deegree Web Processing Service v2.4

lat/lon GmbH
Aennchenstr. 19
53177 Bonn
Germany
Tel ++49 - 228 - 184 96-0
Fax ++49 - 228 - 184 96-29
info@lat-lon.de
www.lat-lon.de

Dept. of Geography
Bonn University
Meckenheimer Allee 166
53115 Bonn

Tel. ++49 228 732098

Change log

Date	*Description*	*Author*
2008-03-11	Initial Document setup	Christian Kiehle
2008-04-15	Document finalized	Christian Kiehle
2009-12-02	Update to v2.3	Sebastian Goerke

Table of Contents

Index of Tables

Illustration Index

1 Introduction

deegree is a Java Framework offering the main building blocks for Spatial Data Infrastructures (SDIs). Its entire architecture is developed using standards of the Open Geospatial Consortium (OGC) and ISO Technical Committee 211 – Geographic information / Geoinformatics (ISO/TC 211). deegree encompasses OGC Web Services as well as clients. deegree is Free Software protected by the GNU Lesser General Public License (GNU LGPL) and is accessible at http://www.deegree.org/.

deegree2 is the new release of deegree supporting a number of features that deegree1 was not able to handle. This documentation describes setup and configuration of the deegree Web Processing Service (WPS), an implementation of OGC's Web Processing Service Deprecated Request for Comments Version 0.4.0 (OGC Document # 05-007r4).

deegree's WPS is able to process Feature Collections based on arbitrary processes. OGC's WPS (Schut & Whiteside 2005) specification describes WPS as follows: "WPS defines a standardized interface that facilitates the publishing of geospatial processes, and the discovery of and binding to those processes by clients. "Processes" include any algorithm, calculation or model that operates on spatially referenced data. "Publishing" means making available machine-readable binding information as well as human-readable metadata that allows service discovery and use. "

Therefore the following WPS requests are supported:

- GetCapabilities
- DescribeProcess
- Execute

Although WPS is not restricted to serve processes based on spatial and/or temporal data it is meant to be a specification for processing mainly spatial and/or temporal data.

The WPS Server acts as a container for an unlimited number of processes. The relation of a WPS Server to a WPS process comprises the same relation as the WMS Server to a WMS Layer.

Besides the WPS, deegree comprises a number of additional services and clients. A complete list of deegree components can be found at:

`http://www.lat-lon.de` → Products

Downloads of packaged deegree components can be found at:

`http://www.deegree.org` → Download

deegree's Web Processing Service offers great flexibility regarding it's configuration and output formats. Although the current implementation is only a basic implementation of the specification it is easily enhanced to process data beyond Feature Collections.

WPS, like any other web service of deegree, is realized as a Java module controlled by one central servlet (the "dispatcher"). This servlet has to be deployed to the respective web server/servlet engine. Most of the common web servers support servlet technology, thus making deegree a universal product. The Apache-Tomcat 5.5 Servlet-Engine is recommended

due to its widespread use and its status as an open-source product.
Processes should be implemented using Java 1.5. Later versions of the Java programming language should work as well but will not be supported within deegree 2.2.
The WPS server API is – unlike data oriented services like WFS and WCS or portrayal oriented services like WMS and WPVS – mainly geared to software developers. This documentation is intended to get you quickly up and running a sample process (spatial buffer) and point you to the relevant parts of the API to implement your own processes with deegree.

2 Download / Installation

Running deegree2.2 Web Processing Service requires the following components to be installed:

- Java (JRE or JSDK) version 1.5.x
- Tomcat 5.5.x

For installation of these components refer to the corresponding documentation at http://java.sun.com and http://tomcat.apache.org.
deegree Web Processing Service can be downloaded from http://www.deegree.org. The release is packed as a WAR-archive. Simply put this file into your `$TOMCAT_HOME$/webapps` directory and (re-)start Tomcat. The installation of deegree WPS is already done with this.
Note: It is also possible to extract the WAR archive into another place of your computer and direct Tomcat to this place. Because of this possibility, in the remainder of this document, the directory you extracted the files to is referred to as `$wps_home$` (=`$TOMCAT_HOME$/webapps/deegree-wps` in the standard case).

Your `$wps_home$` will contain the following structure:

directory	Content
./WEB-INF	Required by Tomcat, containing all libraries, configuration- and data-files
./WEB-INF/conf/wps/processConfigs	XML configuration files for in-detail description of a process (input / output parameters, etc.)

Table 1: Directory structure of the WPS release

After deploying the deegree-wps into Tomcat servlet container you should be able to access the URL http://localhost:8080/deegree-wps/
To ensure a successful deployment process you should request the capabilities of the service:
http://localhost:8080/deegree-wps/services?
service=WPS&version=0.4.0&request=GetCapabilities
This link is provided within the generic client software through http://localhost:8080/deegree-wps/
Another HTTP-GET-based request is provided through the DescribeProcess interface. A process description is being returned by providing the process identifier of one or more processes. Process identifiers can be found inside the WPS Capabilities response inside the ProcessOfferings section. The preconfigured spatial buffer process is identified by 'buffer'. This results in the following DescribeProcess request:
http://localhost:8080/deegree-wps/services?
service=WPS&version=0.4.0&request=DescribeProcess&Identifier=Buffer
The execute interface is only accessible through HTTP-POST and requires an XML payload in the request. An example can be found through the generic client interface.

3 Architecture

Figure 1 shows the overall architecture of the modules involved in the deegree WPS. The HTTP interface is realized by a servlet that has to be registered into a servlet engine like Tomcat or Jetty. The servlet chooses a handler class depending on the incoming request that delegates it to the responsible service (in this case the WPService). Depending on the requested processes the WPService decides which process is responsible for handling it. The central configuration document of the deegree WPS is based on a OGC WPS 0.4.0 capabilities document plus a few custom tags as well as a OGC WPS 0.4.0 process description document plus a few custom tags.

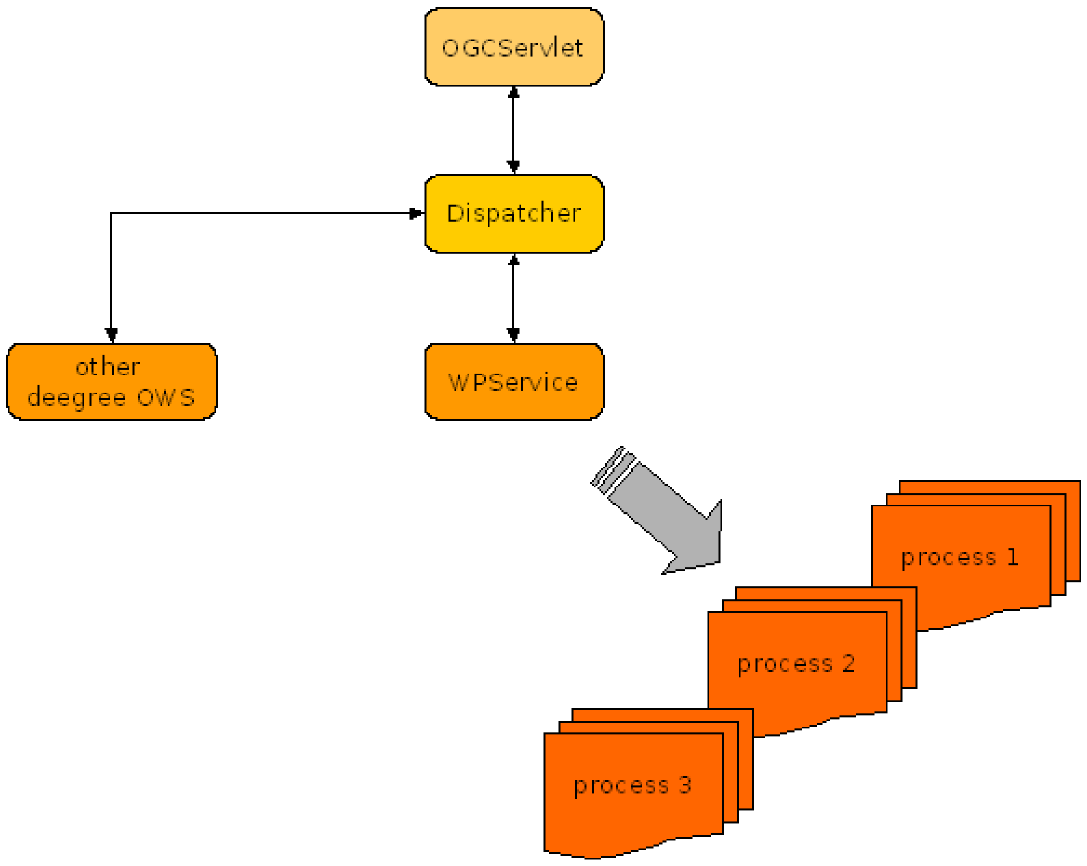

Figure 1: deegree WPS architecture overview

4 Configuration

Figure 2 shows the relationships between different configuration files that determine the behaviour of the deegree WPS.

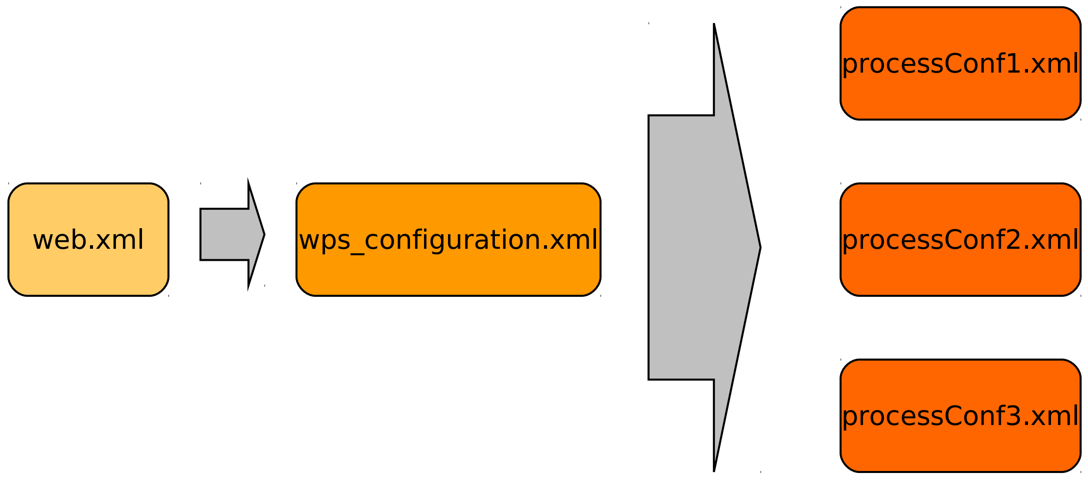

Figure 2: Files affecting the deegree WPS configuration

The WPS-configuration/capabilities document and process configuration files will be described in detail in the following.

The format of the main configuration document of the deegree WPS is based on the OGC WPS 0.4.0 capabilities document plus a few custom elements. deegree-specific elements reside inside the deegree-Namespace (e.g. <deegree:deegreeParams>).

```
<deegree:deegreeParams>
    <deegree:DefaultOnlineResource xmlns:xlink="http://www.w3.org/1999/xlink"
      xlink:type="simple" xlink:href="http://localhost:8080/wps_deegree/wps"/>
    <deegree:CacheSize>100</deegree:CacheSize>
    <deegree:RequestTimeLimit>35</deegree:RequestTimeLimit>
    <deegree:ProcessDirectoryList>
        <deegree:ProcessDirectory>processConfigs</deegree:ProcessDirectory>
    </deegree:ProcessDirectoryList>
    <deegree:RequestQueueManager>
        <deegree:ResponsibleClass>org.deegree.ogcwebservices.wps.execute.DefaultReq
        uestQueueManager</deegree:ResponsibleClass>
    </deegree:RequestQueueManager>
</deegree:deegreeParams>
```

Within the <deegree:deegreeParams> the following options are configured:

- The `deegree:CacheSize` element allows to set the size of cache available for storing feature instances in memory. ***This feature is not implemented yet***.
- To avoid that the service can be blocked altogether by extremely time-consuming requests, a maximum time limit for request processing can be defined using the `deegree:RequestTimeLimit` element. If the request processing exceeds this limit a timeout exception will be returned instead of the requested result.

- The `deegree:ProcessDirectory` points to a directory which will be scanned recursively for configuration documents. The directory resides in an relative path from the main configuration document.
- The `deegree:RequestQueueManager` is responsible for storing requests. This feature will be essential at the time the server supports storage of output. Any RequestQueueManager implementation shall implement the org.deegree.ogcwebservices.wps.execute.RequestQueueManagerinterface The current DefaultRequestQueueManager simply stores requests to a map (java.util.Map) and retrieves them afterwards from the map.

Each process is described and configured by a process configuration xml file (as referenced inside the wps configuration document. Besides the definition of input/output data types (for details have a look at the WPS specification) there is a single deegree-specific parameter as well:

- `deegree:responsibleClass`: The responsibleClass represents the actual process implementing class, identified by it's fully qualified classname. Each process shall only define one responsible class. Although it is of course possible to set up a detailed sub-package structure below the responsible class.

5 Implement your own process

The OGC Web Processing Service specification is intended mainly for developers of geoprocessing routines. It is assumed that you, before starting to build your own processes, have a good grasp on Java and XML. deegree WPS offers the basic building blocks to reflect the data types as defined inside the OGC WPS specification.
The starting point of any process development is the abstract class `org.deegree.ogcwebservices.wps.execute.Process.`
Process offers one public method with the signature

```
public abstract ProcessOutputs execute( Map<String, IOValue> inputs,
                       OutputDefinitions outputDefinitions )  throws
                       OGCWebServiceException;
```

A process implementation has to derive from this abstract class and provide an implementation of the execute method. Furthermore, for each process a process configuration document has to be provided and put into the processConfig directory. deegree will scan this document for the definition of the responsible class and register the to the WPS. As a developer, you do not have to take care about neither the describeProcess nor the GetCapabilities interface programmatically.

6 References

Schut, P.; Whiteside, A. (Eds.) (2005): OpenGIS® Web Processing Service. OGC 05-007r4. Online: http://portal.opengeospatial.org/files/?artifact_id=12184