Dynamic Graph Workflows Manual

Paul Grosu
Northeastern University

pgrosu@gmail.com

Table of Contents

Introduction

This manual will describe the implementation, operation and logic of the dynamic graph workflows (DGW) program.

Description and Launching of the DGW Programs

The program was generated using `Node.js` to demonstrate how one can have a dynamic collection of interactive nodes – created ad hoc – in order to interact and run processes. The program has a broker (`cwl_node.js`) and a node program (`cwl_node.js`). One first launches the broker and then an interactive node program on two separate terminal windows, as follows:

```
node cwl_broker.js
```

```
node cwl_node.js 127.0.0.1 4000 interactive
```

The node program is launched with the local IP address and port of the broker, including name of this node (‘`interactive`’). It is recommended that the node name is kept as ‘`interactive`’ since it is used by the broker to broadcast information.

Sending Commands on the Interactive node

Below are a set of instructions sent on the interactive node, which will be described afterwards – a file called ‘`whale.txt`’ is assumed to be present:

```
make_node 1
make_node 2
on 1 command="wc"
on 2 input="whale.txt"
from 1 command
from 2 input
run 1 with 2 using input store_as wc_result
from 1 wc_result
on 1 command="ls -l"
on 2 input="*.js"
run 1 with 2 using input store_as ls_result
from 1 ls_result
```

The first two commands create a node name 1 and another named 2:

```
make_node 1
make_node 2
```

Then a variable named command is assigned on node 1, and a variable named input assigned the input filename.

```
on 1 command="wc"
on 2 input="whale.txt"
```

Then we inspect the values of those variables on the appropriate nodes to get the information back as follows – notice the returned results:

```
from 1 command
wc
from 2 input
whale.txt
```

Then we run node 1 with the input of node 2 and push the result into a new variable called `wc_result`, which will reside on node 1:

```
run 1 with 2 using input store_as wc_result
```

Then we inspect the contents of `wc_result`:

```
from 1 wc_result
     16     198    1111 whale.txt
```

Next we update the command on node 1 and input on node 2, and run node 1 again with the input from node 2, but now store the results in a new variable:

```
on 1 command="ls -l"
on 2 input="*.js"
run 1 with 2 using input store_as ls_result
from 1 ls_result
-rw-rw-rw-   1 user     group        6300 Jun  2 15:37 cwl_broker.js
-rw-rw-rw-   1 user     group        8357 Jun  2 15:37 cwl_node.js
```

Currently the program is made to run on the local computer, but can be easily adapted to run in a distribution fashion. More commands will be added with time and please feel free to contribute.

This program is *alpha* so it is best to wait a second between entered commands, in order for them to be processed successfully. Additional approaches are being explored and will be added with time, and feel free to contribute.