2.0

Manual

12th February 2009

produced by the myGrid team

http://www.mygrid.org.uk

Table of Contents

1 Introduction

Welcome to the help pages for Taverna 2.0

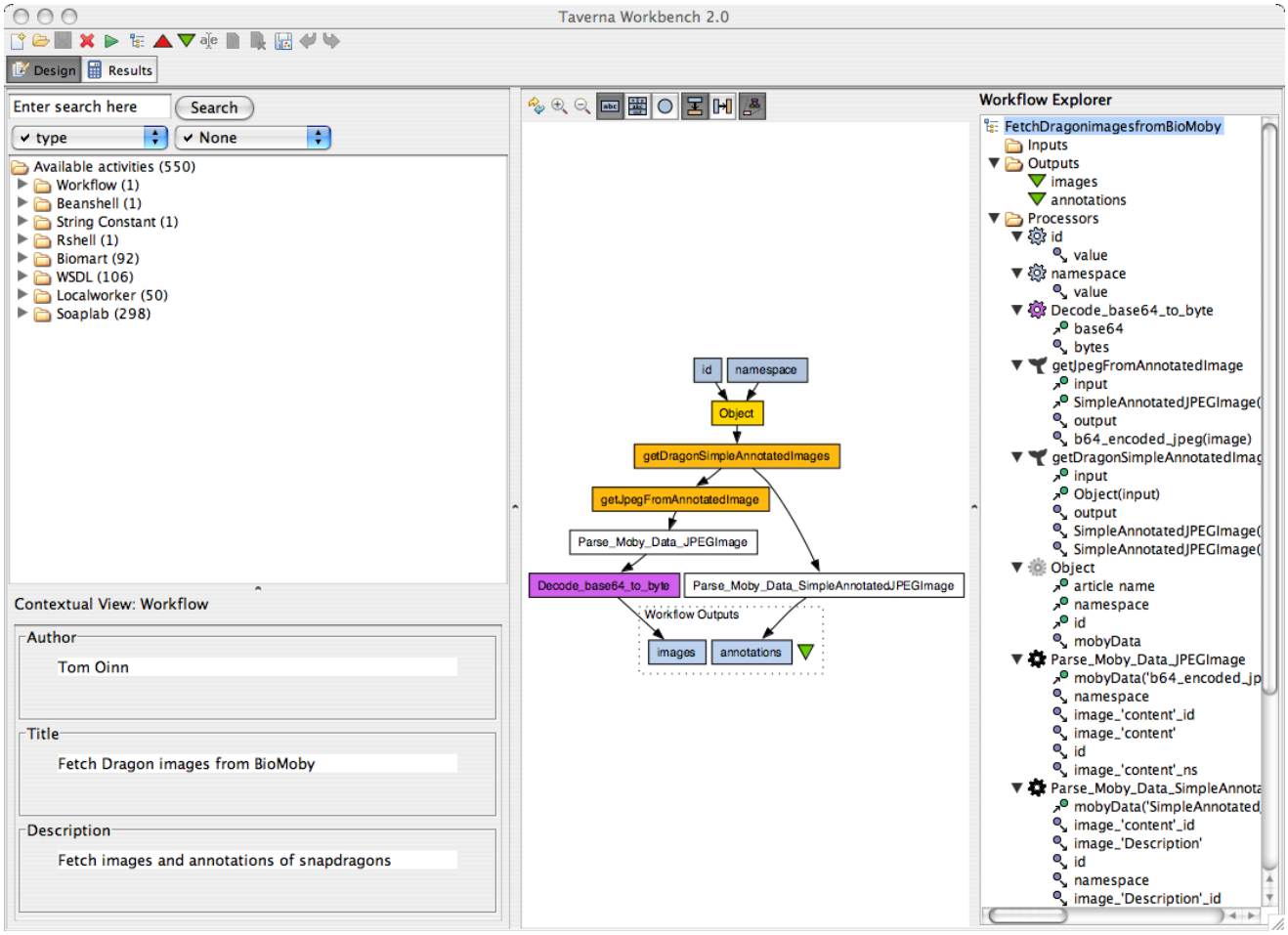
If you are already a Taverna user, this guide will help you become familiar with our new features and changes to the previous version. We have made some big changes to the user interface as well as to the underlying enactor, so please have a look.

If you are a new Taverna user, please see our introductory video tutorial at http://www.mygrid.org.uk and explore these help pages to get started with designing and building workflows.

You can also look at the quick guides to creating and running a workflow.

2 Quick Guide

Here is a quick introduction to what is new or different in the user interface of Taverna 2.0 from Taverna 1.



Activity Palette

The available services panel is replaced by an activity palette (in the upper left of the image). This is still the place to find services available for your workflows, but it is now much more configurable, so you can search for services (or activities) by location, type, etc, The results of a search are returned as a separate list instead of highlighting matches in red. There is still also a free text search,

One possible issue with the activity palette is that by default the Local workers are given in one big list; to show the traditional categories select Type for the first column and Category for the second column.

Service Scavengers

Previously new sets of services were added to the services panel by right-clicking on the Available Processors and choosing an option from the menu. This is replaced by picking an option from the top-level Activities menu.

Graphical Editor

In Taverna 1, the graphical representation of the workflow was "view only". In Taverna 2, the graphical editor (in the centre of the image) is where you create and edit your workflow.

To add a service to the workflow from the activity palette, drag it onto the Graphical Editor.

The diagram is interactive, so if you switch to the *Display all processor ports* mode now a toolbar button, you should be able to drag connections directly from an output port to an input port, or the other way. You can also do this in the simpler *Display no processor ports* mode, but you will then have to right click on the processor first and select *Link from output* or *Link to input* to initiate the connection. When selecting the destination of the new link a pop-up will ask you for the port name if the processor exposes more than one port.

To add input and output ports, right click on a blank space and select Create new input or Create new output. To change a link, just drag it to the new destination, or right click on it (easier on the wide arrow side) and select Remove datalink. Processors can be renamed and removed by right clicking on the processor. Note that in all ports mode you need to right click in the middle box with the processor name.

There are also toolbar buttons for adding input and output ports.

Contextual View

When you left-click on a processor in the workflow explorer or in the graphical editor (you may have to click in the middle of the processor box), the contextual view in the lower left corner is updated to give a description of the processor and its activity (ie. the service). You can usually configure the activity by clicking the Configure button. For instance, for a Beanshell activity, this should give you a dialog box for editing the script and setting the ports.

One new feature is that local workers are now implemented as beanshell scripts, so if a local worker does not do exactly what you want, you can configure it and customise the script. This only changes that instance of the local worker in your workflow, so you will not affect other local workers.

Nested Workflows

In Taverna 1, nested workflows were added to a workflow by a Add nested workflow button in the advanced model explorer. In Taverna 2, nested workflows are added to the current workflow from *Workflow->Nested workflow* in the activity palette.

In Taverna 1, the nested workflow was opened from a file or web location when it was added to the workflow. In Taverna 2, the nested workflow can be configured by clicking on it and altering its settings in the contextual view. You get the option to Edit workflow to modify the nested workflow, or Open from file to replace it with a workflow loaded from a file. When editing a nested workflow, you need to do File -> Save to make your changes be reflected in the outer workflow. This save does not save to a file, but saves to the nested workflow processor in the outer workflow.

Undo and Redo

New! Taverna 2 has undo and redo capabilities. For instance, if you removed a processor and realised you have made a mistake, you can click Edit->Undo in the menu to get it back. Note that at the moment there are a few actions that are not yet undoable, and you might get an error message if this is the case. We are working on making every workflow editing action undoable, and also to be able to save the list of undoes so you can undo even after restarting Taverna.

List Handling Strategy

If you click on the input ports of a processor that has several input ports connected, you can see the list handling strategy for that processor in the contextual view. If you click configure, the iteration strategy can be edited in a similar way as in Taverna 1.

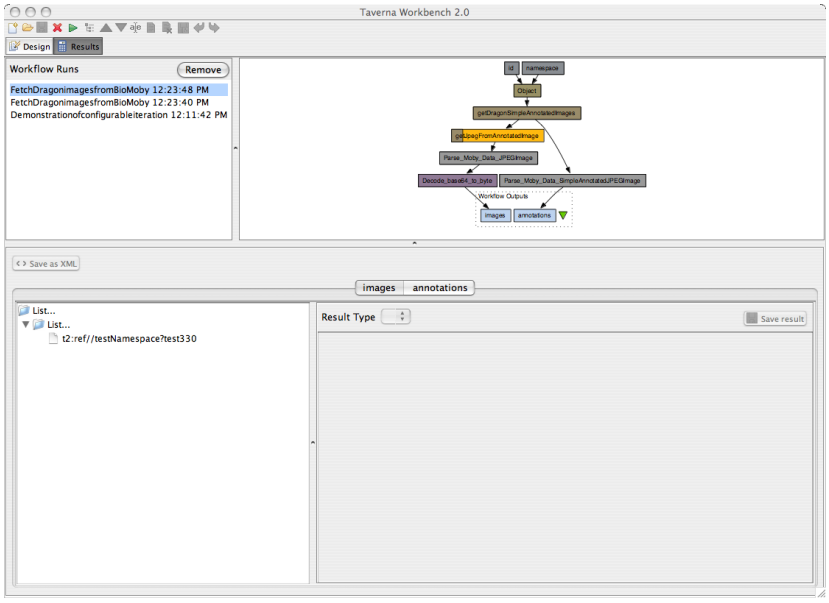
Workflow Explorer

The Workflow Explorer (on the right of the image) replaces the advanced model explorer of Taverna 1.

Running a Workflow

When running a Taverna 2.0 workflow, available through File->Run or the uppermost green Run button, you are able to input the data for your workflow. The capabilities of Taverna 2's dialog are similar to those in Taverna 1.

When you launch the workflow, Taverna 2 switches to the results perspective shown below.



While a workflow is running you will be able to see the progress in the monitor view on the upper right.

You can see the result items in the port results view (lower left) appearing as they are produced instead of having to wait until the workflow is finished. To view the data, click on the item in the left hand column. There is a tab for each output port.

Workflow File Format

The file format of Taverna 2.0 has changed from the Scufl of Taverna 1 to support various new features. You will notice that the file extension is .t2flow, unfortunately you cannot open Taverna 2 workflows in Taverna 1. However, Taverna 2.0 can open most Taverna 1 workflows, and will do a translation to Taverna 2.0.

Note that although some workflows could look slightly different (for instance, default values are exposed as string constants), functionally they should be the same, except in a few cases. If anything goes wrong when translating, an error message should appear; we are working on improving this error message to be more detailed about what could not be translated, and then to still open the workflow to allow you to correct the errors (say with red boxes for services that no longer exist).

If you encounter a Taverna 1 workflow that does not open or does not run correctly with Taverna 2.0, let us know via the taverna-users mailing list (see http://www.mygrid.org.uk for details).

File Menu

For convenience the File menu now has Save all and Close all. From the File->Open dialogue it is now also possible to select several files to open at once. Note that there is currently not a status bar or

feedback while the workflows are opened in the background, you might find that some workflows take a few seconds to open.

3 Creating a Workflow

We will create a very simple Taverna 2.0 workflow to list some cities in a country.

To start creating a new workflow, choose "New workflow" from the file menu.

Adding the service into Taverna

Taverna 2.0 does not, by default, "know" about a suitable activity to perform the functionality To add the capability click "Activities" and select "New Activities" and then "WSDL...". Enter the location of the WSDL, in this case *http://www.webservicex.com/globalweather.asmx?WSDL* and click OK.

After a while, the services will become available in the activitypalette. If you choose to order by "type" and "url" then you can see the two new services under WSDL and the location.

Adding the activity into the workflow

To add the activity into the workflow, click on GetCitiesByCountry and drag it into the graphical editor

Creating the workflow inputs

The workflow needs to have an input, in this case the name of the country to query. You can create a new workflow input by right clicking in a blank area of the graphical editor and select Create New Input. This will then ask you for a name for the input, you can change this later but for now use 'country'. Click on 'OK' and you should see your new input appear in the graphical editor and workflow explorer.

Similarly, the workflow will need an output. Follow an equivalent process but this time clicking on 'Create New Output' and using the name 'cities' to create an output. Again, you should see the output appear in the two windows.

Splitting the WSDL inputs and outputs

For many services, you can connect your inputs and outputs directly to the ports of the activity. However, for WSDL services you may need to "split" the XML used to hold the input and output values. To do this, click on the activity GetCitiesByCountry in the graphical editor.

A description of the activity will appear in the WSDL contextual view in the lower-left. Scroll down and click both "Add input XML splitter" and "Add output XML splitter", choosing "parameters" in both cases.

Connecting everything together

To link the input port click on "country" and drag to the "GetCitiesByCountry_input". A connection will appear in the graphical editor. If there had been several ports that you could have connected to, then the selection would have been offered.

To connect from the "GetCitiesByCountry_output", right click on it and select "Link from output GetCitiesByCountryResult" and drag to "cities".

There are several other ways in which you could make the connections.

Running the workflow

Select 'Run workflow' from the file menu. In the [post="run-workflow"] dialog, as there is only one input port, you can click "New string". Double click "abcd" and replace the string "abcd" by your chosen country. After you have entered it, you can click "Launch workflow".

Taverna 2.0 will switch to the results perspective. After the workflow has run (indicated by GetCitiesByCountry turning grey), you will see a string similar to "t2:ref/testNamespace?test19" appear in the lower left panel. This is the identifier of the output result. If you click on this, you will see the output appear in the rendered result panel.

Congratulations, you have created and run a workflow.

4 Running a Workflow

You should be able to run an installed Taverna 2.0 by invoking the 'run.bat' or 'run.sh' files or installed application, for Windows, Linux/Unix or Mac respectively.

If this fails to work you should re-check the installation, Java version and GraphViz version (linux only). If you have done this and the system still doesn't work please let us know via the mailing lists.

Opening a workflow

As a first quick demonstration of Taverna you can load one of the example workflows. Download one of them from the examples page on *http://www.taverna.org.uk/taverna-2-0/taverna-2-0-examples/*. We'll assume you picked the **Fetch PDB flat file**

Select open workflow from the file menu. You should be presented with a file chooser dialog box, the exact appearance of which will be determined by your operating system. In this demonstration, choose the workflow you downloaded.

When you click OK, if the workflow loads correctly, the graphical editor and workflow explorer panels are populated with the workflow defined in the file. There may be a short delay while Taverna contacts the network to get more information about the resources the workflow uses - if the delay is a very long one followed by an error you probably need to check the configuration of your HTTP Proxy preferences.

Enacting the current workflow

Aside from creating and editing, the most useful thing to be able to do with a workflow is to run it. Select the Run workflow... option from the file menu. If your workflow requires input, you will be presented with a new window - this is the place to enter the starting data on which the workflow is to run.

In this case the workflow has a single input called 'pdbID'. Select *pdbID* and click 'New string'. You should see the string "abcd". Double click "abcd". In the area that has just appeared you should replace "abcd" with the input term, so try entering '1crn'. Once you have entered this input data you can press the 'Launch Workflow' button at the top of this window to actually start the workflow engine off on the supplied input.

Results Perspective

Taverna will switch to the results perspective

The workflow run will be displayed in the monitor view. This window shows you the progress of the workflow - running services are shown green and completed ones in grey.

As the results are produced they are listed, for each output, in the port results view

This particular workflow has just one outputs called 'pdbFlatFile'. Select the only available item in the list on the left hand in the 'pdbFlatFile' tab of the port results view. You should see the contents of the file in the rendered results.

Congratulations, you have run your first workflow.

Closing the Workflow

To close the current workflow, select "Close workflow" from the file menu.

5 Overview

There are by default two views (called perspectives) that Taverna 2 can show of the creation and running of a workflow.

- The design perspective where you can open, edit and configure your workflows
- The results perspective where you can see the results of running your current workflow and browse the results of previous runs

Both perspectives share the top-level menus, the top-level icons and the perspectives tab.

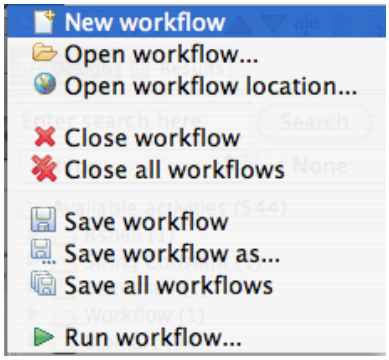
To switch between perspectives select D*esign* or R*esults* as appropriate in the perspectives tab.

5.1 Top Level Menus

The top level menus in Taverna 2 are:

- the File menu
- the Edit menu
- the Activities menu
- the Workflows menu
- the Advanced menu
- the Help option

5.1.1 File Menu



The *File Menu* allows you to:

- New workflow - create a new workflow

- Open workflow... - open a workflow that is on your file system
- Open workflow location... - open a workflow that is located on the web

Taverna 2.0 is able to open most Taverna 1 workflows and translate them into Taverna 2 workflows. However, in some cases there may be problems. Please see the myGrid website http://www.mygrid.org.uk for more details.

The above three options will make that workflow your current workflow which is available for editing and running.

- Close workflow - close the current workflow
- Close all workflows - close all the workflows that Taverna 2 has open

The above two options will prompt you to save the workflow if it has been edited.

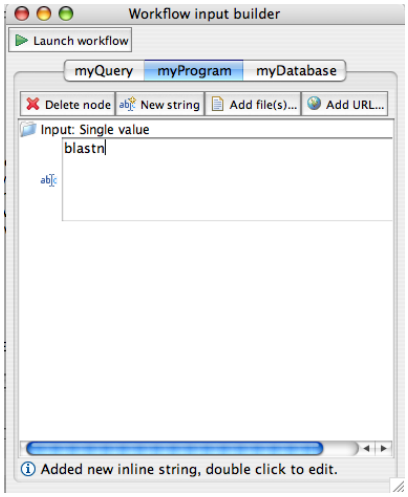
- Save workflow - save the current workflow (only available if the workflow has been edited)
- Save workflow as... - save the current workflow as a different file
- Save all workflows - save all the workflows that Taverna 2 has open and that have been edited

If the current workflow was loaded from the web then, then *Save workflow* will have the same effect as *Save workflow as*

- Run workflow - prompts for any inputs to the current workflow and allows you to launch a run of it
- Preferences - opens the preferences dialog (this may be under the main program menu on a Mac)
- Exit - exit Taverna

If workflows have been edited and not saved, then selecting *Exit* will also cause a *Save all workflows*.

5.1.1.1 Run workflow



The Run workflow dialog has a

- Launch workflow button that when clicked will run the workflow with the specified input values

When the workflow is launched, Taverna switches to the results perspective.

For each input port of the workflow, there is a tab containing the name of the input port. To enter an input value, or values for the port, click on the corresponding tab.

Initially the pane will show a *Input*. The Input indicates the depth of input values that must be specified. For example:

- if the port expects a single string, then *Single Value* is shown
- if the port expects a list of strings, then *List of depth 1* is shown
- if the port expects a list of list of list of strings, *List of depth 3* is shown

There are five types of node that can appear in the tree:

- the *Input* node at the top of the tree
- A *List* node that contains either list nodes or values
- A *String* node, indicated by "abcd" that contains an editable string value
- A reference to a file on your file system
- A URL to a file on the web

The possible operations on the input values are:

- *Delete node* - remove the corresponding string or list from the input values
- *New string* - creates a new String node that will be an input value. The new node initially contains the value "abcd", but this can be edited. The node for the string value is created at the required place in the tree. For example,

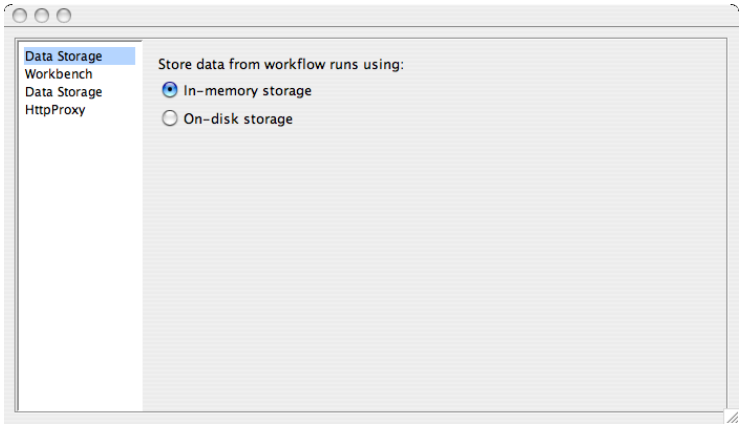
  - if you are on the *Input* node, the tree is empty and the required depth is three, then the string node will be placed within three containing list nodes
  - if you are on the *Input* node and the tree is not empty, then the new string node will be placed at the end of the list of strings in the first list nodes
  - if you are on a List node that is empty, then the new string node will be placed within the list node, possibly contained by several new list nodes
  - if you are on a List node that is not empty, then the new string node will be placed at the end of the list of strings in the first list nodes

- *Add file(s)...* - allows you to browse your file system and select a file or multiple files. The file or files will then be placed in the tree at the same position as a new String node, or new nodes, would appear. During the running of the workflow, the contents of the file(s) are used as the input value(s).
- *Add URL...* - opens a dialog in which you can enter a URL to a file on the web. When the workflow is run, the contents of the file at that URL are used as the input value.
- *Add List* - creates a new empty List node. This option is only available when the input port takes a list of list (of list...) of values. When you create the List node, you choose the required depth of the List node that is created. If you select a value less than the total depth of the input port, then the new List node is placed under other List nodes, using rules similar to that for New String.

5.1.1.1.1 Editing Input String

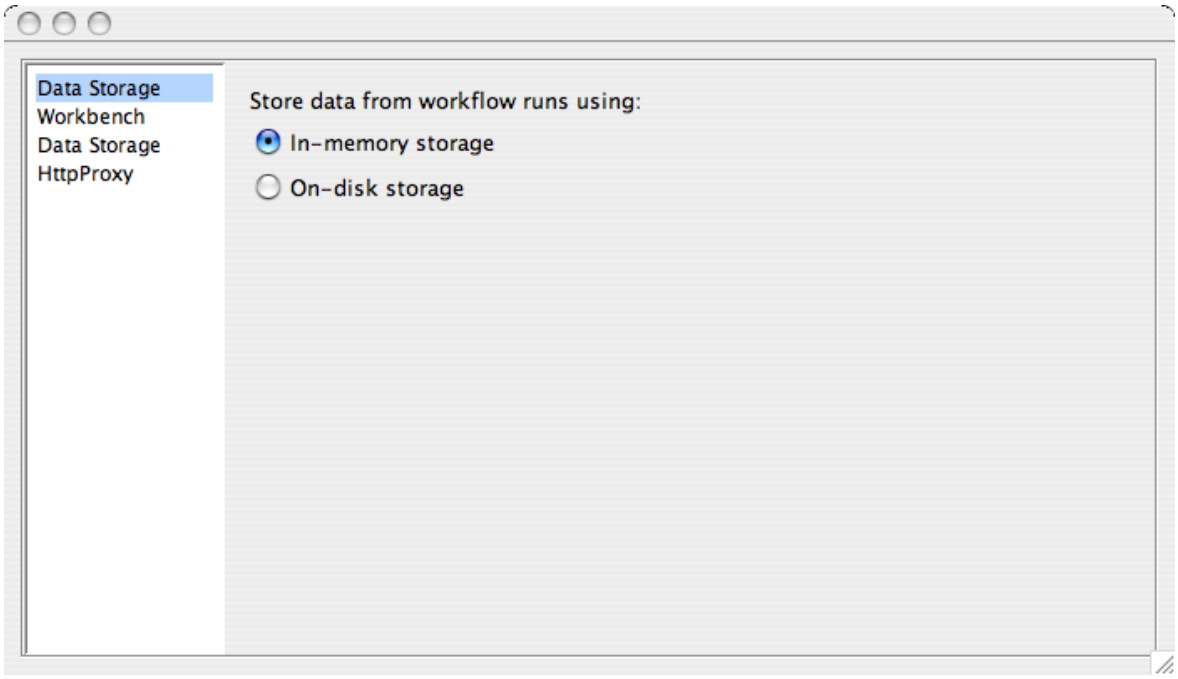
To edit an input string in the Run workflow dialog, click twice on the current string value. (The value is initially set to "abcd".) An area opens within the tree where you can type in the new value. When you have typed in your value, click outside the entry area. Do not type **return** unless you want that **return** as part of the input value.

5.1.1.2 Preferences



The preferences dialog allows you to specify the settings that Taverna 2 uses. To alter a setting, click the corresponding topic in the left hand menu.

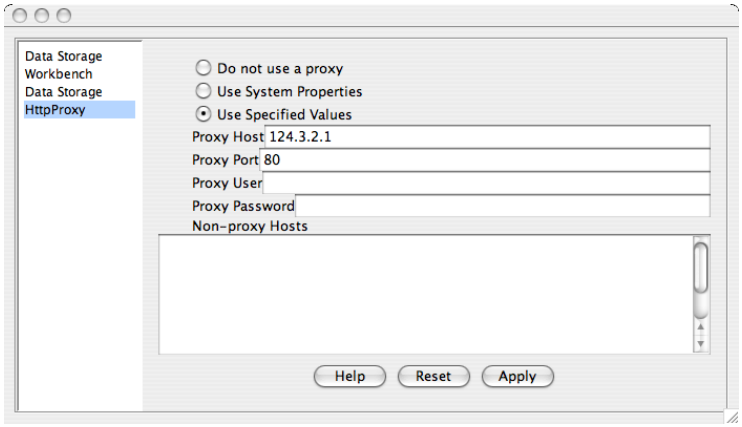
5.1.1.2.1 Data Storage Preferences



The data storage preferences specify how Taverna 2 stores data as it is passed between activities. The default options are:

- *In-memory storage* - Taverna 2 stores the data within the Java virtual machine. This, in general, is the fastest option but can cause memory-usage problems if the data being passed is large
- *On-disk storage* - Taverna 2 stores the data in a database within your file system. This is slower but allows Taverna 2 to handle much larger data

5.1.1.2.2 Http Proxy Preferences



The Http Proxy preferences specifies how Taverna should connect to URLs (addresses) on the web. You may need to use a HttpProxy if the machine on which you are running Taverna is behind a firewall. You can obtain details of the necessary settings from your system administrator or IT department.

There are three options:

- *Do not use a proxy* - this causes Taverna to ignore any system settings and to connect to URLs directly
- *Use System Properties* - use the settings that were specified to the Java virtual machine when Taverna was started. This is the default option and is usually correct
- *Use Specified Values* - use the values that you can enter or edit in the fields within the panel

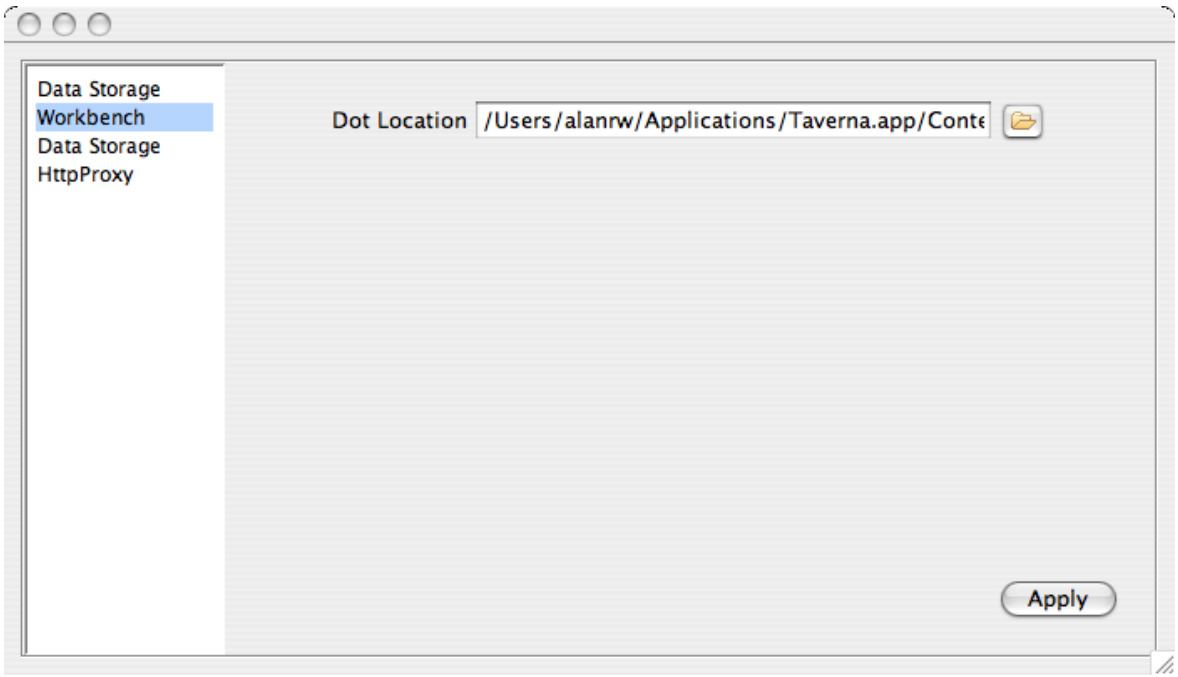
The panel lists the five values that can be altered for the HttpProxy settings. You are only able to edit them if you select the *Use Specified Values* option.

- *Proxy Host* - specifies the address of the machine via which your connections will be made. It can either be a numeric address e.g. 123.123.123.123 or an ordinary URL such as www.somewhere.com
- *Proxy Port* - the number of the port on the Proxy Host to which you make connection
- *Proxy User* and *Proxy Password* - if you need to authenticate (log in) to the Proxy Host then the username and password can be specified
- *Non-proxy Hosts* - A list of machine addresses for which the proxy settings will not be used. This is normally the list of machines on 'your side' of the firewall. The list is separated by '|' or newlines.

If you click *Help* then you will obtain help on this dialog. *Reset* resets the values to those before you started making any current changes. *Apply* takes the shown values.

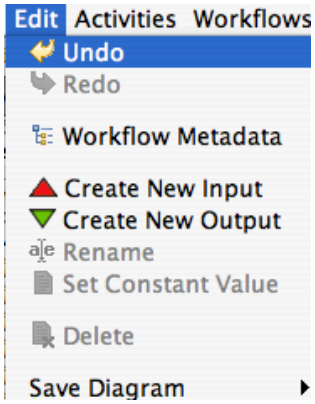
Note that the values are applied immediately and also for subsequent starts of Taverna. It is a good idea to restart Taverna once you have altered these settings as some existing connections may become unpredictable.

5.1.1.2.3 Workbench Preferences



The workbench preferences allows you to specify the location on your file system of the dot program that is used to draw the workflow. For MacOS and Windows installations, this location will be set automatically by Taverna 2 and should not be changed. For Linux installations, you will need to set the location before you can use the graphical features of Taverna 2.

5.1.2 Edit Menu



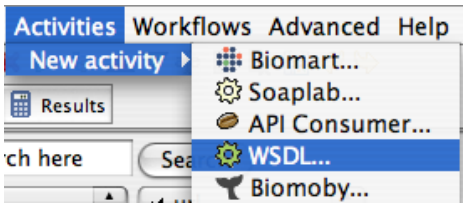
The *Edit Menu* allows you to

- undo or redo edits that you have made to your workflow. For example, if you added an input port and then deleted a data link then
    - *undo, undo* will restore the data link and remove the input port

    - *undo, redo* will have no effect
    - *undo, undo, redo* will restore the data link and keep the input port

- *Workflow Metadata* will bring up the Workflow Contextual View
- *Create New Input* port for the current workflow. Similar to graphical creation
- *Create New Output* port for the current workflow. Similar to graphical creation
- *Rename* or edit the currently selected workflow port or processor. Similar to graphical editing of an input port, output port or processor
- *Set Constant Value* for a processor's input port.
- *Delete* the currently selected workflow port, processor, data link or condition. Similar to graphical deletion of an input port, output port, processor, data link or condition.
- *Save Diagram* that brings up dialogs allowing the user to save the workflow diagram in a variety of graphical formats.

5.1.3 Activities Menu

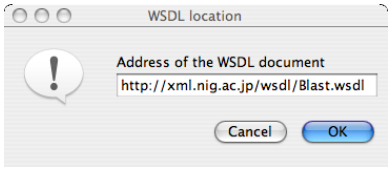


The Activities Menu allows you to make Taverna 2 aware of new sets of activities (services).

When you click on the menu and select the *New activity* item, you are able to choose which type of service you wish to add. The default possibilities are:

- WSDL... - specify the location of a WSDL file that describes the services and operations that will become available to Taverna 2
- Biomoby... - specify information about a BioMoby registry that contains services that will become available to Taverna 2
- Biomart... - specify the location of a BioMart whose queries will become available to Taverna 2 as activities
- Soaplab... - specify the location of a set of SoapLab services
- ApiConsumer... - identify the file that contains a description of APIConsumer services

5.1.3.1 New WSDL activity



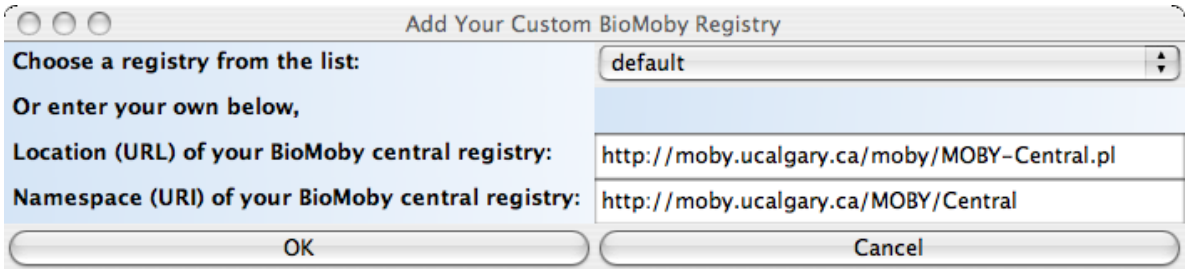
When you select *WSDL...* from the *New activity* item under the *Activities* menu, you are presented with a dialog.

In the dialog, type the URL address of the WSDL file that contains descriptions of the services and operations that you want to add to Taverna 2 as activities.

When you click *OK*, the WSDL file is parsed and the described operations are added into the Activity Palette.

If you click *Cancel* then no action is taken.

5.1.3.2 New Biomoby activity



When you select *Biomoby...* from the *New activity* item under the *Activities* menu, you are presented with a dialog.

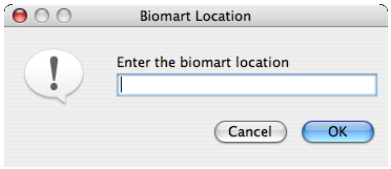
In the dialog you can select a BioMoby registry from the list of known registries in the drop-down menu. The location and namespace of that registry are displayed in the dialog.

In the future, you will be able to specify a new unknown registry and enter its location and BioMoby namespace.

When you click *Okay*, the BioMoby registry is contacted and the services within it are added as activities to the Activity Palette.

If you click *Cancel* then no action is taken.

5.1.3.3 New BioMart activity



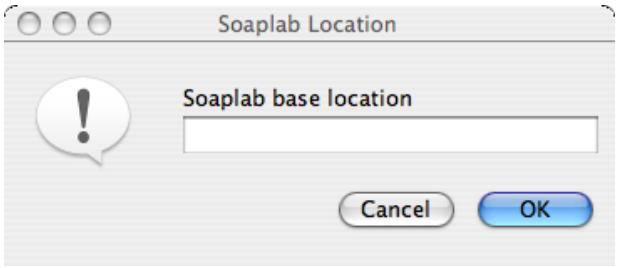
When you select *BioMart...* from the *New activity* item under the *Activities* menu, you are presented with a dialog.

In the dialog, type the URL address of the BioMart registry that allows queries that you want to add to Taverna 2 as activities.

When you click *OK*, the BioMart registry is contacted and its queries are added into the Activity Palette.

If you click *Cancel* then no action is taken.

5.1.3.4 New Soaplab activity



When you select *Soaplab...* from the *New activity* item under the *Activities* menu, you are presented with a dialog.

In the dialog, type the URL address of the Soaplab that contains descriptions of the Soaplab services that you want to add to Taverna 2 as activities.

When you click *OK*, the Soaplab services are added into the Activity Palette.

Note that Taverna 2 currently only supports Soaplab 1 services.

If you click *Cancel* then no action is taken.

5.1.3.5 New API Consumer activity

When you select *APIConsumer...* from the *New activity* item under the *Activities* menu, you are presented with a file selection dialog.

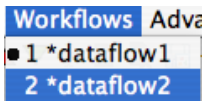
In the dialog you can choose a file on your local file system that contains XML describing APIConsumer services. The file describes the mapping between Java methods and activities.

When you select a file and click *Open*, the file is parsed and the services described within it are added as activities to the Activity Palette.

If you click *Cancel* then no action is taken.

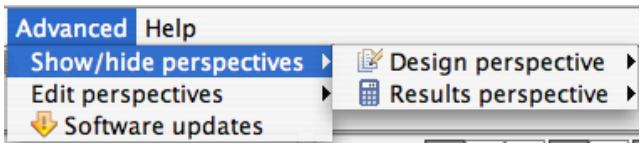
Note that any libraries, including the jar file that contains the Java methods that are being made available as activities, must be copied to Taverna 2's lib directory.

5.1.4 Workflows Menu



The *Workflows Menu* lists the workflows that are currently opened in Taverna 2. When you select a workflow from the list, that workflow becomes the current workflow. It is then displayed in the design perspective and can be edited or run.

5.1.5 Advanced menu



The Advanced menu by default contains three options:

- *Show/hide perspectives* - where you can specify the visibility of perspectives
- *Edit perspectives* - where you can create your own perspective
- *Software updates* - opens the plugin manager.

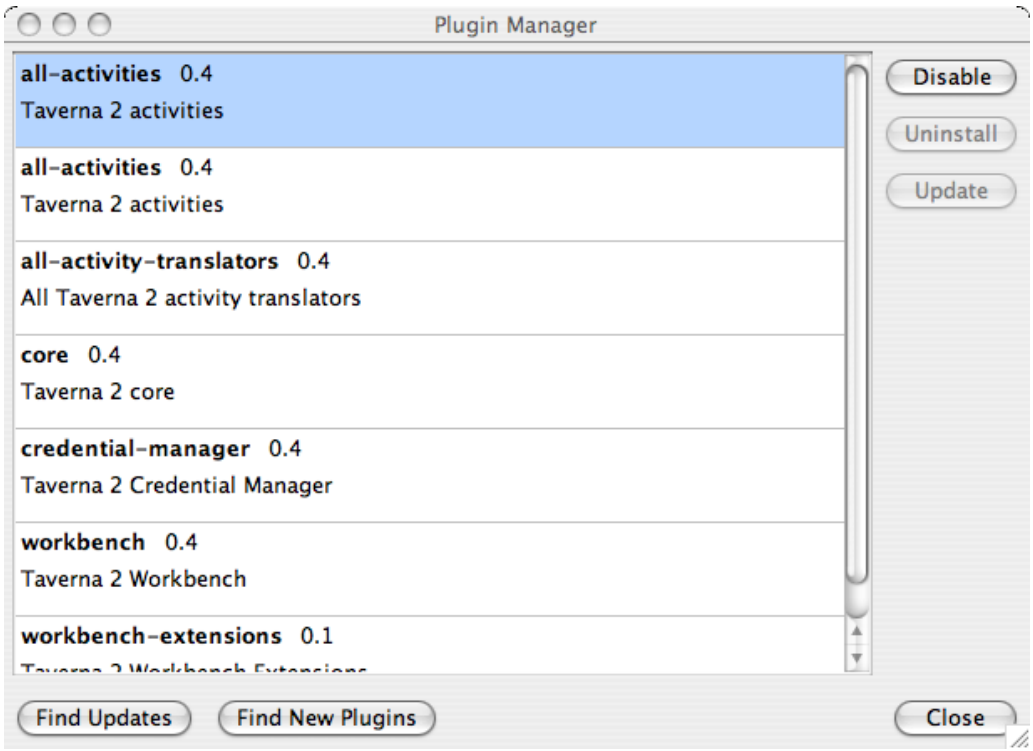
5.1.5.1 Show/hide perspectives

Not yet described

5.1.5.2 Edit perspectives

Not yet described

5.1.5.3 Plugin Manager



The *Plugin Manager* under the *Software updates* option of the *Advanced menu* allows you to:

- install updates to Taverna 2
- install new plugged-in functionality
- enable/disable already installed functionality

The main window of the *Plugin Manager* shows a list of the plugins installed into Taverna 2. If you select a plugin then you are able to:

- *Disable* the plugin - its functionality is no longer available
- *Uninstall* the plugin - it is no longer available unless you reinstall it
- *Update* the plugin - downloads a revised version of the plugin if it is available

The buttons on the bottom of the dialog allow you to:

- *Find Updates* - look for any updates available to the plugins you have installed
- *Find New Plugins* - look for new plugins either on the myGrid site or elsewhere
- Close

5.1.5.3.1 Plugin Sites

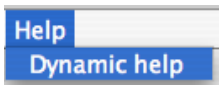
The *Plugin Sites* dialog is available from *Find New Plugins* in the *Plugin Manager*.

The dialog lists the sites from which you can download new plugins for Taverna 2 together with any new plugins available from that site.

You can:

- Select plugins (where available) and then click I*nstall* to add them into Taverna 2
- *Add Plugin Site* - opens a dialog where you can specify the URL of a plugin site and the name by which you want it to be known. If you click *OK* then the site is contacted for any plugins it specifies and the site with its plugins added to the shown list.

5.1.6 Help Option



If you select the *Dynamic Help* option under *Help* in the top level menus, or you press *F1* then it displays this help system.

5.2 Top-level icons

The top-level icons can be broken into two groups. Those corresponding to options under the File menu:

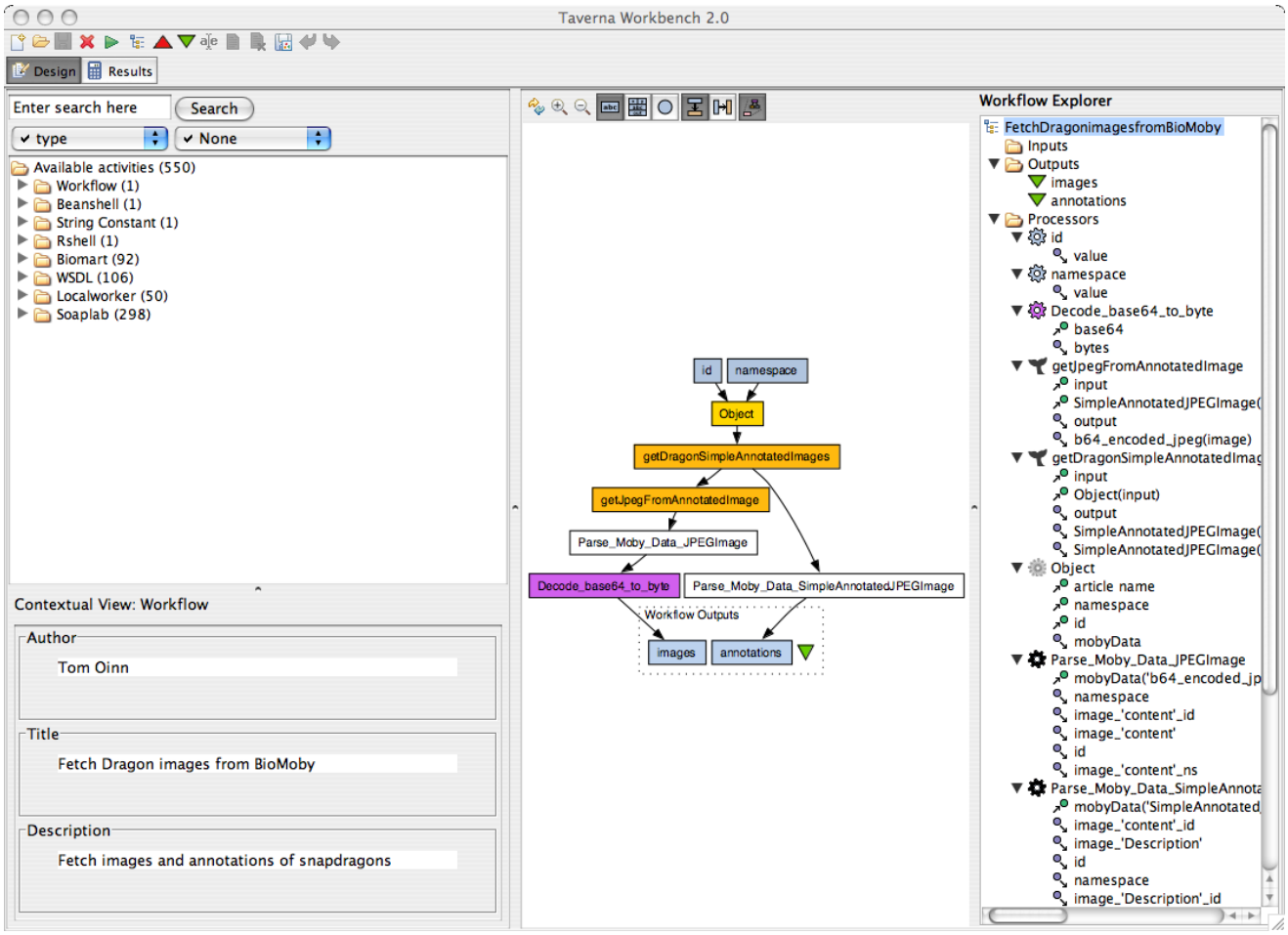
- - equivalent to *New Workflow*
- - equivalent to *Open Workflow...*
- - equivalent to *Save Workflow*
- - equivalent to *Close Workflow*
- - equivalent to *Run Workflow*

and those corresponding to options under the Edit menu:

- - equivalent to *Workflow Metadata*
- - equivalent to *Create New Input*
- - equivalent to *Create New Output*
- - equivalent to *Rename*

- - equivalent to *Set Constant Value*
- - equivalent to *Delete*
- - equivalent to *Save Diagram*
- - equivalent to *Undo*
- - equivalent to *Redo*

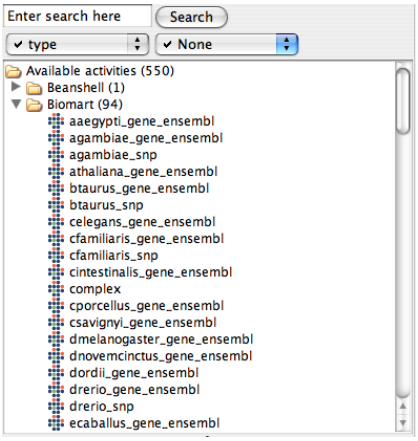
5.3 Design Perspective



The design perspective by default contains four areas:

- the Activity Palette - where you can browse the activities that are known to Taverna
- the Contextual View - that shows detailed information about an activity in the current workflow and may allow you to configure the activity
- the Graphical Editor - that allows you to edit the current workflow (if any)
- the Workflow Explorer - that gives a tree-like representation of the current workflow (if any)

5.3.1 Activity Palette



The activity palette shows in a tree-like representation the activities known to Taverna 2.

To add an activity to the current workflow, select the activity by clicking on it and drag it into a blank area of the Graphical Editor.

The tree is structured (except after a search) according to the values of properties associated with the activities with the root of the tree being all the *Available activities*. Those properties may be selected from the two pull-down menus, for example *type* or *url*.

The property selected in the first pull-down menu is used as the primary property to classify the activities. A partition node appears in the tree corresponding to the values that activities have for that property. Within each partition for the first property, the activities are further categorized according to the second property. Under the partition node for the first property, partition nodes appear for each value (within the group of activities) for the second property. If an activity does not have a value for the second property then *No value* is used. The activities appear as nodes under the partition nodes for the second property.

For example, if you choose *type* as the first property and *url* as the second property then all the WSDL activities will be grouped under a *WSDL* partition node. For each URL at which a WSDL activity is defined, there is a partition node e.g. **http://soap.genome.jp/KEGG.wsdl**. Under the partition node for each URL, the WSDL activities described by that URL are listed. The Localworker activities are grouped under a *Localworker* partition node and then they all appear under a *No value* partition node since Localworker activities are not associated with a URL.

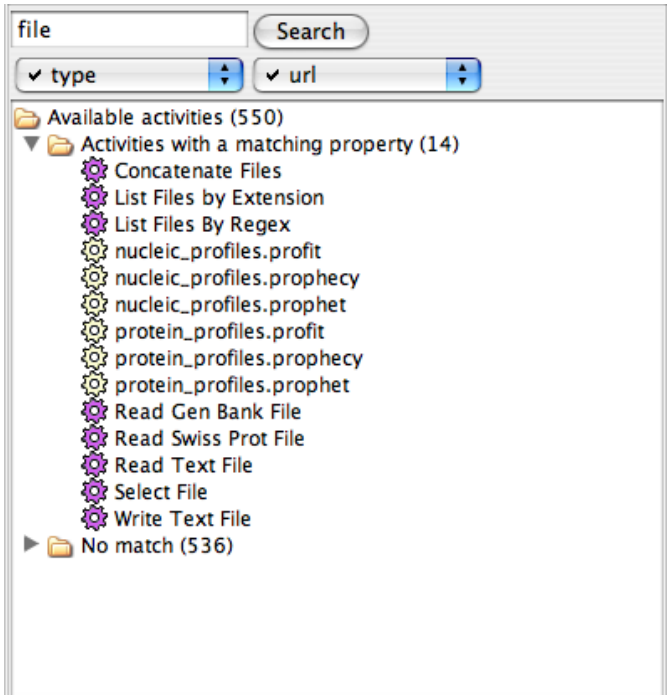
At each level of the tree the number of activities within that partition is given in parentheses. The total number of activities appears by the root *Available activities* node.

Most activities will have an associated icon that shows graphically the type of the activity.

The partitions of the tree can be expanded and collapsed by clicking the icon next to the node (sometimes + or -).

The activities can be searched.

5.3.1.1 Activity Palette Search



The activities within the Activity Palette can be searched. Type a string into the text box next to the *Search* button and either press **return** or click *Search*.

The activity tree will contain one or two intermediate nodes. Under *Activities with a matching property* are listed all the activities where some property value of the activity (it does not matter which property) contains the search string. If no activities have a property that matches the string, then there will not be an *Activities with a matching property* node. The number of activities with a matching property is shown in parentheses after the node.

The second node, *No match*, contains those activities that do not have a property that matches the search string. If no activities have been matched, then this node will contain all the activities known to Taverna 2. The number of activities without a matching property is shown in parentheses after the node.

To return to seeing the activities by category, it is currently necessary to select a property from one of the two pull-down menus.

5.3.1.2 Beanshell Activity Definition

A beanshell activity definition, if dragged into the Graphical Editor allows you to describe a beanshell that is used within the workflow.

When a beanshell activity instance has been included within a workflow it can be configured using the beanshell activity configuration.

5.3.1.3 Biomart Activity Definition

A Biomart activity definition, if dragged into the Graphical Editor corresponds to the ability to query a Biomart database and use the results of the query in your workflow.

The name of the Biomart activity within the Activity Palette gives an indication of the dataset that will be queried.

5.3.1.4 Biomoby Activity Definition

A Biomoby activity definition corresponds to a Biomoby service within a Biomoby registry. You can add new Biomoby services using the *Activities* menu.

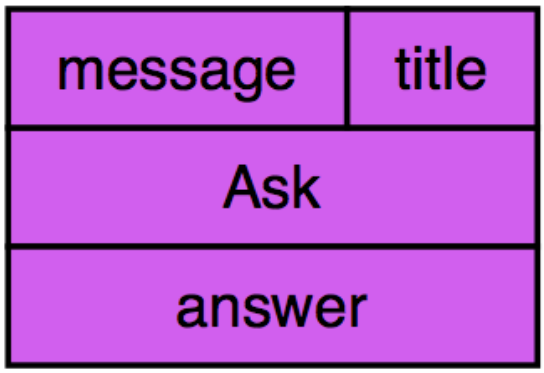
The name of the Biomoby activity within the Activity Palette corresponds to the name of the Biomoby service.

5.3.1.5 Local Worker Activity Definition

A local worker in the Activity Palette is a reusable beanshell that is built into Taverna. In the future, Taverna will allow a general capability for beanshell libraries.

An instance of a local worker, if dragged into the Graphical Editor can then be configured so that it meets your requirements.

5.3.1.5.1 Ask



This service displays a dialog with the specified title (if any). The dialog contains the specified prompt message (if any). The user is expected to enter a string. It returns the user's entered value. The service fails if the user cancels the dialog.

Input ports

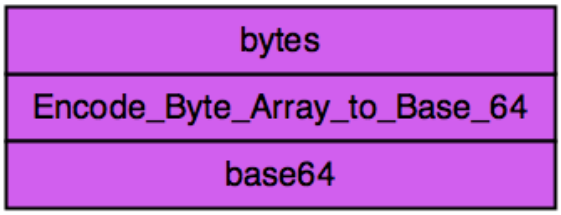
title
: The title used for the dialog. This is an optional parameter.

message
: The prompt message used for the dialog. This is an optional parameter.

Output ports

answer
: The user's entered value.

5.3.1.5.2 Byte[]_to_string



This service encodes the specified byte array as a string in the local character set.

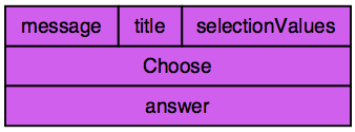
Input ports

bytes
: The byte array to be converted into a string. This is a mandatory parameter.

Output ports

string
: The string representing the byte array in the local character set.

5.3.1.5.3 Choose



This service displays a dialog wih the specified title (if any). The dialog contains the specified message (if any) and a list of values from which the user can choose. The choice defaults to the first element of the list. The valuelist cannot be empty. When the user clicks OK, the service outputs the chosen value to the answer port.

Input ports

selectionValues
: The list of values from which the user makes a choice. The list cannot be empty. This is a mandatory parameter.

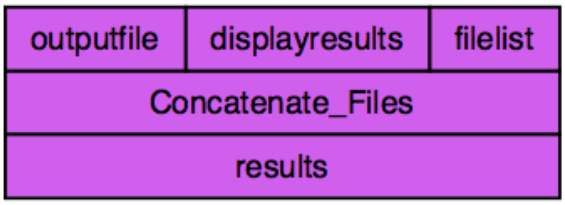
title
: The title used for the dialog. This is an optional parameter.

message
: The informative message used for the dialog. This is an optional parameter.

Output ports

answer
: The value that the user chose from the selectionValues.

5.3.1.5.4 Concatenate_files



This service examines the files whose paths or URLs are specified in the filelist. The content of those files is concatenated, separated by a newline, and the result written to the specified outputfile. If the displayresults port has a value 'true', then the concatenation is also echoed to the results port. If the displayresults port does not have a value 'true', then no connection should be made to the results port.

Input ports

filelist
: The possibly empty list of paths or URLs to the files to be concatenated. This is a mandatory parameter.

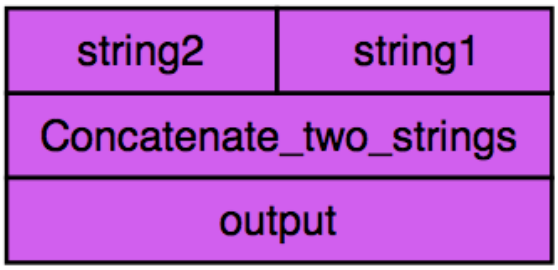
outputfile
: The path to the file that will contain the result of concatenating the files listed in filelist. This is a mandatory parameter.

displayresults
: A string that must contain a boolean value. If the string is 'true', then the result of concatenating the files is output to the results port. Even if the string is 'true', the result is always saved to the file specified by outputfile. This is an optional parameter.

Output ports

results
: If the displayresults port has a value of 'true', then results contains the string corresponding to the concatenation of the files within filelist. If the displayresults port does not have a value 'true', then no connection should be made to the results port.

5.3.1.5.5 Concatenate_two_strings



This service returns the string formed by concatenating its two input values. For example, if string1 is 'good' and string2 is 'bye', the output is 'goodbye'. If string1 and string2 are both '', then the output is ''.

Input ports

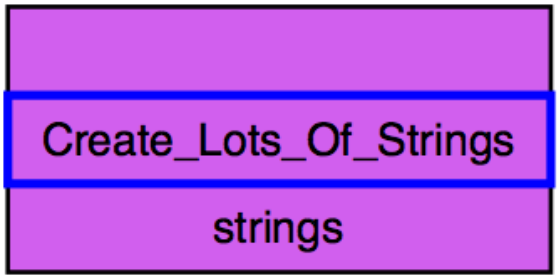
string1
: The string to appear first in the resultant concatenation. This parameter is mandatory.

string2
: The string to appear second in the resultance concatenation. This parameter is mandatory.

Output ports

output
: string1 concatenated with string2.

5.3.1.5.6 Create_lots_of_strings

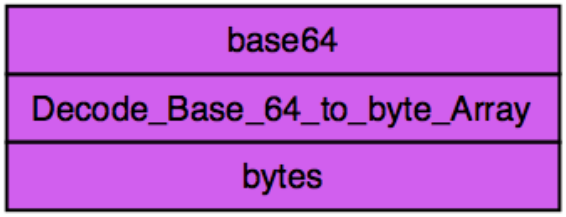


This service produces a list containing a large number of strings. It is normally used for testing workflows and other services.

Output ports

strings
: The list containing lots of strings..

5.3.1.5.7 Decode_base_to_byte[]



This service encodes the specified Base64 string to a byte array.

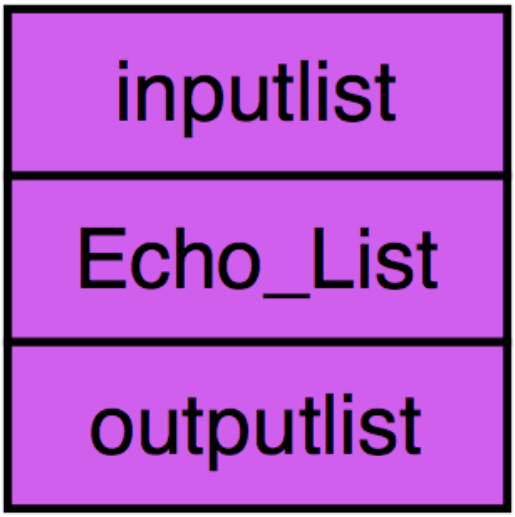
Input ports

base64
: The Base64 string to be converted to byte array. This is a mandatory parameter.

Output ports

bytes
: The byte array representation of the Base64 string.

5.3.1.5.8 Echo_list



This service echoes the input list to the output. It does not perform any processing on the list. This service is intended to be used in conjunction with nested workflows in order to split the iteration out from the previous stage of the workflow..

Input ports

inputlist
: The possibly empty list to be echoed. The list may be of any depth i.e. it can contain other lists as its elements. This is a mandatory parameter.

Output ports

outputlist
: A list containing the same elements in the same order as the inputlist..

5.3.1.5.9 Echo_with_occasional_failure

This service sometimes fails. It is intended to help test the robustness of workflows. If the service does not fail then the input is echoed to the output.

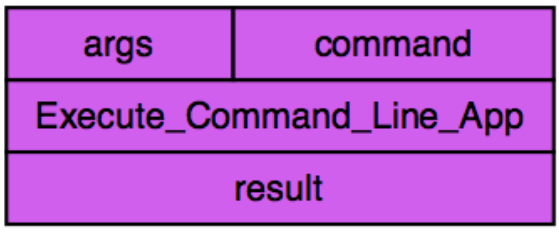
Input ports

in
: A string that is echoed to the out port if the service does not fail. This parameter is mandatory.

Output ports

out
: A copy of the input if the service does not fail.

5.3.1.5.10 Execute Command Line App



This service runs the specified command line application taking the list of given parameters (if any)). It returns the stdout result of the command.

Input ports

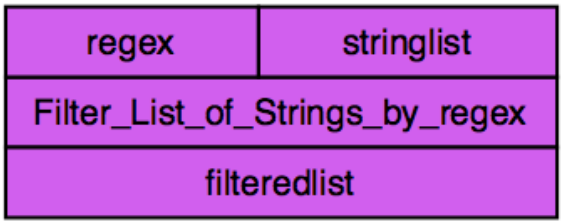
command
: The basic command to be run. This needs to include it s full path, for example '/bin/ls'. This is mandatory parameter.

parameters
: The list of parameters to the command (if any). Note that where a parameter takes a value that is separated by a space e.g. 'somecommand -x fred', '-x' and 'fred' should be given as two separate elements in the parameter list. This is an optional parameter.

Output ports

result
: The stdout result of running the command with the specified parameters. The result can be empty.

5.3.1.5.11 Filter_list_of_strings_by_regex



Filters a list of strings and returns a list containing, in order, the matched group, if any, according to Java match in the elements of stringlist

Input ports

stringlist
: The list of strings to be filtered. This is a mandatory parameter.

regex
: The regular expression that strings must match in order to be included in the filteredlist. The regular expresson must conform to that for java.util.regex. This is a mandatory parameter.

Output ports

filteredlist
: A, possibly empty, list of the elements of stringlist that match the regular expression within regex.

5.3.1.5.12 Filter_list_of_strings_extracting_match_to_a_regex

Filters a list of strings and returns a list containing, in order, the matched group, if any, according to Java match in the elements of stringlist

Input ports

stringlist
: The list of strings to be filtered. This is a mandatory parameter.

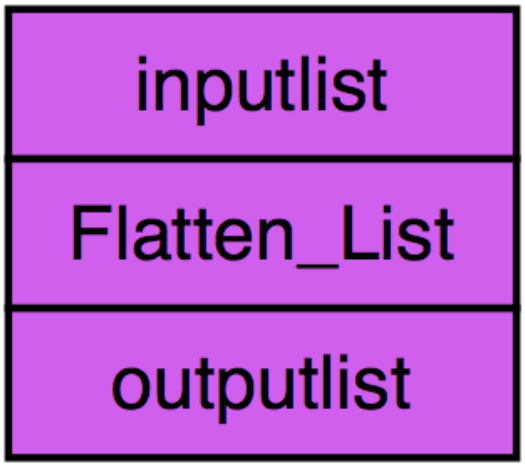
regex
: The regular expression that strings must match in order to be included in the filteredlist. The regular expresson must conform to that for java.util.regex. Only the specified group will be included in the output. This is a mandatory parameter.

group
: The number of the group in the regular expression that will be output to the filteredlis. The group should be a non-negative integer and is described in java.util.regex.Pattern. This is a mandatory parameter.

Output ports

filteredlist
: A, possibly empty, list of the specified group within the elements of stringlist that match the regular expression within regex.

5.3.1.5.13 Flatten list



This service flattens the inputlist by one level. It returns the result of the flattening. For example, given the input list [[1,2,3],[4,5,6],[[7]]] it will will return [1,2,3,4,5,6,[7]].

Input ports

inputlist
: The list of values to be one-level flattened. This is mandatory parameter.

Output ports

outputlist
: The result of one-level flattening the inputlist.

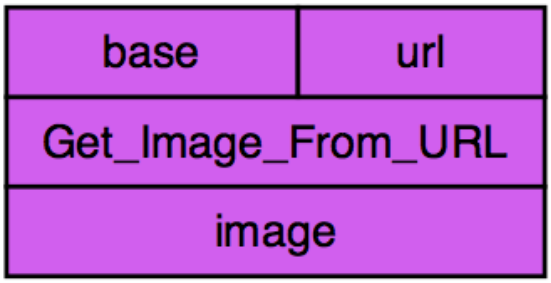
5.3.1.5.14 Get_environment_variables_as_XML

This service returns an XML document listing the environment variables and their values.

Output ports

properties
: The properties string is an XML document that contains a property-list element. Inside the property-list element, there is a property element for each environment variable. Each property element has two attributes, the 'name' of the property and its 'value'.

5.3.1.5.15 Get_image_from_URL



Returns the contents of the image at base (if specified) + url

Input ports

url

The URL of the image to be retrieved, prefixed by base if present. If base is not specified then this parameter must contain a full URL. This is a mandatory parameter.

base

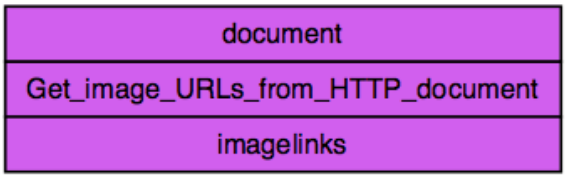
The base of the URL for the image to be retrieved. If specified, the value is prefixed to the url parameter. This is an optional parameter.

Output ports

contents

The contents of the image at the URL (base +) url. The image is returned as an array of bytes.

5.3.1.5.16 Get_image_URLs_from_HTTP_document



Returns the URLs of the images (if any) in the HTML document

Input ports

document

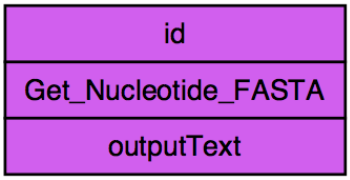
An HTML document. This is a mandatory parameter.

Output ports

imagelinks

A possibly empty list of the URLs to the images within the HTML document..

5.3.1.5.17 Get_Nucleotide_FASTA



Retrieves the nucleotide or protein sequence in fasta format associated to a given GI number or accession number.

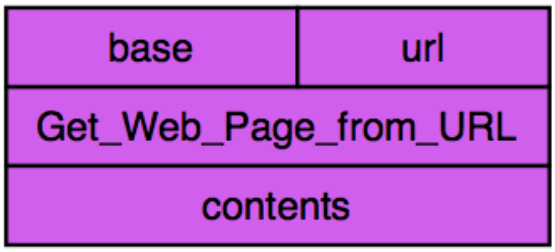
Input ports

id
: GI number or DDBJ/EMBL/SWISSPROT/TrEMBL accession number associated to a sequence. For example: GI--->18450186 | DDBJ--->AY066427 | EMBL--->A00993 | SWISSPROT--->Q09093

Output ports

outputText
: Returns a nucleotide or protein sequence in fasta format

5.3.1.5.18 Get_web_page_from_URL



Returns the contents of the webpage at base (if specified) + url

Input ports

url
: The URL of the web page to be retrieved, prefixed by base if present. If base is not specified then this parameter must contain a full URL. This is a mandatory parameter.

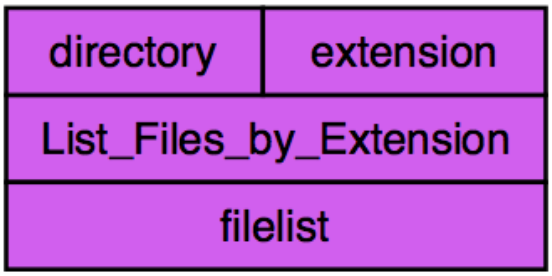
base

The base of the URL for the web page to be retrieved. If specified, the value is prefixed to the url parameter. This is an optional parameter.

Output ports

contents
: The contents of the webpage at the URL (base +) url. The contents of the web page are returned as either a plain String or as an HTML document. The format depends upon that of the source webpage..

5.3.1.5.19 List_files_by_extension



This service examines the names of the files within the specified directory. If a file has the specified extension, then the path to that file is included in the filelist output. The filelist output can be empty if no files within the directory have the specified extension.

Input ports

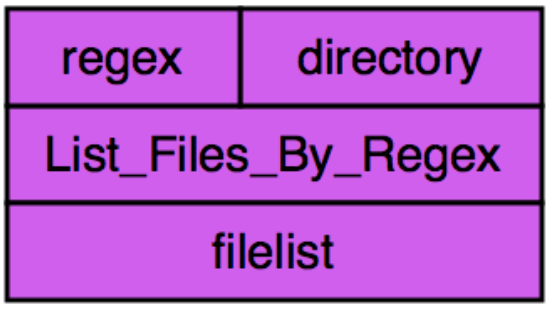
directory
: The fully qualified file path to the directory in which the files are to be found. This is a mandatory parameter.

extension
: The extension that files must have to be included in the output list. This is a mandatory parameter.

Output ports

filelist
: The, possibly empty, list of the paths to the files in the specified directory that have the specified extension.

5.3.1.5.20 List_files_by_regex



This service examines the names of the files within the specified directory. If a file a name that matches the specified regular expression (according to java.util.regex.Pattern.matches), then the path to that file is included in the filelist output. The filelist output can be empty if no files within the directory have names that match the specified regular expression.

Input ports

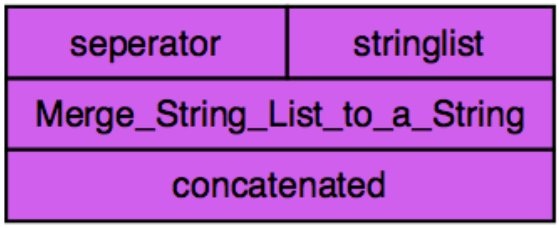
directory
: The fully qualified file path to the directory in which the files are to be found. This is a mandatory parameter.

regex
: The regular expression that the file names must match in order for the path to the file to be included in the output list. This is a mandatory parameter.

Output ports

filelist
: The, possibly empty, list of the paths to the files in the specified directory whose name matches the specified regular expression..

5.3.1.5.21 Merge_string_list_to_string



This service takes in a, possibly empty, list of strings and outputs the result of concatenating the strings, using the value of the seperator to separate them. The seperator defaults to newline. For example, the list ['a','b','c'] will cause output 'a b c'. The same list with seperator ':' will produce 'a:b:c'. If an empty list is given, then the output is an empty string.

Input ports

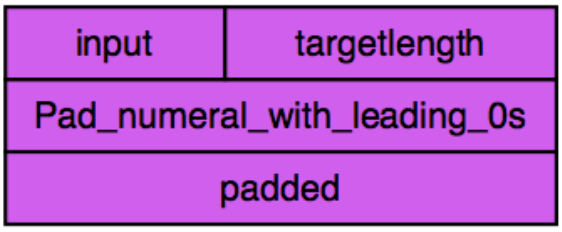
stringlist
: The possibly empty list whose strings are to be merged. This is a mandatory parameter.

seperator
: The string to be used to separate the strings with the stringlist. This is an optional parameter. The default value is newline.

Output ports

concatenated
: The string created by concatenating the elements of the stringlist separated by the seperator..

5.3.1.5.22 Pad_numeral_with_leading_s



Takes the input non-negative integer and returns a string containing it padded to the specified target length. The target length defaults to 7. If the input is already longer than the target length then it is not altered. For example, the input '12' with the default target length will yield '0000012'. The input '12' with target length '4' will yield '0012'. With target length '1' it yields '12'.

Input ports

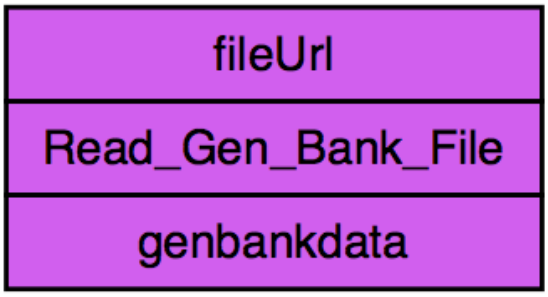
input
: A string containing the non-negative number to be padded. This is a mandatory parameter.

targetlength
: A string containing the strictly positive integer target length for the result. This is an optional parameter. It defaults to '7'.

Output ports

padded
: A string containing the input non-negative integer padded to the specified target length, or the original integer if it is longer than the target length.

5.3.1.5.23 Read_GenBank_file



The service reads the GenBank data held in the file whose path or URL is specified by fileUrl. Note that fileUrl can specify a file path or a URL. The service outputs the data to the genbankdata port in the form of an Agave XML document.

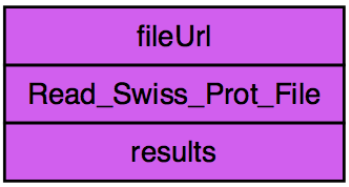
Input ports

fileUrl
: The path or URL to the file in which the GenBank data is held. Note that can be a file path or a URL. This is a mandatory parameter.

Output ports

genbankdata
: An XML document containing the GenBank data in Agave XML format.

5.3.1.5.24 Read_SwissProt_file



The service reads the SwissProt data held in the file whose path or URL is specified by fileUrl. Note that fileUrl can specify a file path or a URL. The service outputs the data to the genbankdata port in the form of an Agave XML document.

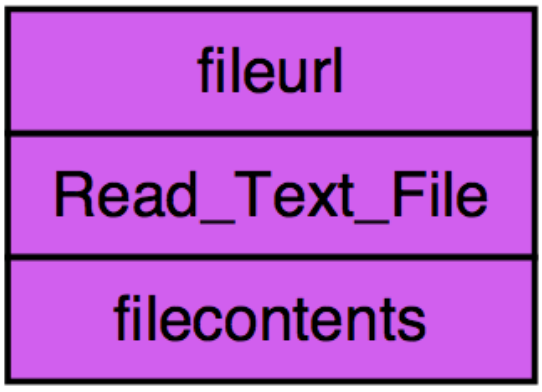
Input ports

fileUrl
: The path or URL to the file in which the SwissProt data is held. Note that this can be a file path or a URL. This is a mandatory parameter.

Output ports

genbankdata
: An XML document containing the SwissProt data in Agave XML format.

5.3.1.5.25 Read_text_file



This service reads the text file at the specified file path or URL and returns the file's content. The fileurl parameter can be given a fully qualified file path or a URL. The line separators present in the original file are replaced by line separators of the virtual Java machine in with Taverna is running.

Input ports

fileurl
: The fully qualified file path or URL to the file to be read. This is a mandatory parameter.

Output ports

filecontents
: The string containing the contents of the file whose filepath or URL is specified by the fileurl.

5.3.1.5.26 Remove_duplicate_strings

This service consumes a list of strings and outputs the string list with duplicate entries removed. The first occurrence of a duplicate is preserved and all subsequen ones omitted. For example, the list ['a','b','c','b','a',d'] gives the output ['a','b','c','d'].

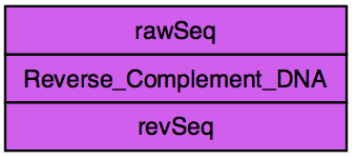
Input ports

stringlist
: The possibly empty list from which duplicate strings are to be removed. This is a mandatory parameter.

Output ports

strippedlist
: The stringlist with duplicate values removed.

5.3.1.5.27 Reverse_complement_DNA



The service reads a DNA sequence and outputs its reversed complement sequence.

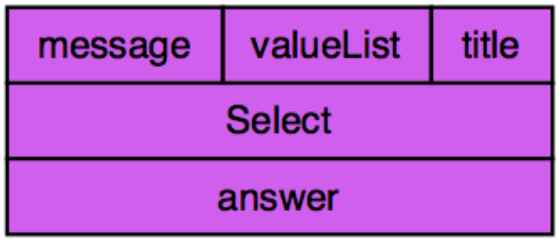
Input ports

raw_seq
: A string containing a DNA sequence in raw format i.e. just the letters of the sequence. This is a mandatory parameter.

Output ports

rev_seq
: A string containing the reverse complement of the DNA sequence in dna_seq.

5.3.1.5.28 Select



This service displays a dialog wih the specified title (if any). The dialog contains the specified message (if any) and a list of values from which the user can select. The selection defaults to the first element of the list. The valuelist cannot be empty. When the user clicks OK, the service outputs the selected value to the answer port. If the user clicks Cancel then a service failure is generated.

Input ports

valuelist
: The list of values from which the user makes a selection. The list cannot be empty. This is a mandatory parameter.

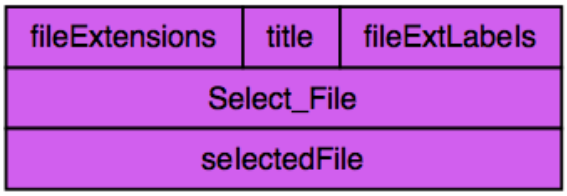
title
: The title used for the dialog. This is an optional parameter.

message
: The informative message used for the dialog. This is an optional parameter.

Output ports

answer
: The value that the user selected from the valuelist.

5.3.1.5.29 Select_file



This service displays a file selection dialog wih the specified title (if any) in which the user can browse the file system to select a file. The fileExtensions port specifies a comma separated list of the file extensions that the user can use to filter their view of the files. The fileExtLabels contains a comma separated list with the same number of elements. Each entry in fileExtLabels specifies the text by which the corresponding file extension will be displayed to the user. For example, if fileExtensions is 'bin,txt' then a sensible value for fileExtLabels is 'binary files, text files'. If no filtering of the files is to be made available then fileExtensions and fileExtLabels should be set to the empty string. The user always has the option to view all files. When the user clicks Open, the service outputs the path to the selected file to the selectedFile port. If the user clicks Cancel then an exception is generated.

Input ports

title
: The title used for the dialog. This is an optional parameter.

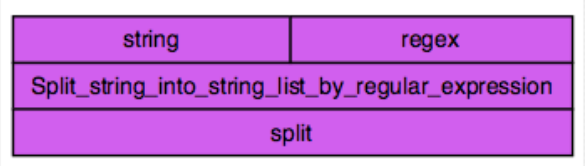
fileExtensions
: A comma separated list of the file extensions by which the shown files can be filtered. If no filtering is allowed then an empty string should be specified. This is a mandatory parameter.

fileExtLabels
: A comma separated list of the text used to describe the file extensions. The number of entries must be the same as for fileExtensions. If no filtering is allowed then an empty string should be specified. This is a mandatory parameter.

Output ports

selectedFile
: The path of the file that the user selected.

5.3.1.5.30 Split_string_into_string_list_by_regular_expression



Splits the string value into a string list at every occurrence of the specified regular expression. For example, splitting 'boo:and:foo' with ':' returns the list ['boo','and','foo']. Splitting it with 'o' returns the list ['b','',':and:f']. The regular expression defaults to ',' so splitting 'a,b,c' returns ['a','b','c']. See java.lang.String.split

Input ports

string
: The string that is to be split. This is a mandatory parameter.

regex
: The regular expression that is used to split the string. This is an optional parameter. It defaults to ','..

Output ports

split
: A list of the parts of string formed by splitting it by regex.

5.3.1.5.31 String_list_difference

This service returns a, possibly empty, list of those strings that are in one of the input lists but not in both. For example, the input parameters ['a','b','c'] and ['a','e','c'] will return ['b','e']. The input values ['a,'b','c'] and ['a','c','b'] will return an empty list []. Note that the order of elements in the returned list should not be relied upon.

Input ports

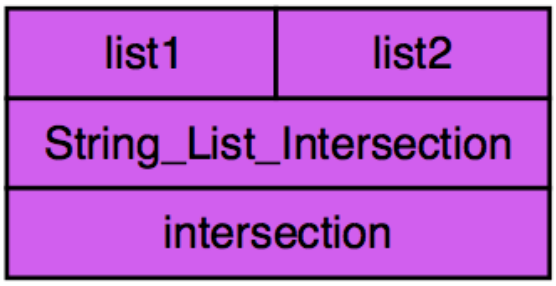
list1
: A list of strings that will be differenced with those in list2. This parameter is mandatory.

list2
: A list of strings that will be differenced with those in list1. This parameter is mandatory.

Output ports

difference
: A, possibly empty, list of those strings that are in list1 or in list2 but not in both.

5.3.1.5.32 String_list_intersection



This service returns a, possibly empty, list of those strings that are in both of the input lists. For example, the input parameters ['a','b','c'] and ['a','e','c'] will return ['a','c']. The input values ['a,'b','c'] and ['x','y','z'] will return an empty list []. Note that the order of elements in the returned list should not be relied upon.

Input ports

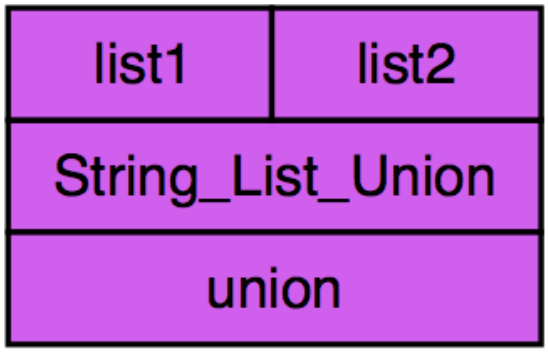
list1
: A list of strings that will be intersected with those in list2. This parameter is mandatory.

list2
: A list of strings that will be intersected with those in list1. This parameter is mandatory.

Output ports

intersection
: A, possibly empty, list of those strings that are both in list1 and in list2.

5.3.1.5.33 String_list_union



This service returns a, possibly empty, list of those strings that are in one of the input lists or in both. For example, the input parameters ['a','b','c'] and ['a','e','c'] will return a list whose elements are in the set ['a','b','c','e']. The input values [] and [] will return an empty list []. Note that the order of elements in the returned list should not be relied upon.

Input ports

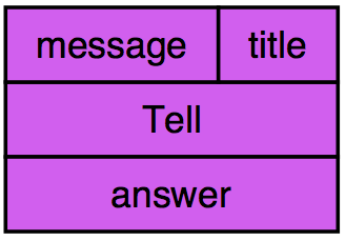
list1
: A list of strings that will be unioned with those in list2. This parameter is mandatory.

list2
: A list of strings that will be unioned with those in list1. This parameter is mandatory.

Output ports

union
: A, possibly empty, list of those strings that are in list1 or in list2 or in both.

5.3.1.5.34 Tell



This service displays a informative dialog with the specified title (if any). The dialog contains the specified message (if any). When the user clicks OK, the service outputs the string 'answer' to the answer port.

Input ports

title
: The title used for the dialog. This is an optional parameter.

message
: The informative message used for the dialog. This is an optional parameter.

Output ports

answer
: The string 'answer'.

5.3.1.5.35 Test_always_fails

This service always fails. The input parameters are both optional and are not used. No output value is ever generated.

Input ports

foo

The value of this parameter is not used. This parameter is optional.

bar

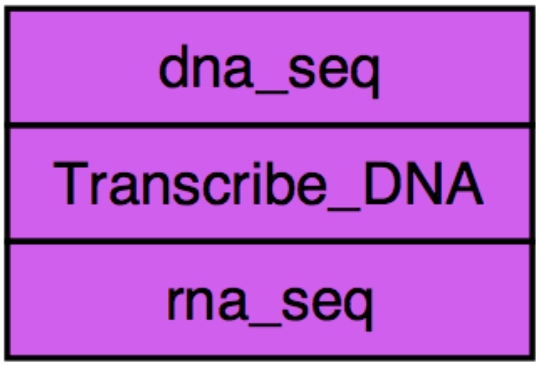
The value of this parameter is not used. This parameter is optional.

Output ports

urgle

No value for this output is ever generated.

5.3.1.5.36 Transcribe_DNA



The service reads a DNA sequence and outputs the transcribed RNA sequence.

Input ports

dna_seq

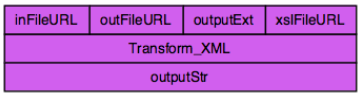
A string containing a DNA sequence in raw format i.e. just the letters of the sequence. This is a mandatory parameter.

Output ports

rna_seq

A string containing the transcribed RNA sequence corresponding to the dna_seq.

5.3.1.5.37 Transform_XML



The service opens an XML file whose path or URL is specified by the inFileURL. The XML document contained in that file is transformed by the XSLT file at the file path or URL specified by by xslFileURL. If outFileURL is specified then the result of the transformation is written to a file whose path or URL is specified by outFileURL. If outFileURL is not specified or is the empty string '', but outputExt is specified, the transformed document is output to the file location specified by inFileURL with the input file's extension replacedby outputStr. Note that if outFileURL is specified and is not the empty string '', then the specification of outputExt has no effect. Regardless of the values of outFileURL and outputExt, the result of the transformation is always output as a string to outputStr.

Input ports

inFileURL
: A string containing the file path or URL to the file that contains the XML document to be transformed. inFileURL takes a file path or a URL. This is a mandatory parameter.

xslFileURL
: A string containing the file path or URL to the file that contains the XSLT document that specifies the transformation to be performed. xslFileURL takes a file path or a URL. This is a mandatory parameter.

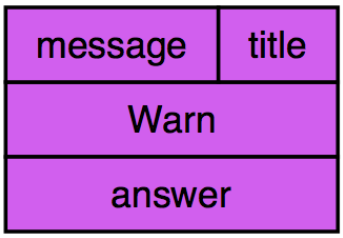
outFileURL
: A string containing the file path or URL to the file that will contain the result of the transformation. outFileURL takes a file path or a URL. This is an optional parameter.

outputExt
: A string containing a file extension. If outFileURL is not specified or is the empty string '', then this string is used to replace the original extension of the inFileURL and that file path or URL is used as the output file. Note that this parameter has no effect if outFileURL is specified and not the empty string ''. This is an optional parameter.

Output ports

outputStr
: A string containing the XML document that is the result of the transformation..

5.3.1.5.38 Warn



This service displays a warning dialog with the specified title (if any). The dialog contains the specified message (if any). When the user clicks OK, the service outputs the string 'answer' to the answer port.

Input ports

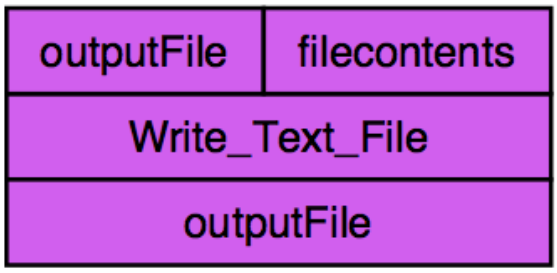
title
: The title used for the dialog. This is an optional parameter.

message
: The warning message used for the dialog. This is an optional parameter.

Output ports

answer
: The string 'answer'.

5.3.1.5.39 Write_text_file



This service writes the specified filecontents to the text file at the specified outputFile path. The new contents of the file are also written to the outputFile output. The original content of the file, if any, are overwritten.

Input ports

outputFile
: The fully qualified file path to the file to be written. This is a mandatory parameter.

filecontents
: The string to be written to the specified file. This is a mandatory parameter.

Output ports

outputFile
: The string containing the contents of the file whose filepath is specified by the outputFile input port. This should be equivalent to the passed filecontents.

5.3.1.6 Nested Workflow Activity Definition

The Nested Workflow Activity Definition, if dragged into the Graphical Editor allows you to describe include another workflow as an activity within the current workflow.

When a nested workflow activity instance has been included within a workflow it can be configured within the nested workflow contextual view. The configuration allows you to specify which workflow is nested.

5.3.1.7 Partition

A partition is a subset of the activity definitions within the Activity Palette. It contains either other partitions or leaf activity definitions. The activity definitions within the partition share the same value for a property selected in the Activity Palette. The number of activity definitions within the partition is given in parentheses.

5.3.1.8 Rshell Activity Definition

A Rshell activity definition, if dragged into the Graphical Editor allows you to describe a Rshell script that is used within the workflow.

When a Rshell activity instance has been included within a workflow it can be configured using the Rshell activity configuration.

5.3.1.9 Soaplab Activity Definition

A Soaplab Activity Definition corresponds to a Soaplab 1 service. If the activity is dragged into the Graphical Editor, then a call to the corresponding Soaplab service is included within current workflow.

Taverna currently only supports Soaplab 1 services although support for Soaplab 2 is being considered.

5.3.1.10 String Constant Activity Definition

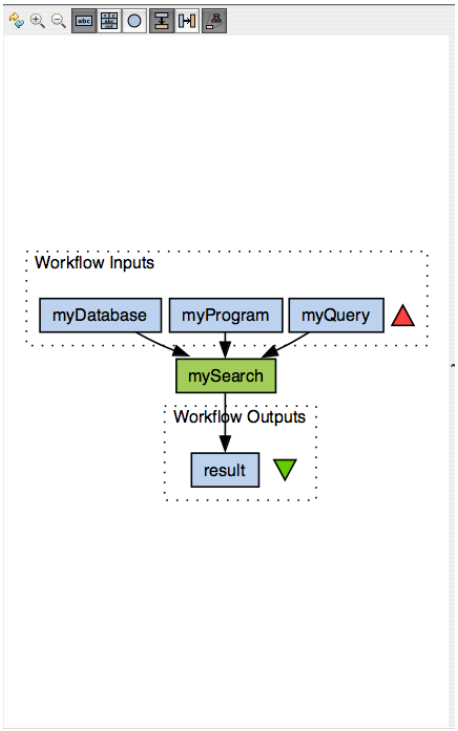
A String Constnt activity definition, if dragged into the Graphical Editor allows you to describe a string constant that is used within the workflow.

When a string constant activity instance has been included within a workflow it can be configured using the string constant configuration.

5.3.1.11 WSDL Activity Definition

A WSDL Activity Definition corresponds to an operation within a WSDL service. If the activity is dragged into the Graphical Editor, then a call to the WSDL operation is included within current workflow.

5.3.2 Graphical Editor



The Graphical Editor displays an editable view of the current workflow.

At the top of editor are a set of icons. These can be divided into:

- the Graphic Operation Icons - general operations on the graphics of the diagram
- the Processor Display Icons - control how processors (which contain an instance of an activity) are displayed
- the Workflow Display Icons - control how the overall workflow is display

Within the diagram representing the workflow you can create and edit the objects in the workflow. The objects that can be manipulated are:

- the current workflow - clicking in an empty area of the diagram will show the Workflow Contextual View.
- workflow input ports - representing where values will enter the workflow when run
- workflow output ports - representing where results will be output by a workflow run
- workflow processors - representing instances of activities within a workflow
- nested workflows - a special type of processor representing the use of another workflow within the current workflow
- data links - showing how data will flow between processors and to/from workflow ports
- condition - showing how the execution of one processor can depend upon that of another
- merges - showing how data flowing into the same port is merged

5.3.2.1 Graphical Operation Icons

There are three graphical operation icons within the Graphical Editor.

Reset Diagram Icon

Clicking on the Reset Diagram Icon resets the diagram. All zooming and panning is undone. The processor and workflow display options are not altered.

Zoom In Icon

Clicking on the Zoom In Icon zooms the diagram towards its centre.

Zoom Out Icon

Clicking on the Zoom Out Icon zooms out from the diagram.

Panning

Panning is not represented by an icon. In order to pan the diagram, hold down shift and drag the diagram with the mouse (using the left-button where appropriate). The diagram will also automatically pan if you are trying to make a link between two components, simply move the mouse past the edge of the Graphical Editor.

5.3.2.2 Processor Display Icons

There are three processor display icons within the Graphical Editor:

- *Display no processor ports* - when clicked this redraws the diagram so that none of the ports of the processors are shown
- *Display all processor ports* - when clicked the diagram shows all the ports of all the processors
- *Display processors as circles* - shows a simplified representation of the workflow where all the processors and ports of the workflow are shown as circles

If you have chosen *Display no processor ports* or *Display all processor ports*, the choice can be overridden for specific processors. However, when you next click a processor display icon, the overrides will be removed.

5.3.2.3 Workflow Display Icons

There are three Workflow Display Icons within the Graphical Editor.

-  - *Align processors vertically* - when clicked this redraws the diagram so that values conceptually flow from top to bottom i.e. the workflow inputs are at the top, the workflow outputs at the bottom.
- - *Align processors horizontally* - when clicked this redraws the diagram so that values conceptually flow from left to right i.e. the workflow inputs are at the left, the workflow outputs at the right.
-  - *Expand nested workflows* - this operates modally. When clicked the first time it will cause all nested workflows to be shown in their expanded form i.e. show their internal processors and connections. The second time it will cause all nested workflows to be shown as just processors - a black box like form.

Although you can override whether a specific nested workflow is shown expanded, when you click *Expand nested workflows* the overrides are removed.

5.3.2.4 Graphical Workflow Input Port Operations

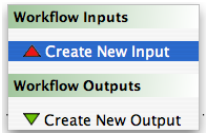
Workflow input ports represent where values will flow into a workflow when it is run. In the Graphical Editor they are shown as "pointed boxes" , unless in *Display processors as circles* mode where they are a red upward pointing triangle - .

The following operations are available for a workflow input port:

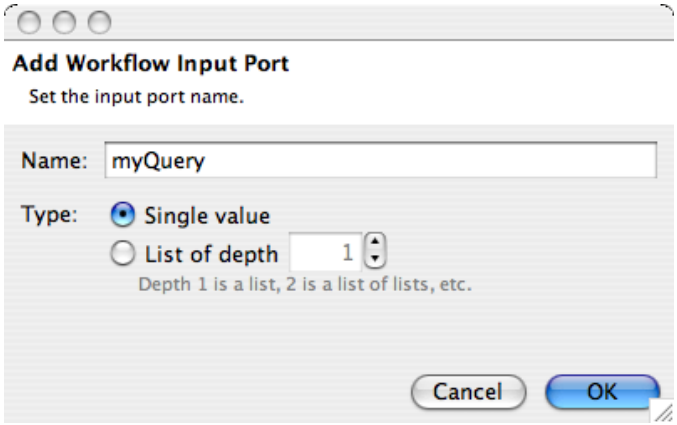
- creation
- editing
- deletion
- contextual viewing
- linking

5.3.2.4.1 Workflow Input Port Creation

To create a workflow input port, right-click in an empty area of the diagram and select *Create New Input...*



This brings up a dialog where you can enter the name of the input port and also its depth.



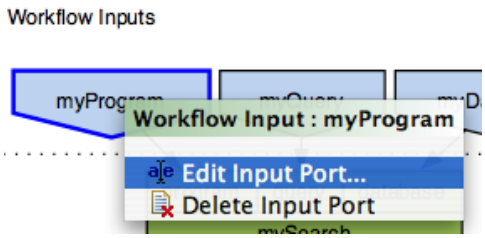
If you choose a depth of 0 it indicates that a single value should be entered when the workflow is run. A depth of 1 indicates a list of values; a depth of 3 a list of list of list of values etc. If you click *OK* then the diagram is redrawn showing the new input port.

The workflow input port creation dialog checks that the name of the input port does not duplicate any port already in the workflow.

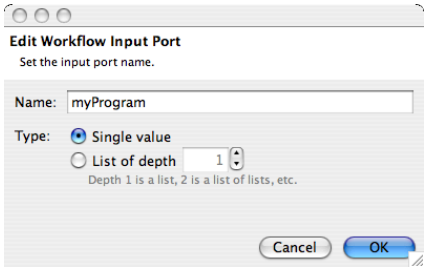
This operation can be undone/redone using the Edit Menu.

5.3.2.4.2 Workflow Input Port Edit

If you right-click on the representation of the workflow input port and select *Edit Input Port...*



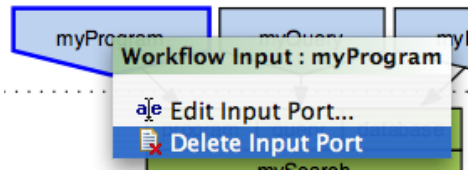
it brings up a dialog similar to that for Workflow Input Creation. The name and depth of the port can be altered.



This operation can be undone/redone using the Edit Menu.

5.3.2.4.3 Workflow Input Port Delete

If you right click on the representation of the workflow input port and select *Delete Input Port...* then the workflow input port is deleted.



This operation can be undone/redone using the Edit Menu.

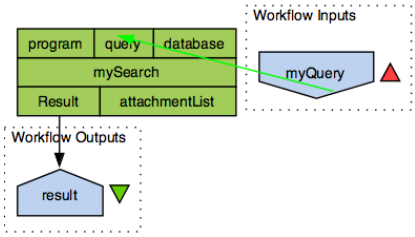
5.3.2.4.4 Workflow Input Port Context

If you left-click on the workflow input port then limited contextual information will be shown about it in the Contextual View

5.3.2.4.5 Workflow Input Port Link

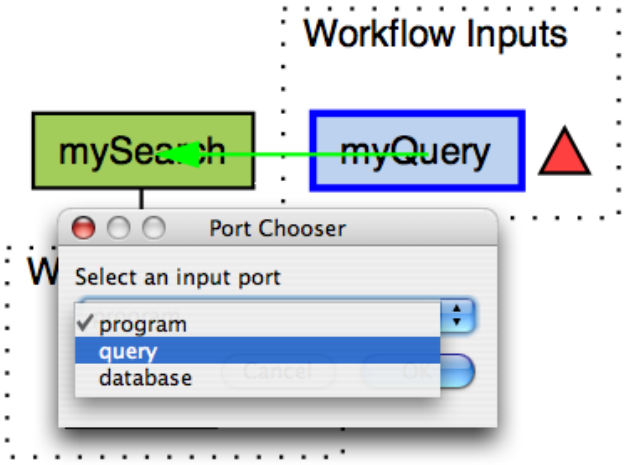
If you left click on a workflow input port, you can create a link by dragging the cursor

- to a processor input port if it is visible



or

- to a processor where you will be given a choice of input ports to connect to



This operation can be undone or redone using the Edit Menu.

5.3.2.5 Graphical Workflow Output Port Operations

Workflow output ports represent where values will flow out of workflow when it is run. In the Graphical Editor they are shown as "pointed boxes", unless in *Display processors as circles* mode, with a green downward pointing triangle - ▼.

The following operations are available for a workflow output port:

- creation
- editing
- deletion
- contextual viewing
- linking

5.3.2.5.1 Workflow Output Port Creation

To create a workflow output port, right-click in an empty area of the diagram and select *Create New Output...* This brings up a dialog where you can enter the name of the output port.

The workflow output port creation dialog checks that the name of the output port does not duplicate any port already in the workflow.

This operation can be undone or redone using the Edit Menu.

5.3.2.5.2 Workflow Output Port Edit

If you right-click on the representation of the workflow output port and select *Edit Output Port...* it brings up a dialog similar to that for Workflow Output Creation. The name of the port can be altered.

This operation can be undone or redone using the Edit Menu.

5.3.2.5.3 Workflow Output Port Delete

If you right click on the representation of the workflow output port and select *Delete Output Port...* then the workflow output port is deleted.

This operation can be undone or redone using the Edit Menu.

5.3.2.5.4 Workflow Output Port Context

If you left-click on the workflow output port then limited contextual information will be shown about it in the Contextual View

5.3.2.5.5 Workflow Output Port Link

If you left click on a workflow output port, then you can create a link to that port by dragging the cursor

- to a processor output port if it is visible or
- to a processor where you will be given a choice of output ports to connect from

This operation can be undone or redone using the Edit Menu.

5.3.2.6 Graphical Processor Operations

A processor within the Graphical Editor represents a container for an activity. In the diagram they are shown as rectangular boxes, unless in *Display processors as circles* mode, with the colour being dependent upon the type of the activity within the processor.

The following operations are available on a processor:

- creation
- rename
- deletion
- coordination from another processor
- showing/hiding of ports
- linking to a visible or invisible input port
- linking from a visible or invisible output port
- viewing of the corresponding activity's context
- viewing or editing of the processor's iteration strategy

5.3.2.6.1 Processor Creation

To create a processor, drag an activity from the Activity Palette into the diagram. The processor is assigned a name based upon the name of the activity. It defaults to copying the activity's name, but if a processor with that name already exists (commonly because the specific activity is being used by two processors), then the activity's name is suffixed by a number. For example, the first time the activity *Ask* is dragged into the diagram, a processor called *Ask* is created. The second time, the new processor is called *Ask0*; the seventh time *Ask6*.

This operation can be undone or redone using the Edit Menu.

5.3.2.6.2 Processor Rename

If you right-click on a processor and select Rename Processor..., a dialog is created where you can enter a new name for the processor. Taverna checks that the processor name does not conflict with the names of other processors within the workflow.

This operation can be undone or redone using the Edit Menu.

5.3.2.6.3 Processor Delete

If you right-click on a processor and select *Delete Processor*, then the processor is removed from the workflow. All links and conditions to and from the processor are removed.

This operation can be undone or redone using the Edit Menu.

5.3.2.6.4 Processor Coordinate

If you right-click on a processor and select Coordinate From, you will be given a choice of other processors within the workflow. If you select one of these processors, a condition link will be created.

This operation can be undone or redone using the Edit Menu.

5.3.2.6.5 Processor Show/Hide Ports

If you right-click on a processor, you can select *Show Ports* if the ports are not visible, or *Hide Ports* if they are. The ports of that specific processor will then be shown or hidden as appropriate.

This selection will be negated if you subsequently choose *Display No Processor Ports* or *Display All Processor Ports*.

This operation can be undone or redone using the Edit Menu.

5.3.2.6.6 Processor Input Port Link

If you left click on an input port of a processor, then you can create a link to that port by dragging the cursor

- to a processor output port (of another processor) if it is visible or
- to another processor where you will be given a choice of output ports to connect from, or
- to a workflow input port

This operation can be undone or redone using the Edit Menu.

5.3.2.6.7 Processor Link Input

If you right-click on a processor in the Graphical Editor and select "Link to input..." then you can select one of the input ports of the processor to create a link to. When you have selected a port drag the cursor

- to a processor output port (of another processor) if it is visible or
- to another processor where you will be given a choice of output ports to connect from, or
- to a workflow input port

If the processor only has one input port, then the menu will explicitly name that port e.g. "Link to myInput".

This operation can be undone or redone using the Edit Menu.

5.3.2.6.8 Processor Output Port Link

If you left click on an output port of a processor, then you can create a link from that port by dragging the cursor

- to a processor input port (of another processor) if it is visible or
- to another processor where you will be given a choice of input ports to connect to, or
- to a workflow output port

This operation can be undone or redone using the Edit Menu.

5.3.2.6.9 Processor Link Output

If you right-click on a processor in the Graphical Editor and select "Link from output..." then you can select one of the output ports of the processor to create a link from. When you have selected a port drag the cursor

- to a processor input port (of another processor) if it is visible or
- to another processor where you will be given a choice of input ports to connect to, or
- to a workflow output port

If the processor only has one output port, then the menu will explicitly name that port e.g. "Link from myOutput".

This operation can be undone or redone using the Edit Menu.

5.3.2.6.10 Processor Context

If you left-click on a processor, the Contextual View will change to show the context of the corresponding activity and possibly allow its configuration.

5.3.2.6.11 Processor Iteration Strategy

If you left-click on an input port of a processor then the Contextual View will contain a representation of the list handling strategy used for data on its inputs. The strategy can also be configured.

5.3.2.6.12 Processor Input Port Constant Value

When the input port of a processor is selected in the current workflow, then if no data link has been made to the port, it can be given a constant value, either by selecting *Set Constant Value* from the Edit menu or by clicking the equivalent top level icon.

Both bring up a dialog in which a String constant value for the port can be specified. If you click *OK* then an instance of a String Constant activity is inserted into the workflow with the specified value and connected to the input port.

5.3.2.7 Graphical Link Operations

A link, shown as a solid line with an arrow head, indicates that during a workflow run, values will flow from the source of the link to its target (where the arrow ends).

A link can be created by left-clicking and dragging from a:

- Workflow input port
- Workflow output port
- visible processor input port
- processor and selecting an input or output port
- visible processor output port

A link can be deleted.

If you attempt to create multiple links to the same port, then Taverna 2 will automatically create a merge.

5.3.2.7.1 Link Delete

If you select a link using left-click and then choose *Delete datalink* from the right-click menu, then the link will be deleted.

This operation can be undone or redone using the Edit Menu.

5.3.2.8 Graphical Condition Operations

A condition is shown by a pale line from one processor to another ending in a circle. It indicates that the target processor (at the circle end) can only start once the other processor has finished its execution.

A condition can be created by right-clicking on a processor.

A condition can be deleted.

5.3.2.8.1 Condition Delete

If you select a condition using left-click and then choose *Delete condition* from the right-click menu, then the condition will be deleted.

This operation can be undone or redone using the Edit Menu.

5.3.2.9 Graphical Merge Operations

A merge indicates that two sets of values are to be merged. Merges are represented in the Graphical Editor as light blue circles with a dark blue circumference. Taverna creates merges automatically if you create more than one link to the same processor input port or workflow output port.

A merge can be deleted.

5.3.2.9.1 Merge Delete

If you select a merge using left-click and then choose *Remove* merge from the right-click menu, then the merge will be deleted. All links to and from the merge are deleted.

This operation can be undone or redone using the Edit Menu.

5.3.2.10 Graphical Nested Workflow Operations

A nested workflow processor indicates that another workflow is used as a processor within the current workflow. Nested workflows are shown in the Graphical Editor as either pink rectangles (if "hidden") or as a workflow surrounded by black boundary.

Nested workflow processors are created by dragging a workflow processor from the Activity Palette into the diagram. To put a workflow into the nested workflow processor, you must click on it and use *Open from File* in its Contextual View.

When hidden, a nested workflow processor has an additional *Show Nested Workflow* option in the right-click menu. If you select this then the workflow will expand and you can see its contents.

When a nested workflow processor is expanded, only the following operations are available:

- rename
- delete
- hide

Even when a nested workflow processor is not hidden, you cannot edit its content when the current workflow in in the Graphical Editor. To edit the workflow associated with a nested workflow processor, you use *Edit workflow* in its Contextual View.

5.3.3 Contextual View

The Contextual View shows details of the activity (within the processor), workflow input port or workflow output port that is selected in the Graphical Editor.

For a workflow port, the Contextual View currently just displays the name of the port and its type (input or output).

The information displayed for an activity (within a processor) varies according to the type of the activity (WSDL, Soaplab etc.) In all cases, the Contextual View displays the type of the activity.

If an input port of a processor has been selected in the Graphical Editor then the Contextual View changes to show, and allow configuration of, the Iteration Strategy.

5.3.3.1 Condition Contextual View

The condition contextual view is shown when a condition is selected in the Graphical Editor. The contextual view shows the names of the two processors linked by the condition. A condition of the form:

```
Condition: A -> B
```

indicates that processor *B* will only run once processor *A* has finished its execution.

5.3.3.2 Link Contextual View

The link contextual view is shown when a data link is selected in the Graphical Editor. The link view shows the names of the source port and the target port of the link. A link of the form:

```
Datalink: A -> B
```

indicates that data from the output port *A* of the source processor will be passed to input port *B* of the target processor.

5.3.3.3 Merge Contextual View

The Merge Contextual View shows information about the selected merge within the current workflow.

5.3.3.4 Processor Contextual View

The processor contextual view consists of three areas:

- The processor-type specific contextual view that gives different information depending upon the type of the activity within the processor
- The List handling contextual view that describes how the processor handles data from lists on its input ports
- The Advanced contextual view that describes the looping, if any, that is done by the processor

5.3.3.4.1 APIConsumer Contextual View

Not yet described

5.3.3.4.1.1 APIConsumer Activity Configuration

Not yet described

5.3.3.4.2 Beanshell Contextual View

The beanshell contextual view shows information about a processor that contains a beanshell activity instance. The *Activity* shows the input and output ports of the beanshell.

An input port corresponds to a variable within the beanshell script that will, when the beanshell script is run, be given the value passed into the port

An output port corresponds to a variable in the beanshell script that after the script has run, the value of that variable will be passed out of the port

Initially a beanshell activity will not have ports.

The depth of the ports indicates the cardinality of the corresponding variables. A depth of 0 indicates that the variable has a singleton value, normally a String. A depth of 1 indiocates that it takes a List of values; a depth of 2 a List of List of values.

The script of the beanshell, its ports and its dependency on libraries can be specified by clicking the *Configure* button.

The *List handling* and *Advanced* possibilities are described in the processor contextual view.

5.3.3.4.2.1 Beanshell Activity Configuration

The beanshell activity configuration is used to give details of a beanshell activity instance.

There are three tabs within the configuration:

- The *Script* tab where you can enter the beanshell script that will be run within the workflow
- The *Ports* tab where you specify the input and output ports (under the two sub-tabs) of the processor for the beanshell. To add a port click *Add Port*. You can then type in the name of the port and select the depth of the corresponding variable in the script - 0 for a single value, 1 for a list, 3 for a list of list of list etc.

  A port that has been specified can be deleted by clicking *Remove* next to the port.
- The *Dependencies* tab where you can specify the external libraries (either as jar files or as Maven artifacts) upon which the script depends - normally indicated by import statements.

If you click *OK* then your configuration of the beanshell is saved. If you click *Cancel* then the configuration (that you have just made) is discarded. Note that cancelling cancels all of the configuration not just that on the currently shown tab.

5.3.3.4.3 Biomart Contextual View

The Biomart contextual view gives details of the selected Biomart activity within the current workflow. It shows:

- The URL (location on the web) of the service that is used to query the database
- The Location of the database e.g. its name and owner
- The Attributes of the database that are being queried. This will be missing if no attributes are queried.
- The name of the Dataset that is being queried

The attributes used for the query can be configured

Note that unlike most contextual views, the Biomart contextual view does not list the ports of the Biomart activity.

5.3.3.4.3.1 Biomart Activity Configuration

The Biomart activity configuration allows you to specify the values to be used when querying the appropriate Biomart database. The exact content of the dialog varies according to the database being queried.

Almost all configurations allow you to gain an approximate *Count* of the number of results that will be returned by the query, using the specified values. A large count will cause your workflow to take longer to run, and may also make it difficult to determine which results are actually relevant.

The *Unique results only* tickbox, when ticked, ensures that duplicate results are not returned. It is often advisable to tick this box.

The output ports of the activity are determined by the attributes that are returned by the query and also whether they are returned as seperate sets of values or in a single file.

Clicking *Cancel* will discard the configuration that you have just made of the Biomart activity. Clicking *OK* will save the configuration and use it when running the activity.

5.3.3.4.4 BioMoby Contextal View

The Biomoby contextual view gives details of the selected Biomoby activity within the workflow. It specifies:

- The *Endpoint* of the Biomoby service i.e. where the definition of it is located.
- The *Authority*
- The name of the *Service* within the Biomoby registry

The call to the Biomoby service can be configured

Note that unlike most contextual views, the Biomoby contextual view does not list the ports of the Biomoby activity.

5.3.3.4.4.1 Biomoby Activity Configuration

Not yet described

5.3.3.4.5 List Handling Contextual View

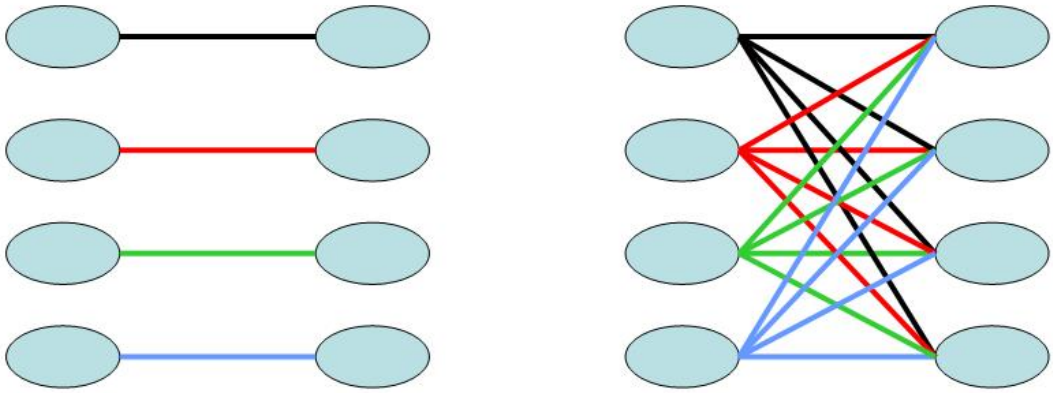
The List Handling Contextual View describes how calls are made of a processor to handle input data if more than one of its input ports is connected to a list and the port is expecting a single value.

Taverna has a built-in iteration system. If you provide a list of data items to a processor that expects one item then Taverna will automatcally feed each item to the processor in turn without the need for you to configure anything. However, if you have a processor that accepts two lists of data, you need to determine how Taverna should combine data items from those lists. There are two options:

Dot product
: item 1 of list 1 is combined with item 1 of list 2. Then item 2 of list 1 is combined with item 2 of list 2, and so on... matching rows in each list. Note that the number of items in the lists must be equal for dot product to work.

Cross product (the default option)
: this is an *all against all* match. Item 1 in list 1 is combined with all items in list 2 in turn, then item 2 of list 1 is combined with all items in list 2, and so on...

The following diagram illustrates the differences:



You can click *Configure list handling* to alter how Taverna iterates over the input data.

5.3.3.4.5.1 Iteration Strategy Configuration

The iteration strategy is used to specify how Taverna handles data when the cardinality of the values is not the same as specified for the input ports..

5.3.3.4.6 Advanced Contextual View

The Advanced Contextual View indicates the looping, if any, that a processor will perform.

If no looping is present then you can click *Add looping*.

Click *Configure* to specify how the processor will loop.

5.3.3.4.6.1 Loop Configuration

Loop Configuration controls how a single call of a processor loops (if at all). By default, the activity within the processor will only be called once.

Note that loop configuration does not control multiple calls of a processor. If a processor is expecting a single value on an input port, then looping controls how it deals with a single call with a single value. If in the containing workflow, a list of values is sent to the processor, then multiple calls of the processor are made, each call taking a single element from the list. The loop configuration does not deal with how the processor handles the list, since the processor only ever sees the single elements within the list.

You can specify that the call of the processor will loop until the value from one of its output ports (selected from a pull down menu) is *equal to*, *not equal to*, *matches* a regular expression, *does not match* a regular expression, *is greater than* or *is less than* a specified string value.

is greater than and *is less than* should only be used if the output port produces a string containing a numeric value

You can click *Customize* if you want to use a non-standard loop condition. This brings up a beanshell configuration where you can define your own beanshell script to evaluate the condition. The customized beanshell must always have a *loop* output port.

By default, all of the calls to the activity use the same input values and so in order to terminate the loop, the activity must be using or altering some internal data. However, it is possible to feed values from one or more output ports back into the input ports for the next call of the activity. In order to do this, check the *Feed back matching ports* box. The matching of the ports is done by port name and occurs for all identically named ports or for none.

5.3.3.4.7 Local Worker Contextual View

The local worker contextual view gives details of the input and output ports of a local worker within a workflow.

The local worker can be configured, for example to alter its script. This will change it to a user-defined local worker activity.

5.3.3.4.7.1 Local Worker Activity Configuration

Local worker activity configuration is used to specialise a local worker activity within a workflow. The configuration is identical to that of a beanshell.

If you alter the local worker and click *OK* then it will become a *user-defined local worker activity*.

5.3.3.4.8 Nested Workflow Contextual View

The Nested Workflow Contextual View shows the name of the workflow that is nested within the selected nested workflow activity.

The nested workflow can be edited by clicking the *Edit workflow* button. It will then be shown as the current workflow. When saved, it will be included within the enclosing workflow.

The workflow to be nested can also be specified by loading it from an external file by clicking *Open from file*.

In order to switch back to the 'main' workflow select it from the Workflows menu, or close the nested workflow.

Note that unlike most contextual views, the Biomoby contextual view does not list the ports of the Biomoby activity.

5.3.3.4.9 Rshell Contextual View

The Rshell contextual view shows information about a processor that contains a Rshell activity instance. The *Activity* shows the input and output ports of the Rshel.

An input port corresponds to a variable within the Rshell script that will, when the Rshell script is run, be given the value passed into the port

An output port corresponds to a variable in the Rshell script that after the script has run, the value of that variable will be passed out of the port

Initially a Rshell activity will not have ports.

The semantic type of the ports indicates the type of the corresponding variables. For example *boolean* indicates that the port corresponds to a Boolean variable.

The *Connection Settings* contains details of how the Rshell activity connects to the machine on which the Rshell script will be run.

The script of the Rshell, its ports and its connection settings can be specified by clicking the *Configure* button.

5.3.3.4.9.1 Rshell Activity Configuration

The Rshell activity configuration is used to specify details of a Rshell activity instance.

There are three tabs within the configuration:

- The *Script* tab where you can enter the Rshell script that will be run within the workflow. The script can be loaded from an external file using the *Load scipt* button. It can also be cleared using the *Clear script* button.
- The *Ports* tab where you specify the input and output ports (under the two sub-tabs) of the processor for the Rshell. To add a port click *Add Port*. You can then type in the name of the port and select the semantic type of the corresponding variable.

  A port that has been specified can be deleted by clicking *Remove* next to the port.
- The *Connection Settings* tab where you can specify the host on which the Rshell script will be run, the *Port* to connect to, and the *Username* and *Password* to log onto the host. In addition, you can choose to start a new R session each time (the default), or to keep a session alive. Keeping the session alive may be more efficient and allow you to share values beteween calls of Rshell scripts.

If you click *OK* then your configuration of the Rshell is saved. If you click *Cancel* then the configuration (that you have just made) is discarded. Note that cancelling cancels all of the configuration not just that on the currently shown tab.

5.3.3.4.10 Soaplab Contextual View

The Soaplab Contextual View shows information about the corresponding selected Soaplab activity within the current workflow.

Endpoint specifies the location of the Soaplab service.

Polling interval, *Polling backoff* and *Polling interval max* indicate the polling settings for the activity. They can be altered by pressing the *Configure* button.

Details of the ports of the Soaplab service are given in the *SOAPLAB Metadata*

5.3.3.4.10.1 Soaplab Activity Configuration

Soaplab Activity Configuration allows you to alter the way in which the service corresponding to the currently selected Soaplab activity is polled.

5.3.3.4.11 String Constant Contextual View

The String Constant Contextual View shows the string value associated with a selected String Constant activity within the current workflow.

The value of the String Constant can be changed by clicking *Configure*.

5.3.3.4.11.1 String Constant Activity Configuration

Within the String Constant Activity Configuration, you can enter the value to be associated with the corresponding String Constant activity instance.

5.3.3.4.12 WSDL Contextual View

The WSDL Contextual View shows information about the corresponding selected WSDL activity within the current workflow. *WSDL* specifies the location of the WSDL file that described the service and operation. *Operation* gives the name of the operation within the WSDL file. *Secured?* indicates whether security authorization is needed to call the corresponding WSDL operation.

For each input and output port of the activity, the name of the port is given, its depth (0 for a single value, 1 for a list of values, 3 for a list of list of list of values etc.) and the XML type of data flowing into/out of the port.

If an input port takes a complex XML structure rather than a simple value such as a String, the *Add input XML splitter* allows you to create an XML splitter activity that can be used to construct the necessary data values.

Similarly, if an output port produces data in a complex XML structure, the *Add output XML splitter* allows you to create an XML splitter activity that can be used to access the basic member values within the output values.

If you use either *Add input XML splitter* or *Add output XML splitter* then a new XML splitter activity instance is included within the current workflow together with the necessary connections.

5.3.3.4.13 XML Splitter Contextual View

The XML Splitter Contextual View shows information about the corresponding selected XML Splitter activity within the current workflow.

The information about the ports of the splitter is shown in a similar manner to that for WSDL activities.

If the XML Splitter is splitting the output of a WSDL activity then it may allow the outputs of the splitter to be further subdivided using *Add output XML splitter*. Similarly, if it is constructing data for the input of a WSDL activity, then additional constructing splitters may be necessary using *Add input XML splitter*.

5.3.3.5 Processor Output Port Contextual View

The Processor Output Port Contextual View shows the name of the selected output port of a processor in the current workflow.

5.3.3.6 Workflow Contextual View

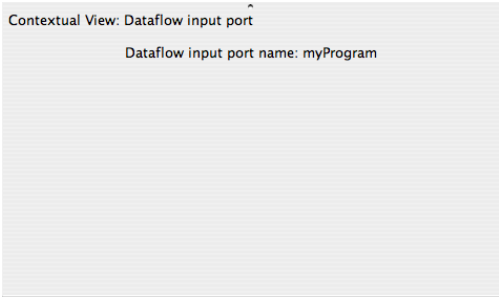
The Workflow Contextual View can be shown by selecting *Workflow Metadata* either from the Edit menu or the equivalent top level icon. It is also shown if you click an empty area of the current workflow.

You can alter these values by entering text in the corresponding fields.

The entered values are automatically saved when you click outside of the field. This may cause a problem on a Mac if you attempt to save the workflow without carrying out any further changes.

5.3.3.7 Workflow Input Port Contextual View

The Workflow Input Port Contextual View shows the name of the selected workflow input port.



5.3.3.8 Workflow Output Port Contextual View

The Workflow Output Port Contextual View shows the name of the selected workflow output port.

5.3.4 Workflow Explorer

The Workflow Explorer shows a tree-like representation of the current workflow. It can be perform some of the manipulations available within the Graphical Editor.

- - Workflow operations
  - Contextual viewing
- - Input port operations
  - editing
  - deletion
  - contextual viewing
- - Output port operations
  - editing
  - deletion
  - contextual viewing
- - Processor operations
  - rename
  - deletion
  - coordination from another processor
  - contextual viewing
- - Data link operations
  - deletion

5.4 Results Perspective

The results perspective consists of four main areas:

- The Workflow Runs list - where you can select which run of a workflow to see information about
- The Monitor View - where you can see the execution of your workflow as it is being run or see a representation of it when it has finished
- The Port Results View pane - where you can select which output of the workflow to examine and choose a specific value to examine
- The Rendered Result pane - where you can view the value of an output, possibly rendered in different ways, and save the result.

5.4.1 Workflow Runs

Workflow Runs shows the workflow runs that have been made during the current use of Taverna. The name of the workflow is listed along with the time when it was run. If you select a workflow run (by clicking on it), then the run is represented graphically in the Monitor View and its results listed in the Port Results View.

You can delete one or more workflow runs by selecting them (use shift-click to select multiple runs) and them clicking *Remove*.

5.4.2 Monitor View

The Monitor View shows a workflow as it is being run, or after it has been run.

If all calls to a processor have been finished, then the processor is shown in gray.

If some calls to a processor have been finished but other have not yet been made, or have not yet terminated, then the processor is shown partially in gray and partially in its ordinary 'design' colour. The proportion of calls that have been finished can be estimated from the size of the gray area within the processor. Blinking blue lines means that data is passing through. For some workflows you should be able to notice the pipelining of Taverna 2.0, i.e. downstream processors start enacting while the ones upstream are still iterating over incoming data. This might make the progress bars behave a bit strangely; for example, at first processor B has done 4 of 10 incoming items, but then 15 new items come in from processor A, then it has done only 4 of 25, reducing the apparent progress.

If no calls have yet been made to the processor, then it is shown in its 'design' colour.

5.4.3 Port Results View

The Port Results View contains a tab for each output port of the workflow corresponding to the selected workflow run.

When you select a tab, the view shows the 'handle' of the value output on the corresponding port. If a list was output, then a tree-like representation of the value is shown that can be expanded and collapsed. When you select a leaf of the tree, the corresponding value (which may be an element of a list) will be shown in the Rendered Result.

When a workflow is being run, the results will appear as soon as they are available rather than waiting for the workflow run to finish.

If an error occurs you should be able to see an error trace for the output showing from which processor the error happened.

An error does not necessarily break the workflow downstream; instead of the expected produced values there will be error documents to fill the blanks. So if you are iterating a list of length 5, and the 3rd iteration fails, you should still get the other 4 results.

5.4.4 Rendered Result

The Rendered Result shows a single value that was output on the port selected in the Port Results View.

The way in which the value is presented can be altered by selected a renderer from the *Result Type* pull-down menu. The value is then rendered in the view.

The value can be saved by clicking the *Save result* button and specifying a file in the resultant dialog. The form in which the result is saved depends upon the renderer that was chosen.

6 Glossary

Activity Definition

A reusable means of calling a particular service. An activity definition allows a particular service or set of services to be consumed in workflows. Currently, there is an activity wrapper for Web Services (WSDL), BioMart, BioMoby, Soaplab, R, Beanshell scripts and nested workflows.

Activity definitions are added to Taverna using the activities menu. They will then appear in the activity palette.

Instances of activity definitions may be included in a workflow. This is done by dragging the activity definition from the activity palette into the graphical editor.

It is common for services to do more than one thing, so in their unconfigured state an activity may have multiple potential uses. Configuration is done by clicking *Configure* in the activity's contextual view.

Activity Instance

An instance of an activity definition within a processor within a workflow. The activity instance may need to be configured before it can correctly call the corresponding service.

APIConsumer

A kind of activity that allows you to take an existing Java API and expose the classes and methods in that API as activities in Taverna.

Beanshell

beanshell is a java scripting language. The beanshell activity in Taverna enables you to write simple java scripts as part of your workflows. Beanshell scripts in Taverna typically perform data manipulation, parsing and formatting functions (so they are often types of Shim service).

BioMart

a federated data warehouse and management system from the EBI (European Bioinformatics Institute) and the OiCR (Ontario Institute for Cancer Research). Biomart contains genomic, proteomic

and any Biomart database if you provide the URL to the BioMart service interface (MartService). A BioMart activity allows you to query a BioMart database.

BioMoby

a collection of semantically described biological web services. A BioMoby activity in Taverna allows you to call a BioMoby service. Each BioMoby service is designed and registered to the BioMoby specification to improve interoperability between services. You can use the BioMoby annotations to discover which services consume the data types produced by other services. BioMoby services differ from ordinary Web Services in that these services consume and produce BioMoby Objects, so in order to use a BioMoby service, you have to first define your data as a BioMoby object.

Condition

A condition (or control link or coordination link) enables you to set dependencies between processors in the workflow that do not directly share data. A condition allows you to delay the invocation of a processor until another has finished.

Configuration (of activities)

some activities, for example, BioMart and Beanshells have input and output ports which are set by the workflow designer after they are imported into the workflow. In the case of BioMart, inputs and outputs are determined by selecting database query and filtering options from a configuration UI. For beanshell scripts, the script itself and the names of the input and output ports must be set in the beanshell configuration UI.

Data Link

A data link enables the results (an output) from one processor to be the input of another.

Depth

input and output ports may consume or produce single string values, lists, lists of lists, and so on. Therefore, as well as understanding the type of data that you are handling, you need to consider where this will appear in an output (for consumption by the next processor). A single value is considered to be of depth 0. A list of values is depth 1. A list of lists of values is depth 3 etc.

Implicit Iteration

if you supply a Taverna workflow with multiple items of data, Taverna will automatically take each piece of data in succession and feed it through the workflow. As results are produced, Taverna will also pass those results through the workflow if the next processor also accepts individual values. This enables data streaming. If two or more streams of data in a workflow are to be combined, the default behaviour of Taverna is to combine them in an *all against all* configuration. If you want to change this, you must change the list combination settings (sometimes referred to as the iteration strategy). Please see list handling contextual view.

Local Worker

a beanshell activity that is part of the Taverna download and helps with gluing scientific services together. The local workers are a collection of data manipulation and formatting services (also known as Shims).

Merge

if the input for a particular processor is to be combined from more than one upstream processor, these inputs can be combined in a special *Merge* operation. This allows you to feed all the data in to the next processor. Merges appear in the graphical editor.

Nested Workflow

a workflow within a workflow. In an abstract sense, a nested workflow is just another kind of activity that can be added into a workflow, except that instead of it being a *black box*, it is a *white box* so you can see what is happening inside. It is often the case that a workflow designed for one purpose can be used again for other experiments and can be imported and added to another workflow.

Nested workflows can be added to a workflow by dragging the nested workflow activity definition into the graphical editor.

Port

in Taverna, a port is a connector, an input or output of a processor (a service), or from a workflow or nested workflow. Typically, an input port is either a data input or an input parameter setting. In a web form, this would be equivalent to text boxes and configuration settings, and on a command line, this would be equivalent to parameter settings (flags). Output ports are the same. They allow you to pass the results of one processor to others.

Processor

a processor contains a configured instance of an activity in a workflow. An activity (that adapts a service) may do many things, but once its containing processor is connected to other processor and its input and output parameters have been set, then what it can do in that instance is defined for that particular run of the workflow. The processor also describes the control layer around the activity e.g. how many times an activity should be retried in the event of a failure or what alternative activities to use. A processor can be manipulated in the graphical editor.

Provenance

Provenance is a history or a trace of (in this case) a workflow experiment. Workflow provenance data allows you to find out which workflows have been executed, with what data, and what the intermediate and final results were. It allows you to compare results between workflow runs and to confirm that all services have completed successfully during a run. It is also useful for debugging the design and development of your workflow.

Renderer

Taverna can display the results of a workflow in many different formats. As well as the standard formats (e.g. HTML, Text, XML etc), Taverna can display data using specialist formats. A renderer is a plugin that controls how to display data in these specialist formats. For example, Taverna can display 3D Jmol images of protein structures.

RShell

An activity that enables analyses using the R statistical package to be incorporated into the workflow

Scientific Service

A service that performs analysis on data or sets of data

Service

This is the generic term we use for any operation, protocol or function that can be made available via activity definitions and instances for use in a a Taverna workflow. These include WSDL web operations, BioMart services, BioMoby services, beanshell scripts, SoapLab services, R scripts and workflows. Each type of service has a dedicated type of activity to adapt it for use in Taverna.

Shim Service

A service that does not perform a scientific function, but acts to 'glue' services together that otherwise have incompatible outputs/inputs. Shim services perform data manipulation, formatting, parsing or database mapping functions.

SoapLab

a tool for wrapping command-line and legacy programs automatically as Web Services. Soaplab is particularly designed for people who prefer to program in perl or python. The services that soaplab produces are a kind of Web Service, but they are slightly different in that 1) the service provider can add more descriptions of how the service works 2) the service can be statefull (i.e. an analysis is submitted to a soaplab service and it produces an identifier. Taverna can then keep checking this identifier for the status and for completion of the job). SoapLab activities can be made known to Taverna and then included in workflows.

String Constant

a String Constant activity allows you to set a fixed input for a workflow or processor. This is useful for inputs that are fixed across multiple runs of the same workflow, and also for publishing your workflows. The string constant provides an example input for people wanting to try out the workflow. They can download it from myExperiment (http://www.myexperiment.org) and just run it with the string constant data instead of trying to find suitable input data

Web Services

a standardised way of programmatically integrating web-based applications for machine to machine interaction over the network. Web services primarily use XML, SOAP and WSDL specifications and enable distributed data stores and analysis tools to be accessed and used from the scientists own desktop computer. Many types of web service can be included in Taverna as activities.

Workflow Run

the name given to one execution of a single workflow instance. This information includes what input data was provided.

WSDL

stands for Web Service Description Language. It is an XML file that is the interface of a Web Service. It is the machine-readable description of the operations (or functions) offered by the service. Taverna can handle any Web Service with a WSDL interface if you provide the URL to the WSDL file.

Workflows

Workflows enable the automation of *in silico* experiments (from small to very large-scale) and the formalisaton of experimental protocols. Taverna is a workflow management system and allows researchers to combine and co-ordinate distributed analysis tools and data resources into workflows. taverna has access to over 3500 resources for use within workflows

7 Contact Information

For more information about Taverna, the myGrid team or help with any problems, please visit the myGrid website http://www.mygrid.org.uk

Appendix

1. fred

fred

Index

Q

R

S

T

W

X