
WPS4R

Creating WPS processes via R scripts

STML Meeting on 04/11/2011

Agenda

- About WPS4R
- Input / Output
- Architecture
- Deploy a WPS4R Process
- WPS4R Annotation Syntax
- Further Tasks

About WPS4R

- Module of the **52n WPS**
- Integrated middleware for WPS → R
- Allows WPS process creation via R-scripts
- R scripts contain **annotations**
 - Supplies process description, input and output information
- **Upload function:**
 - Upload processes during server runtime (administrator)
 - Processes are immediately ready to use

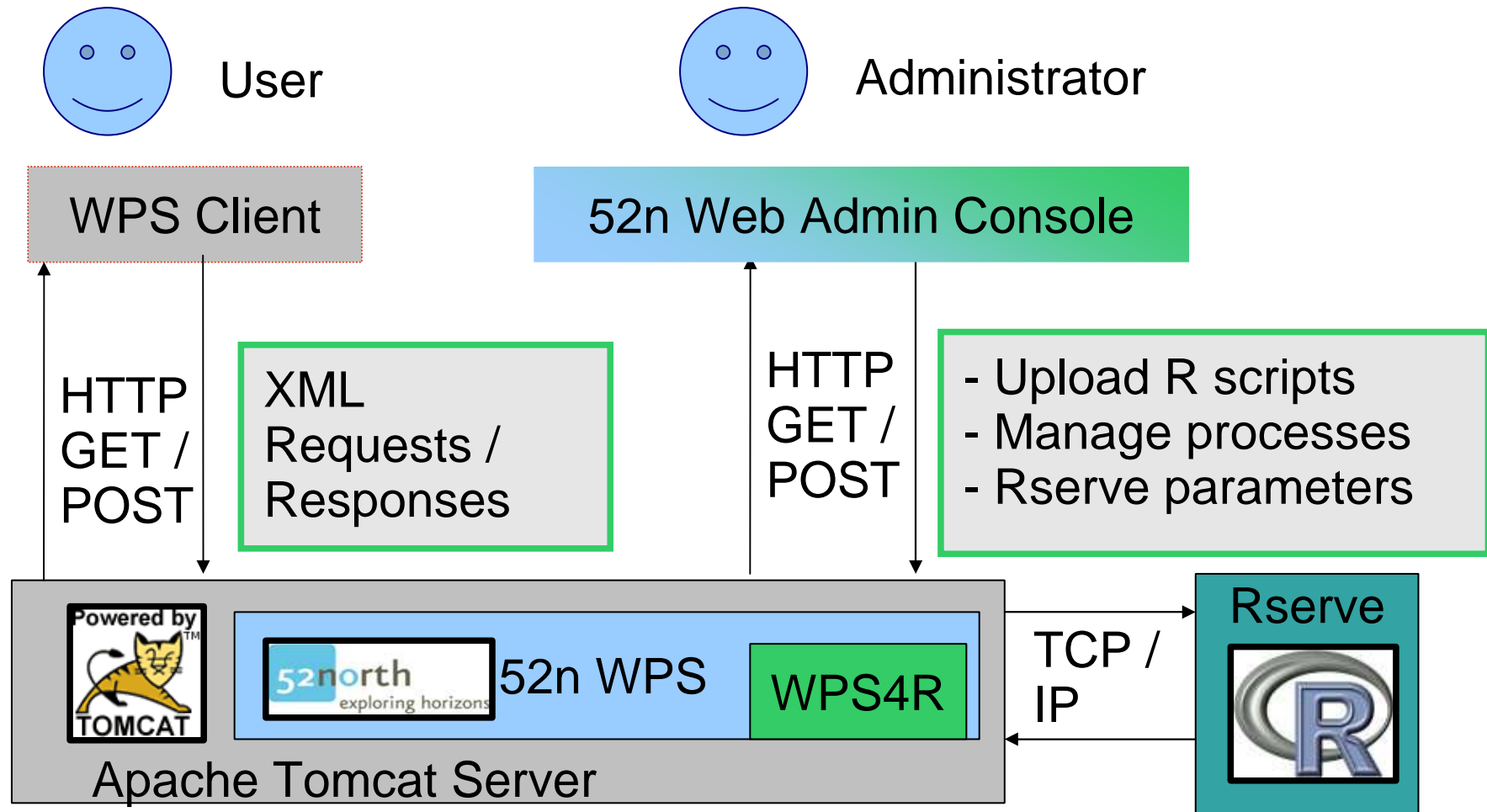
About WPS4R (Rserve backend)

- Processing backend is **Rserve**
- Rserve is an independent TCP / IP Server for R
 - Local or remote connection
 - Allows multiple connections (unix machines only!)
 - Thread safety
- Available as R package
- Binary transport of R objects (faster)

Input/Output

- In principle all data types can be used if...
 - R can read / write them
 - They are formally defined inside WPS4R
- Limitations:
 - No arrays
 - Only one input per identifier
 - One supported type per input / output
- Common spatial data formats (→rgdal drivers, ...)
 - GeoTIFF, ESRI Shapefile, ERDAS Image file, NetCDF, (GML?)
- Simple data types (default values possible)
 - string, double, integer, boolean

WPS4R Architecture



Deploy a WPS4R Process

```
RStudio  
File Edit View Workspace Plots Help  
r_db_examples.R x loadNetCDF.R x intro.R x om2r.R x highlight.R x  
Source on Save  
1 # TODO: R  
2 #  
3 # Author: Matthias Hinz  
4 #####  
5  
6 # wps.des: title="Transforms an R script into HTML/CSS with syntax highlights",  
7 # wps.in: input, type=text, abstract = "R script to highlight";  
8 # wps.out: highlight  
9  
10 output = "output.html"  
11 highlight(file = input, output = output, detective = simple_detective,  
12           renderer = render_html_highlight, parser_output = parser(input, encoding = "UTF-8"))  
13  
14 # wps.out: output, type= text, abstract = "highlighted html text";
```

Annotated R script



52north Web Admin Console

WPS Config Configuration

Save and Activate Config

Server Settings

Algorithm Repositories

Please enter the process name:
(only if process name should be unlike filename)

Please enter the location of an annotated R script

Upload Process | Upload R script

Upload the script to WPS



WPS TestClient

52north exploring horizons

For more information about the 52nd North Web Processing Service visit <http://52north.org/wps>

Service URL:

Request Examples:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes">  
<wps:Execute service="WPS" version="1.0.0">  
  <wps:Request><http://www.opengis.net/ows/1.1*>  
    <wps:RequestLink><http://www.w3.org/1999/xlink xmlns:xlink="http://www.w3.org/2001/XMLSchema-  
      base">  
      <xsi:schemaLocation="http://www.opengis.net/wps/1.0.0  
        http://schemas.opengis.net/wps/1.0.0/wpsExecute_request.xsd">  
        <ows:Identifier>wps.n52.wps.server.r.highlight</ows:Identifier>  
        <wps:DataInput>  
          <wps:Input>  
            <ows:Identifier>input</ows:Identifier>  
            <wps:Reference xlink:href="http://localhost:8080/wps-z/testDocs/ldw.w"  
              mimeType="text/plain" encoding="UTF-8"></wps:Reference>  
          </wps:Input>  
        </wps:DataInput>  
        <wps:ResponseFormat>  
          <wps:RawDataOutput mimeType="text/plain" encoding="UTF-8">  
            <ows:Identifier>output</ows:Identifier>  
          </wps:RawDataOutput>  
        </wps:ResponseFormat>  
      </wps:Request>  
</wps:Execute>
```

Run the script as a Web Service process

Deploy a WPS4R Process

- Write an R script
- Declare Process inputs (`# wps.in: ...`)
- Declare process output (`# wps.out: ...`)
- Add general process information (`#wps.des: ...`)
- Upload script via Web Admin Console
 - Execute process

Deploy a WPS4R Process

1. Write an R script

```
# random number:
```

```
min = 0
```

```
max = 1
```

```
output = runif(1, min=min, max=max)
```

Deploy a WPS4R Process

2. Declare inputs (e.g. identifier, type, default value)

```
# wps.in: min, double, value = 0;  
# wps.in: max, double, value = 1;
```

```
# random number:
```

```
min = 0
```

```
max = 1
```

```
output = runif(1, min=min, max=max)
```

Deploy a WPS4R Process

3.a Declare output (e.g. identifier, type)

```
# wps.in: min, double, value = 0;  
# wps.in: max, double, value = 1;  
  
# random number:  
output = runif(1, min=min, max=max)  
  
# wps.out: output, double;
```

Deploy a WPS4R Process

3.b Declare output (Solution for complex output: textfile)

```
# wps.in: min, double, value = 0;  
# wps.in: max, double, value = 1;
```

```
# random number:
```

```
x = runif(100, min=min, max=max)
```

```
output = "outputfilename"
```

```
write.table(x, output)
```

```
# wps.out: output, text;
```

Deploy a WPS4R Process

4. Add process description

```
# wps.des: id = R_random, title = Random number
# generator, abstract = Generates a single
# random number within a defined interval;
# wps.in: min, double, value = 0;
# wps.in: max, double, value = 1;

# random number:
  output = runif(1, min=min, max=max)

# wps.out: output, double;
```

Deploy a WPS4R Process

5. More process metadata (title, abstract for input / output)

```
# wps.des: id = R_random, title = Random
# number generator, abstract = Generates a single
# random number for uniform distribution;
# wps.in: min, double,
# Minimum, All outcomes are larger than min,
# value = 0;
# wps.in: max, double,
# Maximum, All outcomes are smaller than max,
# value = 1;
  output = runif(1, min=min, max=max)

# wps.out: output, double, Random number;
```

Deploy a WPS4R Process

6. Save script, upload it via WPS Admin Console

The screenshot displays the '52north Web Admin Console' interface. The main content area is titled 'WPS Config Configuration' and 'WPS Test Client'. A modal dialog box is open, prompting the user to enter a process name and the location of an annotated R script. The dialog includes input fields for 'Server Host Name', 'Server Host Port', 'Include Datainput', 'Computation Timeout', 'Cache Capabilities', and 'Web app Path'. It also features a search button labeled 'Durchsuchen...' and buttons for 'Daten absenden' and 'Cancel'. A red arrow points to the 'Upload R script' button in the background interface.

52north Web Admin Console

WPS Config Configuration WPS Test Client

Save and Activate Configuration Upload Process Upload R script

Server Settings

Server Host Name: Server Host Port: Include Datainput: Computation Timeout: Cache Capabilities: Web app Path:

Please enter the process name:
(only if process name should be unlike filename)

Please enter the location of an annotated R script.

Durchsuchen...

Daten absenden Cancel

Process id will be *org.n52.wps.server.r.[filename]* or *org.n52.wps.server.r.[process name]*

Algorithm Repositories
Parsers
Generators

Upload Button

WPS4R Annotation Syntax

wps.des: id, title = id, abstract = null;

→ General process description

**wps.in: id, type, title = id, abstract = null,
value = null, minOccurs = 1, maxOccurs = 1;**

→ Input description

→ WPS4R: value initialization

wps.out: id, type, title = id, abstract = null;

→ Output description

→ WPS4R: returns value / file

Further Tasks

- Online documentation
- Trunk merge
- Modifications according to use cases

Thanks for your attention.
Questions?

Useful Links

- About the 52n WPS:
<http://52north.org/communities/geoprocessing/wps/index.html>
- About Rserve:
<http://www.rforge.net/Rserve/>
- WPS4R SVN repository
<https://svn.52north.org/svn/geoprocessing/main/WPS/branches/WPS-R-Project>

(More information will follow)

```
# wps.des: title = WPS IDW Demo, abstract = idw process
demo for WPS;
# wps.in: points, shp;
# wps.in: raster, img;
# wps.in: attributename, string;
# wps.in: nmax, integer, value = 10, abstract = Optional
input with default value;

points=readOGR(points,sub(".shp", "", points))
raster=readGDAL(raster)

# inverse distance interpolation:
form=formula(paste(attributename, "~ 1"))
idw=idw(form,points,raster, nmax = nmax)
idw@data=data.frame(idw@data$var1.pred)
# parse output, return filepath
output=writeGDAL(idw,"output.img", drivername="HFA",
mvFlag = 0)
result=paste(getwd(),output,sep="/")
# wps.out: result, img;
```