



Eclipse JWT – Java Workflow Tooling

Title of this document

Workflow Editor (WE): Installation and Usage Tutorial

Document information

last changes

13.02.2008

component version

0.4.0

Document created by

Florian Lautenbacher, University of Augsburg, DE

Christian Seitz, University of Augsburg, DE

Project information

Java Workflow Tooling project

Project leads:

- ❖ Marc Dutoo (OpenWide, FR)
- ❖ Florian Lautenbacher (University of Augsburg, DE)

Project web site : <http://www.eclipse.org/jwt/>

Any comments or feedback is welcome on the mailing list, newsgroup or bugzilla !

Content

- 1 Summary 3
- 2 Installation..... 4
 - 2.1 JWT 4
 - 2.2 AgilPro 4
- 3 Creating the first process..... 5
- 4 Adding packages and roles..... 8
- 5 Adding applications and data 10
- 6 Making our process executable 14
- 7 Simulating / Executing our process..... 18
- 8 Different views in the modeling tool..... 21
- 9 Export workflow templates 22
- 10 More complicated example 23
- 11 Subprocesses 27
- 12 Additional applications 28
- 13 Other features 30

1 Summary

This document shows the installation and usage of the JWT Workflow Editor (WE) process modeling tool in version 0.4.0. In addition we show how to execute (simulate or preview) a process model with the AgilPro Simulator. AgilPro is a partner project that is currently not available on Eclipse, but only on SourceForge. We aim to include many of the tools developed in AgilPro into Eclipse JWT. AgilPro and its components (such as the AgilPro Simulator which is described more in this document) can be downloaded from <http://sourceforge.net/projects/agilpro>.

We try to cover all aspects of JWT (and AgilPro) that are of interest for the user and reader of this document. If there is another topic that you are interested in that is not covered in this description, please don't hesitate to contact us.

2 Installation

2.1 JWT

JWT can be downloaded from the update site. You need to have at least Eclipse 3.3 installed with Java 6.0 behind. Please select in your Eclipse „Help → Software Updates → Find and install ... → Search for new features to install

We will add more information about the update site as soon as we have finalized the first release review of the workflow editor. We will then be able to specify more details here.

After Eclipse has been restarted, you should be able to create a new workflow model.

If you experience any problems, don't hesitate to contact us on our newsgroup.

2.2 AgilPro

If you want to install AgilPro, you need to have at least Java 6.0 installed (version 1.4.0 of AgilPro won't work with older versions).

The latest version of AgilPro LiMo (which is an RCP variant of WE) and Simulator is available on SourceForge under <http://sf.net/projects/agilpro>. Please find also more information and examples about AgilPro on <http://www.agilpro.eu>.

Please select the version of your operating system on SourceForge (Mac, Linux i386, PPC, i386 Windows, etc.) and start the installer (the .JAR-file) either with double clicking on it or (if this feature has been disabled on your PC) enter

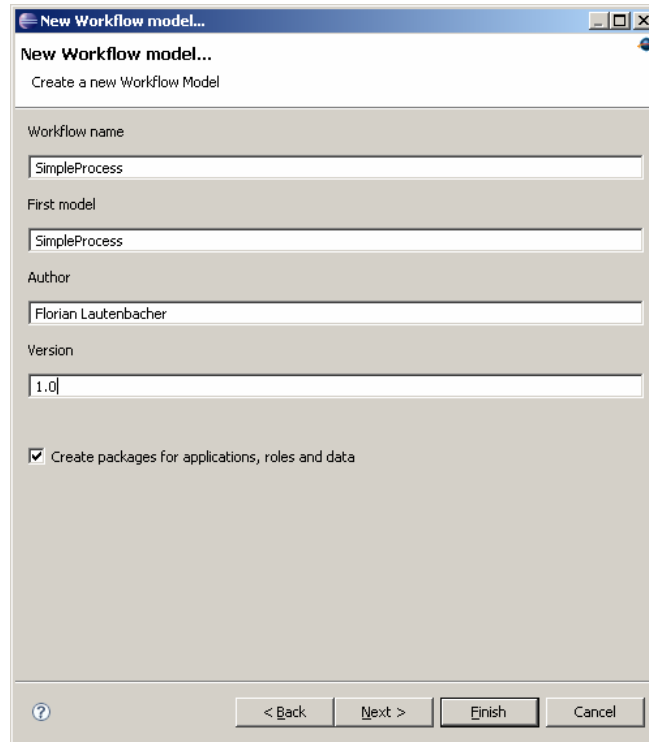
```
java -jar AgilProSimulator-Installer-win32-x86_1.4.0.jar
```

This should start the installer (on a Windows PC) if Java 6.0 has been installed on your PC and can be found in the classpath.

If you experience any problems during the installation or usage of AgilPro, please don't hesitate to contact <mailto:support@agilpro.de>

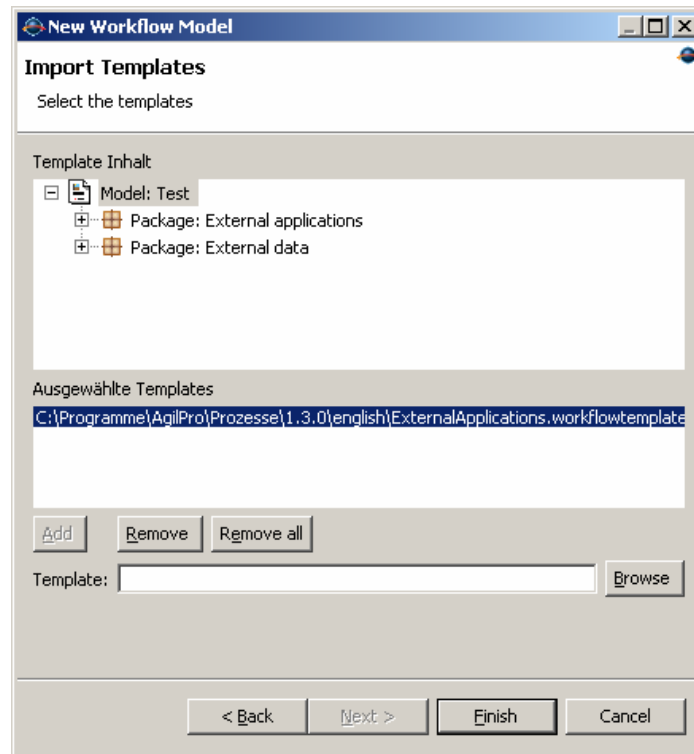
3 Creating the first process

At the beginning we have to create a new process (“File”, “New → Other...”, “JWT Workflow Editor” → “New workflow model”) and specify the name of the main package, of the model, the author and version information as well as the filename.

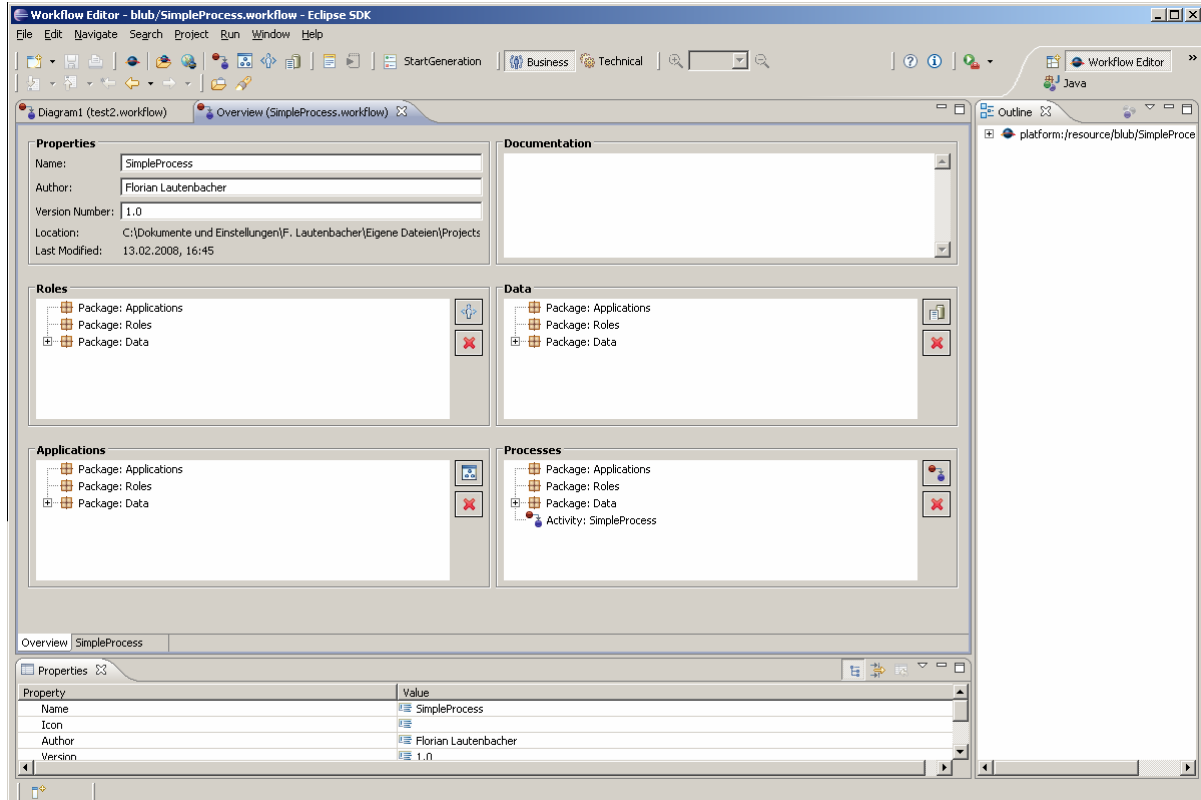


You can also select whether JWT should already create a basic structure of packages for you including one for applications, one for roles and one for data or whether you would like to structure your processes alone.

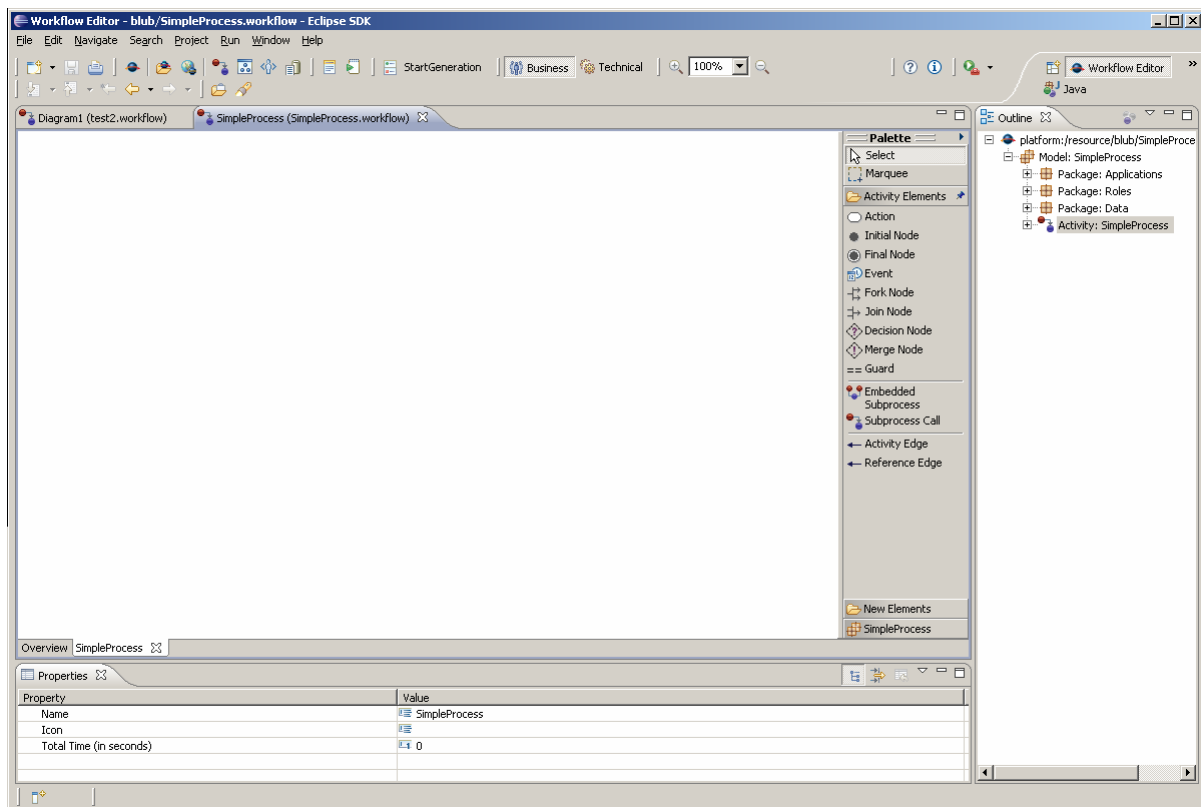
In the next window we can select whether we want to include some external data or applications from existing templates. Existing templates make it much easier for you to reuse information that you have stored once for another process model. In our example we have already specified some external applications and external data which can be reused for this process model. This template is not necessary for the further reading of this document, it shall only describe what *would* be possible with JWT WE.



The overview page which appears when opening the file shows the already added information (such as author, name of the workflow or version number) as well as the name of the current workflow (Diagram1)



To edit the process steps, you need to go to the modeling tab “SimpleProcess”:



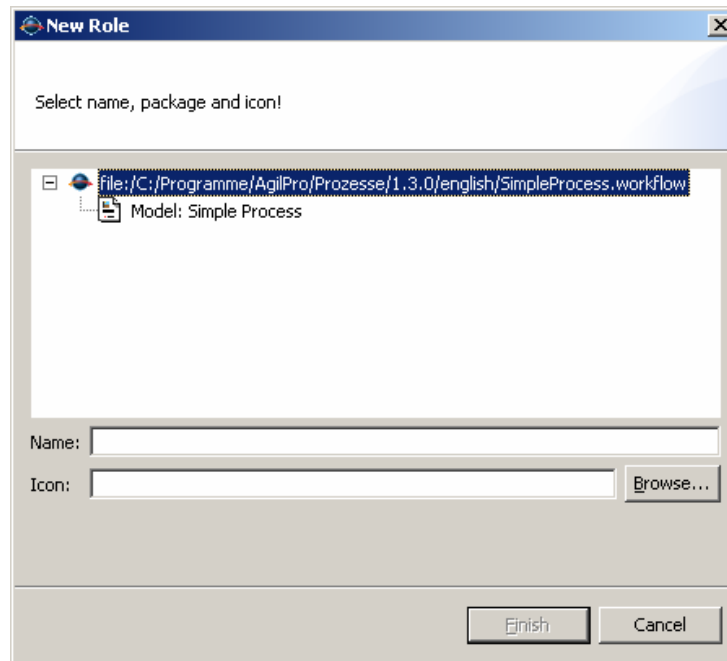
To create your first process, you need an InitialNode at the beginning, a FinalNode at the end and at least one action in the middle. Connect these three elements with ActivityEdges and you'll get something similar to the following figure:



This process is of course not executable or describes anything useful. Therefore, you have to give it a more meaningful name, describe the responsibilities for this action and name the application which should be started and the data that should be opened.

4 Adding packages and roles

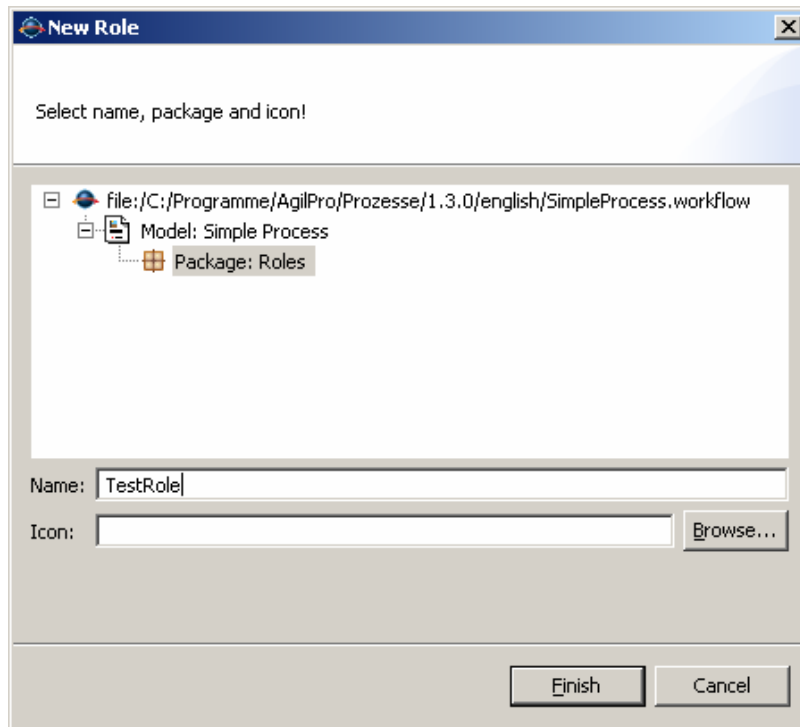
Therefore, first, we will create some packages to order our roles, etc. for the future diagrams. Go to the overview page and select “Add” in the Roles section.



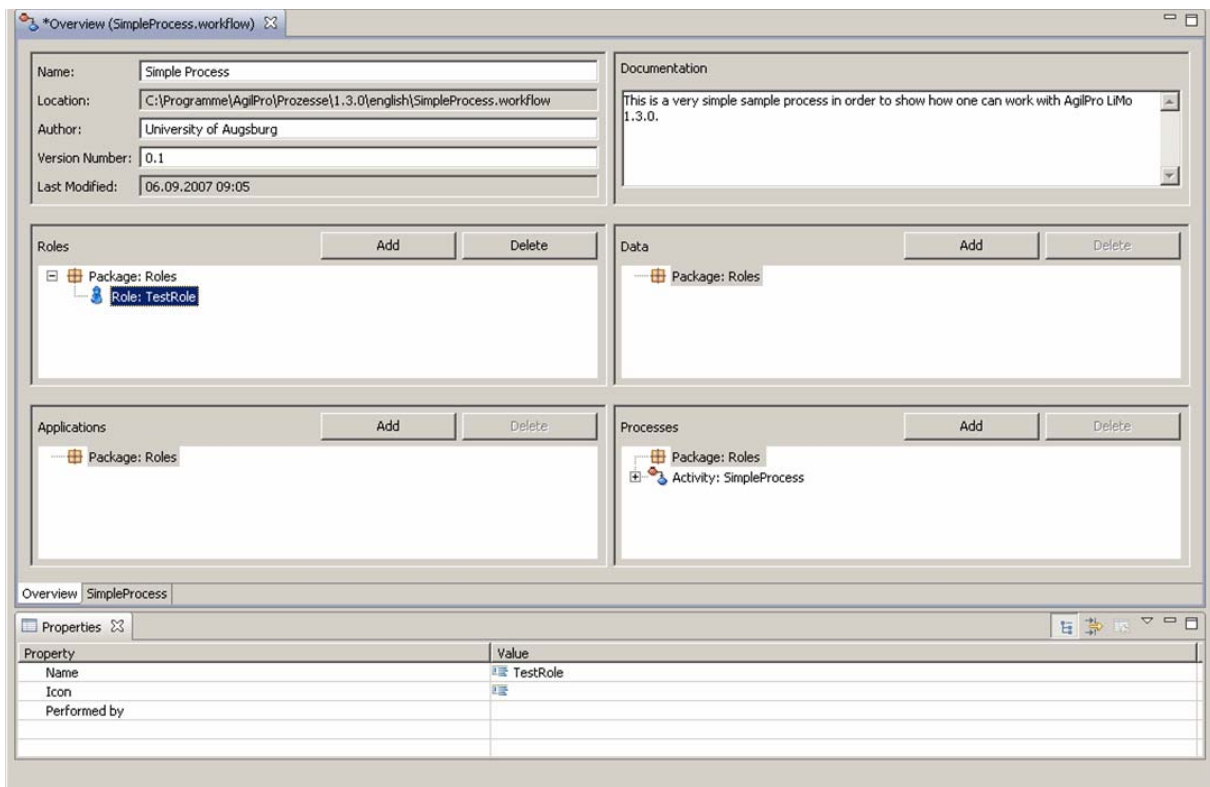
With clicking with the right mouse button on “Model: Simple Process” you can select “New package” and can enter the name of the package:



We enter the name “Roles” and click on “OK” and create a new role called “Testrole” in the roles package. You only need to create the package “Roles”, if you didn’t activate the button for automatically generating the packages, otherwise you can leave that step behind.

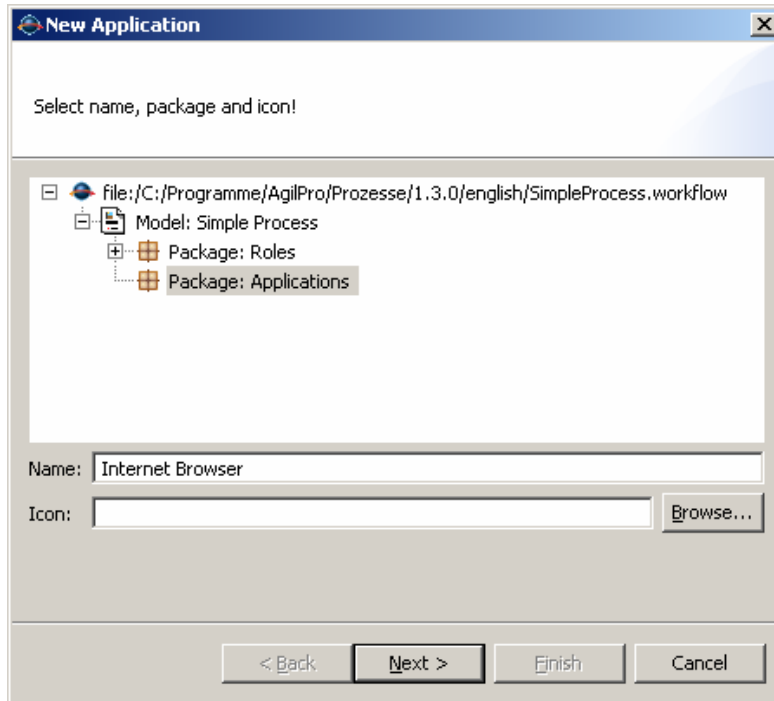


Now your overview page should look similar to the following:

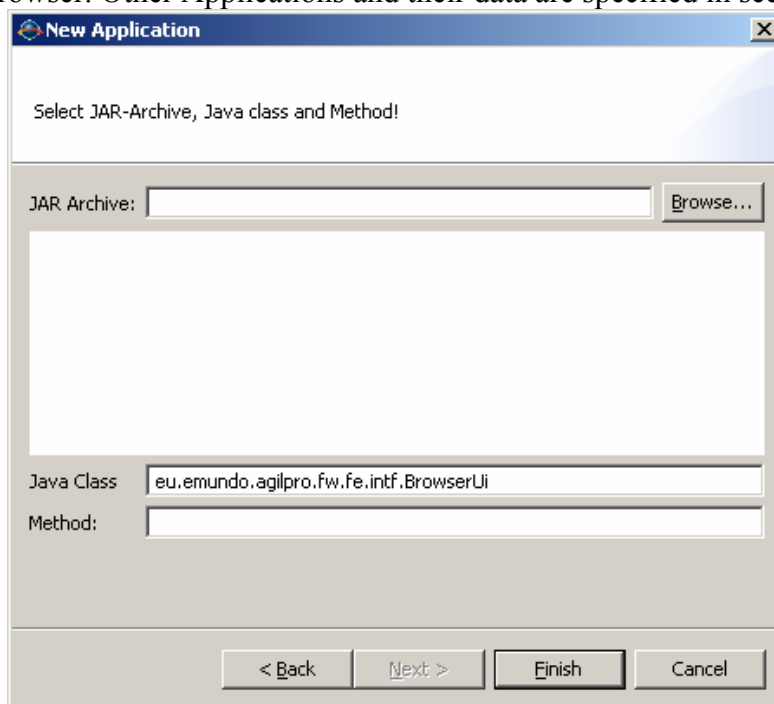


5 Adding applications and data

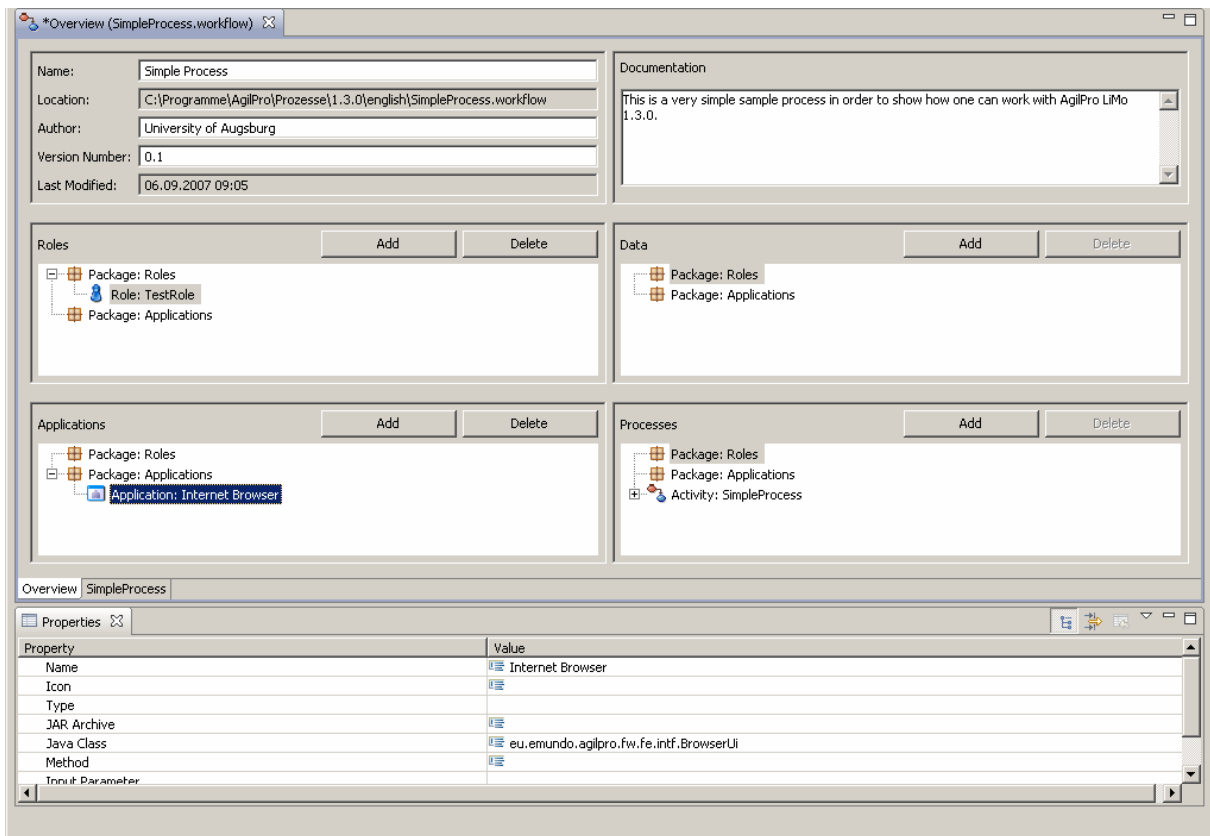
Similar to that we will add two packages for applications and data and insert our first application. In the new package “Applications” we will create an application called “Internet Browser”.



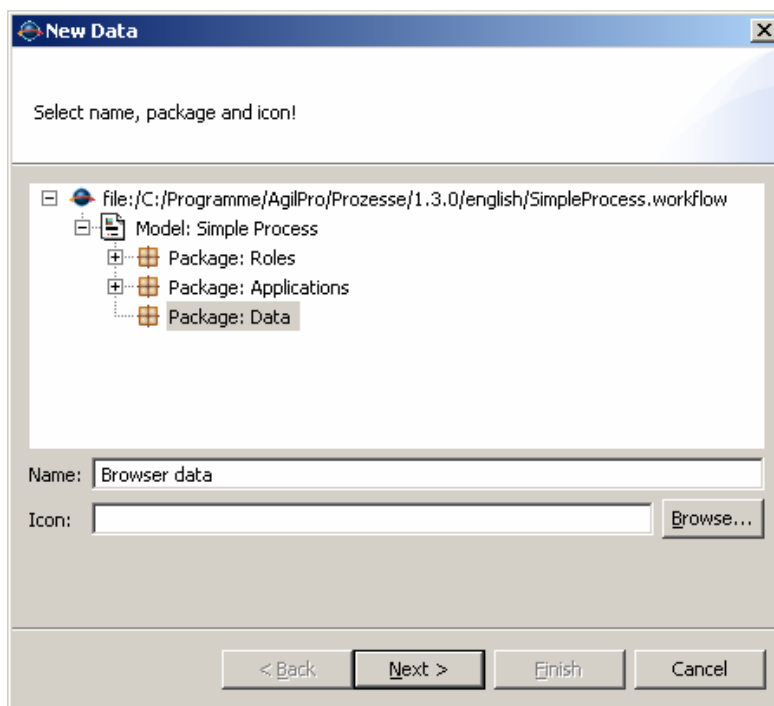
In the next window you can specify the JAR-archive of the program, the Java class and the method. At least you have to specify the Java class. There are several predefined Java classes that the AgilPro Simulator can recognize. One is “eu.emundo.agilpro.fw.fe.intf.BrowserUi” for an internet browser. Other Applications and their data are specified in section 12.



Now you should have one role and one application. What we need is now the URL that the browser should open when executing it.

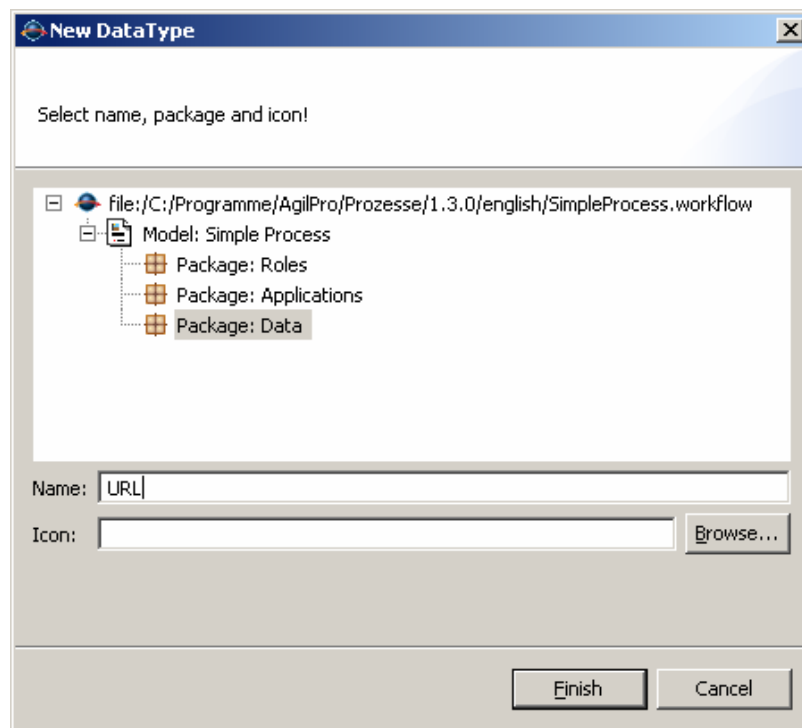


Therefore, we create a new package “Data” (now we have the same as when activating the auto-creation button in the new model wizard) and a new data item “Browser data”:

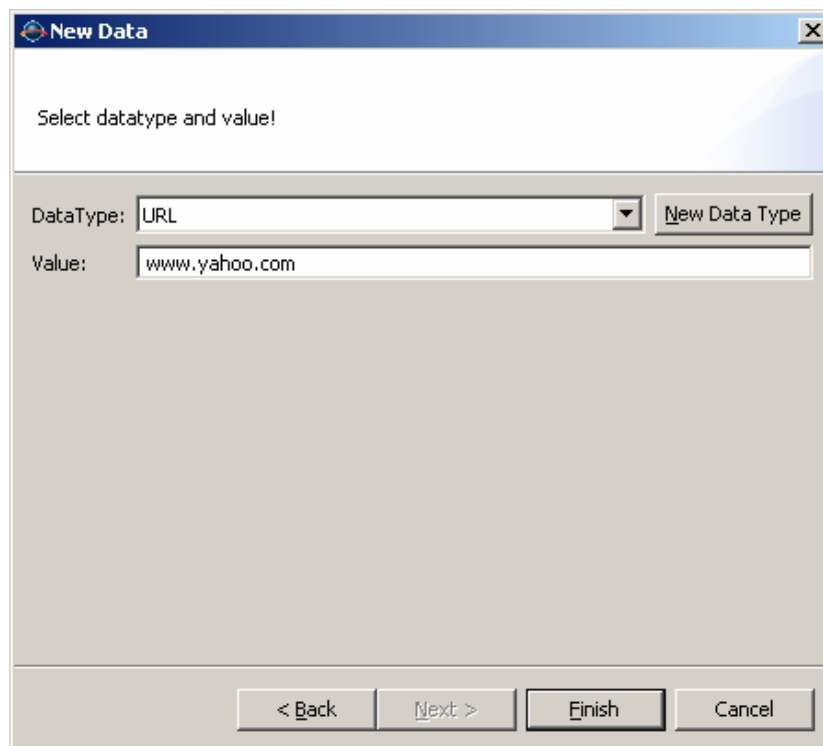


In order for the AgilPro Simulator to recognize what kind of data item this is, we need to specify the data type of the data item. There are several predefined data items that can be used with the AgilPro Simulator and Integration framework. For more information see section 12.

Right now, we want to define a URL for our internet browser. Therefore, we create a new data type in the data package with the name “URL”. This can be achieved with the “Create datatype” button in the wizard for creating a new data.

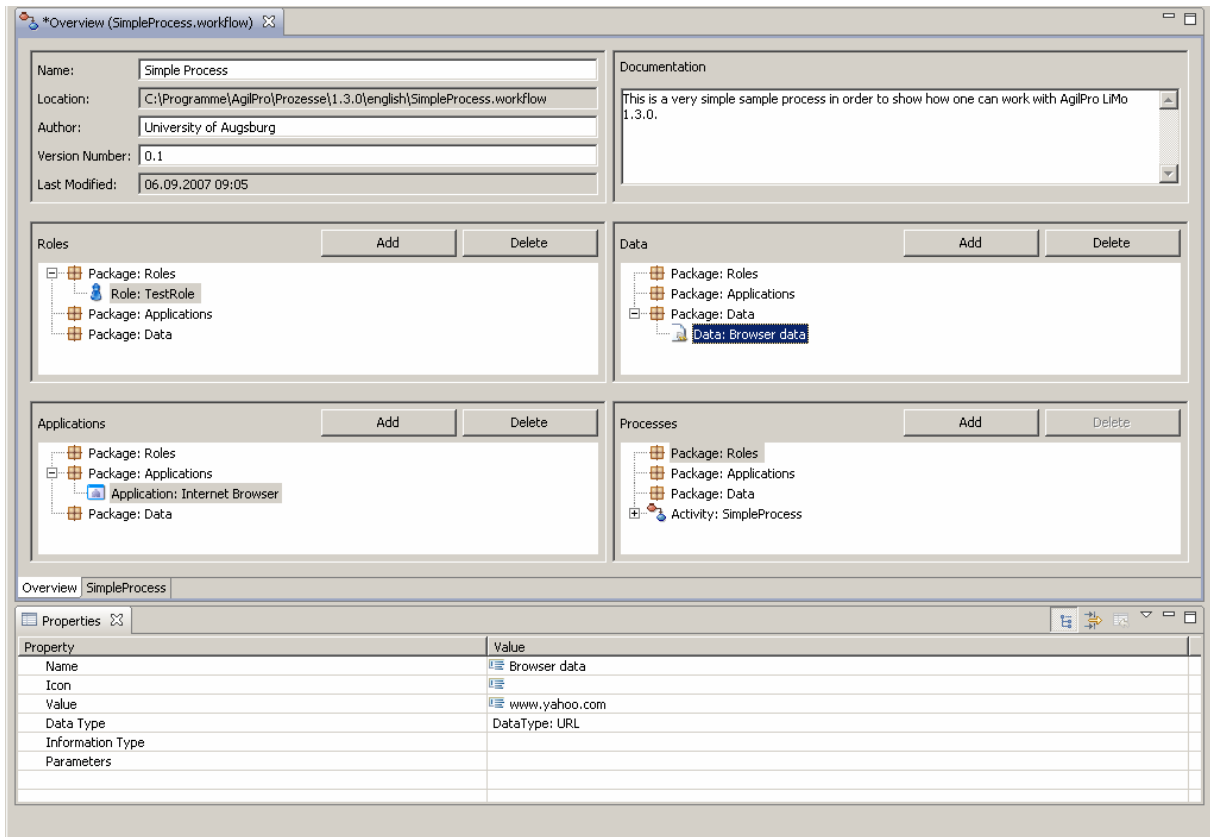


We select this new data type and give it a value: the URL that we want to open. In our case e.g. <http://www.yahoo.com>.



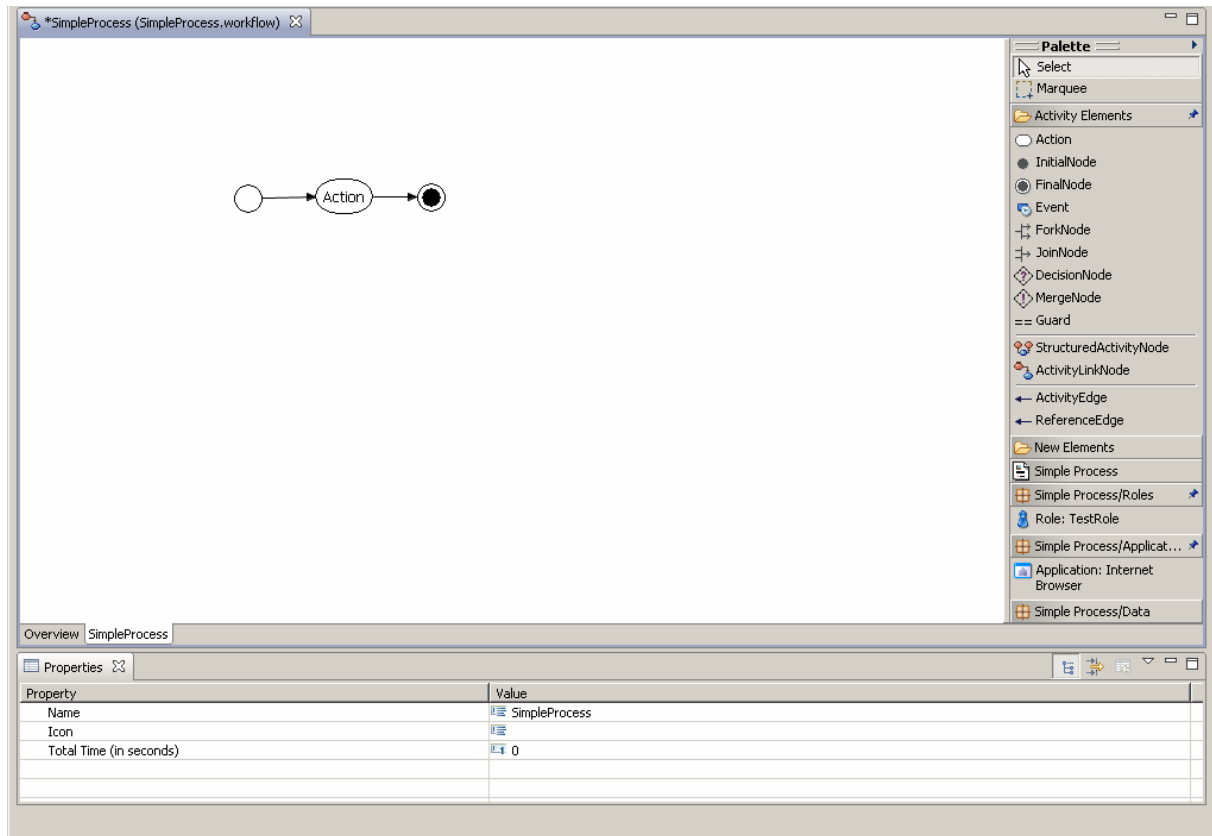
Now we got one role, one application and one data with a specified data type and our overview page should look like the following. Be aware that the delete buttons are only

enabled if the role, application or data is not used in a process model anymore, otherwise the buttons are disabled.

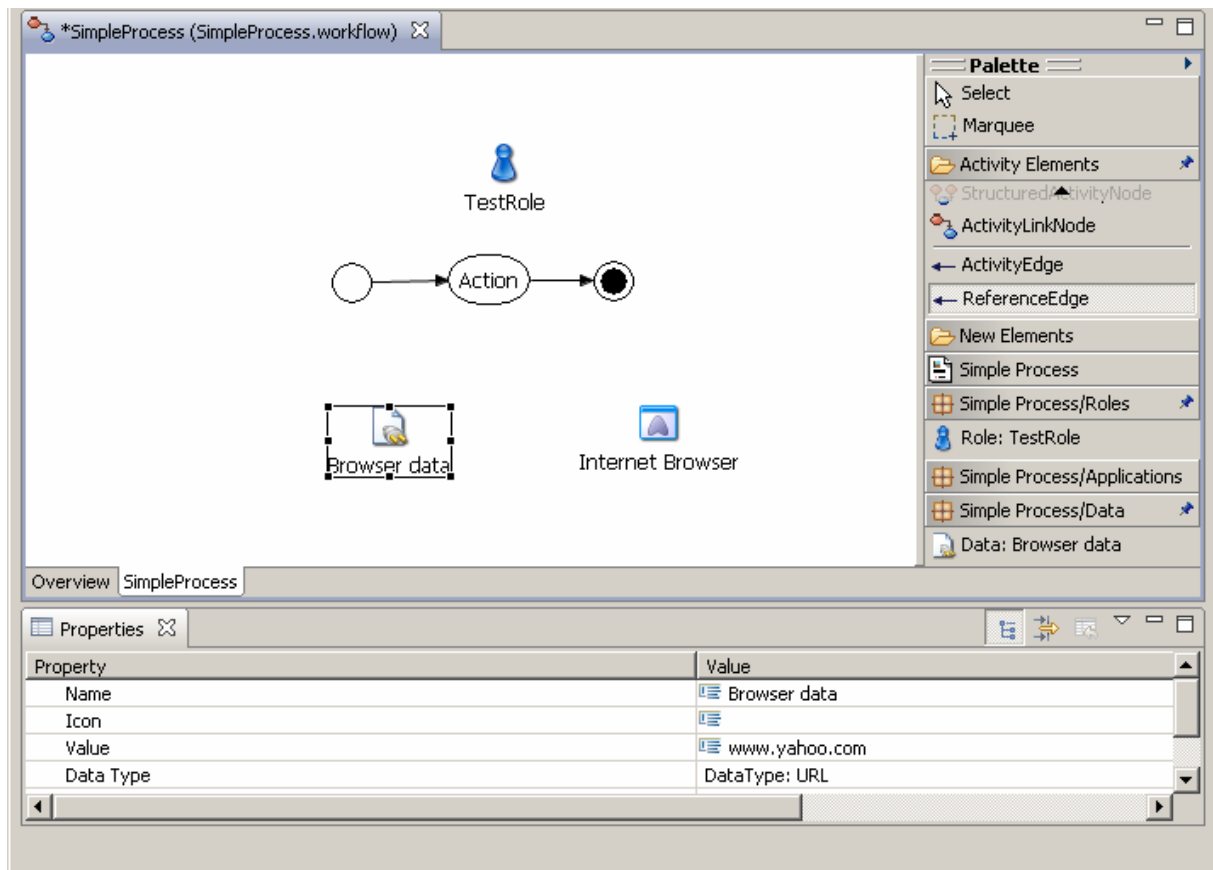


6 Making our process executable

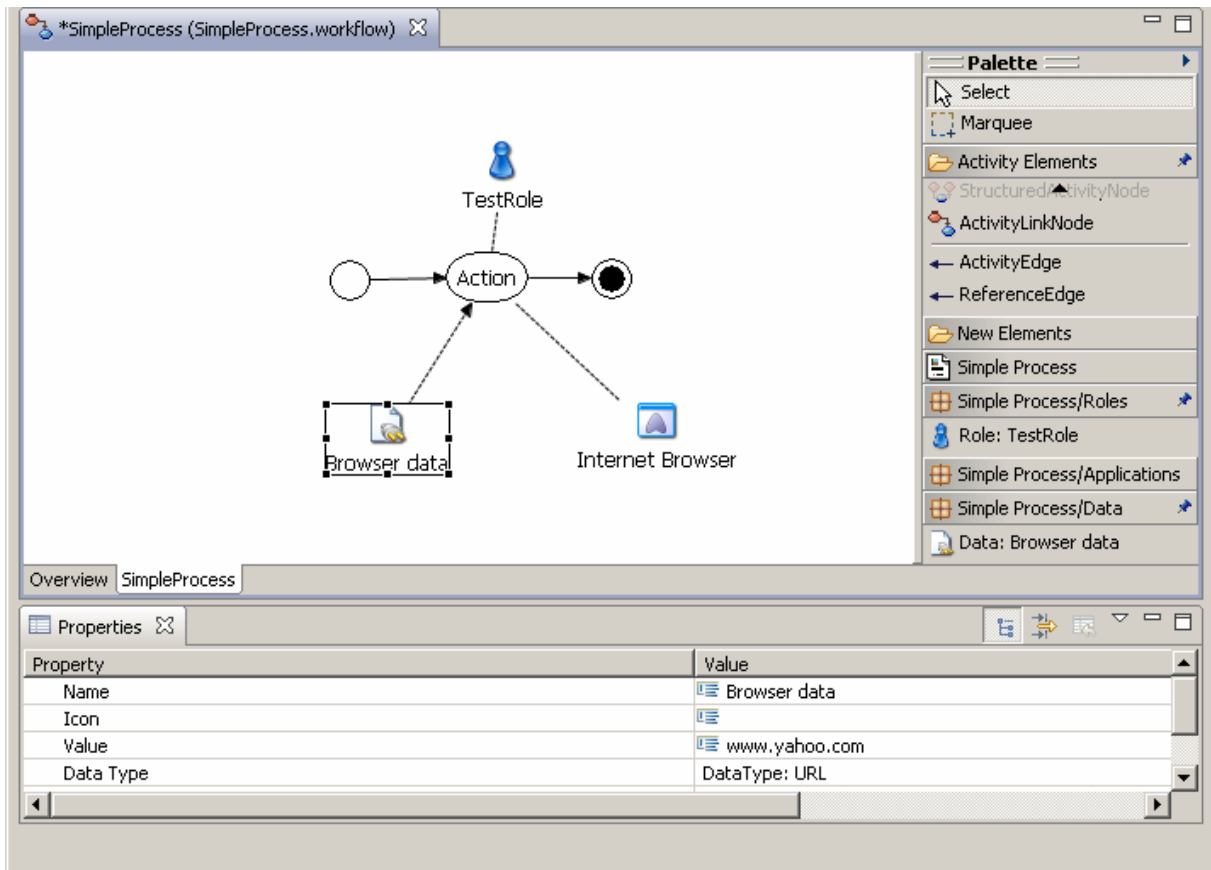
Similar to the changes in the overview page, also our graphical process page has been changed: now you can see the additional packages, the role, the application and the data in the palette on the right side:



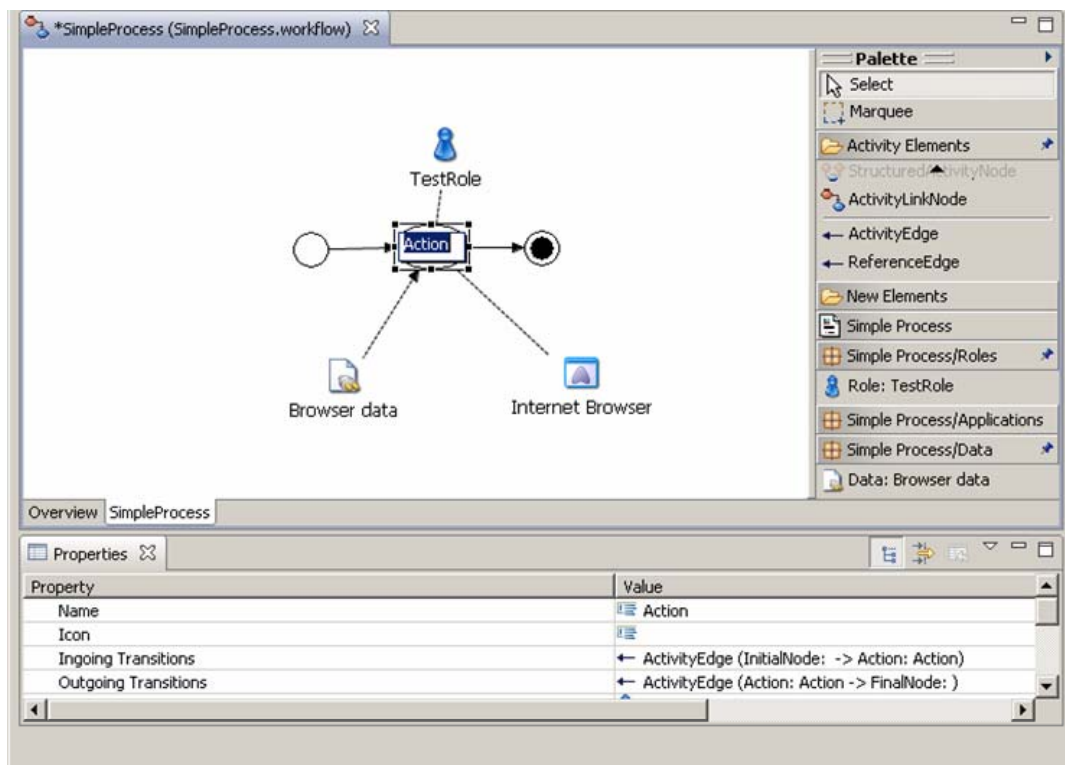
Now simply add these items by drag and drop into your process model:



You can then use the reference edge to connect these items with your action. Make sure that you start with the reference edge at the data and go to the action and not the other way round. With an edge from data to action you get an input edge, means that this data is necessary for executing the action. With the edge pointing the other way you would get an output action, means, that this data is created during the execution of the action.



Maybe we should rename our action to “Show browser”. Therefore, make a double click on the action or change the name in the properties field below.



Now we can save our process (“File”, “Save” or Control-S) and execute the process by clicking on the toolbar or selecting “Process”, “Execute”.

The screenshot displays the Eclipse IDE interface for a workflow project named 'SimpleProcess'. The main workspace shows a UML-like activity diagram with the following elements:

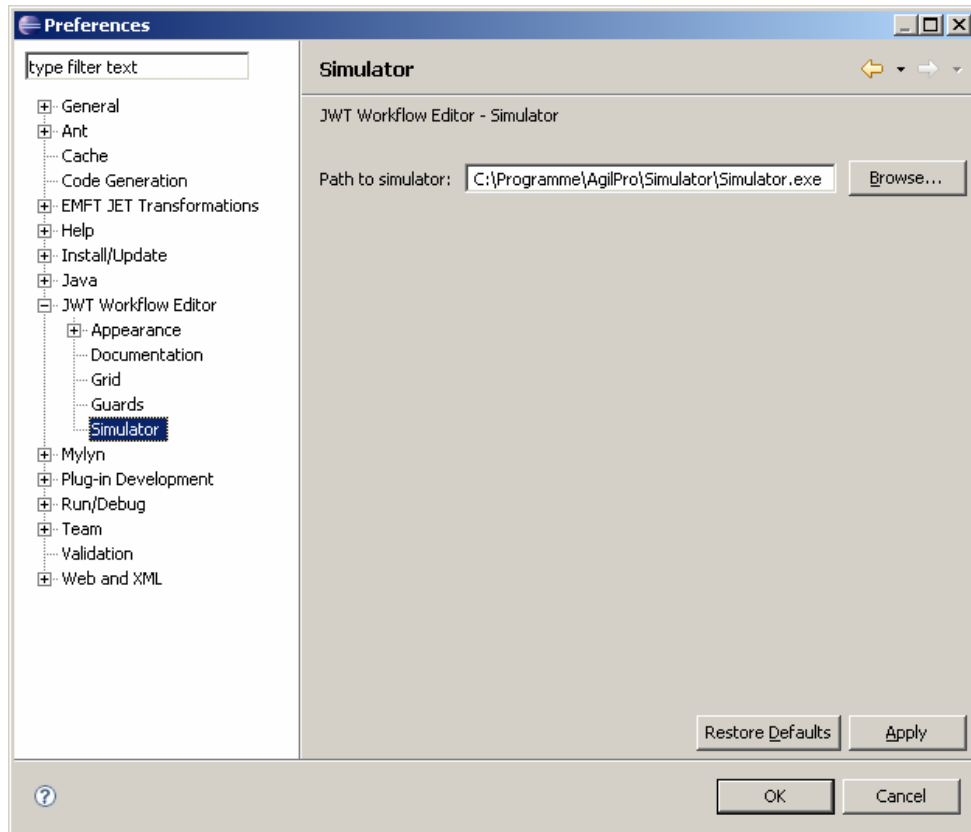
- TestRole**: An actor role represented by a blue pin icon, connected to the 'Show browser' activity.
- Show browser**: A central activity node represented by an oval with a white start circle on the left and a black end circle on the right.
- Browser data**: A data object represented by a document icon with a magnifying glass, connected to the 'Show browser' activity.
- Internet Browser**: A data object represented by a browser icon, connected to the 'Show browser' activity.


On the right side, the **Execute Palette** is visible, listing various tool types such as 'Select', 'Marquee', 'Activity Elements', 'Structured/ActivityNode', 'ActivityLinkNode', 'ActivityEdge', and 'ReferenceEdge'. Below the palette, the **Properties** window shows the following table:

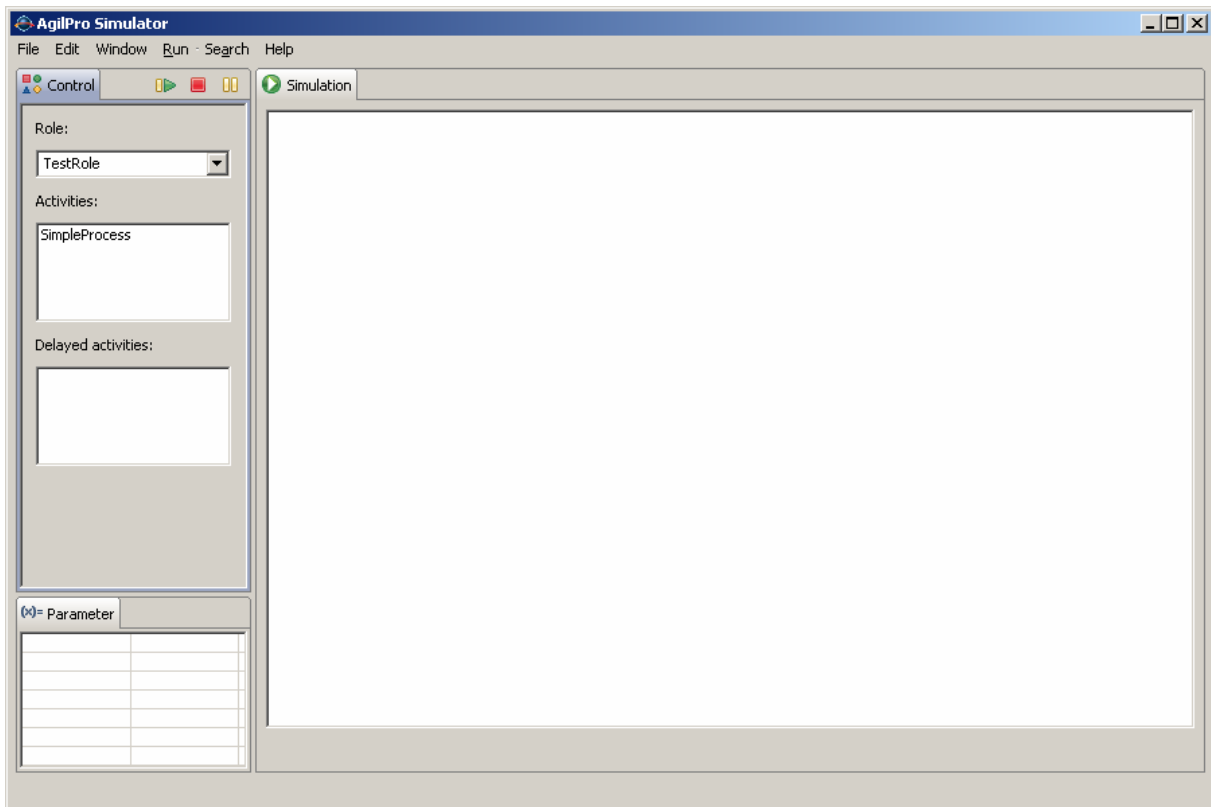
Property	Value
Name	SimpleProcess
Icon	[Icon]
Total Time (in seconds)	0

7 Simulating / Executing our process

Until now we did only specify how the process model shall look like, but we didn't use it to actually execute the process model. This is currently only possible using an external tool (however, we are working on that). This is the AgilPro Simulator which has to be installed on your computer and Eclipse must know where to find it. This can be defined in the preferences.



Now after selecting “Process”, “Execute” the AgilPro Simulator is opening. If you don't find that, click on the symbol  which does the same.



There is one role available for executing this process and this role has the possibilities to start exactly one process (activity). Make a double-click on the process “SimpleProcess” in order to start the execution. When starting the simulation automatically the corresponding application is started with the data specified, so in our case a browser is opening in the Simulator and opening the URL www.yahoo.com.

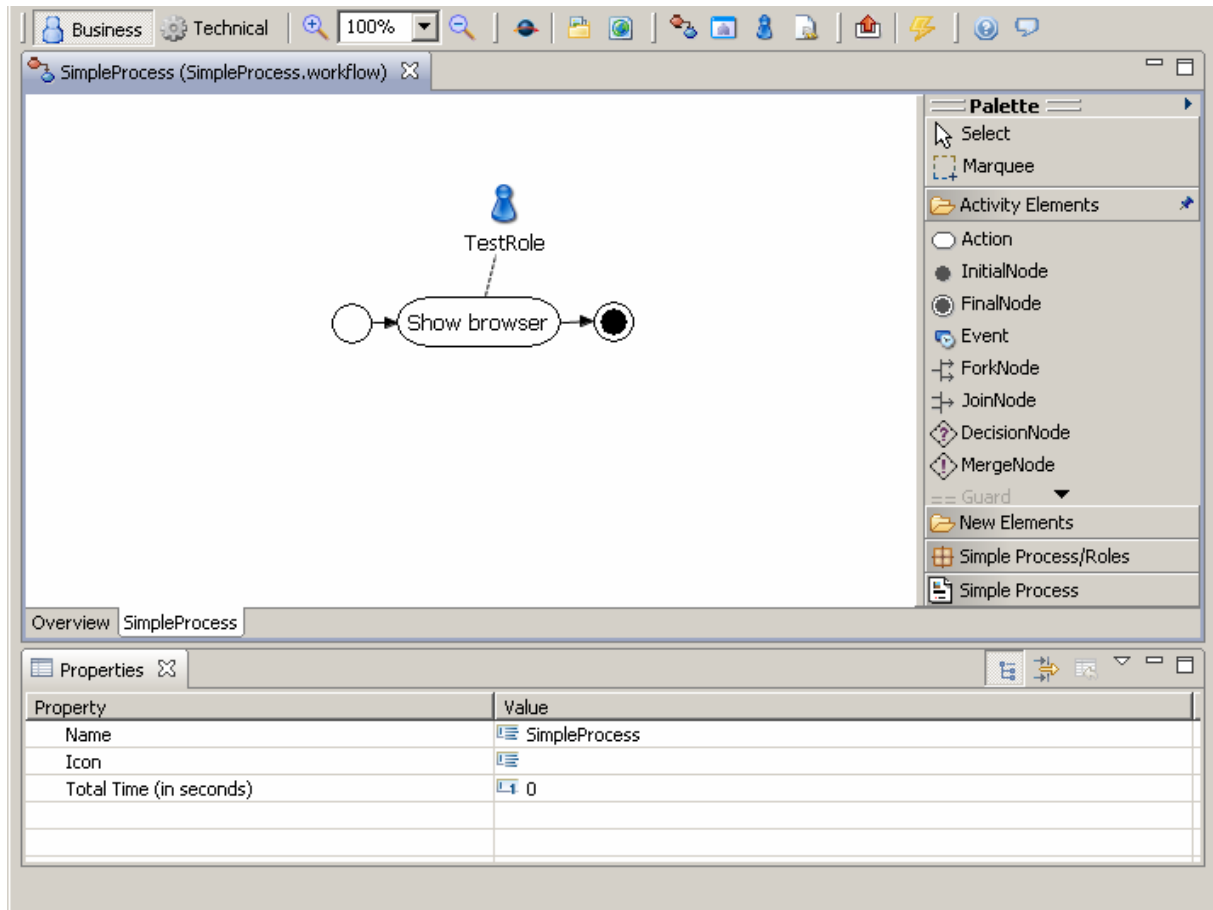


We could go on to the next step with the next button, but in our case there was only one action specified so we can stop our simulation here.



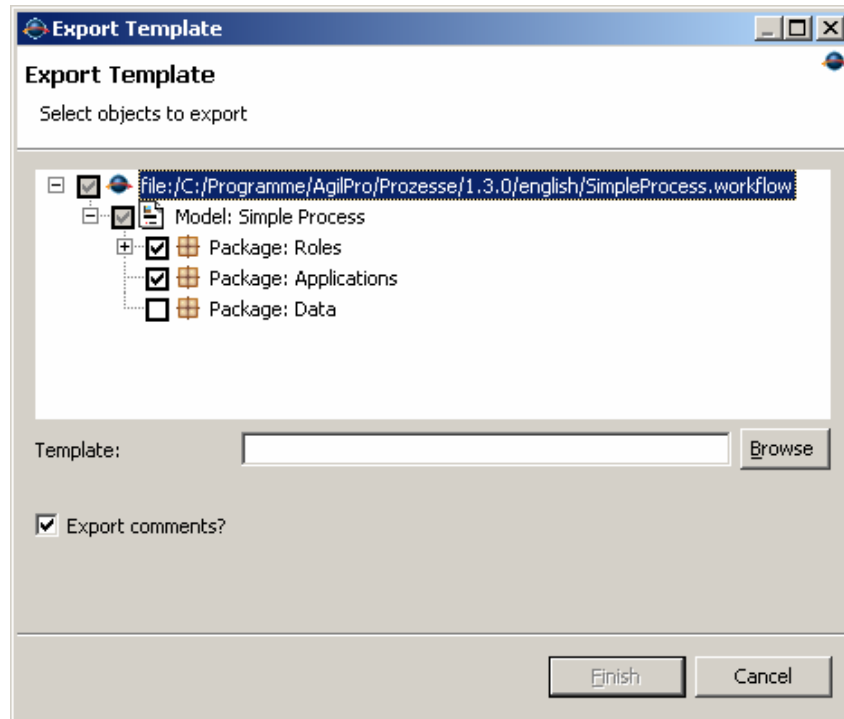
8 Different views in the modeling tool

You can select different views in the workflow editor: there are at least two views available: the business and the technical view. In the technical view you can see all the details described above. In the business view the presentation is limited to the most important facts that are interesting for a business manager: the control flow, the kinds of actions and the responsibilities. If you switch to the business view (by activating the Business entry in the toolbar) you can see the same process model as above like this:



9 Export workflow templates

Once you have created your packages with applications, data and roles you are able to export them to use them in future process models. Therefore, click on “File”, “Export workflow template” and select the packages that you want to export into a template file.



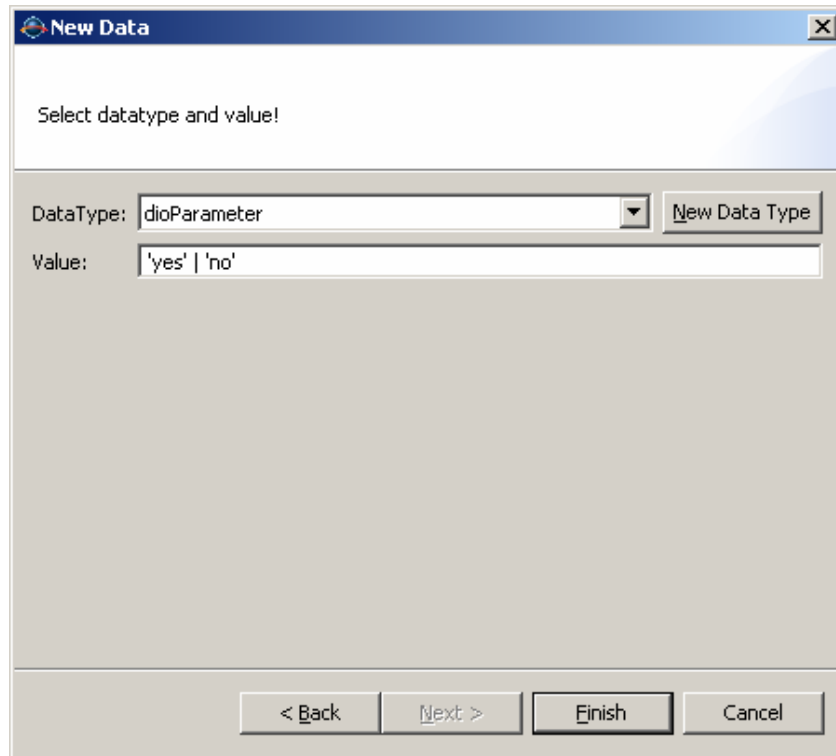
Choose the filename where you want to export the packages and all their content to and click on “Finish”. When you click on “Export comments”, then all comments are exported into the template, too. Comments can be created to each element in the outline view to store some additional information that does not fit anywhere else. Use it if you find it necessary and export the comments then, too. Otherwise simply ignore this option.

10 More complicated example

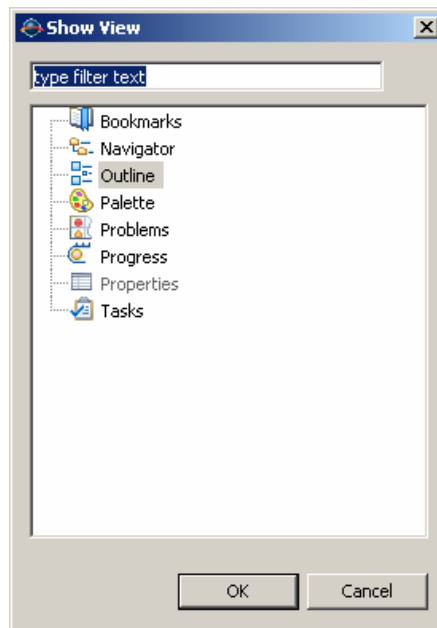
Let's make the example process a little bit bigger and more complicated. Assume that somebody wants to have a decision whether to go to the internet or not. Then you will need some additional applications, data and control nodes.

First we add a new application called "Generic GUI" with the class `eu.emundo.agilpro.fw.fe.intf.GenericUi` to execute.

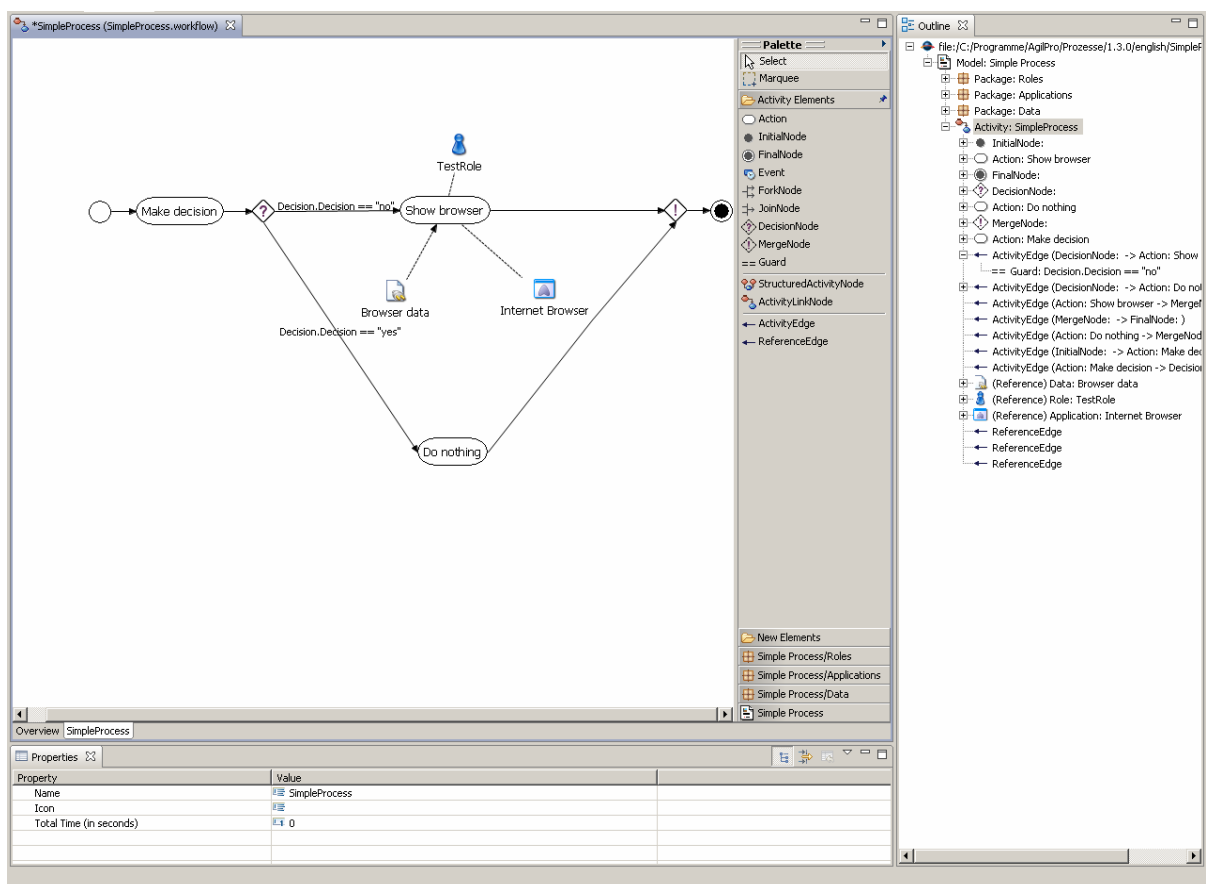
Then we create some data called "Decision" with a new data type called "dioParameter" and the decision possibilities 'yes' and 'no'.



Sometimes it might be necessary to get an additional possibility to select control flow elements. Therefore, go to "Window", "Other", "Show view" and select the "Outline" view:



Then you can see the outline on the right side which is useful for editing the guards:



There is an easy way to create guards so that the conditions can be tested in the Simulator without any additional effort. To use this feature you always have to create two elements in your model first (what you already did if you followed the steps described above).

- An application using 'eu.emundo.agilpro.fw.fe.intf.GenericUi' as Java Class, which offers a generic user interface that let's you choose the values of your guard conditions in drop down menus.
- A datatype named 'dioParameter'

Create data elements for each property you want to use in a guard. As datatype always choose the dioParameter. The value field of the dioParameter has to be filled with all possible values, that should be available for selection, in the following way: 'value1' | 'value2' etc. (remark: the ' ' are really needed). For this example create a data Element “Decision” and use the dioParameter as datatype.

The application for the generic user interface and all data elements that are relevant for the guard you want to specify have to be connected to the activity before the decision node which is the source of the edges with your guards.

Now you are able to create the guards for each activity edge. Guards consist of a *textual description* and a *short description*. The *textual description* is shown if the workflow editor switches to the business view. It is used to simplify the condition or to put it into an expression in natural language and has no influence on the interpretation of the condition.

The *short description* represents the technical expression of the condition. It has to satisfy several requirements regarding the syntax. To construct the expression all letters, brackets and the common operators ==, <=, >=, !, &&, || are allowed to be used.

To create a correct guard using data objects, you have to reference to your data objects in a special way. Assumed you want to use your data element named “Decision”, you can access it in the short description by writing Decision.Decision. The values you have specified for this data element can now be used in the boolean expression for your condition, e.g. a simple short description of a guard could be:

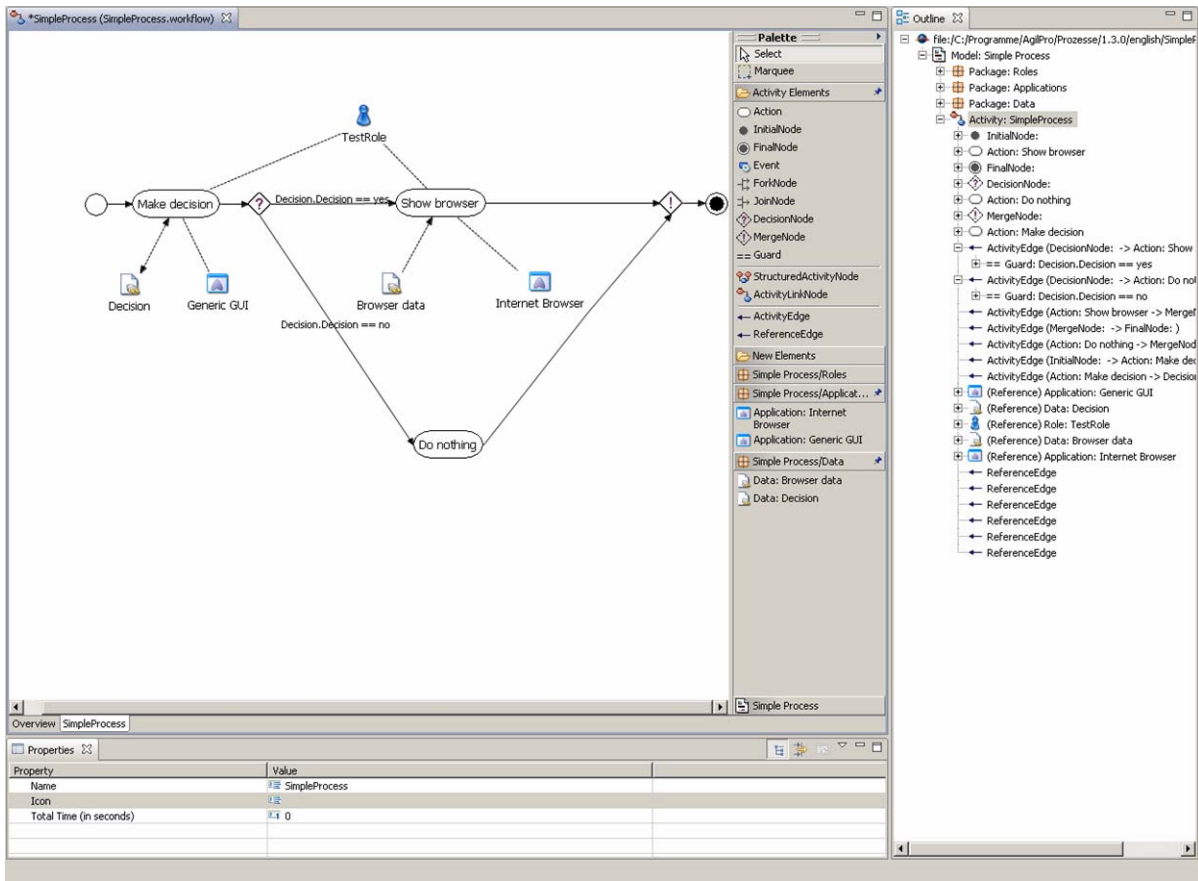
Condition.Condition == yes

(remark: the ' ' you had to use in the specification of the values for the data element have to be omitted now)

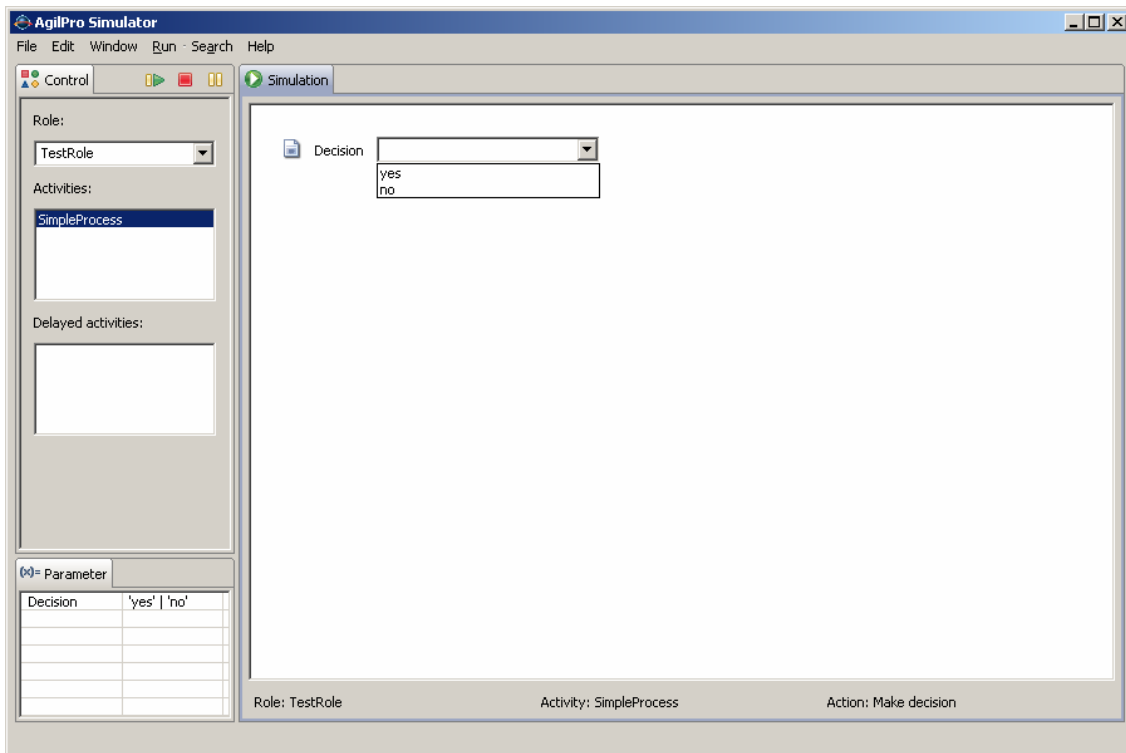
Now take a look at the Outline view. If you have specified your guard correctly, a GuardSpecification is created automatically under your guard. If it is missing in all probability you have made a syntax error.

The guard specifications become more complicated if you use the operators && or || in the short description of your guard. Now there are several guard specifications ordered in the hierarchy you forced by your boolean expression. The simulator evaluates only the bottom guard specifications and connects them with the adequate boolean operator. The guard specifications that collect only other specifications are not part of the evaluation by the simulator.

After you’ve added the guards Decision.Decision == yes and Decision.Decision == no to the activity edges as well as the additional application and data, you might have the following process. Note that the decision has now an InOut edge, meaning that it is as well an input to the action “Make decision” as an output. You can achieve this by drawing the line once from the action to the data and once from the data to the action.



Executing this in the AgilPro Simulator you can see a generic GUI that asks you for “yes” or “no” (okay, quite simple, but you can of course think of much more complicated examples).

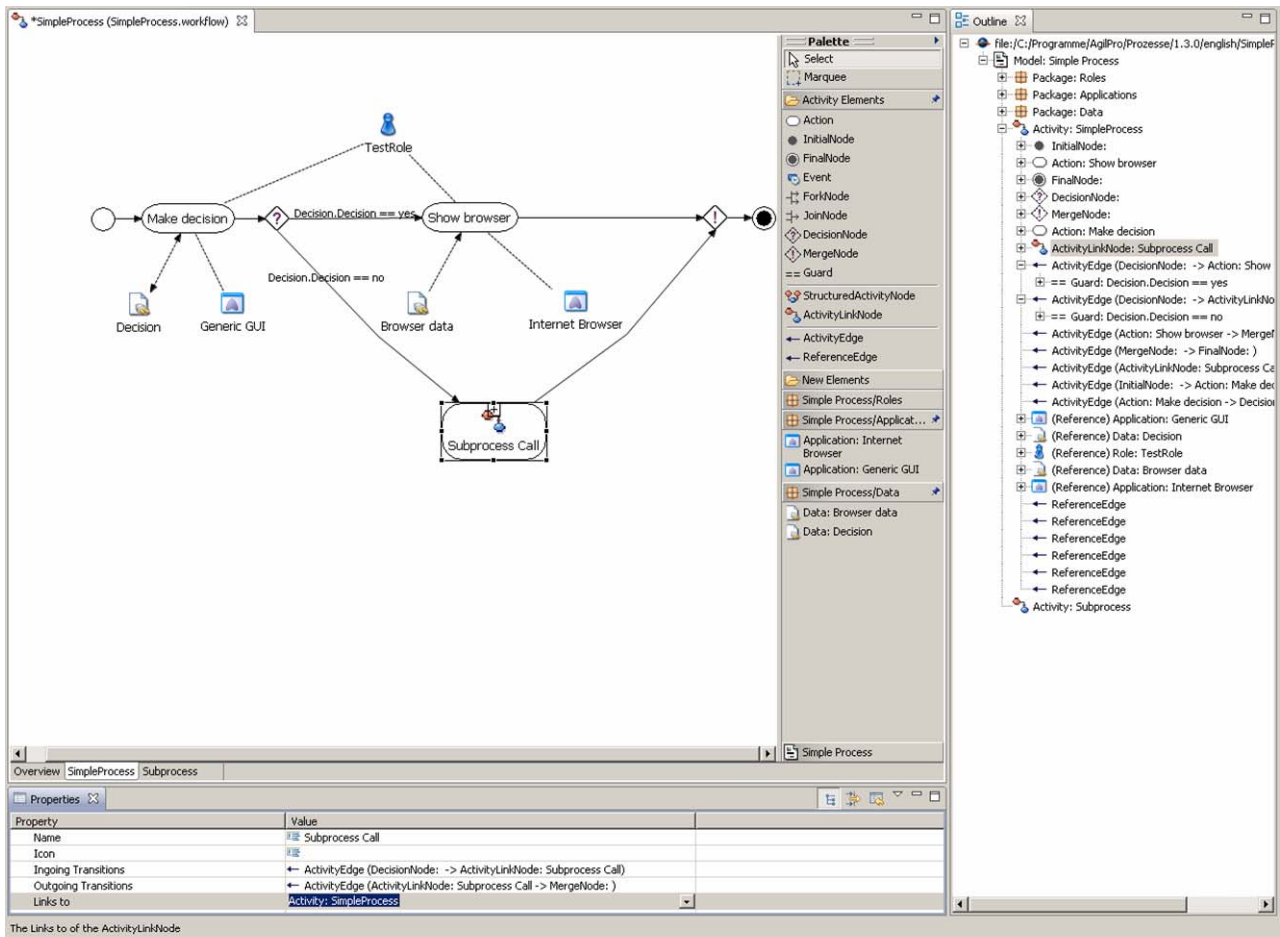


If you select “yes” then you can see the browser, otherwise not.

11 Subprocesses

You can also create some processes which are nested into other processes. Therefore, we will choose an ActivityLinkNode and name it “Subprocess Call”.

In order to link that ActivityLinkNode to another process, we need to create a new subprocess and add this to the properties field “Links To” of the ActivityLinkNode.

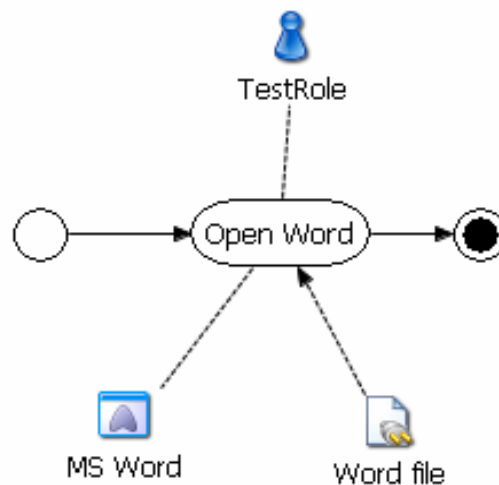


The subprocess can contain every possible combination of actions and again ActivityLinkNodes to point to a third process. Only cycles (from A to B and in B back to A) are not allowed.

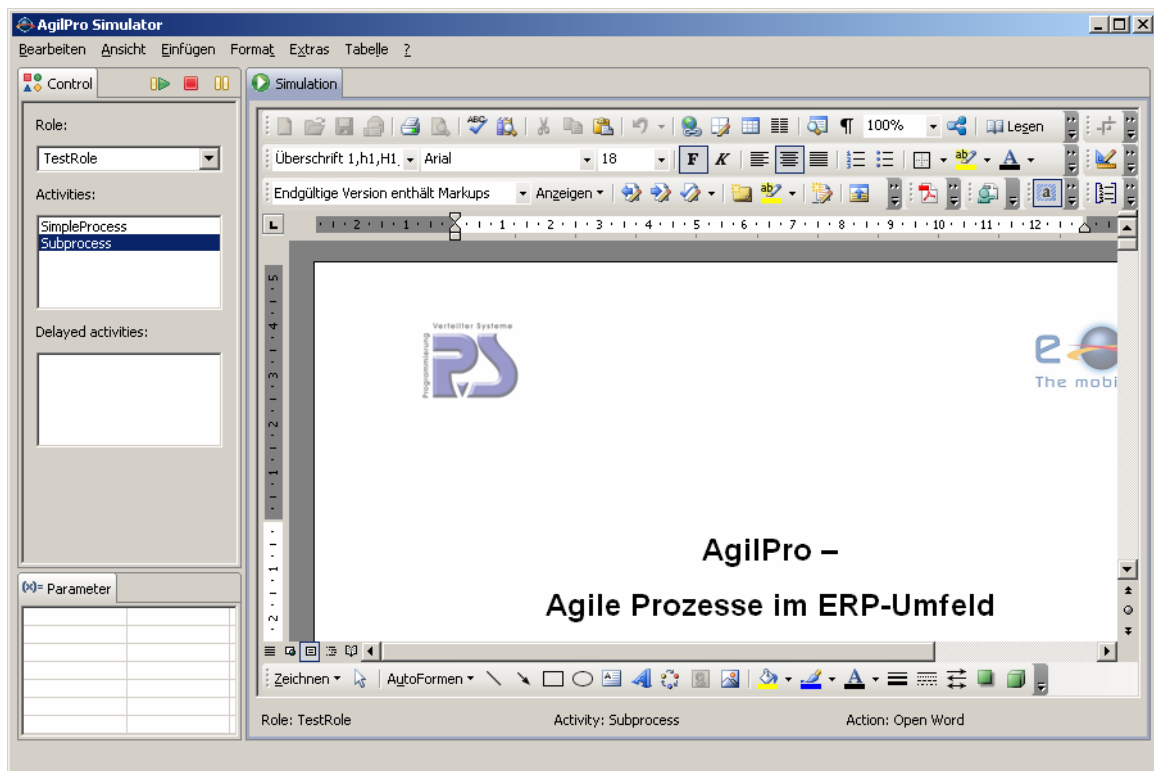
12 Additional applications

It is not only possible to open a browser or to make a generic GUI, but also to Office products (such as OpenOffice or MS Office and here MS Word or MS Excel or the OpenOffice derivatives). In addition you can start a search using a Google web service, make a meta search, open a PDF file, and so on. If you need additional adapters for other applications, please don't hesitate to contact us and we will be happy to assist you.

For opening e.g. a word document we need an application with the Java class "eu.emundo.agilpro.fw.fe.intf.WordUi". Additionally we need a data with the data type "filename" and the link where the file is located as value (e.g. /Testdocuments/Test.doc or also C:\Documents\Doc.doc). This is now the content of our subprocess and looks as following:



The simulated subprocess looks then like this:



Other applications, their data and datatypes can be found in the example process “ExternalApplications.workflow”.

13 Other features

Several other features are available, that haven't been described in this document. Please click with the right mouse button on a free place in the graphical editor and you will find that you can save the workflow model as an image, you can enable a grid (see picture below), enable an auto-alignment function, etc. If you have any further questions concerning what is possible with JWT, please don't hesitate to contact us.

