VOCLEditor Users Guide

Gruppe 4

Sommersemester 2005

VILA

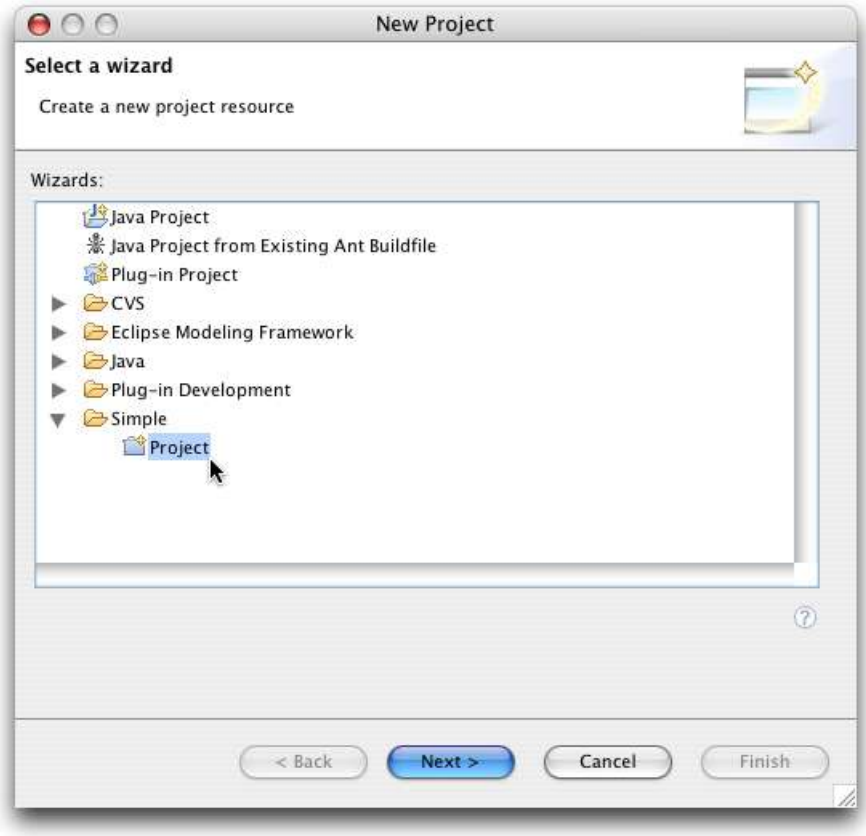
Table of contents

1. Introduction

This is the users guide for the VOCLEditor developed by Gruppe 4 for the Visuelle Sprachen Projekt held in 2005 at the Technical University Berlin.
Assuming that you are familiar with OCL and the VOCL mapping to graphical elements, this guide explains how to use the editor to create VOCL diagrams and convert them to OCL expressions.

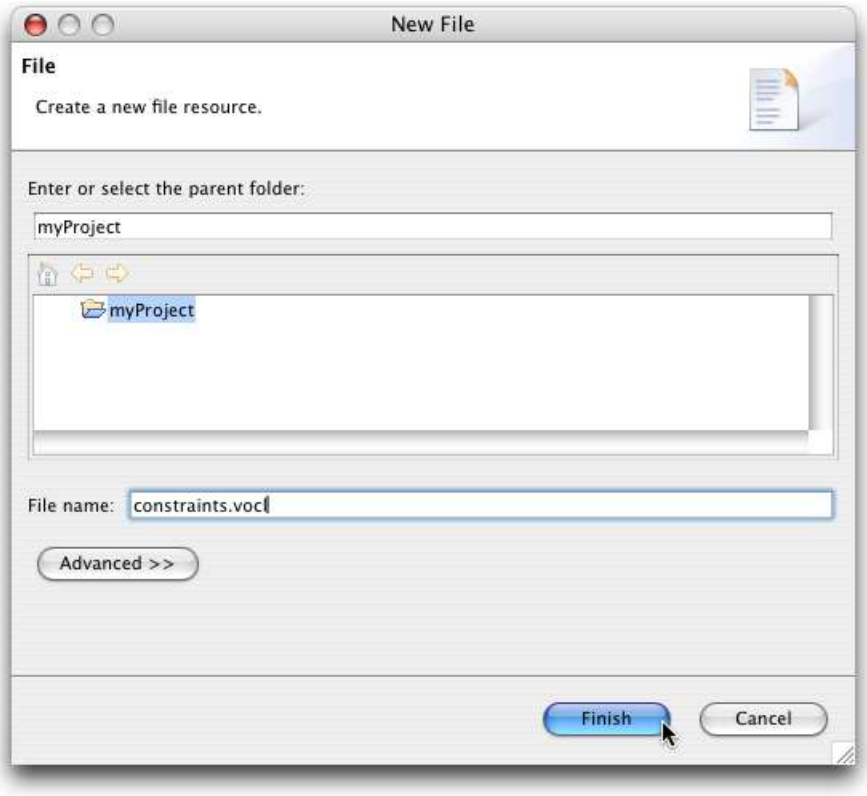
2. Creating a VOCL diagram file

This chapter explains how to create a VOCL diagram file. Since you work inside of the eclipse workbench you first have to create a project to hold the file. A new project is created by selecting *File -> New -> Project* in the eclipse menu or by selecting *New -> Project* in the context menu of the Navigator or Package Explorer view. In the New Project wizzard now appearing select *Simple -> Project*. Enter a name for your new project and close the wizzard.



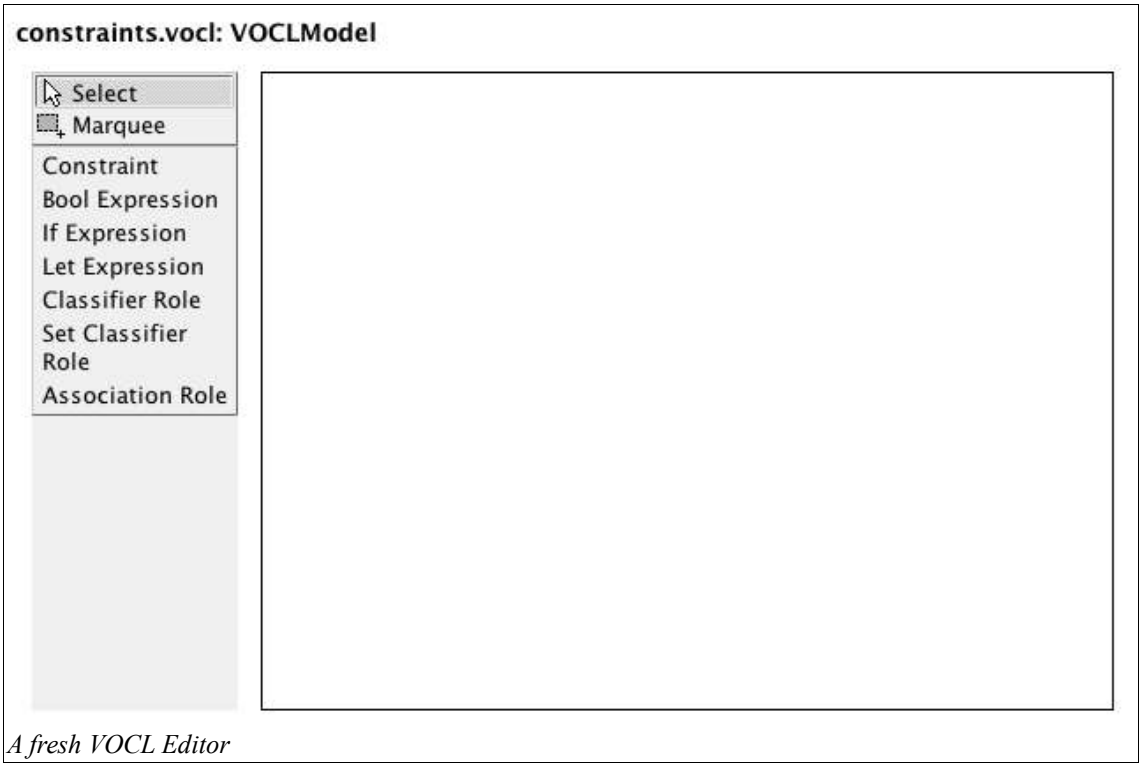
The eclipse new project wizzard

Now you can create a new VOCL diagram, by creating a new file with .vocl as its extension. To do this select your project and open the *New File* wizzard by selecting *File -> New -> File* from the main menu or *New -> File* from the context menu of the Navigator or Package Explorer view. Give your new file a name ending in .vocl and press *Finish*.



The eclipse new file wizzard

You should see a new editor looking like this:



A fresh VOCL Editor

3. Basic editor features

The VOCL Editor allows you to create a structured VOCL diagram by visually assembling VOCL model elements. Theses model elements are:

- constraints with a context
- boolean expressions
- if expressions
- let expressions
- classifier roles
- set classifier roles
- associations
- classifier methods
- classifier attributes
- conditions

The Palette

Most of the model elements are created by selecting an entry from the palette and placing the model element to be created on an appropriate parent element. The mouse cursor indicates whether the element can be placed at a certain position.

Other elements are created from the context menu of already existing model elements. For most elements you need to specify some properties in the properties dialog that appears before the element is actually created.

The topmost entry in the palette is the select tool. It is used to select model elements you wish to modify in some way. After selecting a model element you see a border (possibly with resize handles) around it or it becomes highlighted with a different color.

Some model elements can be moved or resized after being selected. To move an element just drag it to a new place within its parent element. Risizing is done by clicking on one of the resize handles and dragging it to another position within its parent element.

You can delete a selected model element by hitting the Delete button on your keyboard or selecting "Delete" from the elements context menu.

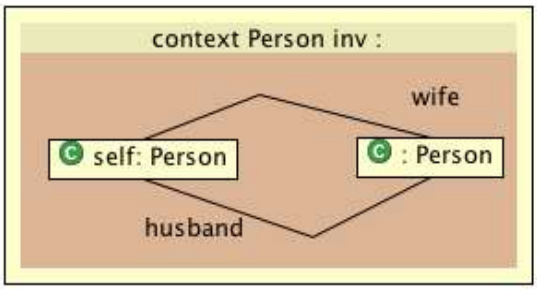
To open an elements properties dialog again, you can choose the "Properties Dialog" entry from the context menu of the selected model element or simply double-cklick on the model element you wish to modify.

Every modification of you VOCL model (like creating, deleting, changing properties or position) can be undone and redone using the Undo and Redo buttons on the eclipse toolbar or by hitting the undo und redo hotkeys (CTRL-Z and CTRL-Y on most systems, Apple key instead of CTRL on MacOSX).

4. Working with Model Elements

4.1. Constraint

A constraint is the first model element you need to have. It serves as a container for VOCL expressions and has a context label. A constraint has a name which can be specified in the context's properties dialog, it doesn't have an own properties dialog.



A simple romantic constraint with two associated classifier roles

Inside of the constraint box you can create:

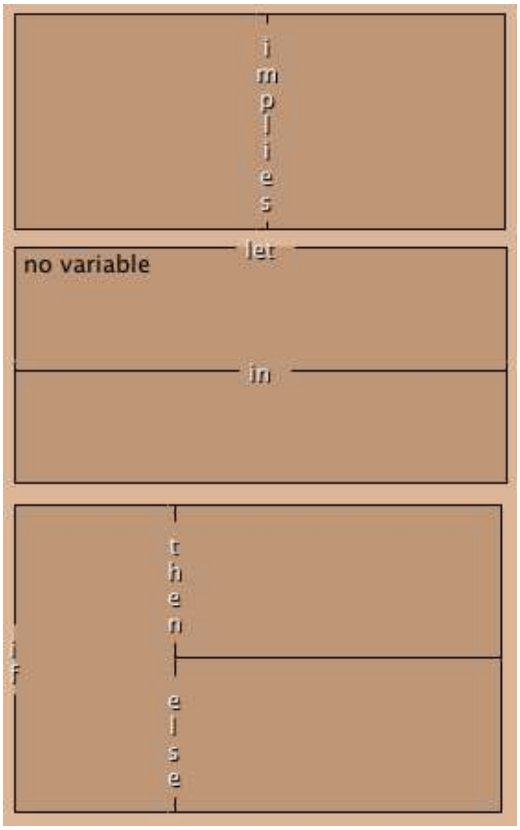
- navigation expressions consisting of classifier roles (possibly with a method and attributes) set classifier roles and associations
- sub expression (boolean-, if- and let-expressions)
- a condition

All the expressions directly contained in a constraint are combined with a logical 'and' in the resulting OCL string.

4.2. Sub Expressions

There are three kinds of sub-expressions in the VOCL model:

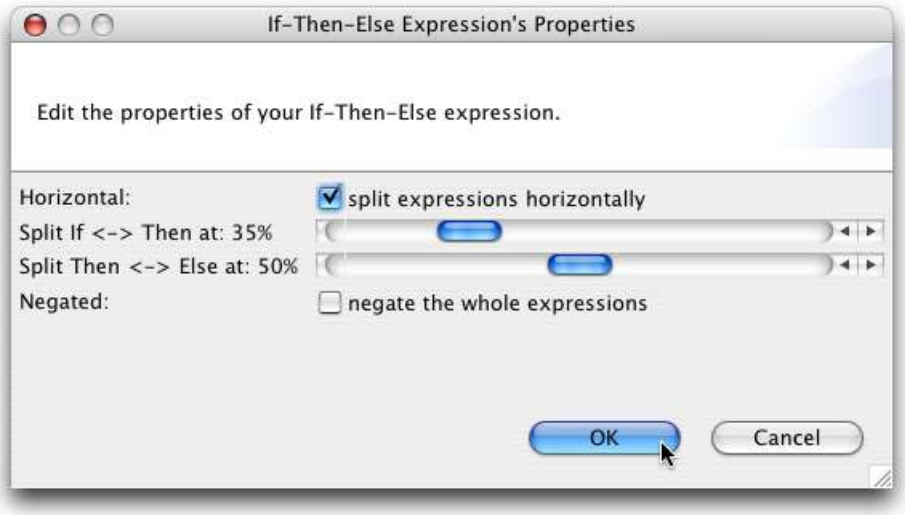
- boolean expressions with two parts containing expressions that are logically combined with the selected boolean operation
- let expressions with a 'let' part and an 'in' part
- if expressions with their three parts: an 'if' a 'then' and an 'else' part



Empty sub expressions

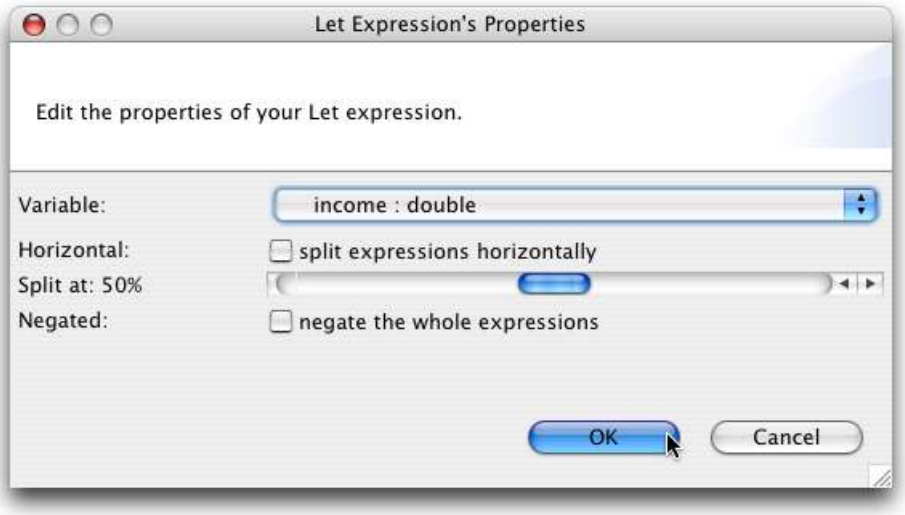
These sub expressions are created from the palette and placed on a constraint or an already existing sub expression. Just like the root model element, the constraint, each part of a sub expression can host classifiers, associations and, again, sub expressions.

There are dialogs for if-, let- and boolean expressions showing up when you create the exression. In each of these dialog you can choose whether the expression box should be split horizontally or vertically and at which point to split the wholebox. It is also possible to negate the expression in the dialogs. A negated expression appears crossed out by red lines.



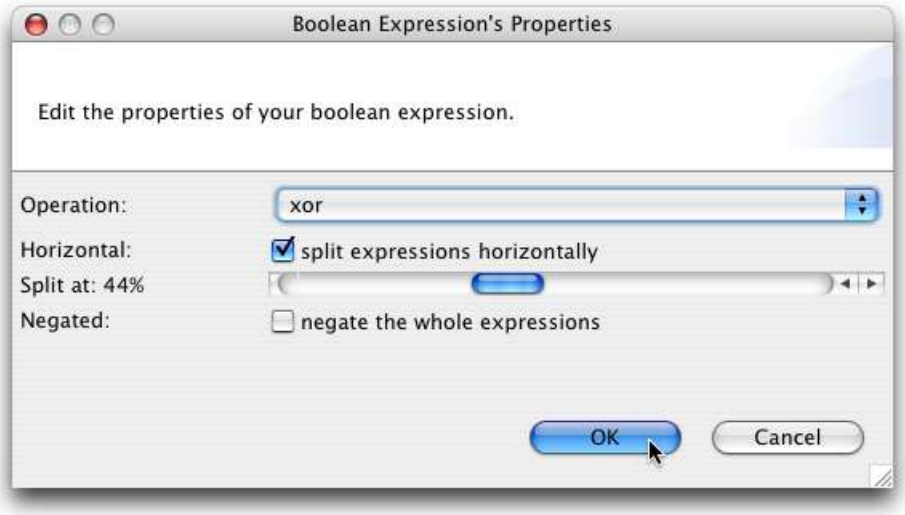
The if expression properties dialog

The if expression dialog additionally allows you to set where the then and else parts of the expression should be split.



The let expression properties dialog

The let expression dialog provides a drop down list of variables defined somewhere in the let part of the expression. Use that list to select the variable being defined in that let expression.

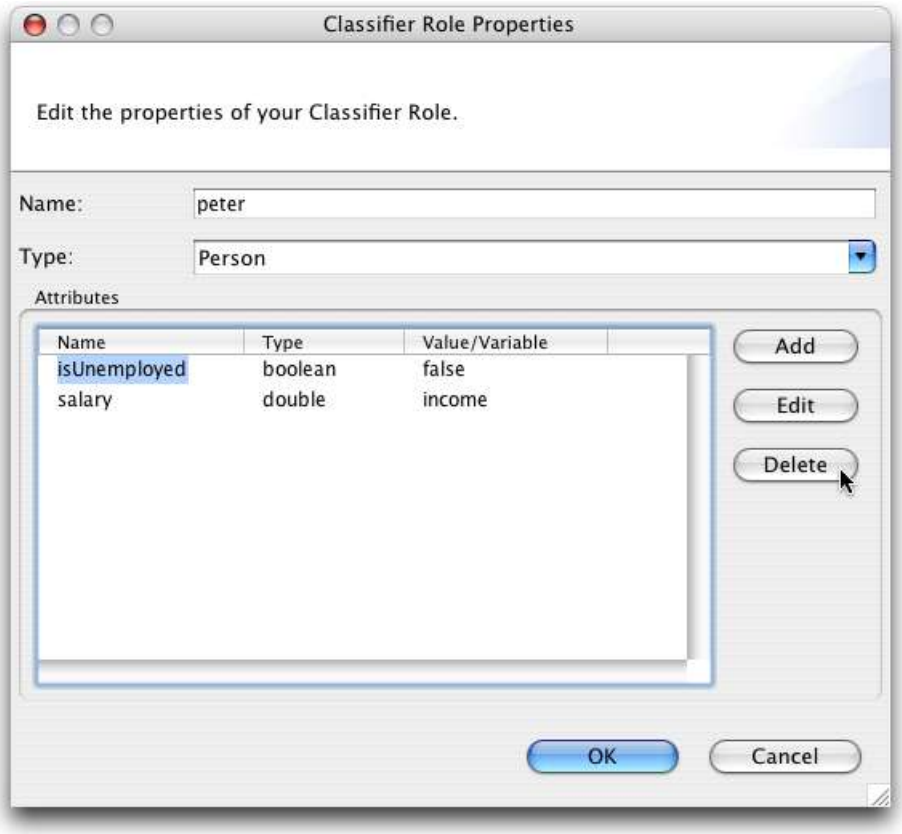


The boolean expression properties dialog

The boolean operation of a boolean expression is chosen from a list in the boolean expression's dialog. Available operations are: or, xor, implies

4.3. Classifier Roles

A classifier role models an instanciated object of some class. Classifier roles are created from the palette and placed on a constrain or in a sub expression part. After placing a classifier role on its parent, a dialog appears. Here you can set the optional classifier name and its mandatory type. The classifier role dialog also allows you to manage a classifier roles attributes.



The classifier role properties dialog

4.3.1. Attributes

A classifier role (as well as set classifier roles) can have attributes just as an object can have member variables. You can add attributes by selecting the "Add Attribute" context menu entry of the classifier role or by using the classifier role's properties dialog. Delete attributes by selecting them an choosing "Delete" from their context menu or pressing the delete key. Attributes can also be deleted from their classifier rols's properties dialog.

Adding an attribute pops up the attribute properties dialog, which is also used to modify the attributes properties later. Within this dialog the attribute's name and type are specified. An attribute has an assigned value or it is bound to a variable. You can choose between value and variable by toggling one of the check boxes and then enter the value or variable name in the appropriate text field.

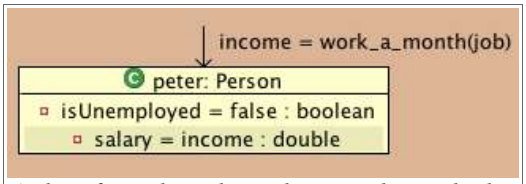
4.3.2. Methods

Classifier roles and set classifier roles might have a method attached to them. To create a method select "Add Method" from the classifier role's context menu.

A method must have a name and can optionally have a return variable with it's return type as well as a number of parameter variables.

The method properties Dialog

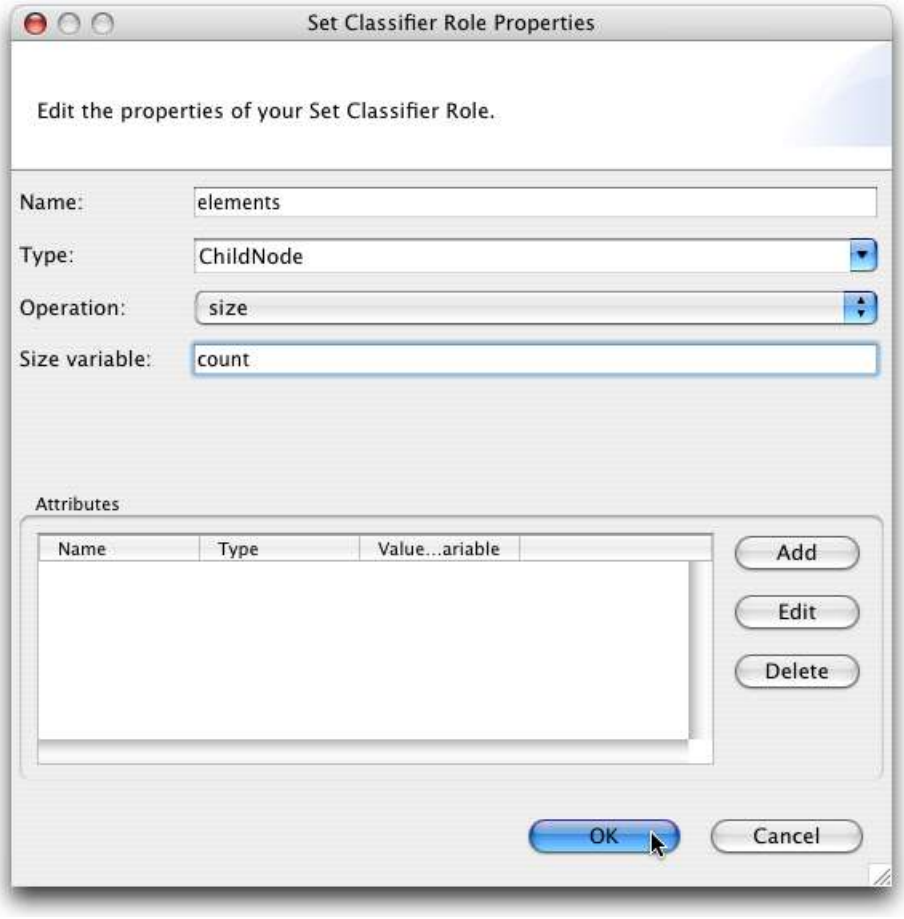
These properties are set in the method properties dialog that pops up when you create a method or if you choose to open it by double clicking or using the context menu.



A classifier role with attributes and a method

4.4. Set Classifier Roles

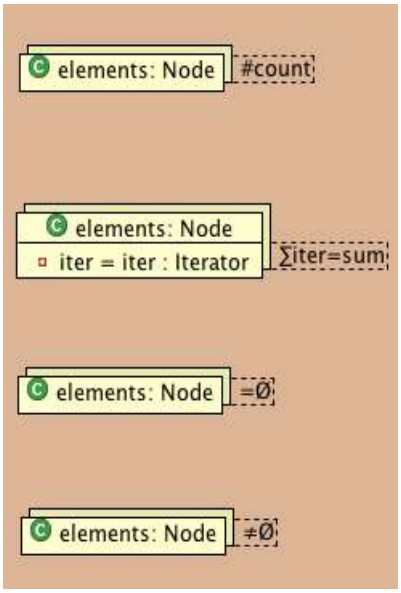
Set classifier roles represent a set of objects of the same kind. They are very similar to normal classifier roles, they can have attributes and a method. Additionally you can specify an operation to be performed on the set.



The set classifier role properties dialog

Available operations on sets are:

- size: with this operation you can make assumptions about the size of the set. You have to name a variable in the properties dialog. This variable represents the number of objects in the set.
- sum: the sum of all objects in the set. This operation requires you to specify an iterator variable (which must be a variable bound to an attribute of that set classifier role) and a sum variable in the properties dialog.
- isEmpty: states that the set is supposed to be empty
- notEmpty: states that the set is supposed not to be empty



Different set operations

4.5. Association Roles

Association roles connect two (set) classifier roles to construct a navigation expression. After selecting the palette entry labeled "Association role", you first have to click on the classifier role you wish the association to start from. Then you have to click on another classifier role, where the association should end. Both classifier roles have to be directly contained in the constraint or in one part of a sub expression.

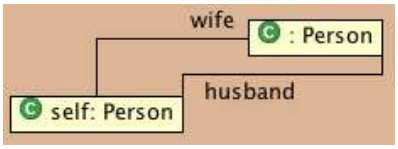
After placing an association role the association role properties dialog appears. Here you can choose which of the connected classifier roles the association actually navigates to. You have to specifiy a name for the associations end classifier.

The association role properties dialog

The graphical representation of an association role is made up of a line connecting the two classifier roles and a label showing the name of the association end role. In the

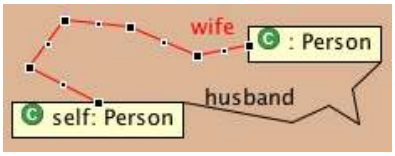
properties dialog you can select one of two ways of routing the association line from one classifier role to the other:

- Manhattan style routing: automatically lays out the connection with an orthogonal route between the start and end classifier roles



Manhattan style connection routing

- Bendpoint routing: enables you to manually create and move bend points that break the line into segments



Bendpoint connection routing

The label showing the name of the association end role can be moved to any position you like (though it's a good idea to place it somewhere near the end classifier role).

4.6. The Context

The context specifies what your OCL constraint is about. In the VOCL editor the context is label placed on top of the constraint. When you create a new constraint the context is undefined since you have not yet declared any classifiers or methods to constraint.

To set the context after creating model elements, open the context properties dialog by double-clicking the context label or using the context label's context menu.

The context properties dialog

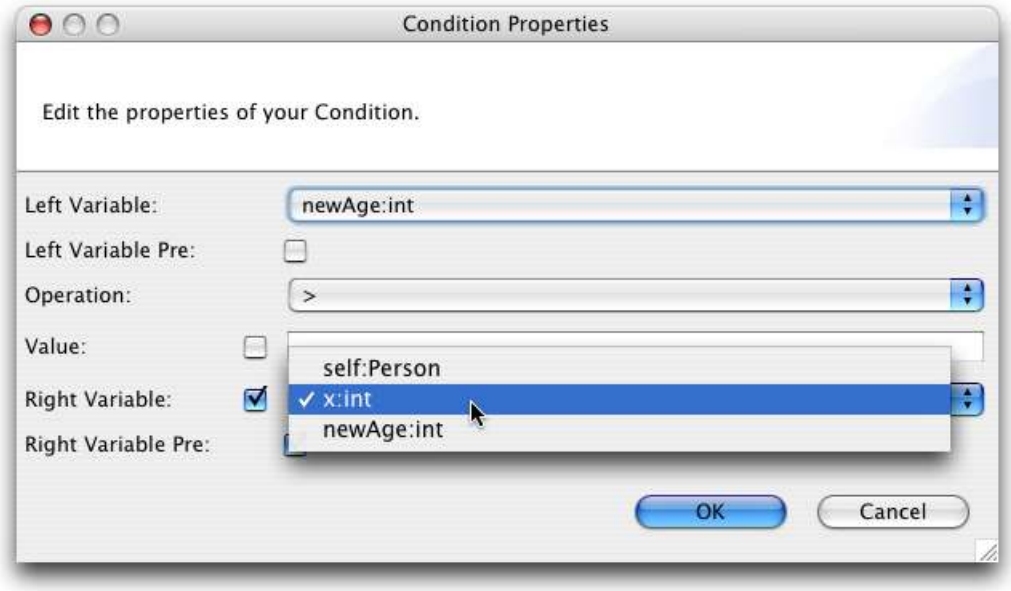
First of all, the context properties dialog allows you to give your constraint a name. More important is to specify the type of the context by toggling one of the three type buttons. It can be an invariant, a pre condition or a post condition. If you choose your constraint to define an invariant you have to select an existing classifier role in the drop down list below the type buttons. Otherwise select a method from that list.

If the method or classifier role used in the context is deleted, the context switches back to its "undefined" state.

4.7. Conditions

The constraint as a whole as well as the different parts of a sub exprerssion can contain an OCL condition. The condition is made up of a variable (the left variable), a comparision operator and a value or another variable to compare to the left variable.

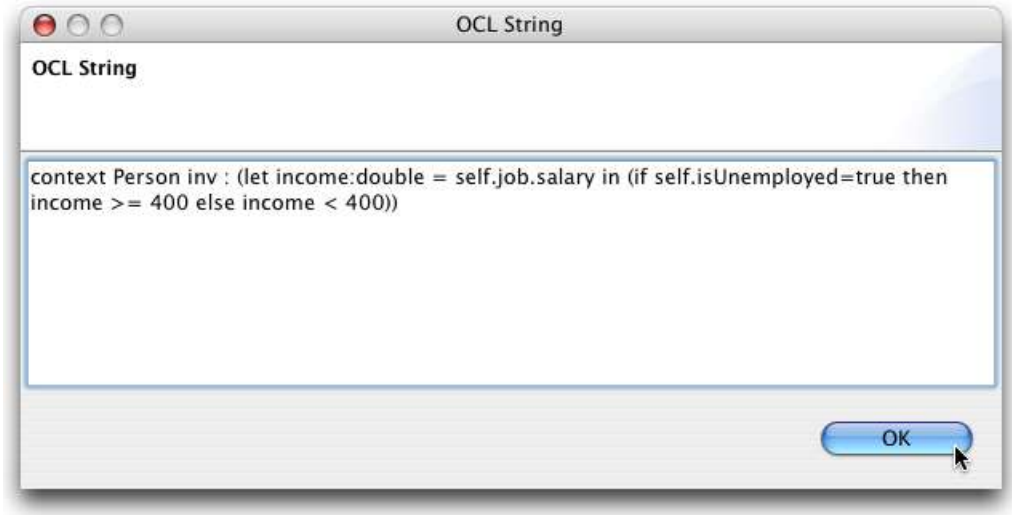
To add a condition to the constraint or an expression part select "Add Condition" from that model element's context menu. The condition properties dialog will pop up. Here you can choose the left variable, the comparision operation and the value or right variable to compare to. To apply the comparision to a variable's value prior to invocation of the context method, enable the "Left Variable Pre" or "Right Variable Pre" checkbox.



The condition properties dialog

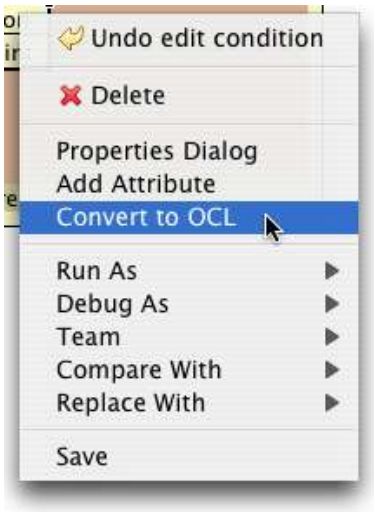
If you delete model elements that hold a variable used in a condition, that condition is deleted. You can, of course, undo the deletion and the condition will be back again.

5. Converting a VOCL Diagram to an OCL Expression



The OCL string dialog

After you modelled you VOCL constraint it's time to generate the textual OCL representation of that constraint. To do so, open the context menu anywhere inside the editor and select the menu entry labelled "Convert to OCL". After a short time of thinking the VOCL editor will show you a message box with a text area holding the OCL string.



Convert action in the conext menu

Have fun using VOCLEditor by Gruppe4.

- end -