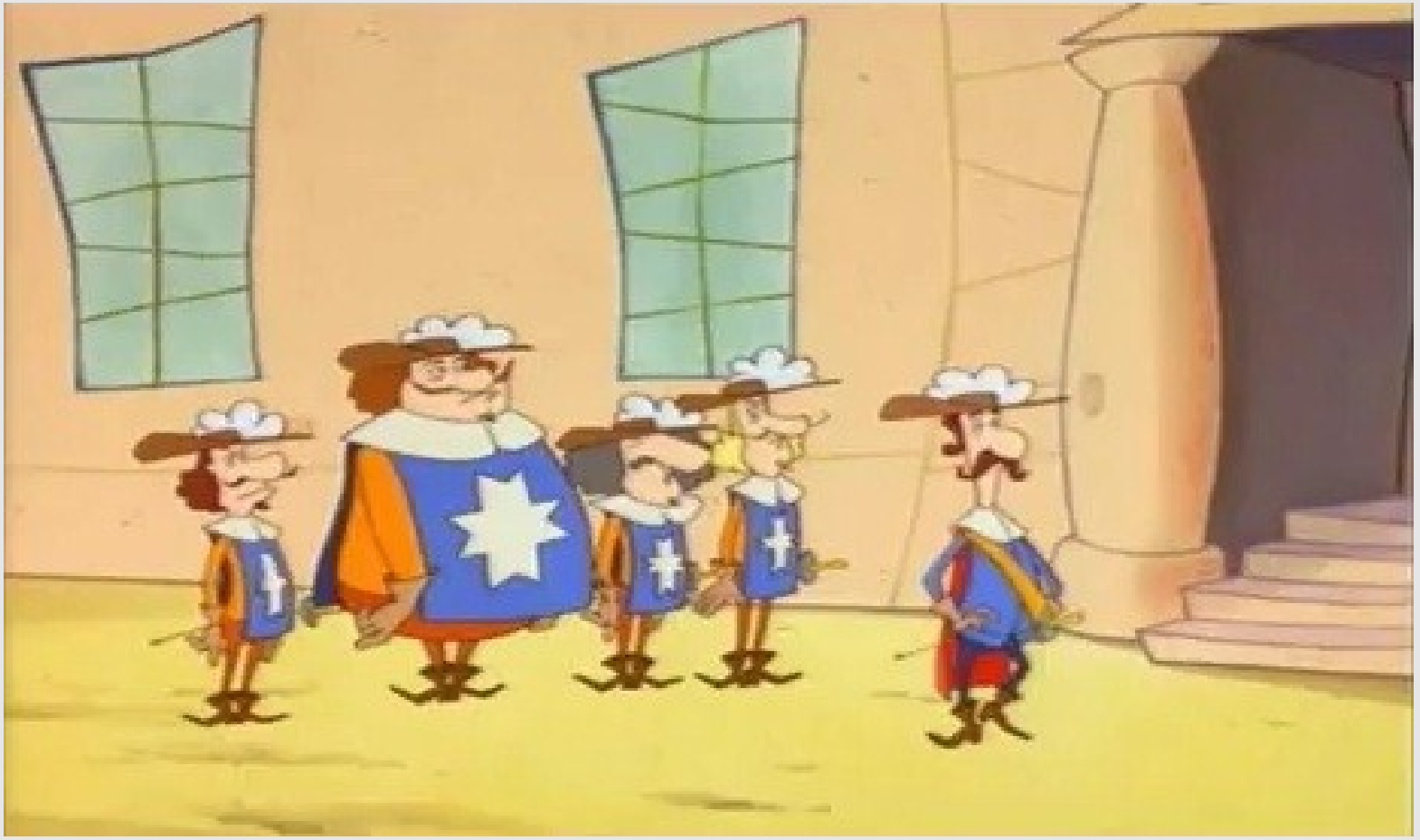
FOSS4G-FR - 2014

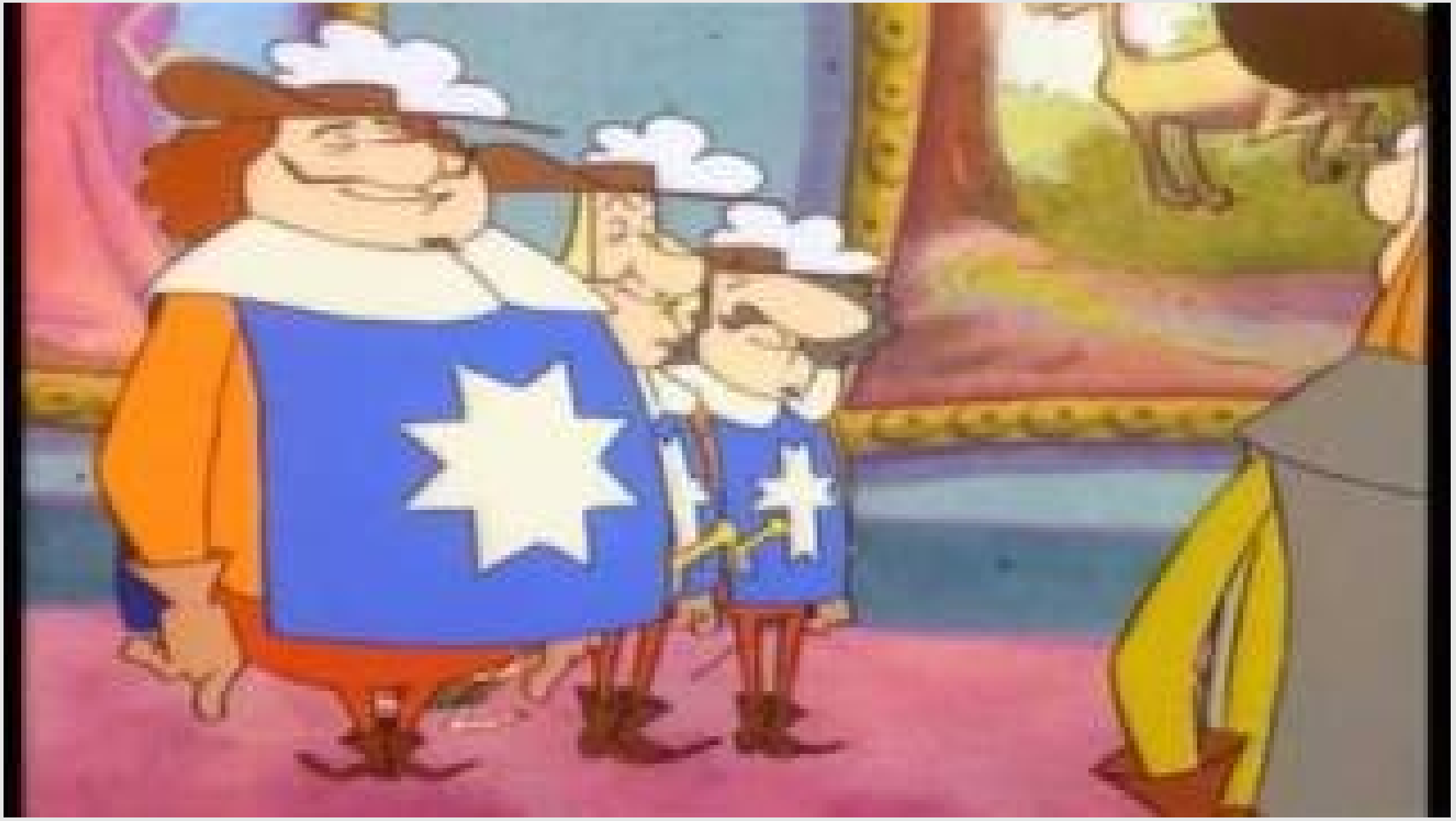
QGIS-Server
et le
Web Processing Service
(WPS)

QGIS-Server et le WPS



Les mousquetaires de l'OGC

Les 3 mousquetaires



WMS, WFS(-T), WCS

D'artagnan

Catalog Service for Web (CS-W)

Il en manque 1

Sans lui, ils sont un peu penaud

Albert de Parmagnan le 5ème Mousquetaire

Le Web Processing Service

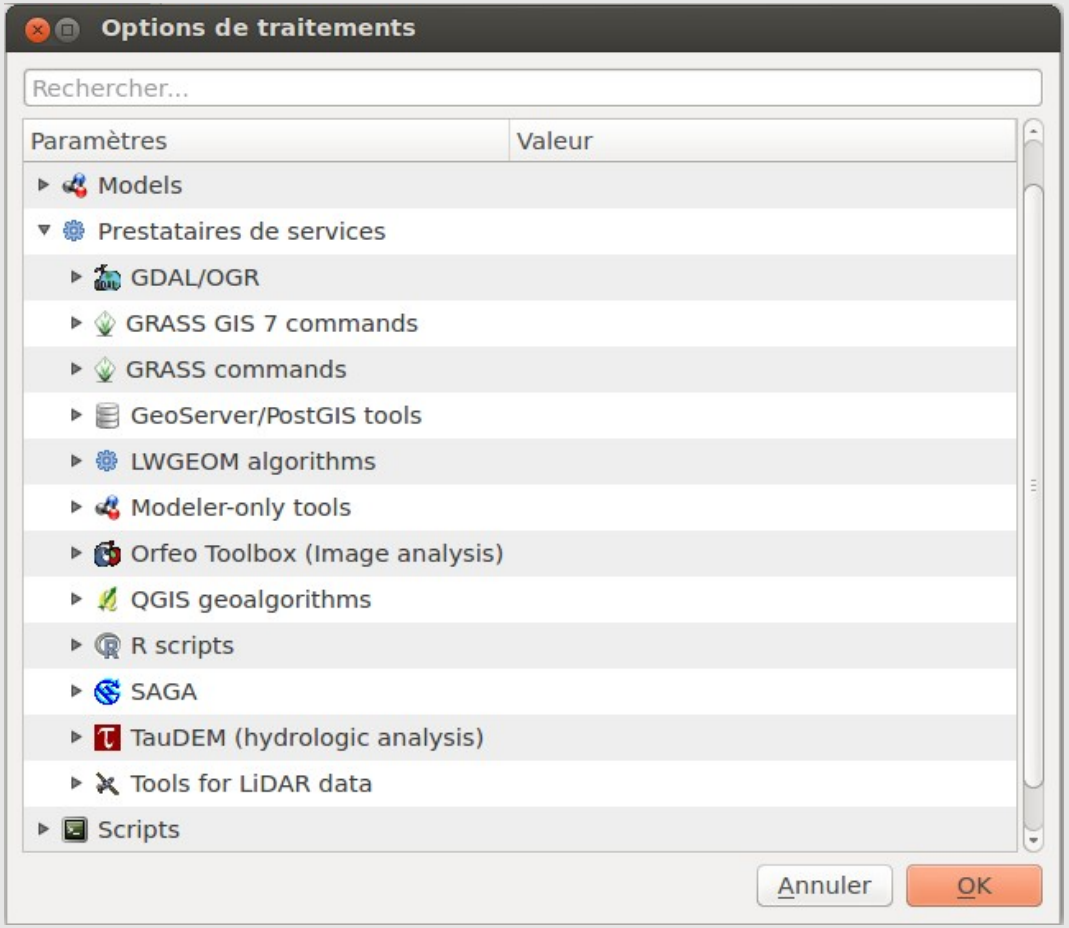
Donc

- QGIS-Server propose
  - WMS : 1.3.0 et 1.1.1
  - WFS + Transaction : 1.0.0
  - WCS : 1.0.0
- Ne propose pas :
  - CS-W
  - WPS...

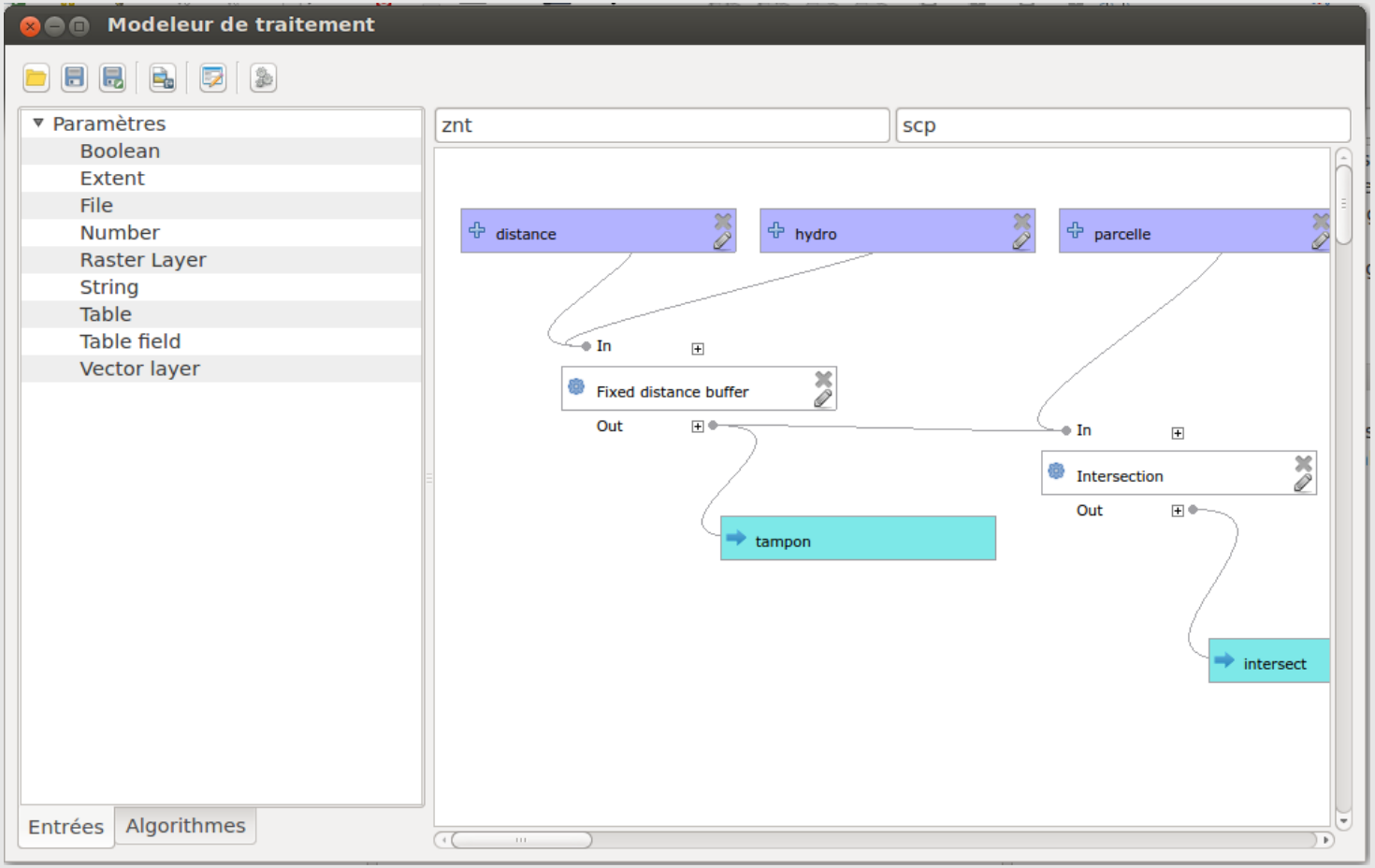
QGIS-Processing

- ETL Spatial
- Entrées
  - Données vecteurs et rasters, fichiers, variables
- Algorithmes
  - Tampon, intersection, modèles scientifiques
- Sorties
  - Données vecteurs, rasters, rapports

QGIS-Processing : Algorithmes



QGIS-Processing : modeleur



QGIS-Processing en WPS

- Commandité par l'Ifremer pour
  - Fédérer
  - Partager
- Des traitements créés par
  - Différents services
  - Différents chercheurs

QGIS-Processing en WPS

- En 3 étapes :
  - QGIS-Processing sur serveur
  - Quelle implémentation ?
  - Quelle méthode de publication des traitements ?

QGIS-Processing sur serveur

Using PyQGIS in custom application

Note: do *not* use `qgis.py` as a name for your test script — Python will not be able to import the bindings as the script's name will shadow them.

First of all you have to import qgis module, set QGIS path where to search for resources — database of projections, providers etc. When you set prefix path with second argument set as **`True`**, QGIS will initialize all paths with standard dir under the prefix directory. Calling **`initQgis()`** function is important to let QGIS search for the available providers.

```
from qgis.core import *

# supply path to where is your qgis installed
QgsApplication.setPrefixPath("/path/to/qgis/installation", True)

# load providers
QgsApplication.initQgis()
```

Now you can work with QGIS API — load layers and do some processing or fire up a GUI with a map canvas. The possibilities are endless :-)

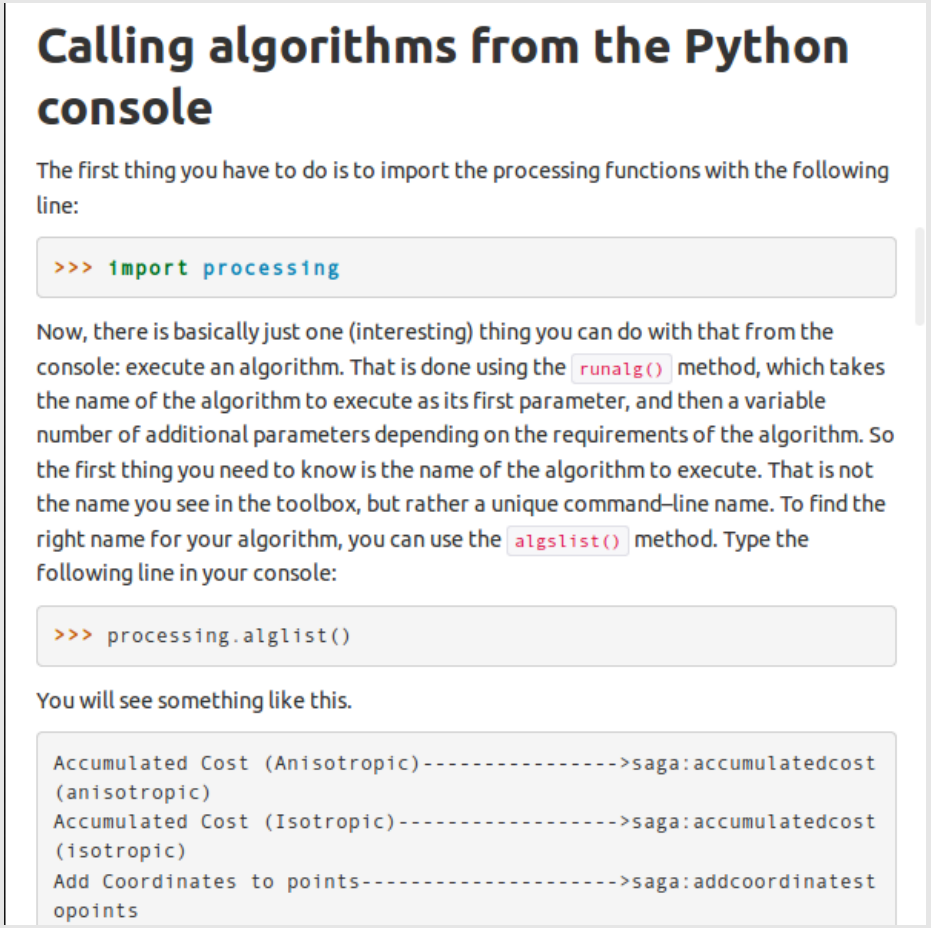
When you are done with using QGIS library, call **`exitQgis()`** to make sure that everything is cleaned up (e.g. clear map layer registry and delete layers):

```
QgsApplication.exitQgis()
```

Running Custom Applications

Un script est une application pyQGIS

QGIS-Processing sur serveur



Calling algorithms from the Python console

The first thing you have to do is to import the processing functions with the following line:

```
>>> import processing
```

Now, there is basically just one (interesting) thing you can do with that from the console: execute an algorithm. That is done using the `runalg()` method, which takes the name of the algorithm to execute as its first parameter, and then a variable number of additional parameters depending on the requirements of the algorithm. So the first thing you need to know is the name of the algorithm to execute. That is not the name you see in the toolbox, but rather a unique command–line name. To find the right name for your algorithm, you can use the `algslist()` method. Type the following line in your console:

```
>>> processing.alglist()
```

You will see something like this.

```
Accumulated Cost (Anisotropic)---------------->saga:accumulatedcost
(anisotropic)
Accumulated Cost (Isotropic)----------------->saga:accumulatedcost
(isotropic)
Add Coordinates to points-------------------->saga:addcoordinatest
opoints
```

Il est possible d'exploiter QGIS-Processing en Python

QGIS-Processing sur serveur

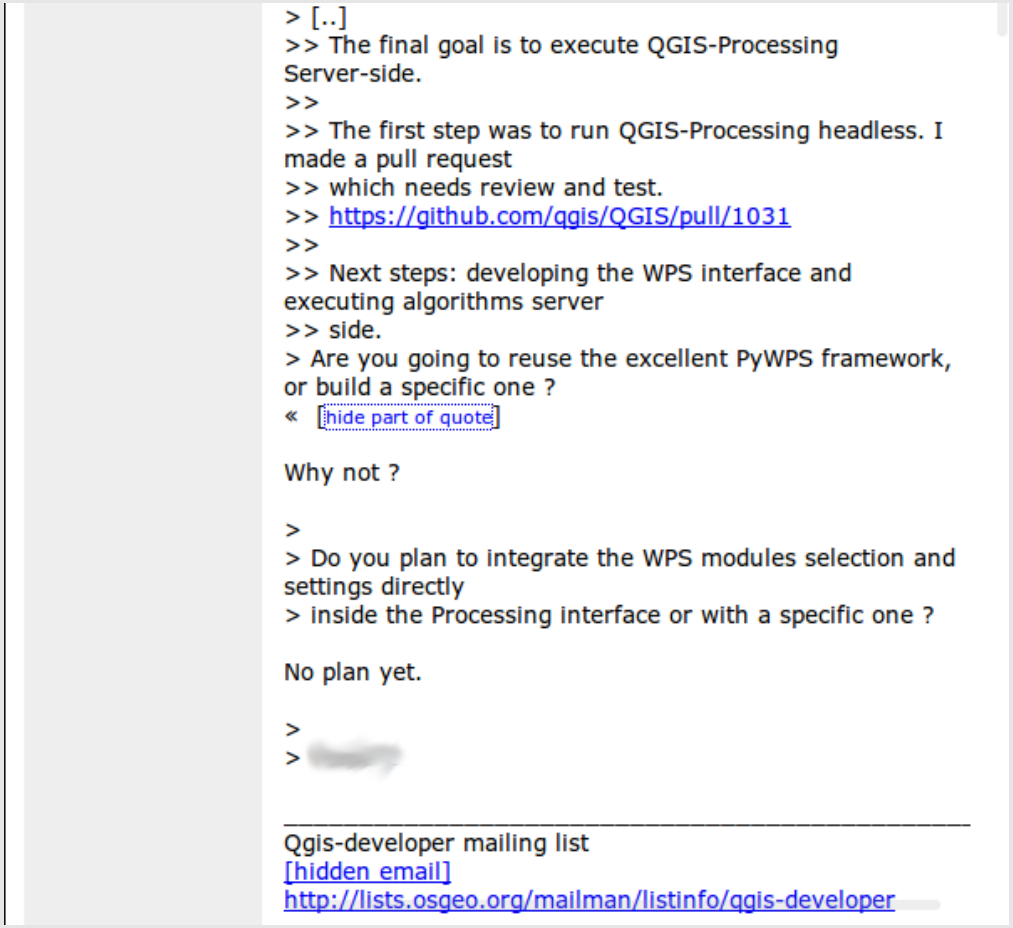
QGIS-Processing n'était pas prêt :

Interface QGIS obligatoire
pyQt obligatoire

QGIS-Processing sur serveur

```
from qgis.core import *
#next PyQt4
from PyQt4.QtCore import *
from PyQt4.QtGui import *
# supply path to where is your qgis installed
QgsApplication.setPrefixPath("/home/[redacted]/qgis_rldhont
/build", True)
# load providers
QgsApplication.initQgis()
# load a project
p = QgsProject.instance()
p.read( QFileInfo( "/home/[redacted]/premier.qgs" ) )
# init QApplication for processing
a = QApplication( sys.argv )
# initialize QGIS-Processing
from processing.core.Processing import Processing
cmd_folder = os.path.split(inspect.getfile(inspect.currentframe()))[0]
if cmd_folder not in sys.path:
    sys.path.insert(0, cmd_folder)
Processing.initialize()
# run algorithm
general.runalg( 'modeler:znt', 5.0, '/home/[redacted]/D
onnees/vecteurs/rivieres.shp', '/home/[redacted]/Donnee
s/vecteurs/uc_simple.shp', '/home/[redacted]/p
ython/wps/results/test_znt.shp' )
# quit
QgsApplication.exitQgis()
```

Quelle implémentation ?



> [..]
>> The final goal is to execute QGIS-Processing
Server-side.
>>
>> The first step was to run QGIS-Processing headless. I
made a pull request
>> which needs review and test.
>> https://github.com/qgis/QGIS/pull/1031
>>
>> Next steps: developing the WPS interface and
executing algorithms server
>> side.
> Are you going to reuse the excellent PyWPS framework,
or build a specific one ?
« hide part of quote

Why not ?

>
> Do you plan to integrate the WPS modules selection and
settings directly
> inside the Processing interface or with a specific one ?

No plan yet.

>
>

Qgis-developer mailing list
[hidden email]
http://lists.osgeo.org/mailman/listinfo/qgis-developer

QGIS-Server ou PyWPS

Solution WPS : QGIS-Server

- Avantage
  - Environnement QGIS clef en main
- Contrainte
  - Charger l'interpréteur Python

Solution WPS : PyWPS

- Avantage
  - Environnement WPS prêt
- Contrainte
  - Charger l'environnement PyQGIS

Proof of Concept : PyWPS

Proof of Concept : PyWPS

```
63          Processing.initialize()
64          # Intialize algorithm
65          alg = Processing.getAlgorithm( "qgis:fixeddistancebuffer" )
66          # add file to the project
67          fileName = self.INPUT.getValue()
68          fileInfo = QFileInfo(fileName)
69          with open(fileName, "r") as f:
70            o = open(fileName+".gml", "w")
71            o.write( f.read() )
72            o.close()
73          layer = QgsVectorLayer( fileName+".gml", fileInfo.baseName(), "ogr"
    )
74          mlr.addMapLayer(layer, False)
75          # run algorithm
76          from processing.tools import general
77          algOutputs = general.runalg( "qgis:fixeddistancebuffer", fileName,
    self.DISTANCE.getValue(), 5, None )
78
79          if algOutputs is None:
80            return "Error in processing"
81
82          outputName = algOutputs["OUTPUT"]
83          outputInfo = QFileInfo(outputName)
84          outputFile = outputInfo.absolutePath()+"/"+fileInfo.baseName()+".gm
    l"
85          outputLayer = QgsVectorLayer( outputName, outputInfo.baseName(), "o
    gr" )
86          error = QgsVectorFileWriter.writeAsVectorFormat(outputLayer, output
    File, "utf-8", None, "GML", False, None, ["XSISCHEMAURI=http://schemas.open
    gis.net/gml/2.1.2/feature.xsd"])
87          self.OUTPUT.setValue( outputFile )
```

Résultat du Proof Of Concept

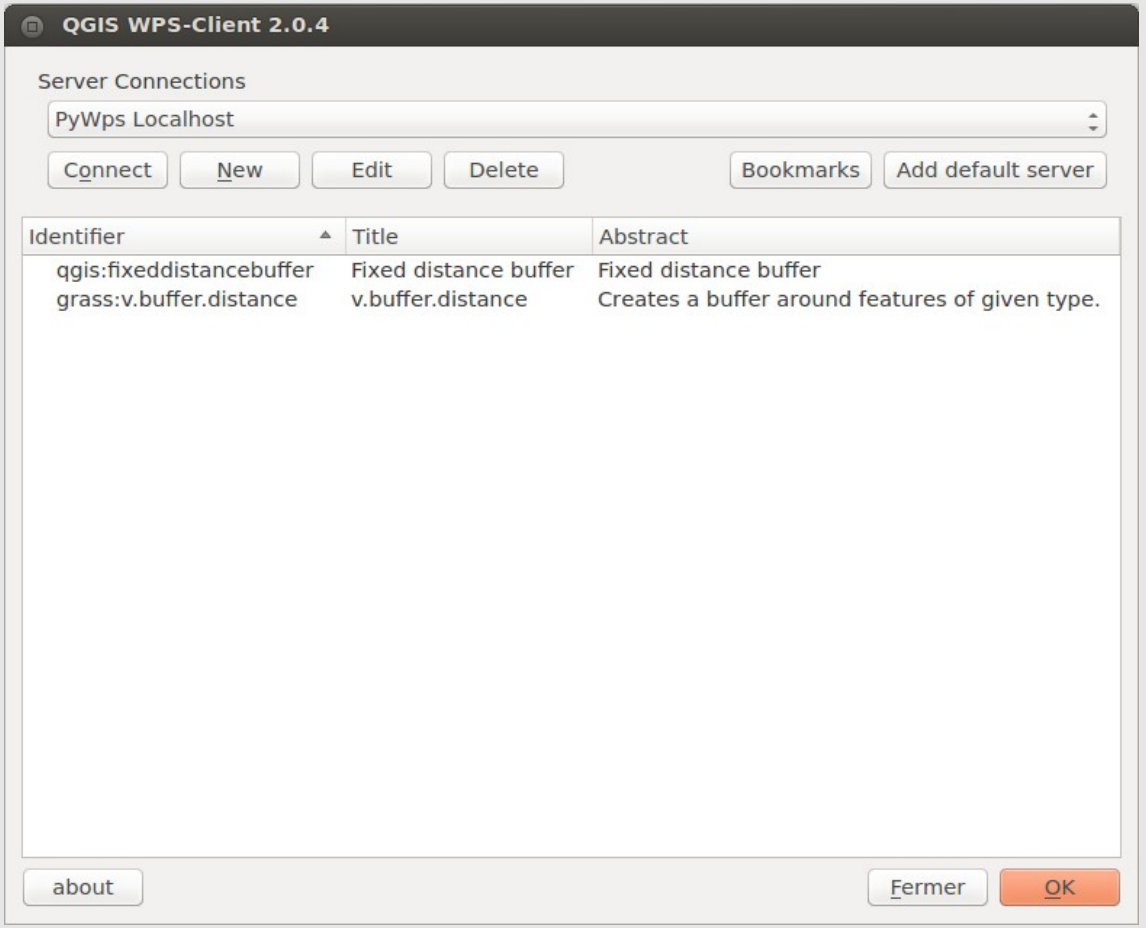
Facilité de déploiement

```
12 from pywps.Process import WPSProcess
13 import qgisWPSProcess
14
15 qgis = qgisWPSProcess.qgisWPSProcess("qgis:fixeddistancebuffer")
16
17 saga = qgisWPSProcess.qgisWPSProcess("saga:shapesbufferfixeddistance")
18
19 grass = qgisWPSProcess.qgisWPSProcess("grass:v.buffer.distance")
20
```

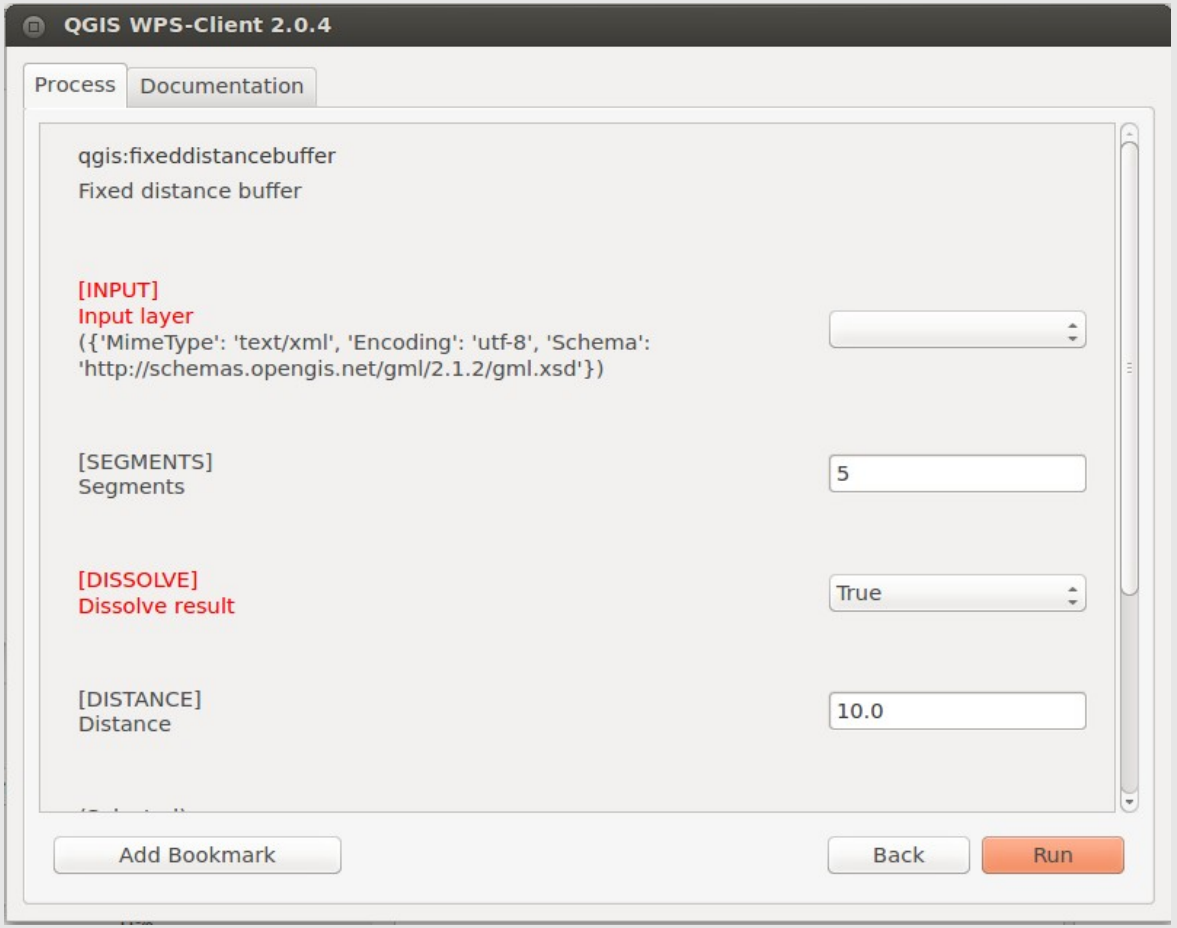
Facilité de déploiement

- Quelques points bloquants
  - Les icônes des fournisseurs
  - Les fenêtres du modeleur
- Utilisation d'une version modifiée de QGIS-Processing pour l'instant

Facilité de déploiement



Facilité de déploiement



Facilité de déploiement

```
12 from pywps.Process import WPSProcess
13 import qgisWPSProcess
14
15 # init QApplication for processing and set the customSettingFolder
16 QgsApplication( sys.argv, False, "/home/
    /python/wps/pywps" )
17
18 # supply path to where is your qgis installed
19 QgsApplication.setPrefixPath("/home/                    /qgis_rldhont
   /build", True)
20
21 # load providers
22 QgsApplication.initQgis()
23
24 from processing.core.Processing import Processing
25 cmd_folder = "/home/                              ./python/wps/pywp
   s"
26 if cmd_folder not in sys.path:
27   sys.path.insert(0, cmd_folder)
28 Processing.initialize()
29
30
31 # filetring text
32 QGISProcesses = {}
33 text = 'znt' #'buffer'
34 for provider in Processing.algs.values():
35     sortedlist = sorted(provider.values(), key=lambda alg: alg.name)
36     for alg in sortedlist:
37         identifier = alg.commandLineName()
38         if text is None or text.lower() in alg.name.lower() or text.lower()
    in identifier.lower() :
39             QGISProcesses[ identifier ] = qgisWPSProcess.qgisWPSProcess( iden
   tifier )
```

Conclusion

- QGIS-Processing est exploitable
  - En mode serveur
  - En mode WPS avec pyWPS
- Tous les algos ne peuvent pas être publiés
  - Problème des sources complexes
  - Problème des options sur-numéraire