

# 1 Introduction

The goal of this document is to provide as clearly as possible a description of the shell input and XML output of the AWS shell interface which is provided with NARVAL software. In order to pass command there is two available mode : local with narval\_shell process or remotely with aws\_shell process.

## 2 Main Shell Commands

The command names aren't case sensitive.

### 2.1 Help

Syntax :

- Help command\_name

example : help get

```
1 <result cmd="get" status="OK">
2   <message type="help">
3     <syntax>get argument_name sub_system_name</syntax>
4     <syntax>get argument_name sub_system_name actor_name</syntax>
5     <syntax>get arguments sub_system_name</syntax>
6     <syntax>get arguments sub_system_name actor_name</syntax>
7     <syntax>get sub_systems</syntax>
8     <syntax>get actors sub_system_name</syntax>
9   </message>
10 </result>
```

- Help commands

don't return xml format for the moment

### 2.2 Get

Object : retrieve parameters values from NARVAL processes.

Syntax :

- get argument\_name sub\_system\_name

retrieve parameter with name argument\_name from sub system coordinator named sub\_system\_name

example : get configuration\_file toto

```
1 <result cmd="get" status="OK" sub_system_name="toto">
2   <data mode="READ_WRITE" monitor="NEVER" type="string" name="configuration_file">
3     <value>multi_data_rate_fifo.xml</value>
4   </data>
5 </result>
```

- get argument\_name sub\_system\_name actor\_name

retrieve parameter with name argument\_name from actor named actor\_name in sub system named sub\_system\_name

example : get bytes\_out toto data\_transmitter

```
1 <result cmd="get" status="OK" sub_system_name="toto" actor_name="data_transmitter">
2   <data mode="READ_ONLY" monitor="REQUEST" type="constant_unsigned_integer" size="64" name="bytes_out">
3     <value> 1701280000000</value>
4   </data>
5 </result>
```

- get arguments sub\_system\_name

retrieve list of parameter's name with associated values from sub system coordinator named sub\_system\_name

example : get arguments toto

```
1 <result cmd="get" status="OK" sub_system_name="toto">
2   <data mode="READ_WRITE" monitor="NEVER" type="string" name="name">
3     <value>toto</value>
4   </data>
5   <data mode="WRITE_ONLY" monitor="NEVER" type="action" name="action">
6     <value></value>
7   </data>
8   <data mode="READ_ONLY" monitor="REQUEST" type="action" name="list_actions">
9     <value>STOP</value>
10    <value>PAUSE</value>
11  </data>
12  <data mode="READ_WRITE" monitor="NEVER" type="string" name="configuration_file">
```

```

13     <value>multi_data_rate_fifo.xml</value>
14 </data>
15 <data mode="READ_ONLY" monitor="REQUEST" type="string" name="state">
16     <value>RUNNING</value>
17 </data>
18 <data mode="READ_WRITE" monitor="REQUEST" type="unsigned_integer" size="31" name="run_number">
19     <value> 1</value>
20 </data>
21 </result>

```

- get arguments sub\_system\_name actor\_name

retrieve list of parameter's name with associated values from actor named actor\_name in sub system named sub\_system\_name

example : get arguments toto data\_transmitter

```

1 <result cmd="get" status="OK" sub_system_name="toto" actor_name="data_transmitter">
2   <data mode="READ_ONLY" monitor="NEVER" type="string" name="name">
3     <value>data_transmitter</value>
4   </data>
5   <data mode="READ_WRITE" monitor="REQUEST" type="boolean" name="web_server">
6     <value>FALSE</value>
7   </data>
8   <data mode="READ_ONLY" monitor="NEVER" type="constant_string" name="clients">
9     <value>data_receiver</value>
10    <value>secondary_data_receiver</value>
11  </data>
12  <data mode="READ_ONLY" monitor="NEVER" type="constant_string" name="kind">
13    <value>A_PRODUCER</value>
14  </data>
15  <data mode="READ_WRITE" monitor="NEVER" type="boolean" name="actif">
16    <value>TRUE</value>
17  </data>
18  <data mode="READ_WRITE" monitor="REQUEST" type="unsigned_integer" size="32" name="port">
19    <value> 7080</value>
20  </data>
21  <data mode="READ_WRITE" monitor="REQUEST" type="log_level" name="log_level">
22    <value>INFO</value>

```

```

23 </data>
24 <data mode="READ_ONLY" monitor="NEVER" type="constant_string" name="host_name">
25   <value>power5</value>
26 </data>
27 <data mode="READ_ONLY" monitor="NEVER" type="constant_natural" size="31" name="rank">
28   <value> 1</value>
29 </data>
30 <data mode="READ_WRITE" monitor="NEVER" type="boolean" name="test_param">
31   <value>TRUE</value>
32 </data>
33 <data mode="READ_ONLY" monitor="REQUEST" type="constant_unsigned_integer" size="64" name="bytes_out">
34   <value> 3063948000000</value>
35 </data>
36 </result>

```

- get sub\_systems

retrieve list of sub systems' name from NARVAL NAMING REGISTRY

```

1 <result cmd="get" status="OK">
2   <data sub_system_number="1" type="string" name="sub_systems">
3     <value>toto</value>
4   </data>
5 </result>

```

- get actors sub\_system\_name

retrieve list of actors' name presents in sub system named sub\_system\_name

example : get actors toto

```

1 <result cmd="get" status="OK" sub_system_name="toto">
2   <data actors_number="5" type="string" name="actors">
3     <value>a_stand_alone_example</value>
4     <value>data_transmitter</value>
5     <value>secondary_data_transmitter</value>
6     <value>data_receiver</value>
7     <value>secondary_data_receiver</value>
8   </data>
9 </result>

```

## 2.3 Launch

Object : instantiate a sub system coordinator (new name for chef d'orchestre) Syntax :

- launch sub\_system\_name host\_name

exec a sub system coordinator process to sustain a sub system  
named sub\_system\_name on the machine named host\_name in an  
xterm

```
1 <result cmd="launch" status="ok" sub_system_name="toto"></result>
```

- launch sub\_system\_name host\_name local

same as first command but no xterm possibility

```
1 <result cmd="launch" status="ok" sub_system_name="toto"></result>
```

## 2.4 Set

Object : set parameters values in NARVAL processes.

Syntax :

- set argument value sub\_system\_name

change the value of sub system coordinator parameter named  
argument in sub system named sub\_system\_name

- set argument value sub\_system\_name actor\_name

change the value of actor parameter named argument in  
sub system named sub\_system\_name for actor named actor\_name

## 3 Deprecated commands

- send\_order

## 4 Special commands

These commands shouldn't be called from remote access because the xml output, when there is one, don't comply with what have been presented before in this document.

- load
- sleep
- quit
- group  
create group of actors to set and get parameters
- wait  
may be used in future to emulate asynchronous messages
- echo  
used for script
- dump  
used for debug purpose only, **no xml output**
- Terminate\_coordinator  
used to force narval\_naming\_registry to end if possible **Expert mode**
- special\_domi  
used to force narval state machine in error **Expert mode**

## 5 XML errors format

The error format follow the following scheme in case of “standard” errors :

```
1 <result cmd="command_name" status="ERROR">
2   <message type="standard">
3     <severity>WARNING|ERROR</severity>
4     <text>some relevant text</text>
5   </message>
6 </result>
```

If the error is an unexpected exception the format is :

```
1 <result cmd="command_name" status="ERROR">
2   <message type="unexpected_exception">
3     <exception_name>name of the exception</exception_name>
4     <exception_message>some text relevant to the exception</exception_message>
5   </message>
6 </result>
```