Introduction to the Taverna Tool service

Stian Soiland-Reyes and Christian Brenninkmeijer
University of Manchester
material by Mr Alan Williams

http://orcid.org/0000-0001-9842-9718
http://orcid.org/0000-0002-2937-7819
http://orcid.org/0000-0003-3156-2105

Bonn University, 2014-09-01 / 2014-09-03
http://www.taverna.org.uk/
What is a tool service?

- Allows you to call a command line script as part of a workflow
  - Simplest case is calling a single tool
- Can be run on your local machine or a machine that you can ssh to
- Data is passed by reference
  - No big transfers to/from Taverna
- Data kept where the script is run until/unless needed
Choose “Tool” from the “Insert” menu
In the tool service popup type `java -version`
Close the configuration
Connect the STDERR and STDOUT ports of the tool service to workflow output ports
Simple tool service configuration

Specify the commands that you want to run. You can use data arriving at an input port to replace parts of the command or to write to a file. You can also take data written to a file and send it to an output port.

```
java -version
```

Valid return codes: 0

- [ ] Show STDIN
- [x] Show STDOUT
- [x] Show STDERR

[Load description] [Export description] [Clear script] [Help] [Apply] [Close]
Run the workflow

STDERR should look similar to:

java version "1.8.0"
Java(TM) SE Runtime Environment (build 1.8.0-b132)
Java HotSpot(TM) 64-Bit Server VM (build 25.0-b70, mixed mode)
We are going to use the *forester* utilities by [Christian Zmasek](https://sites.google.com/site/cmzmasek/home/software/forester/phyloxml-converter)

**Download**

- forester_1028.jar by following the links on [https://sites.google.com/site/cmzmasek/home/software/forester/phyloxml-converter](https://sites.google.com/site/cmzmasek/home/software/forester/phyloxml-converter)
- ..or download it from the myExperiment group
- Remember which folder you downloaded it to
Create a new workflow with a tool service that calls the jar (modify the path)

java -cp C:\Users\stain\Downloads\forester_1035.jar

- Connect STDERR and STDOUT
- Run the workflow
- It fails. We cannot just call the jar
We cannot just call the jar

Look for the parameters of this tool at https://sites.google.com/site/cmzmasek/home/software/forester/phyloxml-converter

Change the tool service so the script says on one line:
```java
java -cp C:\Users\stain\Downloads\forester_1035.jar org.forester.application.phyloxml_converter -f=nn infile outfile
```

This converts the `infill` to PhyloXML and writes it to `outfile`

Run the workflow
We need to pass an input file

Configure the tool service and add a file input called infile
- Add a file output called outfile

You can use a file output to take the content of a file produced by the tool and send it to an output port of the service.

Taverna port name: outfile
Use port name for file: ✓
From file: 
File type: Text

Add file output
The tool service now has two extra ports

- Connect `infile` to a workflow input port and `outfile` to a workflow output port
Run the workflow

As input, you can use the contents of http://www.myexperiment.org/files/1055/versions/1/download/example.nh.txt (or use Set URL)

The outfile is in PhyloXML format

Click Value type: XML tree
- Rename the first tool to **converter**
- Add a new tool service that calls
  ```java
  java -cp C:\Users\stain\Downloads\forester_1035.jar org.forester.archaeopteryx.Archaeopteryx infile
  ```
- Add a file input called **infile**
- Rename the tool service to **display**
- Connect the **outfile** of converter to the **infile** of **display**
- Run the workflow
The archaeopteryx display tool will show – exit it to finish the run.
PhyloXML converter can take options

Add a new String replacement port to the converter service called options
Change the converter script to include the options

```
java -cp C:\Users\stain\Downloads\forester_1035.jar org.forester.application.phyloxml_converter -f=nn %options% infile outfile
```

- `%%options%%` will be replaced by the string passed to the service
- Connect the options port to a workflow input port
- Run the workflow with options as the empty string
- Run the workflow with options as `-o`
- Compare the `outfile` with that from the previous run
Add the **Xpath service** to pick up the species name of the second-level clade branch (bear, raccoon)

Create a **component** family in your local registry called *forester*

Create a components in the forester family for the *converter* and *display* services

Build a workflow using the two components from *Available Services*

What possible problems can you imagine if you want to share a workflow using the External Tool service?

**Expert:** Are you able to modify your workflow to be sharable? Hint: Look at *Advanced* tab of Tool service.