Installation

System Requirements

- Linux-Unix Operating System
- Apache 2 (with mod python enabled)
- MySQL Server 5.
- Java 6
- Python (2.5 or 2.6) with python dev tools and setup tools installed:
- Tavenra 1.7.2 , Taverna 2.2

Basic Installation

1. Download Tavaxy from http://www.tavaxy.org/Downlaods/tavaxy.tar.gz.
2. Extract the downloaded package inside the apache websites folder.
3. Change the ownership and permissions of the folders as follows:

Path	Owner	Read permission	Write permission	Execute Permission
webui	apache	apache	none	none
bin	apache	apache, tavaxy	none	apache, tavaxy
results	apache	apache	apache	none
engines	tavaxy	tavaxy, apache	tavaxy	tavaxy
files/temp	tavaxy	tavaxy, apache	tavaxy, apache	none
files/workflows	apache	apache, tavaxy	apache	none
files/scripts	apache	apache, tavaxy	apache	tavaxy
files/inputFiles	apache	apache, tavaxy	apache	none
files/configuration	tavaxy	tavaxy, apache	none	none
files/toolDesc.json	apache	apache	none	none

apache represents the name of the account that is used by apache and *tavaxy* represents the name of the account that is used to run Tavaxy, this is the account that was used to run the last step.

4. Configure the bin folder to be a mod python enabled directory
   Add this to apache configuration file

```
<Directory ROOT_PATH/bin>
   AddHandler mod_python .py
   PythonHandler mod_python.publisher
   PythonDebug On
</Directory>
```

5. Navigate to the *ROOT_PATH* (tavaxy folder) and run the following commands:

```
sudo python builder.py
python setup.py

This installs required python packages, for any error, contact your system ad
```

6. Create a user at MySQL that *Tavaxy* will use.

7. Create two schemas named **galaxy0** and **users**. Assign all privileges of these two schemas to the created account. Note that changing the names of these two schemas required editing two configuration files, the configuration file at *_PATH*/files/configuration/config.cfg and the configuration file at *ROOT_PATH*/engines/galaxy/universe_wsgi.ini.

8. Navigate to the *ROOT_PATH* and run the following command:

```
python setup.py
```

9. Navigate to *web_address*/webui/config.html to continue the configuration process.

10. To start Tavaxy, navigate to *ROOT_PATH*/engines/galaxy and run the following command:

```
sh run.sh --reload
```

Installation Issues

- Installing MySQLdb will fail if MySQL database development files are not available. For ubuntu users installing the MySQLdb from the package manager solves this issue.

- Tools within tavaxy are not installed within the system, they must be installaled by the system administrator. The prebuilt binaries for the tools withing tavaxy under linux x64 can be found in the *Software Packges* tab at www.tavaxy.org. Links to the each tool pages for installation of other architectures are also available. For tools that are not availale within Tavaxy refer to Adding new tool section.

Cluster Installation

Tavaxy can be configured to run jobs over cluster configured with *pbstorque*. In order to configure Tavaxy to work over cluster the following items must be taken into consideration:

- All tools must be installed on all execution nodes.

- The following directories must be shared between execution nodes with the same paths as the main node (between brackets is the default path):

  - workflows directory ($ROOT\_PATH$/files/workflows)
  - scripts directory ($ROOT\_PATH$/files/scripts)
  - tool data directory ($ROOT\_PATH$/engines/galaxy/tool-data)
  - pbs scripts directory ($ROOT\_PATH$/engines/database/pbs)
  - working directory ($ROOT\_PATH$/engines/database/job_working_directory)
  - temporary directory ($ROOT\_PATH$/engines/database/tmp)

Input and output files are copied from the main node to the execution node prior to job execution and from the execution node to the main node after job execution is completed. Therefore there must be two way access between each node and the main node via ssh without password prompt. To enable the cluster mode edit the following properties in the configuration file located at $ROOT\_PATH$/files/configuration/config.cfg:

- cluster, under the Execution section, set it to *on* to enable job execution on the cluster and to *off* to disable job execution over the cluster.
- user, under the Execution section, set it to the name of the user account to be used for ssh between the nodes, in case there is no user name required, leave it blank.
- parallel, under the Execution section, this is responsible for determining the maximum number of jobs to execute in parallel when the node is being executed iteratively over an input data set, set it to *0* to disable this limit.
- mainnode, under the Execution section, set it to the name of the main node, not the ip-address.

Cloud Installation

Refer to the Tavaxy on Cloud manual.

Adding a new tool

To add a new tool to Tavaxy a tool description file must be created and added to a directory inside the tools directory at $ROOT\_PATH$/engines/galaxy/tools and the path to place it in a specific place on the left panel a record must be added to the tools configuration file at $ROOT\_PATH$/engines/galaxy/configuration/tool$_c onf.xml$.

Tool Description File

This file is responsible for defining the tool to the Tavaxy system, the command line the parameters, the input and output ports of the tool on interface are all generated from the tool description file. This file is written in xml and is composed of 5 main tags *inputs*, *description*, *command*, *outputs* and *help* with a root tag *tool*.

tool This is responsible for defining the id of the tool which must be unique among all tools, whether this tool will support running over the cloud or not and the tool name. It takes the following format:

```
<tool  id='id of the tool' name='name of the tool' cloud='true or false'>
```

description The text within this tag is what appears under the tool name in the left panel in the workflow editor. In order for this description to appear in the left panel a new record must be added to the tool descriptions configuration file at *ROOT_PATH*/files/toolDesc.json. This is a file containing a json object with each tool id (key) assigned to the descriptiont (value).

command This is responsible for defining the command that will be used to run this tool. This tag takes the following format:

```
<command interpreter="...">...</command>
```

The interpreter defines the interpreter to be used to run this tool either perl, python, sh. The command itself is written within the *command* tag. This command is written in Cheetah template where variables are the unique names of inputs and outputs.

inputs This is the parent tag for the list of tags describing the input ports and parameters for this tool. Each input port or parameter is defined with a *param* tag. Within the *param* tag each input port or parameter is given a name which must be unique among all inputs and outputs, *label* which will appear on the interface and *type*. There are four different types:

- **data**, this type is used to define an input port. In this case the following attributes must be set
  - format, the most generic format is data, it can be Fasta, fastq, etc.
  - depth, this defines the depth of the input list that this tool supports, 0 means it handles a single token, 1 means it handles a list of tokens, 2 means it handles a list of a list of tokens, 'inf' means it handles a list with any depth.
- **select**, this type is used to define a parameter which its value is chosen from list of options. In this case the following parameters attributes must be set:
  - multipe, which is either true or false, this defines whether multiple values can be selected from the available options or not. In this case the display must be set to *checkboxes*.

- display, this defines how the options are displayed, the default is a drop down list when display is not set, radio boxes when display is set to *radio* and checkboxes when display is set to *checkboxes*.
- separator, in case that *multiple* is set to true, the separator should be set to the character that will be used to separated the selected values. The default separator is the comma.

After setting the above attributes the options must be listed in the following format:

```
<param type="select" name="unique name" label="...">
<option value="value1" selected="true">option1</option>
<option value="value2">option2</option>
<option value="value3">option3</option>
</param>
```

- **boolean**, this type is used to define a parameter which takes one to two values. The following attributes must be set:
  - checked, this is set to either true or false, it defines whether the default of this paramter is to be set to true or false.
  - truevalue, the value to be assigned to this parameter when it is checked.
  - falsevalue, the value to be assigned to this paramter when it is not checked.

```
<param type="boolean" checked="true or false" name="unique name" truevalue=".." falsevalue=".." label=".."/>
```

- **text, integer, float**, this type is used to define a parameter which takes a value input from the interface. The following paramters must be set:
  - size, this defines the size of the text box that will appear on the interface.
  - value, this, defines the default value assinged to this paramter

```
<param type="float or integer or text" name="unique name" label=".." value=".." size=".."/>
```

outputs This is the parent tag for the list of tags describing the output ports for this tool. Each output port is defined with a *data* tag. Within the *data* tag each output port is given a name which must be unique among all inputs and outputs and *type*, the most generic type is data. The following is a sample for the outputs tag

```
<outputs>
  <data type='data' name='output1'/>
  <data type='data' name='output2'/>
</outputs>
```

help This is the help text that appears in the properties panel on the right. This tag supports help text written in restructured text format.

Tools configuration File

After creating the tool description file and saving it in the desired destination, a record must be added to the the tools configuration file. This file is arranges the tools in to sections, each section has a name and a unique id. Each tool record is defined in the following format:

```
<section id="unique section id" name='..'>
  <tool name='tool name' file="tool path" id='tool id' />
  ..
</section>
```

Tool name is the name that appears on the left panel, the file path is the path to the xml file starting from the *ROOT_PATH*/engines/galaxy/tools/ i.e if a tools is located at *ROOT_PATH*/engines/galaxy/tools/myTools/tool.xml the *file* is set to myTools/tool.xml.