Practical: Integrating Data with Taverna workflows

Dr Katy Wolstencroft

Universiteit Leiden
The Netherlands

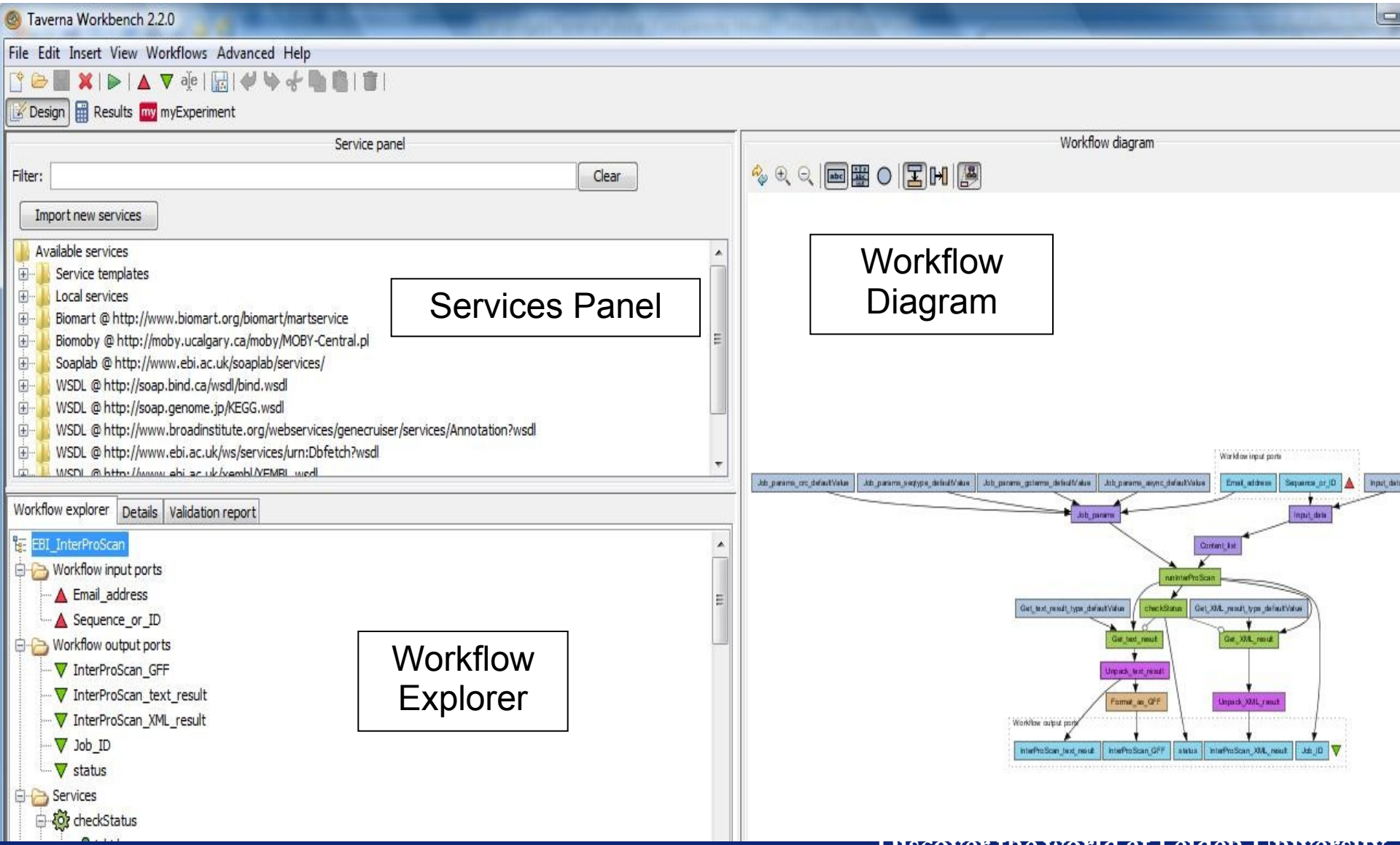
1. Introduction

- This tutorial will give you an introduction to designing, executing and reusing workflows with the Taverna workbench
- Download and install the Taverna workbench by following the instructions at:

  http://www.taverna.org.uk/download/workbench/2-5/

- Select the Bioinformatics edition

1. The Taverna Workbench



1. Workflow Diagram

The workflow diagram is a visual representation

- Shows inputs, outputs, services and data flows
- Allows editing of the workflow by dragging and dropping and connecting services together
- Enables saving of workflow diagrams for publishing and sharing

1. Workflow Explorer

The Workflow Explorer shows the detailed view of your workflow.

- It shows default values and descriptions for service inputs and outputs
- It shows where remote services are located.
- It shows configuration details, such as iteration and looping
- Workflow validation details can also be found here. Before a workflow is run, Taverna checks to see if it is connected correctly and if all services are available.

1. Available Services Panel

Lists services available by default in Taverna

- Local java services
- WSDL Web Service – secure and public
- RESTful Services
- R Processor services (for statistical analyses)
- Beanshell scripts
- Xpath scripts
- Spreadsheet import service

The services panel also allows you to add new services or workflows from the web or from file systems – there are loads more available!

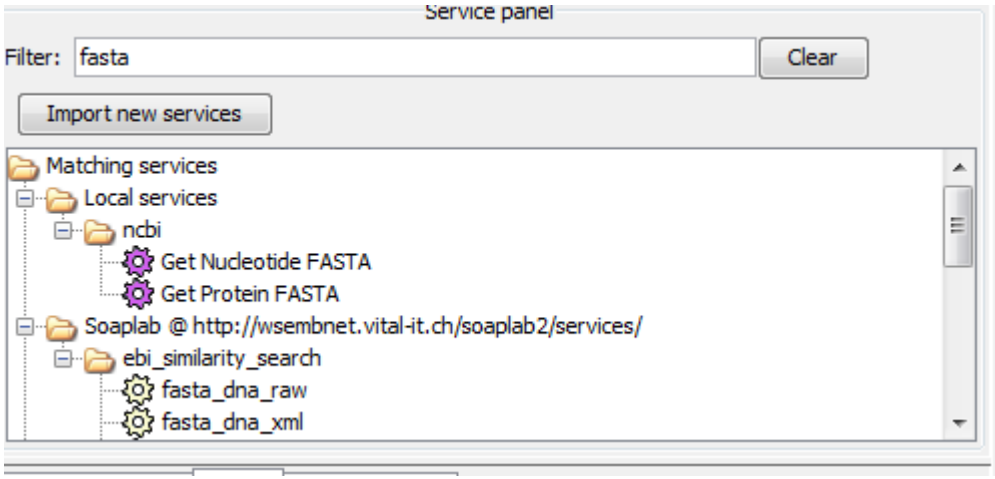
Exercise 2: Building a Simple Workflow

We will start with something easy - retrieving a protein sequence from a remote database and identifying functional motifs

Go to the Services Panel

- Type '*fasta*' into the filter box at the top of the panel
- You will see several services in the search results

Select '*Get Protein FASTA*' and drag-and-drop it into the workflow diagram panel.

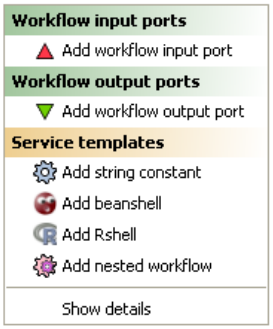


Exercise 2: Building a Simple Workflow

- To see all the inputs and outputs for this service, you need to click the *show ports* button at the top of the workflow design panel

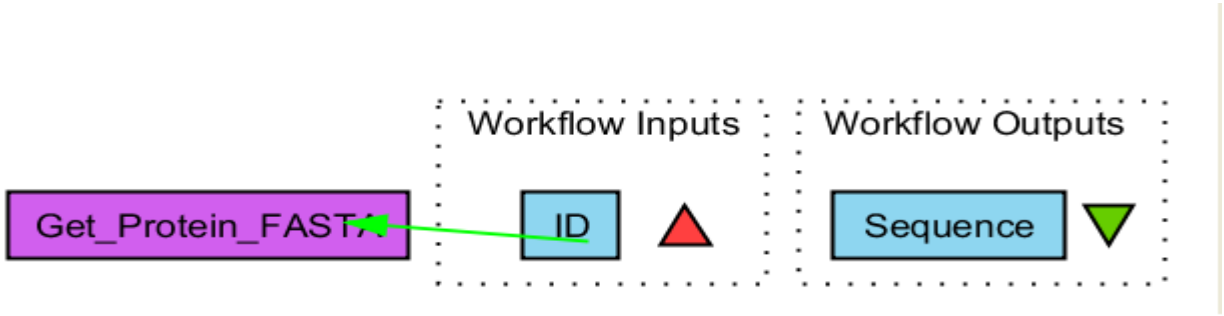
Exercise 2: Building a Simple Workflow

- Now we need to add an input and an output to create a simple workflow with 1 service
- In a blank space in the workflow diagram panel, right-click and select "Workflow Input port"
- Type in a name for this input (e.g. ID)
- Create a workflow output the same way and call it "sequence"



Get_Protein_FASTA

Exercise 2: Building a Simple Workflow

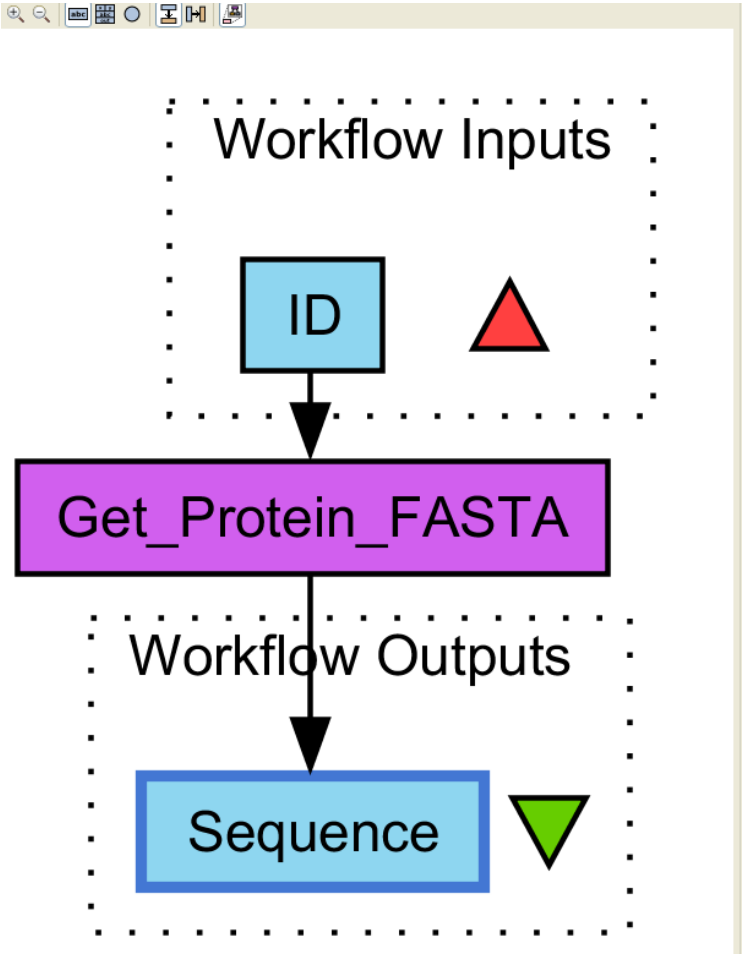


- You now have 3 boxes in the diagram and we need to connect them up
- Click on the input box and drag towards the id in “Get Protein FASTA” and let go. An arrow will connect the two boxes

Exercise 2: Building a Simple Workflow

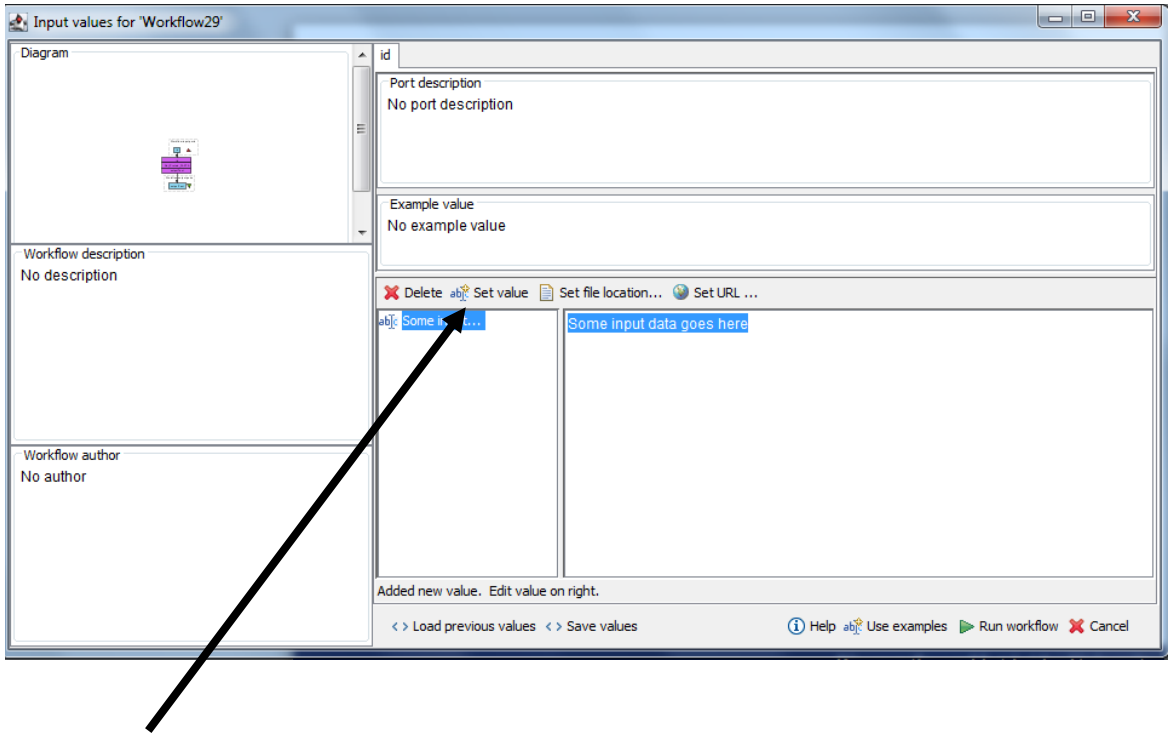
- Click on the output box, drag towards “Get protein fasta”, and let go. An arrow will connect the two boxes
- You have now built your first workflow!

- Run the workflow by selecting “file -> run workflow”, or by clicking on the play button at the top of the workbench



Exercise 2: Building a Simple Workflow

An input window will appear. As you can see, we have not yet added a description of the workflow or of the input



Click on ‘set Value’ in the input window and add a Uniprot protein identifier (e.g. P15409) where it says “some input data goes here”

Exercise 2: Building a Simple Workflow

- Click “run workflow”
- As the workflow runs, you will see its progress in the results window
- In the bottom left of the results window, click on ‘*value 1*’. You will see a protein sequence from Uniprot

  Now we will find out what functional motifs the protein contains by running an InterProScan on the sequence

  Interproscan is a commonly used tool – we will first see if other people have developed workflows by searching myExperiment

Exercise 3: Reusing Workflows

- Go to myexperiment.org and type *Interproscan* in the search box
- As you can see, there are many Interproscan workflows
- Filter the results for Taverna 2 workflows, written by Alan Williams, by selecting filter options from the left-hand side

Expercise 3: Reusing Workflows

- As you can see, there are still quite a few workflows to choose from
- Select the workflow called **EBI_InterproScan**

(http://www.myexperiment.org/workflows/4338.html)

- Find out how many times it has been viewed and downloaded, who the author(s) are and what the workflow description is
- In the *Run* section of the page, copy the URL under Option 1 and go back to the Taverna workbench

Exercise 3: Reusing Workflows

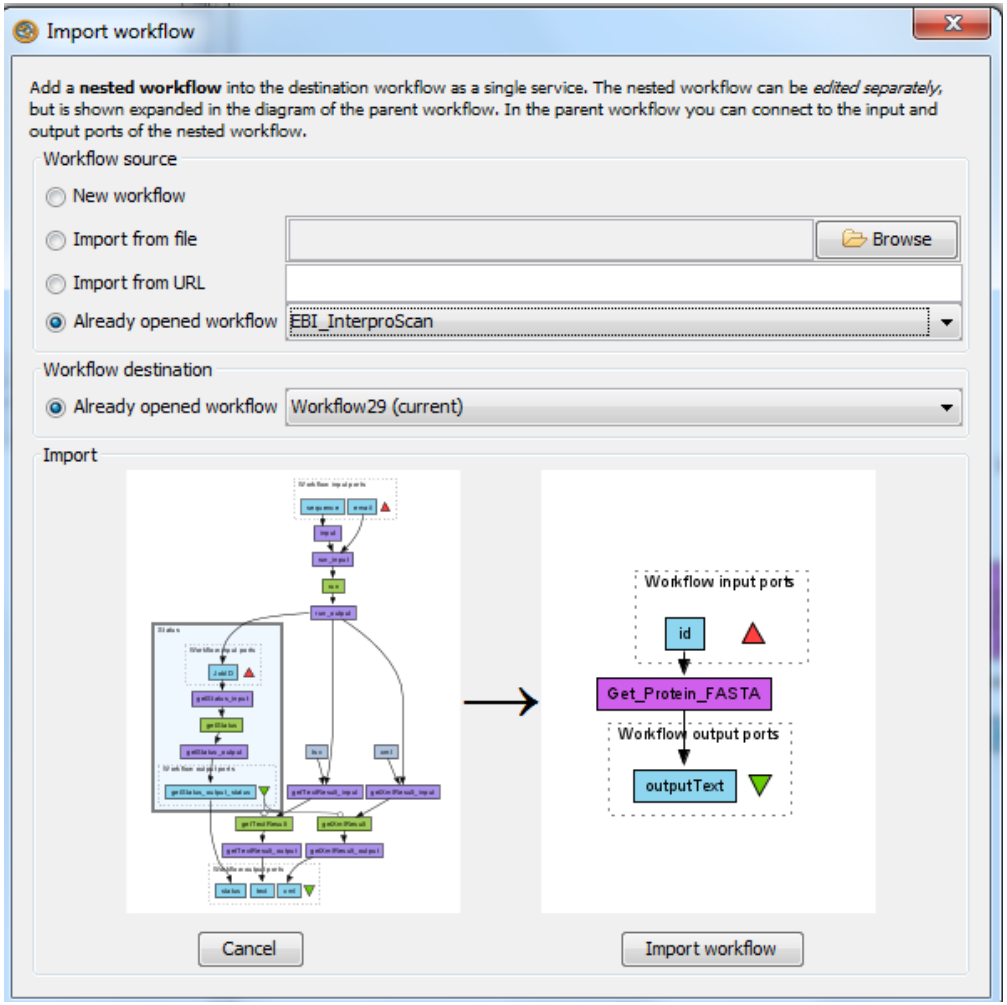
- Go to File → Open workflow location and paste in the URL you copied from myExperiment
- Run the workflow, using the example protein sequence and your own email address
- As the workflow is running, pay attention to what is happening in the nested workflow. This is an example of looping. As the service runs, Taverna keeps checking to see if the job is finished

Exercise 4: Combining Workflows

- Now we will link your first workflow to the Interproscan example
- Go back to your first workflow by selecting it from the list in the workflow menu at the top of the workbench
- Select Insert → Nested Workflow
- In the pop-up window, you will see your workflow and some options for inserting another

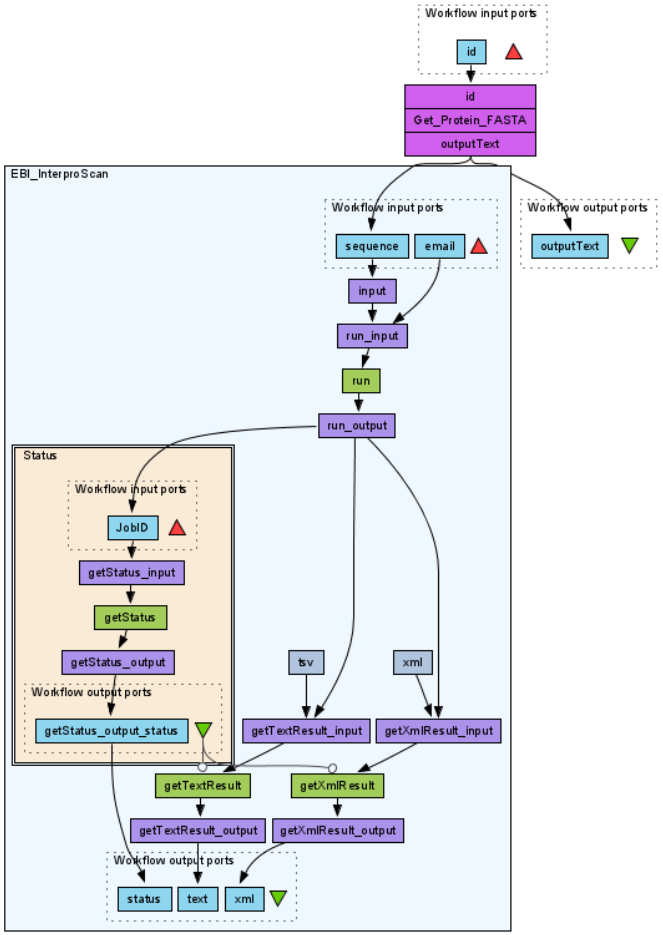
Exercise 4: Combining Workflows

- In workflow source, click on *already opened workflow* and select the EBI_Interproscan workflow
- Select Import workflow



Exercise 4: Combining Workflows

- Drag and drop a connection between Get Protein FASTA:output and the Sequence input in the InterproScan workflow
- Make a new workflow input called Email and connect it to the email input in the nested workflow
- Create a new output port called ScanResults and connect the nested workflow output *'text'* to it
- Save and run the workflow



Exercise 5: RESTful Services

- The workflows we have built so far used local scripts and WSDL web services, but Taverna can connect many different service types
- Go to File → New Workflow and find the service templates in the services panel
- Drag and drop a REST service into the workflow panel
- In the pop-up window, you will see an example REST service from Uniprot
- Look at the URL – variables are in curly brackets
- Click *Apply* and *Close*

Exercise 5: RESTful Services

- As you can see, the variable in curly brackets is an input for the REST service
- Create an input and output port and run the workflow with the same Uniprot ID (P15409)
- Go to the documentation page for the OpenPHACTS API (https://dev.openphacts.org/docs/1.5). We will create workflows from the OpenPHACTS services

Exercise 6: Open PHACTS REST

- On the Open PHACTS documentation page, run the Pathway Information: Get Targets operation with your own API key and API ID, an XML output and the input:

http://www.wikipathways.org/instance/WP1531

- Copy the Get Request URI and paste it into a new REST service in a new workflow in Taverna – remember to remove the starting and trailing “”
- Add an output port and connect it to the REST service
- Run the REST service and check it is working – you don’t need to provide an input

Exercise 6: Open PHACTS REST

- Right click on the REST service and click Configure REST service
- Change the API ID, API Key and pathway URL to parameters by replacing them with curly brackets and a name :
  https://beta.openphacts.org/1.5/pathway?uri={pathway}&app_id={id}&app_key={key}
- Change the name of the service to PathwayInfo, by right-clicking on the service and selecting *Rename*

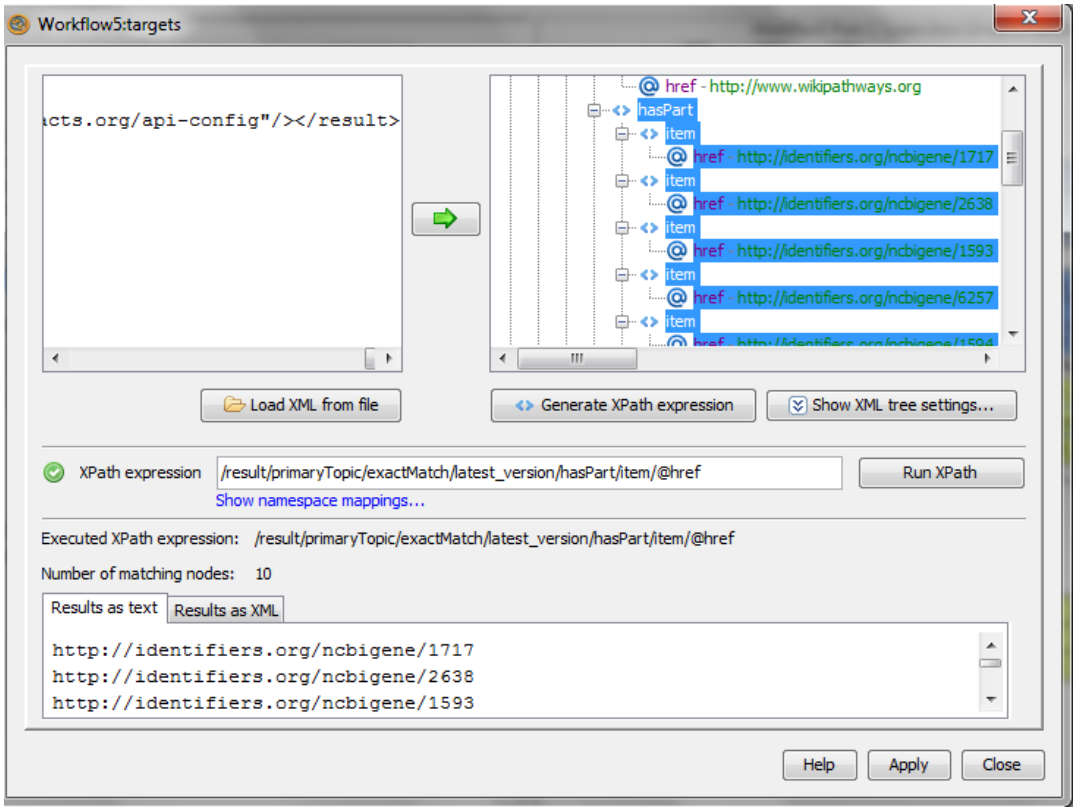
Exercise 6: Open PHACTS REST

- The API ID and API Key will be the same for every OpenPHACTS service, so we will set these values as *Text Constants*
- Go to Insert → Text Constant and enter the API ID as a value and rename the service API ID
- Repeat the process for the API Key and connect the text constants to the *PathwayInfo* inputs
- Create a regular input port for the pathway URL input
- Rerun the workflow, this time, you will need to supply a pathway URL as input
- Save the workflow output as an xml file

Exercise 7: Processing XML Output

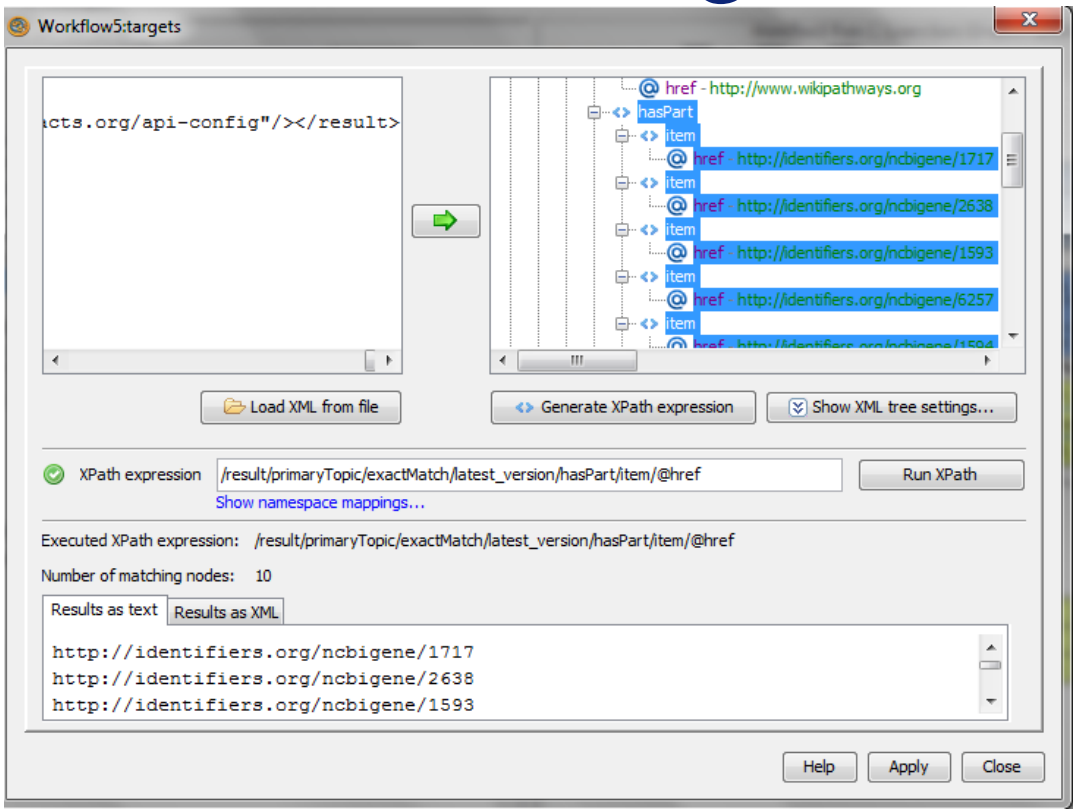
- The workflow output is an XML file. If you wish to use elements of the XML in subsequent services, you need to extract the relevant sections. The Xpath service will help you do that
- Drag and drop an xpath service from the service panel. In the pop-up window, upload the xml file you saved from the workflow run
- Navigate the XML structure to find the XML elements that list the ncbigene ids in the pathway (potential druggable targets) – see following screenshot

Exercise 7: Processing XML Output



- When you have selected the right elements, click ‘Generate Xpath expression’ and ‘Run Xpath’to see your results at the bottom of the window

Exercise 7: Processing XML Output



- Apply and close the service configuration and link the new xpath service to the output of the REST service, create a new workflow output and save
- Rerun the workflow

Exercise 8: Open PHACTS Combinations

- Now you have a list of targets for this pathway, extend the workflow to discover which drugs act upon these targets and which diseases have been associated with them, by using other OpenPHACTS services

Exercise 9: Open PHACTS Resources

- Go to myExperiment and search for workflows using Open PHACTS services
- How many are there? What do they do?
- Download and run one of the workflows written by Stian Soiland-Reyes – what does it do? How does it relate to the workflow you have just written? Could you combine them?

EXTRA ADVANCED EXERCISES

Advanced Exercises

- These exercises have given you a brief introduction to Taverna, but we have just scratched the surface.

- The Taverna engine can also help you control the data flow through your workflows. It allows you to manage iterations and loops, add your own scripts and tools, and make your workflows more robust
- The following exercises give you a brief introduction to some of these features

Iteration

As you have already seen, Taverna can automatically iterate over sets of data.
When 2 sets of iterated data are combined, however, Taverna needs extra information about how they should be combined.
You can have:
A cross product – combining every item from list 1 with every item from list 2 - ***all against all***
A dot product – only combining item 1 from list 1 with item 1 from list 2, and so on – ***line against line***

Iteration

Find and load the workflow ‘Demonstration of configurable iteration’ from myExperiment

- Read the workflow metadata to find out what the workflow does (by looking at the ‘Details’)
- Select the ‘*ColourAnimals*’ service and select the ‘Details’ in the workflow explorer and ‘configure list handling’
- Click on ‘*dot product*’ in the pop-up window. This allows you to switch to cross product

Iteration

- Run the workflow twice – once with '*dot product*' and once with '*cross product*'.
- Save the first results so you can compare them – what is the difference? What does it mean to specify dot or cross product?
  NOTE: The iteration strategies are very important. Setting cross product instead of dot when you have 2000 data items can cause large and unnecessary increases in computation!

Retries: Making your Workflow Robust

- Web services can sometimes fail due to network connectivity
- If you are iterating over lots of data items, you can guard against these temporary interruptions by adding retries to your workflow
- Upload the 'Retry-Example' workflow, by Katy Wolstencroft, from myExperiment. This workflow is designed to fail sometimes.
- Run the workflow as it is and count the number of failed iterations

Retries: Making your Workflow Robust

- Now, select the ‘sometimes_fails’ service and select the ‘details’ tab in the workflow explorer panel
- Click on ‘advanced’ and ‘configure’ for retries
- In the pop-up box, change it so that it retries each service iteration 2 times
- Run the workflow again – how many failures do you get this time?
- Change the workflow to retry 5 times – does it work every time now?

Parallel Service Invocation

- If Taverna is iterating over lots of independent input data, you can improve the efficiency of the workflow by running those iterated jobs in parallel
- Run the Retry workflow again and time how long it takes
- Go back to the design window, right-click on the 'sometimes_fails' service, and select 'configure running'
- This time select 'Parallel jobs' and change the maximum number to 20
- Run the workflow again
- Does it run faster?

Parallel Service Invocation : Use with Caution

- Setting parallel jobs makes your workflows run faster, but you should be careful if you are using remote services. Sometimes they have policies for the number of concurrent jobs individuals should run (e.g. The EBI ask that you do not submit more than 25 at once).
- If you exceed this number, your service invocations may be blocked by the provider. In extreme cases, the provider may block your whole institution!