

# UseCase Diagrams

---

The following elements are available in a usecase diagram.


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

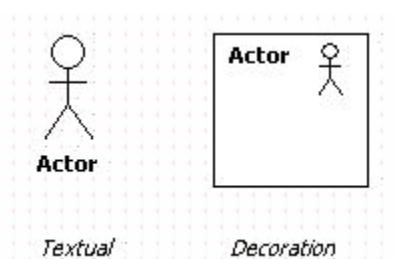
## Actor

### 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

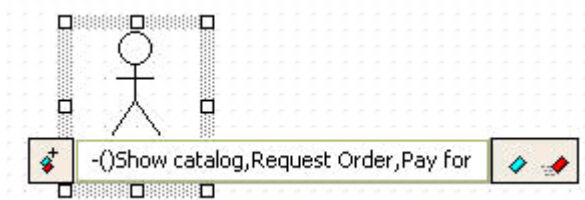
In order to create Actor, click **[Toolbox] -> [UseCase] -> [Actor]** button and click the position where to place Actor. Actor is shown in the form of stick man or rectangle with icon, that is decoration view. To display actor in decoration view, select **[Format] -> [Stereotype Display] -> [Decoration]**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



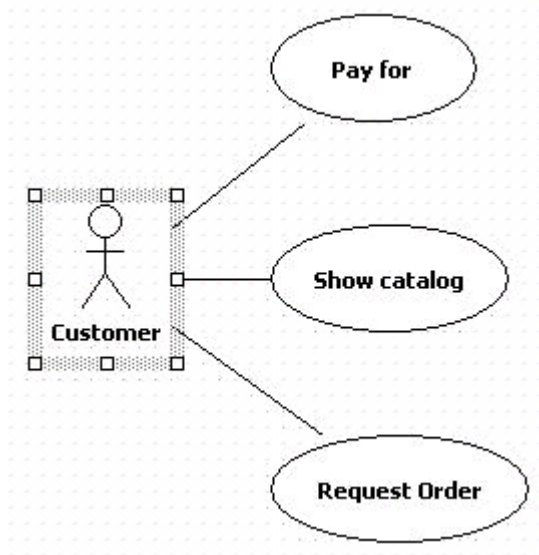
### Procedure for creating multiple UseCases used by Actor at once

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.




## UseCase

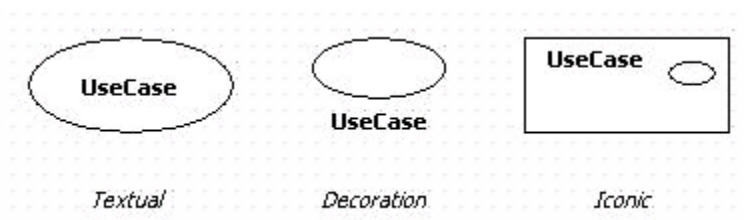
### Semantics

The use case construct is used to define the behavior of a system or other semantic entity without revealing the entity's 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

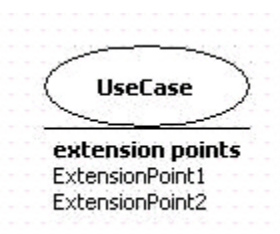
In order to create UseCase, click **[Toolbox]** -> **[UseCase]** button and click the position where to place UseCase on the **[main window]**.

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.

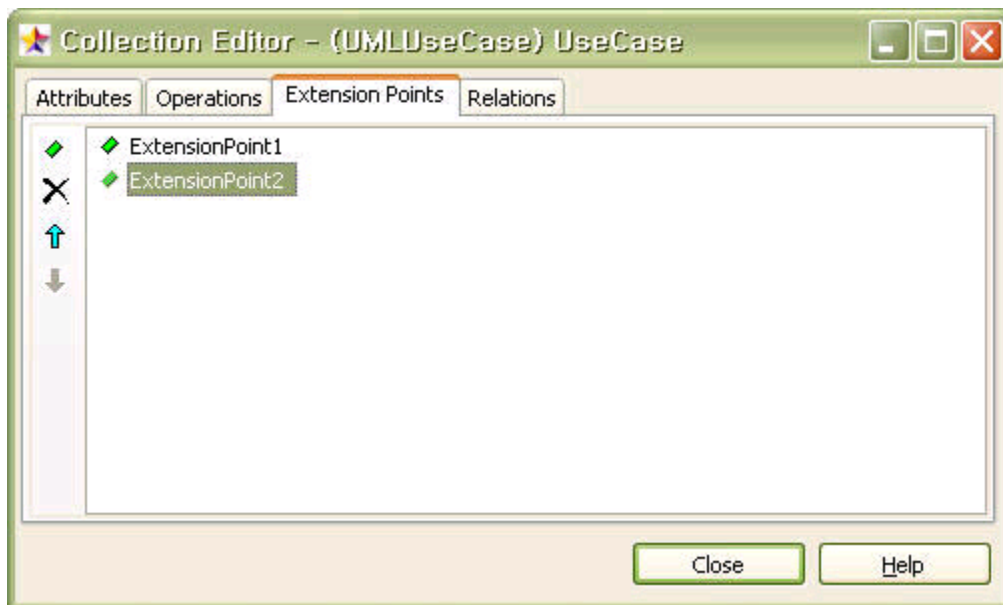


### Procedure for adding Extension

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

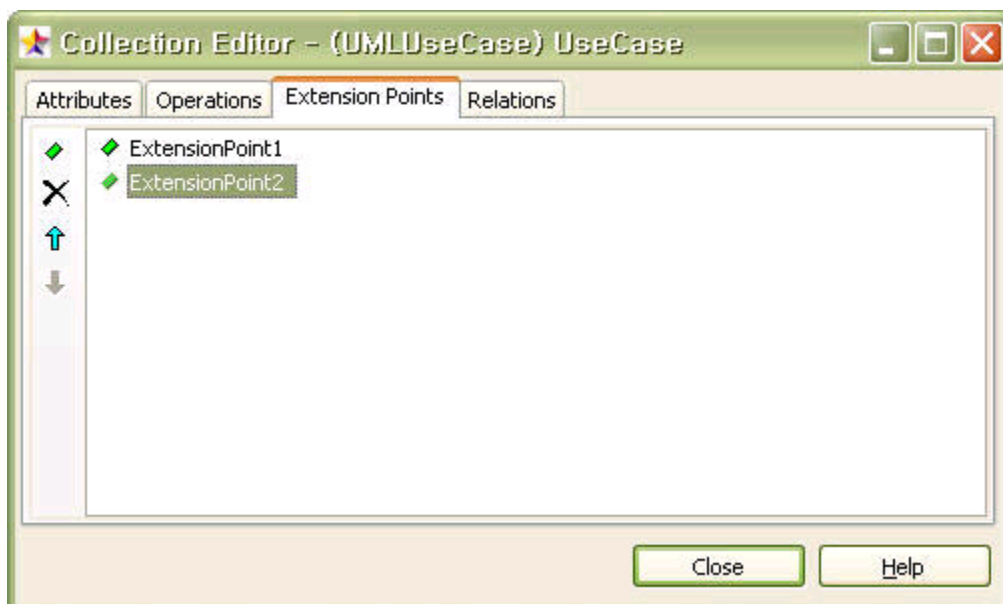


To edit ExtensionPoints of UseCase, click UseCase's **[Collection Editor...]** popup menu or click  button of **[ExtensionPoints]** collection property.



### Procedure for entering UseCase specification

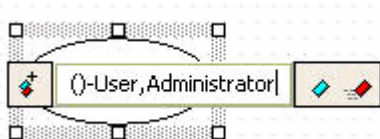
To enter basic flow, alternative flow properties of usecase, select **[Tagged Values...]** popup menu or click **[Ctrl+F7]** button. At tagged value editor, select **[UseCaseSpecification]** item and enter the properties.



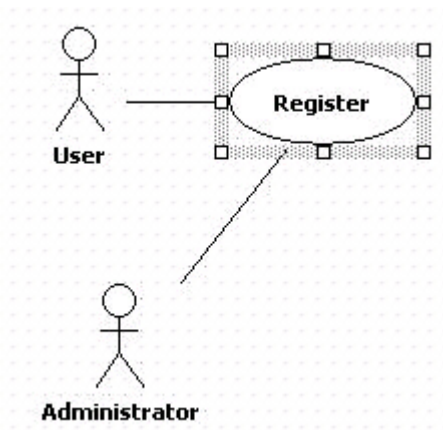
### Procedure for creating Actor from UseCase

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

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



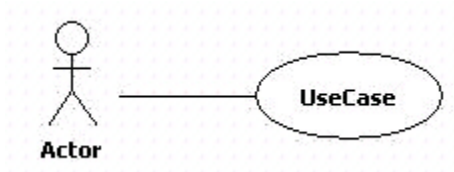
## Association / Directed Association

### Semantics

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

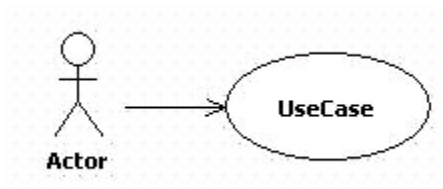
### Procedure for creating association

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

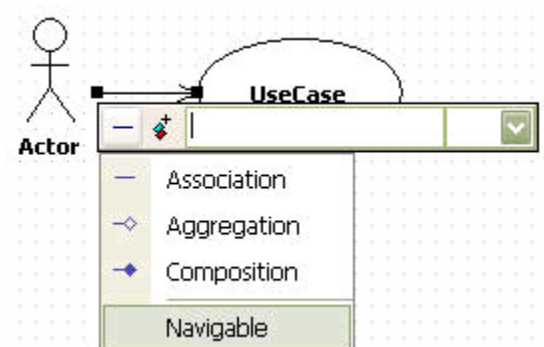


### Procedure for creating directed association

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



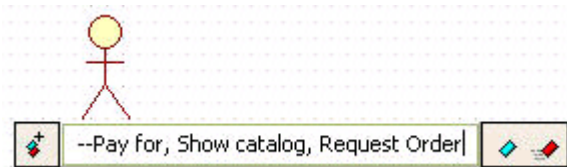
Or create association, click the actor-side association end. At the quick dialog, uncheck navigable and association becomes directed.



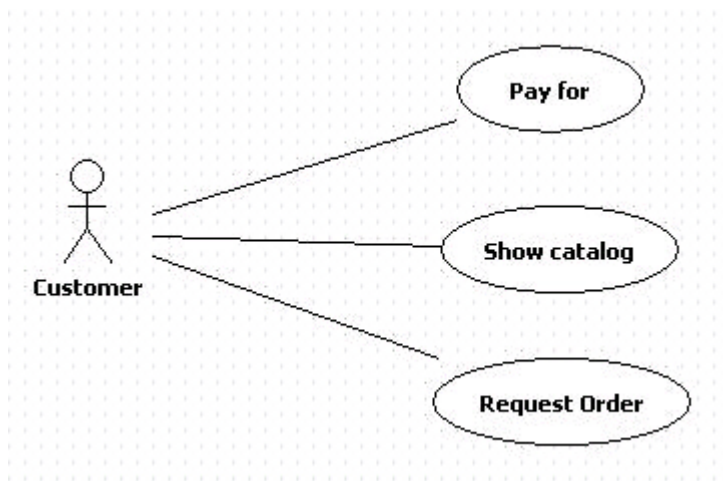
### Procedure for creating element related to association/directed association

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.



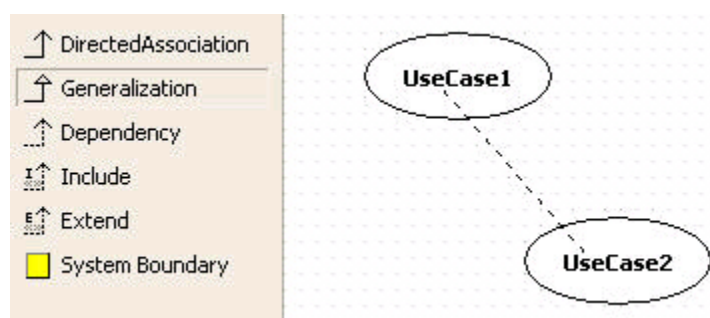
## Generalization

### 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]**.

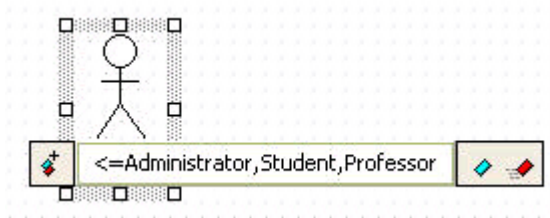


### Procedure for creating multiple child actors inherited from actor

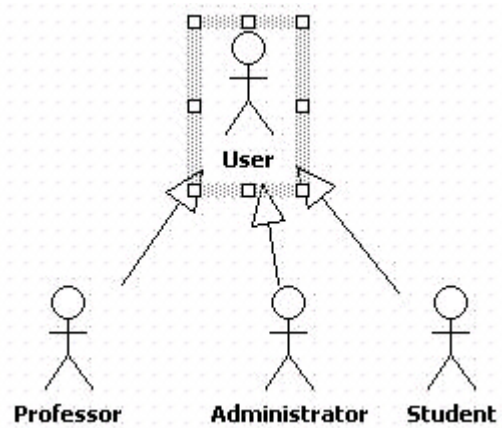
To create multiple elements inherited from some element,

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

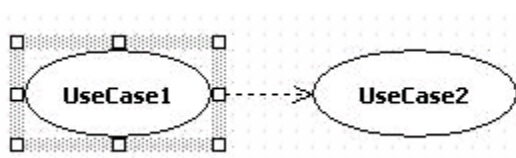
In order to create dependency, click **[Toolbox] -> [UseCase] -> [Dependency]** button, drag element and drop to other element depended.

### 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.



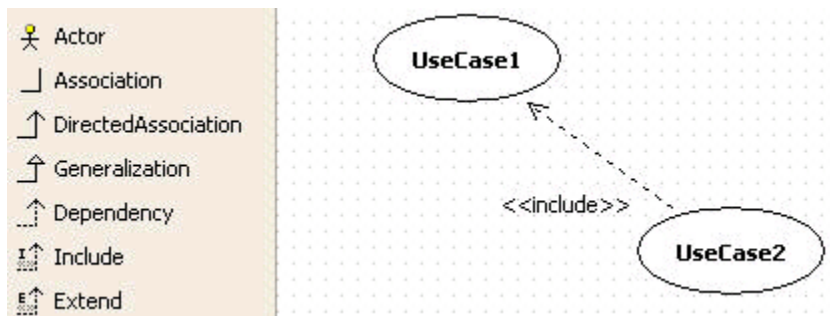
## Include

## Semantics

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

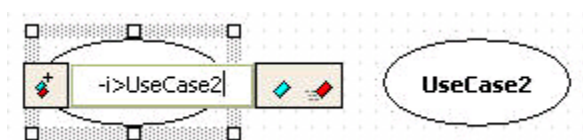
### Procedure for creating include

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

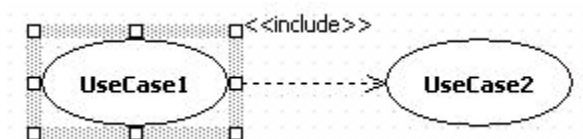


### Procedure for creating other usecase included by current usecase

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



So include relationship is created between two elements.



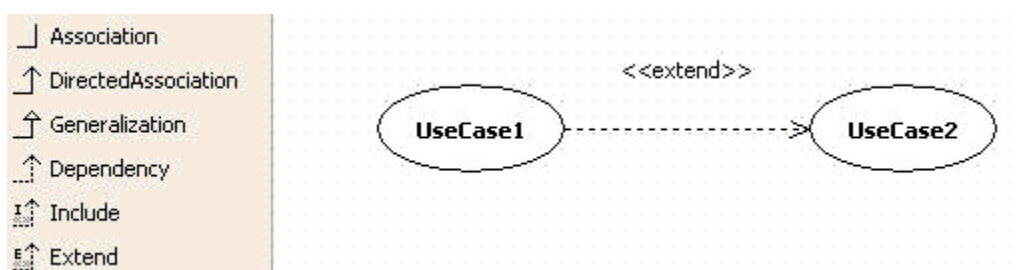
## Extend

### 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

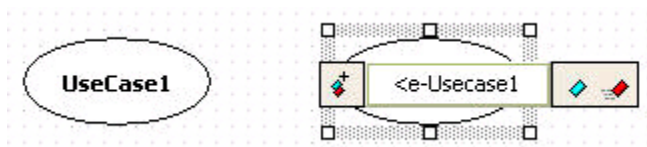
In order to create extend, click **[Toolbox] -> [UseCase] -> [Extend]** button, drag from element extending and drop to element extended in the **[main window]**.



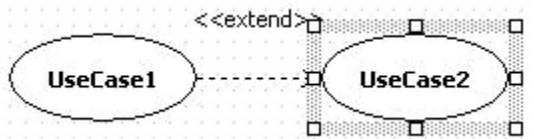
### Procedure for creating other usecase extending current usecase



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



So extend relationship is created between two elements.



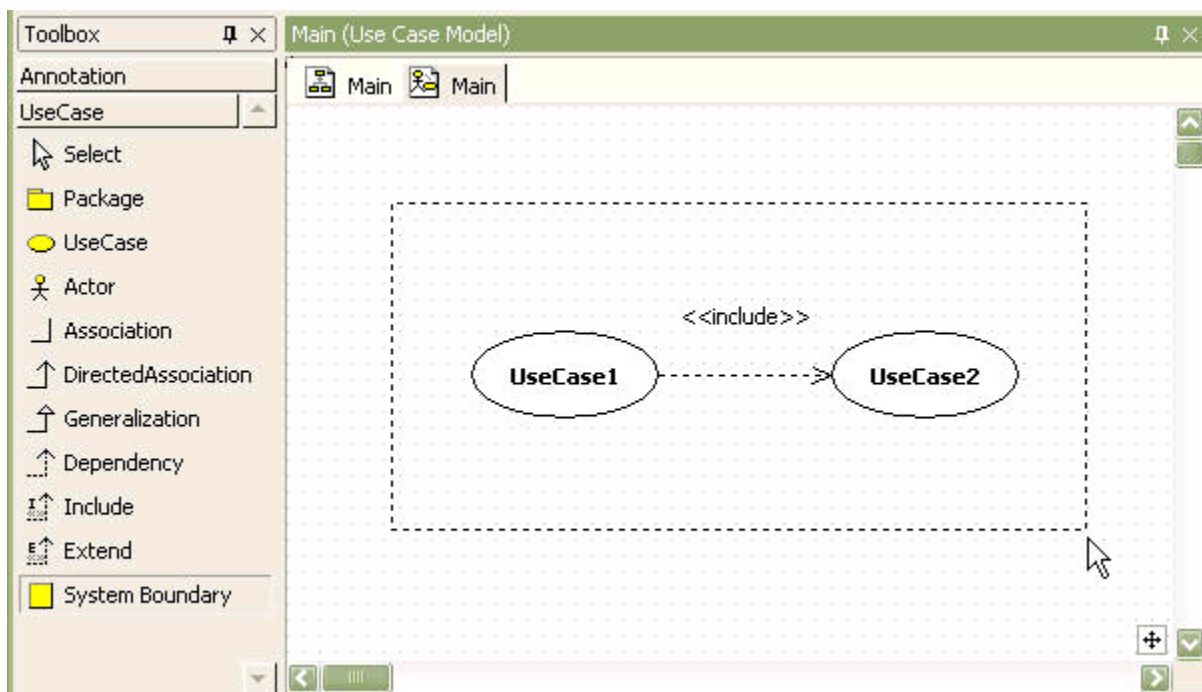
## System Boundary

### 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]**.

