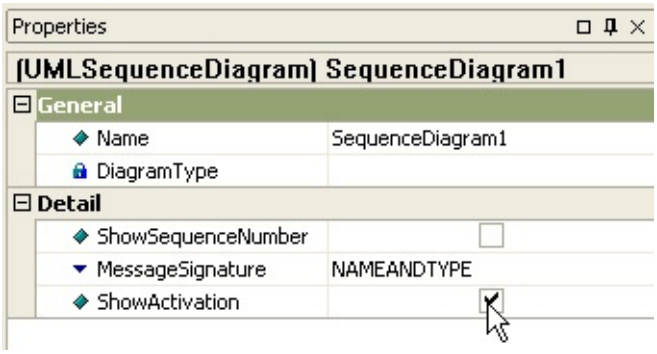


Style	Example
NONE	
NAMEONLY	

TYPEONLY	
NAMEANDTYPE	

Procedure for changing activation style in the diagram

In order to show or hide stimulus activation, select the diagram in the **[model explorer]** or in the **[main window]**, and configure **[ShowActivation]** property of diagram to true or false.



Modeling with Collaboration Diagram

The following elements are available in a collaboration diagram.

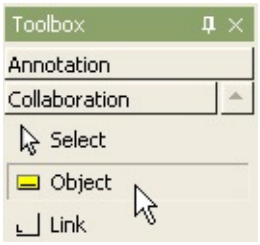
- Object
- Link
- SelfLink
- Stimulus
- Frame

Object

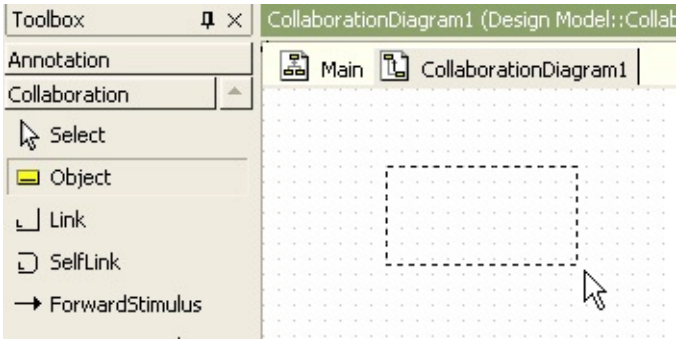
Procedure for creating object

In order to create Object,

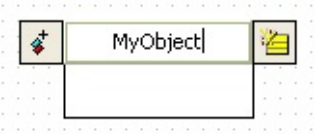
1. Click **[Toolbox]** -> **[Collaboration]** -> **[Object]** button.



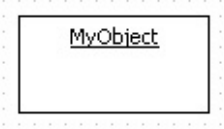
2. And click at the position where Object will be placed in the **[main window]**.



3. Then quick dialog is shown. At the quick dialog, enter the object name.



4. And press **[Enter]** key.



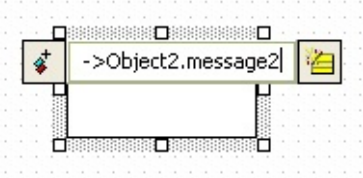
Procedure for creating outgoing from object stimulus by using shortcut creation syntax

In order to create outgoing stimulus from selected object to another object,

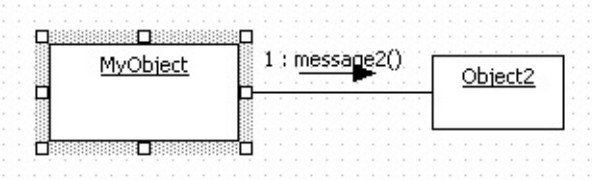
Double-click from-object, or select from-object and press **[Enter]** key to pop

1. up quick dialog.

2. At the quick dialog, enter stimulus name after "->" string ("<-" string for incoming and "<->" for outgoing with return).



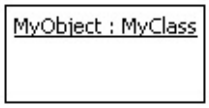
3. Press **[Enter]** key and outgoing stimulus from selected object to target object is created and placed at the last order.



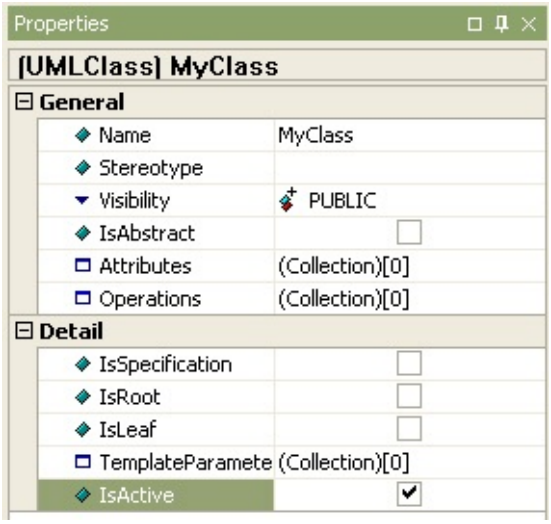
Procedure for setting active object

In order to set class to active object,

1. Set assigned class's **[IsActive]** property to true.



2. For MyObject, change MyClass's **[IsActive]** property.

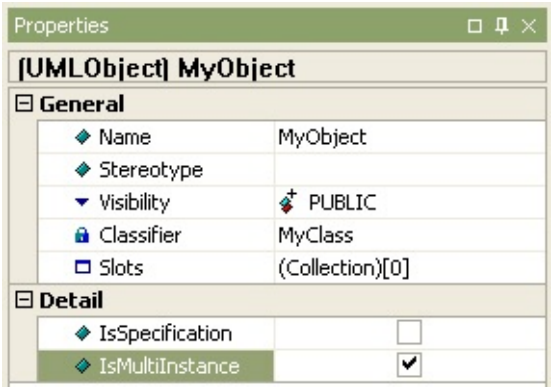


3. If class property is not assigned, you can't change object to active object.

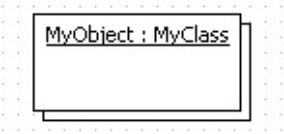
Procedure for setting to multi object

In order to set object to multi object,

1. Set object's IsMultiInstance property to true.



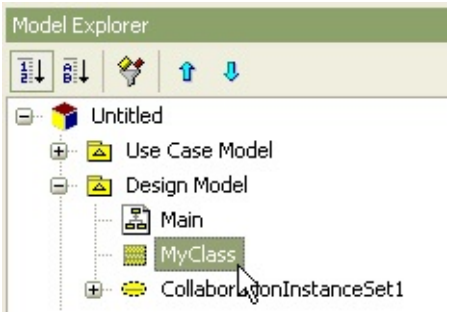
2. Then the object is assigned as multi object.



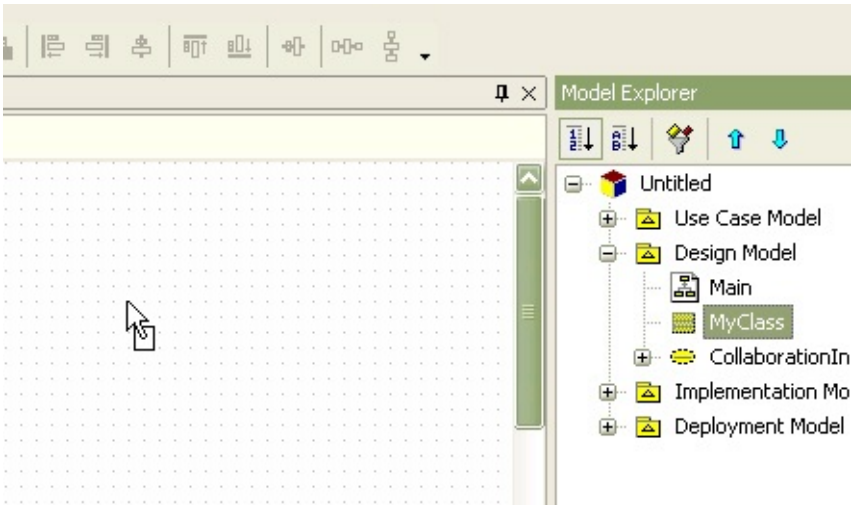
Procedure for creating object from class

In order to create object from class,

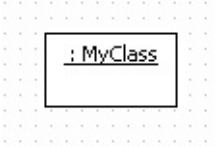
1. Select class in the **[model explorer]**.



2. Drag it into collaboration diagram.



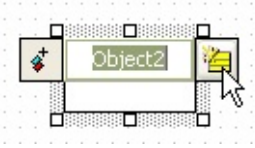
3. Then the object(instance of the class) is created.



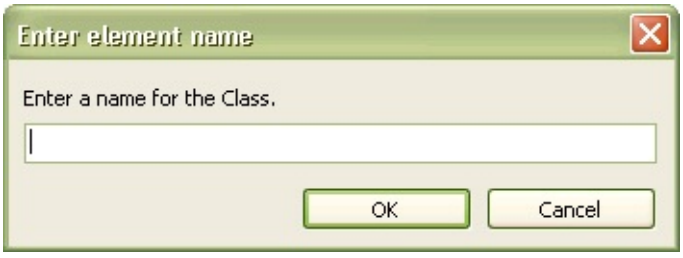
Procedure for creating class from object

If class is not assigned to object,

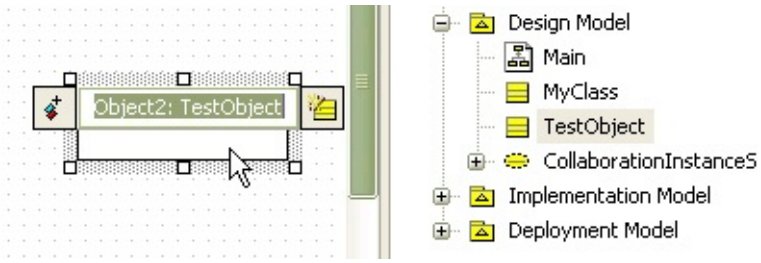
1. Double-click object to pop up quick dialog. Then quick dialog is opened.
2. At the quick dialog, click add class button.



3. At the **[Enter element name]** dialog, enter new class name.



4. Then new class is created and assigned to object.



If you want existing class to be assigned to object, enter the existing class name at the **[Select a model element]** dialog.

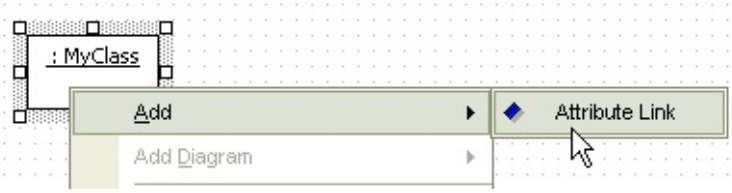
Procedure for adding AttributeLink to object

There are two way to add attribute link to Object.

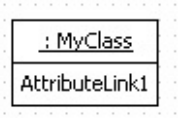
- using object model in the main diagram or the **[model explorer]**
- using **[collection editor]**

In the case of using object model,

1. Select object in the **[main window]** or in the **[model explorer]**.
2. Right-click the selected object, select **[Add] -> [Attribute Link]** popup menu, and you can add Attribute Link.

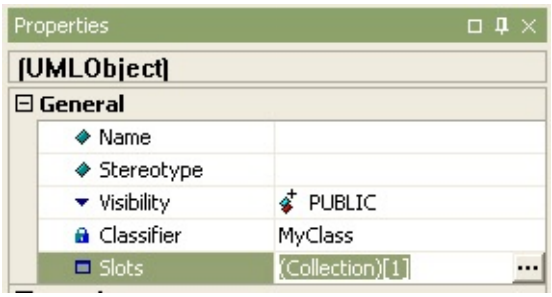


3. Then new attribute link is created.

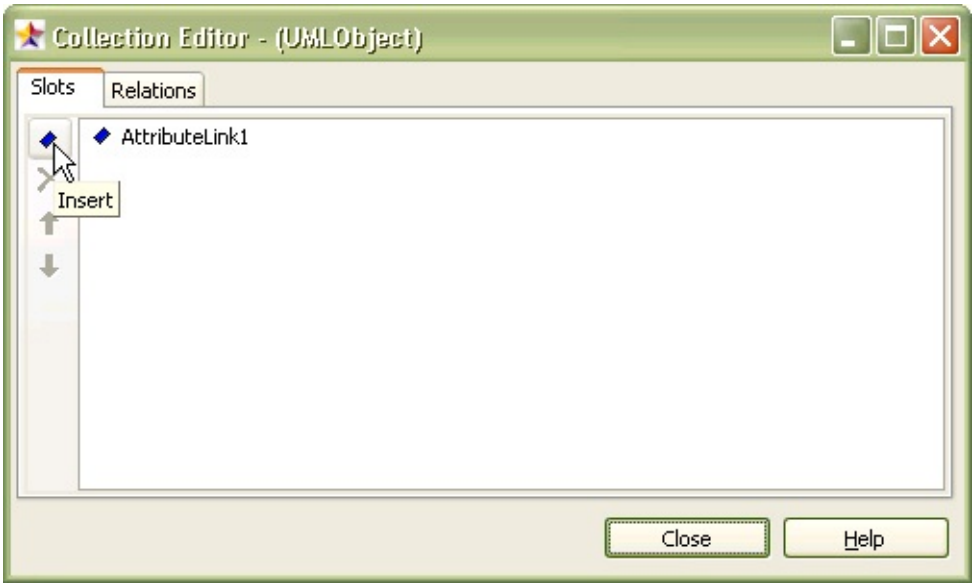


In the other case,

1. Select **[Collection Editor...]** popup menu of object or click ... button in **[Slots]** property on properties window.



2. At slots tab of the **[collection editor]**, you can add attribute link by using  button.

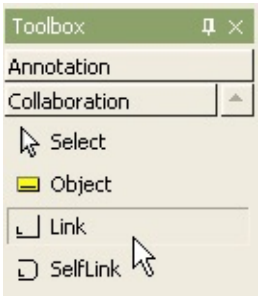


Link

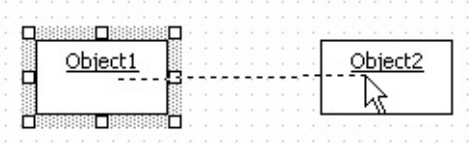
Procedure for creating link

In order to create Link,

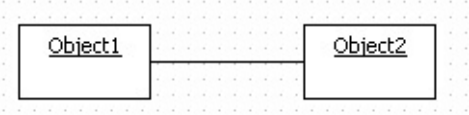
1. Click **[Toolbox]** -> **[Collaboration]** -> **[Link]** button.



2. Drag from one Object and drop to the other Object in the **[main window]**.



3. Between two objects, the link is created.

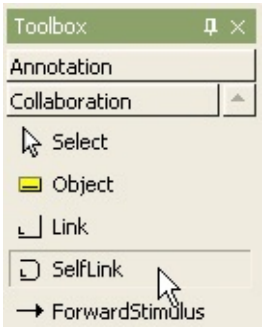


SelfLink

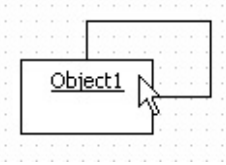
Procedure for creating self-link

In order to create self-link,

1. Click **[Toolbox] -> [Collaboration] -> [SelfLink]** button.



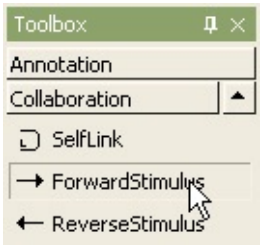
2. And click the object that self-link will connect to in the **[main window]**.



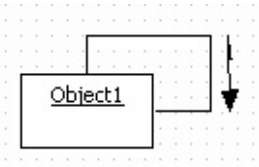
Procedure for creating self-stimulus

In order to create self-stimulus,

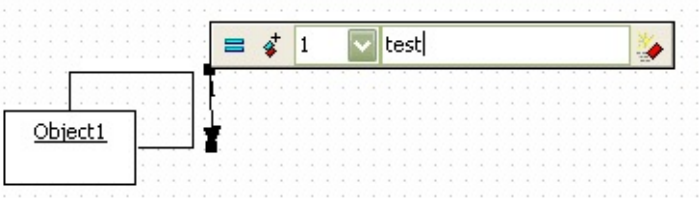
1. Click **[Toolbox] -> [Collaboration] -> [ForwardStimulus/ReverseStimulus]** button.



2. And click the self-link that the stimulus will be placed in the **[main window]**.



3. And double-click the stimulus, enter the stimulus name at the quick dialog.

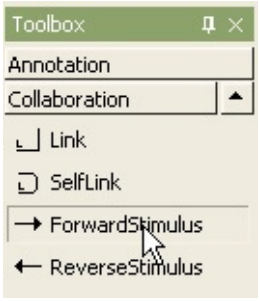


Stimulus

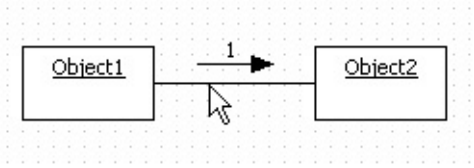
Procedure for creating stimulus

In order to create stimulus,

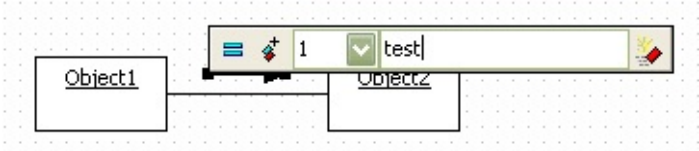
1. Click **[Toolbox]** -> **[Collaboration]** -> **[ForwardStimulus/ReverseStimulus]** button.



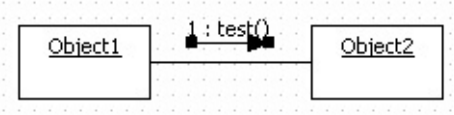
2. Click the link that the stimulus will be placed in the **[main window]**.



3. And double-click the stimulus, enter the stimulus name at the quick dialog.

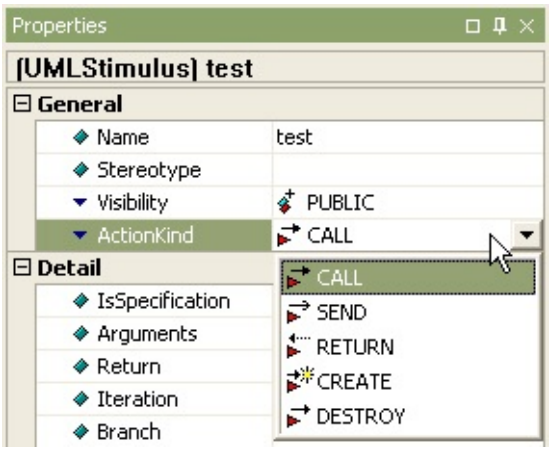


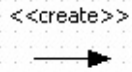
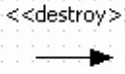
4. The result is as follows.



Procedure for changing ActionKind of stimulus

The **[ActionKind]** property of stimulus should be assigned to one of five sort as following. To change **[ActionKind]** property, select stimulus and select the **[ActionKind]** property on the properties window.



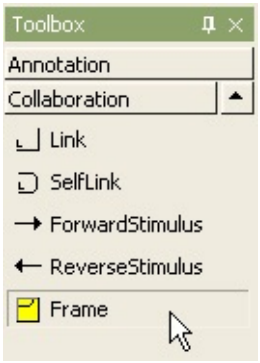
ActionKind	Shape
CALL	
SEND	
RETURN	
CREATE	
DESTROY	

Frame

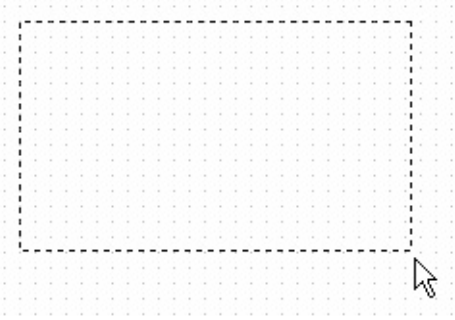
Procedure for creating frame

In order to create Frame,

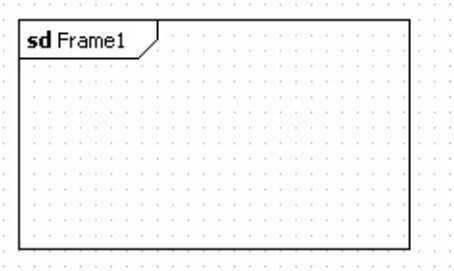
1. Click **[Toolbox] -> [Collaboration] -> [Frame]** button.



2. And click at the position where Frame will be placed in the **[main window]**.



3. The result is as follows.

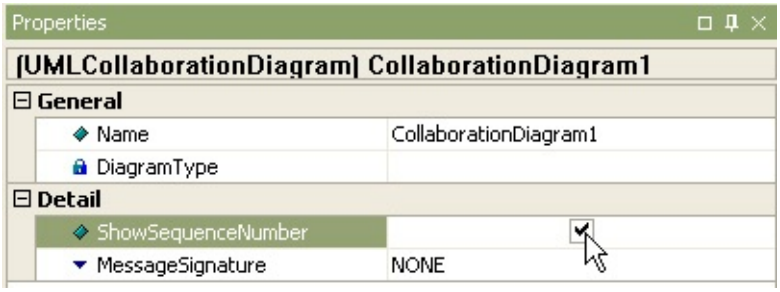


Diagram

Procedure for showing sequence numbers in the diagram

In order to show or hide stimulus sequence number, select the diagram in the

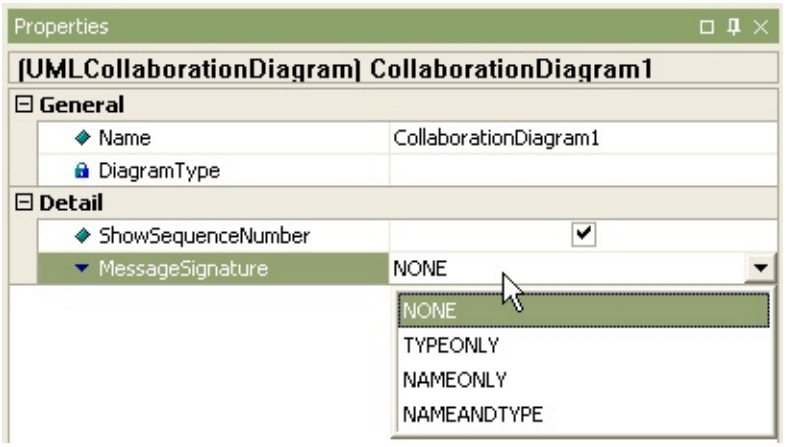
[model explorer] or in the **[main window]**, and configure **[ShowSequence]** property of diagram to true or false.



Procedure for changing signature style of message in the diagram

There are four message style. To change stimulus signature,

1. Select the diagram in the **[model explorer]** or in the **[main window]**.



2. And configure **[MessageSignature]** property of diagram to one of the followings.

Style	Description
NONE	shows only message name
NAMEONLY	shows message name and arguement name
TYPEONLY	shows message name, arguement type, and return type
NAMEANDTYPE	shows message name, arguement name, arguement type, and return type

Modeling with Activity Diagram

The following elements are available in a activity diagram.

- ActionState
- SubactivityState
- InitialState
- FinalState
- Synchronization
- Decision
- Flow Final
- Object Flow
- Signal Accept State
- Signal Send State
- Transition
- SelfTransition
- Swimlane

ActionState

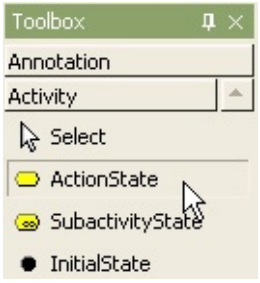
Semantics

An action state represents the execution of an atomic action, typically the invocation of an operation. An action state is a simple state with an entry action whose only exit transition is triggered by the implicit event of completing the execution of the entry action. The state therefore corresponds to the execution of the entry action itself and the outgoing transition is activated as soon as the action has completed its execution.

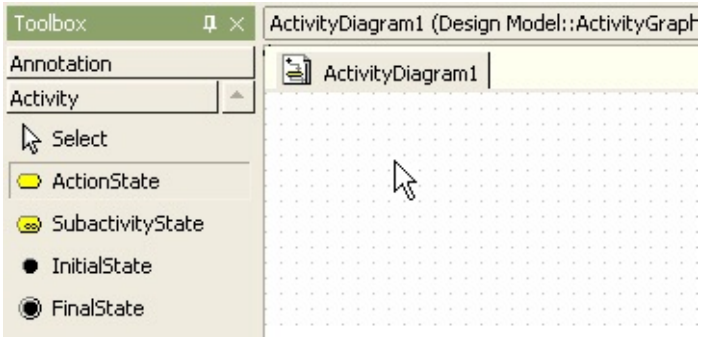
Procedure for creating action state

In order to create ActionState,

1. Click **[Toolbox] -> [Activity] -> [ActionState]** button.



2. And click at the position where ActionState will be placed in the **[main window]**.



3. A action state is created on the diagram and the quick dialog is shown.

4. Enter the action state name at the quick dialog and press **[Enter]** key. The result is as follows.

SubactivityState

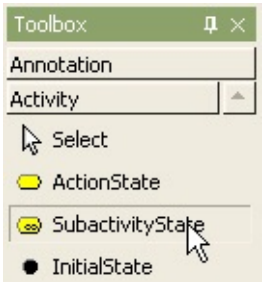
Semantics

A subactivity state represents the execution of a non-atomic sequence of steps that has some duration; that is, internally it consists of a set of actions and possibly waiting for
events. That is, a subactivity state is a hierarchical action, where an associated subactivity graph is executed.

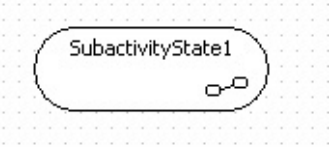
Procedure for creating subactivity state

In order to create SubactivityState,

1. Click **[Toolbox]** -> **[Activity]** -> **[SubactivityState]** button.



2. And click at the position where SubactivityState will be placed in the **[main window]**. A subactivity state is created and the quick dialog is shown. At the quick dialog, enter the subactivity state name and press **[Enter]** key. The result is as follows.

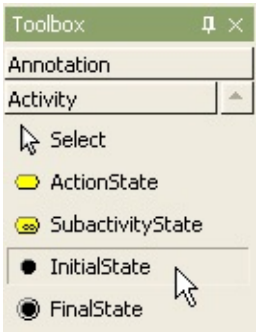


InitialState

Procedure for creating initial state

In order to create InitialState,

1. Click **[Toolbox] -> [Activity] -> [InitialState]** button.



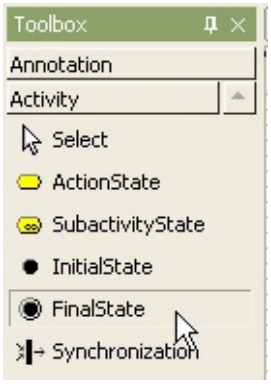
2. And click at the position where InitialState will be placed in the **[main window]**. Then a initial state is created.

FinalState

Procedure for creating final state

In order to create FinalState,

1. Click **[Toolbox] -> [Activity] -> [FinalState]** button.



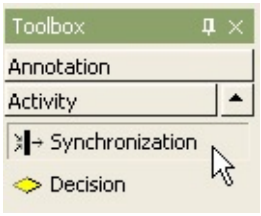
2. And click at the position where FinalState will be placed in the **[main window]**.

Synchronization

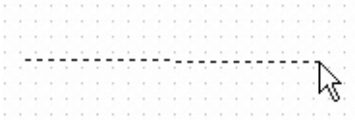
Procedure for creating synchronization bar

In order to create Synchronization,

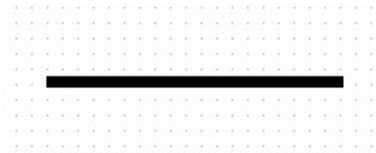
1. Click **[Toolbox] -> [Activity] -> [Synchronization]** button.



2. And click at the position where Synchronization will be placed in the **[main window]** and drag as size as you want.



3. The following figure shows the result of this procedure.



Decision

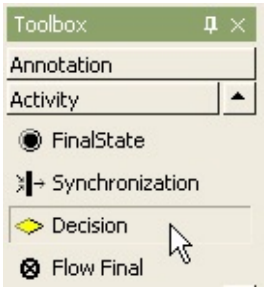
Semantics

A state diagram (and by derivation an activity diagram) expresses a decision when guard conditions are used to indicate different possible transitions that depend on Boolean conditions of the owning object.

Procedure for creating decision

In order to create Decision,

1. Click **[Toolbox]** -> **[Activity]** -> **[Decision]** button.

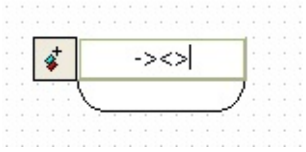


2. And click at the position where Decision will be placed in the **[main window]**. The decision is created on the diagram.

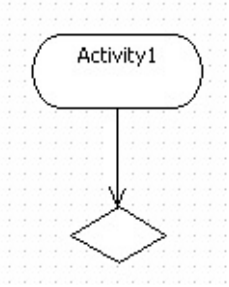
Procedure for creating decision from state

In order to create decision with incoming transition from selected object, use shortcut creation syntax.

1. Double-click state. At the quick dialog, enter "-><>"("<-<>" for incoming from decision) string.



2. Press **[Enter]** key and decision with outgoing transition from selected state is created.

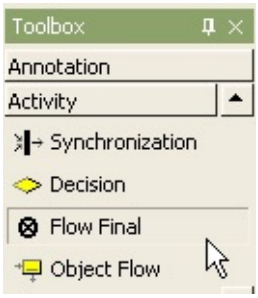

Activity1

Flow Final

Procedure for creating flow final

In order to create Flow Final,

1. Click **[Toolbox] -> [Activity] -> [Flow Final]** button.



2. And click at the position where Flow Final will be placed in the **[main window]**.

Object Flow

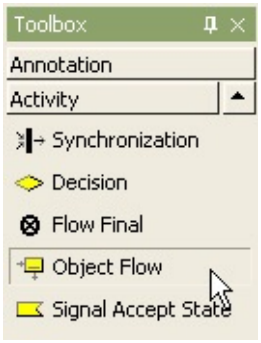
Semantics

An object flow is one of two types of activity edges, which are directed connection (flows) between activity nodes, the other being a control flow. As soon as the activity node at the source (tail) end of the flow is finished it presents tokens to the object flow at the target (arrowhead) end of the flow. An object flow can only carry object (data) tokens; it cannot carry control tokens. There are rules that specify whether tokens can flow along the object flow and these are determined by the type of activity node at the source and target of the flow. In the case of complete activities an object flow may define a weight, which specifies the minimum number of tokens that must flow along the object flow as a group.

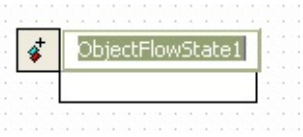
Procedure for creating object flow

In order to create Object Flow,

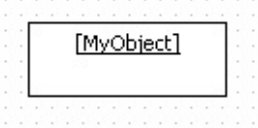
1. Click **[Toolbox]** -> **[Activity]** -> **[Object Flow]** button.



2. And click at the position where Object Flow will be placed in the **[main window]**. Then the quick dialog of object flow state is shown as follows.



3. At the quick dialog, enter the object flow state name and press **[Enter]** key.



Signal Accept State

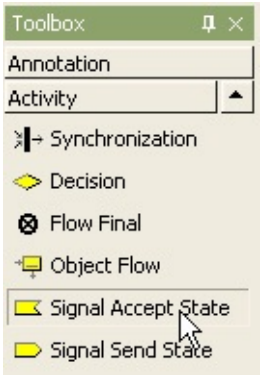
Semantics

The signal accept may be shown as a concave pentagon that looks like a rectangle with a triangular notch in its side (either side). The signature of the signal is shown inside the symbol. An unlabeled transition arrow is drawn from the previous action state to the pentagon and another unlabeled transition arrow is drawn from the pentagon to the next action state. A dashed arrow may be drawn from an object symbol to the notch on the pentagon to show the sender of the signal; this is optional.

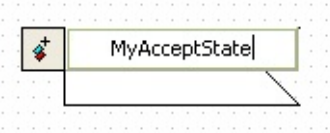
Procedure for creating signal accept state

In order to create Signal Accept State,

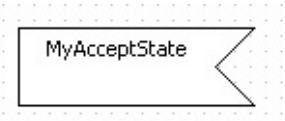
1. Click **[Toolbox] -> [Activity] -> [Signal Accept State]** button.



2. And click at the position where Signal Accept State will be placed in the **[main window]**.



3. At the quick dialog, enter signal accept state name and press **[Enter]** key.



Signal Send State

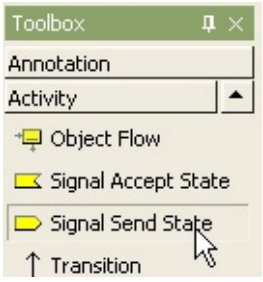
Semantics

The sending of a signal may be shown as a convex pentagon that looks like a rectangle with a triangular point on one side (either side). The signature of the signal is shown inside the symbol. An unlabeled transition arrow is drawn from the previous action state to the pentagon and another unlabeled transition arrow is drawn from the pentagon to the next action state. A dashed arrow may be drawn from the point on the pentagon to an object symbol to show the receiver of the signal, this is optional.

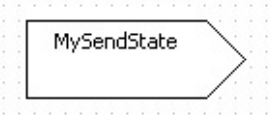
Procedure for creating signal send state

In order to create Signal Send State,

1. Click **[Toolbox] -> [Activity] -> [Signal Send State]** button.



2. And click at the position where Signal Send State will be placed in the [main window]. A signal send state is created and the quick dialog is shown. Enter signal send state name and press **[Enter]** key.

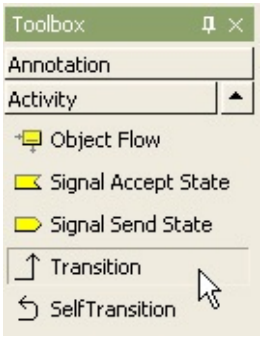


Transition

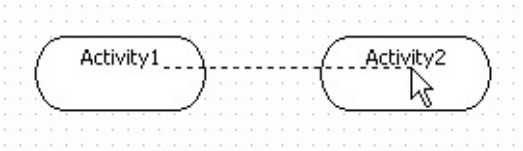
Procedure for creating transition

In order to create Transition,

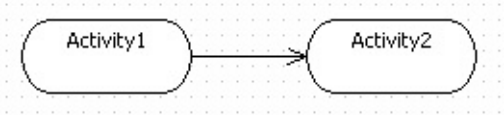
1. Click **[Toolbox] -> [Activity] -> [Transition]** button.



2. Drag and drop between states in transition direction in the **[main window]**.



3. Then the transition is created.

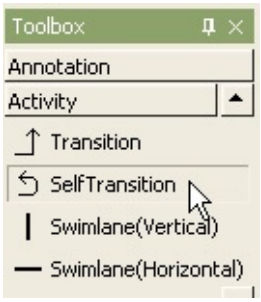


SelfTransition

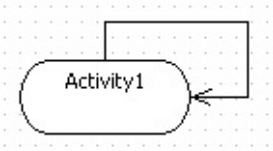
Procedure for creating self-transition

In order to create self-transition,

1. Click **[Toolbox]** -> **[Activity]** -> **[SelfTransition]** button.



2. Click state to have self-transition in the **[main window]**. Then a self-transition is created.



Swimlane

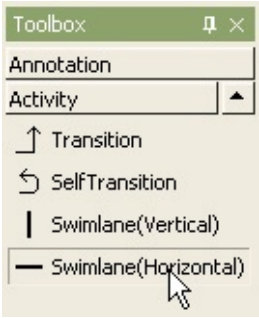
Semantics

Actions and subactivities may be organized into swimlanes. Swimlanes are used to organize responsibility for actions and subactivities. They often correspond to organizational units in a business model.

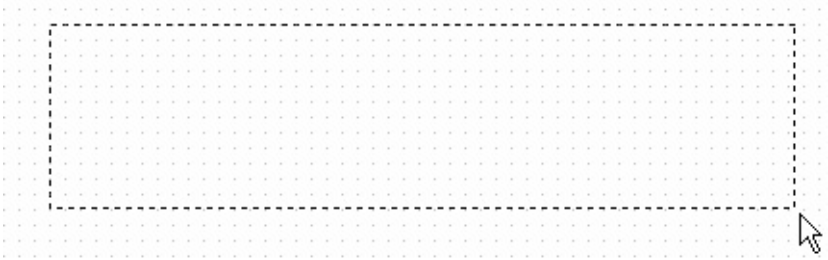
Procedure for creating horizontal swimlane

In order to create Horizontal Swimlane,

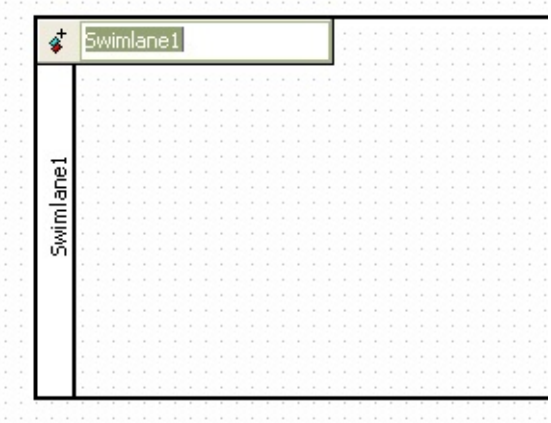
1. Click **[Toolbox]** -> **[Activity]** -> **[Horizontal Swimlane]** button.



2. And drag the boundary where Horizontal Swimlane will be placed in the **[main window]**.



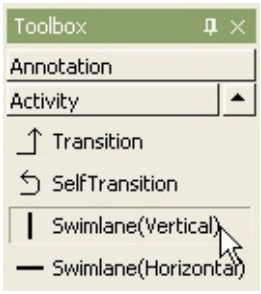
3. Then a horizontal swimlane is created on the diagram. And enter the swimlane name at the quick dialog and press **[Enter]** key.



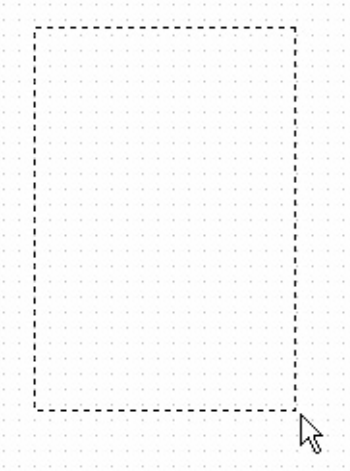
Procedure for creating vertical swimlane

In order to create Vertical Swimlane,

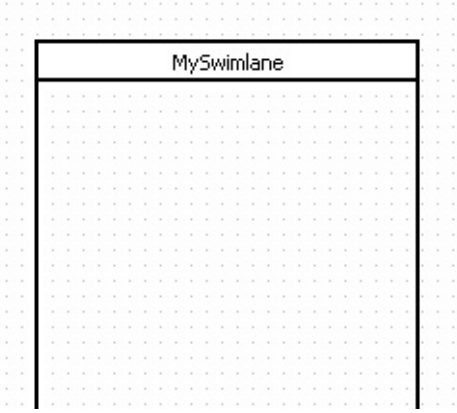
1. Click **[Toolbox] -> [Activity] -> [Vertical Swimlane]** button.



2. And drag the boundary where Vertical Swimlane will be placed in the **[main window]**.



3. A swimlane is created and quick dialog is shown. At the quick dialog, enter the swimlane name and press **[Enter]** to have done this procedure.


MySwimlane

Modeling with Component Diagram

The following elements are available in a component diagram.

- Package
- Interface
- Component
- ComponentInstance
- Artifact
- Port
- Part
- Association
- Dependency
- Realization
- Link
- Connector

Package

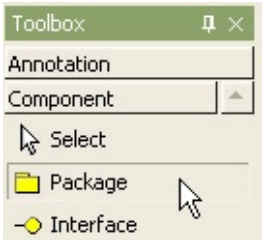
Semantics

A package is a grouping of model elements. Packages themselves may be nested within other packages. A package may contain subordinate packages as well as other kinds of model elements. All kinds of UML model elements can be organized into packages.

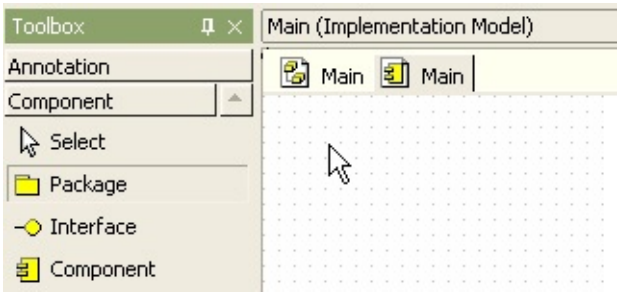
Procedure for creating package

In order to create Package in the component diagram,

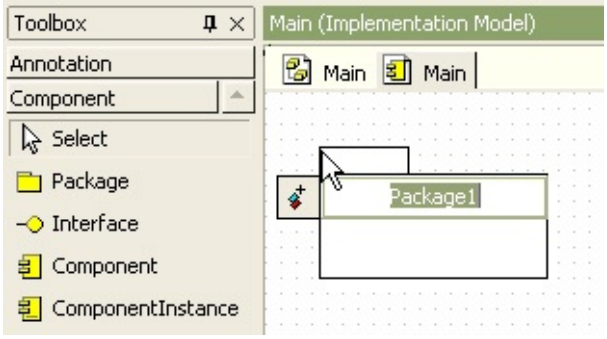
1. Click **[Toolbox]** -> **[Component]** -> **[Package]** button.



2. Click at the position where Package will be placed in the **[main window]**.

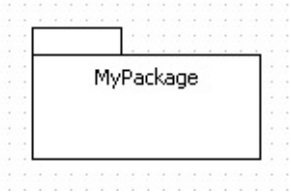


3. A package is created and the quick dialog of package appears.



4. At the quick dialog, enter package name.

5. Press **[Enter]** key. Then the package is shown as follows.

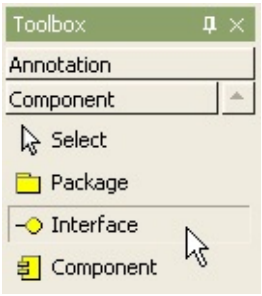


Interface

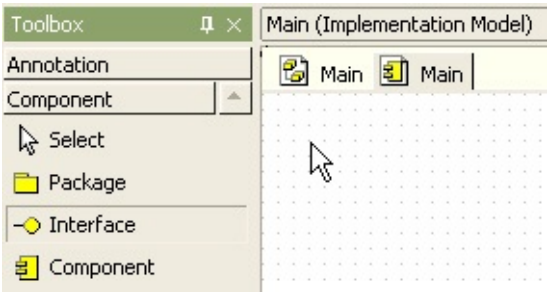
Procedure for creating interface

In order to create Interface,

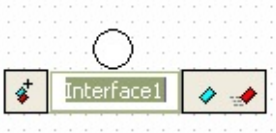
1. Click **[Toolbox] -> [Component] -> [Interface]** button.



2. Click at the position where Interface will be placed in the **[main window]**.



3. At the quick dialog, enter interface name.



4. Press **[Enter]** key. Then the interface is shown as follows.

Component

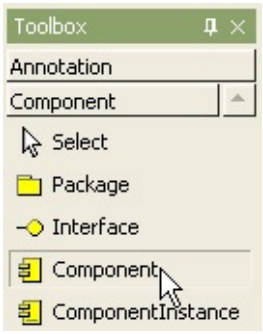
Semantics

A component represents a modular, deployable, and replaceable part of a system that encapsulates implementation and exposes a set of interfaces.

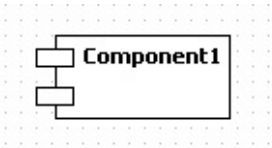
Procedure for creating component

In order to create Component,

1. Click **[Toolbox]** -> **[Component]** -> **[Component]** button.



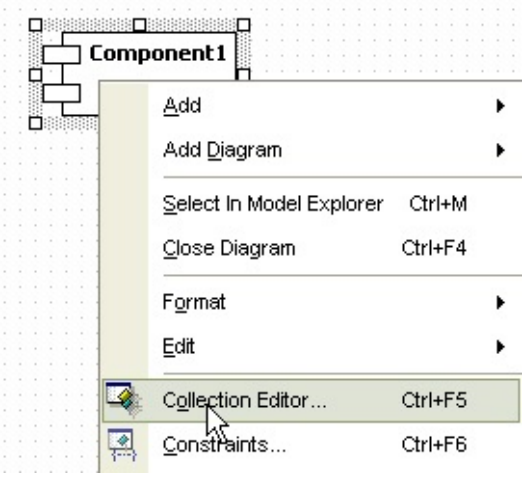
2. Click at the position where Component will be placed in the **[main window]**. And at the quick dialog, enter component name and press **[Enter]** key. The result is as follows.



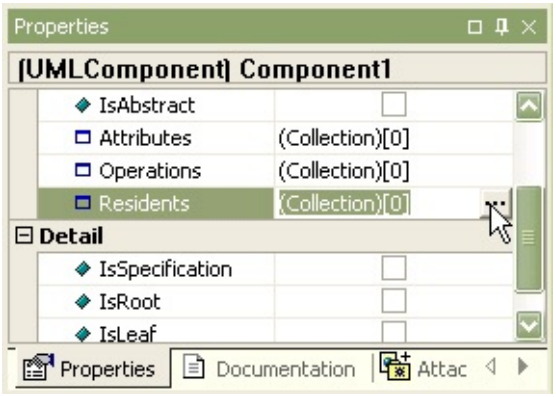
Procedure for adding resident element

In order to add resident element to component,

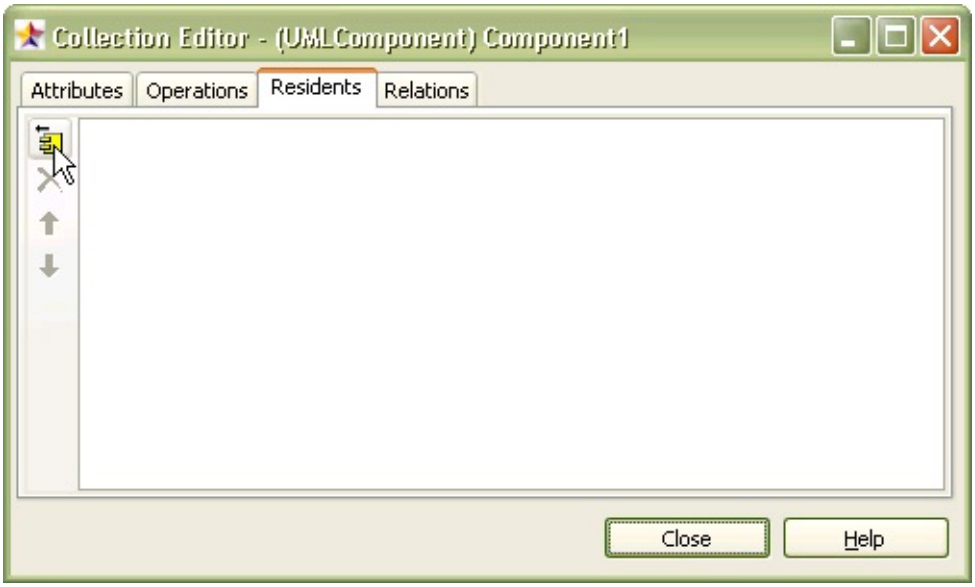
1. Select **[Collection Editor...]** popup menu of component.



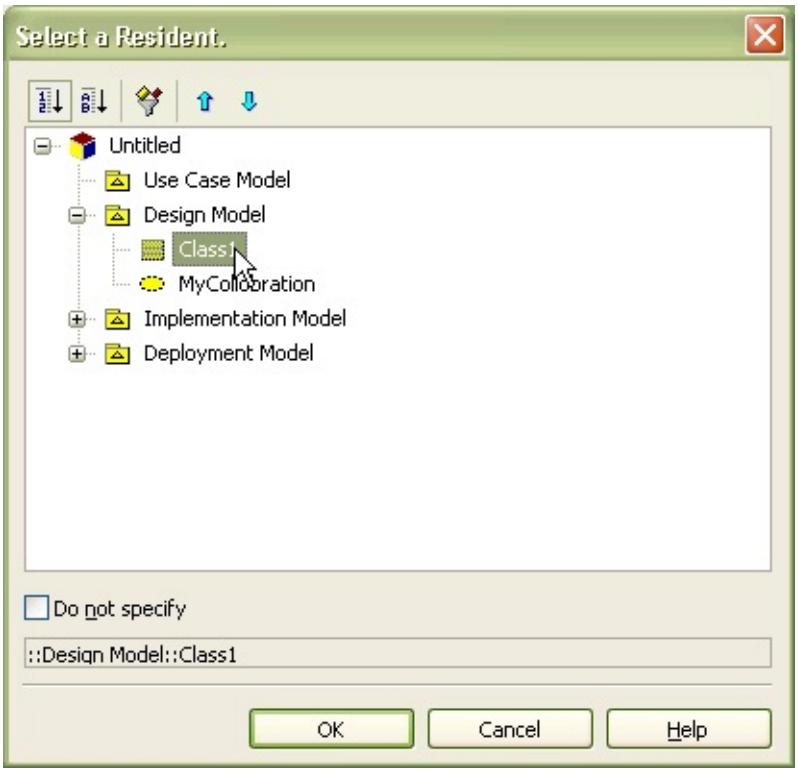
2. Or click ··· button in **[Residents]** property on properties window.



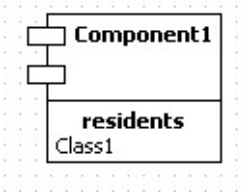
3. At the **[Residents]** tab of the **[collection editor]**, you can add resident element by using button.



4. At the **[Select a Resident]** dialog, select resident component.



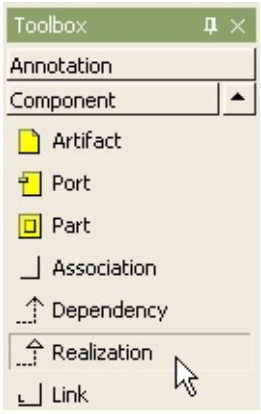
5. The component is assigned to component as resident component and is shown as follows.



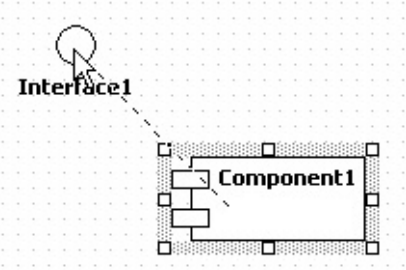
Procedure for creating providing relationship

In order to create providing relationship,

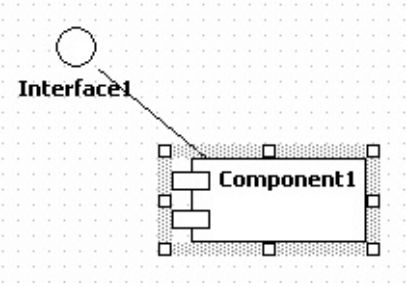
1. Click **[Toolbox] -> [Component] -> [Realization]** button



2. Drag from component and drop to interface in the **[main window]**.



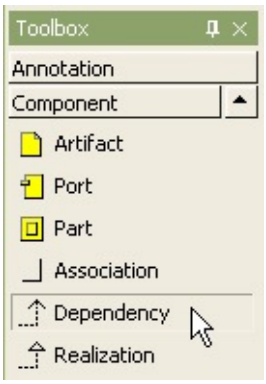
3. The result is as follows.



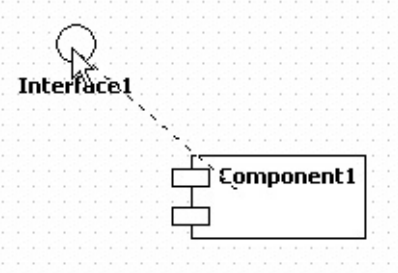
Procedure for creating requiring relationship

In order to create requiring relationship,

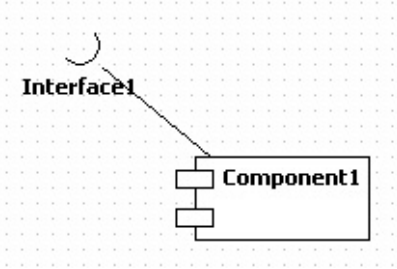
1. Click **[Toolbox] -> [Component] -> [Dependency]** button.



2. Drag from component and drop to interface in the **[main window]**.



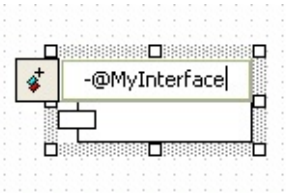
3. Finally, the interface requiring relationship is created.



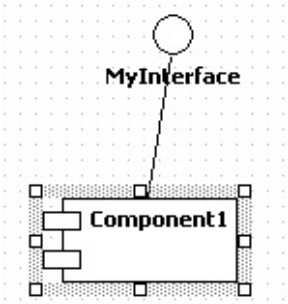
Procedure for creating providing interface of class.

In order to create providing inteface of class, use shortcut creation syntax.

1. Double-click class. At the quick dialog, enter "-@" staring and interface name, separate interface names by "," character.



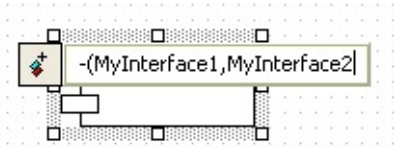
2. And press **[Enter]** key. Several interfaces provided by selected class is created and arranged automatically.



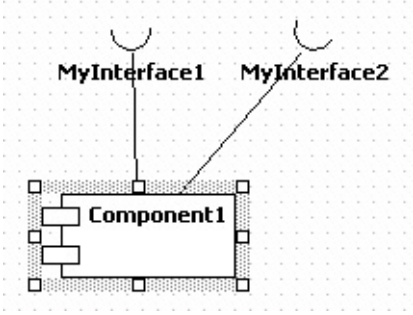
Procedure for creating requiring interface of class.

In order to create requiring inteface of class, use shortcut creation syntax.

1. Double-click class. At the quick dialog, enter "-(" or "-->", and enter interface names, separate interface names by "," character.



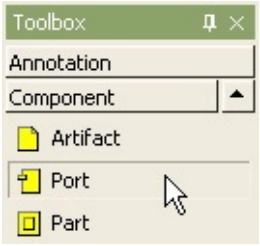
2. And press **[Enter]** key. Several interfaces required by selected class is created and arranged automatically.



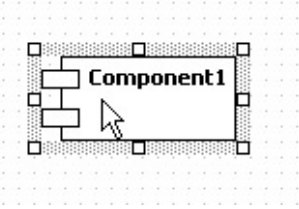
Procedure for creating port

In order to create port on a component,

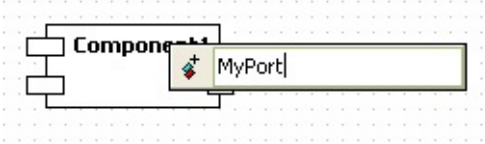
1. Click **[Toolbox] -> [Component] -> [Port]** button.



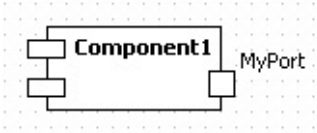
2. And click the component where the port will be contained in the **[main window]**.



3. A port is created on the component. At the quick dialog, enter the port name and press **[Enter]** ken to be complete.



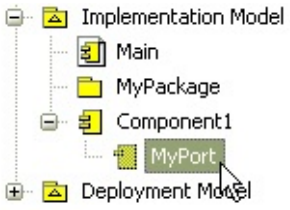
4. The result is as follows.



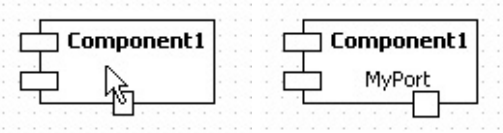
Procedure for creating view by dragging port

You can create port by dragging port from **[model explorer]** to main diagram.

1. Drag port in the **[model explorer]** and drop on the component in the main diagram.



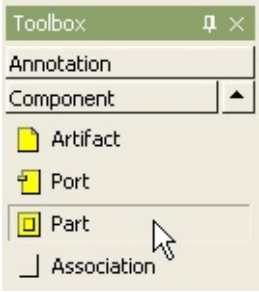
2. A port appears on the component. If it is not dropped on the component but on the other area of the diagram, component with port will be created



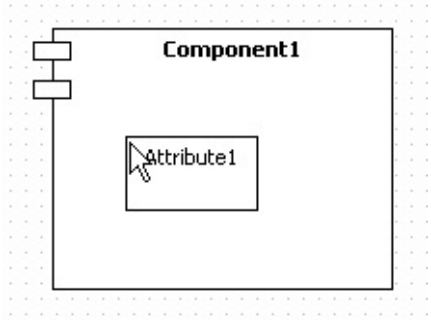
Procedure for creating part

In order to create part,

1. Click **[Toolbox] -> [Component] -> [Part]** button.



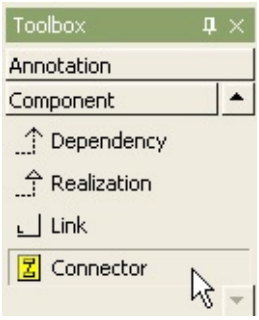
2. And click the component where the part will be contained in the **[main window]**.



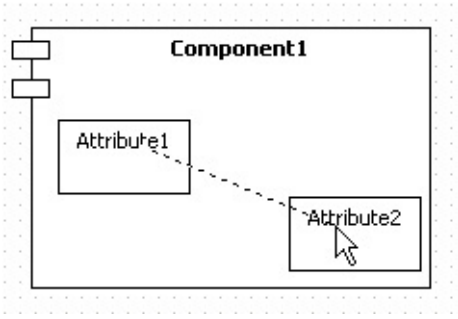
Procedure for creating connector

In order to create connector,

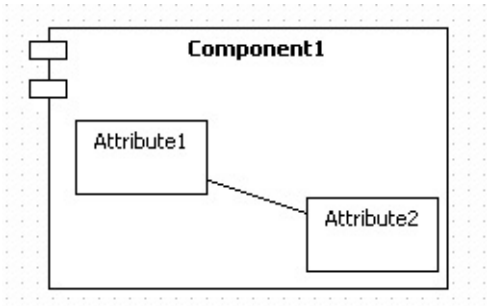
1. Click **[Toolbox] -> [Component] -> [Connector]** button.



2. Drag from one part and drop to the other part in the **[main window]**.



3. The connector between two parts is created finally as follows.



ComponentInstance

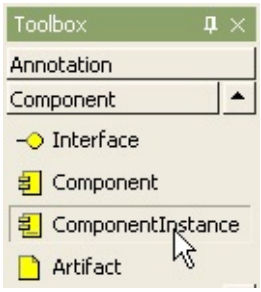
Semantics

A component instance is an instance of a component that resides on a node instance. A component instance may have a state.

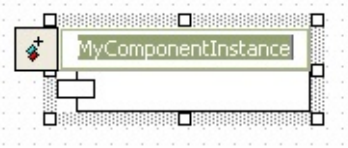
Procedure for creating component instance

In order to create ComponentInstance,

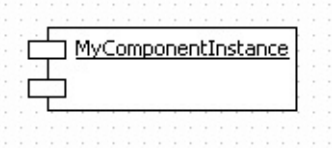
1. Click **[Toolbox] -> [Component] -> [ComponentInstance]** button.



2. And click at the position where ComponentInstance will be placed in the **[main window]**.



3. Enter the component instance name at the quick dialog and press **[Enter]** key. The result is as follows.



Procedure for adding attribute to component instance

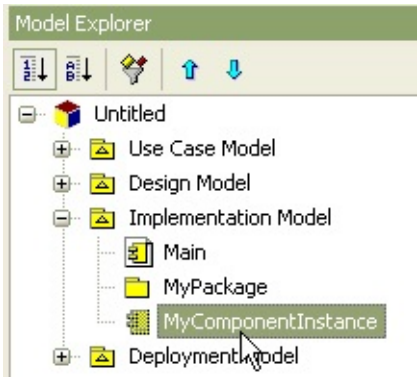
There are two way to add AttributeLink to component instance.

- using ComponentInstance model in the main diagram or the **[model explorer]**

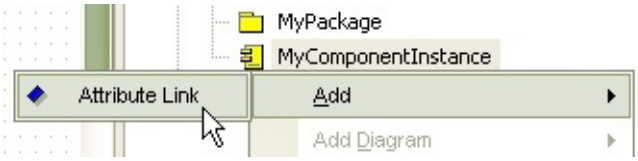
- using **[collection editor]**

In the case of using ComponentInstance model,

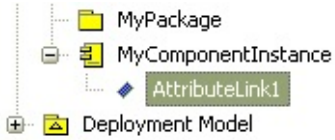
1. Select ComponentInstance in the **[main window]** or in the **[model explorer]**.



2. Right-click the selected ComponentInstance, select **[Add] -> [Attribute Link]** popup menu.

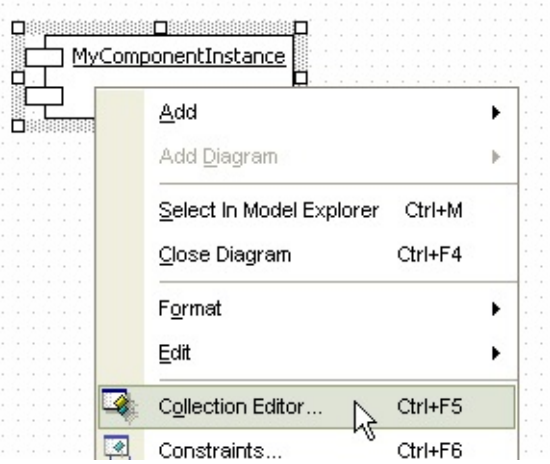


3. and you can add Attribute Link.

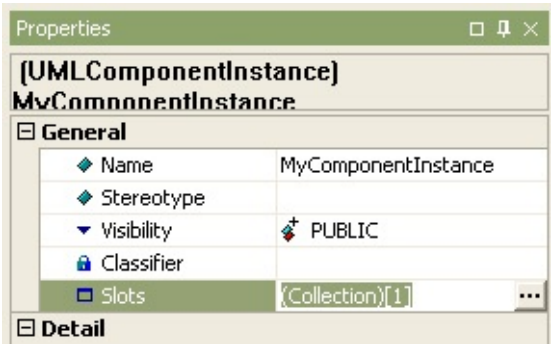


In the other case,

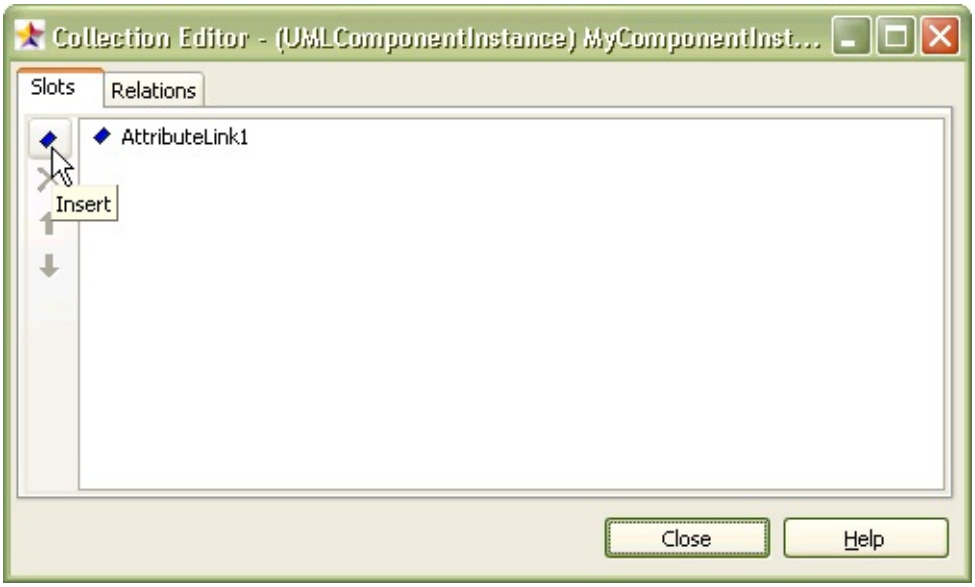
1. Select **[Collection Editor...]** popup menu of ComponentInstance.



2. Click ··· button in slots property on properties window.



3. At **[Slots]** tab of the **[collection editor]**, you can add attribute link by using button.



Artifact

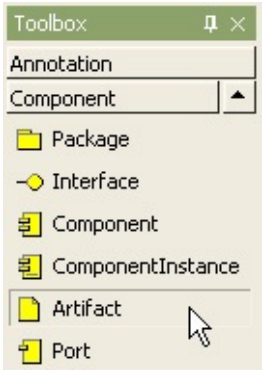
Semantics

An Artifact represents a physical piece of information that is used or produced by a software development process. Examples of Artifacts include models, source files, scripts, and binary executable files. An Artifact may constitute the implementation of a deployable component.

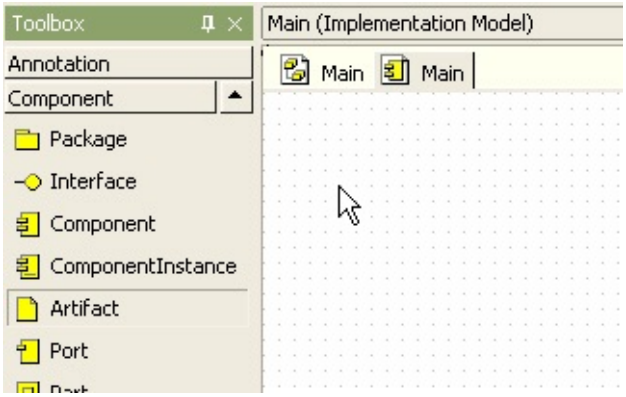
Procedure for creating artifact

In order to create Artifact,

1. Click **[Toolbox] -> [Component] -> [Artifact]** button.



2. And click at the position where Artifact will be placed in the **[main window]**.



3. The artifact is created on the diagram and the quick dialog is shown. At the quick dialog, enter the artifact name

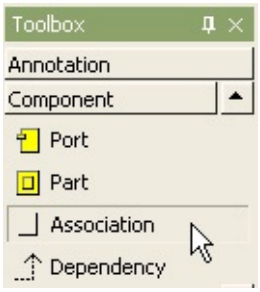
4. Press **[Enter]** Key to have done procedure.

Association

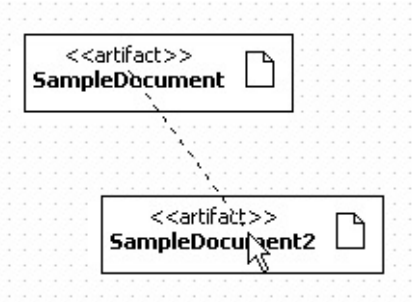
Procedure for creating association

In order to create association,

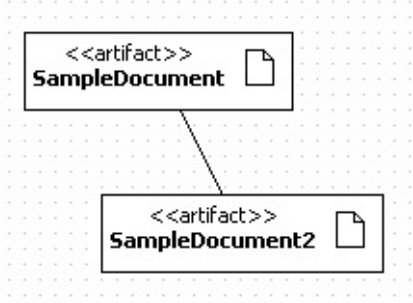
1. Click **[Toolbox] -> [Component] -> [Association]** button.



2. Drag from one associated and drop to another in the **[main window]**.



3. Between two elements, the association is created finally.

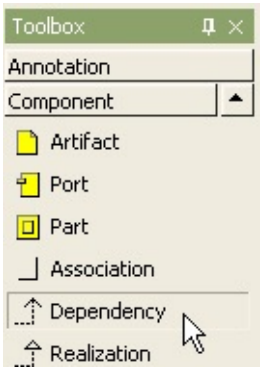


Dependency

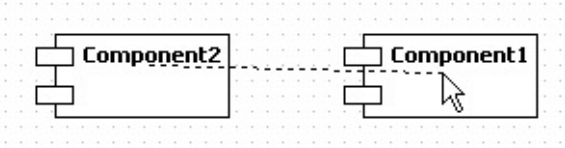
Procedure for creating dependency

In order to create dependency,

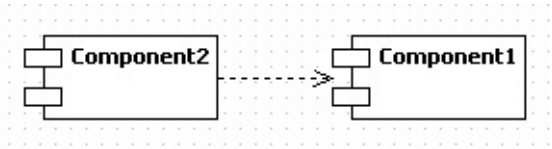
1. Click **[Toolbox] -> [Component] -> [Dependency]** button.



2. Drag and drop between elements in the **[main window]** in depending direction.



3. The dependency between two elements is created.

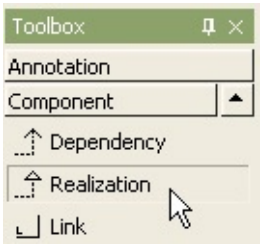


Realization

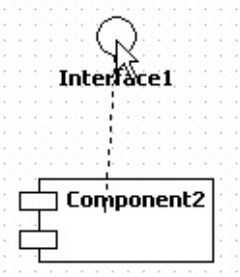
Procedure for creating realization

In order to create realization,

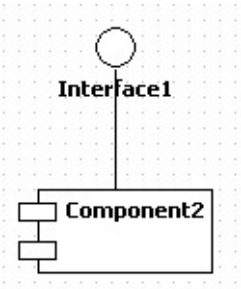
1. Click **[Toolbox] -> [Component] -> [Realization]** button.



2. Drag and drop between elements in the **[main window]** in realization direction.



3. The realization is created as follows.

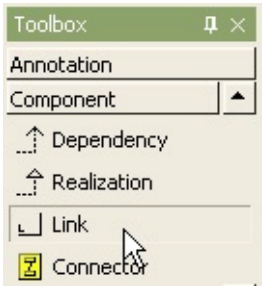


Link

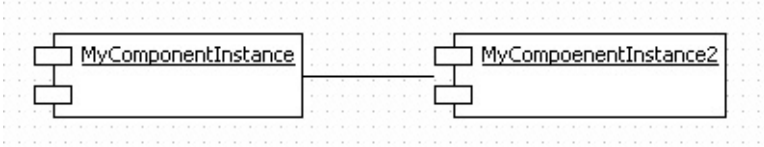
Procedure for creating link

In order to create Link between two components,

1. Click **[Toolbox] -> [Component] -> [Link]** button.



   Drag from one ComponentInstance and drop to the other
2. ComponentInstance in the **[main window]**. Then the link is created as follows.



Modeling with Deployment Diagram

The following elements are available in a deployment diagram.

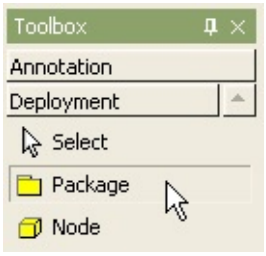
- Package
- Node
- NodeInstance
- Artifact
- Port
- Part
- Association
- DirectedAssociation
- Dependency
- Link
- Connector

Package

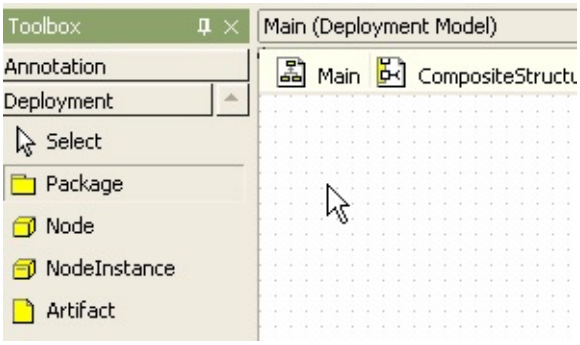
Procedure for creating package

In order to create Package in deployment diagram,

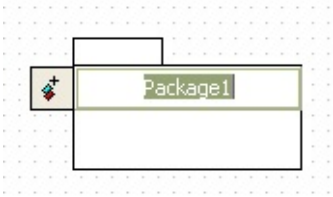
1. Click **[Toolbox]** -> **[Deployment]** -> **[Package]** button.



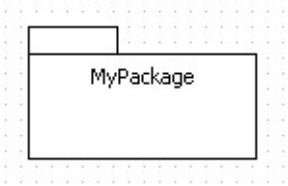
2. Click at the position where package will be placed in the **[main window]**.



3. Then package will be created.



4. At the quick dialog, enter package name and press **[Enter]** key. Then procedure is done.



Node

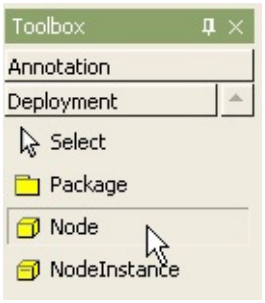
Semantics

A node is a run-time physical object that represents a computational resource, generally having at least a memory and often processing capability as well, and upon which components may be deployed.

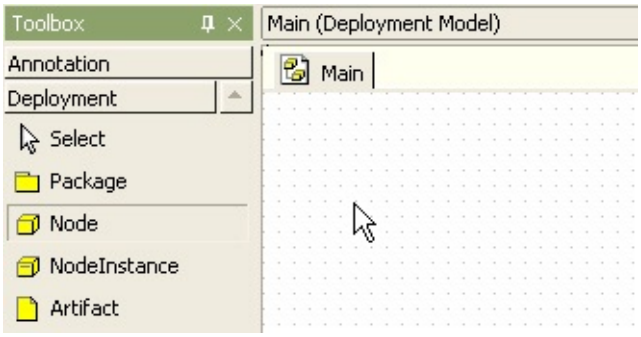
Procedure for creating node

In order to create Node in deployment diagram,

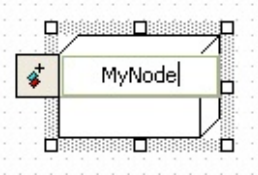
1. Click **[Toolbox]** -> **[Deployment]** -> **[Node]** button.



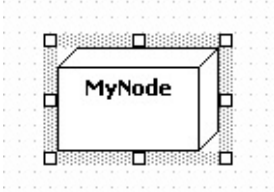
2. Click at the position where Node will be placed in the **[main window]**.



3. Then node is created and the quick dialog appears. Enter the node name at the quick dialog.



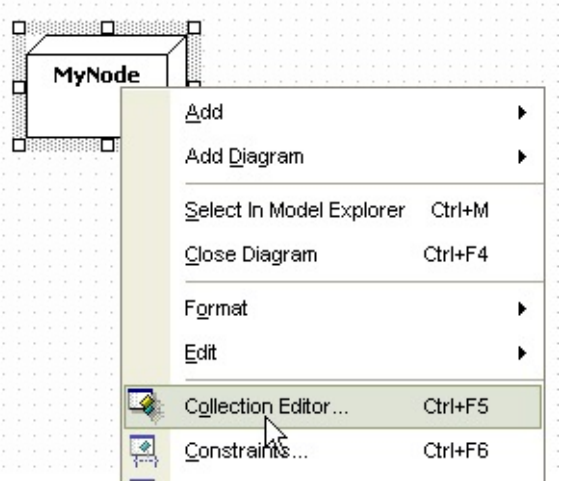
4. And press **[Enter]** key.



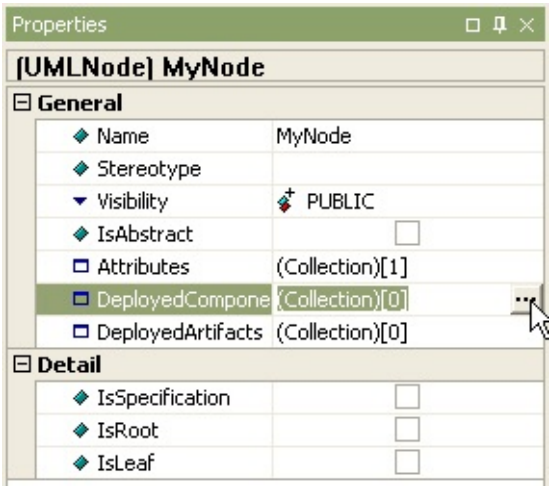
Procedure for adding deployed component

In order to add deployed component to node

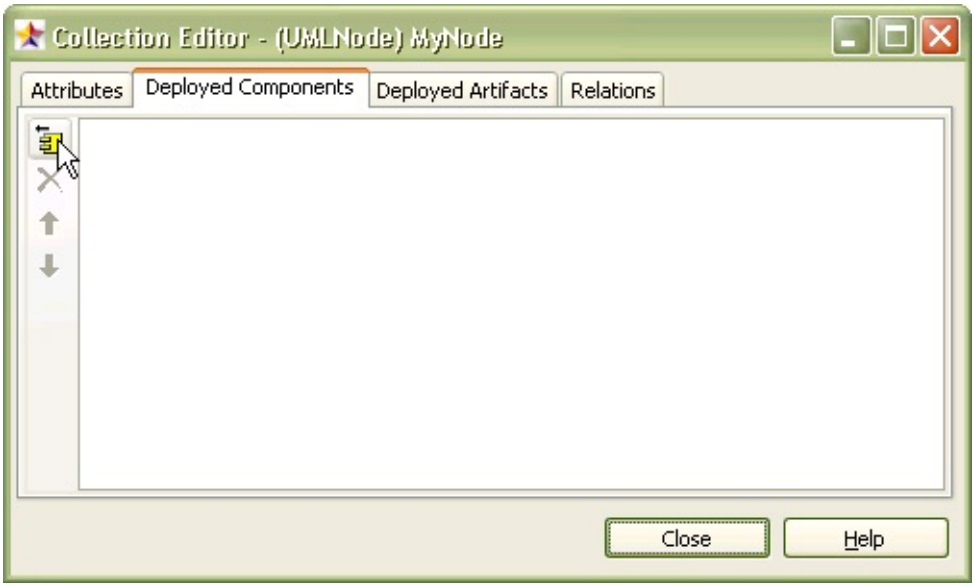
1. Select **[Collection Editor...]** popup menu of node.



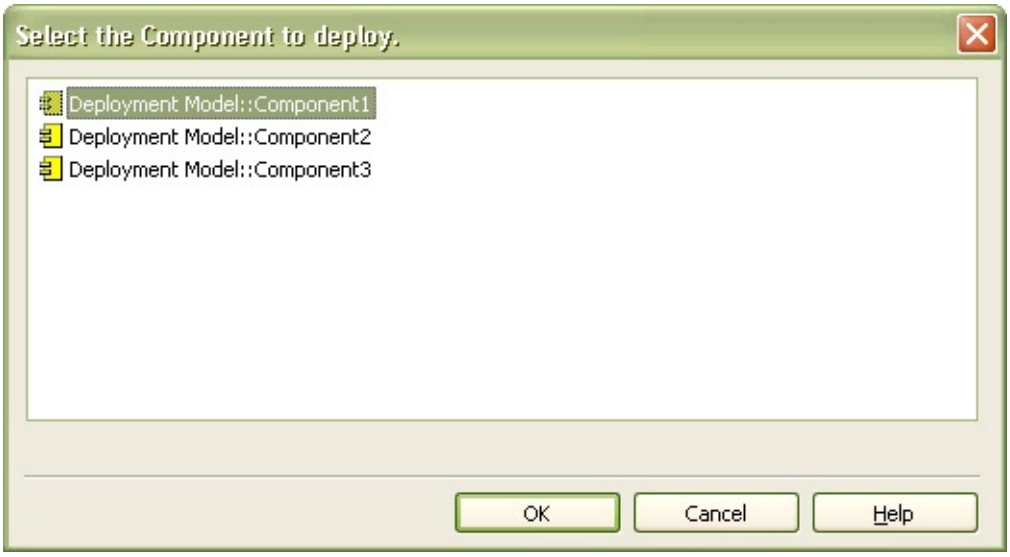
2. Or click ··· button in **[DeployedComponents]** property on properties window.



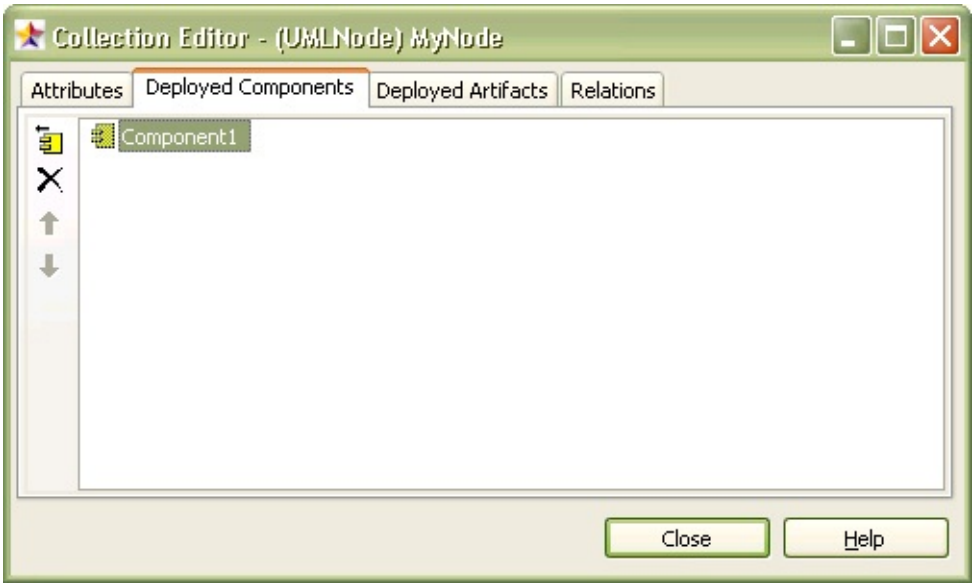
3. At the **[Deployed Components]** tab of the **[collection editor]**, you can add deployed component by using button.



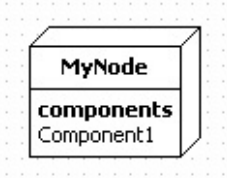
4. At **[Select the Component to deploy]** dialog, select deployed component. To select component, you have already made some component.



5. And click OK button. Then deployed component is added to the node.



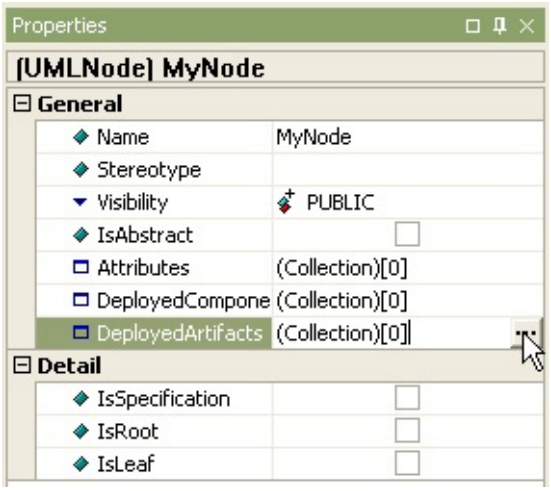
6. The node is shown as following.



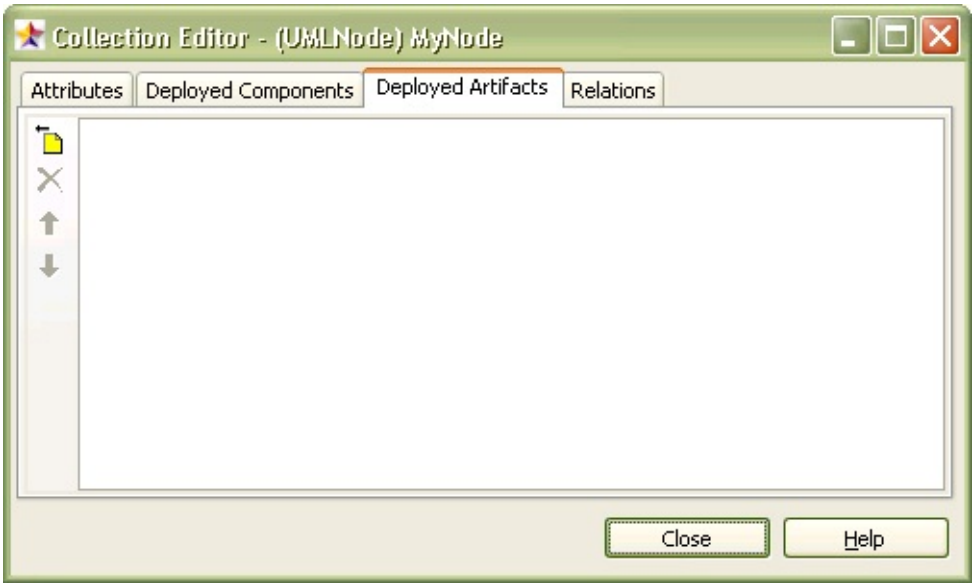
Procedure for adding deployed artifact

In order to add deployed artifact to node,

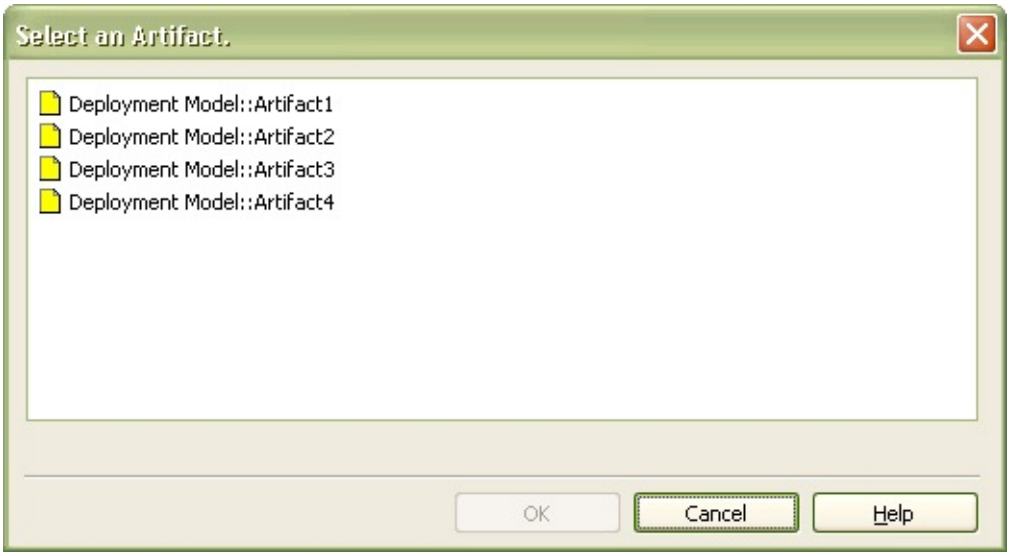
1. Select **[Collection Editor...]** popup menu of node.
2. Or click ··· button in **[DeployedArtifacts]** property on properties window.



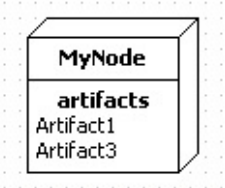
3. At the **[Deployed Artifacts]** tab of the **[collection editor]**, you can add deployed artifact by using  button.



4. At the **[Select a Artifact]** dialog, select a deployed artifact and click **[OK]** button.



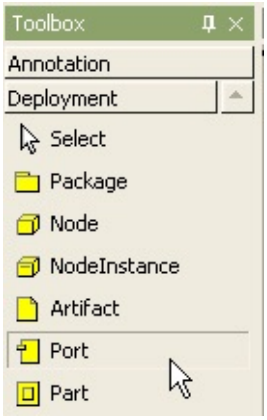
5. Then the artifact is add to the node and the node is shown as following.



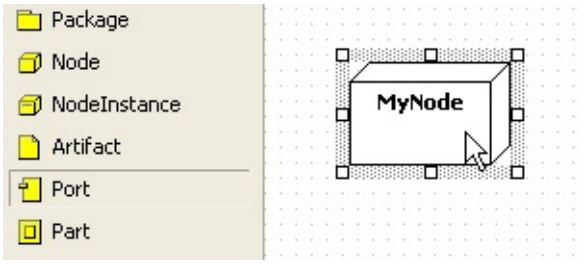
Procedure for creating port

In order to create port on a node,

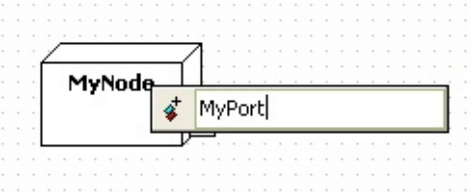
1. Click **[Toolbox]** -> **[Deployment]** -> **[Port]** button.



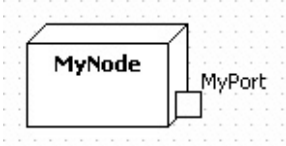
2. Click the node where the port will be contained in the **[main window]**.



3. A port is created on the node and the quick dialog appears. Enter the port name at the quick dialog.



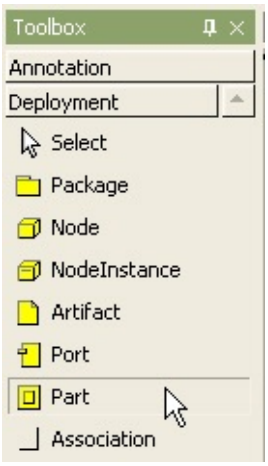
4. And press **[Enter]** key. The result is like the following.



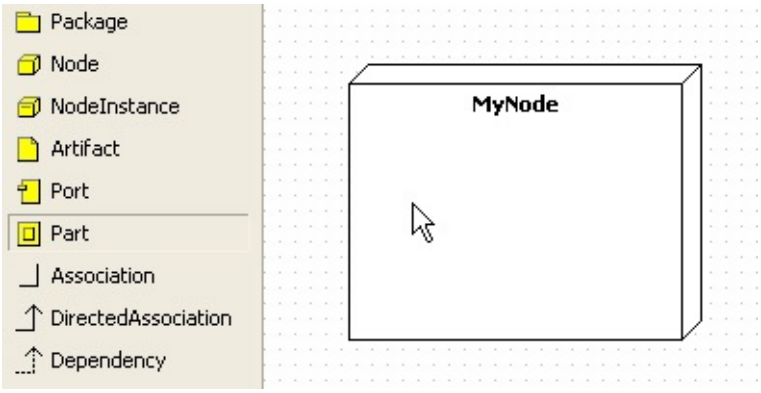
Procedure for creating part

In order to create part on a node

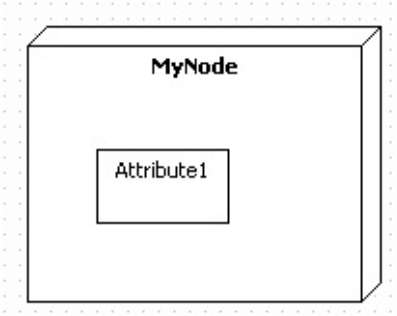
1. Click **[Toolbox]** -> **[Deployment]** -> **[Part]** button.



2. Click the node where the part will be contained in the **[main window]**.



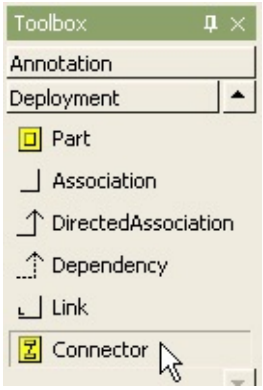
3. Finally, a part is created on the node as following.



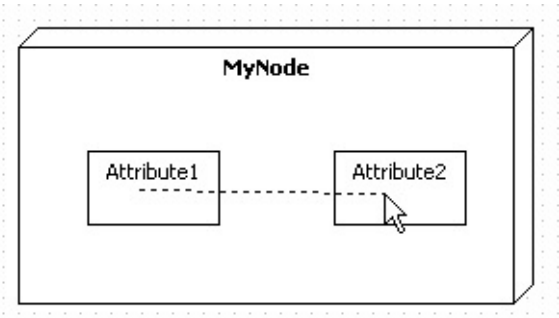
Procedure for creating connector

In order to create connector between two parts,

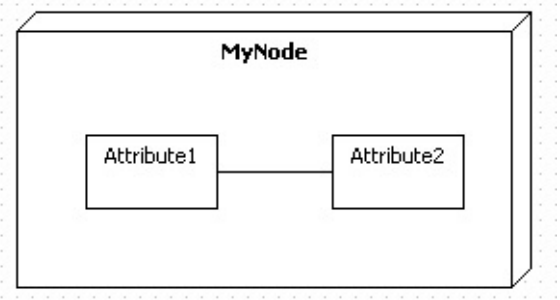
1. Click **[Toolbox]** -> **[Deployment]** -> **[Connector]** button.



2. Drag from one part and drop to the other part in the **[main window]**.



3. The result is as follows.



NodeInstance

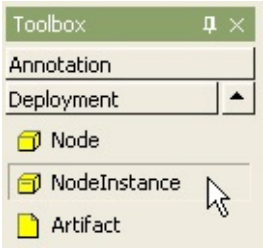
Semantics

A node instance is an instance of a node. A collection of component instances may reside on the node instance.

Procedure for creating node instance

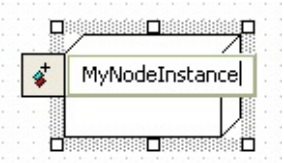
In order to create NodeInstance in deployment diagram,

1. Click **[Toolbox]** -> **[Deployment]** -> **[NodeInstance]** button.

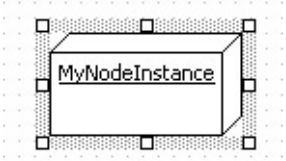


2. Click at the position where NodeInstance will be placed in the **[main window]**, a node is created, and quick dialog appears.

3. Enter the node instance name at the quick dialog and press **[Enter]** key.



4. The result is as follows.



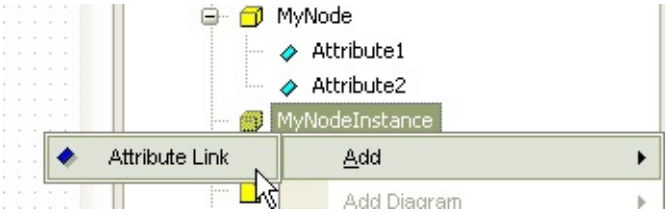
Procedure for adding attribute link to node instance

There are two way to add attribute link to node instance.

- using NodeInstance model in the **[main window]** or the **[model explorer]**
- using **[collection editor]**

In the case of using NodeInstance model

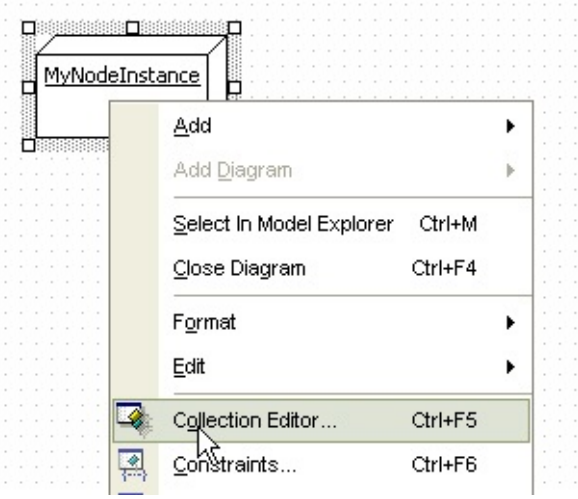
1. Select NodeInstance in the **[main window]** or in the **[model explorer]**,
2. Right-click the selected NodeInstance, select **[Add] -> [Attribute Link]** popup menu, and you can add Attribute Link.



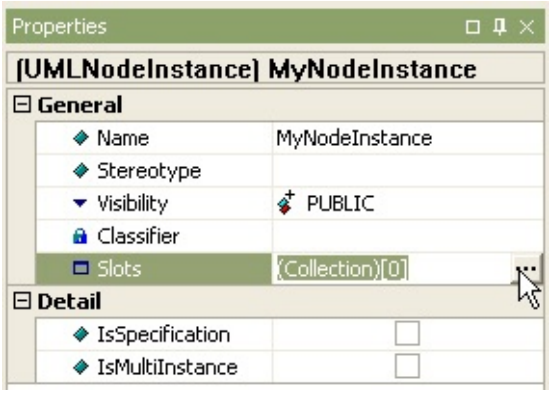
3. The node doesn't show attribute link on the view.

In the other case

1. Select **[Collection Editor...]** popup menu of NodeInstance.

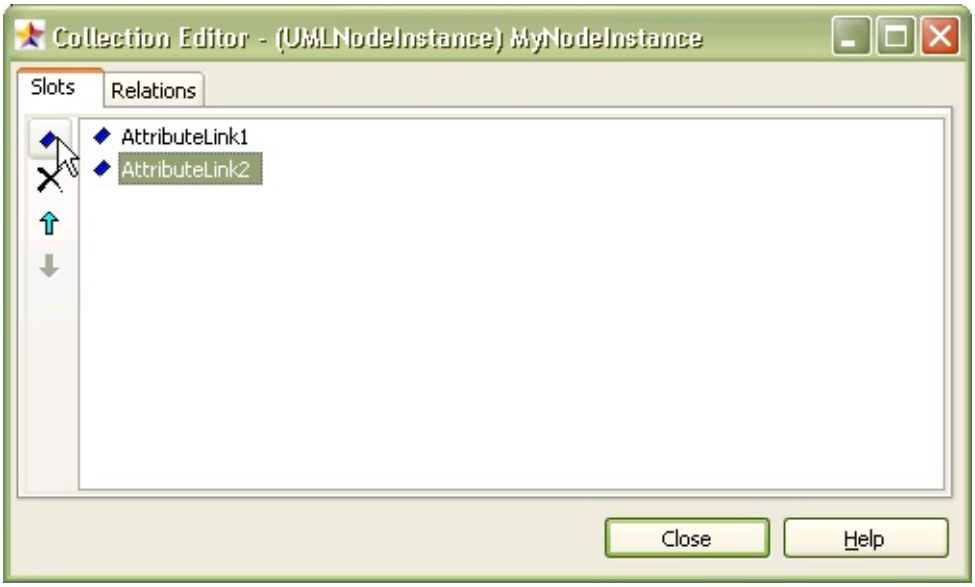


2. Or click ··· button in **[Slots]** property on properties window.



3. At **[Slots]** tab of the **[collection editor]**, you can add attribute link by using

 button.

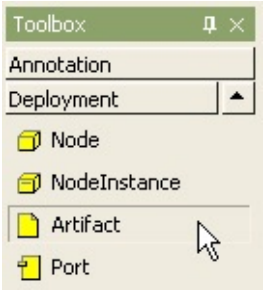

Collection Editor - (UMLNodeInstance) MyNodeInstance
Slots
Relations
AttributeLink1
AttributeLink2
Close
Help

Artifact

Procedure for creating artifact

In order to create Artifact,

1. Click **[Toolbox]** -> **[Deployment]** -> **[Artifact]** button.



2. Click at the position where Artifact will be placed in the **[main window]**.

3. At the quick dialog, enter the artifact name and press **[Enter]** key.

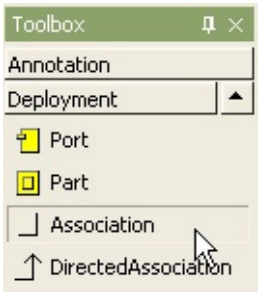
4. The result is as follows.

Association

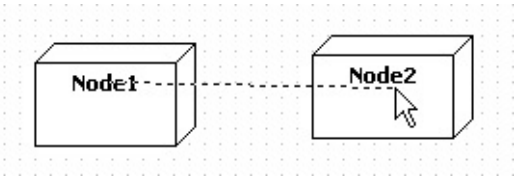
Procedure for creating association

In order to create association,

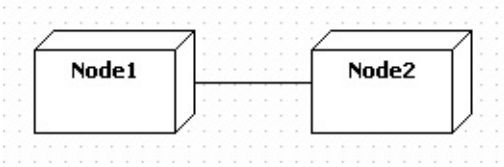
1. Click **[Toolbox]** -> **[Deployment]** -> **[Association]** button.



2. Drag from one associated and drop to another in the **[main window]**.



3. The result is as follows.

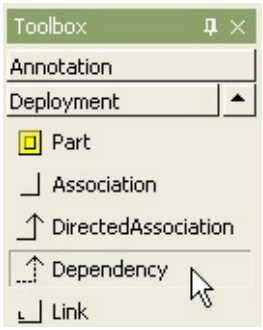


Dependency

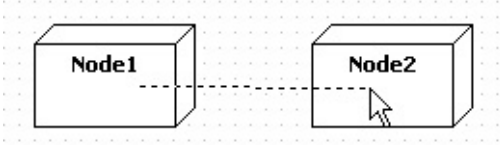
Procedure for creating dependency

In order to create dependency,

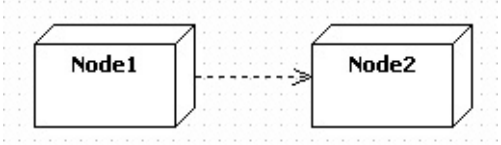
1. Click **[Toolbox]** -> **[Deployment]** -> **[Dependency]** button.



2. Drag and drop between elements in the **[main window]** in depending direction.



3. Then dependency between two elements is created as follows.

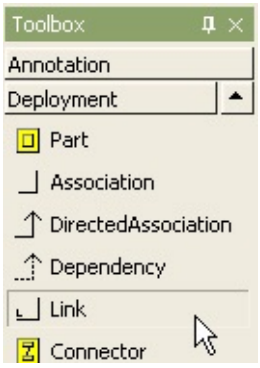


Link

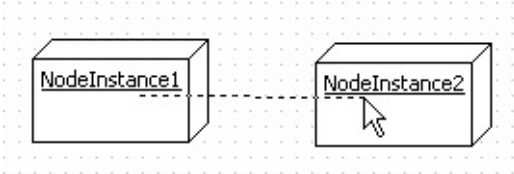
Procedure for creating link

In order to create Link,

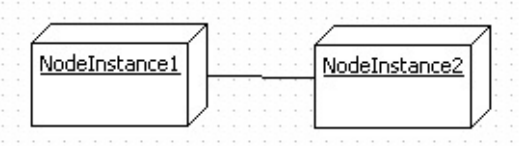
1. Click **[Toolbox]** -> **[Deployment]** -> **[Link]** button.



2. Drag from one NodeInstance and drop to the other NodeInstance in the **[main window]**.



3. Then the link between two node instances is created.



Modeling with Composite Structure Diagram

The following elements are available in a composite structure diagram.

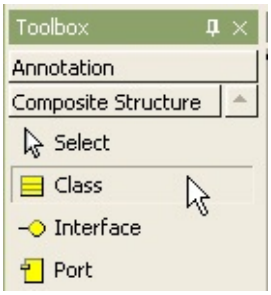
- Class
- Interface
- Port
- Part
- Dependency
- Connector
- Realization
- Collaboration

Class

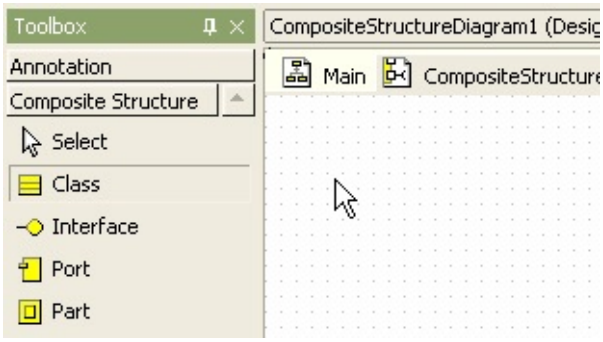
Procedure for creating class

In order to create Class in composite structure diagram,

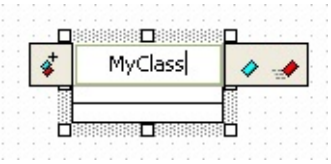
1. Click **[Toolbox] -> [Composite Structure] -> [Class]** button



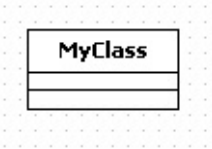
2. Click at the position where Class will be placed in the **[main window]**.



3. At the quick dialog, enter the class name.



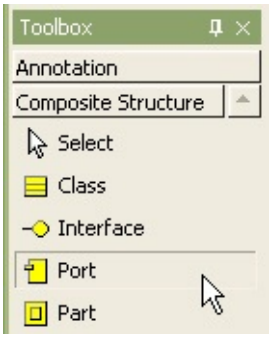
4. Press **[Enter]** key. Then a class is created finally.



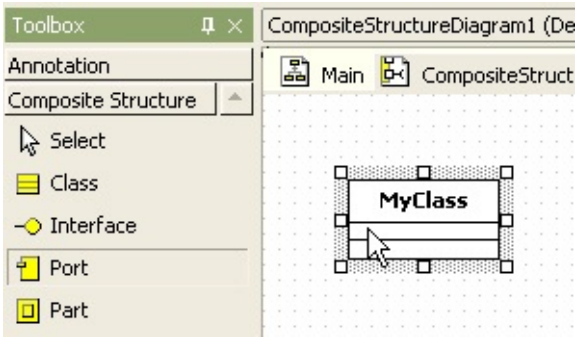
Procedure for creating port

In order to create port in composite structure diagram,

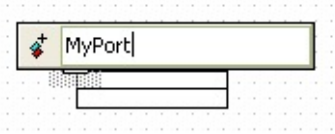
1. Click **[Toolbox] -> [Composite Structure] -> [Port]** button.



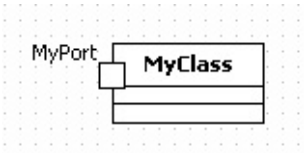
2. And click the class where the port will be contained in the **[main window]**.



3. At the quick dialog, enter the port name.



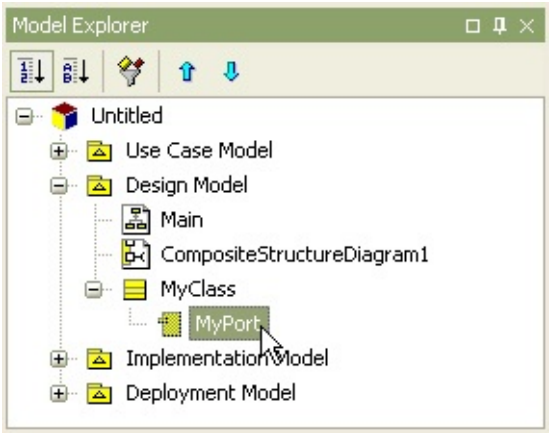
4. Press **[Enter]** key. Then a port is created.



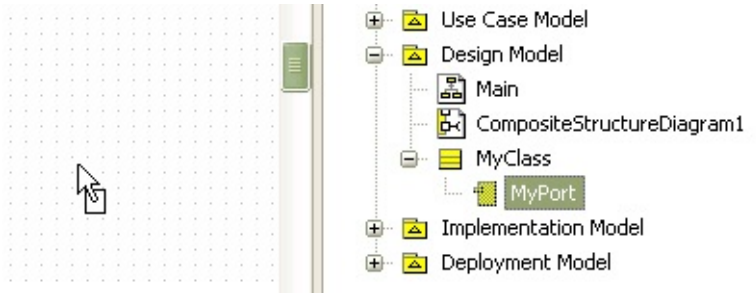
Procedure for creating view by draging port

You can create port by draging port from **[model explorer]** to main diagram.

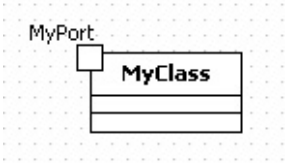
1. Click port in the **[model explorer]**.



2. Drag it and drop on the class in the composite structure diagram.



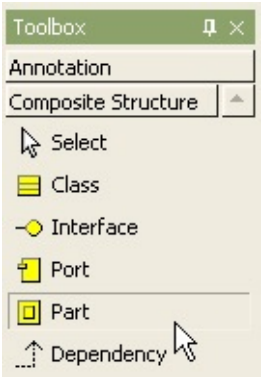
3. If it is not dropped on the component but on the other area of the diagram, component with port will be created.



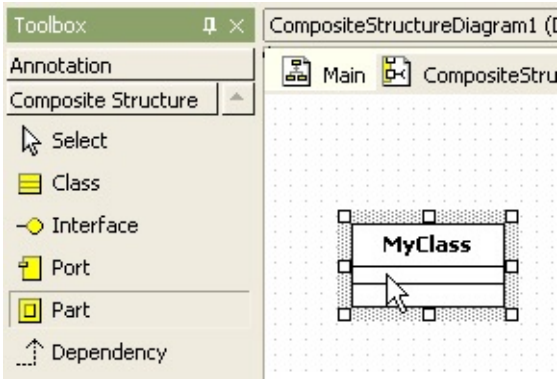
Procedure for creating part

In order to create part in composite structure diagram

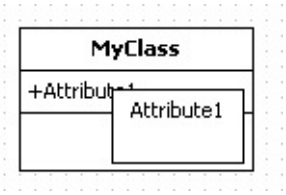
1. Click **[Toolbox] -> [Composite Structure] -> [Part]** button



2. Click a class where the part will be contained in the **[main window]**.



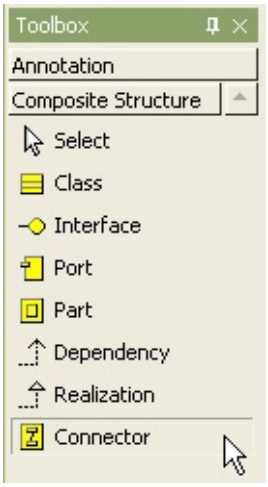
3. Then a part is created in the class.



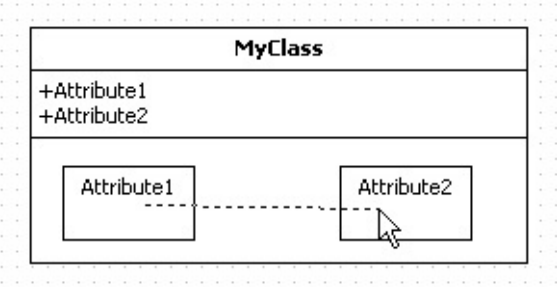
Procedure for creating connector

In order to create connector in composite structure diagram,

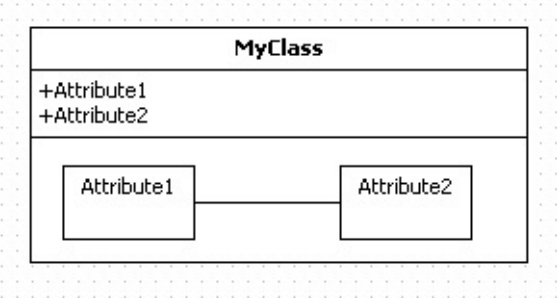
1. Click **[Toolbox]** -> **[Composite Structure]** -> **[Connector]** button.



2. Drag from one part and drop to the other part in the **[main window]**.



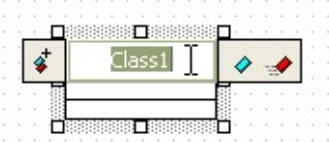
3. Then connector between two parts is created finally.



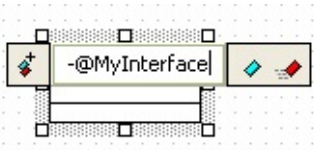
Procedure for creating providing interface of class.

In order to create providing inteface of class in composite structure diagram, use shortcut creation syntax.

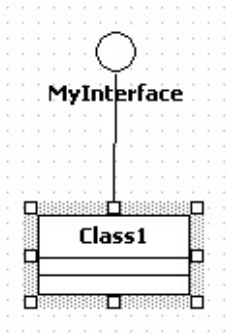
1. Double-click class and quick dialog appears.



2. At the quick dialog, enter "-@" staring and interface name, separate interface names by "," character.



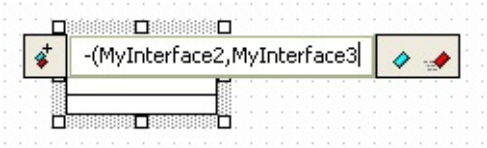
3. And press **[Enter]** key. Several interfaces provided by selected class is created and arranged automatically.



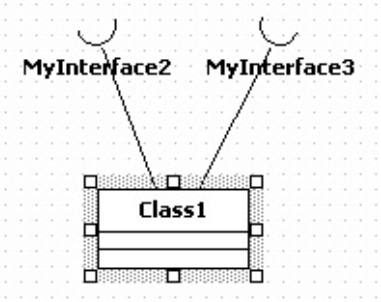
Procedure for creating requiring interface of class.

In order to create requiring inteface of class, use shortcut creation syntax.

1. Double-click class.
2. At the quick dialog, enter "-(" or "-->", and enter interface names, separate interface names by "," character.



3. And press **[Enter]** key. Several interfaces required by selected class is created and arranged automatically.

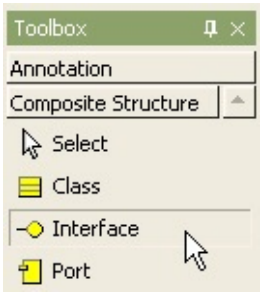


Interface

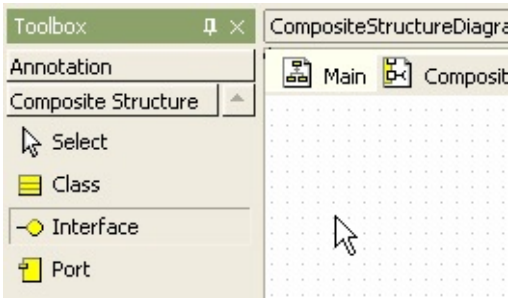
Procedure for creating interface

In order to create Interface in composite strucutre diagram,

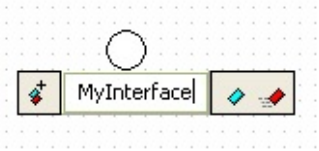
1. Click **[Toolbox] -> [Composite Structure] -> [Interface]** button.



2. Click at the position where Interface will be placed in the **[main window]**.



3. At the quick dialog, enter the interface name.

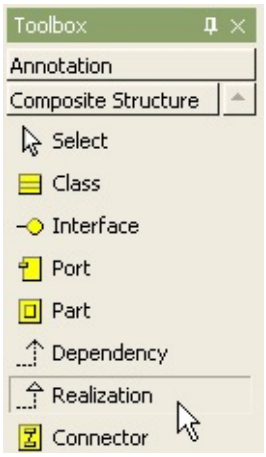


4. And press **[Enter]** key. Then interface creation procedure is done.

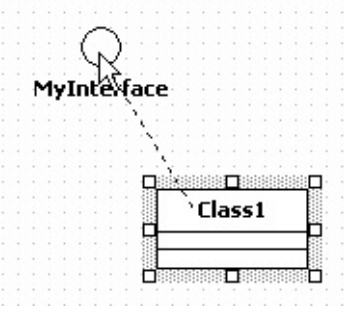
Procedure for creating providing relationship

In order to create providing relationship in composite structure diagram,

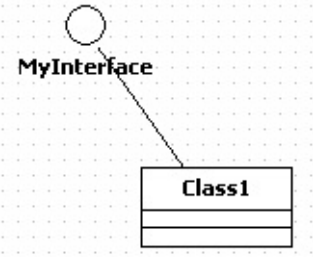
1. Click **[Toolbox] -> [Composite Structure] -> [Realization]** button.



2. Drag from element(Class, Port, Part, Package, Subsystem) and drop to interface in the **[main window]**.



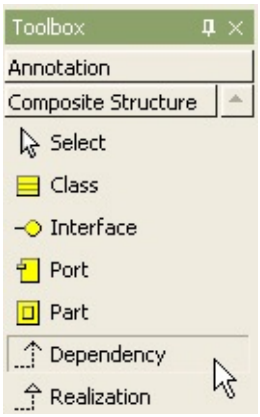
3. Then connection between two elements is created finally.



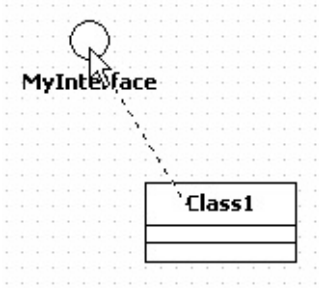
Procedure for creating requiring relationship

In order to create requiring relationship in composite dialog,

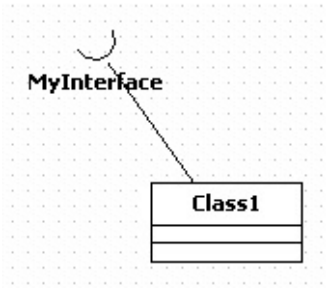
1. Click **[Toolbox] -> [Composite Structure] -> [Dependency]** button.



2. Drag from element(Class, Port, Part, Package, Subsystem) and drop to interface in the **[main window]**.



3. Then interface requiring relationship is created finally as following.

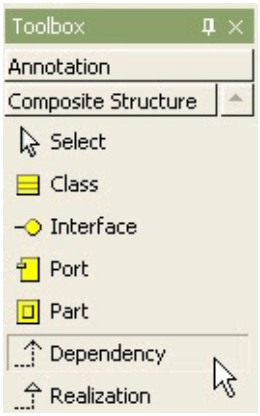


Dependency

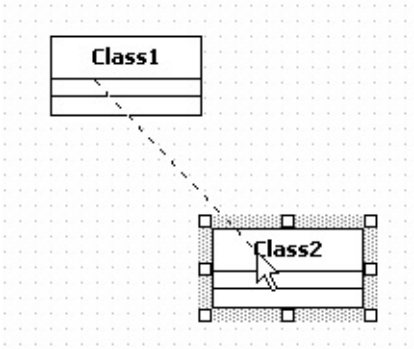
Procedure for creating dependency

In order to create dependency in composite structure diagram,

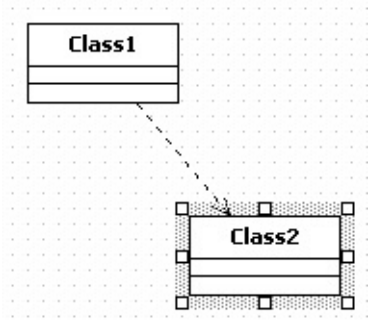
1. Click **[Toolbox] -> [Composite Structure] -> [Dependency]** button.



2. Drag and drop between elements in the **[main window]** in depending direction.



3. Then dependency is created as following.

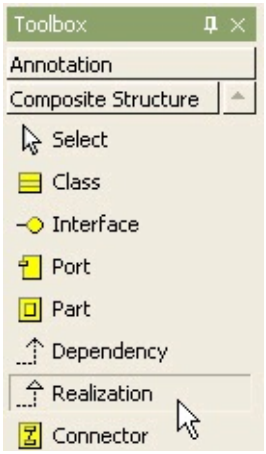


Realization

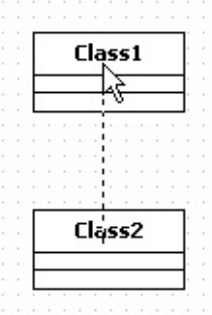
Procedure for creating realization

In order to create realization in composite structure diagram,

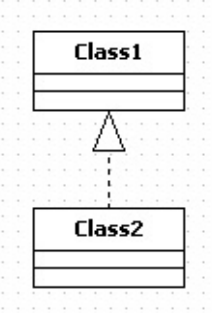
1. Click **[Toolbox] -> [Composite Structure] -> [Realization]** button



2. Drag and drop between elements in the **[main window]** in realization direction.



3. Then realization between two elements is created as following.



Collaboration

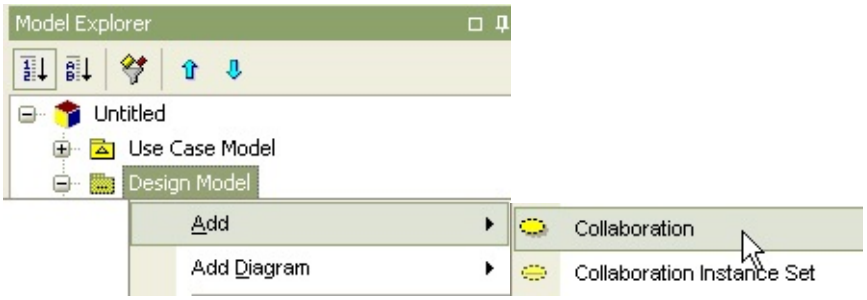
Semantics

Behavior is implemented by ensembles of Instances that exchange Stimuli within an overall interaction to accomplish a task. To understand the mechanisms used in a design, it is important to see only those Instances and their cooperation involved in accomplishing a purpose or a related set of purposes, projected from the larger system of which they are part of. Such a static construct is called a Collaboration.

Procedure for creating collaboration

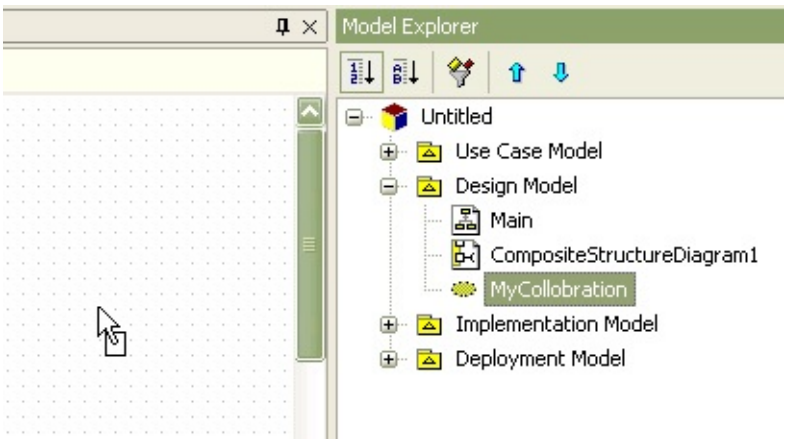
In order to create collaboration in composite structure diagram,

1. select package in the **[model explorer]**, right-click, and select **[Add] -> [Collaboration]** popup menu.



2. Then collaboration is created under the package in the **[model explorer]**. Enter the collaboration name.

3. And drag the collaboration and drop on the **[main window]**.



4. Then the collaboration is placed on the diagram.

Configuring StarUML

This chapter describes in detail the procedures for configuring StarUML™ environments and the available configuration option items.

- General Configuration
- Diagram Configuration
- General View Configuration
- Specific View Configuration

General Configuration

General Configuration is a group of the basic and general option items for the program. This category includes the **[General]** and **[Collection Editor]** subcategories.

[General] Options

Option Item	Default	Description
Max. number of undo actions	30	Specifies the maximum number of actions for undo and redo. The range for numbers of maximum undo is 1~100.
Recent project files	10	Specifies the maximum number of project files to be kept under the recent project menu item. The range for recent project files is 1~20.
Create backup files	True	Specifies whether to create backup files when saving changes.
Open the New Project dialog box at startup	True	Specifies whether to open the new Project dialog box at startup.

[Collection Editor] Options

Option Item	Default	Description
Show stereotypes	True	Specifies whether to show the stereotype name for each element in the collection editor.
Show visibility with icons	True	Specifies whether to show visibility for each element in the collection editor.
		Specifies whether to show the full expression or the

Show names only	False	name only for each element item in the collection editor (e.g. name or the full signature for an operation).

Diagram Configuration

Diagram Configuration is a group of the general option items related to diagrams. This category includes the **[Diagram Size]**, **[Grid]** and **[Interaction Diagram]** subcategories.

[Diagram Size] Options

Option Item	Default	Description
Default diagram width	5000	Specifies the maximum diagram width. Adjust this value if the diagram area is not large enough. The range for default diagram width is 100~50000.
Default diagram height	5000	Specifies the maximum diagram height. Adjust this value if the diagram area is not large enough. The range for default diagram height is 1~5000.

[Grid] Options

Option Item	Default	Description
Grid width	4	Specifies the width of the grid used for editing diagram. The range for grid width is 1~20.
Grid height	4	Specifies the height of the grid used for editing diagrams. The range for grid height is 1~20.
Show grid	True	Specifies whether to show the grid in diagrams.

[Interaction Diagram] Options

Option Item	Default	Description
Message signature	Hide	Specifies how the messages/stimuli will be indicated in sequence or collaboration diagrams (hide, show type only, show name only, and show name and type).
Show sequence	True	Specifies whether the message/stimulus sequence number

number		is shown in sequence or collaboration diagrams.
Show Activation	True	Specifies whether to show activation by message/stimulus in sequence diagrams.

General View Configuration

General View Configuration is a group of the basic and general option items related to view elements. This category includes the **[Default View Style]** and **[Default View Format]** subcategories.

[Default View Style] Options

Option Item	Default	Description
Default fill color	$00B9FFFF	Specifies the default fill color for view elements (default is light yellow).
Default line color	$00000080	Specifies the default line color for view elements (default is maroon).
Default font name	Tahoma	Specifies the default font face for view elements.
Default font size	8	Specifies the default font size for view elements. The range for default font size is 1~50.
Default font color	$00000000	Specifies the default font color for view elements (default is black).

[Default View Format] Options

Option Item	Default	Description
Line style	Rectilinear	Specifies the Line Style for connection elements (either rectilinear or oblique).
Show stereotype	text	Specifies the default stereotype indication method (text, icon, or hide).
Show parent name	False	Specifies whether to show the name of the parent element that contains the model element represented by the view element.
Automatic resize	False	Specifies whether to automatically resize view elements.
Show compartment visibility	True	Specifies whether to show compartment visibility for view elements (e.g., attribute compartment, operation compartment, etc.).

Show compartment stereotype	True	Specifies whether to show compartment stereotype for view elements (e.g., attribute compartment, operation compartment, etc.).
Show operation signature	True	Specifies whether to show signature when expressing operation elements.
Show property	False	Specifies whether to show the property items (e.g. tagged values, changeability attribute, etc.) included in view elements.
Suppress attribute	False	Specifies whether to suppress the attributes for class type view elements.
Suppress operation	False	Specifies whether to suppress the operations for class type view elements.

Specific View Configuration

Specific View Configuration is a group of the option items related to specific views. This category includes the **[UseCase View]**, **[Actor View]**, **[Enumeration View], [Interface View], [Artifact View], [Component View / Component Instance View]** and **[Node View / Node Instance View]** subcategories.

[UseCase View] Option

Option Item	Default	Description
Show stereotype	Text	Specifies the default stereotype indication method for UseCase view elements (Text, Icon, None, Decoration or hide).
Suppress attribute	True	Specifies whether to suppress the attributes for UseCase view elements.
Suppress operation	True	Specifies whether to suppress the operations for UseCase view elements.

[Actor View] Options

Option Item	Default	Description
Show stereotype	Text	Specifies the default stereotype indication method for Actor view elements (Text, Icon, None, Decoration or hide).
Suppress attribute	True	Specifies whether to suppress the attributes for Actor view elements.
Suppress operation	True	Specifies whether to suppress the operations for Actor view elements.

[Enumeration View] Options

Option Item	Default	Description

Suppress literal	False	Specifies whether to suppress the literals for enumeration view elements.

[Interface View] Options

Option Item	Default	Description
Show stereotype	Icon	Specifies the default stereotype indication method for interface view elements (Text, Icon, None, Decoration or hide).
Suppress attribute	True	Specifies whether to suppress the attributes for interface view elements.
Suppress operation	True	Specifies whether to suppress the operations for interface view elements.

[Artifact View] Options

Option Item	Default	Description
Show stereotype	Decoration	Specifies the default stereotype indication method for artifact view elements (Text, Icon, None, Decoration or hide).
Suppress attribute	True	Specifies whether to suppress the attributes for interface view elements.
Suppress operation	True	Specifies whether to suppress the operations for interface view elements.

[Component View / ComponentInstance View] Options

Option Item	Default	Description
Show stereotype	Text	Specifies the default stereotype indication method for component and ComponentInstance view elements (Text, Icon, None, Decoration or hide)

[Node View / NodeInstance View] Options

Option Item	Default	Description
Show stereotype	Text	Specifies the default stereotype indication method for node and NodeInstance view elements (Text, Icon, None, Decoration or hide)

Managing Modules

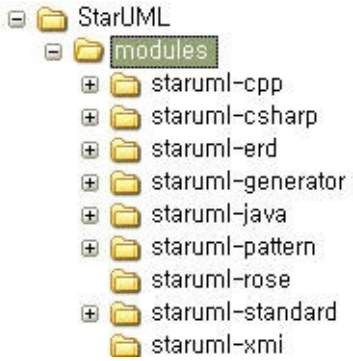
This chapter contains how to manage modules. Included are installing module, component of module and remove of module.

- Installing a Module
- Uninstalling a Module

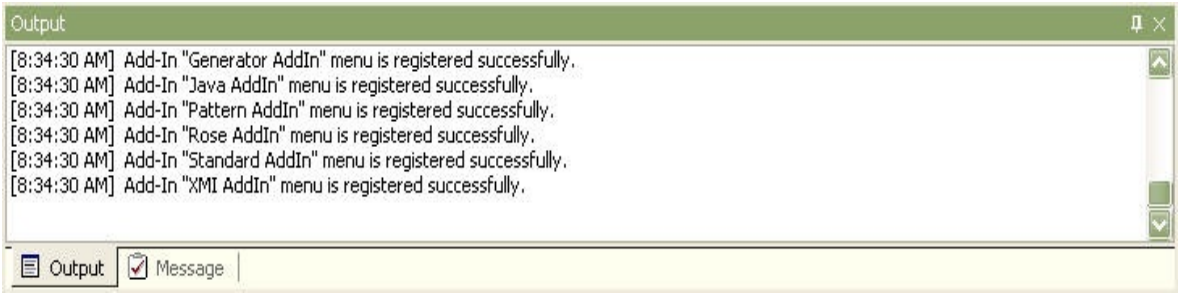
Installing a Module

The method to install the module in StarUML™ is very simple. The user copy a module directory which wants the install at the StarUML™ Module directory beneath. StarUML™ is running.

1. The module to install download from StarUML site(www.staruml.com).
2. Copys downloaded module file under {Installed path of StarUML}\modules directory. If it is zip fie, unzip it in a folder as creating a folder as zip file name. If installed path of StarUML is C:\Program Files\StarUML, copys the folder under C:\Program Files\StarUML\modules.



3. When StarUML is running, stop the running and rerun it.
4. Log about Module install at Output window is marked if Module was installed accurately.



Logs of module

The Log as Loading of Module show the Output window as follows. For detailed descriptions of module, see the module of Basic Concept in the second chapter.

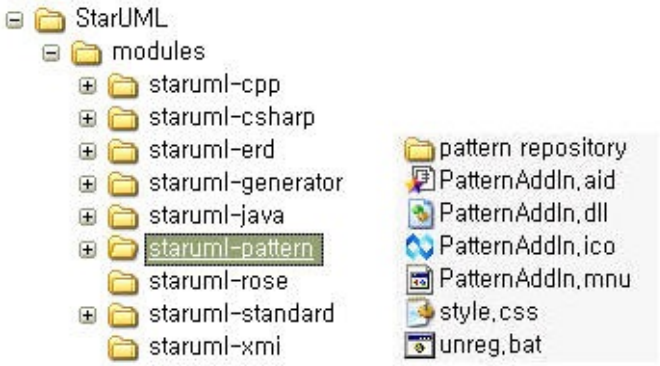
Component	State	Log
Profiles(.prf)	Successed reading profile file.	The Profile "..." is loaded successfully.
	Failed to load the profile file.	Failed to load the Profile "...".
Approaches(.apr)	Successed reading approach file.	The Approach "..." is loaded successfully.
	Failed to load the approach file.	Failed to load the Approach "...".
Framework(.frw)	Successed reading framework file.	The Framework "..." is loaded successfully.
	Failed to load the framework file.	Failed to load the Framework "...".
AddIn(.aid)	Registered in AddIn successfully.	AddIn "... AddIn" is registered successfully.
	Registered other DLLs in AddIn successfully.	AddIn sub module "....dll" is registered successfully.
	Successed reading add-in file.	Add-In "..." is loaded successfully.
	Failed to load error in the add-in file.	Failed to load the Add-In "...".
Menu(.mnu)	Successed reading menu file.	Add-In "... AddIn" menu is registered successfully.
	Failed to load error in the menu file.	Failed to load the menu "...".

Uninstalling a Module

Uninstalling a Module including in AddIn

In the case of a module including in AddIn, the directory in the module has .aid file, and there is stating a path of the AddIn in AID file. These modules provide their uninstall methods. In case of Core Module in StarUML, unreg.bat file is existed in under the relative module directory to remove installed addin. The user can remove the installed AddIn as executing unreg.bat. And the relative module.

1. Move to Uninstalling a Module



2. Run unreg.bat in under the module directory

3. Removing the relative module directory.

Uninstalling a Module exclude in AddIn

In the case of a module exclude in AddIn, the use can uninstall for the installed module by removing the relative module directory.

Generating Codes and Documents

This chapter describes basic concept of generator, how to generate codes and documents, and how to use batch processing.

- Basic Concept
- Generating by Templates
- Using Batches
- Installing and Uninstalling Templates

If you want to write your own template for codes or documents, please refer to Developer Guides.

Basic Concept

What is Generator?

StarUML Generator is platform module to generate various artifacts (like as Microsoft Word, Excel, PowerPoint, and Text-based artifacts) by templates depending on UML model elements in StarUML. User can define his/her own templates and can apply many different kinds of templates to the same UML model, so user can get various artifacts automatically, easily and in fast.

Key Features

StarUML Generator provides following features.

User-definable Template

Template can be defined by user. You can write templates for .doc, .xls, .ppt directly using MS Word, MS Excel, MS PowerPoint without extra template designer.

Parameters for Template

Template provides parameters for variations of user environments, objectives, and so on. Through parameterized template, you can eliminate inconvenience and can avoid defining a new template caused by a little difference.

Batch processing to generate many artifacts at once

You can generate many kinds of artifacts at once using Batch feature. You can register many templates as a Batch and can generate it at once. Using Batch, a large amount of artifacts can be generated without waiting so you can take a rest.

Support native-styles of MS Word like as Header/Footer

You can put generation commands in Header/Footer in MS Word template and can use MS Word's native styles in the template.

Support MS Excel Sheets

You can collect various data from UML model and can insert the data into the Cells of Excel Sheet. Using it, you can get good reports by using Graph, Filtering, Sorting and other functions in the Excel.

Support MS PowerPoint Slides

It is allowed to generate slides by hierarchical structure in MS PowerPoint. There is no restriction making PowerPoint slides, so you can generate various slides for presentation automatically with reduction of writing efforts.

Support Anything of text-based artifacts

You can generate any text-based artifacts like as XML, HTML, Source Codes (Java, C#, C++, ...), DB Schema, and so on.

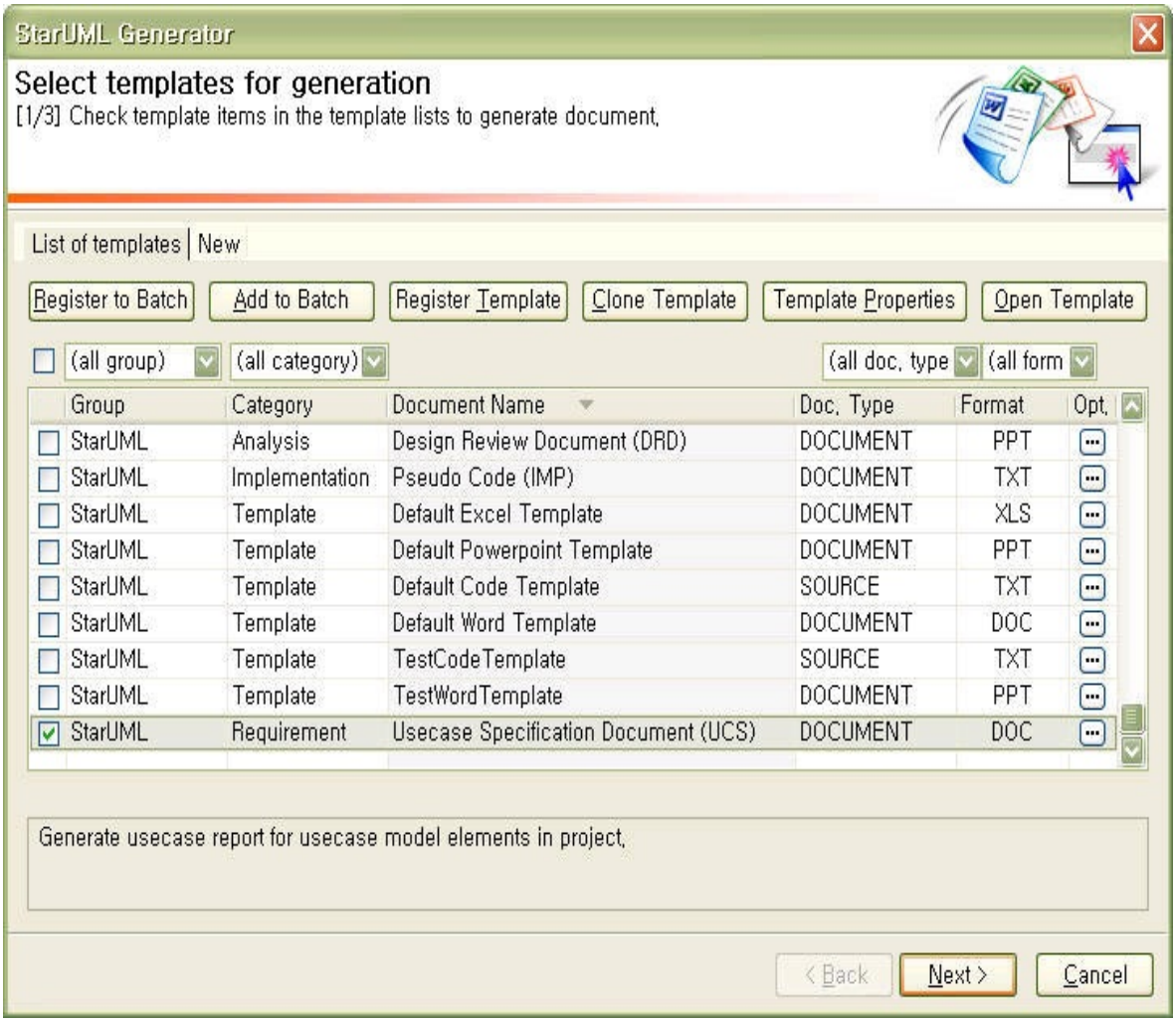
Generating by Templates

Generating Artifacts

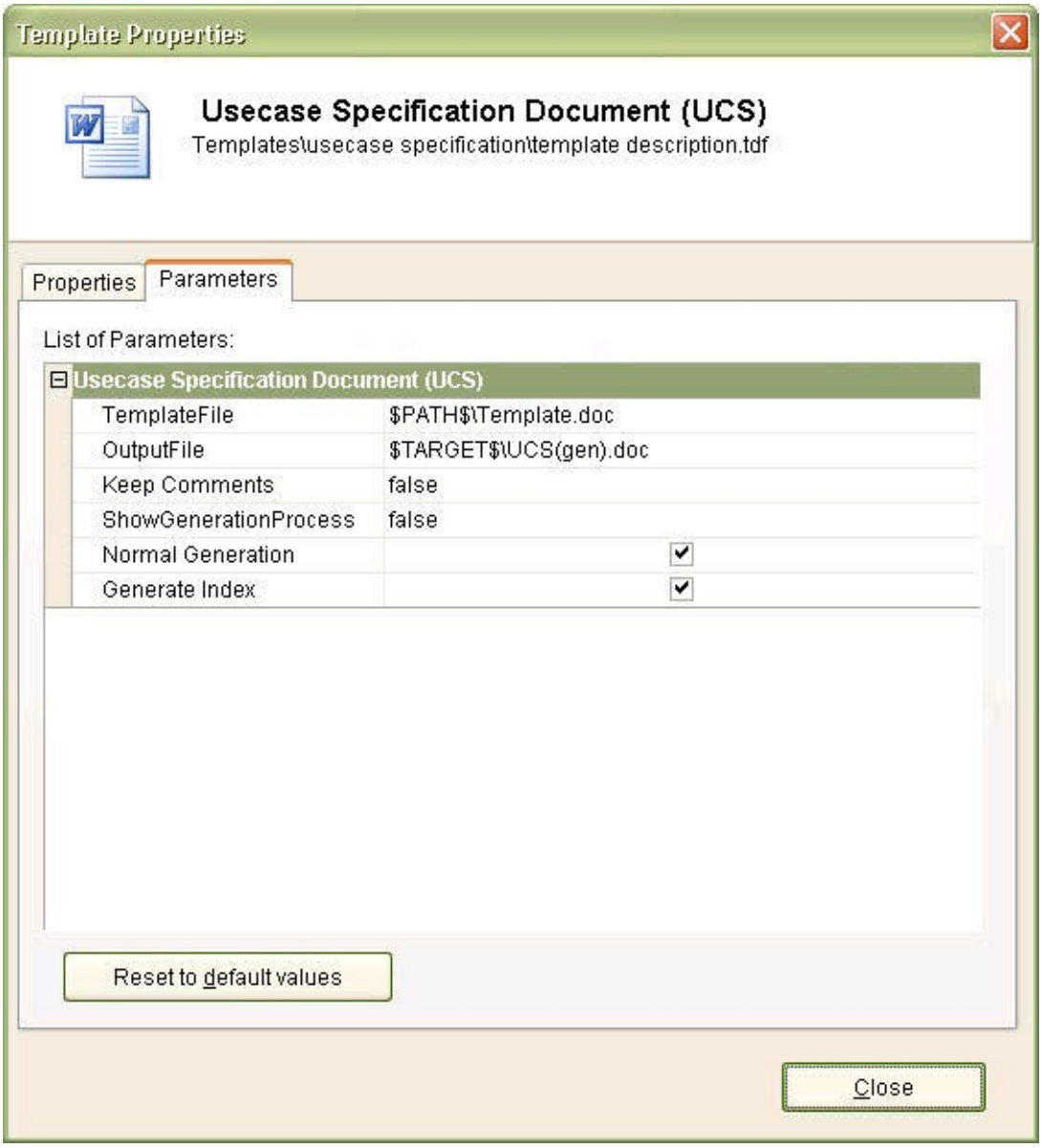
To generate artifacts by template, it must be applicable to current working UML model.

1. Select **[Tools]->[StarUML Generator...]** Menu

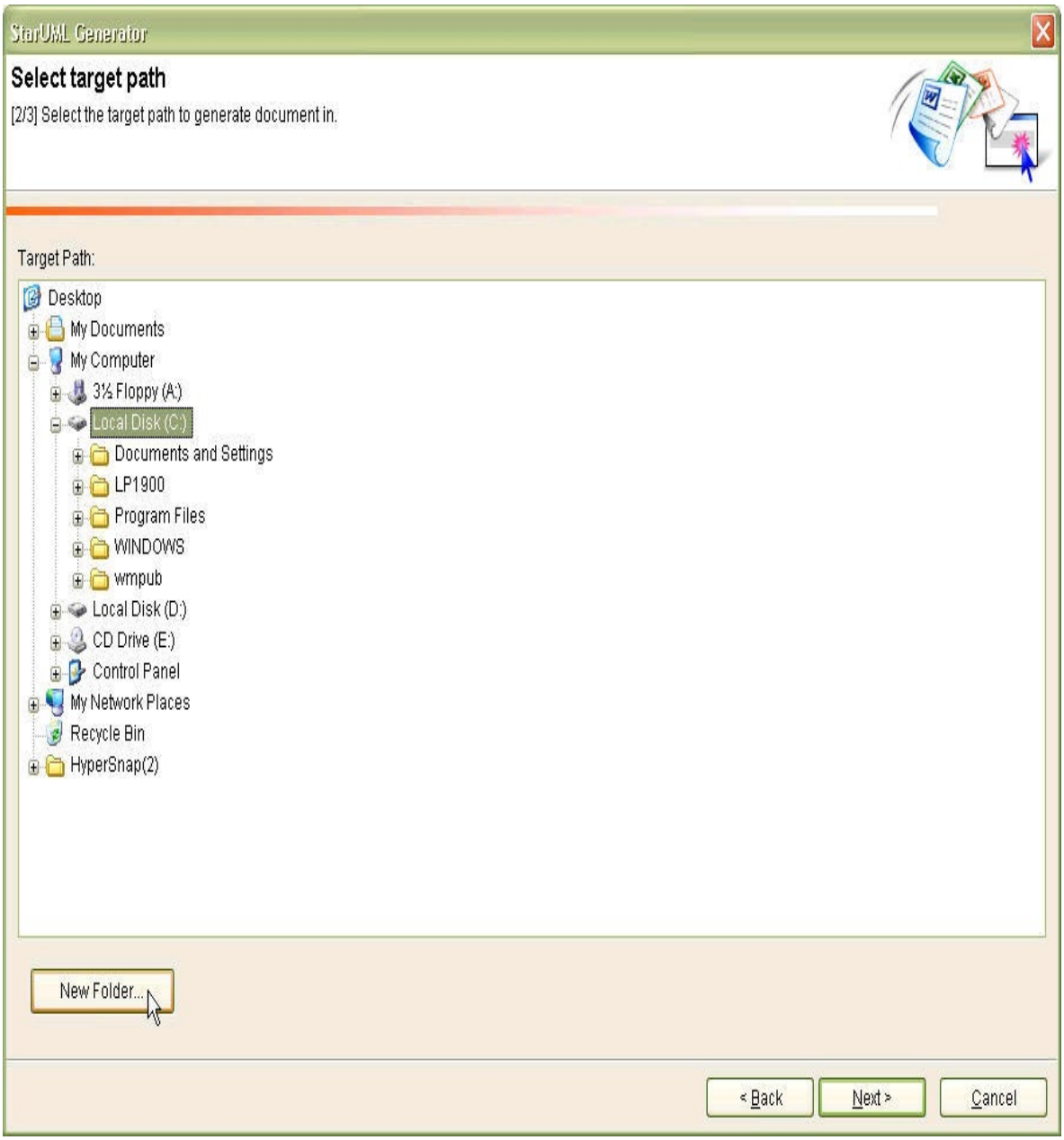
2. In the **[Select templates for generation]** Page, Check templates to generate in ListBox and Click **[Next]** Button.



3. To bind values with parameters, Click ... Button of each template item in the ListBox, and set values of parameter as you want. (Refer to **Registering Template** for more information about template parameters)

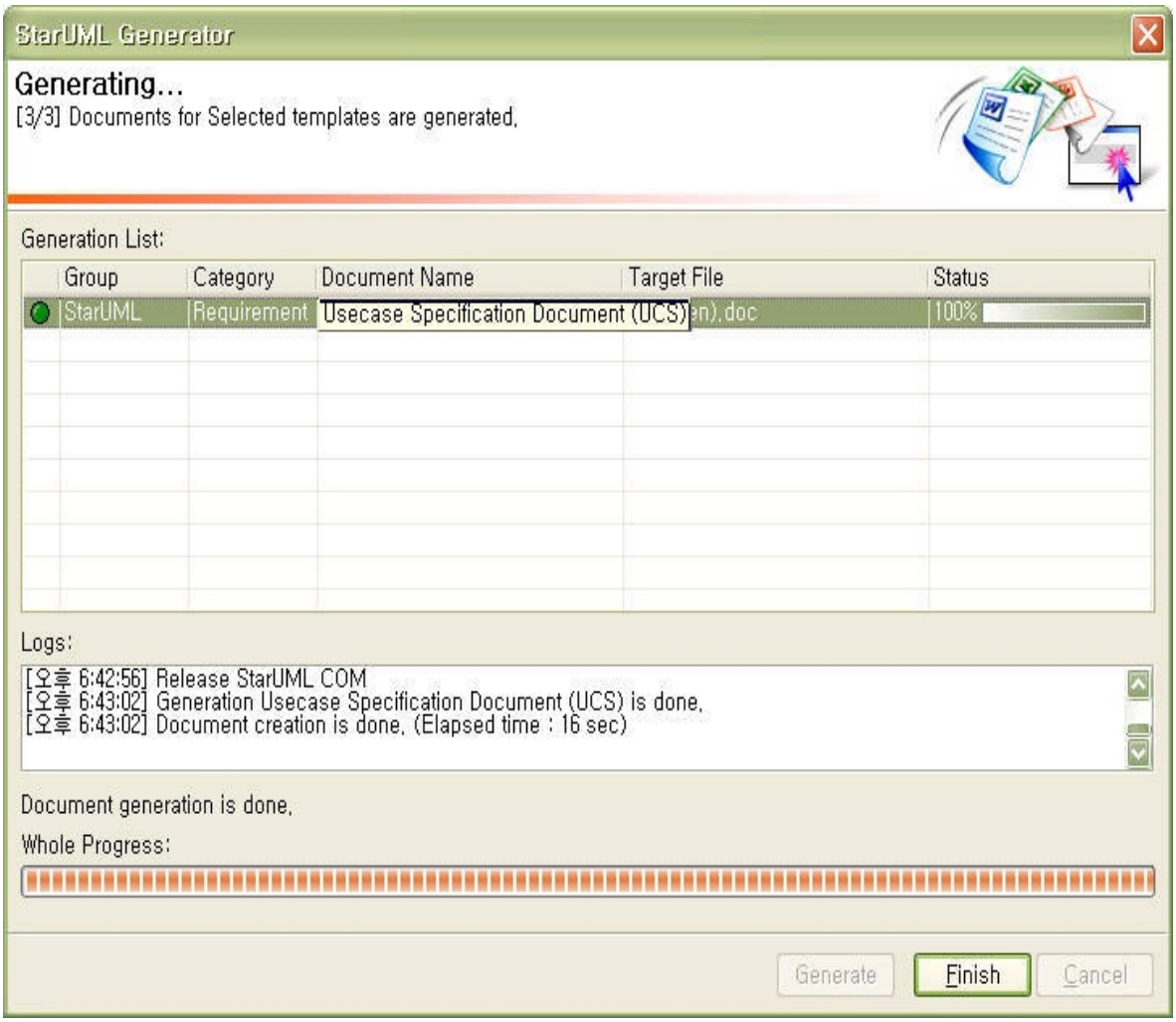


4. In the **[Select target path]** Page, Select a folder that generated artifacts will be placed and click **[Next]** button.



If you want to create a new folder, click **[New Folder...]** button and input nai
of the new folder.

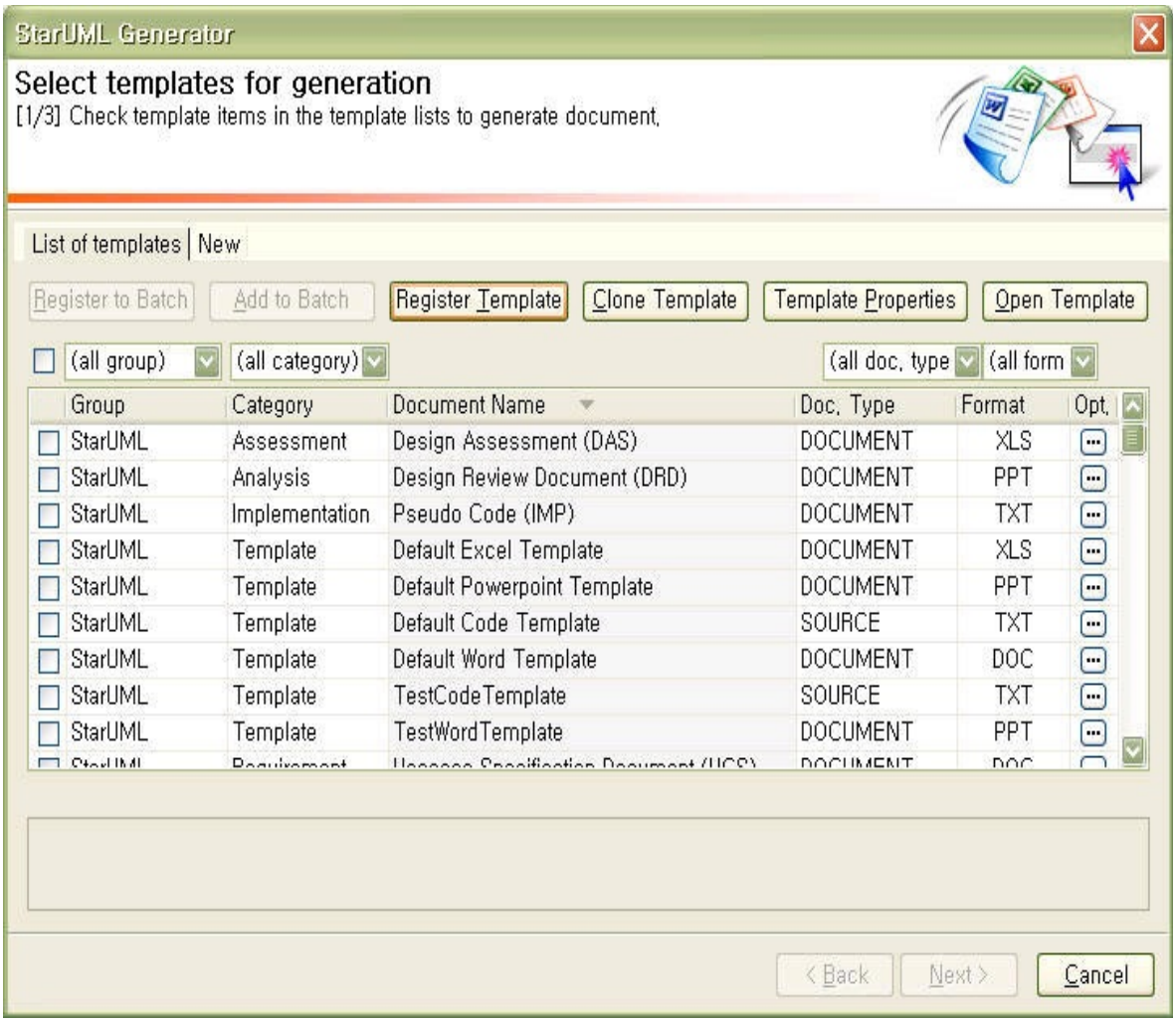
5. In the **[Generating...]** Page, click **[Generate]** button. You can check the progr of generation and it will be logged on **Logs**. If you want to cancel the generati process, click **[Cancel]** button. When all artifacts are successfully generated, **[Finish]** will be enabled and clicking it will finish the artifact generation. To s generated artifacts, double-click the item that want to see in the **[Generation I** then the generated artifact will be opened.



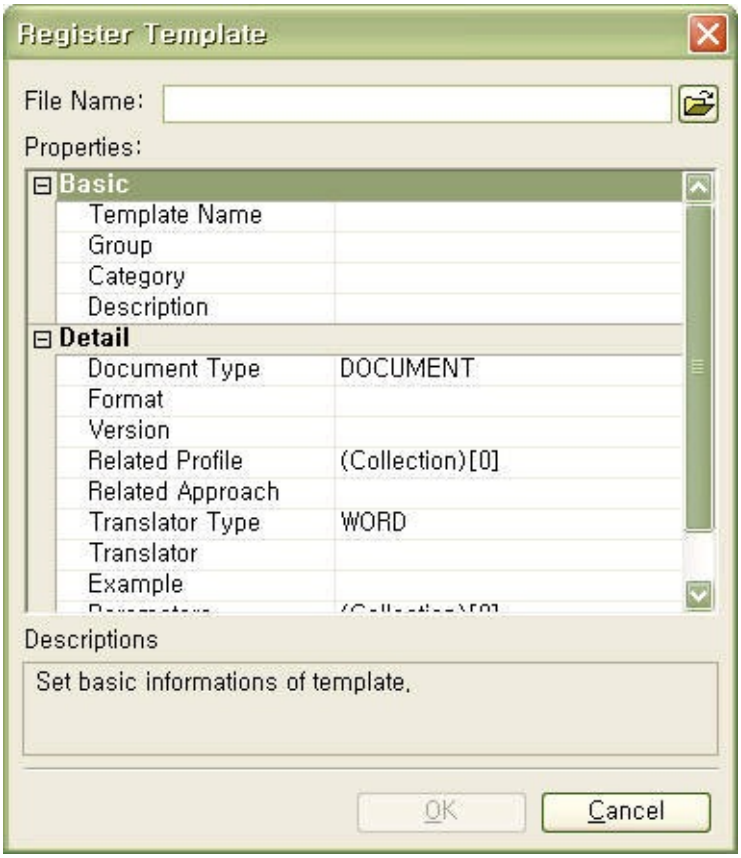
Registering a Template

Your own templates can be registered in StarUML Generator.

1. In the **[Select templates for generation]** Page, click **[Register Template]** but



2. In the **[Register Template]** Dialog, click button and select a folder that the template files will be placed.



3. Input template information on **[Properties:]** and click **[OK]** button to complet registering a template.

[Basic] property section

Basic properties for template registration.

Property	Description
Template Name	Name for the template to register.
Group	Group name for the template. There is no restriction t name a group but to group a set of template, give the same group name for the set of templates. (it is used f horizontal classification like as RUP, CBD, <ModuleName>, <CompanyName>, ...)
Category	Category name for the template. There is no restrictic name a category but to categorize a set of template, g the same category name for the set of templates. (it is used for vertical classification like as Requirements,

	Design, Code, Analysis, ...)
Description	Brief description of the template.

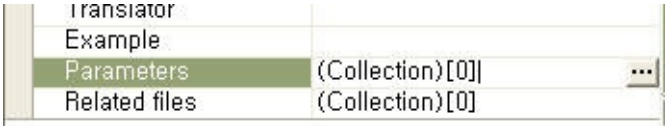
[Detail] property section

Detailed properties for template registration.

Property	Description
Document Type	Type of the template. Select DOCUMENT or CODE.
Format	Type of generated artifact. Input the format name or s one of the already defined formats (TXT, DOC, PPT, XLS)
Version	Version of the template (e.g) 1.0
Related Profile	Profiles related to the template.
Related Approach	Approach related to the template. (it is a declarative property, so it will not effect anything)
Translator Type	Kind of translator for the template. Select one of the following: WORD, EXCEL, POWERPOINT, TEXT, COM(user-defined COM-based generator), SCRIPT(u defined scripts like as JScript, VBScript, ...), EXE(use defined .EXE-based generator).
Translator	Specify filename of user-defined translator. It is used for user want to use his/her own translator not built-in translators(WORD, EXCEL, POWERPOINT, TEXT)
Example	If any, specify an example model for the template.
Parameters	Parameters required for the template.
Related files	If any, specify all related files to the template.

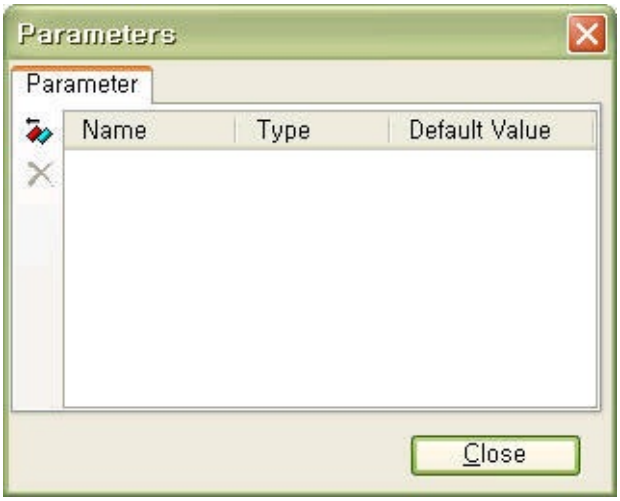
[Parameters] property

1. Click ··· button in Parameters property.

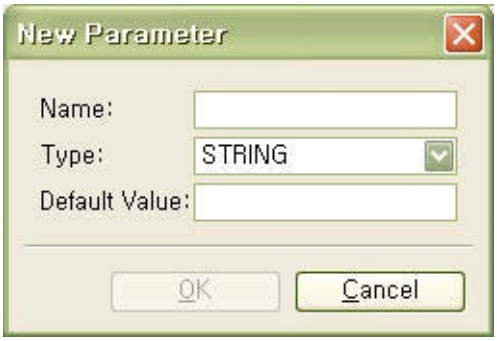


2. In the **[Parameters]** Dialog, click button to create a new parameter and cli

button to delete a existing parameter.



3. In the **[New Parameter]** Dialog, specify Name, Type and Default Value for th parameter and click **[OK]** button.



Default parameters are different according to the Translator Type. Following are default parameters for each Translator Types.

Property	Type	Translator Type	Description
TemplateFile	FILENAME or STRING	WORD,EXCEL, POWERPOINT	Specify file name of the template document.
OutputFile	FILENAME or STRING	WORD,EXCEL, POWERPOINT, TEXT	Specify file name of the generated artifact.
Keep Comment	BOOLEAN	WORD,EXCEL, POWERPOINT	Specify whether to remai the comment used for generation or to delete it.

ShowGenerationProcess	BOOLEAN	WORD,EXCEL, POWERPOINT	Specify whether to show progress of generation or It may affect the perform of the generation.
Normal Generation	BOOLEAN	WORD	If true, the template is ap to top-level package (Project). if false, the template is applied to the package (or element) that currently selected in StarUML.
Generate Index	BOOLEAN	WORD	Specify whether to gener Index or not.
intermediate	STRING	TEXT	Specify file name of intermediate file used for generation.
target	STRING	TEXT	If more than two artifacts generated, specify the pathname the artifacts are placed.

Note:

To specify file name in the parameters, environment variables is required of Star Generator. The variable is as follow.

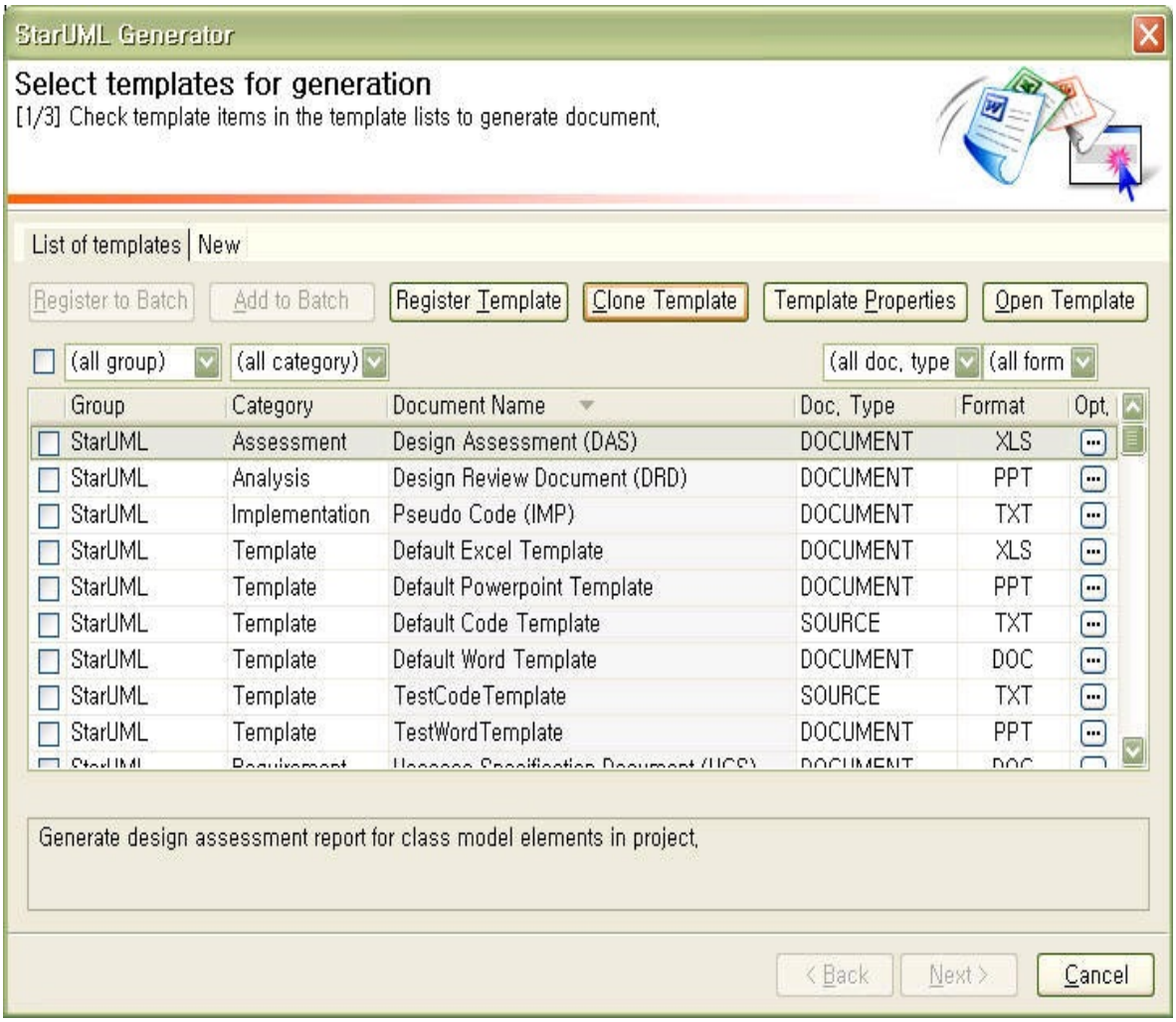
Variable	Description
$PATH$	The path that the template files are placed. (e.g.) $PATH$\BusinessActorReport.doc
$GROUP$	Group name of the template.
$CATEGORY$	Category name of the template.
$NAME$	Name of the template.
$TARGET$	Output path that the user selected.

Cloning a Template

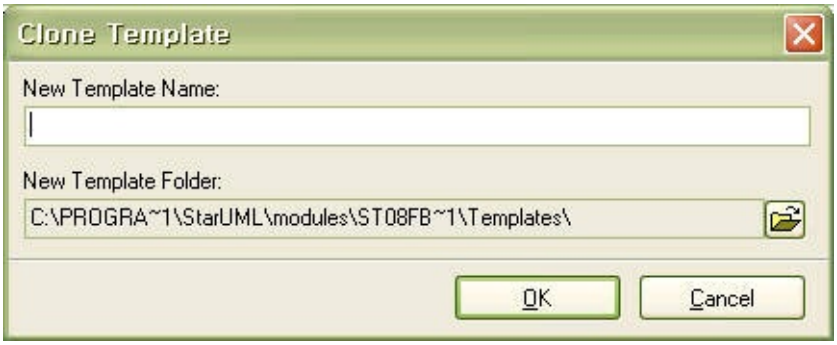
You can start to define a template by cloning an existing template without definir from the scratch.

In the **[Select templates for generation]** Dialog, select a template to clone an

1. click **[Clone Template]** button, or click mouse right button on the template to clone and then click **[Clone Template]** menu on the popup menu.



2. Specify the name of cloned template and click **[OK]** button.

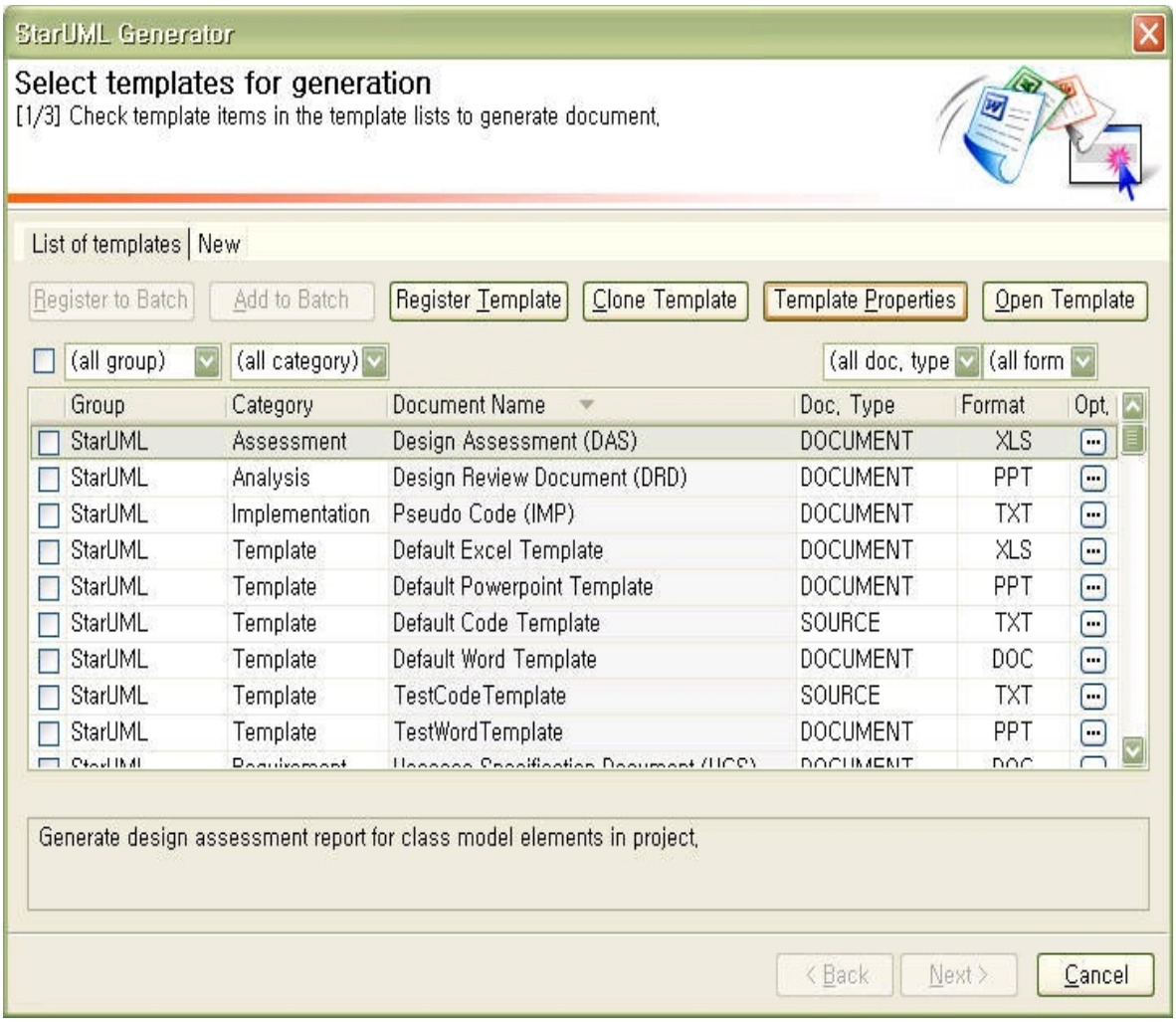


3. You can find the cloned template in the **[List of templates]**. You can edit more information of the cloned template (click **[Template Properties]** button).

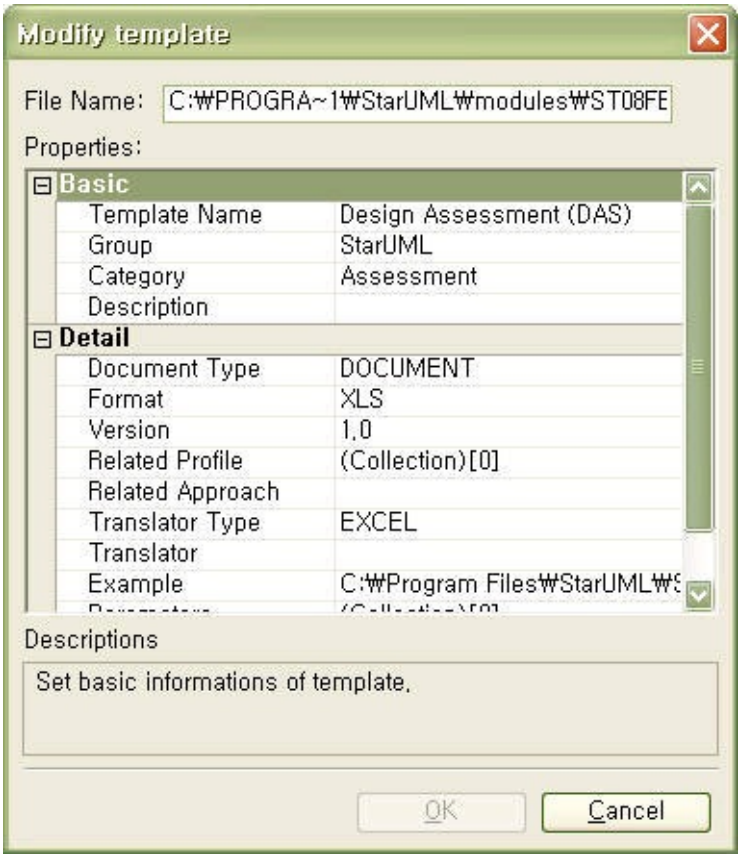
Template Properties

You can edit properties of a registered template.

1. In the **[Select templates for generation]** Dialog, select a template want to edit properties and click **[Template properties]** button, or click mouse right button the template to edit properties and then click **[Show Template Properties]** menu on the popup menu.



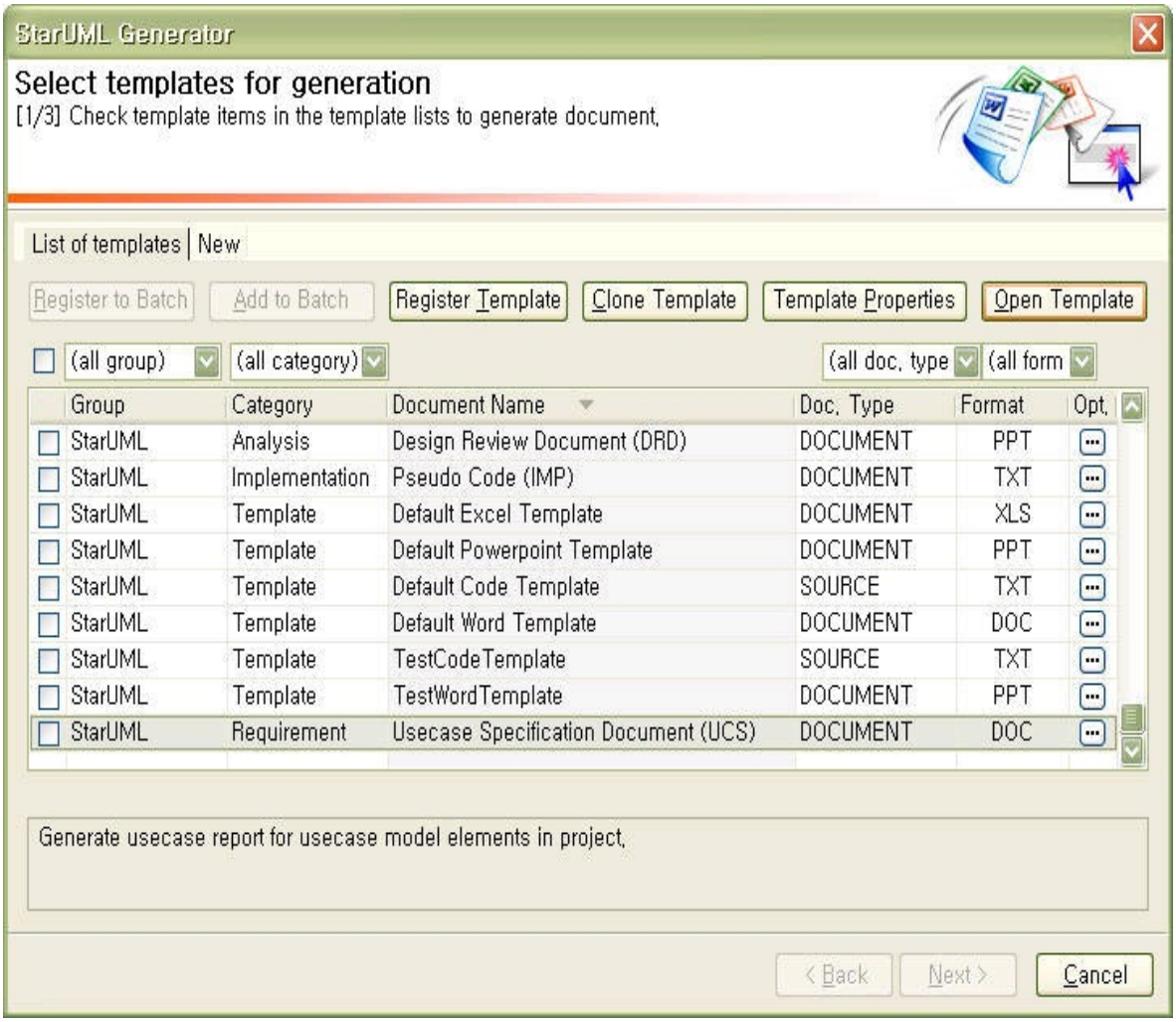
2. **Edit properties in the[Modify Template]** Dialog and click **[OK]** button. (Ple refer to **Registering Template > Basic/Detail Parameters** for detailed information of each property)



Opening a Template

You can open and edit a registered template.

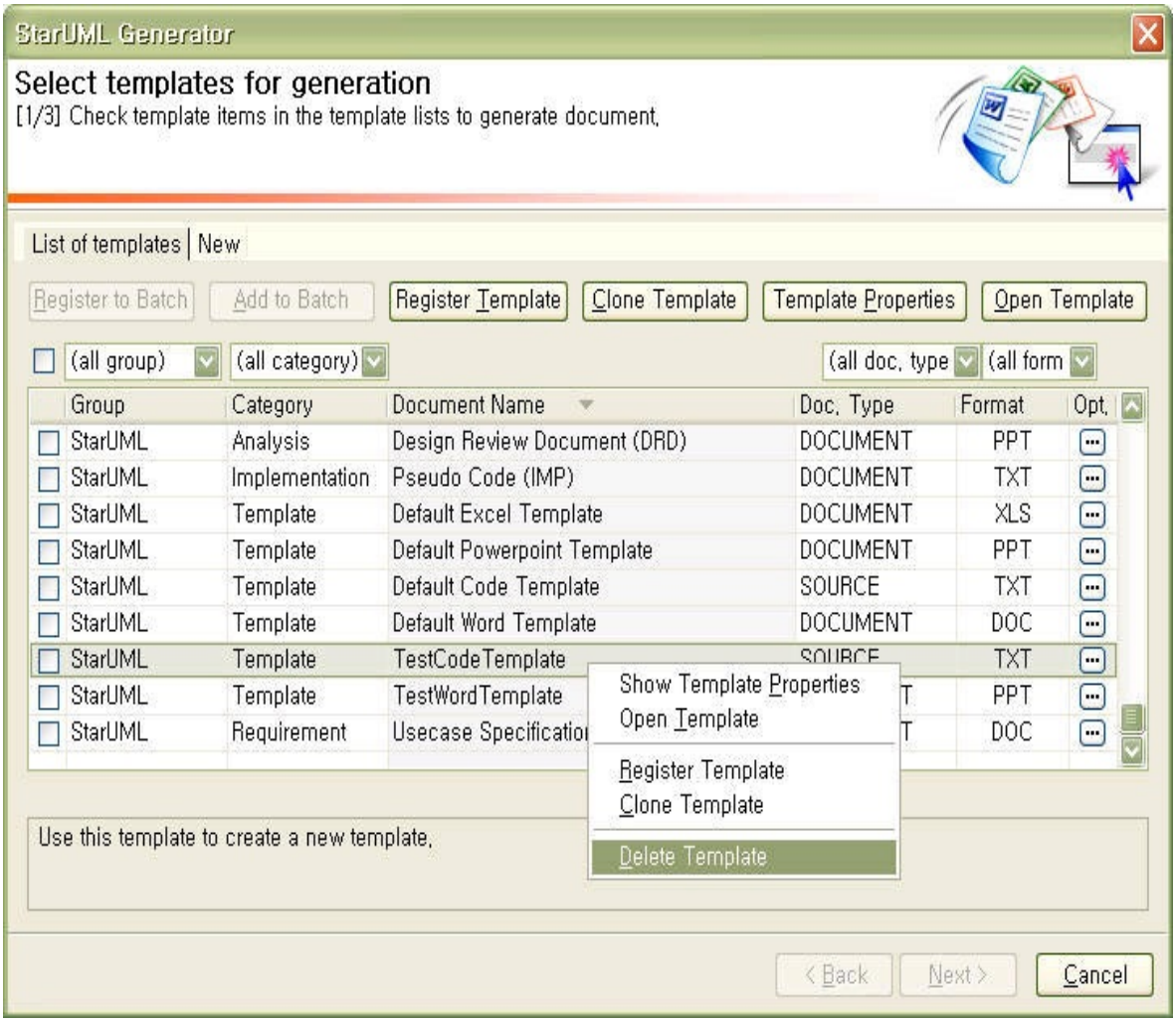
1. In the **[Select templates for generation]** Dialog, select a template to open and click **[Open Template]** button, or click mouse right button on the template to open and then click **[Open Template]** menu on the popup menu.



2. And then, the default application associated with each file extension (.cot, .doc .xls, .ppt) will be executed and you can edit in the application. (Please refer to **StarUML 5.0 Developer Guides > Chapter 11. Writing Template** for how t write template)

Deleing a Template

- In the **[Select templates for generation]** Dialog, select a template to delete an click mouse right button and click **[Delete Template]** menu in the popup menu



- Deleting a template causes deletion of the template folder and all files in the folder, so you must take care about deleting template.

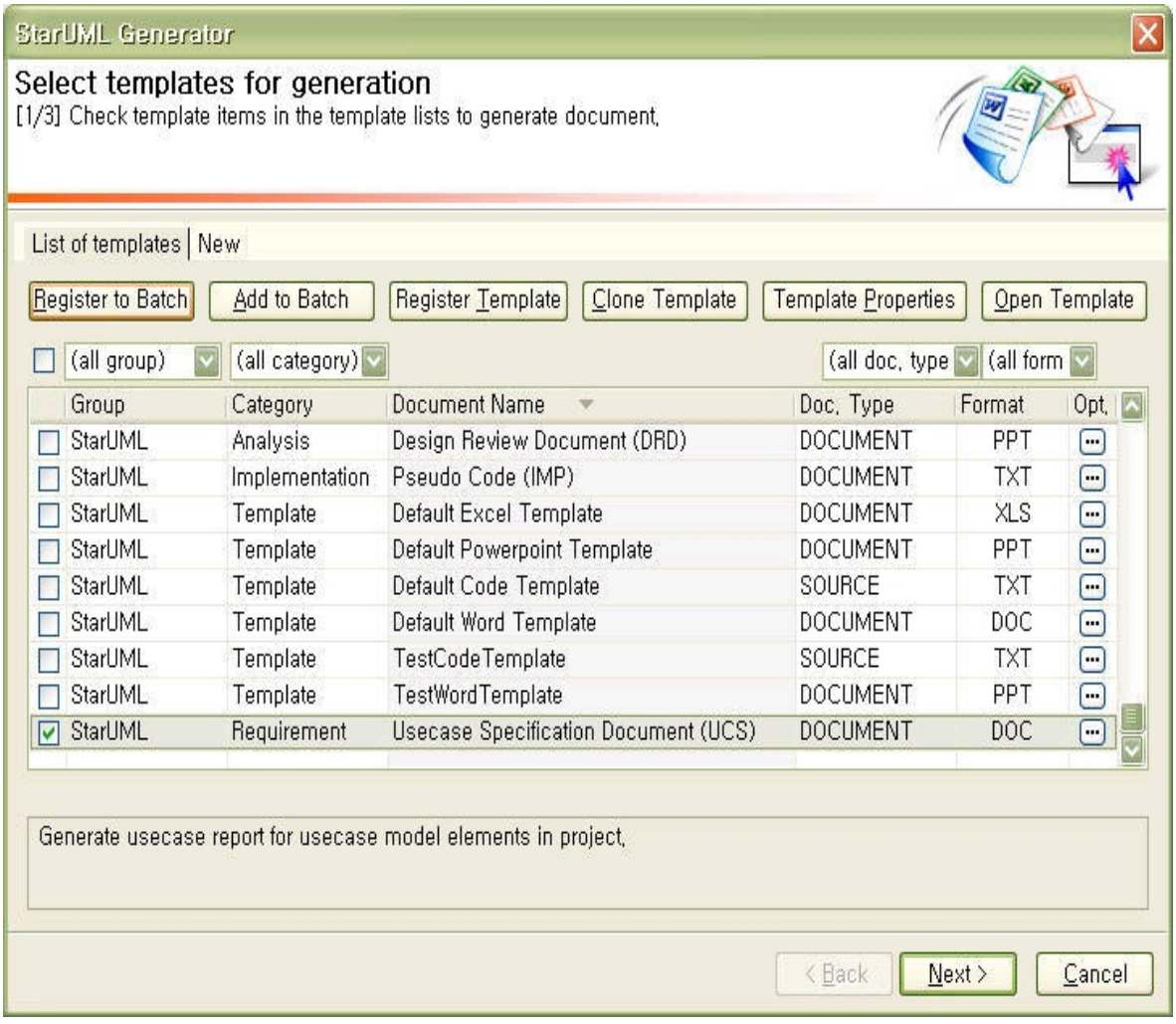
Using Batches

In the [List of templates] tab, registered templates are listed. If you want to generate a set of artifacts at once, you can make a batch that used to generate a set of artifacts and can run the batch without selecting a set of templates.

Creating a new batch

Create a new batch containing selected templates.

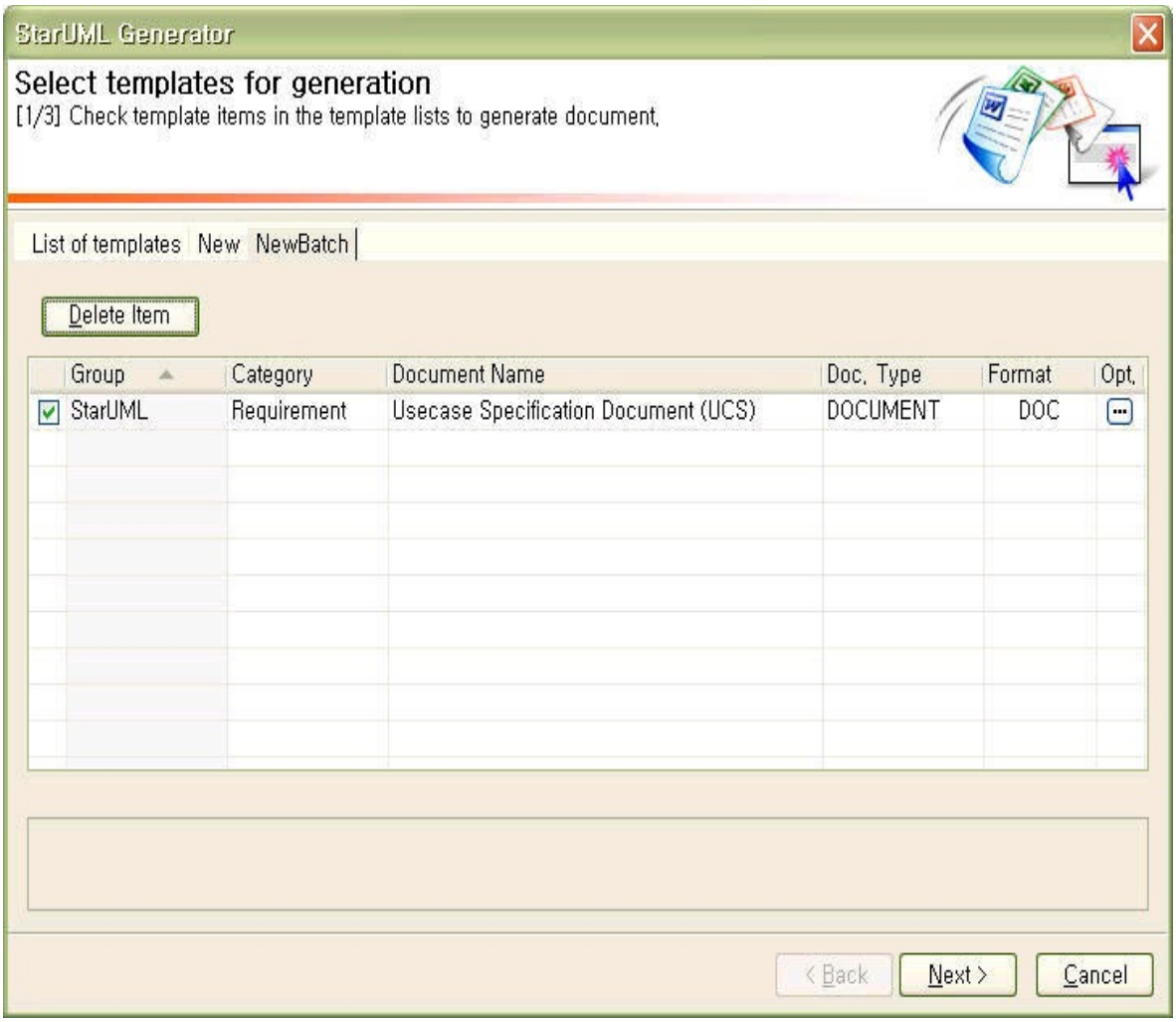
1. In the **[Select templates for generation]** Page, check templates, to make as a batch, in **[List of templates]** tab, and click **[Register to Batch]** button.



2. In the **[Register Batch]** Dialog, specify **[Batch Name]**, **[Description]** and

click **[OK]** button.

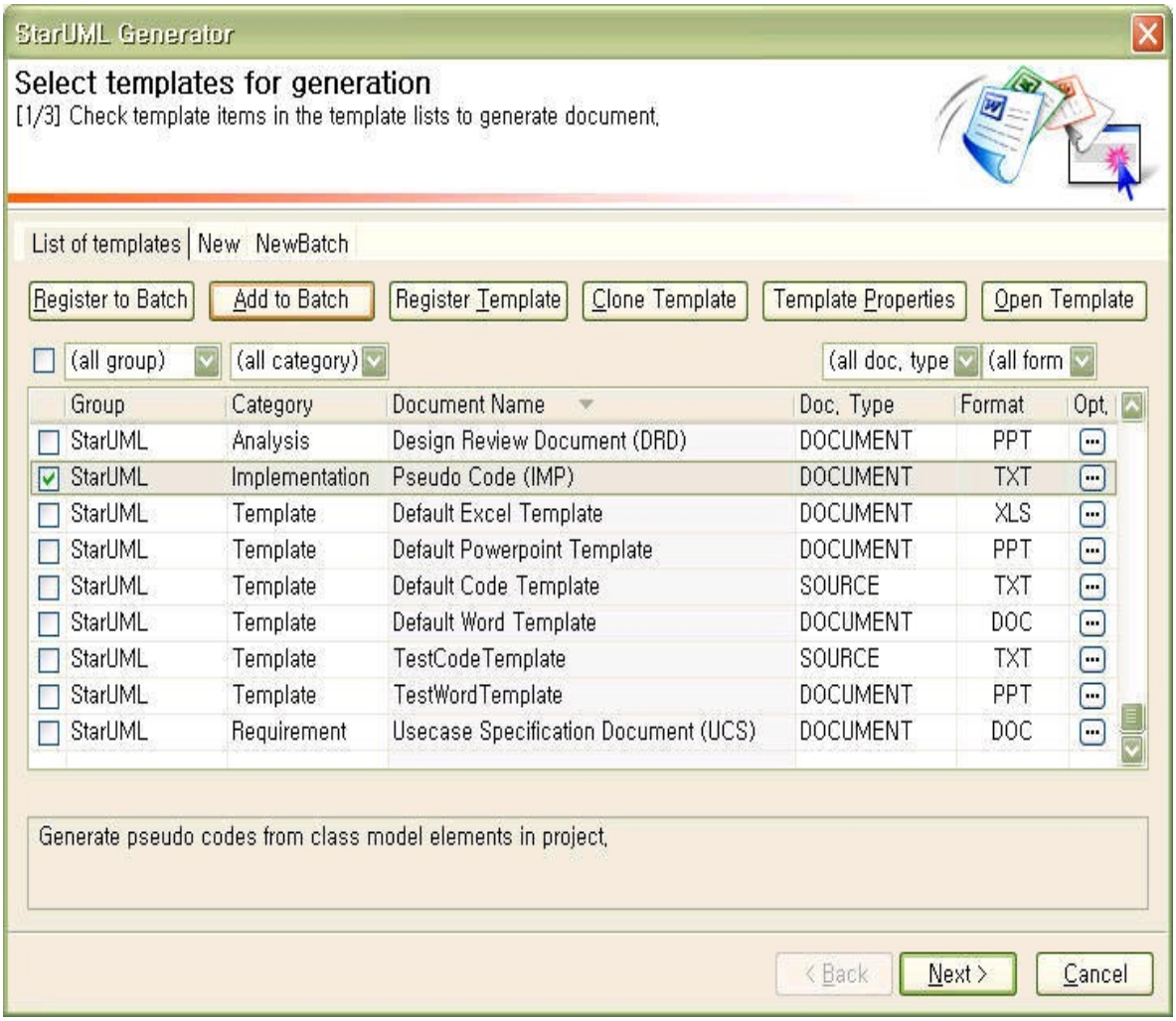
3. You can find a new batch as a tab and selected templates in template list of the batch.



Add templates to existing batch

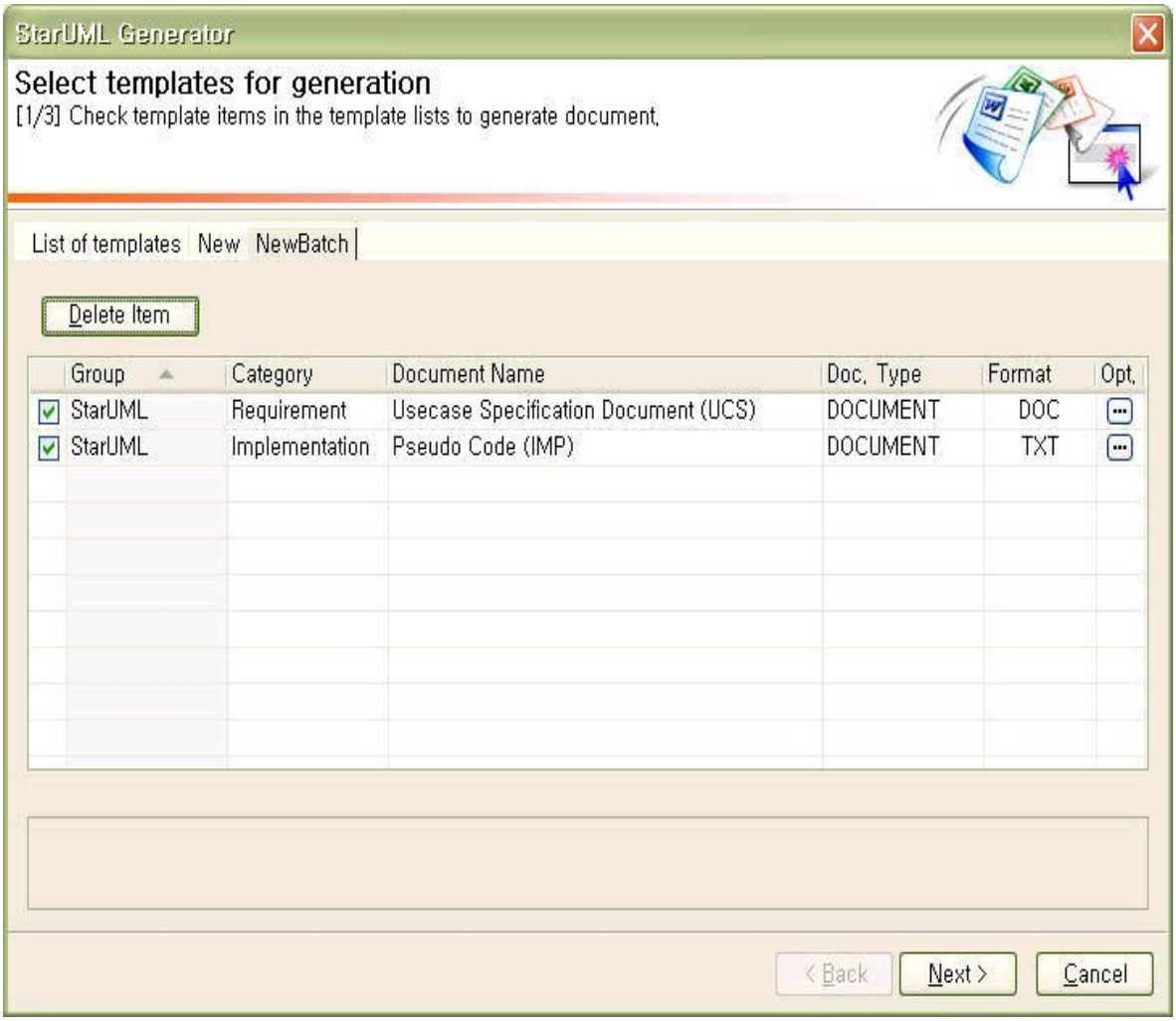
You can add templates to an existing batch.

1. In the **[Select templates for generation]** Page, check templates, to add to existing batch, in the **[List of templates]** tab and click **[Add to Batch]** button.



2. In the **[Select Batch]** Dialog, select a batch and click **[OK]** button.

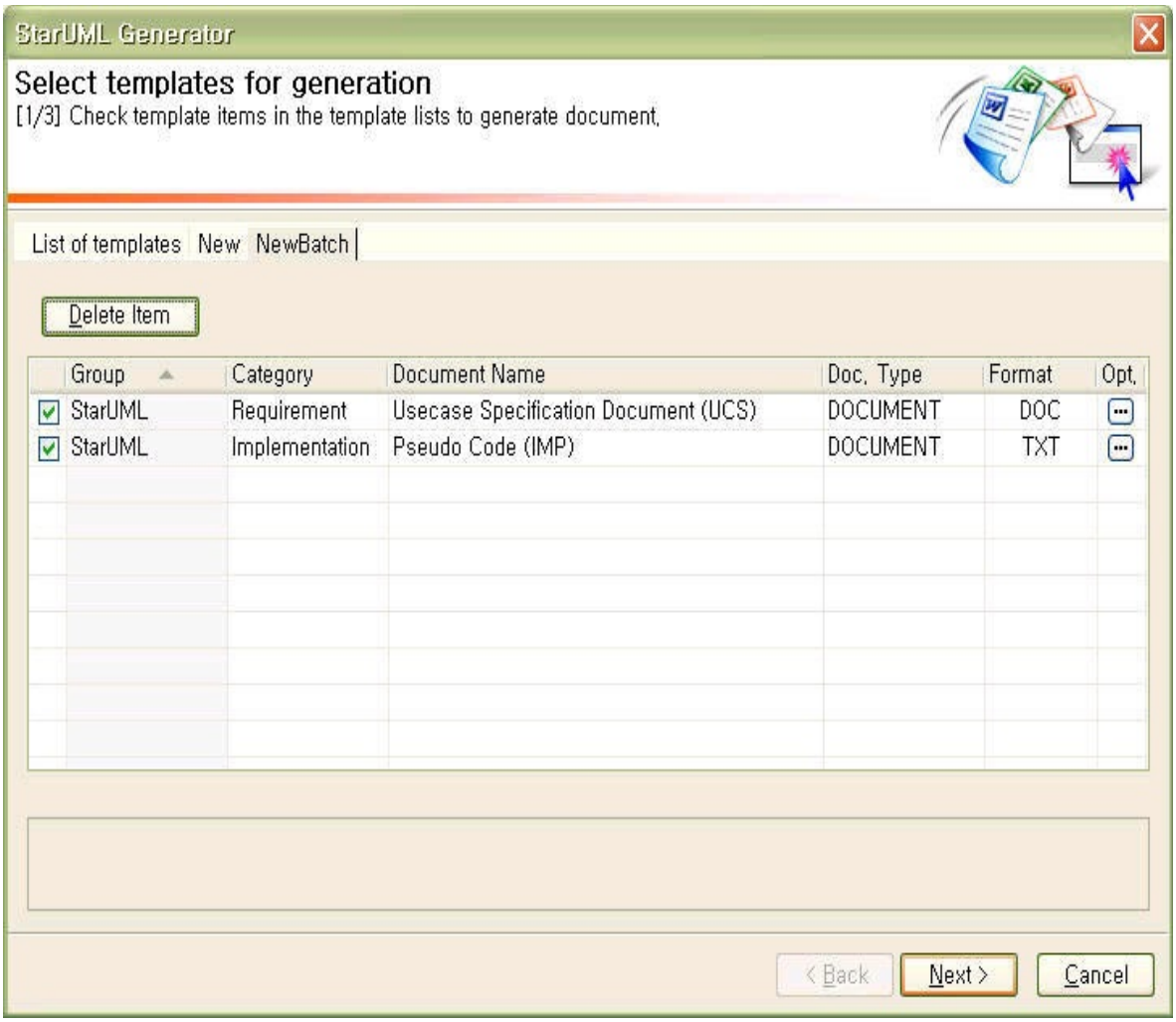
3. You can find templates added to existing batch.



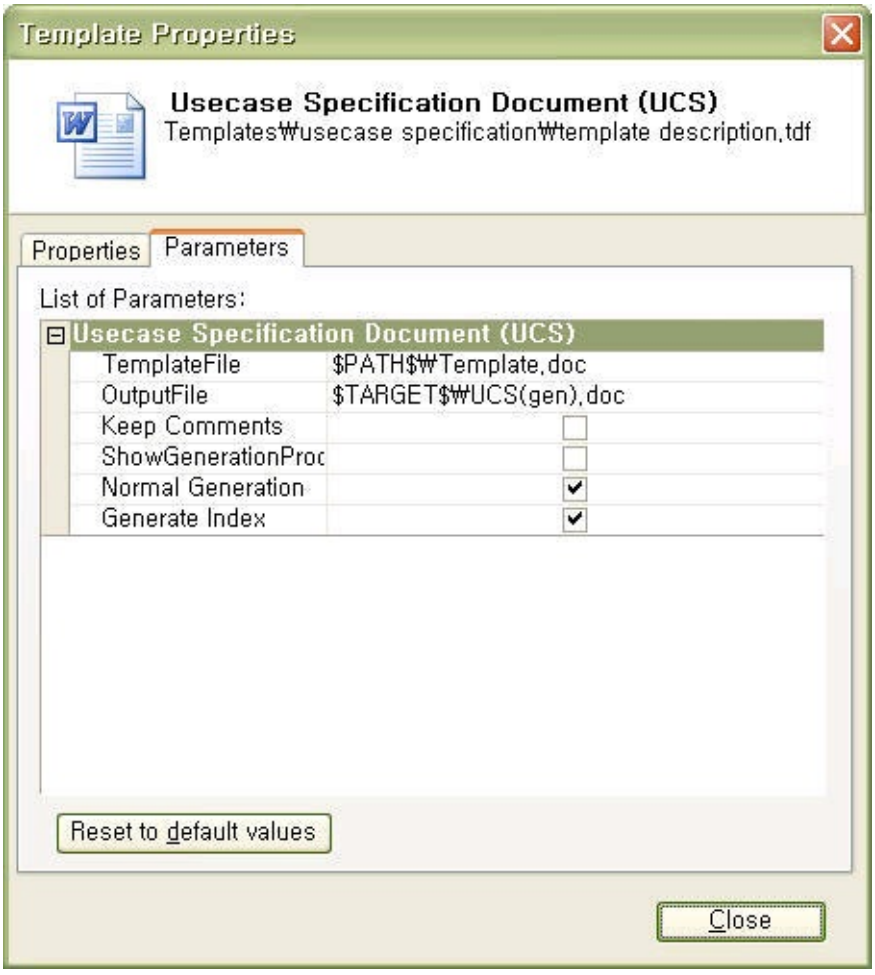
Executing a batch

You can generate many artifacts at once by using batch.

1. In the **[Select templates for generation]** Page, select batch tab to execute.
2. Check templates to generate and click **[Next]** button. (As default, all templates are checked in the batch.)



3. You can generate artifacts in the batch using different property values. If you want to do so, click ⋯ button of each template and set the value of the each property. The change of the properties are applied only once. (Please refer to Registering Template > property for more information about properties of template)



4. When **[Select target path]** page is appeared on the screen, selects a folder to save generate a document to generate, and click **[Next]**. If you want to add a folder under the present selected folder, click **[New Folder]** button on left bottom and write a name for adding folder on name configuration dialog

5. When **[Generating…]** page is appeared on the screen, clicks **[Generate]** button. You can check the statue of each template generating through statue bar as generating documents from the template. And the log of generating process is recorded to **[Logs:]** window. If you want to cancel the present generating document, click **[Cancel]** button. And clicks OK button on Confirm cancel dialog.

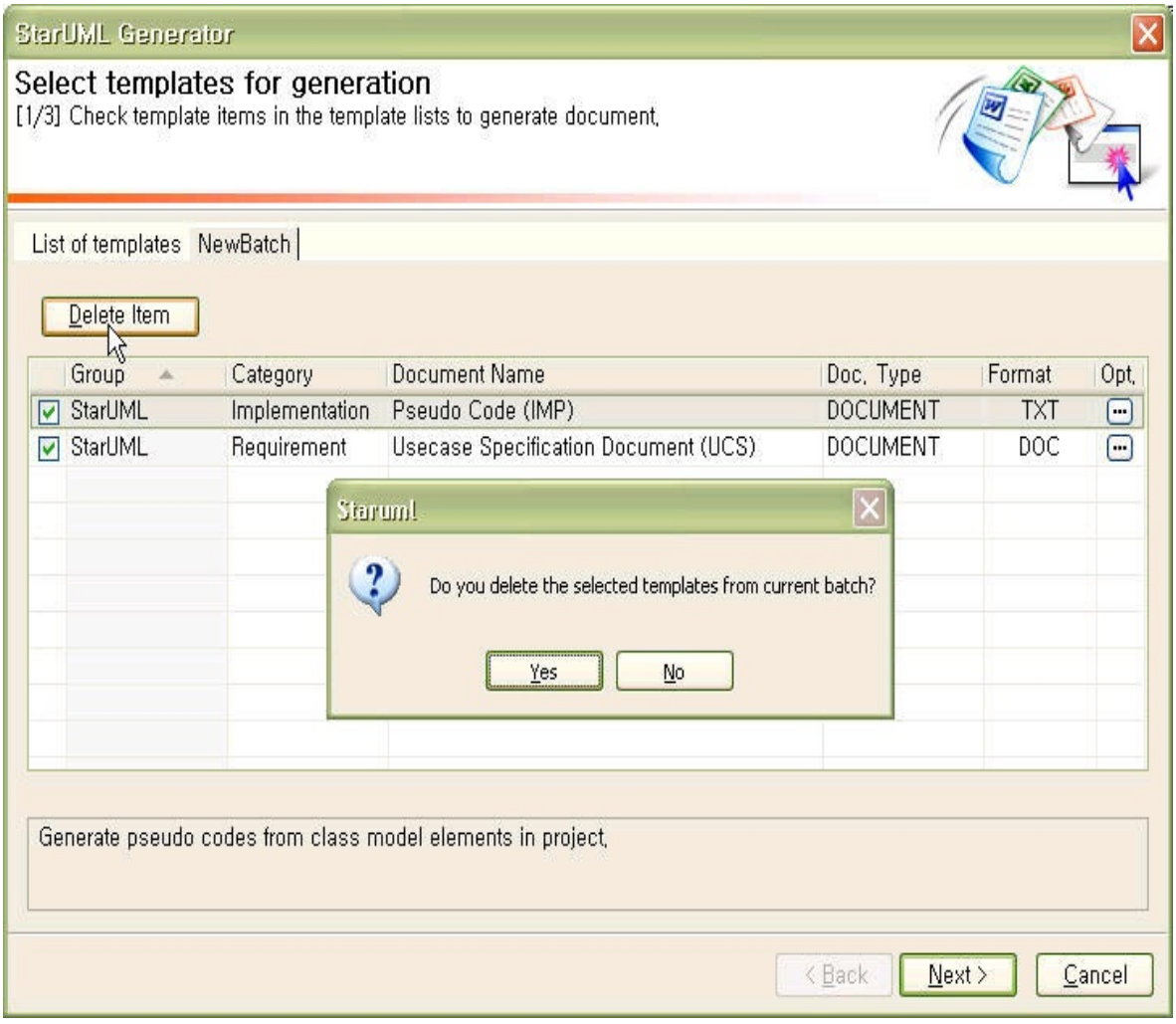
6. Log(Document Creation is done) on log window is recorded, and is activated **[Finish]** button after completing document generation. If you want to finish document generation, close the document generation process as clicking **[Finish]** button. Or you can check the generated document as double-clicking

document list in **[Generation List]**.

Deleting templates in a batch

You can delete templates in the batch. (The deletion in the batch, the template is not remove and only deleted from the batch.)

1. In the **[Select templates for generation]** Page, go to the batch tab and check templates to remove, and click **[Delete Item]** button.

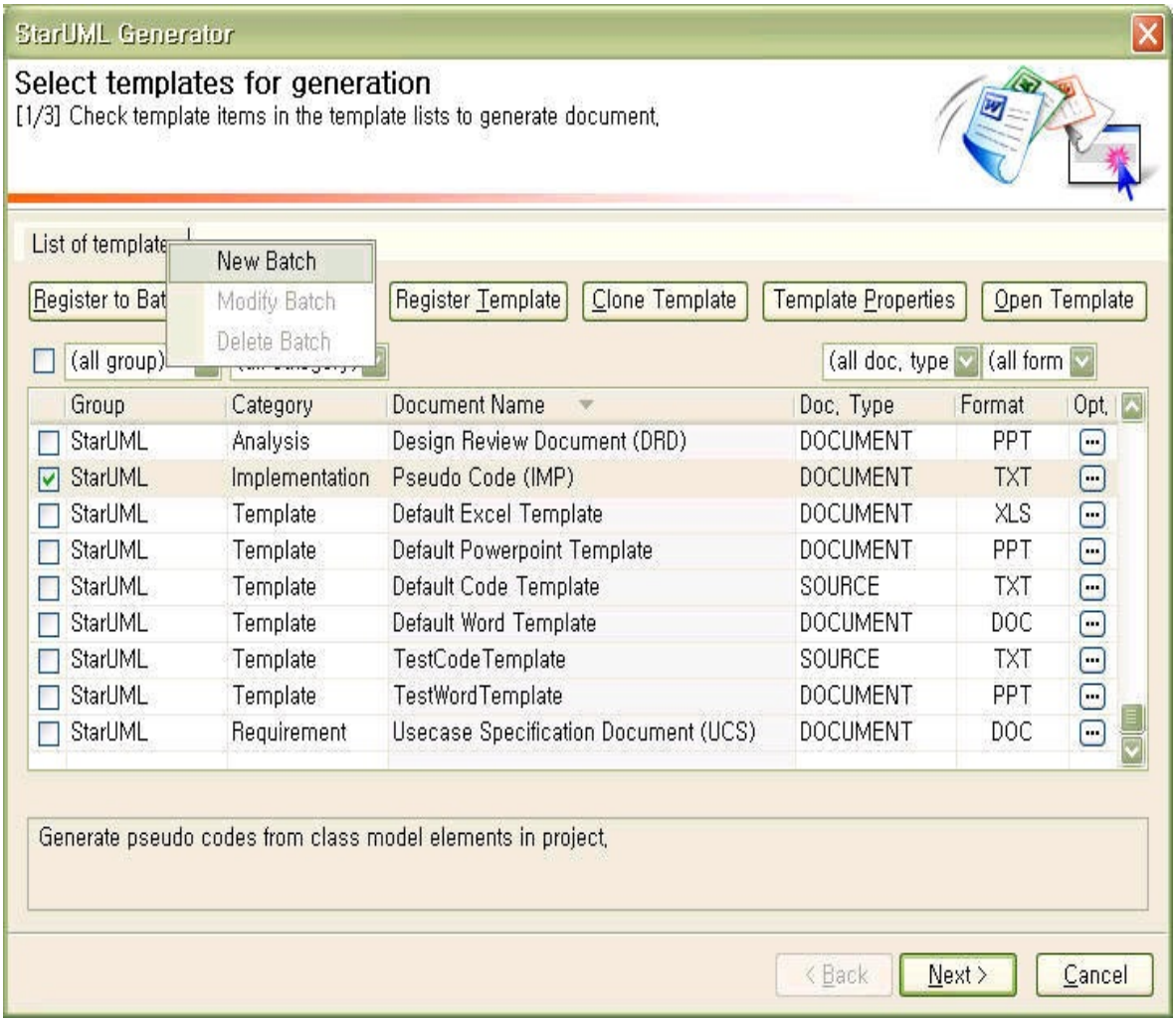


2. You can make a certain the deletion of the checked templates.

Creating an empty batch

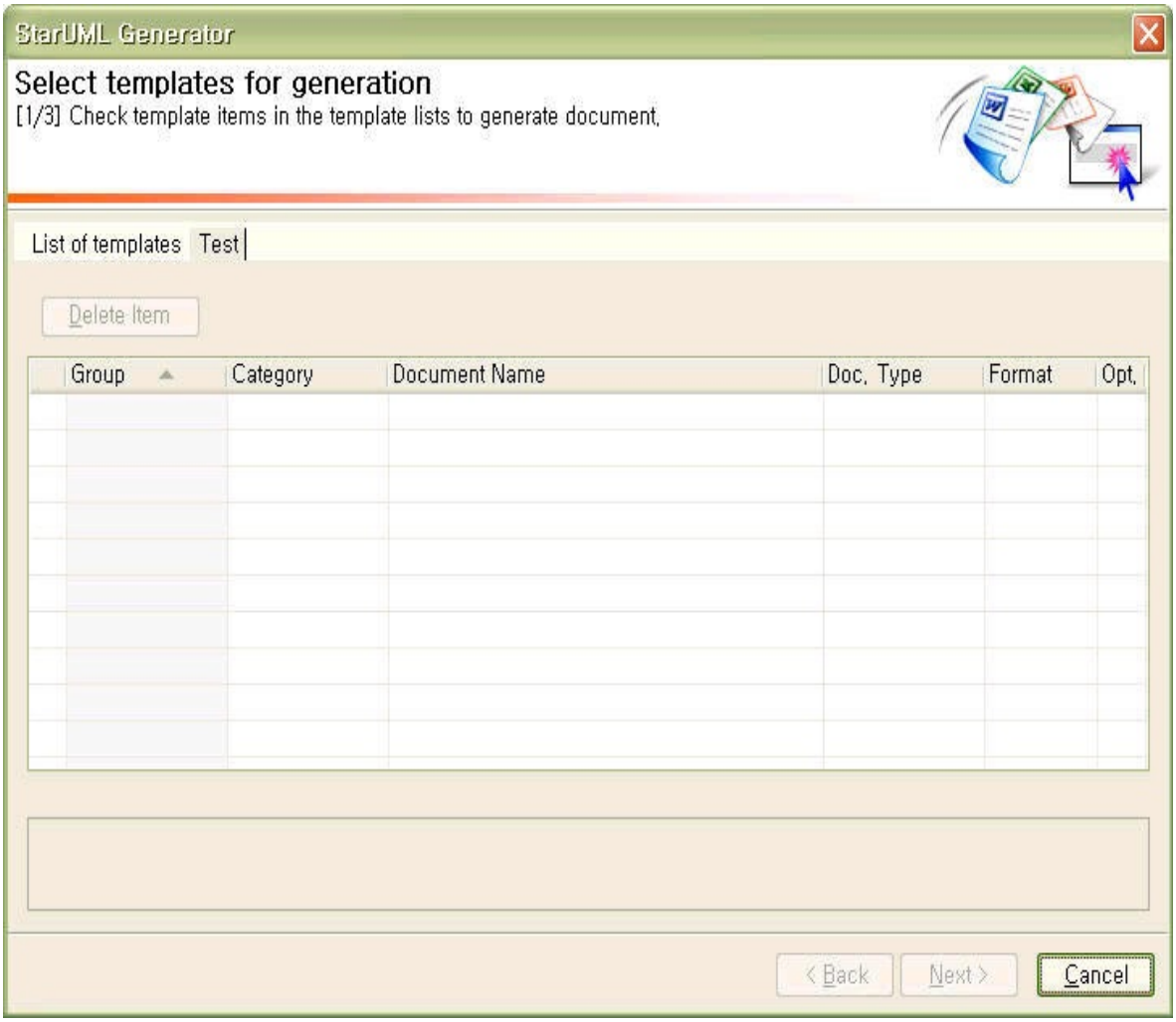
You can create a batch that containing no template.

1. In the **[Select templates for generation]** Page, click mouse right button on the any tab, and click **[New Batch]** menu.



2. In the **[Register Batch]** Dialog, specify **[Batch Name]**, **[Description]** and click **[OK]** button.

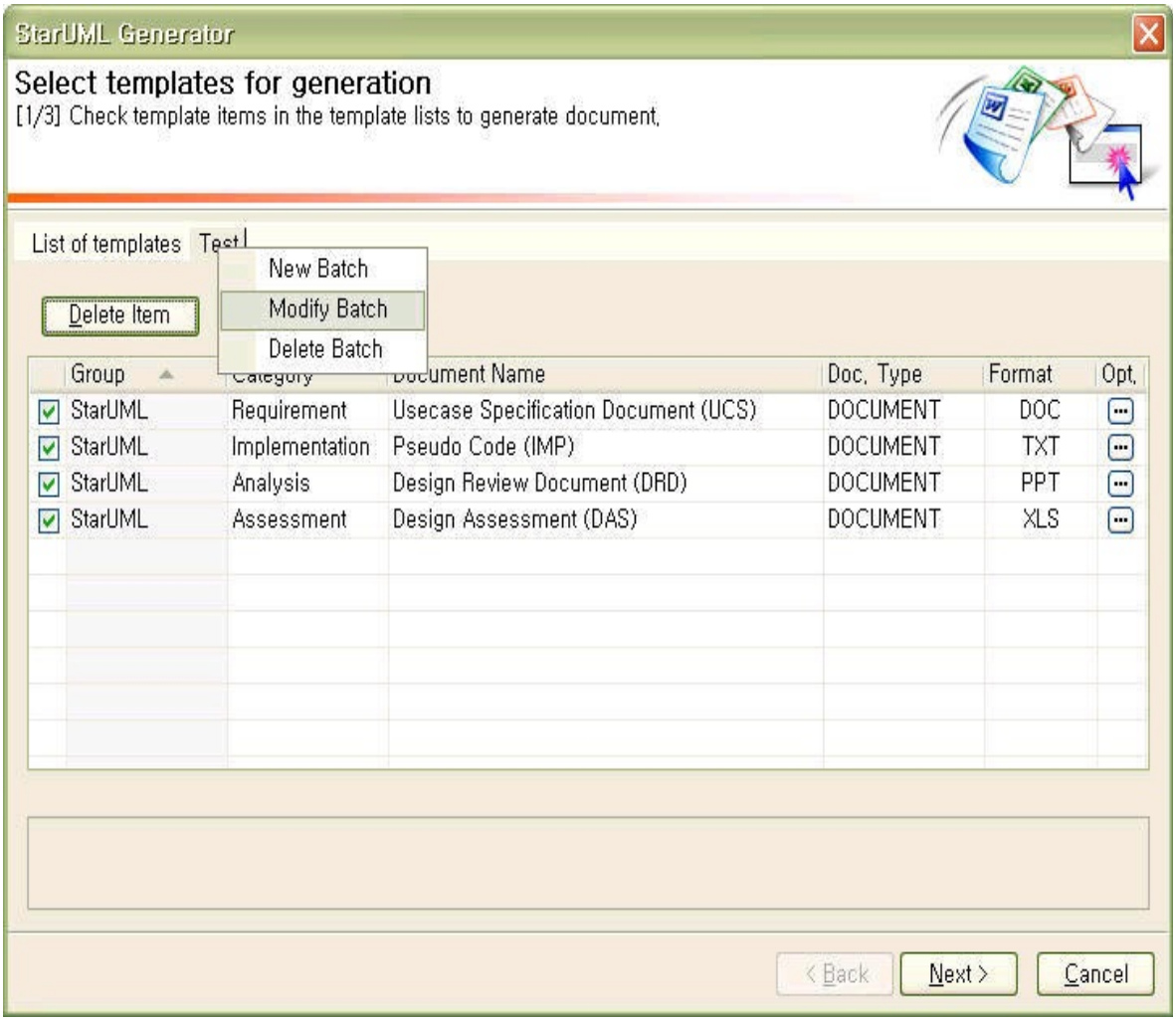
3. In the **[Select templates for generation]** Page, you can find an empty batch tab.



Modifying a batch

You can modify information about a batch.

1. In the **[Select templates for generation]** Page, select batch tab to modify, and click mouse right button and click **[Modify Batch]** menu.

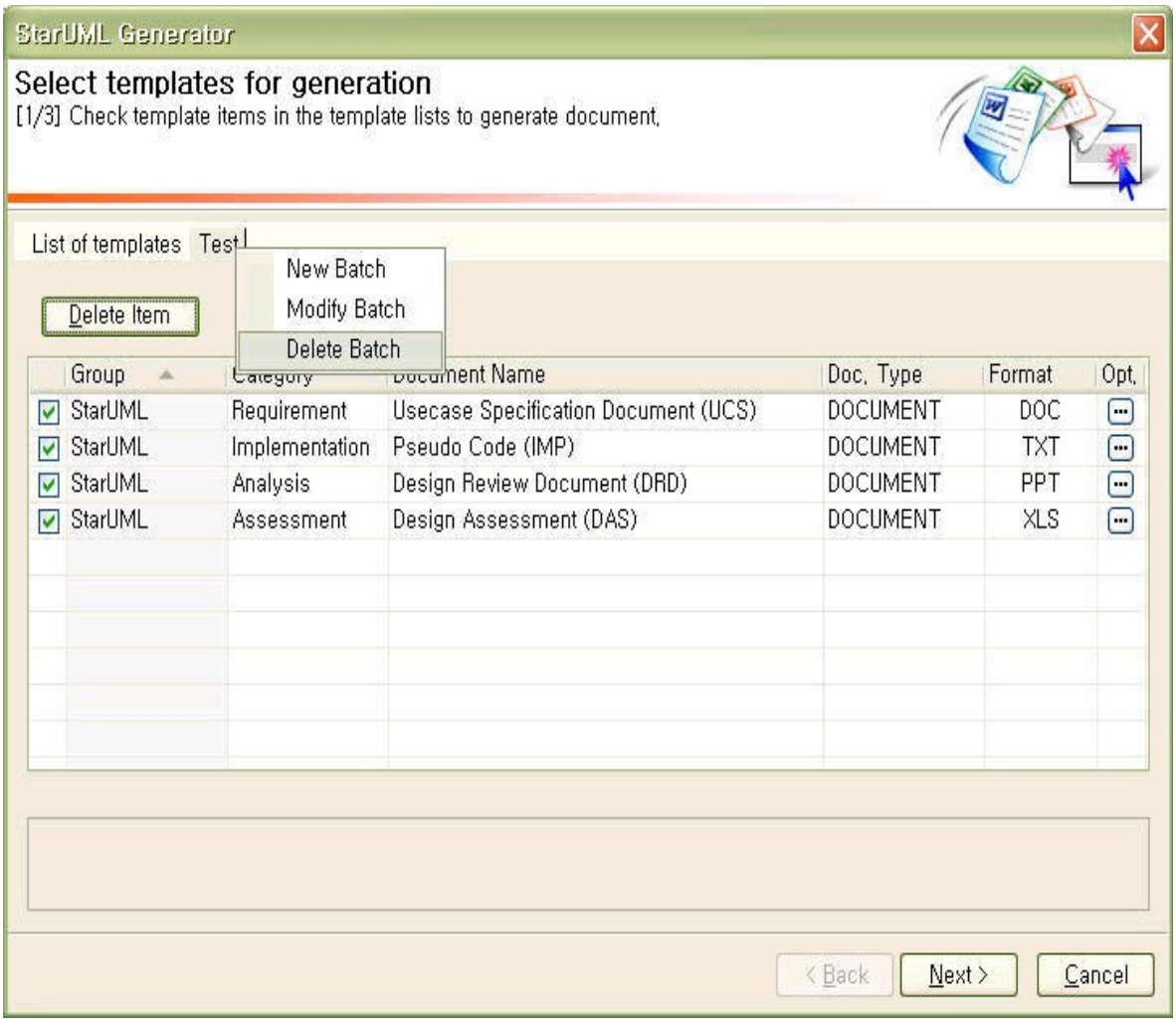


2. In the **[Register Batch]** Dialog, modify **[Batch Name]**, **[Description]** and click **[OK]** button.

Deleting a batch

You can delete a batch.

1. In the **[Select templates for generation]** Page, select a batch tab to delete and click mouse right button and click **[Delete Batch]** menu.



2. You can make a certain that the deletion of the batch tab (Deleting batch not cause deletion of the templates contained by the template)

Installing and Uninstalling Templates

Organization of templates

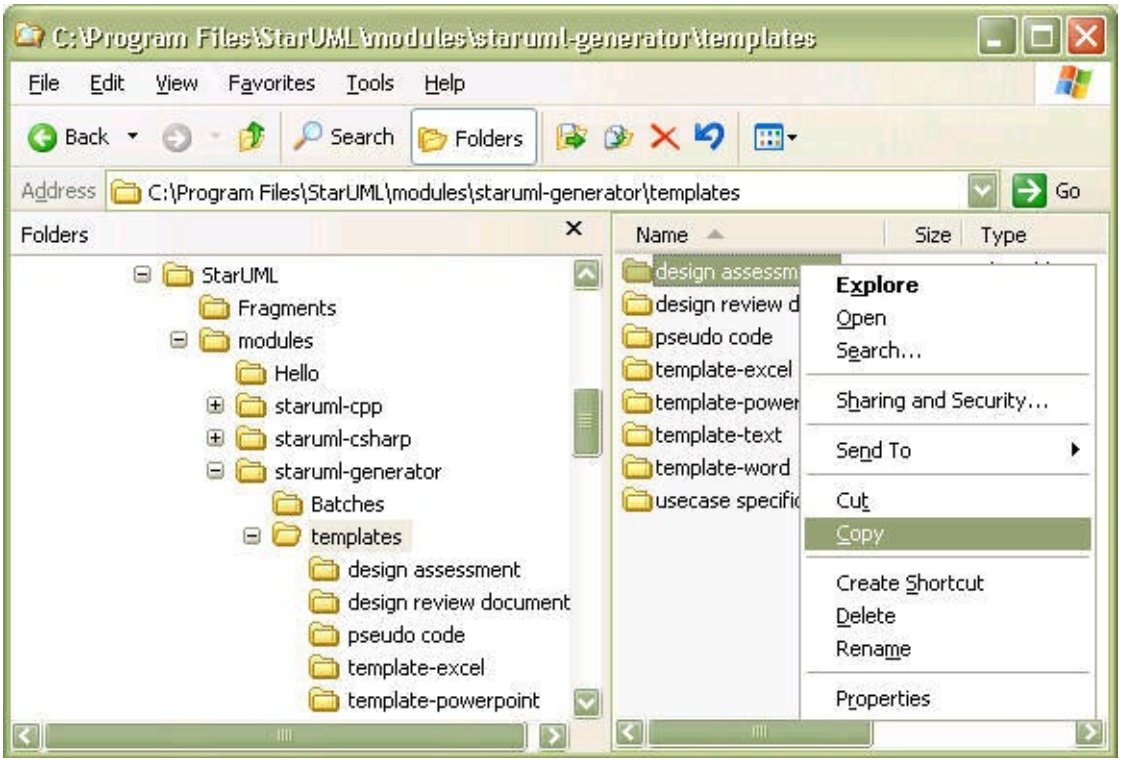
Templates are installed under the folder "<STARUML_INSTALL_PATH>\modules\staruml-generator\templates" and batches are under the folder "<STARUML_INSTALL_PATH>\modules\staruml-generator\batches". In general, one template matches one folder and the folder contains all files associated to the template. A template includes at least two files. The first is template description file (.tdf) and the second is the template document (.cot, .doc, .xls, .ppt, ...). Batch includes one file that is batch file (.btf).

Directory structure of staruml-generator module is as follow.

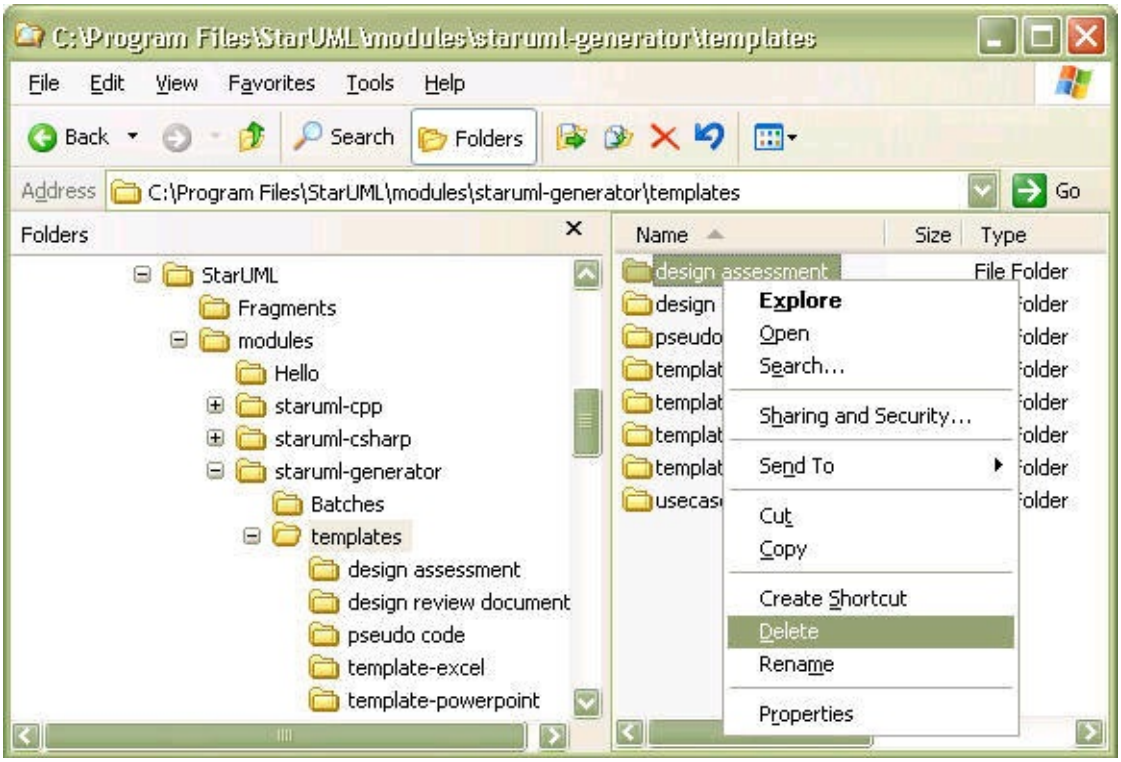
```
staruml-generator\
 templates\
    template1\
       template1.tdf
       template1.doc
    template2\
       ...
 batches\
    batch1.btf
    ...
```

Installation and uninstallation of a template

Installing a template is very simple. Simply copy the template folder to the "<STARUML_INSTALL_PATH>\modules\staruml-generator\templates".

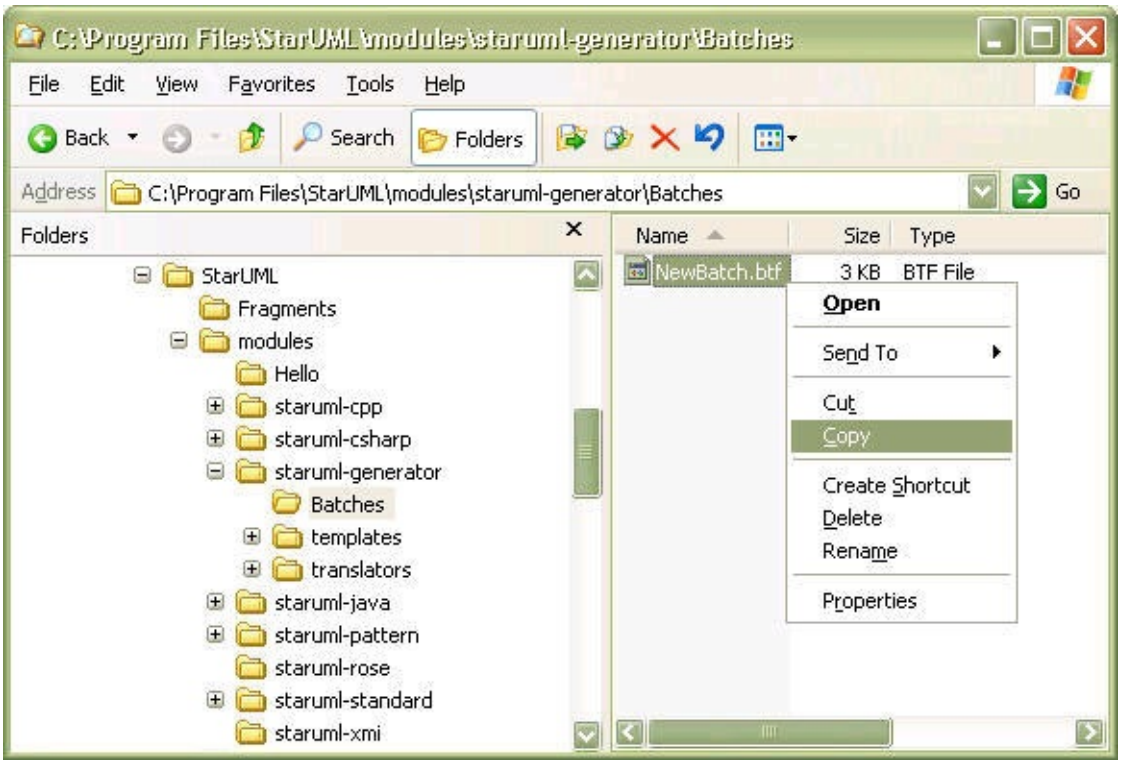


Uninstalling a template is also very simple. Delete the template folder under the "<STARUML_INSTALL_PATH>\modules\staruml-generator\templates". It's all.

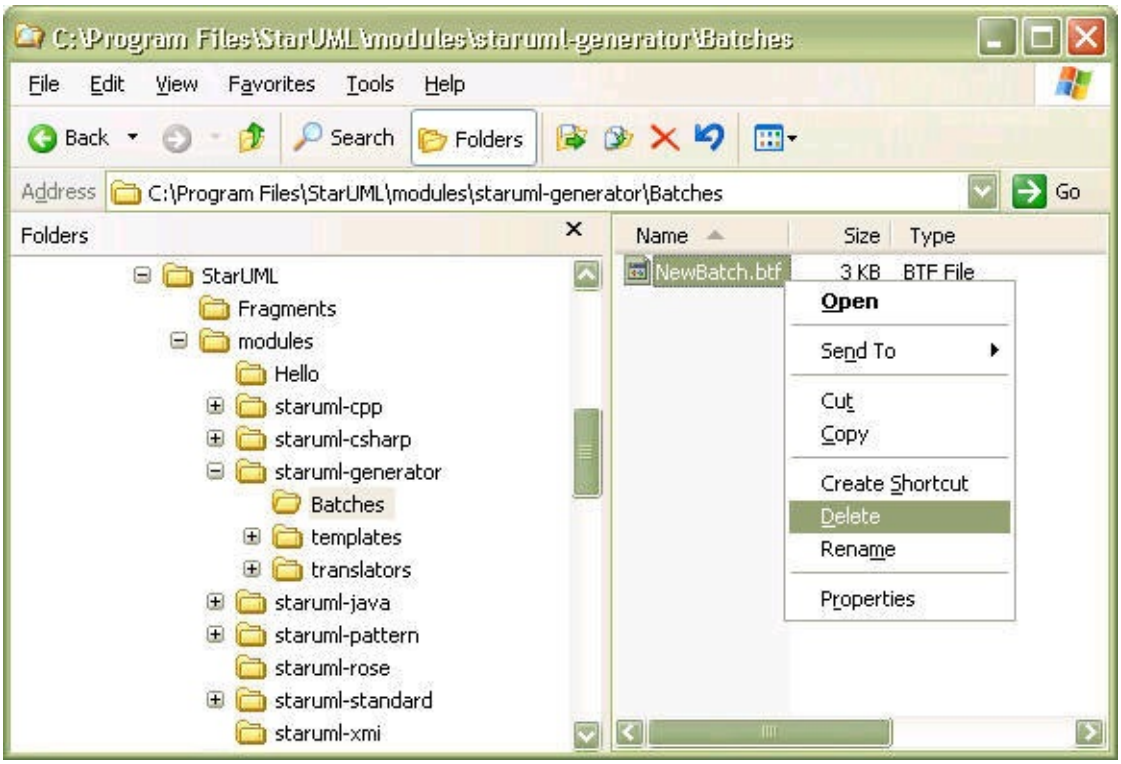


Installation and uninstallation of a batch

Installing a batch is very simple. Simply copy the batch file (.btf) to the "<STARUML_INSTALL_PATH>\modules\staruml-generator\batches".



To uninstall a batch, delete the batch file (.btf) in the "<STARUML_INSTALL_PATH>\modules\staruml-generator\batches".


C:\Program Files\StarUML\modules\staruml-generator\Batches
File Edit View Favorites Tools Help
Back Search Folders
Address C:\Program Files\StarUML\modules\staruml-generator\Batches
Go
Folders
StarUML
Fragments
modules
Hello
staruml-cpp
staruml-csharp
staruml-generator
Batches
templates
translators
staruml-java
staruml-pattern
staruml-rose
staruml-standard
staruml-xmi
Name
Size
Type
NewBatch.btf
3 KB
BTF File
Open
Send To
Cut
Copy
Create Shortcut
Delete
Rename
Properties

Verifying Model

This chapter describes the software model inspection: inspection method and regulations used in inspection.

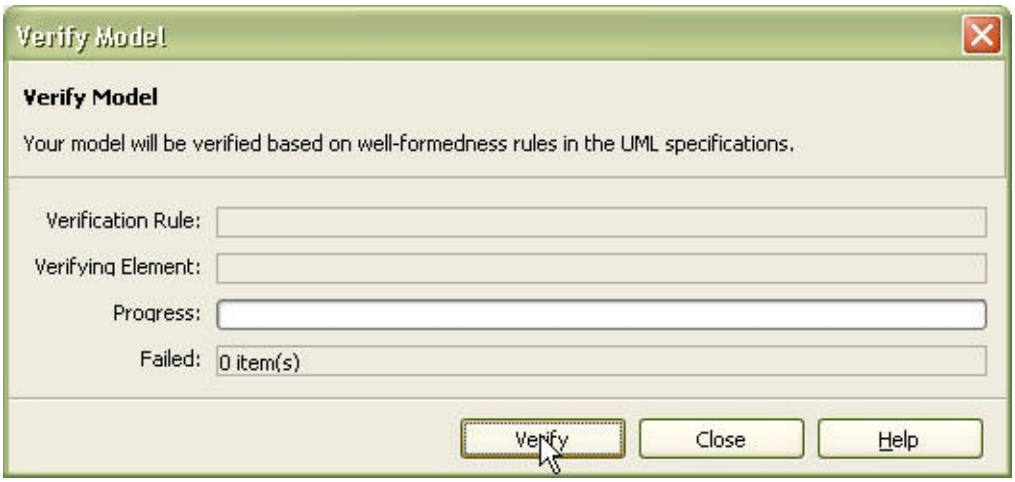
- Verifying Model
- Well-formedness Rules

Verifying Model

Users can make many mistakes during software modeling. Such mistakes can be very costly if left uncorrected until the final coding stage. In order to prevent this problem, StarUML™ allows verification of software models by applying the basic UML regulations.

Procedure for Verifying Model:

1. Select the **[Model] -> [Verify Model…]** menu.
2. At the Verify Model dialog box, click the **[Verify]** button.



3. When the verification finishes, the **[Messages]** section displays the names of the elements that did not pass the verification, and their error contents. Double-click a message to move to the element concerned.

Well-formedness Rules

Thirty-eight regulations are defined for verifying models. These definitions are mostly adaptations of the Well-formedness Rule in the UML specification.

Model Verification Regulation List

No	Regulation Contents	Elements Applied
1	AssociationEnds within an Association must have unique names.	Association
2	Two or more Aggregations or Composite AssociationEnds cannot exist within an Association.	Association
3	Parameters must have unique names.	BehavioralFeature
4	Attributes of the same name cannot exist within a Classifier.	Classifier
5	AssociationEnds on the other side must have unique names.	Classifier
6	An Attribute cannot have the same name as the Association on the other side, or as elements included in Classifier.	Classifier
7	AssociationEnd on the other side cannot have the same name as elements included in Classifier or its Attribute name.	Classifier
8	Root element cannot have elements that are more generalized than itself.	GeneralizableElement
9	Leaf element cannot have elements that are more specialized than itself.	GeneralizableElement
10	Looped inheritance structure is not allowed.	GeneralizableElement
11	All features of interfaces must be public.	Interface
12	ComponentInstance must accurately assign a component as its origin.	ComponentInstance
13	NodeInstance must accurately assign a node as its origin.	NodeInstance
14	AssociationEndRole must be connected with ClassifierRole.	AssociationEndRole

15	ClassifierRole cannot have its own features.	ClassifierRole
16	ClassifierRole cannot become the ClassifierRole for another ClassifierRole.	ClassifierRole
17	Sender and receiver of a message must participate in the collaboration that constitutes the interaction context.	Message
18	Actor can only have associations that are connected to UseCase, Class or Subsystem.	Actor
19	CompositeState can have a maximum of one initial state only.	CompositeState
20	CompositeState can have a maximum of one deep history only.	CompositeState
21	CompositeState can have a maximum of one shallow history only.	CompositeState
22	Concurrent composite state must contain a minimum of two composite states.	CompositeState
23	Concurrent state can only have composite state as its sub state.	CompositeState
24	Final state cannot have outgoing transition.	FinalState
25	Initial state can have a maximum of one outgoing transition and cannot have incoming transition.	Pseudostate
26	History state can have a maximum of one outgoing transition.	Pseudostate
27	Junction vertex must have a minimum of one incoming transition and one outgoing transition each.	Pseudostate
28	Choice vertex must have a minimum of one incoming transition and one outgoing transition each.	Pseudostate
29	StateMachine can be integrated either with Classifier or with BehavioralFeature.	StateMachine
30	Top state must always be composite state.	StateMachine
31	No state can contain top state.	StateMachine
32	Top state cannot have outgoing transition.	StateMachine
33	SubmachineState cannot have concurrency.	SubmachineState
	Transition that points to Pseudostate cannot have	

34	Trigger.	Transition
35	ActivityGraph can express dynamic behavior of Package, Classifier or BehavioralFeature.	ActivityGraph
36	ActionState cannot have internal transition, exit action or do activity.	ActionState
37	Outgoing transition of ActionState cannot have trigger event.	ActionState
38	SubactivityState must have connection to ActivityGraph.	SubactivityState

Printing

This chapter describes the issues related to printing diagrams: page setup, various methods to print diagrams, and previewing print result.

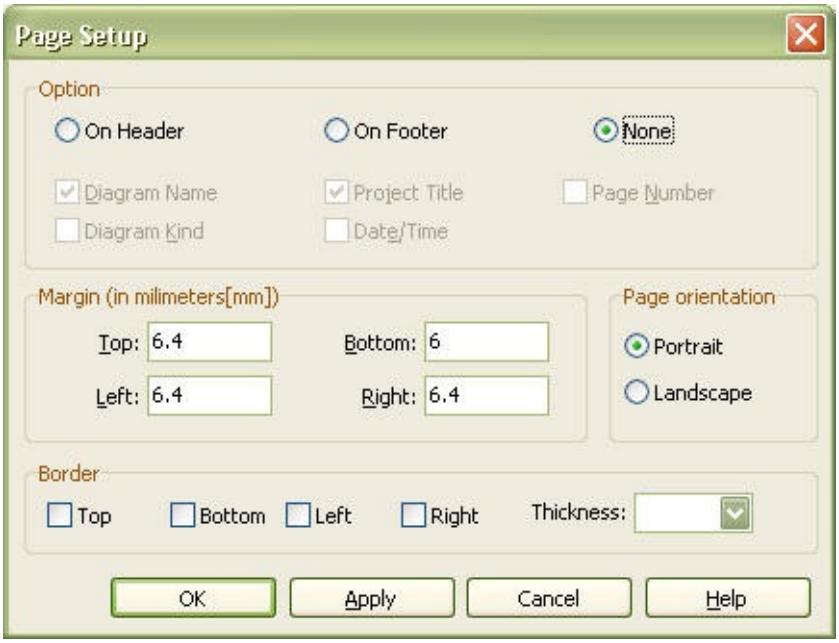
- Page Setup
- Printing Diagram
- Print Preview

Page Setup

Various properties (printing diagram information, page margins, printing outline, etc.) can be configured for the print page.

Procedure for Viewing Diagram Information:

1. Select the **[File]** -> **[Page Setup…]** menu to open the Page Setup dialog box.



2. First, choose where the diagram information will be printed. In the **[Options]** group, select **[None]** if the diagram information does not need to be printed. Select **[Header]** to print the diagram information at the top of the page, and select **[Footer]** to print it at the bottom of the page.

3. Select which information will be printed. Types of information available for printing include: **[Diagram name]**, **[Project name]**, **[Page number]**, **[Diagram type]** and **[Date/Time]**

Procedure for Setting Page Orientation:

1. Select the **[File]** -> **[Page Setup…]** menu to open the Page Setup dialog box.

2. In the **[Page orientation]** group, select **[Portrait]** to print the paper portrait, and **[Landscape]** to print it landscape.

Procedure for Specifying Page Margin:

1. Select the **[File] -> [Page Setup…]** menu to open the Page Setup dialog box.
2. In the **[Margins]** group, enter margin sizes in millimeters in the fields **[Top]**, **[Bottom]**, **[Left]**, and **[Right].**

Procedure for Printing Page Outline:

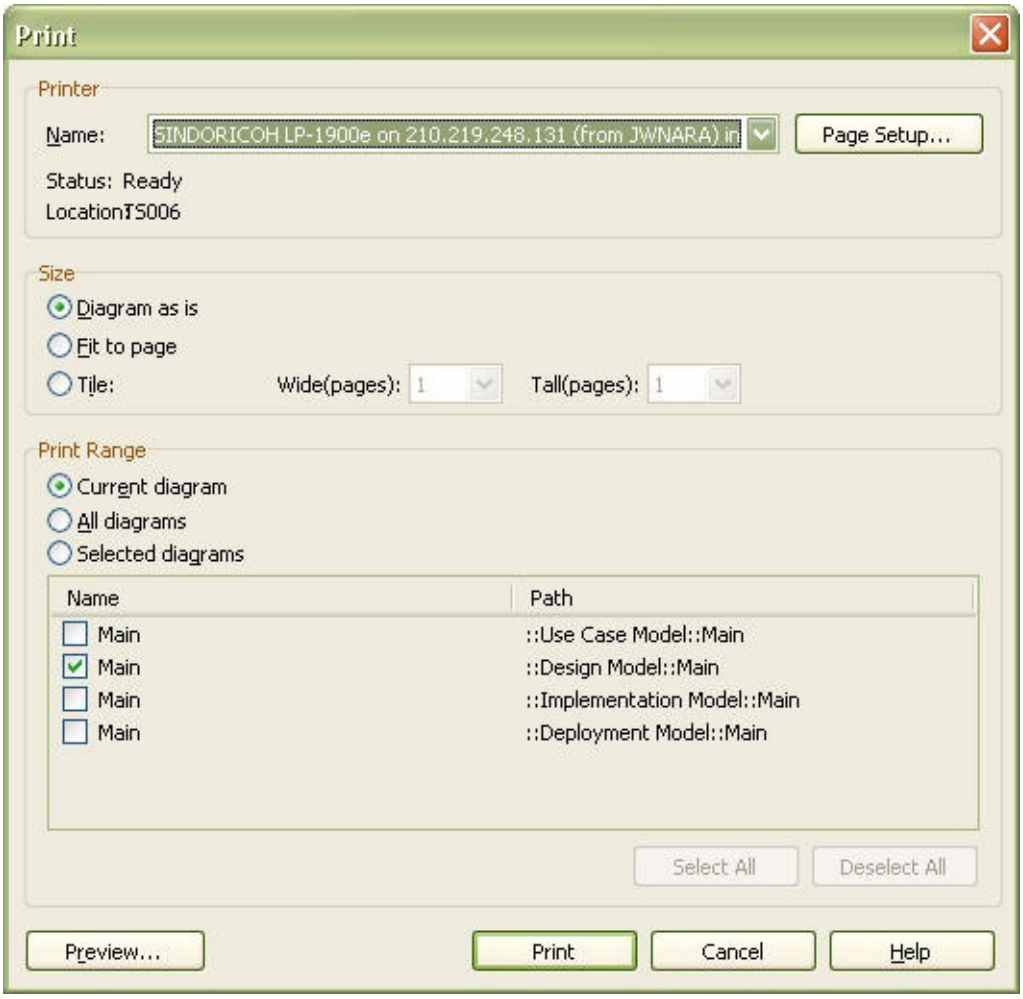
1. Select the **[File] -> [Page Setup…]** menu to open the Page Setup dialog box.
2. In the **[Border]** group, specify the sides for printing outline by checking **[Top]**, **[Bottom]**, **[Left]**, and **[Right].**
3. Enter the border thickness in the **[Thickness]** field.

Printing Diagram

Diagrams can be printed through various methods. This section describes the functions of selecting diagram to print, specifying diagram print size, printing diagram in multiple pages, etc.

Procedure for Printing the Current Diagram:

1. Select the **[File]** -> **[Print…]** menu to open the Print dialog box.



2. In the **[Printer]** group, enter the name of the printer to use in the **[Name]** field.

3. In the **[Print range]** group, select **[Current diagram]** and click the **[Print]** button.

Procedure for Printing Selected Diagrams Only:

1. Select the **[File]** -> **[Print…]** menu to open the Print dialog box.
2. In the **[Printer]** group, enter the name of the printer to use in the **[Name]** field.
3. In the **[Print range]** group, select **[Selected diagram(s)]** and check the diagrams to print in the **[Print range]** list below.
4. Click the **[Print]** button.

Procedure for Printing Diagram to Fit to Page:

1. Select the diagram(s) to print at the Print dialog box.
2. In the **[Size]** group, select **[Fit to page]** and click the **[Print]** button.

Procedure for Printing Diagram in Multiple Pages:

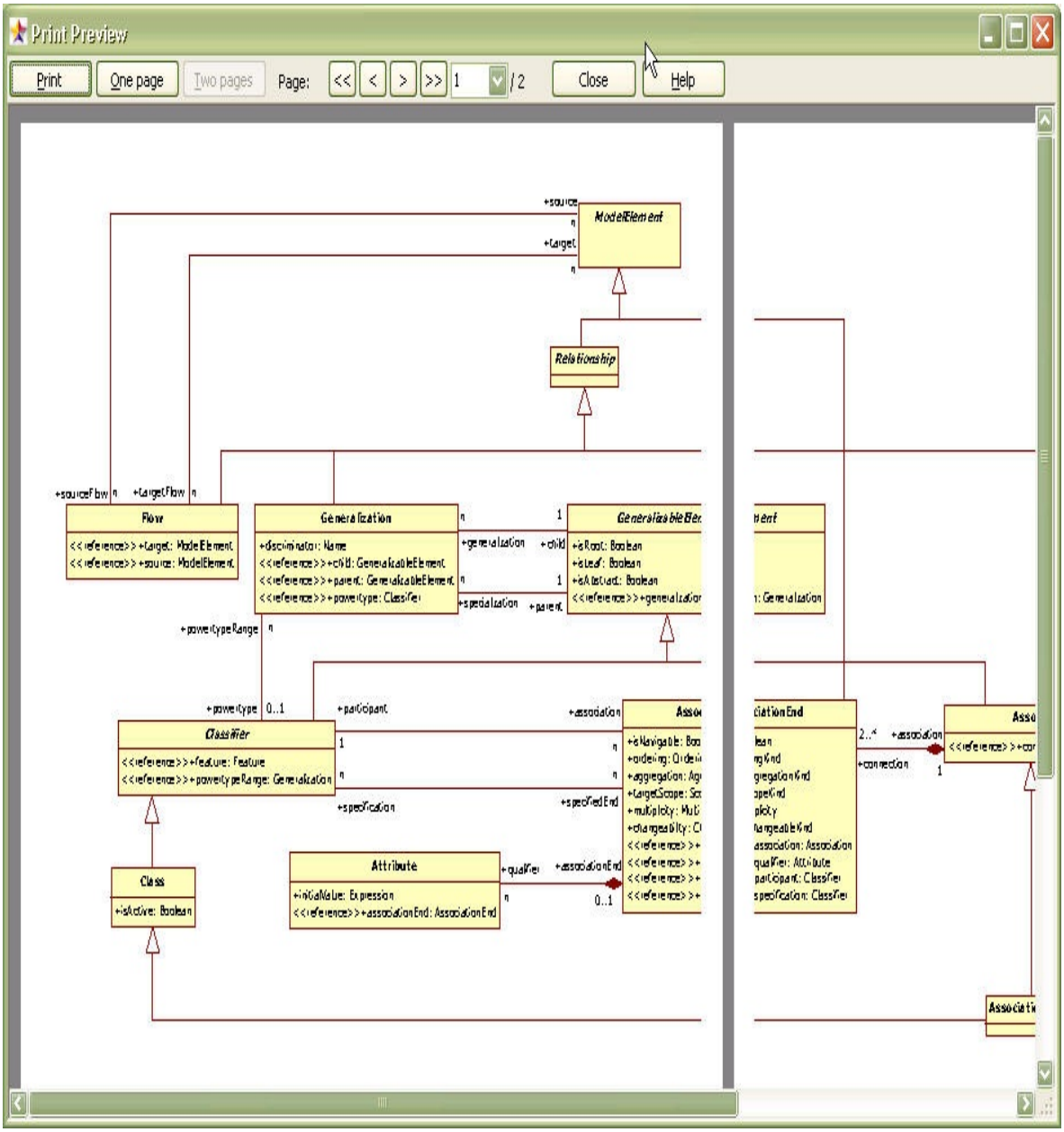
1. Select the diagram(s) to print at the Print dialog box.
2. In the **[Size]** group, select **[Tile]**. Then, enter integers in the **[page(s) wide]** and **[page(s) tall]** fields to specify how many pages will be used (e.g. select 3 pages wide and 2 pages tall to print on 6 pages, i.e., 3*2=6 pages).
3. Click the **[Print]** button.

Print Preview

The print result can be previewed before actually printing it on paper.

Procedure for Previewing Print Result:

1. Select the **[File]** -> **[Print…]** menu to open the Print dialog box and enter the diagram information (see the “Printing Diagram” section).
2. Click the **[Preview…]** button at the bottom of the Print dialog box.
3. At the Print Preview dialog box, preview the print result by selecting to preview by one page or two pages.



4. Click the **[Print]** button to print from this window, or click the **[Close]** button to close the preview window.

User-Interface Reference

This section describes in detail all of the StarUML™ windows.

- Menu
- Toolbars
- Windows
- Dialog Box
- Quick Dialog

Menu

This section describes in detail all of the menu items included in StarUML™s main menu.

- File Menu
- Edit Menu
- Format Menu
- Model Menu
- View Menu
- Tools Menu
- Help Menu
- Shortcuts

File Menu

The File menu contains the following menu items.

Menu Item	Description
New Project[Ctrl+N]	Creates a new project.
New Project By Approach[Ctrl+I]	Opens the Select New Project dialog box.
Open[Ctrl+O]	Opens a project file.
Save[Ctrl+S]	Saves the project file.
Save As[Ctrl+A]	Saves the project as another file.
Close	Closes the current project.
Unit->Control Unit	Separates and saves the currently selected element as a unit.
Unit->Uncontrol Unit	Merges the currently selected unit element to the parent unit (or project).
Unit->Delete Unit	Deletes the currently selected unit element
Unit->Save Unit	Saves the currently selected unit as a file.
Unit->Save Unit As	Saves the currently selected unit as another file.
Import->Framework	Imports a framework into the current project.
Import->Model Fragment	Imports a model fragment into the current project.
Export->Model Fragment	Saves the currently selected element as a model fragment file.
Export Diagram[Shift+Ctrl+D]	Saves the currently active diagram as an image file.
Page Setup	Configures the page for printing
Print[Ctrl+P]	Prints the diagram.
Recent Files	Contains a list of the recently opened files.
Exit	Exits the program.

Edit Menu

The Edit menu contains the following menu items.

Menu Item	Description
Undo[Ctrl+Z]	Cancels the most recent action performed by the user.
Redo[Ctrl+Y]	Repeats the most recent action performed by the user.
Cut[Ctrl+X]	Copies the selected elements to clipboard and removes them from the current location.
Copy[Ctrl+C]	Copies the selected elements to clipboard.
Copy Diagram[Shift+Ctrl+C]	Copies the currently active diagram to clipboard.
Copy Diagram as Bitmap[Shift+Ctrl+C]	Copies the currently active diagram to clipboard as Bitmap.
Paste[Ctrl+V]	Pastes the clipboard contents into the currently selected element (or diagram).
Delete[Del]	Deletes the selected view elements in the diagram.
Delete From Model[Ctrl+Del]	Deletes the selected model elements.
Find[Ctrl+F]	Finds an element.
Select All[Ctrl+A]	Selects all the elements in the current diagram.

Format Menu

The Format menu contains the following menu items.

Menu Item	Description
Font	Specifies the font for the selected view elements.
Line Color	Specifies the line color for the selected view elements.
Fill Color	Specifies the fill color for the selected view elements.
Line Style->Rectilinear[Ctrl+L]	Specifies the line style of the selected connection view element as rectilinear.
Line Style->Oblique[Ctrl+B]	Specifies the line style of the selected connection view element as oblique.
Stereotype Display->None[Shift+Ctrl+N]	Shows nothing for the stereotype of the selected view elements.
Stereotype Display->Textual[Shift+Ctrl+T]	Shows the stereotype of the selected view elements with text.
Stereotype Display->Iconic[Shift+Ctrl+I]	Shows the stereotype of the selected view elements with icons.
Stereotype Display->Decoration[Shift+Ctrl+E]	Shows the stereotype of the selected view elements with decoration.
Suppress Attributes[Shift+Ctrl+A]	Suppresses the section that displays the attributes for the selected view elements (e.g. class, usecase, etc.).
Suppress Operations[Shift+Ctrl+O]	Suppresses the section that displays the operations for the selected view elements (e.g. class, subsystem, etc.).
Suppress Literals[Shift+Ctrl+L]	Suppresses the section that displays the attributes for the selected enumerations.
Word Wrap Name	Shows the Word Wrap of the selected view elements.
Show Parent Name	Shows the parent name of the selected view elements.

Show Operation Signature	Shows the operation signature of the selected view elements (e.g. class, subsystem, etc.).
Show Properties	Shows the property items (e.g. tagged values, changeability attribute, etc.) included in view elements.
Show Compartment Visibility	Shows the visibility of the compartments of the selected view elements (e.g. attribute compartment, operation compartment, etc.).
Show Compartment Stereotype	Shows the stereotypes of the compartments of the selected view elements (e.g. attribute compartment, operation compartment, etc.).
Auto Resize	Automatically resizes the selected view elements.
Alignment-> Bring to Front	Brings the selected elements to front.
Alignment-> Send to Back	Sends the selected elements to back.
Alignment-> Align Left	Aligns the selected elements to left.
Alignment-> Align Right	Aligns the selected elements to right.
Alignment-> Align Middle	Centers the selected elements horizontally.
Alignment-> Align Top	Aligns the selected elements to top.
Alignment-> Align Bottom	Aligns the selected elements to bottom.
Alignment-> Align Center	Centers the selected elements vertically.
Alignment-> Space Equally, Horizontally	Evenly distributes the selected elements horizontally.
Alignment-> Space Equally, Vertically	Evenly distributes the selected elements vertically.
Layout Diagram	Lays out the view elements in the current diagram.

Model Menu

The Model menu contains the following menu items.

Menu Item	Description
Add->...	Adds a model element. The model elements that can be added under the currently selected model elements are displayed in the sub menu.
Add Diagram->...	Adds a diagram. The diagrams that can be added under the currently selected model elements are displayed in the sub menu.
 Collection Editor[Ctrl+F5]	Opens the collection editor that can be used to edit the child elements of the currently selected model element.
 Constraints[Ctrl+F6]	Opens the constraint editor that can be used to edit the constraints of the currently selected model element.
 TaggedValues[Ctrl+F7]	Opens the tagged value editor that can be used to edit the tagged values of the currently selected model element.
 Profiles[Ctrl+F8]	Opens the profile manager.
 Verify Model[F9]	Opens the Verify Model dialog box that can be used to inspect the model elements in the current project.
Convert Diagram->Convert Sequence(Role) to Collaboration(Role)	Generates a new diagram by converting the currently selected sequence (role) diagram into a collaboration (role) diagram (default Add-In function).
Convert Diagram->Convert Collaboration(Role) to Sequence(Role)	Generates a new diagram by converting the currently selected collaboration (role) diagram into a sequence (role) diagram (default Add-In function).

View Menu

The View menu contains the following menu items.

Menu Item	Description
Close Diagram	Closes the currently active diagram.
Close All Diagrams	Closes all open diagrams.
Select In Model Explorer	Shows the currently selected element in the model explorer.
 Referesh	Refreshes the current diagram.
Model Explorer	Toggles the Model Explorer on and off.
Diagram Explorer	Toggles the Diagram Explorer on and off
Properties	Toggles the Properties Editor on and off.
Documentations	Toggles the Documentation Editor on and off.
Attachments	Toggles the Attachments Editor on and off.
Output	Toggles the Output Window on and off.
Messages	Toggles the Message Window on and off.
Toolbox	Toggles the Toolbox on and off.
Zoom->  Zoom In	Makes the diagram look larger.
Zoom->  Zoom Out	Makes the diagram look smaller.
Zoom->  Fit To Window	Automatically adjusts the zoom ratio to fit the whole diagram in the window.
Zoom->50%	Shows the current diagram at 50% zoom ratio.
Zoom->75%	Shows the current diagram at 75% zoom ratio.
Zoom->100%	Shows the current diagram at 100% zoom ratio.
Zoom->125%	Shows the current diagram at 125% zoom ratio.
Zoom->150%	Shows the current diagram at 150% zoom ratio.
	Shows the current diagram at 175% zoom

Zoom->175%	ratio.
Zoom->200%	Shows the current diagram at 200% zoom ratio.
Toolbars->Standard	Toggles the Standard toolbar on and off.
Toolbars->Format	Toggles the Format toolbar on and off.
Toolbars->View	Toggles the View toolbar on and off.
Toolbars->Alignment	Toggles the Align toolbar on and off.

Tools Menu

The Tools menu contains the following menu items.

Menu Item	Description
Options...	Opens the Options dialog box that can be used to edit various environment configuration options.
Add-In Manager...	Opens the Add-In Manager that can be used to manage the additionally installed Add-In programs.

Help Menu

The Help menu contains the following menu items.

Menu Item	Description
 Contents...[F1]	Opens the StarUML™ help.
StarUML On the Web	Moves to the StarUML™ website.
About	Shows the StarUML™ information.

Shortcuts

StarUML™ provides shortcuts to menu functions. The shortcuts can increase efficiency and convenience in software modeling.

Shortcuts List

Shortcut	Menu Item
Del	Delete
F1	StarUML™ Help
F5	Refresh
F6	Browser Window
F7	Inspector Window
F8	Information Window
F9	Verify Model
Ctrl+F4	Close Diagram
Ctrl+F5	Collection Editor
Ctrl+F6	Constraint Editor
Ctrl+F7	Tagged Values
Ctrl+A	Select All
Ctrl+B	Oblique
Ctrl+C	Copy
Ctrl+F	Find
Ctrl+I	Select New Project
Ctrl+L	Rectilinear
Ctrl+M	Show in Model Explorer
Ctrl+N	New Project
Ctrl+O	Open
Ctrl+P	Print
Ctrl+S	Save
Ctrl+V	Paste
Ctrl+X	Cut
Ctrl+Y	Redo
Ctrl+Z	Undo

Ctrl+Del	Delete Model
Shift+Ctrl+F4	Close All Diagrams
Shift+Ctrl+A	Suppress Attributes
Shift+Ctrl+C	Copy Diagram
Shift+Ctrl+D	Export Diagram
Shift+Ctrl+E	Decoration(Stereotype Display)
Shift+Ctrl+I	Icon (Stereotype Display)
Shift+Ctrl+L	Suppress Literals
Shift+Ctrl+N	None (Stereotype Display)
Shift+Ctrl+O	Suppress Operations
Shift+Ctrl+S	Save As
Shift+Ctrl+T	Text (Stereotype Display)

Toolbars

This section describes in detail all of the toolbar items in StarUML™.

- Standard Toolbar
- Format Toolbar
- View Toolbar
- Align Toolbar
- Pallet Toolbar

Standard Toolbar

The Standard toolbar contains the following functions.

Toolbar	Description
New Project[Ctrl+N]	Creates a new project.
Open[Ctrl+O]	Opens a project file.
Save[Ctrl+S]	Saves the project file
Print[Ctrl+P]	Prints the diagram.
Cut[Ctrl+X]	Copies the selected elements to clipboard and removes them from the current location.
Copy[Ctrl+C]	Copies the selected elements to clipboard.
Paste[Ctrl+V]	Pastes the clipboard contents into the currently selected element (or diagram).
Delete[Del]	Deletes the selected view elements in the diagram.
Undo[Ctrl+Z]	Cancels the most recent action performed by the user.
Redo[Ctrl+Y]	Repeats the most recent action performed by the user.
Find[Ctrl+F]	Finds an element.
Collection Editor[Ctrl+F5]	Opens the collection editor that can be used to edit the child elements of the currently selected model element.
Constraints[Ctrl+F6]	Opens the constraint editor that can be used to edit the constraints of the currently selected model element.
TaggedValues[Ctrl+F7]	Opens the tagged value editor that can be used to edit the tag definitions of the currently selected model element.
Profiles[Ctrl+F8]	Opens the profile manager.
Verify Model[F9]	Opens the Verify Model dialog box that can be used to inspect the model elements in the current project.

Format Toolbar

The Format toolbar contains the following functions.

Tool	Description
(Combo) Font Name	Specifies the font name for the selected view elements.
(Combo) Font Size	Specifies the font size for the selected view elements.
 Font...	Specifies the font for the selected view elements.
 Line Color...	Specifies the line color for the selected view elements.
 Fill Color...	Specifies the fill color for the selected view elements.
 Auto Resize	Automatically resizes the selected view elements.
 Stereotype Display	Specifies how the stereotypes will be shown for the selected view elements.
 Show As Extended Notation	Specifies how the extended notation will be shown for the selected view elements.
 Line Style	Specifies the line style for the selected connection view elements.
 Suppress Attributes[Shift+Ctrl+A]	Suppresses the section that displays the attributes for the selected view elements (e.g. class, usecase, etc.).
 Suppress Operations[Shift+Ctrl+O]	Suppresses the section that displays the operations for the selected view elements (e.g. class, subsystem, etc.).
 Suppress Literals[Shift+Ctrl+L]	Suppresses the section that displays the attributes for the selected enumerations.
 Word Wrap Name	Shows the word wrap of the selected view elements.
 Show Parent Name	Shows the parent name of the selected view elements.

Show Operation Signature	Shows the operation signature of the selected view elements (e.g. class, subsystem, etc.).
Show Properties	Shows the property items (e.g. tagged values, changeability attribute, etc.) included in view elements.
Show Compartment Visibility	Shows the visibility of the compartments of the selected view elements (e.g. attribute compartment, operation compartment, etc.).
Show Compartment Stereotype	Shows the stereotype of the compartments of the selected view elements (e.g. attribute compartment, operation compartment, etc.).

View Toolbar

The View toolbar contains the following functions.

Tool	Description
(Combo) Zoom	Selects the zoom ratio for the current diagram.
 Zoom In	Makes the diagram look larger.
 Zoom Out	Makes the diagram look smaller.
 Fit To Window	Automatically adjusts the zoom ratio to fit the whole diagram in the window.
 Refersh[F5]	Refreshes the current diagram.

Align Toolbar

The Align toolbar contains the following functions.

Tool	Description
Layout Diagram	Lays out the view elements in the current diagram.
Bring to Front	Brings the selected elements to front.
Send to Back	Sends the selected elements to back.
Align Left	Aligns the selected elements to left.
Align Right	Aligns the selected elements to right.
Align Middle	Centers the selected elements horizontally.
Align Top	Aligns the selected elements to top.
Align Bottom	Aligns the selected elements to bottom.
Align Center	Centers the selected elements vertically.
Space Equally, Horizontally	Evenly distributes the selected elements horizontally.
Space Equally, Vertically	Evenly distributes the selected elements vertically.

Pallet Toolbar

The Pallet toolbar contains the following functions for selecting and creating elements in the diagram.

Common Pallet Tool

The following functions are always available in the Pallet toolbar regardless of the diagram types.

Tool	Description
Select	The most basic tool that selects, moves or resizes an element in the diagram.
Note	Creates a note element in the current diagram.
Note Link	Links a note in the current diagram to another element.
Text	Creates a string element in the current diagram.
Rectangle	Create a figure of rectangle in the current diagram.
Ellipse	Creates a figure of ellipse in the current diagram.
Rounded Rectangle	Creates a figure of rounded rectangle in the current diagram.

Pallet Tool by Diagram Types

The following functions create elements by diagram types.

Tool	Description	Diagram
Select	The most basic tool that selects, moves or resizes an element in the diagram.	All diagrams
Subsystem	Creates a subsystem element in the current diagram.	Class Diagram
Package	Creates a package element in the current diagram..	Class Diagram, Component Diagram, Deployment Diagram, UseCase Diagram

Class	Creates a class element in the current diagram.	Class Diagram, Composite Diagram
Interface	Creates an interface element in the current diagram.	Class Diagram, Component Diagram, Composite Diagram
Enumeration	Creates an enumeration element in the current diagram.	Class Diagram
Signal	Creates a signal element in the current diagram.	Class Diagram
Except	Creates an exception element in the current diagram.	Class Diagram
Component	Creates a component element in the current diagram.	Component Diagram
ComponentInstance	Creates a component instance element in the current diagram.	Component Diagram
Node	Creates a node element in the current diagram.	Deployment Diagram
NodeInstance	Creates a node instance element in the current diagram.	Component Diagram, Deployment Diagram
Artifact	Creates a artifact in the current diagram.	UseCase Diagram
UseCase	Creates a usecase element in the current diagram.	UseCase Diagram
Actor	Creates an actor element in the current diagram.	UseCase Diagram
SystemBoundary	Creates an system boundary in the current diagram.	UseCase Diagram
Object	Creates an object element in the current diagram.	Class Diagram, Sequence Diagram, Collaboration Diagram

Part	Creates a Part element with a Classifier in the current diagram.	Class Diagram, Component Diagram, Deployment Diagram, Composite Diagram
Port	Creates a Port element with a Classifier in the current diagram.	Class Diagram, Component Diagram, Deployment Diagram, Composite Diagram
ClassifierRole	Creates a ClassifierRole element in the current diagram.	Sequence Role Diagram, Collaboration Role Diagram
Combined Fragment	Creates a Combined Fragment element in the current diagram.	Sequence Diagram, Sequence Role Diagram, Collaboration Diagram, Collaboration Role Diagram
Interaction Operand	Creates a Interaction Operand element with a Combined Fragment in the current diagram.	Sequence Diagram, Sequence Role Diagram, Collaboration Diagram, Collaboration Role Diagram
Frame	Creates a Frame element in the current diagram	Sequence Diagram, Sequence Role Diagram, Collaboration Diagram, Collaboration Role

		Diagram
CompositeState	Creates a CompositeState element in the current diagram.	Statechart Diagram
SubmachineState	Creates a SubmachineState element in the current diagram.	Statechart Diagram
InitialState	Creates an InitialState (Pseudostate) element in the current diagram.	Statechart Diagram, Activity Diagram
FinalState	Creates a FinalState element in the current diagram.	Statechart Diagram, Activity Diagram
Flow Final	Creates a DeepHistory(FlowFinalState) element in the current diagram.	Statechart Diagram, Activity Diagram
ChoicePoint	Creates a Choice (Pseudostate) element in the current diagram.	Statechart Diagram
JunctionPoint	Creates a Junction (Pseudostate) element in the current diagram.	Statechart Diagram
ShallowHistory	Creates a ShallowHistory (Pseudostate) element in the current diagram.	Statechart Diagram
DeepHistory	Creates a DeepHistory (Pseudostate) element in the current diagram.	Statechart Diagram
Synchronization	Creates a Synchronization (Pseudostate) element in the current diagram.	Statechart Diagram, Activity Diagram
ActionState	Creates an ActionState element in the current diagram.	Activity Diagram
SubactivityState	Creates a Subactivity State element in the current diagram.	Activity Diagram
Decision	Creates a Decision (Pseudostate) element in the current diagram.	Activity Diagram
ObjectFlow	Creates a ObjectFlowState element in the current diagram.	Activity Diagram
Signal Accept State	Creates a SignalAcceptState element in the current diagram.	Activity Diagram

Signal Send State	Creates a SignalSendState element in the current diagram.	Activity Diagram
Swimlane(Vertical)	Creates a Swimlane by vertical solid lines in the current diagram.	Activity Diagram
Swimlane(Horizontal)	Creates a Swimlane by horizontal solid lines in the current diagram.	Activity Diagram
Association	Links a semantic association between two classes in the current diagram.	Class Diagram, Component Diagram, Deployment Diagram, UseCase Diagram
DirectedAssociation	Links a semantic association between two classes in the current diagram.	Class Diagram, Deployment Diagram, UseCase Diagram
Aggregation	Links a semantic association between two classes in the current diagram.	Class Diagram
Composition	Links a semantic association between two classes in the current diagram.	Class Diagram
Generalization	Links a generalized element and a specialized element with a generalization relation in the current diagram.	Class Diagram, UseCase Diagram
Dependency	Links two elements with a dependency relation in the current diagram.	Class Diagram, Component Diagram, Deployment Diagram, UseCase Diagram, Composite Diagram
Realization	Links a specification element and its implementation element with a realization relation in the current	Class Diagram, Component Diagram,

	diagram.	Composite Diagram
AssociationClass	Links a class and an association in the current diagram so that the association itself can have the significance of a class.	Class Diagram
Include	Links two UseCases with an Include relation in the current diagram so that one UseCase includes the other UseCase behaviors.	UseCase Diagram
Extend	Links two UseCases with an Extend relation in the current diagram so that one UseCase can be extended with the behaviors defined in the other UseCase.	UseCase Diagram
AssociationRole	Links two roles with an AssociationRole in the current diagram.	Collaboration Role Diagram
SelfAssociationRole	Creates an AssociationRole from one role to the same role in the current diagram.	Collaboration Role Diagram
Link	Links two objects in the current diagram.	Class Diagram, Collaboration Diagram
SelfLink	Links an object with itself in the current diagram.	Class Diagram, Collaboration Diagram
ForwardMessage	Defines a message between two roles in the current diagram.	Sequence Role Diagram, Collaboration Role Diagram
ReverseMessage	Defines a message between two roles in the current diagram.	Sequence Role Diagram, Collaboration Role Diagram

SelfMessage	Creates a message from a role to the same role in the current diagram.	Sequence Role Diagram, Collaboration Role Diagram
ForwardStimulus	Defines a stimulus between two objects in the current diagram.	Sequence Diagram, Collaboration Diagram
ReverseStimulus	Defines a stimulus between two objects in the current diagram.	Sequence Diagram, Collaboration Diagram
SelfStimulus	Creates a stimulus from an object to the same object in the current diagram.	Sequence Diagram, Collaboration Diagram
Transition	Links a source state and a target state with a transition in the current diagram.	Statechart Diagram, Activity Diagram
SelfTransition	Links an original state and a target state with a transition in the current diagram.	Statechart Diagram, Activity Diagram
Connector	Links a original feature and a target feature with a connector in the current diagram.	Class Diagram, Component Diagram, Deployment Diagram, Composite Diagram

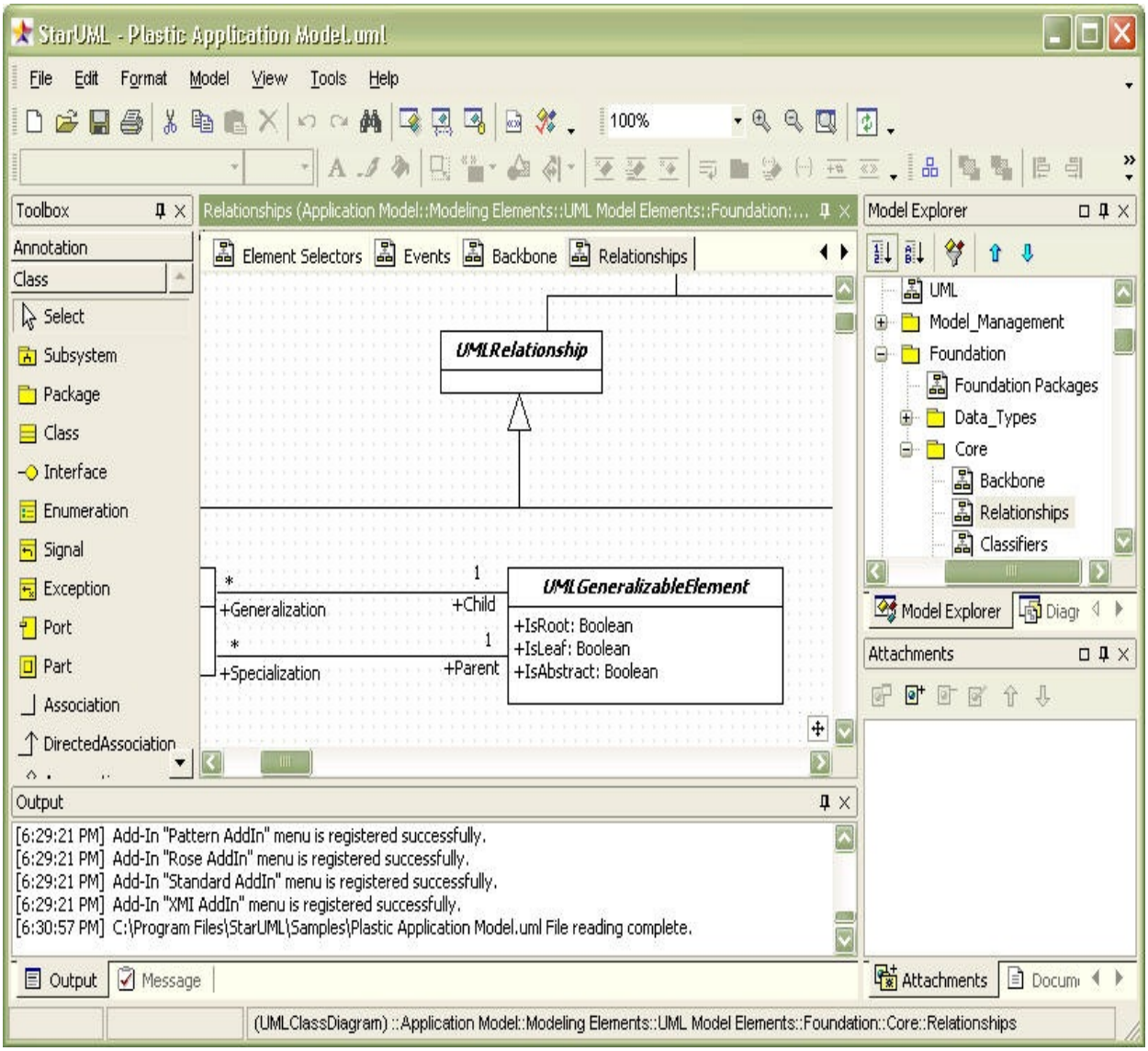
Windows

This section describes in detail all of the StarUML(tm) windows.

- Main Window
- Model Explorer
- Diagram Explorer
- Property Editor
- Documentation Editor
- Attachments Editor
- Output Window
- Messages Window
- Collection Editor
- Tagged Value Editor
- Profile Manager
- Add-In Manager

Main window

StarUML™ Main window consists of the following components.



Main Menu

The main menu is at the top of the screen. Most of StarUML™'s functions are accessible through the main menu.

Toolbars

Toolbars are right below the main menu. They contain frequently used menu items.

Browser Area

The browser area is located in the upper left corner of the screen. This area contains the functions to facilitate easy exploring of the software project component elements. This area includes **[Model Explorer]** which shows the model elements in hierarchical structures, and **[Diagram Explorer]** which shows the diagram types.

Inspector Area

The inspector area is located in the lower left corner of the screen. This area contains the functions to facilitate editing of the detailed information for elements. This area includes **[Property Editor]** which edits properties, **[Documentation Editor]** which records detailed descriptions, and **[Attachments Editor]** which attaches additional files or URLs.

Information Area

The information area is located in the lower right corner of the screen. This area contains the functions to show various types of information throughout the StarUML™ application. This area includes **[Output Window]** which shows log recordings, **[Messages Window]** which shows the model search and inspection results.

Diagram Area

The diagram area is located in the upper right corner of the screen. This area contains the functions to edit and manage the diagrams.

Pallet

Located on the left-hand side of the area is **Pallet,** which contains the elements that can be created.

Model Explorer

The Model Explorer supports the user to effectively manage and explore the model elements by showing them in hierarchical structures. Select the **[Model Explorer]** tab in the **[Browser]** area to open the Model Explorer.

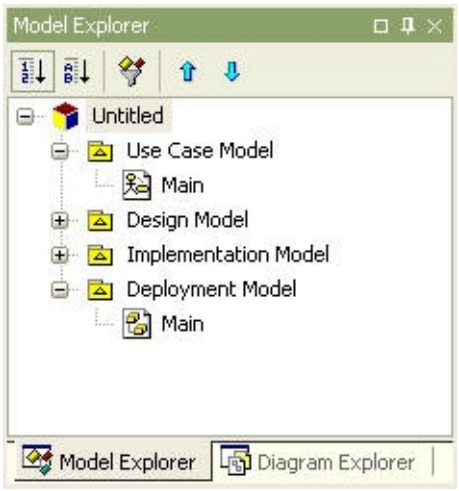
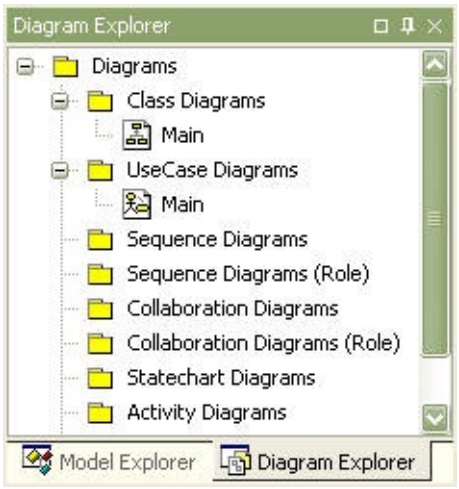


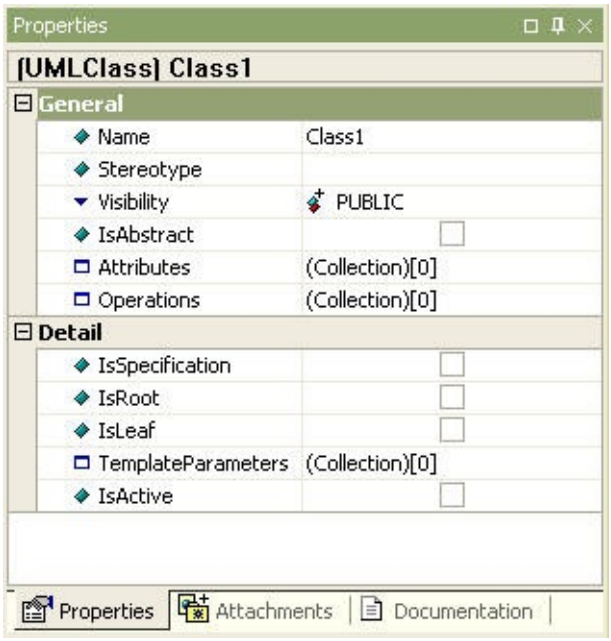
Diagram Explorer

The Diagram Explorer supports the user to effectively manage and explore the diagrams by listing them by their types. Select the **[Diagram Explorer]** tab in the **[Browser]** area to open the Diagram Explorer.



Property Editor

The Property Editor is used for editing the detailed properties of the currently selected model element. Select the **[Properties]** tab in the **[Inspector]** area to open the Property Editor.

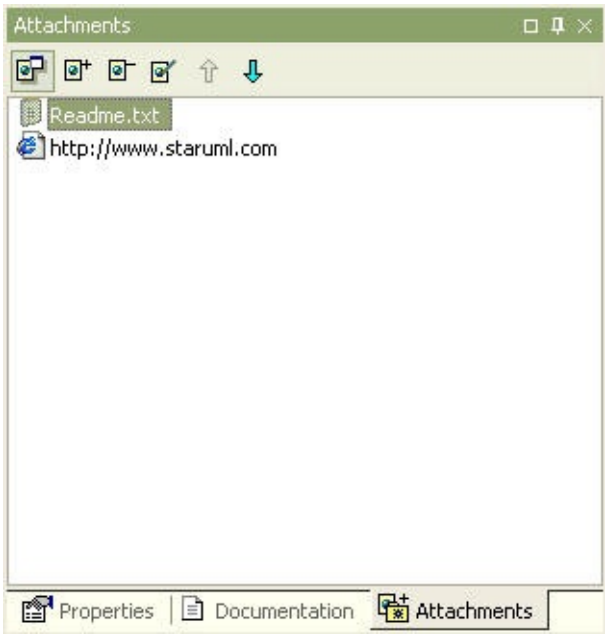


Documentation Editor

The Documentation Editor is used for recording additional descriptions of the currently selected element. Select the **[Documentation]** tab in the **[Inspector]** area to open the Documentation Editor.

Attachments Editor

The Attachments Editor allows the user to attach files or web URLs to a specific element. Select the **[Attachments]** tab in the **[Inspector]** area to open the Attachments Editor.



Attachment List

Shows a list of the files or URLs attached to the element.

Open Button

Opens the selected attachment file or URL with the associated program. For example, if a .doc file is selected, it is automatically opened in Microsoft Word, and if a web address such as http://www.staruml.com is selected, it is opened in the web browser.

Add Button

Attaches a new file or URL. Click this button to open the Attachment Item dialog box.

Remove Button

Removes the selected item from the attachment list.

Edit Button

Edits the selected item from the attachment list. The Attachment Item Editor can be used to change the file name or enter another URL.

Move Up Button

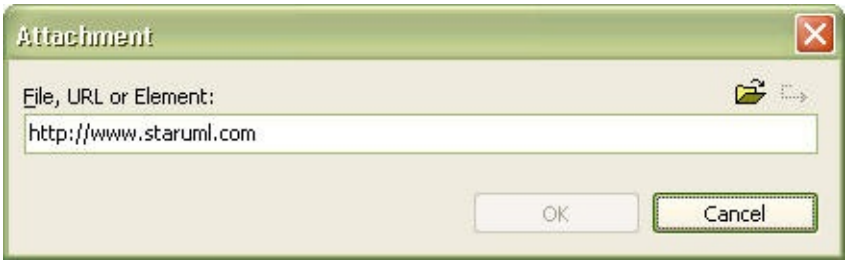
Moves the selected item up in the attachment list.

Move Down Button

Moves the selected item down in the attachment list.

Attachment Item Dialog Box

Edits the attachment item name. Enter a URL or pathname for a file. The button on the right can be used to select a file.

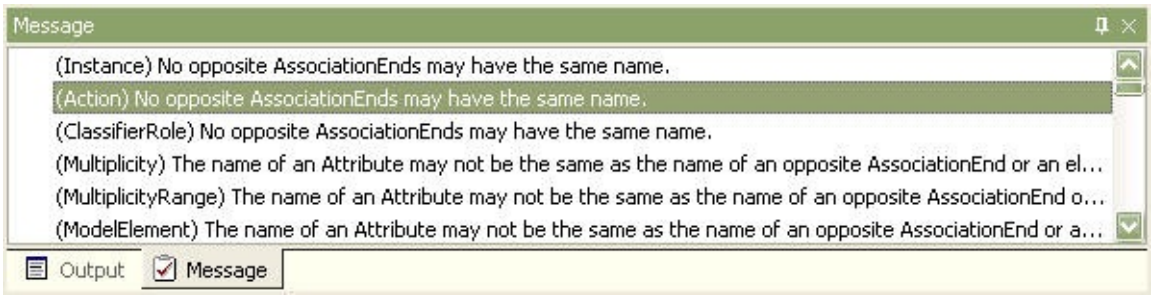


Output Window

The Output Window keeps and shows a record of the events in StarUML™. Select the **[Output]** tab in the **[Information]** area to open the Output Window.

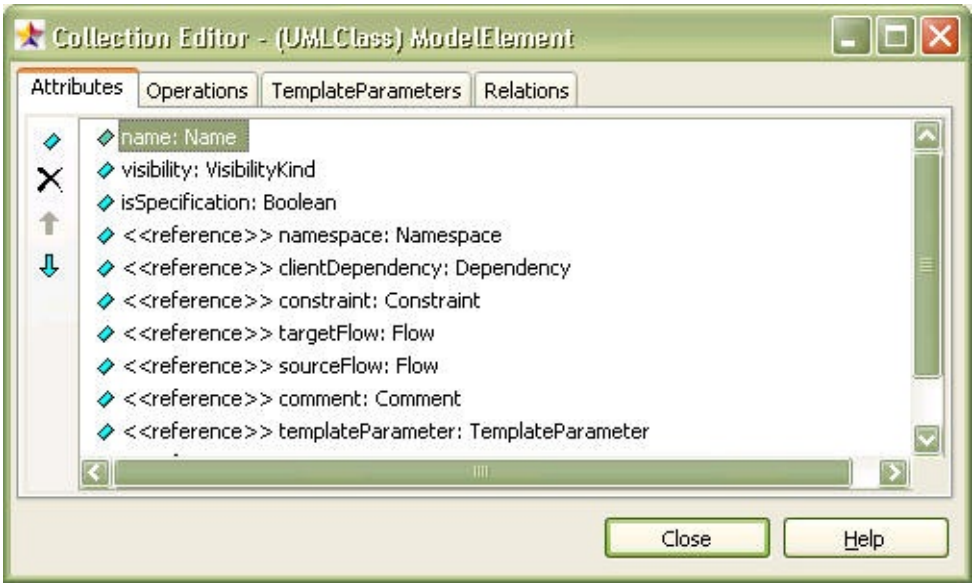
Messages Window

The Message Window shows the results of element search or software model inspection. Select the **[Messages]** tab in the **[Information]** area to open the Message Window.



Collection Editor

The Collection Editor is used for managing a list of child elements for a specific element.



Tab

Shows collections (a list of child elements contained in the current element) contained in the element by tabs. Different types of elements have different collections, and therefore have different tabs. For example, Class element has tabs for Attributes and Operations. The Relations tab is always present regardless of the element type.

Collection Element List

Shows a list of the child elements. Select an element here and edit it using the property editor, documentation editor, and attachment editor in the inspector area. For showing element stereotype, visibility/stereotype, etc., please refer to the section on General Configurations, in Environment Configurations.

Add Button

Creates a new element and adds it to the list. This button may connect existing elements instead of creating a new element (e.g. Residents, DeployedComponents, RaisedSignals).

Delete Button

Deletes the selected element in the collection element list. This button may remove the element from the list instead of deleting it (e.g. Residents, DeployedComponents, RaisedSignals).

Move Up Button

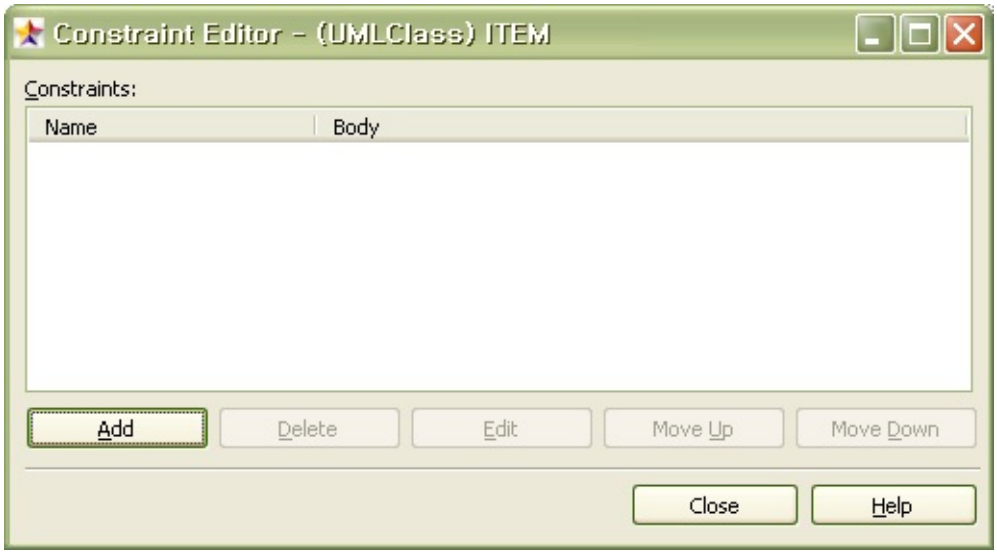
Moves the selected element up in the collection element list.

Move Down Button

Moves the selected element down in the collection element list.

Constraint Editor

The Constraint Editor is used for managing the constraints for elements.



Constraints

Shows the names and contents of the constraints for elements.

Add

Adds a new constraint to the element. This button opens the Constraints dialog box.

Delete

Deletes the selected constraint in the constraints list.

Edit

Edits the selected constraint in the constraints list.

Move Up

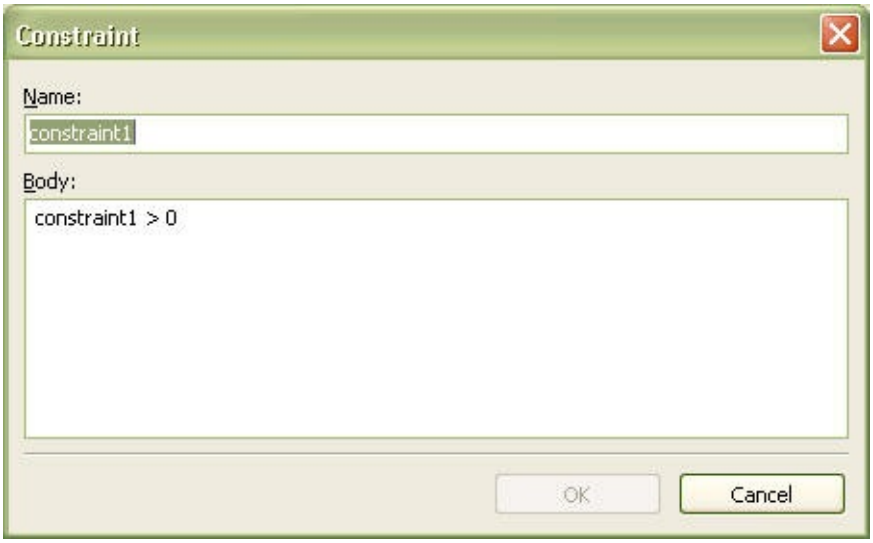
Moves the selected constraint up in the constraints list.

Move Down

Moves the selected constraint down in the constraints list.

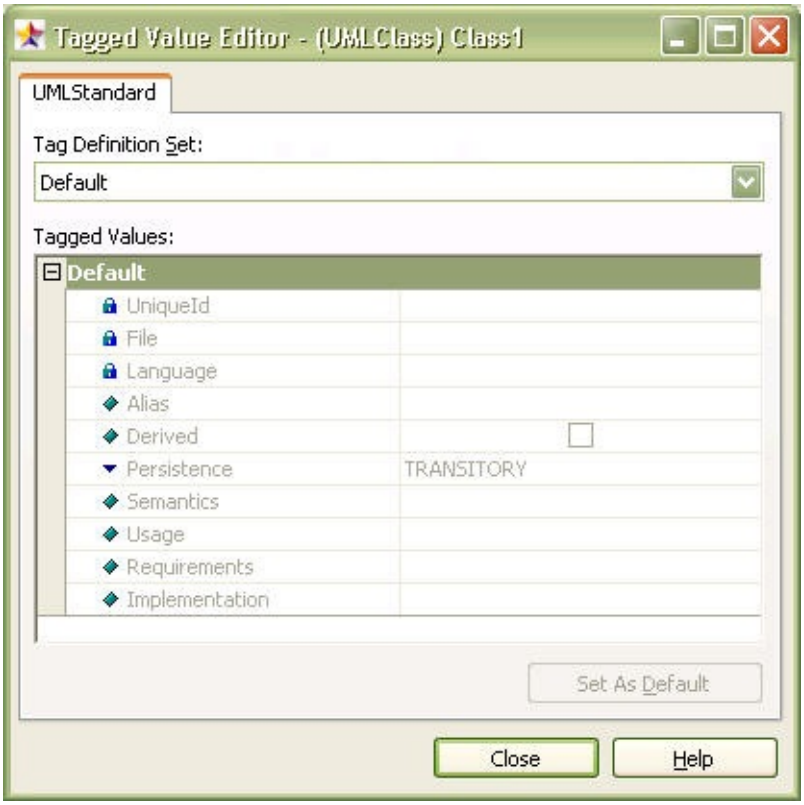
Constraint Dialog Box

This is used for adding a new constraint or editing the name and/or contents of existing constraints in the Constraint Editor. Enter the name of the constraint in the **[Name]** field and enter the contents of the constraint in the **[Expression]** field. The user may freely enter any contents or write in the UML OCL (Object Constraint Language).



Tagged Value Editor

The Tagged Value Editor is used for editing the tagged values that can be added to specific elements.



Profile Tab

By default, tagged values are defined in profiles. If there is a profile that contains the tagged values which can be applied to the currently selected element, it is shown as a tab. The tag definitions defined in the profile are displayed in the **[Tag Definition Set]** and **[Tagged Values]** fields.

Tag Definition Set

Shows the tag definition set that can be applied to the currently selected element. The tagged values included in this set are displayed in the **[Tagged Values]** field.

Tagged Values

Lists the definitions and their values included in the tag definition set selected in **[Tag Definition Set]**. The user may directly change the values.

Set As Default

Every tag definition has a default value. Select a tag definition in **[Tagged Values]** and click this button to clear the changed value and set it back to the default value.

Profile Manager

The Profile Manager can be used for including or excluding the UML profiles for the current project.

Available profiles

Shows a list of the UML profiles registered for use in StarUML™. Profiles currently in use by the current project are not shown here.

Include profiles

Shows a list of the UML profiles in use by the current project.

Large Icon/Small Icon Button

Toggles the profile list icon size between large and small. Select the Small Icon Button if the profile names are only partially shown and difficult to read.

Include

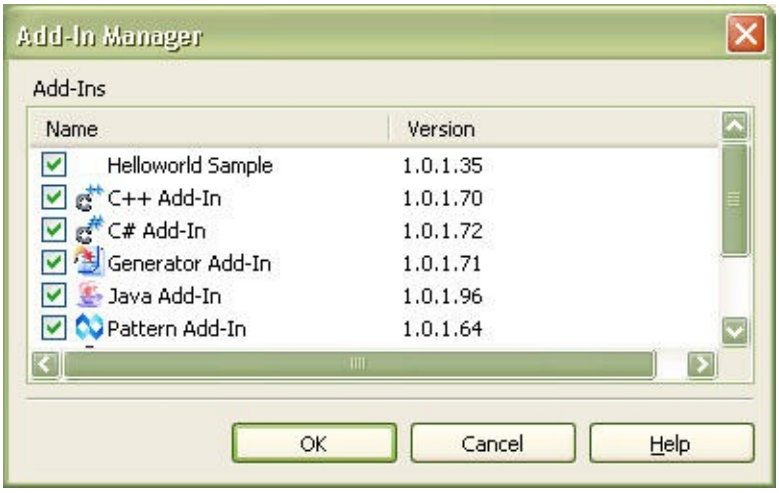
Includes the profile selected in the available profile list for use by the current project.

Exclude

Excludes the profile selected in the included profile list so that it is no longer used by the current project.

Add-In Manager

The Add-In Manager can be used to view a list of the installed Add-Ins and to enable or disable the Add-Ins.



Add-Ins List

Shows a list of the installed Add-Ins. The user can check or uncheck each item to enable or disable the respective Add-In.

Note

- The list of Add-Ins in the Add-In Manager window may vary according to the users installation environment.

Dialog Box

This section describes in detail the dialog box available in StarUML(tm).

- Select New Project Dialog Box
- Import Framework Dialog Box
- Page Setup Dialog Box
- Print Dialog Box
- Print Preview Dialog box
- Find Dialog Box
- Verify Model Dialog Box
- Options Dialog Box
- Select Stereotype Dialog Box
- Select Element Dialog Box
- Element List Dialog Box
- Model Filtering Dialog Box

Select New Project Dialog Box

The Select New Project dialog box provides various selection methods when creating a new project. The New Project dialog box consists of three pages: Select Approach, Open Existing File, and Open Recent File.

Approaches

The user can apply a specific approach for creating a new project.

- **Approaches**: The approaches list displays the names and icons of the registered approaches. Select the Empty Project item if no approach is needed.

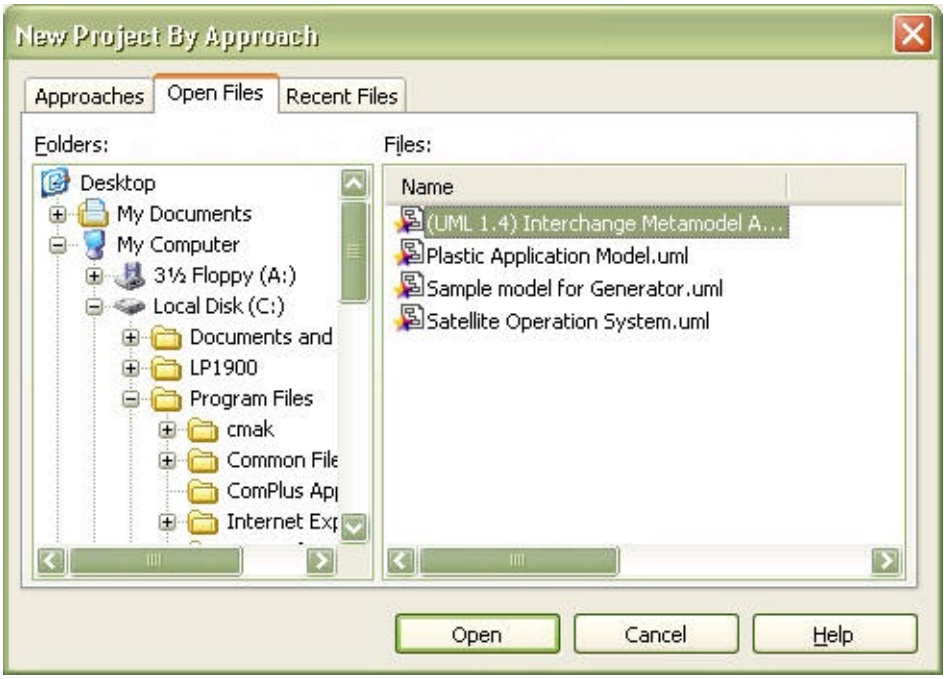
- **Large Icon/Small Icon Button:** This toggles the icon size for the approach list. Select the small icon button if the approach names are shortened and difficult to read.

- Description: This area shows a brief description of the approach item selected from the list.

  Use the selected Approach by default : Select an approach from the list and check this check box to set the approach as the default approach. The default

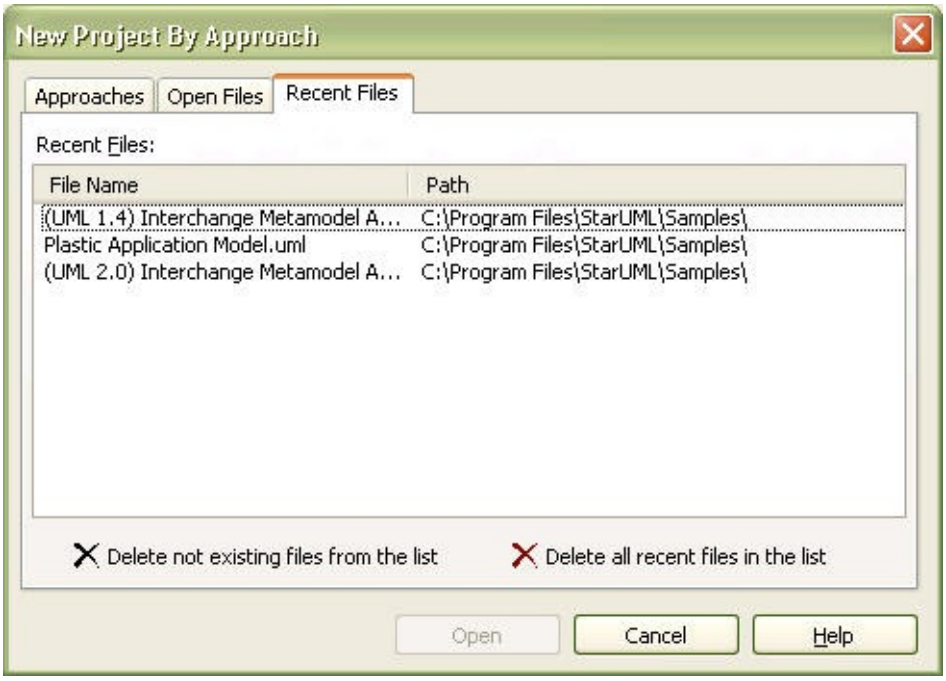
- approach is applied when creating a new project by selecting the **[File] -> [New] -> [New Project]** menu.

Open Files



The user can open a previously created file. The tree view on the left shows the user systems folder structure, and the file list area on the right shows the project files in the selected folder. Select a file from this file list and click the **[Open]** button to open the selected file.

Recent Files

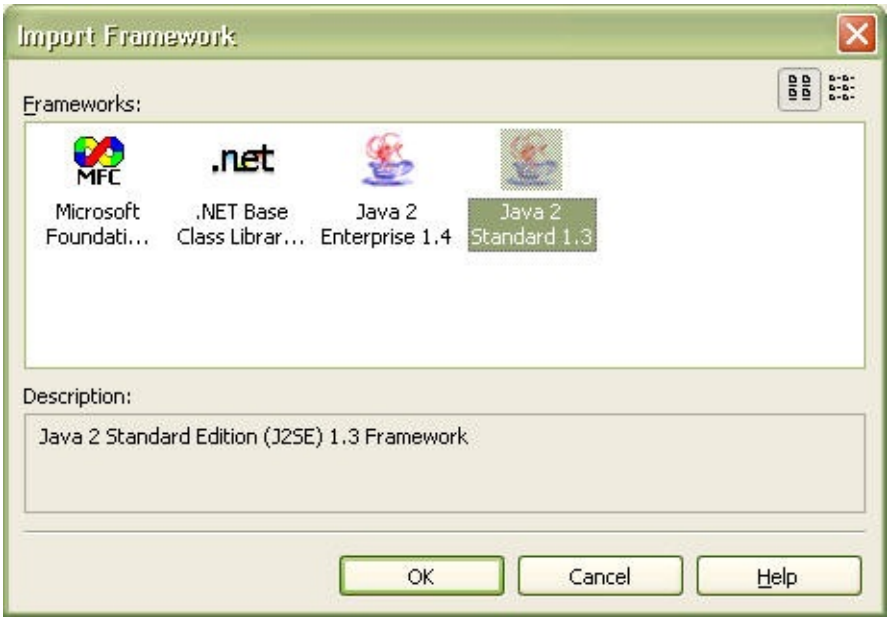


The user can see a list of the recently edited files and open them.

- **Recently modified files:** Shows a list of the recently edited files.
- **Remove non-existent files from the list:** Checks for files that no longer exist and removes them from the recent files list.
- **Clear the recent files list:** Clears all the files in the recent file list. The recent file list in the system registry is deleted.

Import Framework Dialog Box

The Import Framework dialog box allows the user to select an available framework and load it to the current project.



Frameworks list

The frameworks list displays the names and icons of the registered frameworks. Select a framework to load.

Large Icon/Small Icon Button

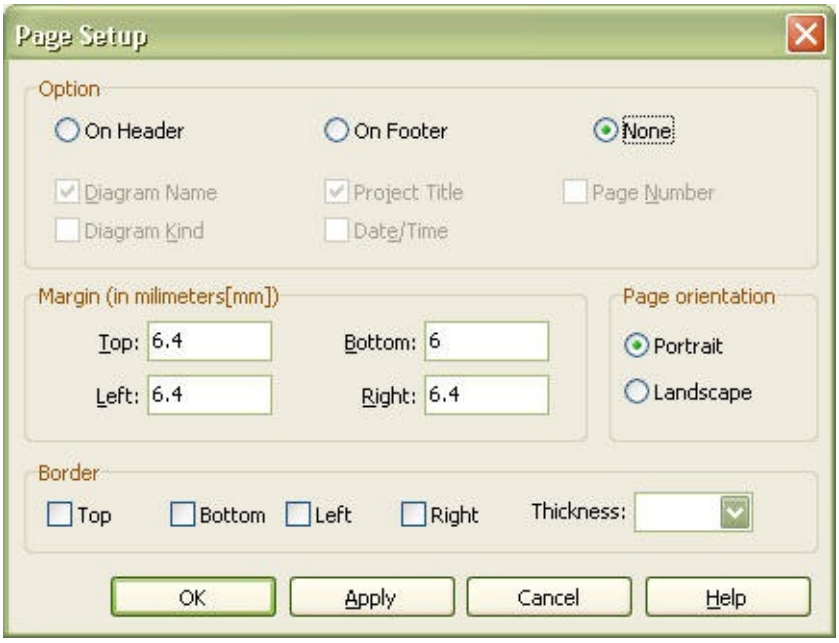
This toggles the icon size for the framework list. Select the small icon button if the framework names are shortened and difficult to read.

Description

This area shows a brief description of the framework item selected from the list.

Page Setup Dialog Box

The Page Setup dialog box allows the user to specify what and how diagram information is printed, the paper orientation, margins, outlines, etc.



Option

The user can specify some of the diagram information to be printed.

- On Header : Prints the diagram information in the page header.
- On Footer : Prints the diagram information in the page footer.
- None : Does not print the diagram information.
- Diagram Name : Prints the diagram name.
- Project Title : Prints the project name of the diagram.
- Page Number : Prints the page number.
- Diagram Kind : Prints the diagram kind.
- Date/Time :Prints the current date and time.

Margin

The user can specify the top, bottom, left, and right page margins in millimeters.

Page orientation

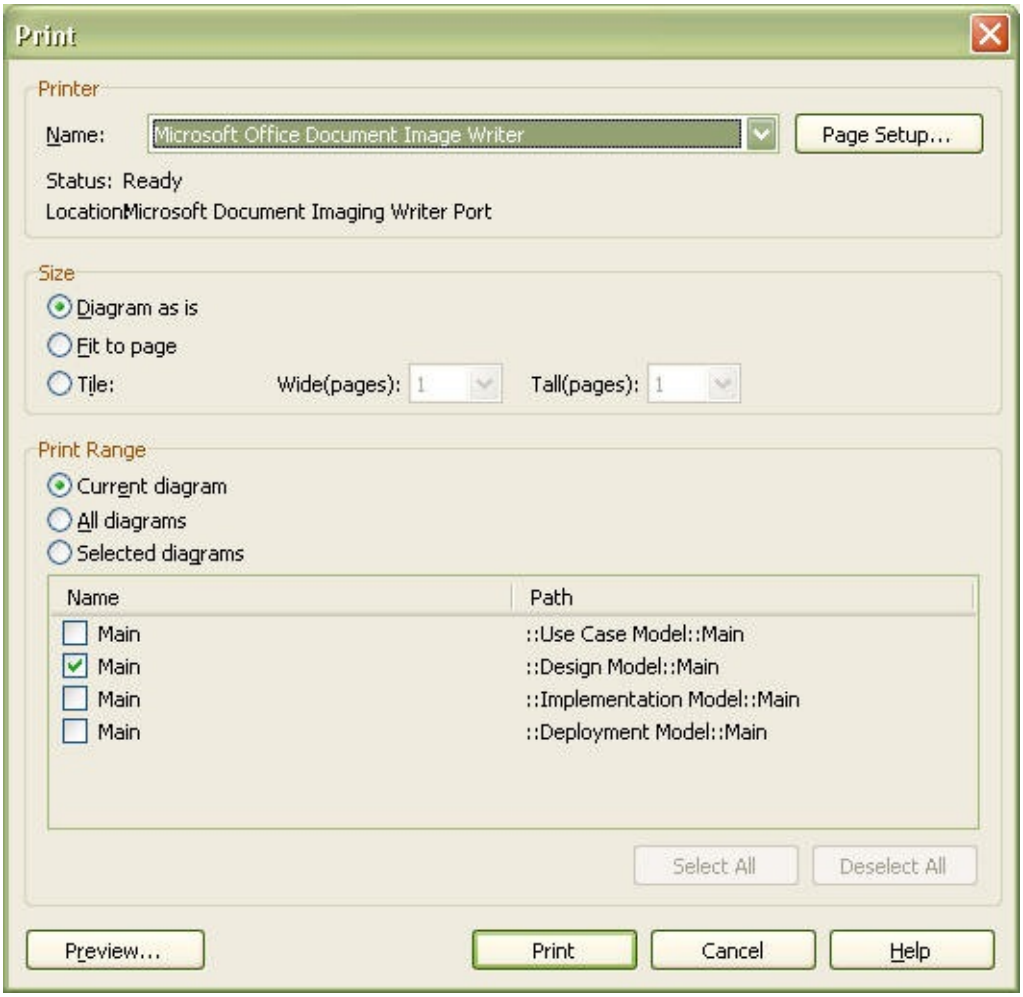
The user can specify whether to print the page in portrait or landscape.

Border

The user can specify how the page border will be printed. Select top, bottom, left, or right for drawing border and specify the border thickness.

Print Dialog Box

The Print dialog box appears when the user prints a diagram. The user can select and specify various options related to printing.



Printer

The user can configure the printer-related options.

- Name : Select a printer to use from the installed printers.
- Status : Indicates the status of the selected printer.
- Location : Indicates the location of the selected printer.
- Page Setup : Opens the Page Setup dialog box.

Size

Specifies the size of the diagram to print.

- Diagram as is : Prints the diagram in its original size. The diagram is printed in multiple pages if it does not fit in one page.
- Fit to page : Prints the diagram to fit in one page. This option prevents printing of multiple pages if the diagram is large.
- Tile : Prints the diagram to fit in multiple pages. The user can specify the number of pages to print by width and height (e.g. 3 pages wide and 4 pages tall = total 12 pages).

Print Range

Specifies the range of the diagram to print.

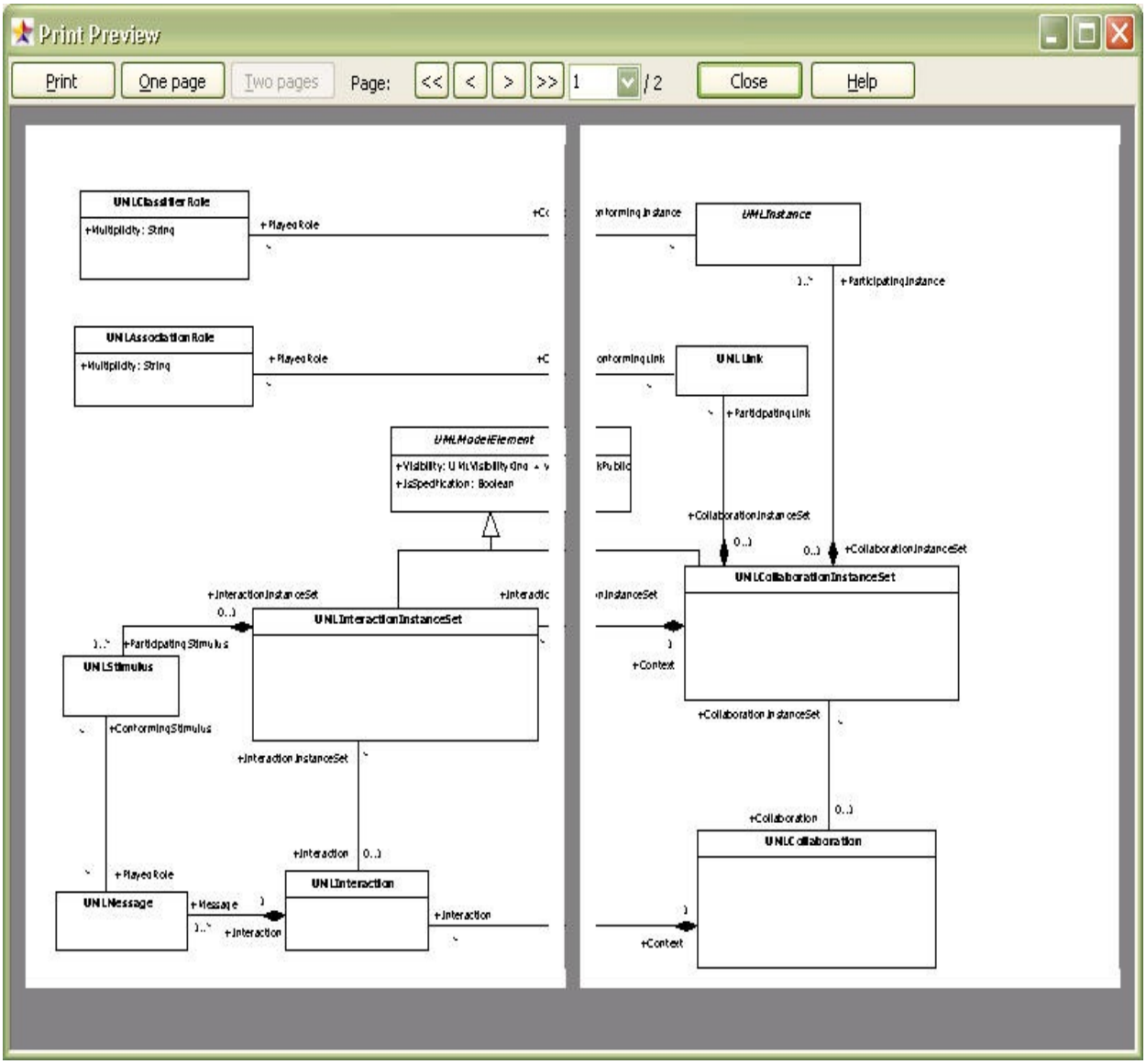
- Current diagram : Prints only the currently active diagram.
- All diagrams : Prints all of the diagrams in the current project.
- Selected diagrams : Prints only the selected diagram. The **[Select All]** button selects all diagrams, and the **[Deselect All]** button deselects all diagrams.

Preview

Opens the Preview dialog box.

Print Preview Dialog Box

The Print Preview dialog box allows the user to preview the print result before actually printing the diagram.



Print

Starts printing.

One Page / Two Pages

Toggles preview by one page or two pages.

<<, <, >, >>

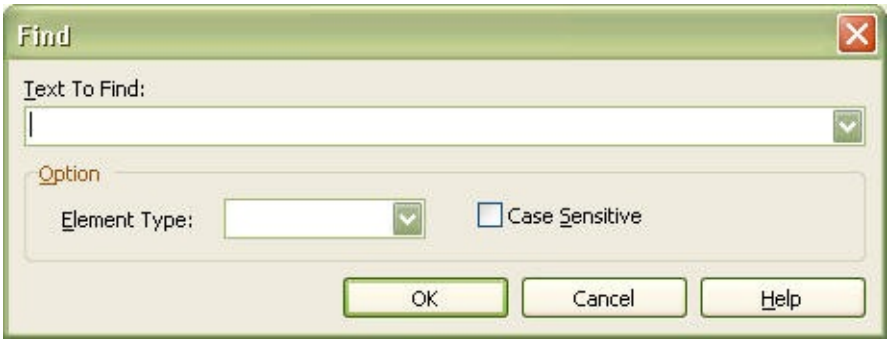
Allows navigation to the first page, previous page, next page, and last page.

Page Selection

The user can move to a specific page by directly entering the page number.

Find Dialog Box

The Find dialog box allows the user to find elements quickly and easily.



Text to Find

Enter the full or partial string for the element to find. The user can also select from the previously entered strings.

Option-Element Type

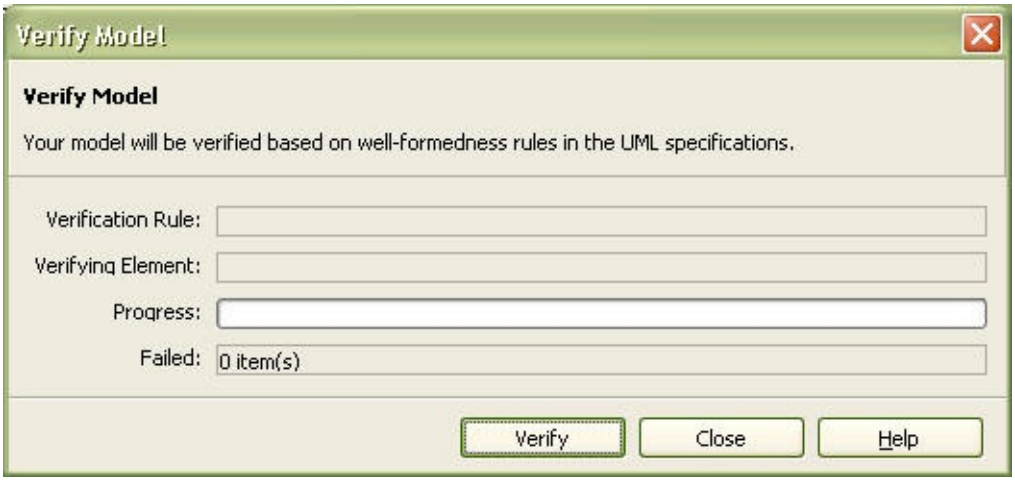
This specifies the range of elements to find. Available ranges: All elements, Model, Subsystem, Package, Class, Interface, Enumeration, Signal, Exception, Component, Node, Instance, UseCase, and Actor.

Option-Case Sensitive

This specifies lowercase or uppercase for the element to find.

Verify Model Dialog Box

The Verify Model dialog box is used to inspect the model elements and their definitions.



Verification Rule

Shows the verification rule currently being applied.

Verifying Element

Shows the name of the element currently being verified.

Progress

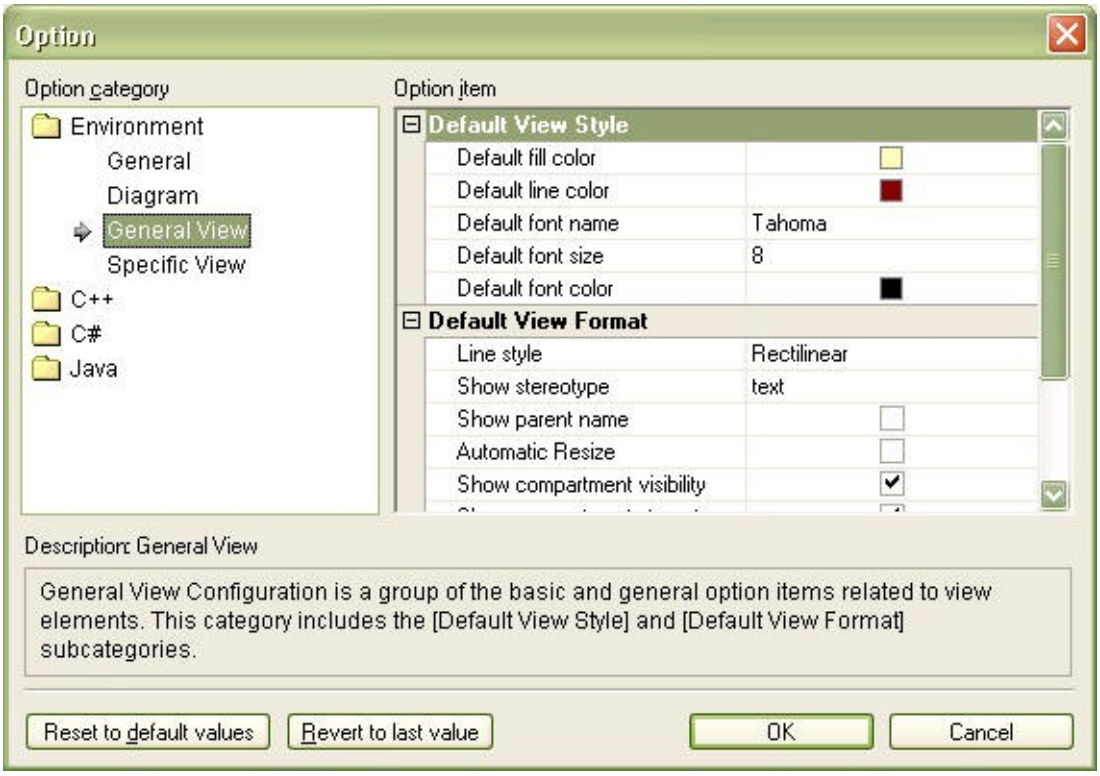
Visually displays the progress of the verification.

Failed

Indicates the number of the elements that failed the verification.

Options Dialog Box

The Options dialog box lists the various option items for environment configuration of StarUML™ and allows the user to edit them.



Option category

This list shows the option categories. The top category is Environment which contains the sub-categories General, Diagram, General View, and Specific View. Additional option categories may be present depending on the module of StarUML™.

Option item

Shows the option items contained in the selected option category. The option values can be edited.

Description

Shows a brief description of the selected option category or item.

Reset to default values

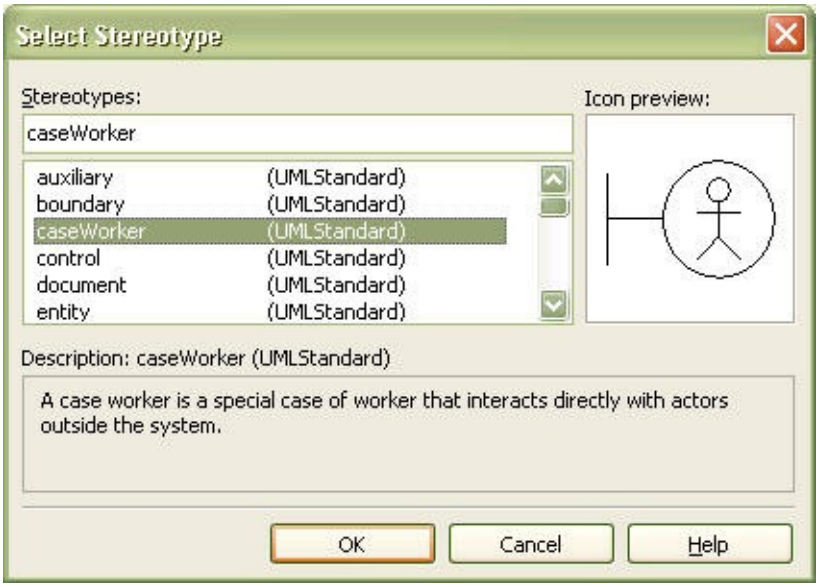
Sets the selected option item value to the default value.

Revert to last value

Reverts the selected option item value to the last saved value.

Select Stereotype Dialog Box

The Select Stereotype dialog box appears when the user needs to specify a stereotype for the selected element. The stereotype can be entered directly or selected from the list.



Stereotypes

The user can directly enter the stereotype. If a stereotype has been registered, it is indicated in the stereotype list.

Stereotypes List

Shows the stereotypes defined in the UML profiles that are in use by the current project. The name of the stereotype and the name of the project that contains it are shown together. The user can select a stereotype from the list.

Icon preview

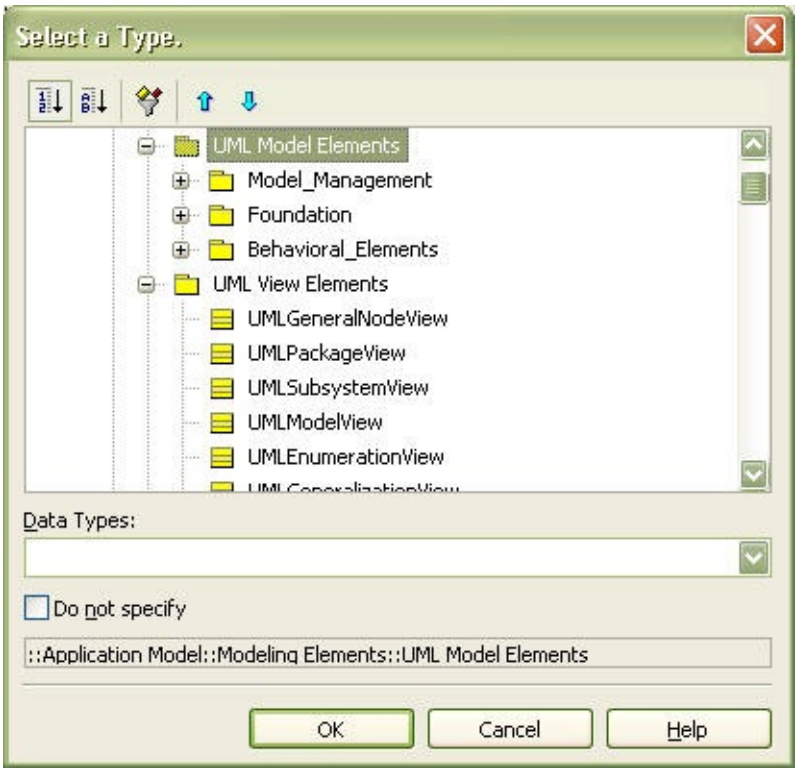
The icon is shown if the selected stereotype is associated with an icon.

Description

Shows the description for the selected stereotype.

Select Element Dialog Box

The Select Element dialog box allows the user to select an element from the hierarchical structure of the project elements. The Select Element dialog box appears when the user needs to assign an element at the property editor, collection editor, etc. Unlike the Element List dialog box, the Select Element dialog box lists the elements in a hierarchical structure.



Dialog Box Title

The dialog box title changes according to the type of the element to be selected. An appropriate title is displayed for defining the attribute type, or the object type (i.e., Classifier).

Element List

Shows the available elements. For example, only the StateMachine elements are displayed when selecting a StateMachine element.

Data Types

Shows the available data types. The data types shown here are those defined in the UML profiles which are in use by the current project. This list may not be shown if a data type does not need to be specified.

Do not specify

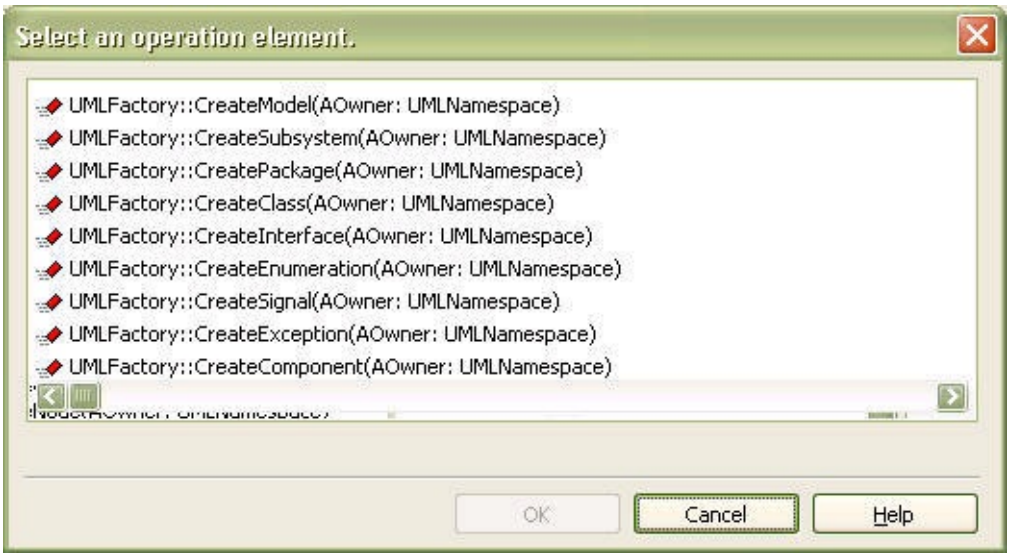
Check this to specify nothing. This actually assigns a null value.

Selected Element

The bottom part of the dialog box shows the full pathname of the selected element. This information can be used to verify which element is currently selected.

Element List Dialog Box

The Element List dialog box allows the user to select an element from a list. The Select Element dialog box appears when the user needs to assign an element to a specific property at the property editor, collection editor, etc. Unlike the Select Element dialog box, the Element List dialog box lists the elements in a one-dimensional list.



Dialog Box Title

The dialog box title changes according to the type of the element to be selected. An appropriate title is displayed for selecting a StateMachine, or for assigning a component in a node.

Element List

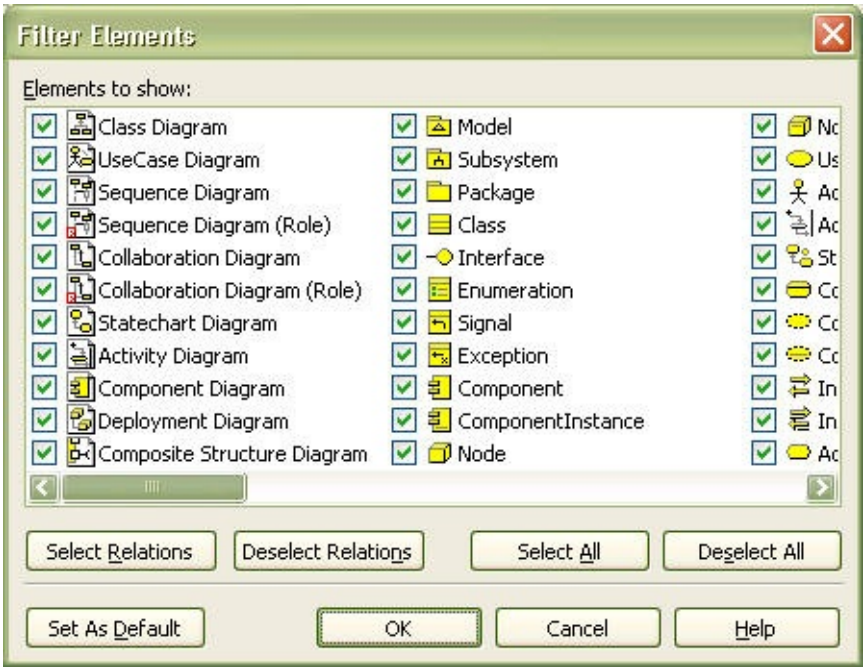
Shows the available elements. For example, only the StateMachine elements are displayed when selecting a StateMachine element.

Do not specify

Check this to specify nothing. This actually assigns a null value.

Model Filtering Dialog Box

The Model Filter dialog box can be used to show or hide specific elements in the model explorer.



Element to show

Shows all the elements that can be displayed in the model explorer. Only those checked are displayed in the model explorer.

Select Relations

Selects all the relationship elements (*Transition, Dependency, Association, AssociationClass, Generalization, Link, AssociationRole, Stimulus, Message, Include, Extend, and Realization*) from the elements list.

Deselect Relations

Deselects all the relation elements.

Select All

Selects all elements.

Deselect All

Deselects all elements.

Set As Default

Selects the elements set as default by the program.

Quick Dialog

This section describes in detail all the quick dialogs available in StarUML™.

- General Quick Dialog
- Subsystem Quick Dialog
- Classifier Quick Dialog
- Enumeration Quick Dialog
- Literal Quick Dialog
- Attribute Quick Dialog
- Operation Quick Dialog
- AssociationEnd Quick Dialog
- Object Quick Dialog
- ClassifierRole Quick Dialog
- Message/Stimulus Quick Dialog
- State Quick Dialog
- Action Quick Dialog
- Note/Text Quick Dialog

General Quick Dialog

General Quick Dialog is the most general form of the quick dialogs. This is used for most of the elements. This appears when an element is double-clicked in diagram. Hitting the **[Enter]** key or clicking outside the quick dialog applies the changes.

Visibility Button

Element visibility can be selected from Public, Protected, Private, and Package.

Edit Field

Element name, visibility and stereotype can be entered in the edit field according to the syntax.

Syntax

*<< **stereotype** >> **visibility name***

- *<< **stereotype** >>* : Enter the stereotype name. This may be omitted.
- ***visibility*** : Enter the character that corresponds to the elements visibility (+: public, #: protected, -: private, ~: package). This may be omitted.
- ***name*** : Enter the elements name.

Subsystem Quick Dialog

Subsystem Quick Dialog is applied only to subsystem elements. This appears when a subsystem is double-clicked in a diagram. Hitting the **[Enter]** key or clicking outside the quick dialog applies the changes.

Visibility Button 

Subsystem visibility can be selected from  Public,  Protected,  Private, and  Package.

Edit Field 

Subsystem name, visibility and stereotype can be entered in the edit field according to the syntax.

Syntax

<< ***stereotype*** >> ***visibility name***

- << ***stereotype*** >> : Enter the stereotype name. This may be omitted.
- ***visibility*** : Enter the character that corresponds to the subsystems visibility (+: public, #: protected, -: private, ~: package). This may be omitted.
- ***name*** : Enter the subsystems name.

Add Operation Button 

Creates and adds a new operation.

Elements Applied

Subsystem

Classifier Quick Dialog

Classifier Quick Dialog is applied only to the elements that fall in the Classifier category (e.g. Class, Actor, Signal, ...). Hitting the **[Enter]** key or clicking outside the quick dialog applies the changes.

Visibility Button

Element visibility can be selected from Public, Protected, Private, and Package.

Edit Field

Element name, visibility and stereotype can be entered in the edit field according to the syntax.

Syntax

<< ***stereotype*** >> ***visibility name***

- << ***stereotype*** >> : Enter the stereotype name. This may be omitted.
- ***visibility*** : Enter the character that corresponds to the elements visibility (+: public, #: protected, -: private, ~: package). This may be omitted.
- ***name*** : Enter the elements name.

Add Attribute Button

Creates and adds a new attribute.

Add Operation Button

Creates and adds a new operation.

Elements Applied

Class, Interface, Signal, Exception, Actor, UseCase, Artifact

Enumeration Quick Dialog

Enumeration Quick Dialog is applied only to enumeration elements. This appears when an enumeration is double-clicked in a diagram. Hitting the **[Enter]** key or clicking outside the quick dialog applies the changes.

Visibility Button

Enumeration visibility can be selected from Public, Protected, Private, and Package.

Edit Field

Enumeration name, visibility and stereotype can be entered in the edit field according to the syntax.

Syntax

<< ***stereotype*** >> ***visibility name***

- << ***stereotype*** >> : Enter the stereotype name. This may be omitted.
- ***visibility*** : Enter the character that corresponds to the enumerations visibility (+: public, #: protected, -: private, ~: package). This may be omitted.
- ***name*** : Enter the enumerations name.

Add Literal Button

Creates and adds a new literal.

Add Operation Button

Creates and adds a new operation.

Elements Applied

Enumeration

Literal Quick Dialog

Literal Quick Dialog is applied only to literal elements. This appears when a literal is double-clicked in a diagram. Hitting the **[Enter]** key or clicking outside the quick dialog applies the changes.

Visibility Button

Literal visibility can be selected from Public, Protected, Private, and Package.

Edit Field

Literal name, visibility and stereotype can be entered in the edit field according to the syntax.

Syntax

<< ***stereotype*** >> ***visibility name***

- << ***stereotype*** >> : Enter the stereotype name. This may be omitted.
- ***visibility*** : Enter the character that corresponds to the literals visibility (+: public, #: protected, -: private, ~: package). This may be omitted.
- ***name*** : Enter the literals name.

Add Button

This adds a new literal in the next location. Hitting **[Ctrl + Enter]** has the same effect. To insert in the current location, hit the **[Ins]** key.

Delete Button

This deletes the literal. Hitting **[Ctrl + Del]** has the same effect.

This moves the current literal up. Hitting **[Ctrl + Up]** has the same effect. To edit the upper literal, just hit the **[Up]** key.

Move Up Button 

This moves the current literal up. Hitting **[Ctrl + Up]** has the same effect. To edit the upper literal, just hit the **[Up]** key.

Mouse Down Button 

This moves the current literal down. Hitting **[Ctrl + Down]** has the same effect. To edit the lower literal, just hit the **[Down]** key.

Elements Applied

Literal

Attribute Quick Dialog

Attribute Quick Dialog is applied only to attribute elements. This appears when an attribute is double-clicked in a diagram. Hitting the **[Enter]** key or clicking outside the quick dialog applies the changes.

Visibility Button

Attribute visibility can be selected from Public, Protected, Private, and Package.

Edit Field

Attribute stereotype, visibility, name, type, multiplicity, order and default value can be entered in the edit field according to the syntax.

Syntax

<< ***stereotype*** >> ***visibility name : type = initialvalue***

- << ***stereotype*** >> : Enter the stereotype name. This may be omitted.
- ***visibility*** : Enter the character that corresponds to the attributes visibility (+: public, #: protected, -: private, ~: package). This may be omitted.
- ***name*** : Enter the attributes name.
- ***: type*** : Enter the attributes type. This may be omitted.
- ***= initialvalue*** : Enter the attributes default value. This may be omitted.

Note

- Quick Dialog doesn't supports a part of [multiplicity ordered] among attribute notations in UML Specification. Because it has been used part of type as [] symbol to the meaning of array.

Add Button

This adds a new attribute in the next location. Hitting **[Ctrl + Enter]** has the same effect. To insert in the current location, hit the **[Ins]** key.

Delete Button

This deletes the attribute. Hitting **[Ctrl + Del]** has the same effect.

Move Up Button 

This moves the current attribute up. Hitting **[Ctrl + Up]** has the same effect. To edit the upper attribute, just hit the **[Up]** key.

Move Down Button 

This moves the current attribute down. Hitting **[Ctrl + Down]** has the same effect. To edit the lower attribute, just hit the **[Down]** key.

Elements Applied

Attribute

Operation Quick Dialog

Operation Quick Dialog is applied only to operation elements. This appears when an operation is double-clicked in a diagram. Hitting the **[Enter]** key or clicking outside the quick dialog applies the changes.

Visibility Button 

Operation visibility can be selected from  Public,  Protected,  Private, and  Package.

Edit Field 

Operation stereotype, visibility, name, parameter, and return type can be entered in the edit field according to the syntax.

Syntax

*<< **stereotype** >> **visibility name(parameters) : returntype***

- << ***stereotype*** >> : Enter the stereotype name. This may be omitted.
- ***visibility*** : Enter the character that corresponds to the operations visibility (+: public, #: protected, -: private, ~: package). This may be omitted.
- ***name*** : Enter the operations name
- ***(parameters)*** : Enter the operations parameters. Parameters follow the syntax of *direction name : type* and the parameters are separated by comma (,). Parameter relay direction is indicated by direction; it can be in, inout, or out. Parameter name is indicated by name, and parameter type is indicated by type. This may be omitted.
- ***: returntype*** : Enter the operations return type. This may be omitted.

Add Button 

This adds a new operation in the next location. Hitting **[Ctrl + Enter]** has the

same effect. To insert in the current location, hit the **[Ins]** key.

Delete Button

This deletes the operation. Hitting **[Ctrl + Del]** has the same effect.

Move Up Button 

This moves the current operation up. Hitting **[Ctrl + Up]** has the same effect. To edit the upper operation, just hit the **[Up]** key.

Move Down Button 

This moves the current operation down. Hitting **[Ctrl + Down]** has the same effect. To edit the lower operation, just hit the **[Down]** key.

Elements Applied

Operation

AssociationEnd Quick Dialog

AssociationEnd Quick Dialog is applied only to AssociationEnd elements. This appears when an association is double-clicked at the end in a diagram. Hitting the **[Enter]** key or clicking outside the quick dialog applies the changes.

Aggregation Button

AssociationEnd aggregation can be selected from Association, Aggregation, and Composition Navigability can be configured by checking.

Visibility Button

AssociationEnd visibility can be selected from Public, Protected, Private, and Package.

Edit Field

AssociationEnd name, visibility and stereotype can be entered in the edit field according to the syntax.

Syntax

<< ***stereotype*** >> ***visibility name***

- << ***stereotype*** >> : Enter the stereotype name. This may be omitted.
- ***visibility*** : Enter the character that corresponds to the AssociationEnds visibility (+: public, #: protected, -: private, ~: package). This may be omitted.
- ***name*** : Enter the AssociationEnds name.

Multiplicity Combo

AssociationEnds multiplicity can be selected from 0..1, 1, 0..*, 1..*, and * or entered directly.

Elements Applied

AssociationEnd, LinkEnd, AssociationEndRole

Object Quick Dialog

Object Quick Dialog is applied only to object elements. This appears when an object is double-clicked in a diagram. Hitting the **[Enter]** key or clicking outside the quick dialog applies the changes.

Visibility Button 

Object visibility can be selected from  Public,  Protected,  Private, and  Package.

Edit Field 

Object stereotype, visibility, name and type can be entered in the edit field according to the syntax.

Syntax

<< ***stereotype*** >> ***visibility name : type***

- << ***stereotype*** >> : Enter the stereotype name. This may be omitted.
- ***visibility*** : Enter the character that corresponds to the objects visibility (+: public, #: protected, -: private, ~: package). This may be omitted.
- ***name*** : Enter the objects name.
- ***: type*** : Enter the objects type name. This has to be one of the classifiers defined in the current project. This may be omitted.

Create New Class Element Button 

This creates a new class element in the parent namespace of the CollaborationInstanceSet where the object element belongs, and references the new class element in the objects classifier attribute.

Elements Applied

Object

ClassifierRole Quick Dialog

ClassifierRole Quick Dialog is applied only to ClassifierRole elements. This appears when a Classifier is double-clicked in a diagram. Hitting the **[Enter]** key or clicking outside the quick dialog applies the changes.

Visibility Button

ClassifierRole visibility can be selected from Public, Protected, Private, and Package.

Edit Field

ClassifierRole name, visibility and stereotype can be entered in the edit field according to the syntax.

Syntax

<< ***stereotype*** >> ***visibility name : type***

- << ***stereotype*** >> : Enter the stereotype name. This may be omitted.
- ***visibility*** : Enter the character that corresponds to the ClassifierRoles visibility (+: public, #: protected, -: private, ~: package). This may be omitted.
- ***name*** : Enter the ClassifierRoles name.
- ***: type*** : Enter the ClassifierRoles type name. This has to be one of the classifiers defined in the current project. This may be omitted.

Create New Class Element Button

This creates a new class element in the parent namespace of the collaboration where the ClassifierRole element belongs, and references the new class element in the ClassifierRoles base attribute.

Elements Applied

ClassifierRole

Message/Stimulus Quick Dialog

Message/Stimulus Quick Dialog is applied only to message and stimulus elements. This appears when a message or a stimulus is double-clicked in a diagram. Hitting the **[Enter]** key or clicking outside the quick dialog applies the changes.

 (for sequence diagram)

 (for collaboration diagram)

Connect Element Button

This connects specific elements according to the message or stimulus type. If the message or stimulus has CallAction, one of the operations of the object on the other end can be selected. If it is a CreateAction, it can connect a Classifier. If it is a SendAction, it can connect a Signal element.

Visibility Button

Message or stimulus visibility can be selected from Public, Protected, Private, and Package.

Edit Field

Message or stimulus name, visibility and stereotype can be entered in the edit field according to the syntax.

Syntax

<< stereotype >> *[iteration] [condition] return := messagename (arguments)

- ***<< stereotype >>*** : Enter the stereotype name. This may be omitted.
- ****[iteration]*** : Enter the message or stimuluss iteration. This can be in the format of *[i=1..100]. This may be omitted.
-

[condition] : Enter the message or stimuluss condition. This may be omitted.

- ***return :=*** : Enter the expression for the message or stimuluss result. This may be omitted.
- ***messagename*** : Enter the message or stimuluss name.
- ***(arguments)*** : Enter the expression for the arguments passed to the message or stimulus. This may be omitted.

Sequence number (for Collaboration Diagram) 

The sequence number, which indicates the execution order of the message or stimulus, can be changed.

Create New Operation Button 

If the message or stimulus has a CallAction, this button creates a new operation in the other object, and references the new operation in the CallActions operation attribute.

Elements Applied

Message, Stimulus

State Quick Dialog

State Quick Dialog is applied only to state elements (CompositeState and SubmachineState). Hitting the **[Enter]** key or clicking outside the quick dialog applies the changes.

Visibility Button 

State visibility can be selected from  Public,  Protected,  Private, and  Package.

Edit Field 

State name, visibility and stereotype can be entered in the edit field according to the syntax.

Syntax

<< ***stereotype*** >> ***visibility name***

- << ***stereotype*** >> : Enter the stereotype name. This may be omitted.
- ***visibility*** : Enter the character that corresponds to the states visibility (+: public, #: protected, -: private, ~: package). This may be omitted.
- ***name*** : Enter the states name.

Add EntryAction Button 

Creates and adds a new EntryAction.

Add DoAction Button 

Creates and adds a new EntryAction.

Add ExitAction Button 

Creates and adds a new ExitAction.

Elements Applied

CompositeState, SubmachineState

Action Quick Dialog

Action Quick Dialog is applied only to action elements (EntryAction, DoAction, and ExitAction). Hitting the **[Enter]** key or clicking outside the quick dialog applies the changes.

Visibility Button

Action visibility can be selected from Public, Protected, Private, and Package.

Edit Field

Action name, visibility and stereotype can be entered in the edit field according to the syntax.

Syntax

<< ***stereotype*** >> ***visibility name***

- << ***stereotype*** >> : Enter the stereotype name. This may be omitted.
- ***visibility*** : Enter the character that corresponds to the actions visibility (+: public, #: protected, -: private, ~: package). This may be omitted.
- ***name*** : Enter the actions name.

Add Button

This adds a new action in the next location. Hitting **[Ctrl + Enter]** has the same effect. To insert in the current location, hit the **[Ins]** key.

Delete Button

This deletes the action. Hitting **[Ctrl + Del]** has the same effect.

Move Up Button

This moves the current action up. Hitting **[Ctrl + Up]** has the same effect. To edit the upper action, just hit the **[Up]** key.

Move Down Button 

This moves the current action down. Hitting **[Ctrl + Down]** has the same effect. To edit the lower action, just hit the **[Down]** key.

Elements Applied

UninterpretedAction(EntryAction, DoAction, ExitAction)

Note/Text Quick Dialog

Note/Text Quick Dialog is applied only to note elements and text elements. This appears when a note or text element is double-clicked in a diagram. Hitting **[Ctrl + Enter]** or clicking outside the quick dialog applies the changes.

Edit Field

The edit field can contain any contents.

Elements Applied

Note, Text