Open Geospatial Consortium, Inc.

Date: 2009-07-24

Reference number of this document: OGC 09-041r3

Version: 0.3.0

Category: OGC® Public Engineering Report

Editor: Bastian Baranski

OGC® OWS-6 WPS Grid Processing Profile Engineering Report

Document type:	Public Engineering Report
Document subtype:	NA
Document stage:	Approved for public release
Document language:	English

Preface

This Engineering Report was created as a deliverable for the OGC Web Services, Phase 6 (OWS-6) initiative of the OGC Interoperability Program. This document presents the work completed with respect to the Geoprocessing Workflow (GPW) activities within OWS-6.

This document is not a normative.

Suggested additions, changes, and comments on this draft report are welcome and encouraged. Such suggestions may be submitted by email message or by making suggested changes in an edited copy of this document.

The changes made in this document version, relative to the previous version, are tracked by Microsoft Word, and can be viewed if desired. If you choose to submit suggested changes by editing this document, please first accept all the current changes, and then make your suggested changes with change tracking on.

Forward

OWS-6 Testbed

OWS testbeds are part of OGC's Interoperability Program, a global, hands-on and collaborative prototyping program designed to rapidly develop, test and deliver Engineering Reports and Chnage Requests into the OGC Specification Program, where they are formalized for public release. In OGC's Interoperability Initiatives, international teams of technology providers work together to solve specific geoprocessing interoperability problems posed by the Initiative's sponsoring organizations. OGC Interoperability Initiatives include test beds, pilot projects, interoperability experiments and interoperability support services - all designed to encourage rapid development, testing, validation and adoption of OGC standards.

In April 2008, the OGC issued a call for sponsors for an OGC Web Services, Phase 6 (OWS-6) Testbed activity. The activity completed in June 2009. There is a series of on-line demonstrations available here: http://www.opengeospatial.org/pub/www/ows6/index.html

The OWS-6 sponsors are organizations seeking open standards for their interoperability requirements. After analyzing their requirements, the OGC Interoperability Team recommended to the sponsors that the content of the OWS-6 initiative be organized around the following threads:

1. Sensor Web Enablement (SWE)
2. Geo Processing Workflow (GPW)
3. Aeronautical Information Management (AIM)
4. Decision Support Services (DSS)
5. Compliance Testing (CITE)

The OWS-6 sponsoring organizations were:

- U.S. National Geospatial-Intelligence Agency (NGA)
- Joint Program Executive Office for Chemical and Biological Defense (JPEO-CBD)
- GeoConnections - Natural Resources Canada
- U.S. Federal Aviation Agency (FAA)
- EUROCONTROL
- EADS Defence and Communications Systems
- US Geological Survey
- Lockheed Martin

- BAE Systems
- ERDAS, Inc.

The OWS-6 participating organizations were:

52North, AM Consult, Carbon Project, Charles Roswell, Compusult, con terra, CubeWerx, ESRI, FedEx, Galdos, Geomatys, GIS.FCU, Taiwan, GMU CSISS, Hitachi Ltd., Hitachi Advanced Systems Corp, Hitachi Software Engineering Co., Ltd., iGSI, GmbH, interactive instruments, lat/lon, GmbH, LISAsoft, Luciad, Lufthansa, NOAA MDL, Northrop Grumman TASC, OSS Nokalva, PCAvionics, Snowflake, Spot Image/ESA/Spacebel, STFC, UK, UAB CREAF, Univ Bonn Karto, Univ Bonn IGG, Univ Bunderswehr, Univ Muenster IfGI, Vightel, Yumetech.

Contents

Page

Figures Page

Tables Page

OGC® OWS-6 WPS Grid Processing Profile Engineering Report

1 Introduction

1.1 Scope

This OGC™ Engineering Report describes and reviews the Grid Computing related activity completed during the OGC OWS-6 Interoperability testbed. The document describes the WPS processes deployed in the different demonstration scenarios and offers recommendations to the OGC community as to how to better harmonize the standards work of the OGC with Grid Computing platforms and related concepts and technologies.

The original goal of the OWS GRID activity was to investigate, define and implement an OGC WPS Grid Processing Profile that is integrated with a grid computing infrastructure using relevant specifications. As such, this document the participants should record the findings and recommendations resulting from the effort to implement the WPS Grid Processing Profile service. Due to the nature of the WPS standard, in the majority of cases it is not necessary to create specific profiles (but see §5.2.3 defining conventions for including JSDL-related parameters in a WPS request). The team found that it is mostly sufficient to use the existing capabilities described in the WPS standard for accessing distributed Grid resources. Thus, instead of developing several candidate profiles for the WPS, this document reviews the Grid Computing related WPS processes that were implemented in the OWS-6 GPW thread.

This OGC™ document gives guidelines for users and developers who would like to implement a WPS with access to distributed resources in a Grid Computing backend.

1.2 Document contributor contact points

All questions regarding this document should be directed to the editor or the contributors:

Name	Organization	E-Mail
Bastian Baranski	University of Münster, Institute for Geoinformatics (IfGI)	baranski@uni-muenster.de
Arif Shaon	Science and Technology Facilities Council (STFC)	arif.shaon@stfc.ac.uk
Andrew Woolf	Science and Technology Facilities Council (STFC)	andrew.woolf@stfc.ac.uk
Stefan Kurzbach	University of Hamburg, Department of River and Coastal Engineering	stefan.kurzbach@tuhh.de

1.3 Revision history

Date	Release	Editor	Primary clauses modified	Description
2008-12-15	0.0.1	Bastian Baranski	Initial document	Initial document
2009-04-14	0.0.2	Bastian Baranski	5, and various others	Incorporated content provided by STFC
2009-04-15	0.0.3	Bastian Baranski	5, and various others	Incorporated content provided by GIS.FCU
2009-04-16	0.0.4	Bastian Baranski	7, and various others	Incorporated content provided by TUHH
2009-04-27	0.0.5	Andrew Woolf Arif Shaon	5, 7, and various others	Incorporated thoughts about WSRF and general review
2009-04-27	0.1.0	Bastian Baranski	All	Incorporated several reviews and finalization of document
2009-07-09	0.3.0	Carl Reed	Various	Prepare for publication

1.4 Future work

Improvements in this document are desirable to:

- **Web Service Resource Framework**

  Future work with WSRF and WPS could be incorporated into revisions of this document, or else described separately. In section 6.3 some specific limitations of the OGC standards that would benefit from a refactoring under WSRF where quoted.

In general, the WPS semantics could also be captured by things like SAGA (Simple API for Grid Applications)[1] and GridRPC[2]. Therefore, the G-OWS[3] group is looking at

[1] http://forge.gridforum.org/sf/projects/saga-rg

[2] http://www.ogf.org/documents/GFD.102.pdf

implementing the basic OGC standards on the EGEE software stack, gLite[4]. They are not doing WPS - at least not now - but presumably WPS could be hosted on EGEE.

Interesting topics related to Grid Computing and interesting for further research activities (maybe in further OWS Interoperability testbeds) are:

- **Service Level Agreements**

  The availability of OWS and OWS-based SDI's rapidly grow for the last decade and they'll supposable play a major role in emerging business models. Monitoring the performance of loosely coupled Web Services in a distributed infrastructure and the ability to react quickly on service quality fluctuations is overall an essential skill for service consumers and service providers. Particularly with regard to the Infrastructure for Spatial Information in the European Community (INSPIRE) directive, in which several performance criteria requirements for geospatial network services are defined. To make OWS and OWS-based SDI's ready for commercial usage, the OGC Geo Rights Management (GeoRM) Working Group for example aims at establishing a trusted infrastructure for purchasing, managing and protecting rights of digital content. But existing OWS specifications and standards do not support any kind of service quality enforcement functionality. Therefore, normally a formal contract between a service consumer and a service provider - the Service Level Agreement (SLA) - is negotiated [14]. The SLA formalizes a business relationship and enables contractual parties to measure, manage and enforce certain Quality of Service (QoS) guarantees. Therefore, attaching SLA functionality to OWS will pose a great advancement for future business models. Grid Computing is a promising technology for actively enforcing the promised service quality goals from within the SLAs.

- **Cloud Computing**

  The emerging term Cloud Computing as one of the latest trends in the mainstream IT overlaps with some concepts of distributed and Grid Computing [9]. It uses a cloud image to represent the internet or other large networking infrastructures. The key characteristics of the cloud are the ability to scale and provision computing power dynamically in a cost efficient way and the ability of the consumer to make the most of that power without having to manage the underlying complexity of the technology. These characteristics lead to a set of core value propositions like scalability on demand, streamlining the data center, improving business processes and minimizing startup costs for new business model. In essence, the term Cloud Computing collects a family of well known methods and technologies (like Software as a Service (SaaS), Virtualization and Grid Computing) and describes a paradigm of outsourcing specific tasks to a scalable infrastructure and consequently enabling new business models. There are a number of open issues for cloud computing (for example see the so-called

---

[3] http://meetingorganizer.copernicus.org/EGU2009/EGU2009-3424.pdf

[4] http://glite.web.cern.ch/glite/

"Open Cloud Manifesto"[5]). However, the Cloud Computing paradigm is promising for geospatial applications to enable new and promising business models.

1.5 References

The following documents are referenced in this document. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. For undated references, the latest edition of the normative document referred to applies.

OGC 05-007r7, *OpenGIS® Web Processing Service Standard*

OGC 06-094r1, *OWS Common change request: Add SOAP encoding*

OGC 06-182r1, *OWS-4 WPS IPR - Discussions, findings, and use of WPS in OWS-4*

OGC 06-121r3, *OpenGIS® Web Services Common Standard*

NOTE This OWS Common Standard contains a list of normative references that are also applicable to this Implementation Standard.

OGC 07-158, *Wrapping OGC HTTP-GET and -POST Services with SOAP - Discussion Paper*

OGC 08-009r1, *OWS 5 SOAP/WSDL Common Engineering Report*

OGF GFD-R-P.114, *HPC Basic Profile*

OGF GFD-R.056, *Job Submission Description Language (JSDL)*

OGF GFD-R-P.108, *OGSA Basic Execution Service (OGSA-BES)*

OGF GFD-R-P.111, *JSDL HPC Profile Application Extension*

OGF GFD-R-P.135, *HPC File Staging Profile*

OGF GFD-I.100, *HPC Job Scheduling: Base Case and Common Cases*

WS-I, *Basic Security Profile*

NOTE This specification incorporates several other important specifications by reference and defines extensibility points within them

2 Terms and definitions

For the purposes of this report, the definitions specified in Clause 4 of the OWS Common Implementation Standard [OGC 06-121r3] shall apply. In addition, the following terms and definitions apply.

[5] http://www.opencloudmanifesto.org

3.1
process
model or calculation that is made available at a service instance

3.2
input
data provided to a process

3.3
output
result returned by a process

3.4
job
task to be performed in a computational environment

3.5
middleware
middleware can be seen as an operating system of a virtual super computer that in reality consists of many independent and spatial distributed resources, but appears to the user as one computational unit

3.6
grid
a grid is a system that: coordinates resources that are not subject to centralized control using standard, open, general-purpose protocols and interfaces to deliver nontrivial qualities of service [5].

3 Conventions

3.1 Abbreviated terms

Some more frequently used abbreviated terms:

ER	Engineering Report
GI	Geoinformation
GIS	Geo Information Systems
GSI	Grid Security Infrastructure
HTTP	Hypertext Transfer Protocol
HPC	High Performance Computing
OGC	Open Geospatial Consortium

OGF	Open Grid Forum
OGSA	Open Grid Service Architecture
OGSI	Open Grid Service Infrastructure
OWS	OGC Web Service
QoS	Quality of Service
SDI	Spatial Data Infrastructure
SLA	Service Level Agreement
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
SWE	Sensor Web Enablement
WPS	Web Processing Service
WSDL	Web Services Description Language
WSRF	Web Services Resource Framework
XML	Extensible Markup Language
XSLT	Extensible Stylesheet Language Transformations

3.2 UML notation

Most diagrams that appear in this standard are presented using the Unified Modeling Language (UML) static structure diagram, as described in Subclause 5.2 of [OGC 06-121r3].

4 OWS-6 Grid Computing Overview

Highly specialized geospatial applications based on large volumes of distributed data such as live sensor data streams at different scales combined with high resolution geodata, which have to be analyzed in real-time for risk management issues, require often the functionality of multiple processes. In such highly specialized large-scale geospatial applications, not every processing step can potentially be handled by a single processing entity (for example with the resources of a single computer). To improve the computational performance of processing large amounts of dynamic geodata, Grid Computing provides appropriate tools.

The goal of the OWS-6 Interoperability testbed was, that WPS should be able to benefit from and integrate with distributed computing resources and technologies to enable applications to scale out.

In the context of linking geospatial applications and SDI with Grid Computing only little research has been conducted. Baranski [1] and Lanig et al. [13] extended the work of Di et al. [3] and accomplished first experiments using Grid Computing technology for improving processing performance by distributing and parallel execution of processes. Recently, the GDI-GRID[6] project has started focusing on the integration and processing of geospatial data and services in a grid infrastructure. In particular, it addresses the implementation of security mechanisms and definition of service interfaces. The purpose of SEE-GEO[7] project is to provide access to the EDINA[8] national datacenter hosting geospatial services running on the UK National Grid Service (NGS)[9]. The SLA4D-GRID[10] project (supposable starting in June 2009) focuses on the development of a common software stack for managing and enforcing Service Level Agreements (SLA) in the German grid infrastructure D-GRID via advanced resource reservation mechanisms. An SLA-enabled WPS is one important use-case in this research project. Furthermore, WSRF was proposed by Woolf [18] as a mechanism to provide a refactoring of OGC services around a resource-oriented view of data within a SOAP-based web services framework.

This chapter provides relevant background information for understanding the Grid Computing related work in the OWS-6 GPW thread.

4.1 Web Processing Service

With the advent of the Service Oriented Architecture (SOA) paradigm [4] and general advancements in Web Services technology [17], the geoinformation (GI) world underwent a substantial change from stand-alone applications to distributed service architectures manifested in Spatial Data Infrastructures (SDI) [15]. Existing SDIs are mainly focused on data retrieval, data processing and data visualization [11]. An open standard based SDI mostly supports the retrieval and visualization of data through Web Services. But the data processing was normally done by human actors with more or less proprietary and monolithic Geo Information Systems (GIS).

The OGC Web Processing Service (WPS) became an official OGC standard (OGC 05-007r7) in mid 2007 and is a first but major attempt to address geospatial processes in a standardized way. The WPS standard defines a standardized interface to publish and perform geospatial processes over the web. Such a process can range from a simple

[6] http://www.gdi-grid.de/

[7] http://edina.ac.uk/projects/seesaw/index.html

[8] http://edina.ac.uk/

[9] http://www.grid-support.ac.uk/

[10] http://www.sla4d-grid.de/

geometric calculation (for example a simple intersect operation) to a complex simulation process (for example a global climate change model).

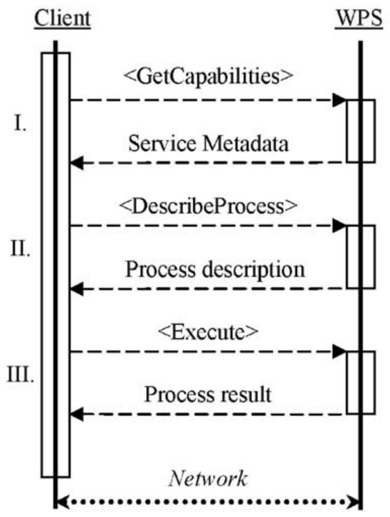


Figure 1 WPS communication pattern based on HTTP-GET and –POST

The WPS interface is based on three operations, which are all handled in a stateless manner: GetCapabilities, DescribeProcess and Execute. The GetCapabilities operation is common to any type of OGC Web Service (OWS) and returns service metadata. In case of WPS it also returns a brief description of the processes offered by the specific WPS instance. To get more information about the hosted processes, the WPS can return more detailed process metadata through the DescribeProcess operation. This operation provides the detailed description of all parameters, which are required to run the process. Based on this information the client can perform the Execute operation upon the designated process. This course of action is also depicted in Figure 1. As any OWS, the WPS communicates through HTTP-GET and POST using a messaging based on an OGC-specific XML-encoding.

4.2 Application Profiles

The WPS standard provides the ability to define and use WPS Application Profiles for defining processes in a reusable way, so that fully automated interoperability can be achieved. While it is possible to write a generic client for WPS, the use of a profile enables optimization of interoperable client user interface behavior, as well as the publish-find-bind paradigm. To achieve high interoperability, each process shall be specified in an Application Profile of the WPS standard.

A WPS Application Profile describes how WPS shall be configured to serve a process that is recognized by OGC. An Application Profile consists of

1. An OGC URN that uniquely identifies the process (mandatory)

2. A reference response to a DescribeProcess request for that process (mandatory).
3. A human-readable document that describes the process and its implementation (optional, but recommended).
4. A WSDL description for that process (optional).

WPS Application Profiles are intended for consumption by web service registries that maintain searchable metadata for multiple service instances.

Nash summarized in [16] his experiences with creating WPS Application Profiles. He presents some initial ideas as to how profiles could be structured and he made some suggestions of generic processes for which global profiles could and should be created. Amongst other things he noted that the structure of the OGC namespace which is to be used for WPS profiles is currently unclear.

This document does not present a specific common WPS Grid Processing Profile. It reviews the Grid Computing related WPS processes that were implemented in the OWS-6 GPW thread. By providing an example response to a DescribeProcess request, and a WSDL document for each of these processes, this document presents several profiles for executing and monitoring applications in a Grid Computing infrastructure, amongst other things.

4.3 Open Grid Forum

The Open Grid Forum (OGF)[11] is an open community leading the global standardization effort for grid computing. It was formed in 2006 in a merger of the Global Grid Forum (GGF) and the Enterprise Grid Alliance (EGA).

OGF has a family of specifications that can be applied to accessing distributed computing and data resources. The most important OGF standards for submitting, monitoring and managing computational jobs are listed in Table 1. Essential for the presented work is the OGF Job Submission Description Language (JSDL) specification (OGF GFD-R.056) (OGF GFD-R-P.111). JSDL is an extensible XML specification focusing on the description of computational task submissions to traditional high-performance computer systems like batch schedulers[12].

Table 1 Standards used in the OGF HPC Basic Profile family of specifications

Standard	Version	Specification Body	Description
HPC Basic Profile	1.0	OGF	Profiles the use of existing OGF specifications (JSDL and OGSA-BES) to implement the use case of HPC batch job management. It consists of clarifications, refinements, interpretations and amplifications of those specifications in

[11] http://ogf.org/

[12] http://en.wikipedia.org/wiki/Batch_scheduler

			order to promote interoperability.
Job Submission Description Language (JSDL)	1.0	OGF	Provides a XML-based language for describing the submission requirements of computational jobs to computing resources, as in, but not limited to, those found in grids.
OGSA Basic Execution Service (OGSA-BES)	1.0	OGF	Presents a specification for a service to which clients can send requests to initiate, monitor, and manage computational jobs.
JSDL HPC Profile Application Extension	1.0	OGF	Describes a normative application extension to JSDL, but removes some of the features of the original JSDL POSIX application that present barriers to interoperability.
HPC File Staging Profile	1.0	OGF	Profiles the file staging capabilities of JSDL for use by HPC Basic Profile-compliance services. It includes clarifications, refinements, interpretations and amplifications of JSDL which promote interoperability.
HPC Job Scheduling: Base Case and Common Cases	-	OGF	Describes a set of use cases for batch job scheduling of scientific/technical applications, also broadly referred to as the core high performance computing (HPC) use case.
Basic Security Profile	1.0	WS-I	Based on a set of non-proprietary Web services specifications. It makes a number of security recommendations intended to improve security.

The complex dependencies between standards used in the OGF HPC Basic Profile family of specifications are shown in Figure 2.

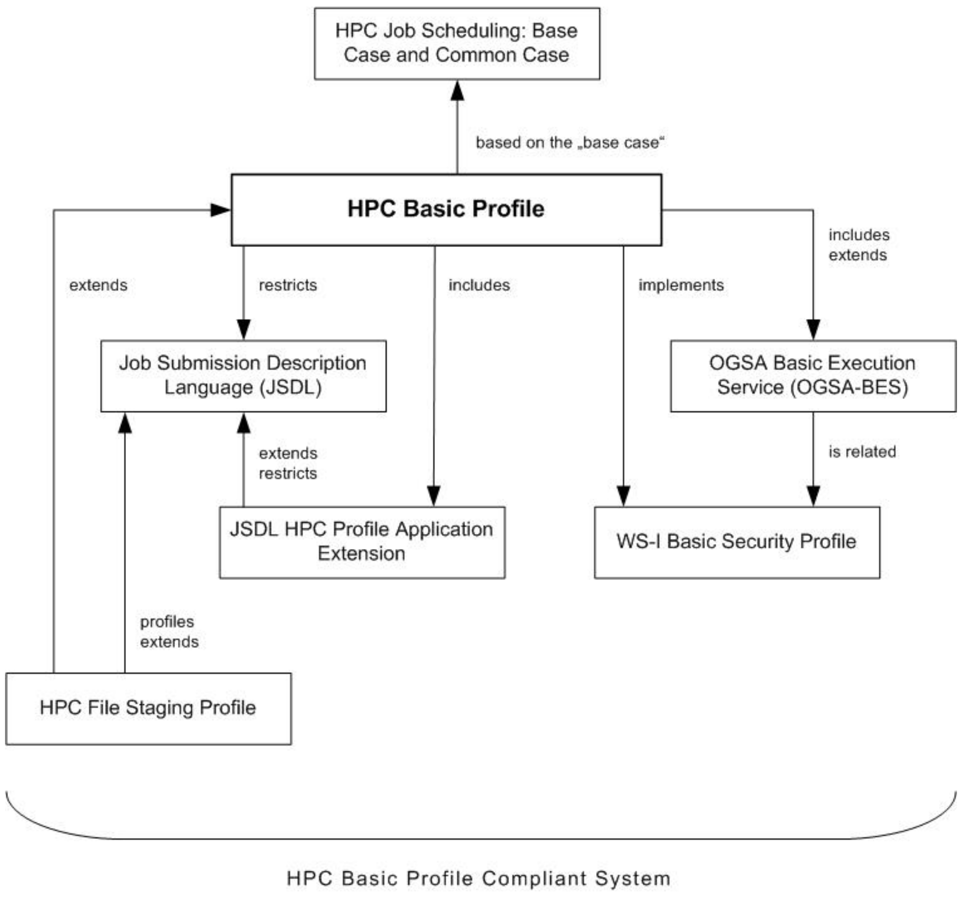

HPC Job Scheduling: Base Case and Common Case
based on the „base case“
HPC Basic Profile
extends
restricts
includes
implements
includes
extends
Job Submission Description Language (JSDL)
OGSA Basic Execution Service (OGSA-BES)
extends
restricts
is related
JSDL HPC Profile Application Extension
WS-I Basic Security Profile
profiles
extends
HPC File Staging Profile
HPC Basic Profile Compliant System

Figure 2 Dependencies between standards in OGF HPC Basic Profile

In addition to the HPC Basic Profile family of specifications, there exist several other standards and specifications related somehow to Grid Computing. The most important standards from other standardisation bodies that might be relevant for the work in OWS-6 are listed in Table 2.

Table 2 Standards used in Grid Computing infrastructures

Standard	Version	Specification Body	Description
Web Service Resource Framework (WSRF)	1.2	OASIS	Defines a set of conventions and extensions on the use of WSDL and XML Schema to enable stateful Web services.

In late 2007 the OGC and the OGF signed a formal Memorandum of Understanding (MoU). This MoU defines particular areas for collaboration, such as

- Integration of WPS specification with a range of back-end processing environments to enable large-scale processing. The WPS could also be used as a front-end to interface to multiple grid infrastructures (for example the United Kingdom's National Grid Service, NGS).

- Integration of WPS with workflow management tools.

- Integration of OGC federated catalogs and data repositories with Grid data movement tools. The GridFTP protocol is one possibility that supports secure, third-party transfers that are useful when moving large amount of data from a repository to a remote service.

OGC and OGF are collaborating to accelerate development and adoption of geospatial GRID computing worldwide, because Grids have the potential to boost the power and usefulness of geospatial technologies in a wide range of important application areas. OGC and OGF recently organized some collaborative meetings; for example the OGC-OGF Collaboration Workshop at OGF-22[13] and OGC-OGF Collaboration Ad-hoc Meeting at December '08 OGC Technical Committee Meeting in Valencia.

4.4 Encapsulation vs. Integration

The goal of the OWS-6 Interoperability testbed was, that WPS should be able to benefit from and integrate with distributed computing resources and technologies to enable applications to scale out. In preparation for the OWS-6 Interoperability testbed, two potential ways to make use of OGF and related specifications, concepts and their implementations were identified. First, WPS can interact with distributed computing resource in the backend (encapsulating other resources). Secondly, the WPS can become a fully embedded resource accessed by middleware (integration alongside other services).

In Figure 3 these two potential ways and their impact on typical WPS implementations (encapsulating and integrated) are shown.

In the first case, a WPS could use other Grid services and well defined mechanisms are used to access additional external computation or data storage resource. This approach will be referred to as “encapsulation” as the clients to the WPS are not exposed to any changes.

In the second case, the WPS would be integrated within other Grid Computing environments. This approach offers opportunities to realize benefit from different existing middleware solutions such as intelligent workflow management, reliability, security and other Quality of Service attributes. This approach is referred to as “integration” as the WPS becomes a service that clients may utilize alongside others in distributed service oriented environment.

[13] http://www.ogf.org/OGF22/materials/1075/OGC-OGF-Workshop-Report_OGF-22.doc

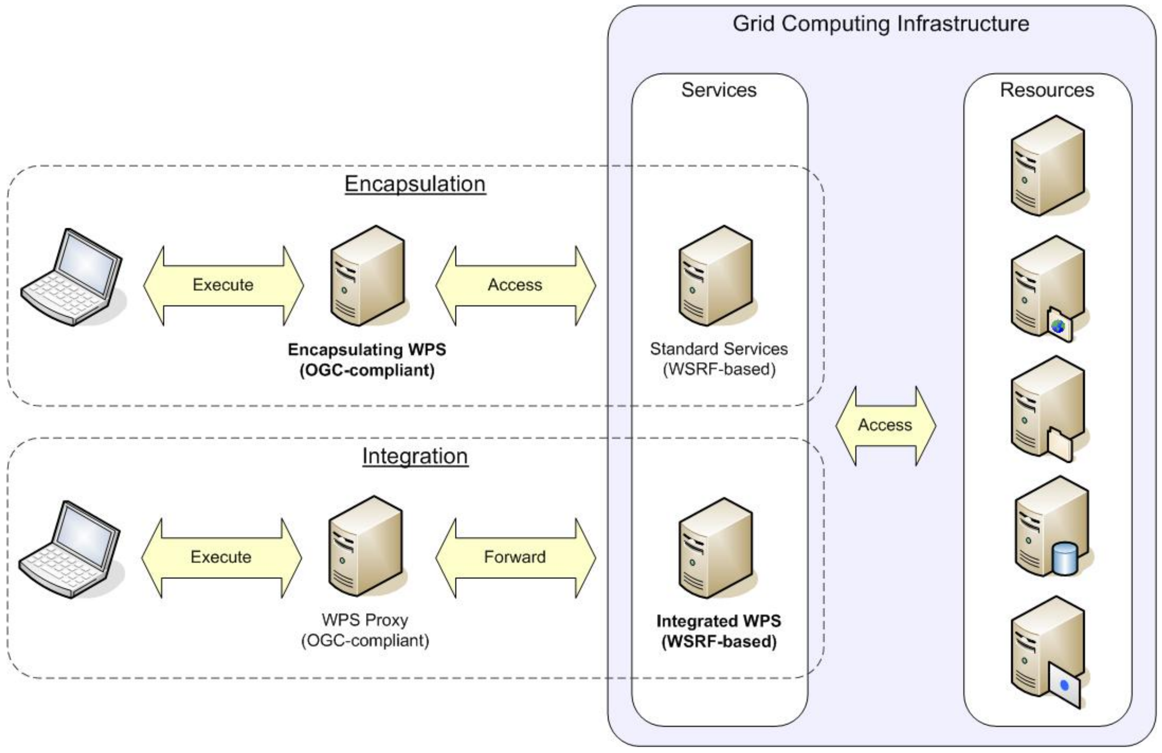


Figure 3 Two potential ways for the gridification of WPS.

The Open Grid Services Architecture (OGSA) [7] describes an architecture for service-oriented Grid Computing environments and is based on several web service technologies (for example WSDL and SOAP). An integral part of OGSA and of widespread grid middleware solutions (for example the Globus Toolkit[14] and UNICORE[15]) is the Web Services Resource Framework (WSRF) specification which provides the users with the ability to access and manipulate state of web services. To realize entire benefit from existing grid middleware solutions (such as intelligent workflow management, security and other service qualities), the WPS must be completely integrated within grid middleware beside other Grid services and therefore a WSRF binding for WPS must be defined and implemented.

Current OGC standards define typically a stateless message exchange via plain HTTP-GET and HTTP-POST protocol and therefore, in the integration use-case an OGC-compliant proxy component for accessing integrated and now WSRF-based WPS must be developed. Thus, users will be absolutely shielded from the underlying WSRF-based implementation and so existing OGC-compliant clients are not exposed to any changes.

---

[14] http://www.globus.org/

[15] http://www.unicore.eu/

5 OWS-6 Grid Processes

In the OWS-6 GPW thread four geospatial services for performing distributed processes in a Grid Computing environment were implemented as WPS processes (encapsulation of distributed resources; section 4.4).

Two services where developed for the OWS-6 GPW Airport Demonstration Scenario:

- WPS Trajectory Service process (section 5.2)
- WPS Plume Service process (section 5.3)

Further information about the OWS-6 GPW Airport Demonstration Scenario can be found in the OWS-6 Request For Quotation and Call For Participation[16] or at the OWS-6 project pages[17].

Two other services where developed for the OWS-6 GPW Debris Flow Demonstration Scenario:

- WPS Rainfall Interpolation process (section 5.4)
- WPS Geophone Analysis process (section 5.5)

An introduction into the architecture of existing Debris Flow monitoring system in Taiwan and further information about the OWS-6 GPW Debris Flow Demonstration Scenario is provided in the following paragraph (section 5.1).

5.1 Debris Flow Demonstration Scenario

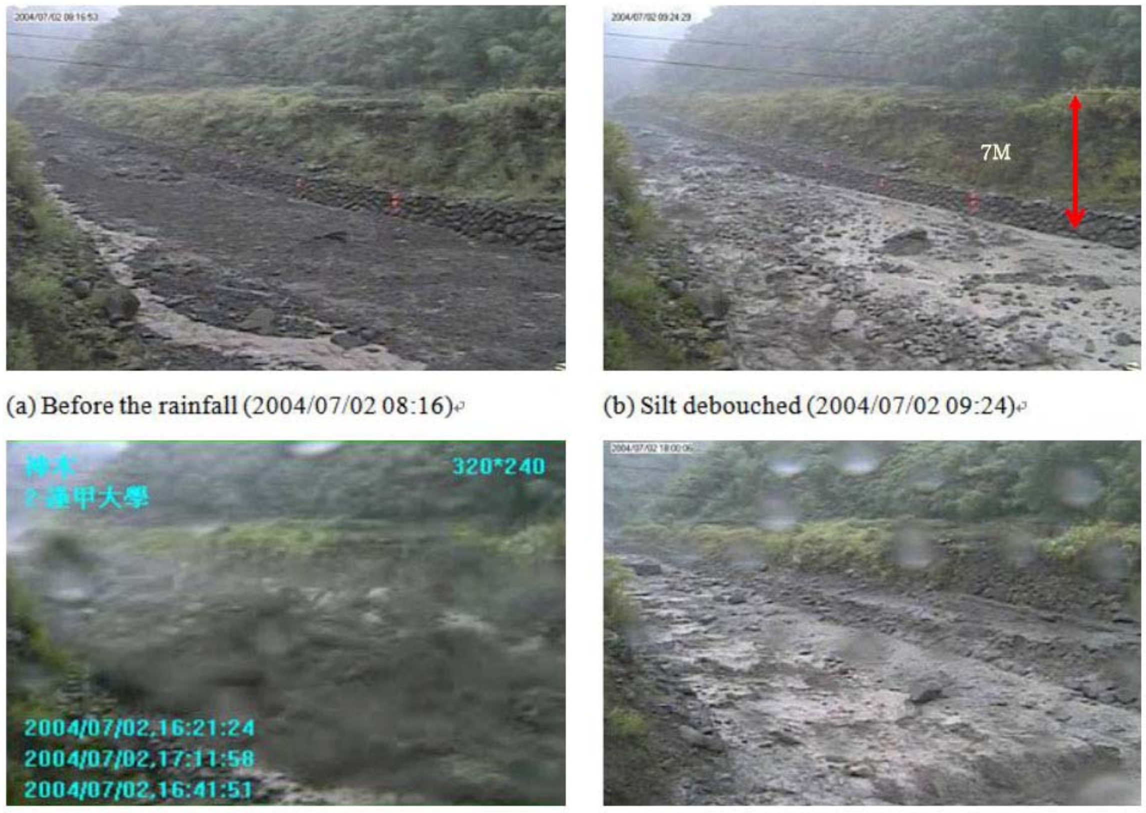
Taiwan is located on the collision boundary of the Philippine Sea plate and the Eurasian plate. The mountain terrain is precipitous and the region, on the whole, is characterized by fragile rocks and frequent seismic activity. In addition, the concentrated torrential rainfall brought by typhoons and Mei-Yu cause extensive disasters. There are two reasons for the occurrence of a Debris Flow after a strong earthquake. One is that the land collapses after earthquake and the soil gets mixed with groundwater or surface runoff. The second reason is that many crevices are formed in the earth surface after earthquake and hence, when the groundwater level increases or surface runoff concentrates, the land collapses and Debris Flow occurs.

A Debris Flow is a fast moving mass of unconsolidated, saturated debris that looks like flowing concrete. They differentiate from a mudflow by terms of the viscosity of the flow. Flows can carry rocks ranging in size from clay particles to boulders, and also often contains a large amount of woody debris. Debris Flows can be triggered by large amounts of rainfall, or glacial melt, or a combination of the two. The speed of a Debris Flow can

[16] http://www.opengeospatial.org/standards/requests/50

[17] http://portal.opengeospatial.org/?m=projects&a=view&project_id=280

vary from 1 mph to 35 mph in extreme conditions. Variations in the conditions that affect the Debris Flow can include slope and available loose sediment. Debris flows are extremely destructive to life and property.



(a) Before the rainfall (2004/07/02 08:16)
(b) Silt debouched (2004/07/02 09:24)
(c) Debris flows arrived (2004/07/02 16:41)
(d) Debris flows passed (2004/07/02 18:00)

Figure 4 Typical Debris Flow event in 2004

Since 2002, the Soil and Water Conservation Bureau (SWCB)[18], which is responsible for the conservation and administrative management of hillside in Taiwan, has been cooperated with GIS Research Center[19] at Feng Chia University (FCU)[20]. Together, they have successively carried out the establishment and maintenance of 13 fixed Debris Flow monitoring stations over Taiwan and 2 mobile Debris Flow monitoring stations. There are three rain gauges near each station and three “geophones” in monitoring station. The rain gauges are used to record on-site rainfall. The warning model for the Debris Flow alert uses accumulated precipitation as warning indexes to determine whether rainfall is reached a specific threshold or not. When debris flows occur, the geophone can detect the ground vibration generated by the collision between boulders and channel bed.

---

18 http://en.swcb.gov.tw/

19 http://www.gis.fcu.edu.tw/

20 http://en.fcu.edu.tw/

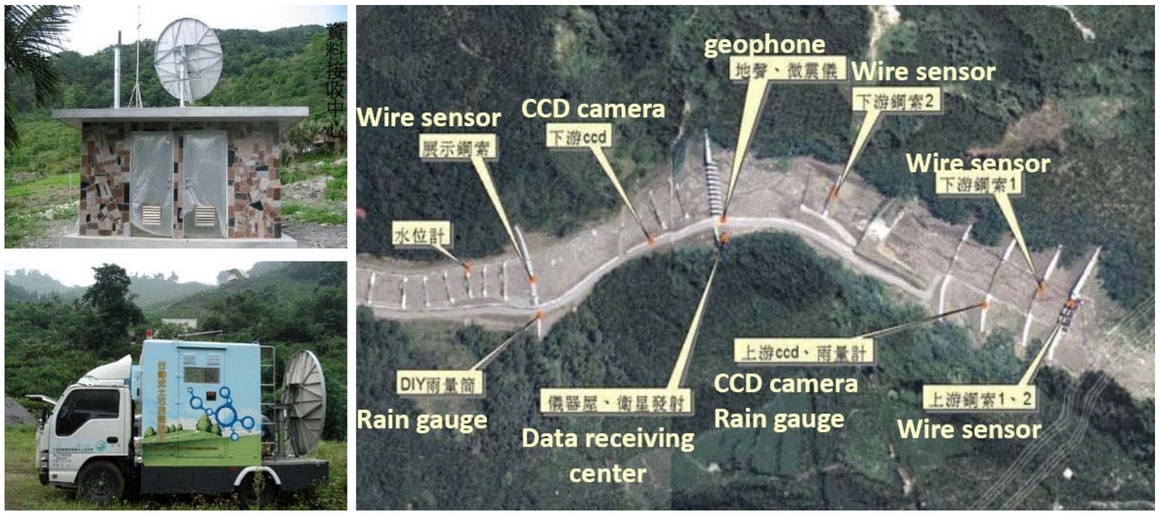


Figure 5 A typical layout of a fixed monitoring station

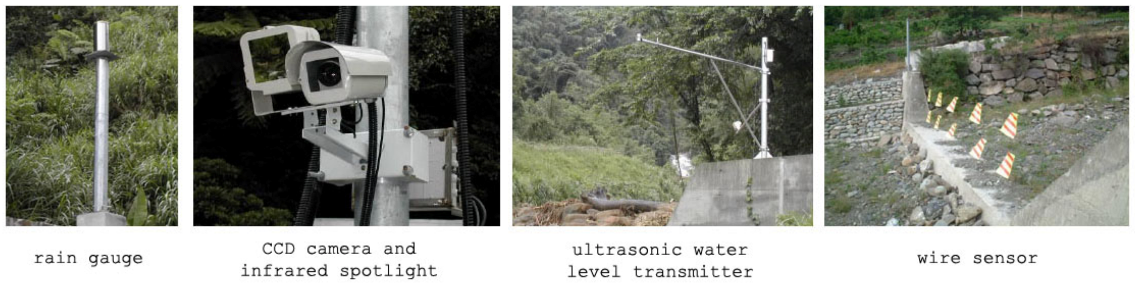

rain gauge
CCD camera and infrared spotlight
ultrasonic water level transmitter
wire sensor

Figure 6 Different sensors deployed at a fixed monitoring stations

Each monitoring station has an established data acquisition center (Figure 7) to collect and store the observation information from each sensor.

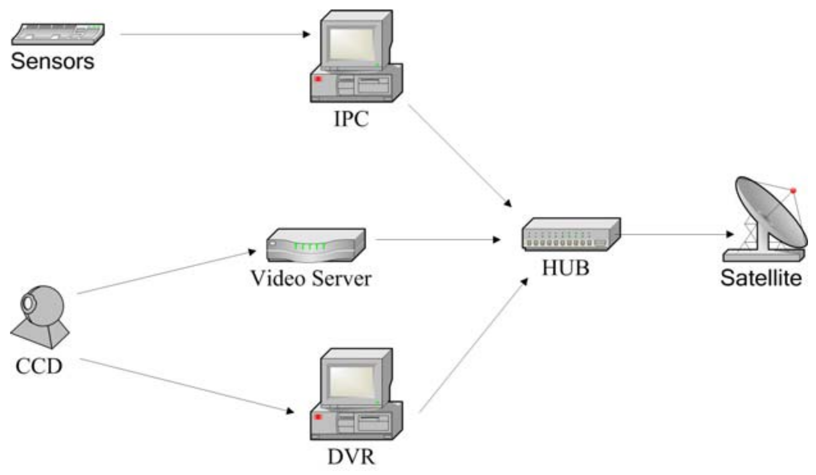


Figure 7 Primary communication network of data acquisition center

The Primary communication network of Debris Flows monitoring station is mainly through Zhongxin No. 1 satellite (Figure 8). Unlike the service of cell mobile that cannot extend to the mountains and wilderness, satellite has the advantage of wider bandwidth and full coverage. Thus an enhanced transmission quality of the observation information is attained. In order to maintain the transmission of information to the Debris Flow Emergency Response Center even when the satellite communication is disconnected, according to the communication status of each monitoring station, ADSL, GSM, GPRS or PSTN network is used as the backup transmission system.

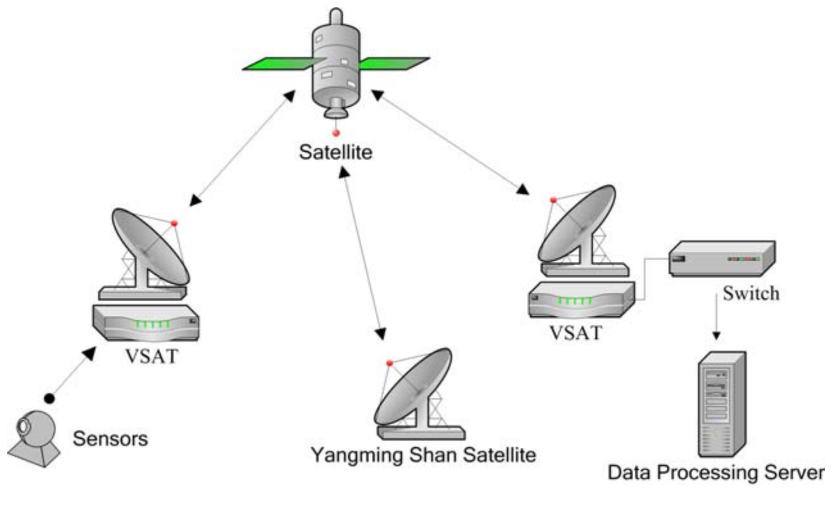


Figure 8 Structure of the satellite transmission

The Debris Flow emergency response system[21] is responsible for the collection and display of real-time Debris Flow information and the integration of observation and forecast information from Central Weather Bureau (CWB)[22]. The integrated hardware structure of the emergency response system is shown in Figure 9.

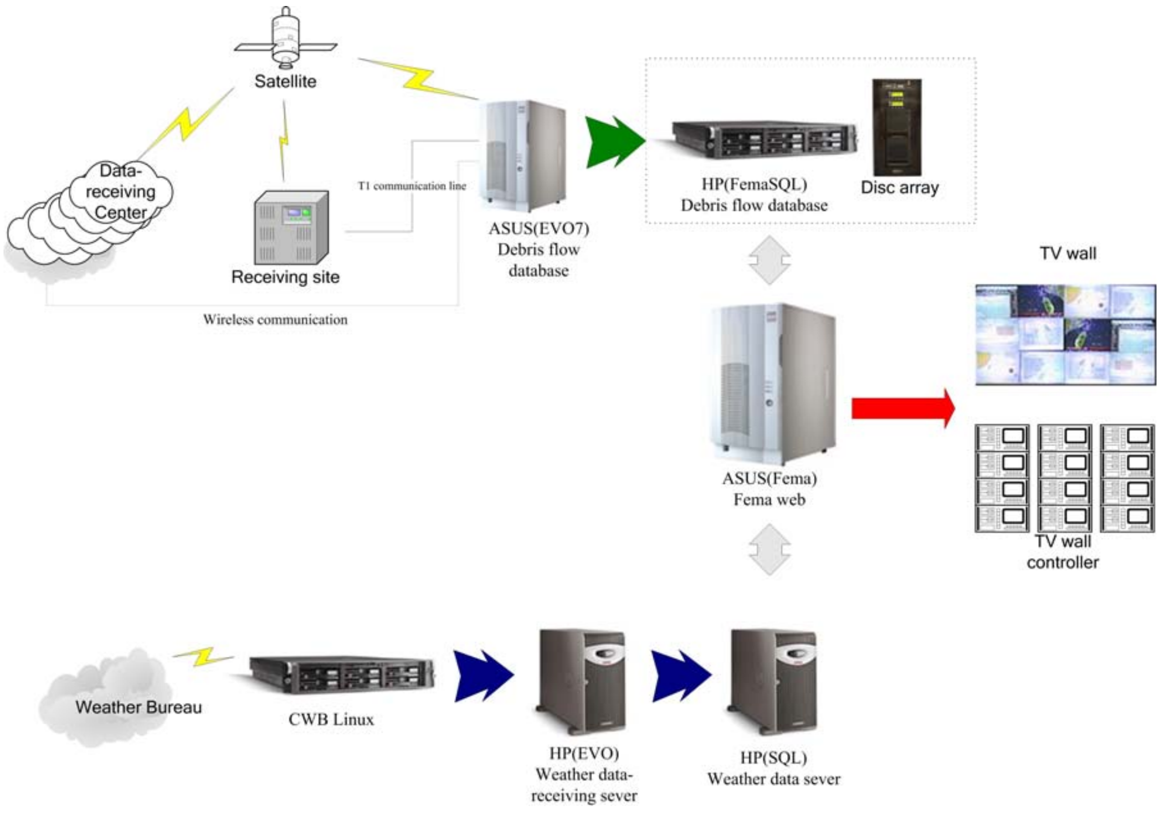


Figure 9 Integrated hardware structure of the Debris Flow emergency response system

The debris flow management life cycle is a typical disaster management life cycle, it conforms mitigation, preparedness, response and recovery. These works will not be achieved and done by only one technology or single organization, hence, integration of heterogeneous sensors and real-time data streams will be the critical successful factor of this issue. To handle those heterogeneous sensors, databases and situation varied from minute to minute, it needs a flexible, scalable and loosely coupled architecture. The Service Oriented Architecture (SOA) paradigm seems to be the optimal one to deal with integrating heterogeneous resources from different operational domains. SWCB, who takes charge of Debris Flow disaster management in Taiwan, already has been adopting open standards based SOA (for example OASIS WS-* standards) into their framework. Furthermore, OGC WMS and WFS standards have been applied to internal GIS map exchange.

[21] http://246.swcb.gov.tw

[22] http://www.cwb.gov.tw

The OWS-6 GPW Debris Flow Demonstration Scenario was developed in cooperation with GIS Research Center at (FCU) and Institut für Geoinformatik (IfGI) at University of Münster. The major incitement for GIS.FCU to participate in OWS-6 was, to investigate the application of OGC WPS standard in combination with Grid Computing for the Debris Flow disaster management. Future research at GIS.FCU will focus on the development of SWE technologies for heterogeneous sensor integrating in order to meet the goal of sensors plug-and-play.

The OWS-6 GPW Debris Flow Demonstration Scenario consists of the following steps.

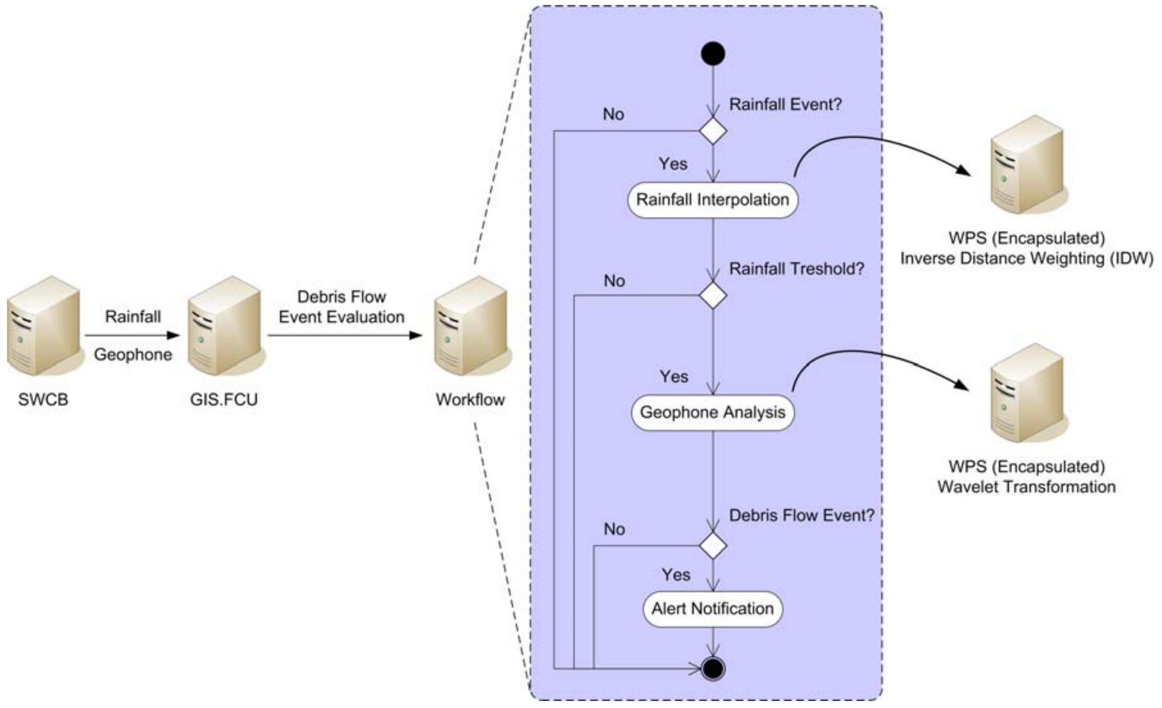


Figure 10 OWS-6 GPW Debris Flow Demonstration Scenario workflow overview

When Typhoon approaches Taiwan, CWB announces the typhoon alert to SWCB. GIS.FCU shoulders the evaluation of the Debris Flow event according to those sensors which captured or published by SWCB and CWB. GIS.FCU uses the real-time rainfall and geophone data from SWCB, to evaluate a Debris Flow event.

Then the evaluation of the Debris Flow event starts:

- To evaluate the current accumulated precipitation whether rainfall is reached the threshold (event mode is 100 mm)

- If it is a rainfall event, a WPS process for rainfall interpolation will be invoked. This WPS process uses spatial analysis like Inverse Distance Weighting (IDW) algorithm and it will utilize a Grid Computing infrastructure in the backend in order to accelerate the calculating time.

- If the result of rainfall interpolation comes up to the threshold of a Debris Flow alert (250 mm), then a WPS process for geophone analysis (done by Wavelet Transform algorithm) will be invoked. This process also utilizes a Grid Computing infrastructure in the backend in order to accelerate the calculating time. If the geophone analysis reached the threshold as wavelet accumulated energy, the Debris Flow event occurred.

- If it is a debris flow event, GIS.FCU would notify SWCB.

An abstract architecture and workflow overview of OWS-6 GPW Debris Flow Demonstration Scenario is provided in Figure 10.

5.2 Trajectory Service

This chapter describes the WPS Trajectory Service process that was developed by the Science and Technology Facilities Council (STFC)[23] for use in the OWS-6 GPW Airport Demonstration Scenario workflow.

5.2.1 Description

The British Atmospheric Data Centre (BADC) Trajectory Service[24] provides a simple interface to an atmospheric trajectory model. The model is seeded with one or more particles at specific locations and times, and these particles are then advected forward in time according to four-dimensional wind fields resulting from a pre-computed atmospheric circulation model.

The BADC service is configured to use European Centre for Medium-Range Weather Forecasts (ECMWF)[25] wind data[26]. This wind data is available internally to STFC, since the BADC is co-located on campus at the Rutherford Appleton Lab. It was beyond the scope of this project to modify the existing codebase of BADC Trajectory Service to use an arbitrarily-specified input atmospheric model dataset. The model also doesn't focus explicitly on diffusion, as does the Met Office's NAME[27] model, for instance. To simulate diffusion, the model would need to be seeded with a large number of individual particles (although strictly speaking, this still wouldn't correctly model the diffusive terms in the scalar transport equations).

---

[23] http://www.scitech.ac.uk/

[24] http://badc.nerc.ac.uk/community/trajectory/

[25] http://www.ecmwf.int/

[26] http://badc.nerc.ac.uk/community/trajectory/wind.html

[27] http://en.wikipedia.org/wiki/NAME

The standard BADC Trajectory service has a browser-based interface that operates in two stages. The first involves specifying the initial conditions for the trajectory run based on user inputs. The second is the calculation and dissemination of the trajectory results. While the service is capable of calculating the traces of particles in the atmosphere over a specified time, it is not able to calculate the dispersion of the particles.

For this project, the existing BADC Trajectory Service was ported to the UK National Grid Service (NGS)[28] and encapsulated through the WPS interface (section 4.4). To use the gridified BADC Trajectory Service, permission must be obtained to use the underlying ECMWF data. This research project has (username / password) protected the demonstration. Further information regarding the secure access to NGS via MyProxy mechanisms are given in section 7.

5.2.2 Interface

Inputs required by the BADC Trajectory Service may be divided into the following different categories in terms of their relevance to the OWS-6 GPW Airport Demonstration Scenario.

- Dataset to be used by the service. The datasets used by the BADC trajectory service are mainly comprised of time-series of wind fields. The code is currently designed to use ECMWF model data for this purpose; while it could in principle be modified to work with any suitable wind data this would involve recoding beyond the scope of this project. The existing service has hard-coded local filesystem access to required files, but this could in principle be modified to support data staging from a separate host to simulate on-demand access to discovered wind datasets.

- Information about the time of the release of particles. The required time-related parameters are:

  - Start Date in `[yymmdd]` format
  - Start time in hours; available options are `00:00`, `06:00`, `12:00` or `18:00`
  - Length of run in days
  - Trajectory direction; available options are forward or backward in time
  - Output frequency in hours (i.e. trajectory update interval)

- Vertical advection; available options are 3D (relevant to the OWS-6 GPW Airport Demonstration Scenario as calculations are done in terms of pressure) or Isentropic (calculations are done in terms of temperature)

- Information about initial positions of the particles. The required parameters are:

[28] http://www.grid-support.ac.uk/

- o The location at which the particles were released
- o Release pressure in hectopascal (hPa, ground level corresponds to roughly 950 hPa)
- o Distance between particles if a cluster of particles is released (this typically must be ~1 km or greater).

- Invocation of the BADC Trajectory Service through the developed Grid encapsulating WPS is (username / password) protected. Therefore, it is necessary to include appropriate security credentials in the WPS SOAP request for authentication[29].

The outputs of the standard BADC Trajectory Service run are mainly presented in the form of a downloadable netCDF file (binary based format), which can be visualized using any netCDF visualization tool, such as ncBrowse. An example of such a netCDF file is available at the OWS-6 Twiki pages[30].

Alternatively, the standard BADC Trajectory Service can be configured to return outputs as graphical plots (Figure 11) or NASA Ames files (ASCII based format). Dissemination of the outputs in formats other than the netCDF format involves the use of additional conversion tools. For this project, the output is transformed into GML.

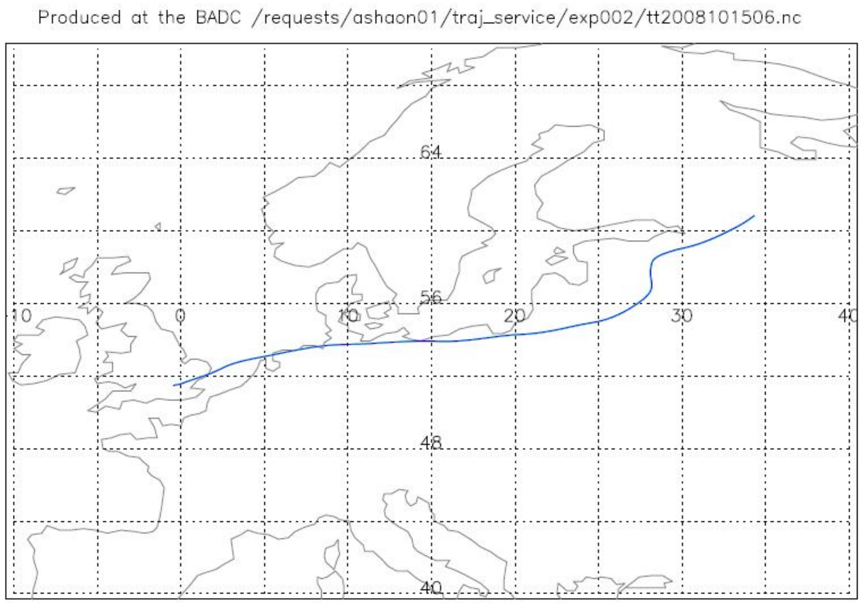

Produced at the BADC /requests/ashaon01/traj_service/exp002/tt2008101506.nc
64
56
48
40
-10
0
10
20
30
40

Figure 11 Example output of BADC Trajectory Service

[29] Contact Arif Shaon (arif.shaon@stfc.ac.uk) or Andrew Woolf (andrew.woolf@stfc.ac.uk) for access information.

[30] https://portal.opengeospatial.org/twiki/pub/OWS6/TrajectoryService/tt2008101506.nc

For this project, the existing BADC Trajectory Service was ported to the UK NGS and encapsulated through the WPS interface. Some of the inputs required by the BADC Trajectory Service are static for this demonstration and can be handled via configuration file or hard coded. But some of the inputs are used in the JSDL document that is submitted to UK NGS and must be set dynamically through the WPS interface in the Execute request.

The parameter names, meanings, data types shall be as specified in Table 3.

Table 3 WPS Trajectory Service process parameter overview

Parameters	I/O	Value Type	Note
Username	Input	String	A username used for logging purposes.
JSDL	Input	String	The proposed JSDL Profile input parameter approach.
MyProxy	Input	String	Some details for the MyProxy connection (server hostname and port, username, password)
Datetime	Input	Timestamp	Start date and time of the particle in timestamp format [yyyymmddhh]
Length	Input	Integer	Length of trajectory run in days
Time_step	Input	Integer	Output frequency in seconds (i.e. trajectory update interval)
Lat	Input	Float	The latitude of the initial position of the particle
Lon	Input	Float	The longitude of the initial position of the particle
Lev [a]	Input	Float	The atmospheric pressure in hectopascals (representing height) of the initial position of the particle
output	Output		An URL to a GML file containing the predicted particle path.

a Atmospheric pressure decreases with altitude, a value of 950 is close to the surface, while the default (100) is in the upper atmosphere. Values should be in the range 10-950. Using pressures greater than 950 hPa often results in parcels going into the ground.

An example SOAP-based WPS Execute request document is listed in Appendix C.1.

5.2.3 JSDL Profile

A WPS JSDL Profile has been developed in order to facilitate invoking and controlling a process in a Grid Computing infrastructure (namely the UK NGS) through a standard WPS interface. The WPS JSDL Profile approach involves encoding JSDL-related information (OGF GFD-R.056) as part of the DataInput parameter of the WPS Execute GET request URL defined in the WPS 1.0.0 standard. The JSDL-related information can be used by a WPS for submitting a process for execution to a Grid backend through a JSDL-enabled Grid Client such as Grid Job Submission and Monitoring Web Service

(GridSAM)[31]. Such a WPS should also be able to monitor the status of the process in the Grid using the same Grid Client. The rationale of this approach is to incorporate JSDL handling ability into a WPS while ensuring its conformance to the WPS 1.0.0 standard. This allows the capability of Grid Computing to be combined with the simplicity of the WPS interface.

Any JSDL information to be included in the WPS Execute GET request URL, may be encoded as the value of a special DataInput parameter, JSDL. The use of this special parameter JSDL enables a WPS to distinguish between JSDL-related parameters and other process-related parameters.

For example,

```
http://foo.bar.1/wps?version=1.0.0&request=Execute&service=WPS&Identifier=Weath
erGenerator&DataInput=otherinput@datatype=string;JSDL=[jsdlinformation]@Format=
text/kvp&storeExecuteResponse=true&status=true
```

This WPS JSDL Profile approach permits two methods of encoding JSDL information in a WPS Execution GET request URL.

5.2.3.1 Key-Value Pair Method

This method enables users to specify JSDL-related parameters and their corresponding values in the WPS Execution GET request as key-value pairs. This information is then used by the provided WPS implementation to dynamically construct a JSDL document for the process in question and submit it to the Grid Client for executing the process on the Grid back end. This method for encoding JSDL parameters as key-value pairs follows the rules outlined below:

- All JSDL key-value pairs are enclosed within [].

- The JSDL key-value pairs are separated by ‘;’ in line with the WPS 1.0.0 specification. For example:

```
http://foo.bar.1/wps?version=1.0.0&request=Execute&service=WPS&Identifier=We
atherGenerator&DataInput=other_inputs=xxx;JSDL=[TotalDiskSpace=5;TotalCPUCou
nt=1]@Format=text/kvp&storeExecuteResponse=true&status=true
```

- Multiple values for a JSDL parameter are separated using “,”. For example:

```
http://foo.bar.1/wps?version=1.0.0&request=Execute&service=WPS&Identifier=We
atherGenerator&DataInput=other_inputs=xxx;JSDL=[DataStaging#Target#URI=http:
//www.foo.com/myfoo.xml,http://www.foo.com/myfoo2.xml]@Format=text/kvp&store
ExecuteResponse=true&status=true
```

- To ensure uniqueness of the JSDL keywords used in JSDL key-value pairs, an XPath[32]-like syntax can be used to specify a keyword. This special syntax uses

[31] http://gridsam.sourceforge.net/2.1.4/index.html

[32] http://en.wikipedia.org/wiki/XPath

'#' symbol (instead of XPath's '/' symbol) to separate elements. For example, both target and source elements of the `DataStaging` element in JSDL specification have an URI element to record the corresponding target and source URL respectively. In a WPS Execute GET request following the JSDL Profile approach, these two elements can be uniquely specified. For example,

```
http://foo.bar.1/wps?version=1.0.0&request=Execute&service=WPS&Identifier=We
atherGenerator&DataInput=other_inputs=xxx;JSDL=[DataStaging#Source#URI=http:
//www.foo.com/foo;DataStaging#Target#URI=http://source.com/source;TotalCPUCo
unt=1]@Format=text/kvp&storeExecuteResponse=true&status=true
```

A simplified version of the JSDL key-value pair method – a list of JSDL parameter names and corresponding parameter names from within the `JSDL` input parameter – is listed in Table 4.

Table 4 A simplified version of JSDL key-value pair method

WPS `JSDL` input parameter name	JSDL Parameter Name
DiskSpace	TotalDiskSpace
DiskSpaceMax	TotalDiskSpace#Max
DiskSpaceMin	TotalDiskSpace#Min
CPU	TotalCPUTime
CPUMax	TotalCPUTime#Max
CPUMin	TotalCPUTime#Min
Source_URI	DataStaging#Source#URI
Target_URI	DataStaging#Target#URI
User_Dir_Name	FileSystem#name
User_Dir_Path	FileSystem#MountPoint

There are, in principle, no restrictions on the length of a HTTP GET URL beyond the HTTP specification (RFC 2616), which allows URIs of unbounded length.

The generation by the WPS server of a conformant and appropriate JSDL document for submission to a Grid Computing infrastructure in the backend is implementation-specific. Typically, an internal configuration would assign a default JSDL template document to a specific WPS process, with user-supplied JSDL parameters over-riding the defaults. It is left to specific implementations to determine how best to manage the transformation from user-supplied WPS parameters to a JSDL document. This should ensure, for instance, that any sensible set of WPS request parameters results in a valid JSDL document. Invalid JSDL parameters in an Execute GET request may be reported by raising a standard WPS *InvalidParameterValue*[33] exception. However, future work may need to specify an approach to extending the WPS-specific error reporting technique to provide more JSDL-specific error notification.

[33] http://www.opengeospatial.org/standards/wps

5.2.3.2 URL Method

It is expected that most implementations of the JSDL-based WPS-Grid profile will use the ‘Key-Value pair’ method (§5.2.3.1). However, for the sake of full generality, an alternative method has been specified allowing a simple ‘pass-through’ option for pre-created JSDL documents. Only further testing and experimentation will allow the relative merits of the two approaches to be evaluated.

This method enables users to use a pre-created JSDL document to specify their requirements for running a process in a Grid Computing infrastructure. Using this method, a user can specify the URL to his/her JSDL document as part of the `DataInput` parameter of the WPS Execute GET request.

For example:

```
http://foo.bar.1/wps?version=1.0.0&request=Execute&service=WPS&Identifier=Weath
erGenerator&DataInput=other_inputs=xxx;JSDL=http://www.foo.com/myfoo.jsdl@Forma
t=text/xml&storeExecuteResponse=true&status=true
```

In this case, the WPS is expected to download the JSDL document from the URL specified and use it for submitting the process requested to the Grid Computing infrastructure in the backend. It should be noted that the use of a pre-created JSDL document requires a specific WPS instance to perform both syntactic and semantic validation on the document. Syntactic validation is essentially checking the structure of the JSDL document against the JSDL specification. Semantic validation, on the other hand, should involve ensuring the validity of the information included in the document for the WPS process to which it corresponds (for example if an application stated in the submitted JSDL document is actually available in the Grid backend).

To notify the WPS of the correct type of JSDL information encoded in the WPS Execute request, the JSDL information must be followed by the `Format` keyword, the value of which will indicate the type of the JSDL information specified (Table 5 Different types of JSDL encoding in WPS-Grid profile). This is in line with the KVP syntax for a complex value of an Execute DataInput parameter specified in the WPS 1.0 standard, as both of these methods in effect assign a complex value to the special DataInput parameter, JSDL.

Table 5 Different types of JSDL encoding in WPS-Grid profile

Key-Value Pair Method	Format=text/kvp	http:…../wps….&DataInput=…;JSDL[key=value@Format=text/kvp]
URL Method	Format=text/xml	http:…../wps….&DataInput=…;JSDL[ftp://domain.com/file.jsdl@Format=text/xml]

Any standard Grid JSDL service is expected to enable uploading the output of a process to a user-specified location (using the DataStaging#Target#URI JSDL parameter). However, to be consistent with the definition of an asynchronous process in the WPS 1.0.0 standard (for example `storeExecuteResponse=true`), the WPS should also keep a local copy of a Grid-based process output and make it available to the user. Implementation of such a feature involves a data upload service (for example via

GridFTP[34]) being co-located with the WPS, and appending the JSDL DataStaging/Target element with a WPS-specific location. This applies to both of the aforementioned methods of JSDL encoding in the WPS Execute request.

5.2.4 Implementation

STFC has implemented a WPS Trajectory Service process as a proof-of-concept implementation of the proposed WPS JSDL profile. This service provides an interface to existing BADC Trajectory Service. The existing BADC Trajectory Service has also been deployed on the UK National Grid Service (NGS) so that it may be invoked and controlled through the WPS Trajectory Service process.

An overview about the architecture of Grid-enabled WPS implementation is given in Figure 12. The Grid-enabled WPS implementation is based on the following major components.

- **WPS SOAP / Proxy Layer**

  At the top level of the architecture, there is the WPS SOAP/Proxy layer that provides a SOAP wrapper outside the STFC firewall to proxy SOAP requests through HTTP GET requests to the Grid-enabled WPS. The SOAP interface provided by the SOAP wrapper conforms to the WPS standard 1.0.0. In addition, this layer is also used for forwarding other HTTP GET requests to the Grid-enabled WPS, such as request for downloading process output and status check request for a process.

- **Grid-enabled WPS**

  Central to the architecture is the Grid-enabled WPS implementation, a service with HTTP-GET protocol conforming to the proposed WPS JSDL Profile. It uses request parameters to generate dynamically a JSDL document in the backend for submission to a Grid Computing infrastructure.

- **GridSAM Client[35]**

  GridSAM is a service for submitting jobs to be run on a computing Grid, implements the JSDL standard for job description. There is a GridSAM client installed on the machine that is hosting the Grid-enabled WPS service. The GridSAM client receives the WPS-generated JSDL job description and submits it to the GridSAM server. The WPS also interacts with the GridSAM client for checking the status of a job on the Grid; the job status result is used by the WPS to produce and record WPS-specific process status (for example ProcessCompleted, ProcessSucceeded, etc.)[36].

[34] http://dev.globus.org/wiki/GridFTP

[35] Note that while GridSAM was chosen in this project as the grid middleware for consuming JSDL documents and job management on the Grid, future work could replace this with any middleware conforming to the 'HPC Basic Profile', subject to associated restrictions on the allowable scope of JSDL (see Figure 2).

[36] Standard GridSAM status codes are mapped onto WPS status codes.

- **GridSAM Service**

At the bottom level of the architecture, there is a GridSAM service that receives a JSDL job submission from the GridSAM client and executes the requested job on the Grid (the UK NGS in this case). This service also receives queries (for example job status check, etc.) from a GridSAM client and responds to them by liaising with the Grid with which it is associated.

- **MyProxy**

MyProxy is a Web-based security service that is used to store user authentication credentials. By supplying MyProxy details (server hostname and port, username, password) in either the JSDL for a job or as part of the command line parameters (which includes a path to the corresponding JSDL file on the client's machine) for submitting the job through the GridSAM client, the GridSAM service is able to retrieve a user (for example individual, service) credential for authenticating the request for running the job on the Grid.

The STFC Grid-enabled WPS enables a user to embed MyProxy parameters (host, port, username, password) within the WPS request using the "Key-Value" encoding mechanism mentioned earlier ($5.2.3.1), as the value of a special DataInput parameter, "**MyProxy**". These are processed by the WPS server and used at job submission for authentication by the GridSAM client. For example:

http://foo.bar.1/wps?version=1.0.0&request=Execute&service=WPS&Identifier=WeatherGenerator&DataInput=otherinput@datatype=string;JSDL=[TotalDiskSpace=5]@Format=text/kvp;**MyProxy=[username=xxxx;password=xxxx;host=xxx;port=xxxx]@Format=text/kvp**&storeExecuteRepose=true&status=true

Including sensitive information, such as a user's MyProxy credential in a URL does pose security risks. Therefore, this approach assumes that implementation of this WPS-JSDL profile will incorporate necessary security protocol (e.g. SSL) to ensure the security of MyProxy information in the WPS Process Execution Get Request. Moreover, future evolution of the WPS-Grid profile will need to align with best practice for adding security to other OGC service interfaces.

A public available Endpoint Reference (EPR) to the developed Grid-enabled WPS implementation and the allocated process (namely the Trajectory Service process) is provided in **Error! Reference source not found.**.

Table 6 Trajectory Service process implementation links

GetCapabilities	http://glue.badc.rl.ac.uk/wps-jsdl/services/wps?service=WPS&version=1.0.0&Request=GetCapabilities
DescribeProcess	http://glue.badc.rl.ac.uk/wps-jsdl/services/wps?service=WPS&version=1.0.0&Request=DescribeProcess&Identifier=RunTraj

WSDL	n/a

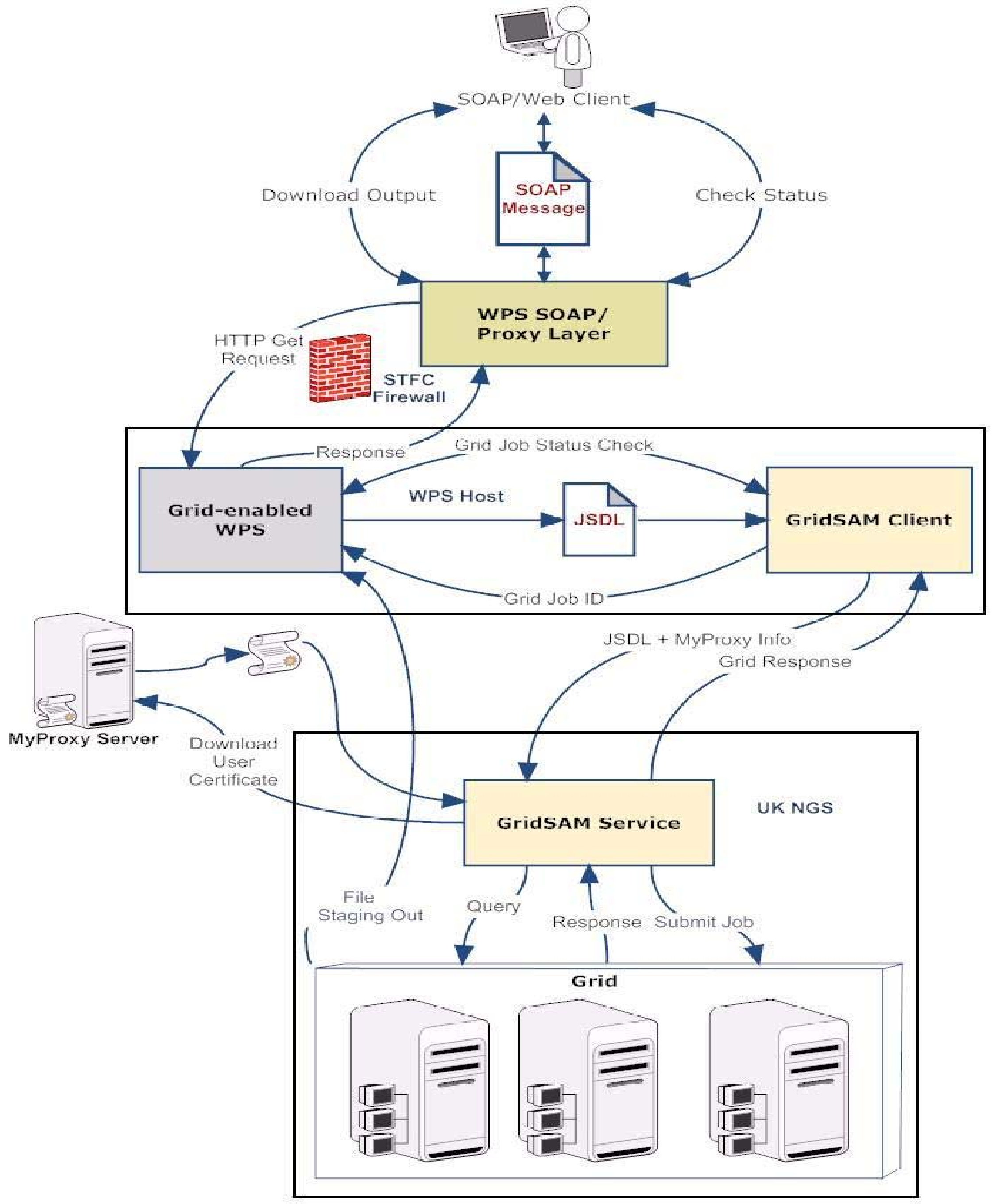


Figure 12 Architecture of STFC Grid-enabled WPS implementation

5.2.5 Results

An example WPS Execute response document of a typical WPS Trajectory Service process run is listed in Appendix C.2.

The core output of the WPS Trajectory Service process is a GML file containing the predicted particle path:

```
<?xml version="1.0" encoding="UTF-8"?>
<LineString xmlns=http://www.opengis.net/gml/3.2
xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/gml/3.2
http://schemas.opengis.net/gml/3.2.1/gml.xsd" gml:id="tt2008101506"
srsName="urn:x-ogc:def:crs:EPSG:6.6:4326">
    <posList>
51.47 -0.4689932
51.461 -0.46
51.47 -0.46
51.47898 -0.46
51.47 -0.4510068
51.51016 -0.277997
(…)
60.83254 34.30986
60.82917 34.32023
60.82578 34.33073
60.82758 34.31038
</posList>
</LineString>
```

This GML output data is used by the WPS Plume Service process (section 5.3) for rendering the probable plume dispersion outbound from the original fire location nearby the airport facilities.

Some difficulties were encountered in the implementation, primarily concerning the GridSAM middleware, e.g.:

- JSDL 'Resource' elements are unsupported[37], thus preventing functionality based on specifying required Grid resources for a job

- the JSDL data staging 'FileSystemName' element is not supported[38], preventing data staging to specified system directories (e.g. a user's 'home' directory)

- multiple instances of GridSAM may not be run within the same container[39], meaning that an installation may not be set up to enable submission to different nodes within a Grid (despite the gridsam-submit '-s' option to target a specific node)

- the latest version of GridSAM client (version 2.1.4)[40] does not work behind a proxy due to a bug in its implementation. Also, inclusion of a new Grid job state, "active-queued" in the GridSAM schema has resulted in a significant backward-incompatibility between the new version of GridSAM service and its older clients, which fail to validate job status reports received from the corresponding service(s).

[37] http://gridsam.sourceforge.net/2.1.3/gridsam-client/jsdl.html

[38] *ibid*

[39] http://wiki.ngs.ac.uk/index.php?title=GridSAM#Known_Issues

[40] http://sourceforge.net/project/showfiles.php?group_id=141177

5.3 Plume Service

This chapter describes the WPS Plume Service process that was developed by the Institute for Geoinformatics (IfGI)[41] for use in the OWS-6 GPW Airport Demonstration Scenario workflow.

5.3.1 Description

The WPS Plume Service process takes the output of WPS Trajectory Service process - a GML file containing a predicted particle path - and renders a plume dispersion based on the delivered data. Note, that the WPS Plume Service process renders the plume dispersion without an underlying real-world simulation model. It's all a "fake" and the underlying component was developed from an existing Phenomenon Generator Tool from IfGI for testing purposes in the OWS-6 GPW Airport Demonstration Scenario. The original tool depends on several input parameters that dynamically affect the randomized output.

5.3.2 Interface

The WPS Plume Service process requires the following list of input parameters.

- The URL of the to the GML output data from Trajectory Service.
- The number of pixel columns in the produced GeoTIFF files.
- The number of pixel rows in the produced GeoTIFF files.
- The amount of physical memory required on each individual computation node in the Grid.
- The total number of CPU seconds required on each individual computation node in the Grid to execute the job.
- The total number of CPUs required for the computation of this process.

The process produces the following list of output parameters.

- A time series of GeoTIFF files displaying the plume dispersion over time.

The parameter names, meanings, data types shall be as specified in Table 7.

Table 7 WPS Plume Service process parameter overview

Parameters	I/O	Value Type	Note
TRAJECTORY [a]	Input	String	The URL of the to the GML output

[41] http://ifgi.uni-muenster.de/

Parameters	I/O	Value Type	Note
			data from Trajectory Service.
WIDTH	Input	Integer	The number of pixel columns in the produced GeoTIFF files.
HEIGHT	Input	Integer	The number of pixel rows in the produced GeoTIFF files.
IndividualPhysicalMemory	Input	Integer	The amount of physical memory required on each individual computation node in the Grid. The amount is given in bytes.
IndividualCPUTime	Input	Integer	The total number of CPU seconds required on each individual computation node in the Grid to execute the job.
TotalCPUCount	Input	Integer	The total number of CPUs required for the computation of this process.
IMAGE_0	Output	ComplexValueReference	One of the rendered image files.
IMAGE_1	Output	ComplexValueReference	One of the rendered image files.
IMAGE_2	Output	ComplexValueReference	One of the rendered image files.
IMAGE_3	Output	ComplexValueReference	One of the rendered image files.
IMAGE_4	Output	ComplexValueReference	One of the rendered image files.
IMAGE_5	Output	ComplexValueReference	One of the rendered image files.
IMAGE_6	Output	ComplexValueReference	One of the rendered image files.
IMAGE_7	Output	ComplexValueReference	One of the rendered image files.
IMAGE_8	Output	ComplexValueReference	One of the rendered image files.
IMAGE_9	Output	ComplexValueReference	One of the rendered image files.

NOTE The WPS 1.0.0 standard does not allow returning multiple output parameters with the same identifier and so the WPS Plume Service process used 10 output parameters for delivering the time-series of GeoTIFF files.

a The TRAJECTORY input parameter was realized through a LiteralData type input parameter instead of a (referenced) ComplexValue type input parameter because of simplicity and for testing purposes.

An example SOAP-based WPS Execute request document is listed in Appendix C.3.

5.3.3 Mapping JSDL to WPS

When executing a process through the WPS interface, a client typically performs the following steps.

- First, a client typically executes the WPS GetCapabilities operation. This operation returns some general metadata about the specific WPS instance and a list of all offered processes.

- Second, after receiving the list of all offered processes a client typically executes the WPS DescribeProcess operation for a specific process identifier. This operation returns detailed information (name, format and occurrence) about required input parameters and output parameters for the process.

- Third, the client executes the WPS Execute operation for the specific process identifier with all required input parameters either in a synchronous or in an asynchronous manner.

The first step could be skipped if the client already knows the Endpoint Reference (EPR) of the specific WPS instance and the unique identifier of the process to be executed. The second step could be skipped if the client already knows the process description of the process to be executed. Further information and thoughts about accessing the WPS in a synchronous or in an asynchronous way could be found for example in the OWS-6 GeoProcessing Workflow Engineering Report (OGC 09-053r3).

However, when invoking and controlling remotely a process in a Grid Computing infrastructure typically a JSDL document has to be created. Not each grid middleware and not each batch system supports the submission of jobs through JSDL documents, but the JSDL specification collects and combines general information that are typically needed for the remote invocation of applications in a distributed environment and so it is a good point to start with.

This section provides a dynamic mechanism for generating WPS DescribeProcess response documents for already deployed applications in a Grid Infrastructure and for generating JSDL documents from a WPS Execute request document on demand. The basis of the presented approach is a JSDL template document and two XSLT Stylesheet documents for each process that invokes an application in Grid Computing infrastructure.

The following example JSDL document is taken from the JSDL Specification Version 1.0 and defines an invocation of the application “gnuplot”. Such a document could be used as the mentioned JSDL template document.

```
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:JobDefinition xmlns="http://www.example.org/"
xmlns:jsdl="http://schemas.ggf.org/jsdl/2005/11/jsdl" xmlns:jsdl-
posix="http://schemas.ggf.org/jsdl/2005/11/jsdl-posix"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <jsdl:JobDescription>
        <jsdl:JobIdentification>
            <jsdl:JobName>My Gnuplot invocation</jsdl:JobName>
            <jsdl:Description>Simple application invocation: User wants to run
the application 'gnuplot' to produce a plotted graphical
file...</jsdl:Description>
        </jsdl:JobIdentification>
        <jsdl:Application>
            <jsdl:ApplicationName>gnuplot</jsdl:ApplicationName>
            <jsdl-posix:POSIXApplication>
                <jsdl-posix:Executable>/usr/local/bin/gnuplot</jsdl-
posix:Executable>
                <jsdl-posix:Argument>control.txt</jsdl-posix:Argument>
                <jsdl-posix:Input>input.dat</jsdl-posix:Input>
                <jsdl-posix:Output>output1.png</jsdl-posix:Output>
            </jsdl-posix:POSIXApplication>
        </jsdl:Application>
        <jsdl:Resources>
            <jsdl:IndividualPhysicalMemory>
                <jsdl:LowerBoundedRange>2097152.0</jsdl:LowerBoundedRange>
            </jsdl:IndividualPhysicalMemory>
```

```
           <jsdl:TotalCPUCount>
               <Range>
               <LowerBound exclusiveBound="xsd:boolean">1.0</LowerBound>
               <UpperBound exclusiveBound="xsd:boolean">10.0</UpperBound>
               </Range>
           </jsdl:TotalCPUCount>
       </jsdl:Resources>
       <jsdl:DataStaging>
           <jsdl:FileName>control.txt</jsdl:FileName>
           <jsdl:CreationFlag>overwrite</jsdl:CreationFlag>
           <jsdl:DeleteOnTermination>true</jsdl:DeleteOnTermination>
           <jsdl:Source>
               <jsdl:URI>http://foo.bar.com/~me/control.txt</jsdl:URI>
           </jsdl:Source>
       </jsdl:DataStaging>
       <jsdl:DataStaging>
           <jsdl:FileName>input.dat</jsdl:FileName>
           <jsdl:CreationFlag>overwrite</jsdl:CreationFlag>
           <jsdl:DeleteOnTermination>true</jsdl:DeleteOnTermination>
           <jsdl:Source>
               <jsdl:URI>http://foo.bar.com/~me/input.dat</jsdl:URI>
           </jsdl:Source>
       </jsdl:DataStaging>
       <jsdl:DataStaging>
           <jsdl:FileName>output1.png</jsdl:FileName>
           <jsdl:CreationFlag>overwrite</jsdl:CreationFlag>
           <jsdl:DeleteOnTermination>true</jsdl:DeleteOnTermination>
           <jsdl:Target>
               <jsdl:URI>rsync://spoolmachine/userdir</jsdl:URI>
           </jsdl:Target>
       </jsdl:DataStaging>
   </jsdl:JobDescription>
</jsdl:JobDefinition>
```

A WPS implementation could use for example a JSDL template repository from which the WPS GetCapabilities response document could be created dynamically. Each template document from within the JSDL template repository influences the WPS GetCapabilities response document for example by inserting the JSDL `jsdl:JobName` element as process identifier and process title under the WPS `ProcessOfferings` element in the WPS GetCapabilities response document (see example below).

```
(...)
   <ProcessOfferings>
   <Process wps:processVersion="2">
       <Identifier>My Gnuplot invocation</Identifier>
       <Title>My Gnuplot invocation</Title>
   </Process>
       (...)
   </ProcessOfferings>
(...)
```

After receiving the list of all offered processes a client typically executes the WPS DescribeProcess operation for a specific process identifier. Assuming that the process identifier refers to an existing application in a Grid Computing infrastructure, a WPS ProcessDescription response document could be created for example from the corresponding template document from within the JSDL template repository. Some of the JSDL parameters should better not be modified (for example the JSDL `jsdl-`

`posix:Executable` parameter which directly refers to the deployed application binaries). But maybe a user or a client should be able to modify other JSDL parameters (for example the JSDL `jsdl:TotalCPUCount` parameter which specifies the total number of CPUs that are used for maybe parallel processing). Using XSLT transformation mechanisms, the WPS DescribeProcess response document could be created dynamically utilizing an XSLT document and the corresponding template document from within the JSDL template repository. Each template document from within the JSDL template repository influences the WPS DescribeProcess response document for example by inserting the JSDL `jsdl:JobName` element as process identifier and process title under the WPS `ProcessDescription` element. Each JSDL parameter value that should be affected by the user or the client could be inserted in a dynamic manner under the WPS `DataInputs` element (see example below).

```
(...)
    <ProcessDescription processVersion="2" statusSupported="true"
storeSupported="true">
        <Identifier>My Gnuplot invocation</Identifier>
        <Title>My Gnuplot invocation</Title>
        <Abstract>Simple application invocation: User wants to run the
application 'gnuplot' to produce a plotted graphical file...</Abstract>
        <DataInputs>
            <Input minOccurs="1" maxOccurs="1">
                <Identifier>TotalCPUCount</Identifier>
                <Title>TotalCPUCount</Title>
                <Abstract>Simple application invocation: User wants to run the
application 'gnuplot' to produce a plotted graphical file...</Abstract>
                <LiteralData>
                    <DataType ows:reference="xs:int"/>
                    <AllowedValues>
                        <Range>
                            <MinimumValue>1</MinimumValue>
                            <MaximumValue>10</MaximumValue>
                        </Range>
                    </AllowedValues>
                </LiteralData>
            </Input>
            (...)
        <DataInputs>
        (...)
    </ProcessDescription>
(...)
```

When the user or the client executes the WPS Execute operation for the specific process identifier with all required input parameters and assuming that the process identifier refers to an application in a Grid Computing infrastructure, a final JSDL document could be dynamically generated from the WPS Execute request document using XSLT transformation mechanisms again. From the following WPS Execute request document a corresponding JSDL document for the “gnuplot” application could be created with a XSLT document very easily.

```
(...)
<Execute version="1.0.0">
  <Identifier>My Gnuplot invocation</Identifier>
  <DataInputs>
```

```
    <Input>
      <Identifier>TotalCPUCount</Identifier>
      <Data>
        <LiteralData dataType="xs:int">8</LiteralData>
      </Data>
    </Input>
(...)
```

Figure 13 shows the overall transformation process from a JSDL document template to the XML WPS ProcessDescription response document and from a XML WPS Execute request document to a filled out JSDL document (assuming that each offered process consists of a JSDL template document and two XSLT Stylesheet). The WPS GetCapabilities response document could be for example a static file or dynamically created from a JSDL template document repository on the server side.

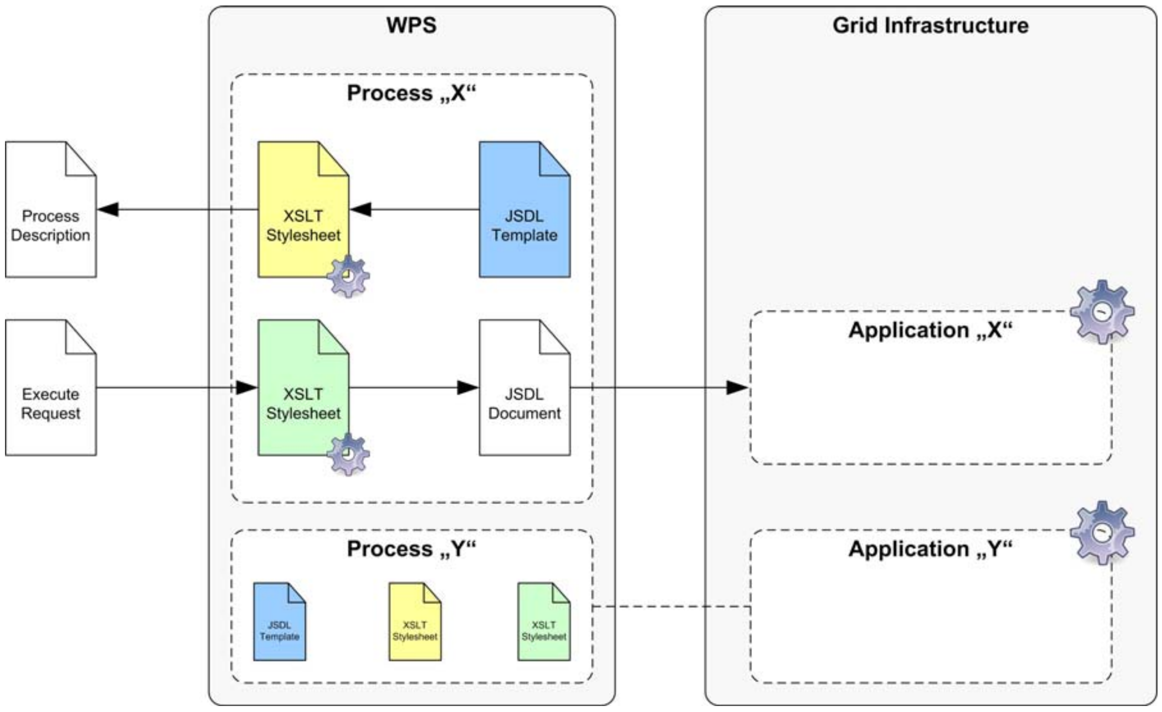


Figure 13 Generic approach for mapping JSDL to WPS and vice versa

Note, that this approach is not an alternative draft to the presented JSDL Profile approach (section 5.2.3). It focuses more on the way of creating dynamically the WPS ProcessDescription response document upon a JSDL document template and for generating dynamically a final JSDL document from a WPS Execute request document. The JSDL

5.3.4 Implementation

The implementation of the WPS Plume Service process is based on the 52° North implementation of WPS 1.0.0 standard[42] and the open source grid middleware UNICORE. An architecture overview of the provided Grid-enabled 52° North WPS implementation is shown in Figure 14. In OWS-6 the WPS Plume Service uses the UNCIORE Testgrid[43] at Jülich Supercomputing Centre (JSC)[44] at Research Center Jülich (FZJ)[45] for performing distributed processes.

For executing distributable processes, the implementation divides the processing problem into smaller sub-problems by splitting up the input data into smaller chunks. Then these chunks and necessary application binaries will be distributed upon several computational nodes in the grid. The number of computation nodes depends on the complexity of the process and can be either adjusted manually (for example dynamically through the WPS Interface or statically via a configuration file) or assessed automatically. After preparing each computational node by submitting the required input data and application binaries (Java libraries) onto each node, the application binaries will be executed in parallel. During this execution phase, the CPU at the WPS server side is idle and waiting for process results or other incoming requests. When all computational nodes have performed the execution successfully, the WPS fetches all result datasets and merges them together.

To improve service availability, incoming processing tasks are typically submitted for execution to available computing nodes on the Grid Computing infrastructure. To speed up performance, the processing tasks are typically divided into smaller sub-tasks if applicable (following a classic Divide-and-Conquer[46] approach and inspired by the Google Map/Reduce[47] framework). Geoprocesses are often considered to be very complex and therefore the effort to divide and conquer them is in most cases very high [10]. However, in some cases the Divide-and-Conquer approach is not applicable. However, if tasks are indivisible, the Grid Computing infrastructure can still be used to find the laziest node for processing.

[42] http://52north.org/

[43] http://www.unicore.eu/testgrid/

[44] http://www.fz-juelich.de/jsc/index.php?index=3

[45] http://www.fz-juelich.de/portal/home

[46] http://en.wikipedia.org/wiki/Divide_and_conquer_algorithm

[47] http://en.wikipedia.org/wiki/MapReduce

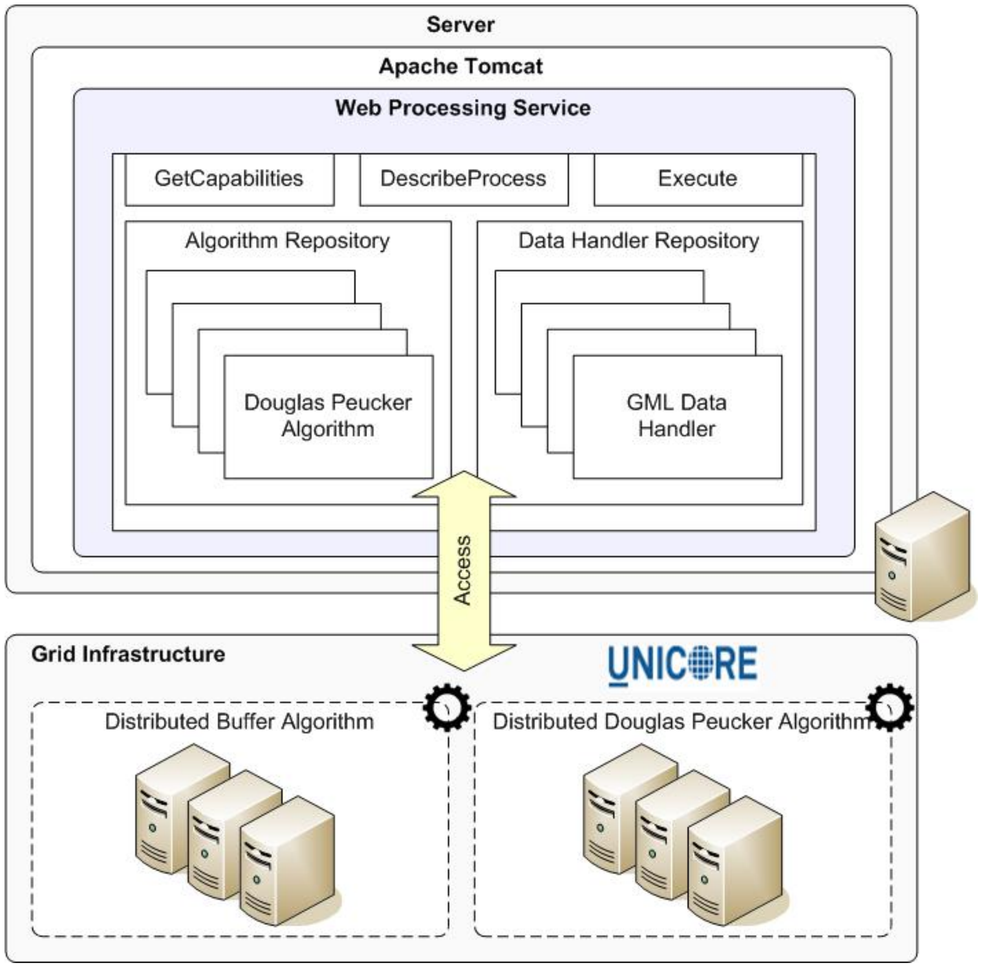


Figure 14 Architecture overview of Grid-enabled 52° North WPS implementation

A public available Endpoint Reference (EPR) to the developed Grid-enabled WPS implementation and the allocated process (namely the Plume Service process) is provided in Table 8.

Table 8 Plume Service process implementation links

GetCapabilities	http://giv-bandog.uni-muenster.de:8080/wps/WebProcessingService?Request=GetCapabilities&Service=WPS
DescribeProcess	http://giv-bandog.uni-muenster.de:8080/wps/WebProcessingService?Request=DescribeProcess&Service=WPS&Identifier=org.n52.wps.server.algorithm.simulation.PlumeTimeSeriesService
WSDL	http://giv-bandog.uni-muenster.de:8080/wps/PlumeTimeSeriesServiceWSDL.wsdl

The source code of the developed Grid-enabled WPS implementation is public available under the GPL license at the Open Source repository of 52° North.

5.3.5 Results

An example WPS Execute response document of a typical WPS Plume Service process run is listed in Appendix C.4.

Figure 15 shows the core result (a time-series of GeoTIFF files) of rendering the probable plume dispersion based on the GML output of the WPS Trajectory Service process.

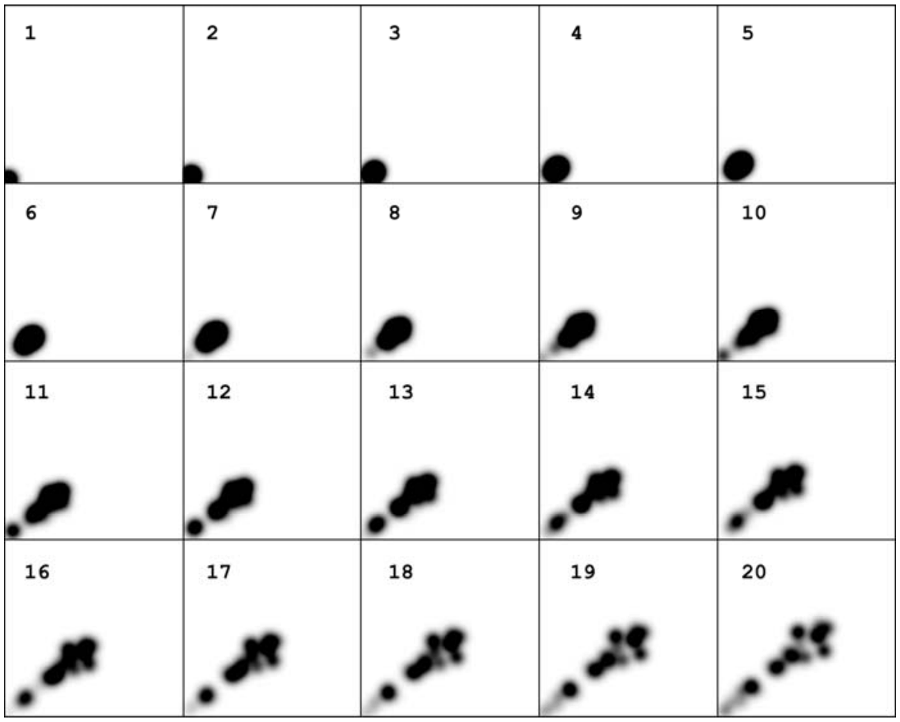


Figure 15 Example Plume Service output based on the results of Trajectory Service

5.4 Rainfall Interpolation

This chapter describes the WPS Rainfall Interpolation process that was developed by the Institute for Geoinformatics (IfGI) and the GIS Research Center (GIS.FCU) for use in the OWS-6 GPW Debris Flow Demonstration Scenario.

5.4.1 Description

The Rainfall Interpolation process implements an Inverse Distance Weighting (IDW)[48] algorithm and performs the following scenario-specific steps:

- Define the bounding box of map

[48] http://en.wikipedia.org/wiki/Inverse_distance_weighting

- Define the cell size
- For each cell, find out the nearest three or more rain gauge's rainfall value (r1, r2 and r3) and calculate the distance from cell to rain gauge (d1, d2 and d3)
- For each cell, calculate the rainfall value via IDW interpolation as follow:

```
value = ((d1^2)/(d1^2+d2^2+d3^2))*r1 + ((d2^2)/(d1^2+d2^2+d3^2))*r2 +
((d3^2)/(d1^2+d2^2+d3^2))*r3
```

Figure 16 shows several rain gauges's deployed at a fixed Debris Flow monitoring station and a grid for the interpolated rainfall data measurements.

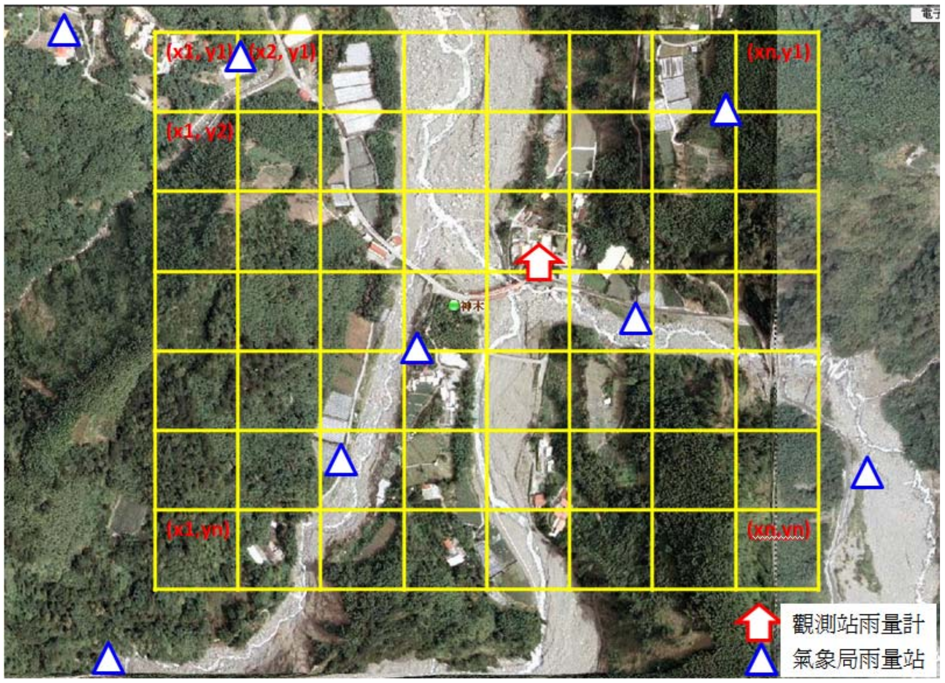


Figure 16 Interpolation for rain gauge measurements

5.4.2 Interface

The WPS Rainfall Interpolation process requires the following list of input parameters.

- The coordinates of left-top point of the map
- The coordinates of right-down point of the map
- The map will be cut into many little cells according to the cell size (each cell is a square and its width is cell size)

- For each cell, the n nearest stations will be chosen to calculate the rainfall interpolation

The process produces the following list of output parameters.

- The interpolated rainfall measurements

The parameter names, meanings, data types shall be as specified in Table 7.

Table 9 WPS Rainfall Interpolation process parameter overview

Parameters	I/O	Value Type	Note
RAINFALL [a]	Input	String	An URL to a text file containing the original rainfall data for rain gauges.
X1 [b]	Input	Double	The x coordinate of the right-down point of the map.
Y1 [b]	Input	Double	The y coordinate of the right-down point of the map.
X2 [b]	Input	Double	The x coordinate of the top-down point of the map.
Y2 [b]	Input	Double	The y coordinate of the top-down point of the map.
CELLWIDTH	Input	Double	The cell width of the map.
TOPN	Input	Integer	The number of nearest stations that will be chosen to calculate the rainfall interpolation.
INTERPOLATION [c]	Output	String	An URL to a text file containing the interpolated rainfall data for each cell of the map.
a The RAINFALL input parameter was realized through a LiteralData type input parameter instead of a (referenced) ComplexValue type input parameter because of simplicity and for testing purposes. b The BoundingBox input parameter type was not supported by the underlying original WPS implementation and so it was decided to submit the bounding box coordinates of the map as single input parameters. c The INTERPOLATION output parameter was realized through a LiteralData type input parameter instead of a (referenced) ComplexValue type input parameter because of simplicity and for testing purposes.			

An example SOAP-based WPS Execute request document is listed in Appendix C.5.

5.4.3 Mapping JSDL to WPS

The implementation of the WPS Rainfall Interpolation process is based on the same JSDL transforming mechanisms as described in section **Error! Reference source not found.**.

5.4.4 Implementation

The implementation of the WPS Rainfall Interpolation process is based on the 52° North implementation of WPS 1.0.0 standard and the open source grid middleware UNICORE. Further information about the implementation can be found in section 5.3.4.

A public available Endpoint Reference (EPR) to the developed Grid-enabled WPS implementation and the allocated process (namely the Rainfall Interpolation process) is provided in Table 8.

Table 10 Rainfall Interpolation process implementation links

GetCapabilities	http://giv-bandog.uni-muenster.de:8080/wps/WebProcessingService?Request=GetCapabilities&Service=WPS
DescribeProcess	http://giv-bandog.uni-muenster.de:8080/wps/WebProcessingService?Request=DescribeProcess&Service=WPS&Identifier=org.n52.wps.server.algorithm.interpolation.RainfallDataInterpolation
WSDL	n/a

The source code of the developed Grid-enabled WPS implementation is public available under the GPL license at the Open Source repository of 52° North.

5.4.5 Results

An example WPS Execute response document of a typical WPS Rainfall Interpolation process run is listed in Appendix C.6. Figure 17 shows the visualized result of such a typical run.

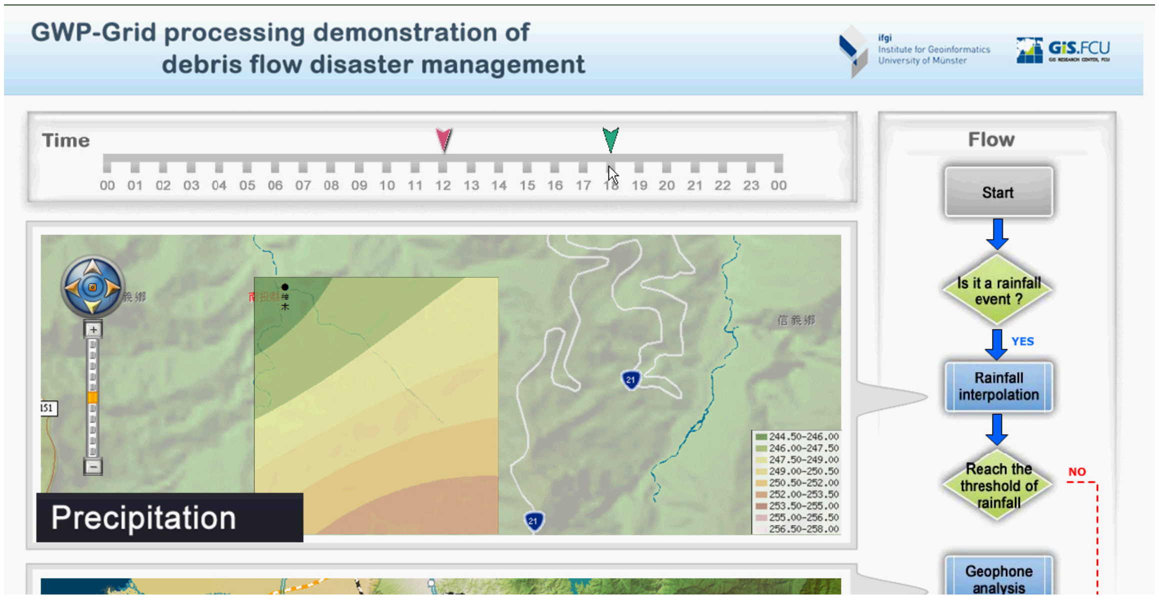


Figure 17 Visualization of rainfall interpolation (screenshot from OWS-6 demo video)

5.5 Geophone Analysis

This chapter describes the WPS Geophone Interpolation process that was developed by the Institute for Geoinformatics (IfGI) and the GIS Research Center (GIS.FCU) for use in the OWS-6 GPW Debris Flow Demonstration Scenario.

5.5.1 Description

The Geophone Interpolation process implements a Wavelet Transformation algorithm[49] and calculates if the ground vibration measured by the geophone sensors reaches a specific threshold.

5.5.2 Interface

The WPS Geophone Analysis process requires the following list of input parameters.

- The measurements of the geophone sensors (sound data)

The process produces the following list of output parameters.

- The transformed measurements of geophone sensors (transformed sound data)

The parameter names, meanings, data types shall be as specified in Table 7.

Table 11 WPS Geophone Analysis process parameter overview

Parameters	I/O	Value Type	Note
WAVELET [a]	Input	String	The URL of the original binary file for sound data.
TRANSFORMED WAVELET [b]	Output	String	An URL to a binary file containing the transformed wavlet data.
a The WAVELET input parameter was realized through a LiteralData type input parameter instead of a (referenced) ComplexValue type input parameter because of simplicity and for testing purposes.			
b The TRANSFORMED WAVELET input parameter was realized through a LiteralData type input parameter instead of a (referenced) ComplexValue type input parameter because of simplicity and for testing purposes.			

An example SOAP-based WPS Execute request document is listed in Appendix C.7.

5.5.3 Mapping JSDL to WPS

The implementation of the WPS Geophone Analysis process is based on the same JSDL transforming mechanisms as described in section **Error! Reference source not found.**.

[49] http://en.wikipedia.org/wiki/Wavelet_transform

5.5.4 Implementation

The implementation of the WPS Rainfall Interpolation process is based on the 52° North implementation of WPS 1.0.0 standard and the open source grid middleware UNICORE. Further information about the implementation can be found in section 5.3.4.

A public available Endpoint Reference (EPR) to the developed Grid-enabled WPS implementation and the allocated process (namely the Rainfall Interpolation process) is provided in Table 12.

Table 12 Geophone Analysis process implementation links

GetCapabilities	http://giv-bandog.uni-muenster.de:8080/wps/WebProcessingService?Request=GetCapabilities&Service=WPS
DescribeProcess	http://giv-bandog.uni-muenster.de:8080/wps/WebProcessingService?Request=DescribeProcess&Service=WPS&Identifier=org.n52.wps.server.algorithm.interpolation.WaveletTransformation
WSDL	n/a

The source code of the developed Grid-enabled WPS implementation is public available under the GPL license at the Open Source repository of 52° North.

5.5.5 Results

An example WPS Execute response document of a typical WPS Geophone Analysis process run is listed in Appendix C.8. Figure 18 shows the visualized result of such a typical run.

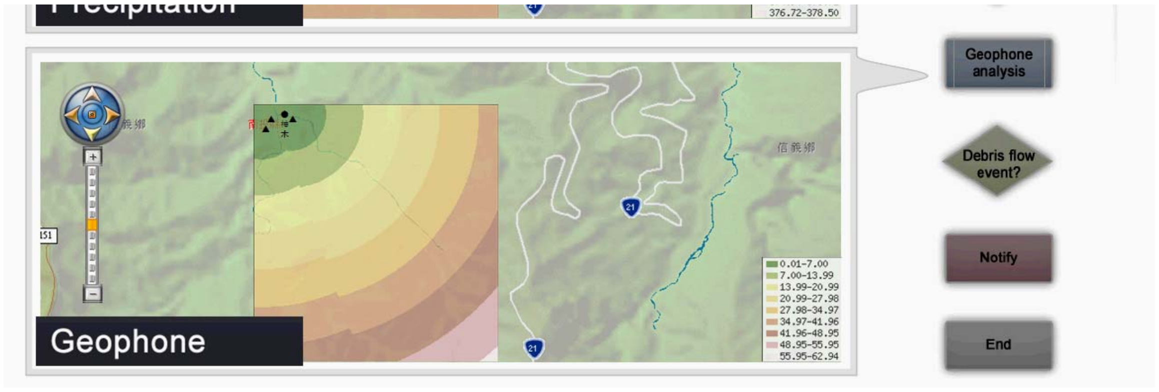


Figure 18 Visualization of geophone data (screenshot from OWS-6 demo video)

6 Web Service Resource Framework

The Open Grid Services Architecture (OGSA) describes an architecture for service-oriented Grid Computing environments and is based on several Web Service technologies (for example WSDL and SOAP). An integral part of OGSA and of widespread grid middleware solutions (for example the Globus Toolkit and UNICORE) is the Web Services Resource Framework (WSRF) specification which provides the users with the ability to access and manipulate state of web services. To realize entire benefit from existing grid middleware solutions (such as intelligent workflow management, security and other service qualities), the WPS must be completely integrated within grid middleware beside other Grid Services and therefore a WSRF binding for WPS must be defined and implemented.

6.1 Introduction

The Web Services Resource Framework (WSRF) is a set of specifications for accessing, managing and manipulating state of Web Services. It was actually originated in OGF but it was sent through the OASIS standardization process to get "buy-in" from the web services community. The concept of a WS-Resource[50] represents the combination of a Web Service with a stateful resource. In this context “state” refers to some information associated with a Web Service that persists over service interactions. Stateful Web services integrated into a Grid Computing infrastructure are also called Grid Services.

WSRF is based on WSDL for describing the Grid Service interface. Messages are encoded using a document/literal style SOAP binding as defined in the WS-I Basic Profile[51]. WS-Resources are interacted with according to the WS-Addressing[52] specification. This standard defines constructs to be submitted as a SOAP header in order to identify a reference to a Web service and send additional message information needed to access a specific WS-Resource (typically a URI).

6.2 Suggestions for a WSRF Binding for WPS

The nature of WSRF implies that a WSRF binding for WPS is also a SOAP/WSDL binding. A typical sample WPS process “buffer” could define a WSDL operation for the execute operation (in document/literal mode) as follows:

```
<types>
  <xsd:schema>
    <xsd:element name="BufferParameters" type="xsd:string" />
    <xsd:element name="BufferResponse" type="xsd:string" />
  </xsd:schema>
<types>

<message name="Execute_bufferRequest">
  <part name="parameters" element="tns:BufferParameters"/>
```

[50] http://docs.oasis-open.org/wsrf/wsrf-ws_resource-1.2-spec-os.pdf

[51] http://www.ws-i.org/Profiles/BasicProfile-1.0.html

[52] http://www.w3.org/Submission/ws-addressing/

```
</message>
<message name="Execute_bufferResponse">
  <part name="parameters" element="tns:BufferResponse"/>
</message>

<portType name="WPSPortType">
  <operation name="Execute_buffer">
    <input name="Execute_bufferRequest" message="tns:Execute_bufferRequest"/>
    <output name="Execute_bufferResponse"
message="tns:Execute_bufferResponse"/>
  </operation>
</portType>

<wsdl:binding name="WPSPortTypeSOAPBinding" type="tns:WPSPortType">
  <soap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="Execute_buffer">
    <soap:operation
soapAction="http://example.ns/WPSPortType/Execute_bufferRequest"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
  </wsdl:operation>
</wsdl:binding>
```

Further information about experiences with OGC Web Services and SOAP/WSDL can be found, among others, in the following documents:

- OGC 05-007r7, *OpenGIS® Web Processing Service Standard*
- OGC 06-094r1, *OWS Common change request: Add SOAP encoding*
- OGC 06-182r1, *OWS-4 WPS IPR - Discussions, findings, and use of WPS in OWS-4*
- OGC 07-158, *Wrapping OGC HTTP-GET and -POST Services with SOAP - Discussion Paper*
- OGC 08-009r1, *OWS 5 SOAP/WSDL Common Engineering Report*

The main difference that has to be dealt with is the stateful nature of Grid Services, for example how the WS-Resource and WPS standards can be combined. State information of a WS-Resource is stored as a set of so-called Resource Properties in conformance with the WS-ResourceProperties[53] specification. Resource properties have to be defined in an XML Schema document as part of the service's WSDL description. It is legitimate for a service to have an empty set of resource properties (no state). A simple, but correct, WSRF binding for WPS is thus a WS-Resource without state. However, we suggest a straightforward and unobtrusive way of using resource properties with WPS.

[53] http://docs.oasis-open.org/wsrf/wsrf-ws_resource_properties-1.2-spec-os.pdf

The following suggestions for a WSRF binding have been developed as part of the German GDI-Grid research project at the Department for River and Coastal Engineering[54] of Hamburg University of Technology[55]. A prototype for a WSRF-based WPS in the field of automatic mesh generation for hydraulic simulation models from digital terrain data was implemented. The developed WPS offers three processes (“createGrid”, “detectBreaklines” and “createTin”) that are generally called sequentially, the outputs of one process being the inputs for the next. The processes are provided as WSDL operations of a 3d Terrain Discretization Grid Service. Further information can be found in [12].

The Grid Service operations strongly depend on data produced by previous calls. In order to reduce the amount of data transferred between service invocations, only references to data is passed. The WPS 1.0.0 models this in form of `wps:InputReferenceType` or `wps:OutputReferenceType`. We use a compound parameter type in the WSDL binding that wraps a value (for example String, URI) together with the WPS parameter identifier (`ows:CodeType`).

In addition to providing the inputs to an `execute` operation directly, we support storing the parameters as a WSRF resource property of a WS-Resource and mark all parameters as optional. The identifier of a parameter is used as the identifier of a corresponding resource property. Properties defined in this way can serve either as input or as output parameters of a process. In case a process is called without directly providing a parameter, the input is taken from the current value of the parameter's resource property of the WS-Resource.

The necessary changes to the WSDL document can be seen here:

```
<types>
  <xsd:schema targetNamespace="http://example.ns">
    <xsd:complexType name=”BufferIOType”>
      <xsd:sequence>
        <xsd:element name="identifier" type=”ows:CodeType” />
        <xsd:element name="value" type=”xs:string” />
      </xsd:sequence>
    </xsd:complexType>
    <xsd:element name="BufferInput" type=”tns:BufferIOType” />
    <xsd:element name="BufferParameters">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element ref="tns:BufferInput" minOccurs="0" />
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
    <xsd:element name="BufferOutput" type=”tns:BufferIOType” />
    <xsd:element name="BufferResponse">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element ref="tns:BufferOutput" minOccurs="0" />
        </xsd:sequence>
      </xsd:complexType>
```

[54] http://www.tu-harburg.de/wb/english_site/

[55] http://www.tu-harburg.de/index_e.html

```
    </xsd:element>
    <xsd:element name="WPSResourceProperties">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element ref="tns:BufferInput" minOccurs="0" />
          <xsd:element ref="tns:BufferOutput" minOccurs="0" />
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
  </xsd:schema>
<types>
...
<portType name="WPSPortType"
wsrp:ResourceProperties="tns:WPSResourceProperties">
  <operation name="Execute_buffer">
    <input name="Execute_bufferRequest" message="tns:Execute_bufferRequest"/>
    <output name="Execute_bufferResponse"
message="tns:Execute_bufferResponse"/>
  </operation>
</portType>
```

The complete WSDL document for the elaborate 3d Terrain Discretization Service example (Gaja3dService) can be found in Appendix B4–B9. It has been implemented based on Globus Toolkit 4.

The procedure above greatly simplifies the interaction with the service in a typical usage scenario as the data is passed implicitly with the stateful service resource in between calls. This resource can be thought of as state information associated with the current session and can hold other information as well. For example, we can secure access to the service and its resource properties using standard grid security mechanisms such as authorization and authentication. Only the creator of a resource may access the results generated by the service.

An advantage of the presented approach is that with little extra effort it is possible to provide both the stateful and the stateless (pure WPS) interface in parallel. Using the WSRF bindings gives the caller advanced control over the data handled by the process, but the WPS interface allows regular clients to access the service as well. In this respect, the suggested WSRF binding is a conservative extension of the WPS capabilities with backward compatibility.

We are currently investigating other possibilities to make use of the potentials of WSRF for asynchronous WPS processes (with status information). The status document of a WPS (or parts of it) could also be provided as resource properties enabling clients to subscribe for notification when a process' status changes or when results are created. This would eliminate the need for status polling on the client side. Other use-cases may exist that the WSRF could easily support.

6.3 WSRF and other OGC interface standards

We consider some specific limitations of the OGC standards that would benefit from a refactoring under WSRF.

- *Service tied to data.* The OGC data services (WMS, WFS, WCS) make no distinction between the service instance and the data source – the data must be 'known' to the service. Typically, a data provider must also provide the OGC service. This can lead to 'brittle' architectures, and prohibits resource-sharing. Grid technology, on the other hand, encourages architectures that leverage multiple (separate) data and compute resources – a data provider need not also provide services.

- *Latency.* The OGC data services (WMS, WFS, WCS) operate in a synchronous request-response mode. There is no mechanism for handling latency associated with very large requests. Again, this leads to brittle architectures – especially where workflow requires data outputs to be chained to processing service inputs. The WSRF suites of specifications for Grid services provide standard notification mechanisms for dealing with latency.

- *Service coherence model.* There is no formal model integrating data resources behind OGC service instances – that is, there is no mechanism for asserting that the *same data* is exposed as visualization (WMS), gridded coverage (WCS) or feature instances (WFS). The *implied resource pattern* of WSRF, however, provides a mechanism for identifying unambiguously the data resource behind a service.

- *Workflow abstraction.* Since the OGC standards bind data and services, there is no mechanism to abstract workflows in terms only of data and operations.

- *OGC service consistency.* The OGC Web Processing Service (WPS) uses a different model for identifying data than the data services (WMS, WFS, WCS) – an external data resource may be identified by URL. Again, the use of WS-Addressing in WSRF offers an opportunity to unify the mechanism for identifying data across all the OGC services.

The 'implied resource pattern' of WSRF provides a standard mechanism for identifying the resource upon which a service invocation should act. In the case of the OGC web services, this allows a data resource to be identified for access (WFS, WCS), visualization (WMS), or processing (WPS). The WSRF 'Factory pattern' may be used to instantiate and manage the lifetime of data resources. This enables a looser coupling between data resources and the services that operate on them, and supports the possibility of novel WSRF workflow patterns (Figure 19).

In addition, the WS-Notification family of standards is a sister to WSRF, and provides a standard mechanism for asynchronous notifications to be provided on state changes of resources.

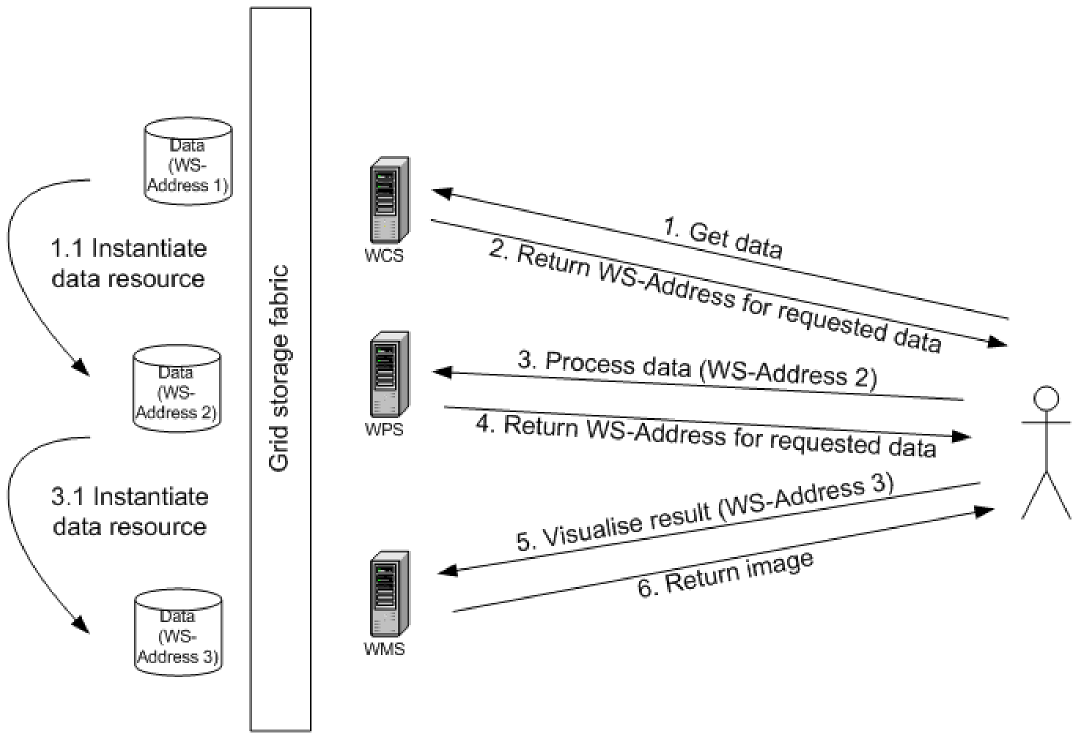

Data (WS-Address 1)
1.1 Instantiate data resource
Data (WS-Address 2)
3.1 Instantiate data resource
Data (WS-Address 3)
Grid storage fabric
WCS
WPS
WMS
1. Get data
2. Return WS-Address for requested data
3. Process data (WS-Address 2)
4. Return WS-Address for requested data
5. Visualise result (WS-Address 3)
6. Return image

Figure 19: Example WSRF workflow with WCS, WPS, WMS (from [18]).

7 Security Mechanisms

For invoking computational processes and accessing distributed resources in a Grid Computing environment (for example through an OGF "encapsulated" WPS), the job executor must authenticate himself against the specific Grid infrastructure (for example UK NGS or the German D-GRID). The Grid Security Infrastructure (GSI)[56] [8] provides secure, robust and interoperable communication with the Grid. Authentication in the GSI architecture is based on public-key infrastructure (PKI) mechanisms with x.509 certificates containing information for identifying users or services. Further information can be found in the HPC Basic Profile specification[57].

In general two different ways for realizing authentication and secure access to a Grid infrastructure can be identified.

- Server authentication

  There is one x.509 certificate per server and so there is no reason for sending user-specific security credentials in the request. The WPS implementation presented in section 5.3.4 realizes the authentication against and secure combination with the UNICORE Testgrid at JSC in this manner.

[56] http://www.globus.org/security/overview.html

[57] http://www.ogf.org/documents/GFD.114.pdf

- User authentication

  The WPS client could send security credentials (for example username/password or other user token) via the standard WPS interface to the server. So the user-specific x.509 certificate can be generated on-the-fly (for example from a certificate repository at server site). The utilization of MyProxy security credential management service is a common way for creating proxy-certificates for accessing a Grid infrastructure. The WPS implementation presented in section 5.2.4 realizes the authentication against and secure combination with the UK NGS in this manner. But unencrypted sending of security credentials is a big security hole and so it is strongly recommended to use for example WS-Security (Web Services Security)[58] for establishing a secure communication between a user/client and the WPS.

In the OWS-6 Interoperability testbed not very much work on these security topics had been done, recognizing that at the same time a specific security thread was addressing these issues in detail. It is expected that security approaches developed in that thread could be integrated with Grid security mechanisms in future work. The WPS Trajectory Service process realizes user authentication via sending a username/password security token directly through the WPS interface and utilize a MyProxy infrastructure for generating proxy certificates for accessing the Grid on-the-fly.

[58] http://en.wikipedia.org/wiki/WS-Security

Annex A

XML Documents

A.1 Trajectory Service process (WPS DescribeProcess response)

```
<?xml version="1.0" encoding="UTF-8"?>
<wps:ProcessDescriptions xmlns:wps="http://www.opengis.net/wps/1.0.0"
xmlns:ows="http://www.opengis.net/ows/1.1"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" service="WPS"
version="1.0.0" xml:lang="en-GB"
xsi:schemaLocation="http://www.opengis.net/wps/1.0.0
http://schemas.opengis.net/wps/1.0.0/wpsDescribeProcess_response.xsd">
    <ProcessDescription wps:processVersion="1.0" storeSupported="true"
statusSupported="true">
        <ows:Identifier>RunTraj</ows:Identifier>
        <ows:Title>BADC Trajectory Service</ows:Title>
        <ows:Abstract>The service provides a user friendly interface to an
atmospheric trajectory model.</ows:Abstract>
        <ows:Metadata xlink:title="none"/>
        <ows:Metadata xlink:title="none"/>
        <DataInputs>
            <Input minOccurs="0" maxOccurs="1">
                <ows:Identifier>Username</ows:Identifier>
                <ows:Title>Username</ows:Title>
                <ows:Abstract>Username</ows:Abstract>
                <LiteralData>
                    <ows:AnyValue/>
                </LiteralData>
            </Input>
            <Input minOccurs="0" maxOccurs="1">
                <ows:Identifier>JSDL</ows:Identifier>
                <ows:Title>JSDL</ows:Title>
                <ows:Abstract>JSDL</ows:Abstract>
                <LiteralData>
                    <ows:AnyValue/>
                </LiteralData>
            </Input>
            <Input minOccurs="0" maxOccurs="1">
                <ows:Identifier>Lon</ows:Identifier>
                <ows:Title>Lon</ows:Title>
                <ows:Abstract>Lon</ows:Abstract>
                <LiteralData>
                    <ows:AnyValue/>
                </LiteralData>
            </Input>
            <Input minOccurs="1" maxOccurs="1">
                <ows:Identifier>Datetime</ows:Identifier>
                <ows:Title>Datetime</ows:Title>
                <ows:Abstract>Datetime</ows:Abstract>
                <LiteralData>
                    <ows:AnyValue/>
                </LiteralData>
            </Input>
            <Input minOccurs="1" maxOccurs="1">
                <ows:Identifier>Source</ows:Identifier>
                <ows:Title>Source</ows:Title>
                <ows:Abstract>Source</ows:Abstract>
                <LiteralData>
                    <ows:AnyValue/>
                    <DefaultValue>ecmwf_2.5p</DefaultValue>
                </LiteralData>
            </Input>
            <Input minOccurs="0" maxOccurs="1">
                <ows:Identifier>Length</ows:Identifier>
                <ows:Title>Length</ows:Title>
                <ows:Abstract>Length</ows:Abstract>
                <LiteralData>
                    <ows:AnyValue/>
                </LiteralData>
            </Input>
            <Input minOccurs="0" maxOccurs="1">
                <ows:Identifier>MyProxy</ows:Identifier>
                <ows:Title>MyProxy</ows:Title>
                <ows:Abstract>MyProxy</ows:Abstract>
                <LiteralData>
                    <ows:AnyValue/>
                </LiteralData>
            </Input>
            <Input minOccurs="0" maxOccurs="1">
                <ows:Identifier>Lev</ows:Identifier>
                <ows:Title>Lev</ows:Title>
                <ows:Abstract>Lev</ows:Abstract>
                <LiteralData>
                    <ows:AnyValue/>
                </LiteralData>
            </Input>
            <Input minOccurs="0" maxOccurs="1">
                <ows:Identifier>Lat</ows:Identifier>
                <ows:Title>Lat</ows:Title>
                <ows:Abstract>Lat</ows:Abstract>
                <LiteralData>
                    <ows:AnyValue/>
                </LiteralData>
            </Input>
            <Input minOccurs="0" maxOccurs="1">
                <ows:Identifier>Time_step</ows:Identifier>
                <ows:Title>Time_step</ows:Title>
                <ows:Abstract>Time_step</ows:Abstract>
                <LiteralData>
                    <ows:AnyValue/>
                </LiteralData>
            </Input>
        </DataInputs>
        <ProcessOutputs>
            <Output>
                <ows:Identifier>output</ows:Identifier>
                <ows:Title>output</ows:Title>
                <ows:Abstract>output</ows:Abstract>
                <ComplexOutput>
                    <Default>
                        <Format>
                           <MimeType>text/xml</MimeType>
                           <Encoding>UTF-8</Encoding>

     <Schema>http://schemas.opengis.net/gml/3.2.1/gml.xsd</Schema>
                        </Format>
                    </Default>
                    <Supported>
                        <Format>
                           <MimeType>text/xml</MimeType>
```

```
                          <Encoding>UTF-8</Encoding>

    <Schema>http://schemas.opengis.net/gml/3.2.1/gml.xsd</Schema>
                      </Format>
                  </Supported>
              </ComplexOutput>
          </Output>
      </ProcessOutputs>
    </ProcessDescription>
</wps:ProcessDescriptions>
```

A.2 Plume Service process (WPS DescribeProcess response)

```
<?xml version="1.0" encoding="UTF-8"?>
<ns:ProcessDescriptions xmlns:ns="http://www.opengis.net/wps/1.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/wps/1.0.0
http://schemas.opengis.net/wps/1.0.0/wpsDescribeProcess_response.xsd"
xml:lang="en-US" service="WPS" version="1.0.0">
    <ProcessDescription xmlns:wps="http://www.opengis.net/wps/1.0.0"
xmlns:ows="http://www.opengis.net/ows/1.1"
xmlns:xlink="http://www.w3.org/1999/xlink" wps:processVersion="2"
statusSupported="true" storeSupported="true">
    <ows:Identifier>org.n52.wps.server.algorithm.simulation.PlumeTimeSeriesServi
ce</ows:Identifier>
        <ows:Title>Plume Forecast</ows:Title>
        <ows:Abstract>Plume Forecast.</ows:Abstract>
        <DataInputs>
            <Input minOccurs="1" maxOccurs="1">
                <ows:Identifier>TRAJECTORY</ows:Identifier>
                <ows:Title>The location (HTTP-URL) of the to the GML output data
from Trajectory Service (STFC).</ows:Title>
                <ows:Abstract>The location (HTTP-URL) of the to the GML output
data from Trajectory Service (STFC).</ows:Abstract>
                <LiteralData>
                    <ows:DataType ows:reference="xs:string"/>
                    <ows:AllowedValues>
                        <ows:Value/>
                    </ows:AllowedValues>
                </LiteralData>
            </Input>
            <Input minOccurs="1" maxOccurs="1">
                <ows:Identifier>WIDTH</ows:Identifier>
                <ows:Title>The number of pixel columns (width).</ows:Title>
                <ows:Abstract>The number of pixel columns (width).</ows:Abstract>
                <LiteralData>
                    <ows:DataType ows:reference="xs:int"/>
                    <ows:AllowedValues>
                        <ows:Range>
                            <ows:MinimumValue>0</ows:MinimumValue>
                            <ows:MaximumValue>800</ows:MaximumValue>
                        </ows:Range>
                    </ows:AllowedValues>
                </LiteralData>
            </Input>
            <Input minOccurs="1" maxOccurs="1">
                <ows:Identifier>HEIGHT</ows:Identifier>
                <ows:Title>The number of pixel rows (height).</ows:Title>
                <ows:Abstract>The number of pixel rows (height).</ows:Abstract>
                <LiteralData>
                    <ows:DataType ows:reference="xs:int"/>
                    <ows:AllowedValues>
                        <ows:Range>
                            <ows:MinimumValue>0</ows:MinimumValue>
                            <ows:MaximumValue>800</ows:MaximumValue>
                        </ows:Range>
                    </ows:AllowedValues>
                </LiteralData>
            </Input>
            <Input minOccurs="1" maxOccurs="1">
                <ows:Identifier>IndividualPhysicalMemory</ows:Identifier>
                <ows:Title>Individual Physical Memory</ows:Title>
                <ows:Abstract>This element is a range value specifying the amount
of physical memory required on each individual resource. The amount is given in
bytes.</ows:Abstract>
                <LiteralData>
                    <ows:DataType ows:reference="xs:int"/>
                    <ows:AllowedValues>
                        <ows:Range>
                            <ows:MinimumValue>268435456</ows:MinimumValue>
                            <ows:MaximumValue>536870912</ows:MaximumValue>
                        </ows:Range>
                    </ows:AllowedValues>
                </LiteralData>
            </Input>
            <Input minOccurs="1" maxOccurs="1">
                <ows:Identifier>IndividualCPUTime</ows:Identifier>
                <ows:Title>Individual CPU Time</ows:Title>
                <ows:Abstract>This element is a range value specifying the total
number of CPU seconds required on each resource to execute the
job.</ows:Abstract>
                <LiteralData>
                    <ows:DataType ows:reference="xs:int"/>
                    <ows:AllowedValues>
                        <ows:Range>
                            <ows:MinimumValue>60</ows:MinimumValue>
                            <ows:MaximumValue>1800</ows:MaximumValue>
                        </ows:Range>
                    </ows:AllowedValues>
                </LiteralData>
            </Input>
            <Input minOccurs="1" maxOccurs="1">
                <ows:Identifier>TotalCPUCount</ows:Identifier>
                <ows:Title>Total CPU Count</ows:Title>
                <ows:Abstract>This element is a range value specifying the total
number of CPUs required for this job submission.</ows:Abstract>
                <LiteralData>
                    <ows:DataType ows:reference="xs:int"/>
                    <ows:AllowedValues>
                        <ows:Range>
                            <ows:MinimumValue>1</ows:MinimumValue>
                            <ows:MaximumValue>6</ows:MaximumValue>
                        </ows:Range>
                    </ows:AllowedValues>
                </LiteralData>
            </Input>
        </DataInputs>
        <ProcessOutputs>
            <Output>
                <ows:Identifier>IMAGE_0</ows:Identifier>
                <ows:Title>One of the rendered image files.</ows:Title>
                <ows:Abstract>One of the rendered image files</ows:Abstract>
                <ComplexOutput>
                    <Default>
                        <Format>
                            <MimeType>image/tiff</MimeType>
                        </Format>
                    </Default>
                    <Supported>
                        <Format>
                            <MimeType>image/tiff</MimeType>
                        </Format>
                    </Supported>
                </ComplexOutput>
            </Output>
            <Output>
                <ows:Identifier>IMAGE_1</ows:Identifier>
                <ows:Title>One of the rendered image files.</ows:Title>
                <ows:Abstract>One of the rendered image files</ows:Abstract>
                <ComplexOutput>
                    <Default>
                        <Format>
                            <MimeType>image/tiff</MimeType>
                        </Format>
                    </Default>
                    <Supported>
                        <Format>
                            <MimeType>image/tiff</MimeType>
                        </Format>
                    </Supported>
                </ComplexOutput>
            </Output>
            <Output>
                <ows:Identifier>IMAGE_2</ows:Identifier>
                <ows:Title>One of the rendered image files.</ows:Title>
                <ows:Abstract>One of the rendered image files</ows:Abstract>
                <ComplexOutput>
                    <Default>
                        <Format>
                            <MimeType>image/tiff</MimeType>
                        </Format>
                    </Default>
                    <Supported>
                        <Format>
                            <MimeType>image/tiff</MimeType>
                        </Format>
                    </Supported>
                </ComplexOutput>
            </Output>
            <Output>
                <ows:Identifier>IMAGE_3</ows:Identifier>
                <ows:Title>One of the rendered image files.</ows:Title>
                <ows:Abstract>One of the rendered image files</ows:Abstract>
                <ComplexOutput>
                    <Default>
                        <Format>
                            <MimeType>image/tiff</MimeType>
                        </Format>
                    </Default>
                    <Supported>
                        <Format>
                            <MimeType>image/tiff</MimeType>
                        </Format>
                    </Supported>
                </ComplexOutput>
            </Output>
            <Output>
                <ows:Identifier>IMAGE_4</ows:Identifier>
                <ows:Title>One of the rendered image files.</ows:Title>
                <ows:Abstract>One of the rendered image files</ows:Abstract>
                <ComplexOutput>
```

```
            <Default>
                <Format>
                    <MimeType>image/tiff</MimeType>
                </Format>
            </Default>
            <Supported>
                <Format>
                    <MimeType>image/tiff</MimeType>
                </Format>
            </Supported>
        </ComplexOutput>
    </Output>
    <Output>
        <ows:Identifier>IMAGE_5</ows:Identifier>
        <ows:Title>One of the rendered image files.</ows:Title>
        <ows:Abstract>One of the rendered image files</ows:Abstract>
        <ComplexOutput>
            <Default>
                <Format>
                    <MimeType>image/tiff</MimeType>
                </Format>
            </Default>
            <Supported>
                <Format>
                    <MimeType>image/tiff</MimeType>
                </Format>
            </Supported>
        </ComplexOutput>
    </Output>
    <Output>
        <ows:Identifier>IMAGE_6</ows:Identifier>
        <ows:Title>One of the rendered image files.</ows:Title>
        <ows:Abstract>One of the rendered image files</ows:Abstract>
        <ComplexOutput>
            <Default>
                <Format>
                    <MimeType>image/tiff</MimeType>
                </Format>
            </Default>
            <Supported>
                <Format>
                    <MimeType>image/tiff</MimeType>
                </Format>
            </Supported>
        </ComplexOutput>
    </Output>
    <Output>
        <ows:Identifier>IMAGE_7</ows:Identifier>
        <ows:Title>One of the rendered image files.</ows:Title>
        <ows:Abstract>One of the rendered image files</ows:Abstract>
        <ComplexOutput>
            <Default>
                <Format>
                    <MimeType>image/tiff</MimeType>
                </Format>
            </Default>
            <Supported>
                <Format>
                    <MimeType>image/tiff</MimeType>
                </Format>
            </Supported>
        </ComplexOutput>
    </Output>
    <Output>
```

```
                <ows:Identifier>IMAGE_8</ows:Identifier>
                <ows:Title>One of the rendered image files.</ows:Title>
                <ows:Abstract>One of the rendered image files</ows:Abstract>
                <ComplexOutput>
                    <Default>
                        <Format>
                            <MimeType>image/tiff</MimeType>
                        </Format>
                    </Default>
                    <Supported>
                        <Format>
                            <MimeType>image/tiff</MimeType>
                        </Format>
                    </Supported>
                </ComplexOutput>
            </Output>
            <Output>
                <ows:Identifier>IMAGE_9</ows:Identifier>
                <ows:Title>One of the rendered image files.</ows:Title>
                <ows:Abstract>One of the rendered image files</ows:Abstract>
                <ComplexOutput>
                    <Default>
                        <Format>
                            <MimeType>image/tiff</MimeType>
                        </Format>
                    </Default>
                    <Supported>
                        <Format>
                            <MimeType>image/tiff</MimeType>
                        </Format>
                    </Supported>
                </ComplexOutput>
            </Output>
        </ProcessOutputs>
    </ProcessDescription>
</ns:ProcessDescriptions>
```

A.3 Rainfall Interpolation process (WPS DescribeProcess response)

```
<?xml version="1.0" encoding="UTF-8"?>
<ns:ProcessDescriptions xmlns:ns="http://www.opengis.net/wps/1.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/wps/1.0.0
http://schemas.opengis.net/wps/1.0.0/wpsDescribeProcess_response.xsd"
xml:lang="en-US" service="WPS" version="1.0.0">
    <ProcessDescription xmlns:wps="http://www.opengis.net/wps/1.0.0"
xmlns:ows="http://www.opengis.net/ows/1.1"
xmlns:xlink="http://www.w3.org/1999/xlink" wps:processVersion="2"
statusSupported="true" storeSupported="true">

    <ows:Identifier>org.n52.wps.server.algorithm.interpolation.RainfallDataInter
polation</ows:Identifier>
        <ows:Title>Rainfall Interpolation Service</ows:Title>
        <ows:Abstract>Rainfall Interpolation Service</ows:Abstract>
        <DataInputs>
            <Input minOccurs="1" maxOccurs="1">
                <ows:Identifier>RAINFALL</ows:Identifier>
                <ows:Title>Station Rainfall Data</ows:Title>
                <ows:Abstract>An HTTP URL to a file that includes the data of
each observation station.</ows:Abstract>
                <ComplexData>
                    <Default>
                        <Format>
```

```
                            <MimeType>text/XML</MimeType>
                            <Schema>http://giv-bandog.uni-
muenster.de:8080/wps/RainfallDataInterpolation.xsd</Schema>
                        </Format>
                    </Default>
                    <Supported>
                        <Format>
                            <MimeType>text/XML</MimeType>
                            <Schema>http://giv-bandog.uni-
muenster.de:8080/wps/RainfallDataInterpolation.xsd</Schema>
                        </Format>
                    </Supported>
                </ComplexData>
            </Input>
            <Input minOccurs="1" maxOccurs="1">
                <ows:Identifier>X1</ows:Identifier>
                <ows:Title>leftTopX</ows:Title>
                <ows:Abstract>The coordinates of leftTop Point of the map is (x1,
y1).</ows:Abstract>
                <LiteralData>
                    <ows:DataType ows:reference="xs:double"/>
                    <ows:AllowedValues>
                        <ows:Value/>
                    </ows:AllowedValues>
                </LiteralData>
            </Input>
            <Input minOccurs="1" maxOccurs="1">
                <ows:Identifier>Y1</ows:Identifier>
                <ows:Title>leftTopY</ows:Title>
                <ows:Abstract>The coordinates of leftTop Point of the map is (x1,
y1).</ows:Abstract>
                <LiteralData>
                    <ows:DataType ows:reference="xs:double"/>
                    <ows:AllowedValues>
                        <ows:Value/>
                    </ows:AllowedValues>
                </LiteralData>
            </Input>
            <Input minOccurs="1" maxOccurs="1">
                <ows:Identifier>X2</ows:Identifier>
                <ows:Title>rightDownX</ows:Title>
                <ows:Abstract>The coordinates of rightDown Point of the map is
(x2, y2).</ows:Abstract>
                <LiteralData>
                    <ows:DataType ows:reference="xs:double"/>
                    <ows:AllowedValues>
                        <ows:Value/>
                    </ows:AllowedValues>
                </LiteralData>
            </Input>
            <Input minOccurs="1" maxOccurs="1">
                <ows:Identifier>Y2</ows:Identifier>
                <ows:Title>rightDownY</ows:Title>
                <ows:Abstract>The coordinates of rightDown Point of the map is
(x2, y2).</ows:Abstract>
                <LiteralData>
                    <ows:DataType ows:reference="xs:double"/>
                    <ows:AllowedValues>
                        <ows:Value/>
                    </ows:AllowedValues>
                </LiteralData>
            </Input>
            <Input minOccurs="1" maxOccurs="1">
                <ows:Identifier>CELLWIDTH</ows:Identifier>
                <ows:Title>cellWidth</ows:Title>
                <ows:Abstract>Each cell is a square and its width is cellWidth 
(Meter).</ows:Abstract>
                <LiteralData>
                    <ows:DataType ows:reference="xs:double"/>
                    <ows:AllowedValues>
                        <ows:Value/>
                    </ows:AllowedValues>
                </LiteralData>
            </Input>
            <Input minOccurs="1" maxOccurs="1">
                <ows:Identifier>TOPN</ows:Identifier>
                <ows:Title>topN</ows:Title>
                <ows:Abstract>For each cell, we choose topN nearest stations to 
calculate the result.</ows:Abstract>
                <LiteralData>
                    <ows:DataType ows:reference="xs:int"/>
                    <ows:AllowedValues>
                        <ows:Range>
                            <ows:MinimumValue>1</ows:MinimumValue>
                        </ows:Range>
                    </ows:AllowedValues>
                </LiteralData>
            </Input>
        </DataInputs>
        <ProcessOutputs>
            <Output>
                <ows:Identifier>INTERPOLATION</ows:Identifier>
                <ows:Title>An HTTP URL to a file that includes the interpolated 
rainfall data for each cell.</ows:Title>
                <ows:Abstract>An HTTP URL to a file that includes the 
interpolated rainfall data for each cell.</ows:Abstract>
                <LiteralOutput>
                    <ows:DataType ows:reference="xs:string"/>
                </LiteralOutput>
            </Output>
        </ProcessOutputs>
    </ProcessDescription>
</ns:ProcessDescriptions>
```

A.4 Geophone Analysis process (WPS DescribeProcess response)

```
<?xml version="1.0" encoding="UTF-8"?>
<ns:ProcessDescriptions xmlns:ns="http://www.opengis.net/wps/1.0.0" 
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" 
xsi:schemaLocation="http://www.opengis.net/wps/1.0.0 
http://schemas.opengis.net/wps/1.0.0/wpsDescribeProcess_response.xsd" 
xml:lang="en-US" service="WPS" version="1.0.0">
    <ProcessDescription xmlns:wps="http://www.opengis.net/wps/1.0.0" 
xmlns:ows="http://www.opengis.net/ows/1.1" 
xmlns:xlink="http://www.w3.org/1999/xlink" wps:processVersion="1" 
statusSupported="true" storeSupported="true">

    <ows:Identifier>org.n52.wps.server.algorithm.interpolation.WaveletTransforma
tion</ows:Identifier>
        <ows:Title>Wavelet Transformation Service</ows:Title>
        <ows:Abstract>Wavelet Transformation Service</ows:Abstract>
        <DataInputs>
            <Input minOccurs="1" maxOccurs="1">
                <ows:Identifier>WAVELET</ows:Identifier>
                <ows:Title>Original brinary file for sound data.</ows:Title>
                <ows:Abstract>The location (HTTP-URL) of the original brinary 
file for sound data.</ows:Abstract>
```

```
               <LiteralData>
                   <ows:DataType ows:reference="xs:string"/>
                   <ows:AllowedValues>
                       <ows:Value/>
                   </ows:AllowedValues>
               </LiteralData>
           </Input>
       </DataInputs>
       <ProcessOutputs>
           <Output>
               <ows:Identifier>TRANSFORMED WAVELET</ows:Identifier>
               <ows:Title>The location (HTTP-URL) of the transformed wavelet
file.</ows:Title>
               <ows:Abstract>The location (HTTP-URL) of the transformed wavelet
file.</ows:Abstract>
               <LiteralOutput>
                   <ows:DataType ows:reference="xs:string"/>
               </LiteralOutput>
           </Output>
       </ProcessOutputs>
   </ProcessDescription>
</ns:ProcessDescriptions>
```

Annex B

WSDL Documents

B.1 Trajectory Service process

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:stfc="http://www.stfc.ac.uk" xmlns:ns1="http://org.apache.axis2/xsd"
xmlns:wsaw="http://www.w3.org/2006/05/addressing/wsdl"
xmlns:wps="http://www.opengis.net/wps/1.0.0" xmlns:tns="http://www.stfc.ac.uk"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
targetNamespace="http://www.stfc.ac.uk">
    <wsdl:documentation>WpsJsdl</wsdl:documentation>
    <wsdl:types>
        <xs:schema attributeFormDefault="qualified"
elementFormDefault="qualified" targetNamespace="http://www.stfc.ac.uk">
            <xs:import namespace="http://www.opengis.net/wps/1.0.0"
schemaLocation="http://schemas.opengis.net/wps/1.0.0/wpsAll.xsd"/>
            <xs:element name="Execute_TrajService">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element ref="wps:Execute"/>
                        <xs:element minOccurs="1" name="Username"
type="xs:string"/>
                        <xs:element minOccurs="1" name="Password"
type="xs:string"/>
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
        </xs:schema>
    </wsdl:types>
    <wsdl:message name="GetCapabilitiesRequest">
        <wsdl:part name="parameters" element="wps:GetCapabilities"/>
    </wsdl:message>
    <wsdl:message name="GetCapabilitiesResponse">
        <wsdl:part name="parameters" element="wps:Capabilities"/>
    </wsdl:message>
    <wsdl:message name="Describe_TrajServiceRequest">
        <wsdl:part name="parameters" element="wps:DescribeProcess"/>
    </wsdl:message>
    <wsdl:message name="Describe_TrajServiceResponse">
        <wsdl:part name="parameters" element="wps:ProcessDescriptions"/>
    </wsdl:message>
    <wsdl:message name="Execute_TrajServiceRequest">
        <wsdl:part name="parameters" element="stfc:Execute_TrajService"/>
    </wsdl:message>
    <wsdl:message name="Execute_TrajServiceResponse">
        <wsdl:part name="parameters" element="wps:ExecuteResponse"/>
    </wsdl:message>
    <wsdl:portType name="WpsJsdlPortType">
        <wsdl:operation name="GetCapabilities">
            <wsdl:input message="stfc:GetCapabilitiesRequest"
wsaw:Action="urn:GetCapabilities"/>
```

```
            <wsdl:output message="stfc:GetCapabilitiesResponse"
wsaw:Action="urn:GetCapabilitiesResponse"/>
        </wsdl:operation>
        <wsdl:operation name="Describe_TrajService">
            <wsdl:input message="stfc:Describe_TrajServiceRequest"
wsaw:Action="urn:Describe_TrajService"/>
            <wsdl:output message="stfc:Describe_TrajServiceResponse"
wsaw:Action="urn:Describe_TrajServiceResponse"/>
        </wsdl:operation>
        <wsdl:operation name="Execute_TrajService">
            <wsdl:input message="stfc:Execute_TrajServiceRequest"
wsaw:Action="urn:Execute_TrajService"/>
            <wsdl:output message="stfc:Execute_TrajServiceResponse"
wsaw:Action="urn:Execute_TrajServiceResponse"/>
        </wsdl:operation>
    </wsdl:portType>
    <wsdl:binding name="WpsJsdlSoap11Binding" type="tns:WpsJsdlPortType">
        <soap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
        <wsdl:operation name="Describe_TrajService">
            <soap:operation soapAction="urn:Describe_TrajService"
style="document"/>
            <wsdl:input>
                <soap:body use="literal"/>
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal"/>
            </wsdl:output>
        </wsdl:operation>
        <wsdl:operation name="Execute_TrajService">
            <soap:operation soapAction="urn:Execute_TrajService"
style="document"/>
            <wsdl:input>
                <soap:body use="literal"/>
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal"/>
            </wsdl:output>
        </wsdl:operation>
        <wsdl:operation name="GetCapabilities">
            <soap:operation soapAction="urn:GetCapabilities" style="document"/>
            <wsdl:input>
                <soap:body use="literal"/>
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal"/>
            </wsdl:output>
        </wsdl:operation>
    </wsdl:binding>
    <wsdl:binding name="WpsJsdlSoap12Binding" type="tns:WpsJsdlPortType">
        <soap12:binding transport="http://schemas.xmlsoap.org/soap/http"
style="document"/>
        <wsdl:operation name="Describe_TrajService">
            <soap12:operation soapAction="urn:Describe_TrajService"
style="document"/>
            <wsdl:input>
                <soap12:body use="literal"/>
            </wsdl:input>
            <wsdl:output>
                <soap12:body use="literal"/>
            </wsdl:output>
        </wsdl:operation>
        <wsdl:operation name="Execute_TrajService">
```

```
            <soap12:operation soapAction="urn:Execute_TrajService"
style="document"/>
            <wsdl:input>
                <soap12:body use="literal"/>
            </wsdl:input>
            <wsdl:output>
                <soap12:body use="literal"/>
            </wsdl:output>
        </wsdl:operation>
        <wsdl:operation name="GetCapabilities">
            <soap12:operation soapAction="urn:GetCapabilities"
style="document"/>
            <wsdl:input>
                <soap12:body use="literal"/>
            </wsdl:input>
            <wsdl:output>
                <soap12:body use="literal"/>
            </wsdl:output>
        </wsdl:operation>
    </wsdl:binding>
    <wsdl:service name="WpsJsdl">
        <wsdl:port name="WpsJsdlHttpSoap12Endpoint"
binding="stfc:WpsJsdlSoap12Binding">
            <soap12:address location="http://glue.badc.rl.ac.uk/wps-
jsdl/services/WpsJsdl/"/>
        </wsdl:port>
        <wsdl:port name="WpsJsdlHttpSoap11Endpoint"
binding="stfc:WpsJsdlSoap11Binding">
            <soap:address location="http://glue.badc.rl.ac.uk/wps-
jsdl/services/WpsJsdl/"/>
        </wsdl:port>
    </wsdl:service>
</wsdl:definitions>
```

B.2 Plume Service process

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:tns="http://new.webservice.namespace"
xmlns:wps="http://www.opengis.net/wps/1.0.0"
xmlns:plnk="http://schemas.xmlsoap.org/ws/2003/05/partner-link/"
targetNamespace="http://new.webservice.namespace">
    <wsdl:types>
        <xs:schema targetNamespace="http://new.webservice.namespace"
elementFormDefault="qualified"/>
        <xs:import namespace="http://www.opengis.net/wps/1.0.0"
schemaLocation="http://schemas.opengis.net/wps/1.0.0/wpsExecute_request.xsd"/>
        <xs:import namespace="http://www.opengis.net/wps/1.0.0"
schemaLocation="http://schemas.opengis.net/wps/1.0.0/wpsExecute_response.xsd"/>
    </wsdl:types>
    <wsdl:message name="ExecuteMessageRequest">
        <wsdl:part element="wps:Execute" name="InPart"/>
    </wsdl:message>
    <wsdl:message name="ExecuteMessageResponse">
        <wsdl:part element="wps:ExecuteResponse" name="OutPart"/>
    </wsdl:message>
    <wsdl:portType name="PlumeServicePortType">
        <wsdl:operation name="ExecuteOperation">
```

```
            <wsdl:input message="tns:ExecuteMessageRequest"/>
            <wsdl:output message="tns:ExecuteMessageResponse"/>
        </wsdl:operation>
    </wsdl:portType>
    <wsdl:binding name="PlumeServiceSoap11Binding"
type="tns:PlumeServicePortType">
        <soap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
        <wsdl:operation name="ExecuteOperation">
            <soap:operation soapAction="urn:#ExecuteOperation"/>
            <wsdl:input>
                <soap:body use="literal"/>
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal"/>
            </wsdl:output>
        </wsdl:operation>
    </wsdl:binding>
    <wsdl:binding name="PlumeServiceHttpPostBinding"
type="tns:PlumeServicePortType">
        <http:binding verb="POST"/>
        <wsdl:operation name="ExecuteOperation">
            <http:operation location="/WebProcessingService"/>
            <wsdl:input>
                <mime:mimeXml part="InPart"/>
            </wsdl:input>
            <wsdl:output>
                <mime:mimeXml part="OutPart"/>
            </wsdl:output>
        </wsdl:operation>
    </wsdl:binding>
    <wsdl:service name="PlumeService">
        <wsdl:port name="PlumeServiceSoap11Endpoint"
binding="tns:PlumeServiceSoap11Binding">
            <soap:address location="http://giv-bandog.uni-
muenster.de:8080/wps/services/SoapWPS"/>
        </wsdl:port>
        <wsdl:port name="PlumeServiceHttpPostEndpoint"
binding="tns:PlumeServiceHttpPostBinding">
            <http:address location="http://giv-bandog.uni-
muenster.de:8080/wps/WebProcessingService"/>
        </wsdl:port>
    </wsdl:service>
    <plnk:partnerLinkType name="partnerLinkType">
        <plnk:role name="partnerLinkTypeRole">
            <plnk:portType name="tns:PlumeServicePortType"/>
        </plnk:role>
    </plnk:partnerLinkType>
</wsdl:definitions>
```

B.3 Rainfall Interpolation process

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:tns="http://new.webservice.namespace"
xmlns:wps="http://www.opengis.net/wps/1.0.0"
xmlns:plnk="http://schemas.xmlsoap.org/ws/2003/05/partner-link/"
targetNamespace="http://new.webservice.namespace">
    <wsdl:types>
```

```
        <xs:schema targetNamespace="http://new.webservice.namespace"
elementFormDefault="qualified"/>
        <xs:import namespace="http://www.opengis.net/wps/1.0.0"
schemaLocation="http://schemas.opengis.net/wps/1.0.0/wpsExecute_request.xsd"/>
        <xs:import namespace="http://www.opengis.net/wps/1.0.0"
schemaLocation="http://schemas.opengis.net/wps/1.0.0/wpsExecute_response.xsd"/>
    </wsdl:types>
    <wsdl:message name="ExecuteMessageRequest">
        <wsdl:part element="wps:Execute" name="InPart"/>
    </wsdl:message>
    <wsdl:message name="ExecuteMessageResponse">
        <wsdl:part element="wps:ExecuteResponse" name="OutPart"/>
    </wsdl:message>
    <wsdl:portType name="PlumeServicePortType">
        <wsdl:operation name="ExecuteOperation">
            <wsdl:input message="tns:ExecuteMessageRequest"/>
            <wsdl:output message="tns:ExecuteMessageResponse"/>
        </wsdl:operation>
    </wsdl:portType>
    <wsdl:binding name="PlumeServiceSoap11Binding"
type="tns:PlumeServicePortType">
        <soap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
        <wsdl:operation name="ExecuteOperation">
            <soap:operation soapAction="urn:#ExecuteOperation"/>
            <wsdl:input>
                <soap:body use="literal"/>
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal"/>
            </wsdl:output>
        </wsdl:operation>
    </wsdl:binding>
    <wsdl:binding name="PlumeServiceHttpPostBinding"
type="tns:PlumeServicePortType">
        <http:binding verb="POST"/>
        <wsdl:operation name="ExecuteOperation">
            <http:operation location="/WebProcessingService"/>
            <wsdl:input>
                <mime:mimeXml part="InPart"/>
            </wsdl:input>
            <wsdl:output>
                <mime:mimeXml part="OutPart"/>
            </wsdl:output>
        </wsdl:operation>
    </wsdl:binding>
    <wsdl:service name="PlumeService">
        <wsdl:port name="PlumeServiceSoap11Endpoint"
binding="tns:PlumeServiceSoap11Binding">
            <soap:address location="http://giv-bandog.uni-
muenster.de:8080/wps/services/SoapWPS"/>
        </wsdl:port>
        <wsdl:port name="PlumeServiceHttpPostEndpoint"
binding="tns:PlumeServiceHttpPostBinding">
            <http:address location="http://giv-bandog.uni-
muenster.de:8080/wps/WebProcessingService"/>
        </wsdl:port>
    </wsdl:service>
    <plnk:partnerLinkType name="partnerLinkType">
        <plnk:role name="partnerLinkTypeRole">
            <plnk:portType name="tns:PlumeServicePortType"/>
        </plnk:role>
    </plnk:partnerLinkType>
</wsdl:definitions>
```

B.4 Geophone Analysis process

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:tns="http://new.webservice.namespace"
xmlns:wps="http://www.opengis.net/wps/1.0.0"
xmlns:plnk="http://schemas.xmlsoap.org/ws/2003/05/partner-link/"
targetNamespace="http://new.webservice.namespace">
    <wsdl:types>
        <xs:schema targetNamespace="http://new.webservice.namespace"
elementFormDefault="qualified"/>
        <xs:import namespace="http://www.opengis.net/wps/1.0.0"
schemaLocation="http://schemas.opengis.net/wps/1.0.0/wpsExecute_request.xsd"/>
        <xs:import namespace="http://www.opengis.net/wps/1.0.0"
schemaLocation="http://schemas.opengis.net/wps/1.0.0/wpsExecute_response.xsd"/>
    </wsdl:types>
    <wsdl:message name="ExecuteMessageRequest">
        <wsdl:part element="wps:Execute" name="InPart"/>
    </wsdl:message>
    <wsdl:message name="ExecuteMessageResponse">
        <wsdl:part element="wps:ExecuteResponse" name="OutPart"/>
    </wsdl:message>
    <wsdl:portType name="PlumeServicePortType">
        <wsdl:operation name="ExecuteOperation">
            <wsdl:input message="tns:ExecuteMessageRequest"/>
            <wsdl:output message="tns:ExecuteMessageResponse"/>
        </wsdl:operation>
    </wsdl:portType>
    <wsdl:binding name="PlumeServiceSoap11Binding"
type="tns:PlumeServicePortType">
        <soap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
        <wsdl:operation name="ExecuteOperation">
            <soap:operation soapAction="urn:#ExecuteOperation"/>
            <wsdl:input>
                <soap:body use="literal"/>
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal"/>
            </wsdl:output>
        </wsdl:operation>
    </wsdl:binding>
    <wsdl:binding name="PlumeServiceHttpPostBinding"
type="tns:PlumeServicePortType">
        <http:binding verb="POST"/>
        <wsdl:operation name="ExecuteOperation">
            <http:operation location="/WebProcessingService"/>
            <wsdl:input>
                <mime:mimeXml part="InPart"/>
            </wsdl:input>
            <wsdl:output>
                <mime:mimeXml part="OutPart"/>
            </wsdl:output>
        </wsdl:operation>
    </wsdl:binding>
    <wsdl:service name="PlumeService">
        <wsdl:port name="PlumeServiceSoap11Endpoint"
binding="tns:PlumeServiceSoap11Binding">
```

```
            <soap:address location="http://giv-bandog.uni-
muenster.de:8080/wps/services/SoapWPS"/>
        </wsdl:port>
        <wsdl:port name="PlumeServiceHttpPostEndpoint"
binding="tns:PlumeServiceHttpPostBinding">
            <http:address location="http://giv-bandog.uni-
muenster.de:8080/wps/WebProcessingService"/>
        </wsdl:port>
    </wsdl:service>
    <plnk:partnerLinkType name="partnerLinkType">
        <plnk:role name="partnerLinkTypeRole">
            <plnk:portType name="tns:PlumeServicePortType"/>
        </plnk:role>
    </plnk:partnerLinkType>
</wsdl:definitions>
```

B.5 `Gaja3d.wsdl` (Gaja3dService)

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="Gaja3dService" xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://org.kalypso.gaja3d.service"
xmlns:tns="http://org.kalypso.gaja3d.service" xmlns:wsrp="http://docs.oasis-
open.org/wsrf/2004/06/wsrf-WS-ResourceProperties-1.2-draft-01.xsd"
xmlns:wsrpw="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-
ResourceProperties-1.2-draft-01.wsdl" xmlns:wsrlw="http://docs.oasis-
open.org/wsrf/2004/06/wsrf-WS-ResourceLifetime-1.2-draft-01.wsdl"
xmlns:wsdlpp="http://www.globus.org/namespaces/2004/10/WSDLPreprocessor"
xmlns:wsntw="http://docs.oasis-open.org/wsn/2004/06/wsn-WS-BaseNotification-
1.2-draft-01.wsdl" xmlns:wsbf="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-
BaseFaults-1.2-draft-01.xsd" xmlns:wsbfw="http://docs.oasis-
open.org/wsrf/2004/06/wsrf-WS-BaseFaults-1.2-draft-01.wsdl"
xmlns:wps="http://www.opengis.net/wps/1.0.0"
xmlns:ows="http://www.opengis.net/ows/1.1">
    <!--
        ===================== I M P O R T S ======================
        ==========================================================
    -->
    <import namespace="http://docs.oasis-open.org/wsn/2004/06/wsn-WS-
BaseNotification-1.2-draft-01.wsdl" location="../wsrf/notification/WS-
BaseN.wsdl"/>
    <import namespace="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-
ResourceProperties-1.2-draft-01.wsdl" location="../wsrf/properties/WS-
ResourceProperties.wsdl"/>
    <import namespace="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-
ResourceLifetime-1.2-draft-01.wsdl" location="../wsrf/lifetime/WS-
ResourceLifetime.wsdl"/>
    <import namespace="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-
BaseFaults-1.2-draft-01.wsdl" location="../wsrf/faults/WS-BaseFaults.wsdl"/>
    <import namespace="http://org.kalypso.gaja3d.service"
location="wpsExecute_gaja3d_1.0.xsd"/>
    <!--
        ======================= T Y P E S ==========================
        ============================================================
    -->
    <types>
```

```
        <xsd:schema targetNamespace="http://org.kalypso.gaja3d.service"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/03/addressing"
xmlns:wsrl="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceLifetime-
1.2-draft-01.xsd" xmlns:wsbf="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-
BaseFaults-1.2-draft-01.xsd" xmlns:tns="http://org.kalypso.gaja3d.service"
xmlns:ows="http://www.opengis.net/ows/1.1"
xmlns:wps="http://www.opengis.net/wps/1.0.0"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
            <xsd:import
namespace="http://schemas.xmlsoap.org/ws/2004/03/addressing"
location="../ws/addressing/WS-Addressing.xsd"/>
            <xsd:include schemaLocation="wpsExecute_gaja3d_1.0.xsd"/>
            <xsd:element name="Gaja3dResourceProperties">
                <xsd:complexType>
                    <xsd:sequence>
                        <xsd:element ref="tns:Boundary" minOccurs="0"
maxOccurs="unbounded"/>
                        <xsd:element ref="tns:Breaklines" minOccurs="0"
maxOccurs="unbounded"/>
                        <xsd:element ref="tns:DemGrid" minOccurs="0"
maxOccurs="unbounded"/>
                        <xsd:element ref="tns:DemPoints" minOccurs="0"
maxOccurs="1"/>
                        <xsd:element ref="tns:DistanceTolerance" minOccurs="0"
maxOccurs="1"/>
                        <xsd:element ref="tns:EdgeFilter" minOccurs="0"
maxOccurs="1"/>
                        <xsd:element ref="tns:FeatureDetector" minOccurs="0"
maxOccurs="1"/>
                        <xsd:element ref="tns:GridX" minOccurs="0"
maxOccurs="1"/>
                        <xsd:element ref="tns:GridY" minOccurs="0"
maxOccurs="1"/>
                        <xsd:element ref="tns:MaxArea" minOccurs="0"
maxOccurs="1"/>
                        <xsd:element ref="tns:MinAngle" minOccurs="0"
maxOccurs="1"/>
                        <xsd:element ref="tns:ModelTin" minOccurs="0"
maxOccurs="1"/>
                        <xsd:element ref="tns:SmoothFilter" minOccurs="0"
maxOccurs="1"/>
                        <xsd:element name="GramEndpointReference"
type="wsa:EndpointReferenceType" minOccurs="0" maxOccurs="1"/>
                    </xsd:sequence>
                </xsd:complexType>
            </xsd:element>
        </xsd:schema>
    </types>
    <!--
        ====================== M E S S A G E S =====================
        ============================================================
    -->
    <!-- Generic WPS messages -->
    <message name="GetCapabilitiesRequest">
        <part name="parameters" element="ows:GetCapabilities"/>
    </message>
    <message name="GetCapabilitiesResponse">
        <part name="Body" element="wps:Capabilities"/>
    </message>
    <message name="DescribeProcessRequest">
        <part name="parameters" element="wps:DescribeProcess"/>
    </message>
    <message name="DescribeProcessResponse">
```

```
        <part name="Body" element="wps:ProcessDescriptions"/>
    </message>
    <message name="ExecuteRequest">
        <part name="parameters" element="wps:Execute"/>
    </message>
    <message name="ExecuteResponse">
        <part name="Body" element="wps:ExecuteResponse"/>
    </message>
    <!-- Special Gaja3d messages -->
    <message name="Execute_createGridRequest">
        <part name="parameters" element="tns:CreateGridParameters"/>
    </message>
    <message name="Execute_createGridResponse">
        <part name="Body" type="tns:CreateGridResponseType"/>
    </message>
    <message name="Execute_detectBreaklinesRequest">
        <part name="parameters" element="tns:DetectBreaklinesParameters"/>
    </message>
    <message name="Execute_detectBreaklinesResponse">
        <part name="Body" type="tns:DetectBreaklinesResponseType"/>
    </message>
    <message name="Execute_createTinRequest">
        <part name="parameters" element="tns:CreateTinParameters"/>
    </message>
    <message name="Execute_createTinResponse">
        <part name="Body" type="tns:CreateTinResponseType"/>
    </message>
    <!--
        ==================== P O R T T Y P E =======================
        ============================================================
    -->
    <portType name="Gaja3dPortType" wsdlpp:extends="wsrpw:GetResourceProperty
                    wsrpw:GetMultipleResourceProperties
                    wsrpw:SetResourceProperties
                    wsrpw:QueryResourceProperties
                    wsrlw:ImmediateResourceTermination
                    wsrlw:ScheduledResourceTermination
                    wsntw:NotificationProducer"
wsrp:ResourceProperties="tns:Gaja3dResourceProperties">
        <!-- Generic WPS operations -->
        <operation name="GetCapabilities">
            <documentation>This standard web method is automatically
                generated to give a full description of this operation. The
response
                will be an embedded WPS XML GetCapabilities Response. Please
consult
                the OGC Web Site for the WPS 1.0.0 standard.</documentation>
            <input name="GetCapabilitiesRequest"
message="tns:GetCapabilitiesRequest"/>
            <output name="GetCapabilitiesResponse"
message="tns:GetCapabilitiesResponse"/>
        </operation>
        <operation name="DescribeProcess">
            <documentation>This standard web method is automatically
                generated to give a full description of all processes provided by
                this operation. The response will be an embedded WPS XML
                DescribeProcess Response. Please consult the OGC Web Site for the
                WPS 0.4.0 standard.</documentation>
            <input name="DescribeProcessRequest"
message="tns:DescribeProcessRequest"/>
            <output name="DescribeProcessResponse"
message="tns:DescribeProcessResponse"/>
        </operation>
        <operation name="Execute">
```

```
            <documentation>This is the generic Execute operation
            </documentation>
            <input name="ExecuteRequest" message="tns:ExecuteRequest"/>
            <output name="ExecuteResponse" message="tns:ExecuteResponse"/>
        </operation>
        <!-- Specific Gaja3d operations -->
        <operation name="Execute_createGrid">
            <input name="Execute_createGridRequest"
message="tns:Execute_createGridRequest"/>
            <output name="Execute_createGridResponse"
message="tns:Execute_createGridResponse"/>
        </operation>
        <operation name="Execute_detectBreaklines">
            <input name="Execute_detectBreaklinesRequest"
message="tns:Execute_detectBreaklinesRequest"/>
            <output name="Execute_detectBreaklinesResponse"
message="tns:Execute_detectBreaklinesResponse"/>
        </operation>
        <operation name="Execute_createTin">
            <input name="Execute_createTinRequest"
message="tns:Execute_createTinRequest"/>
            <output name="Execute_createTinResponse"
message="tns:Execute_createTinResponse"/>
        </operation>
    </portType>
</definitions>
```

B.6 `wpsExecute_gaja3d_1.0.xsd` (Gaja3dService)

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:tns="http://org.kalypso.gaja3d.service"
xmlns:gml="http://www.opengis.net/gml"
xmlns:wps="http://www.opengis.net/wps/1.0.0"
xmlns:ows="http://www.opengis.net/ows/1.1"
xmlns:xlink="http://www.w3.org/1999/xlink"
targetNamespace="http://org.kalypso.gaja3d.service"
elementFormDefault="qualified">
    <!--
        ===================== I M P O R T S =====================
        =========================================================
    -->
    <import namespace="http://www.opengis.net/wps/1.0.0"
schemaLocation="wps/1.0.0/wpsAll.xsd"/>
    <import namespace="http://www.opengis.net/ows/1.1"
schemaLocation="ows/1.1.0/owsAll.xsd"/>
    <import namespace="http://www.w3.org/1999/xlink"
schemaLocation="xlink/1.0.0/xlinks.xsd"/>
    <!--
        ============ R E S O U R C E P R O P E R T I E S ===========
        ==========================================================
    -->
    <element name="_Gaja3DResourcePropertyLink"
type="tns:Gaja3DResourcePropertyLinkType" abstract="true"/>
    <complexType name="Gaja3DResourcePropertyLinkType">
        <complexContent>
            <extension base="tns:Gaja3DResourcePropertyType">
                <sequence>
                    <element name="ref" type="anyURI"/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <complexType name="Gaja3DResourcePropertyType">
```

```
        <sequence>
            <element name="identifier" type="ows:CodeType"/>
        </sequence>
    </complexType>
    <element name="_Gaja3DResourceProperty"
type="tns:Gaja3DResourcePropertyType" abstract="true"/>
    <!-- property types -->
    <element name="Boundary"
substitutionGroup="tns:_Gaja3DResourcePropertyLink">
        <complexType>
            <complexContent>
                <extension base="tns:Gaja3DResourcePropertyLinkType"/>
            </complexContent>
        </complexType>
    </element>
    <element name="DemPoints"
substitutionGroup="tns:_Gaja3DResourcePropertyLink">
        <complexType>
            <complexContent>
                <extension base="tns:Gaja3DResourcePropertyLinkType"/>
            </complexContent>
        </complexType>
    </element>
    <element name="DemGrid" substitutionGroup="tns:_Gaja3DResourcePropertyLink">
        <complexType>
            <complexContent>
                <extension base="tns:Gaja3DResourcePropertyLinkType"/>
            </complexContent>
        </complexType>
    </element>
    <element name="Breaklines"
substitutionGroup="tns:_Gaja3DResourcePropertyLink">
        <complexType>
            <complexContent>
                <extension base="tns:Gaja3DResourcePropertyLinkType"/>
            </complexContent>
        </complexType>
    </element>
    <element name="ModelTin"
substitutionGroup="tns:_Gaja3DResourcePropertyLink">
        <complexType>
            <complexContent>
                <extension base="tns:Gaja3DResourcePropertyLinkType"/>
            </complexContent>
        </complexType>
    </element>
    <!-- parameter types -->
    <element name="GridX" substitutionGroup="tns:_Gaja3DResourceProperty">
        <complexType>
            <complexContent>
                <extension base="tns:Gaja3DResourcePropertyType">
                    <sequence>
                        <element name="dx" type="double"/>
                    </sequence>
                </extension>
            </complexContent>
        </complexType>
    </element>
    <element name="GridY" substitutionGroup="tns:_Gaja3DResourceProperty">
        <complexType>
            <complexContent>
                <extension base="tns:Gaja3DResourcePropertyType">
                    <sequence>
                        <element name="dy" type="double"/>
                    </sequence>
                </extension>
            </complexContent>
        </complexType>
    </element>
    <element name="EdgeFilter" substitutionGroup="tns:_Gaja3DResourceProperty">
        <complexType>
            <complexContent>
                <extension base="tns:Gaja3DResourcePropertyType">
                    <sequence>
                        <element name="method" default="ood">
                            <simpleType>
                                <restriction base="string">
                                    <enumeration value="ood"/>
                                    <enumeration value="sobel"/>
                                    <enumeration value="prewitt"/>
                                </restriction>
                            </simpleType>
                        </element>
                    </sequence>
                </extension>
            </complexContent>
        </complexType>
    </element>
    <element name="SmoothFilter"
substitutionGroup="tns:_Gaja3DResourceProperty">
        <complexType>
            <complexContent>
                <extension base="tns:Gaja3DResourcePropertyType">
                    <sequence>
                        <element name="method" default="none">
                            <simpleType>
                                <restriction base="string">
                                    <enumeration value="bilateral"/>
                                    <enumeration value="gauss"/>
                                    <enumeration value="none"/>
                                </restriction>
                            </simpleType>
                        </element>
                        <element name="smooth" type="int" default="9"/>
                    </sequence>
                </extension>
            </complexContent>
        </complexType>
    </element>
    <element name="FeatureDetector"
substitutionGroup="tns:_Gaja3DResourceProperty">
        <complexType>
            <complexContent>
                <extension base="tns:Gaja3DResourcePropertyType">
                    <sequence>
                        <element name="method" default="simple">
                            <simpleType>
                                <restriction base="string">
                                    <enumeration value="simple"/>
                                    <enumeration value="canny"/>
                                </restriction>
                            </simpleType>
                        </element>
                        <element name="lowThresh" type="double"/>
                        <element name="highThresh" type="double"/>
                    </sequence>
                </extension>
            </complexContent>
        </complexType>
    </element>
    <element name="DistanceTolerance"
substitutionGroup="tns:_Gaja3DResourceProperty">
        <complexType>
            <complexContent>
                <extension base="tns:Gaja3DResourcePropertyType">
                    <sequence>
                        <element name="tolerance" type="double"/>
                    </sequence>
                </extension>
            </complexContent>
        </complexType>
    </element>
    <element name="MinAngle" substitutionGroup="tns:_Gaja3DResourceProperty">
        <complexType>
            <complexContent>
                <extension base="tns:Gaja3DResourcePropertyType">
                    <sequence>
                        <element name="angle" type="double"/>
                    </sequence>
                </extension>
            </complexContent>
        </complexType>
    </element>
    <element name="MaxArea" substitutionGroup="tns:_Gaja3DResourceProperty">
        <complexType>
            <complexContent>
                <extension base="tns:Gaja3DResourcePropertyType">
                    <sequence>
                        <element name="area" type="double"/>
                    </sequence>
                </extension>
            </complexContent>
        </complexType>
    </element>
    <element name="CreateGridParameters" type="tns:CreateGridParametersType"/>
    <element name="DetectBreaklinesParameters"
type="tns:DetectBreaklinesParametersType"/>
    <element name="CreateTinParameters" type="tns:CreateTinParametersType"/>
    <!--
        =================== P A R A M E T E R S ===================
        ===========================================================
    -->
    <complexType name="CreateGridParametersType">
        <sequence>
            <element ref="tns:Boundary" minOccurs="0" maxOccurs="unbounded"/>
            <element ref="tns:DemPoints" minOccurs="0"/>
            <element ref="tns:GridX" minOccurs="0"/>
            <element ref="tns:GridY" minOccurs="0"/>
        </sequence>
    </complexType>
    <complexType name="CreateGridResponseType">
        <sequence>
            <element ref="tns:DemGrid" maxOccurs="unbounded"/>
        </sequence>
    </complexType>
    <complexType name="DetectBreaklinesParametersType">
        <sequence>
            <element ref="tns:Boundary" minOccurs="0" maxOccurs="unbounded"/>
            <element ref="tns:DemGrid" minOccurs="0" maxOccurs="unbounded"/>
            <element ref="tns:EdgeFilter" minOccurs="0"/>
            <element ref="tns:SmoothFilter" minOccurs="0"/>
            <element ref="tns:FeatureDetector" minOccurs="0"/>
            <element ref="tns:DistanceTolerance" minOccurs="0"/>
        </sequence>
    </complexType>
    <complexType name="DetectBreaklinesResponseType">
        <sequence>
            <element ref="tns:Breaklines" maxOccurs="unbounded"/>
        </sequence>
    </complexType>
    <complexType name="CreateTinParametersType">
        <sequence>
            <element ref="tns:Boundary" minOccurs="0" maxOccurs="unbounded"/>
            <element ref="tns:Breaklines" minOccurs="0" maxOccurs="unbounded"/>
            <element ref="tns:DemGrid" minOccurs="0" maxOccurs="unbounded"/>
            <element ref="tns:MinAngle" minOccurs="0"/>
            <element ref="tns:MaxArea" minOccurs="0"/>
        </sequence>
    </complexType>
    <complexType name="CreateTinResponseType">
        <sequence>
            <element ref="tns:ModelTin"/>
        </sequence>
    </complexType>
</schema>
```

B.7 `Gaja3d_flattened.wsdl` (Gaja3dService)

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="Gaja3dService"
targetNamespace="http://org.kalypso.gaja3d.service"
xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:wsrpw="http://docs.oasis-
open.org/wsrf/2004/06/wsrf-WS-ResourceProperties-1.2-draft-01.wsdl"
xmlns:wsbf="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-BaseFaults-1.2-
draft-01.xsd" xmlns:wsntw="http://docs.oasis-open.org/wsn/2004/06/wsn-WS-
BaseNotification-1.2-draft-01.wsdl" xmlns:wsbfw="http://docs.oasis-
open.org/wsrf/2004/06/wsrf-WS-BaseFaults-1.2-draft-01.wsdl"
xmlns:wsdlpp="http://www.globus.org/namespaces/2004/10/WSDLPreprocessor"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/03/addressing"
xmlns:tns="http://org.kalypso.gaja3d.service"
xmlns:wps="http://www.opengis.net/wps/1.0.0"
xmlns:ows="http://www.opengis.net/ows/1.1" xmlns:wsrlw="http://docs.oasis-
open.org/wsrf/2004/06/wsrf-WS-ResourceLifetime-1.2-draft-01.wsdl"
xmlns:wsrp="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceProperties-
1.2-draft-01.xsd" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <import namespace="http://docs.oasis-open.org/wsn/2004/06/wsn-WS-
BaseNotification-1.2-draft-01.wsdl" location="../wsrf/notification/WS-
BaseN.wsdl"/>
    <import namespace="http://org.kalypso.gaja3d.service"
location="wpsExecute_gaja3d_1.0.xsd"/>
    <import namespace="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-
ResourceLifetime-1.2-draft-01.wsdl" location="../wsrf/lifetime/WS-
ResourceLifetime.wsdl"/>
    <import namespace="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-
BaseFaults-1.2-draft-01.wsdl" location="../wsrf/faults/WS-BaseFaults.wsdl"/>
    <import namespace="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-
ResourceProperties-1.2-draft-01.wsdl" location="../wsrf/properties/WS-
ResourceProperties.wsdl"/>
    <types>
```

```
       <xsd:schema targetNamespace="http://org.kalypso.gaja3d.service"
xmlns:ows="http://www.opengis.net/ows/1.1" xmlns:rpns0="http://docs.oasis-
open.org/wsn/2004/06/wsn-WS-BaseNotification-1.2-draft-01.xsd"
xmlns:tns="http://org.kalypso.gaja3d.service"
xmlns:wps="http://www.opengis.net/wps/1.0.0"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/03/addressing"
xmlns:wsbf="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-BaseFaults-1.2-
draft-01.xsd" xmlns:wsrl="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-
ResourceLifetime-1.2-draft-01.xsd" xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
           <xsd:import
namespace="http://schemas.xmlsoap.org/ws/2004/03/addressing"
schemaLocation="../ws/addressing/WS-Addressing.xsd"/>
           <xsd:import namespace="http://docs.oasis-open.org/wsrf/2004/06/wsrf-
WS-ResourceLifetime-1.2-draft-01.xsd" schemaLocation="../wsrf/lifetime/WS-
ResourceLifetime.xsd"/>
           <xsd:import namespace="http://docs.oasis-open.org/wsn/2004/06/wsn-
WS-BaseNotification-1.2-draft-01.xsd" schemaLocation="../wsrf/notification/WS-
BaseN.xsd"/>
           <xsd:import location="../ws/addressing/WS-Addressing.xsd"
namespace="http://schemas.xmlsoap.org/ws/2004/03/addressing"/>
           <xsd:include schemaLocation="wpsExecute_gaja3d_1.0.xsd"/>
           <xsd:element name="Gaja3dResourceProperties">
               <xsd:complexType>
                   <xsd:sequence>
                       <xsd:element maxOccurs="unbounded" minOccurs="0"
ref="tns:Boundary"/>
                       <xsd:element maxOccurs="unbounded" minOccurs="0"
ref="tns:Breaklines"/>
                       <xsd:element maxOccurs="unbounded" minOccurs="0"
ref="tns:DemGrid"/>
                       <xsd:element maxOccurs="1" minOccurs="0"
ref="tns:DemPoints"/>
                       <xsd:element maxOccurs="1" minOccurs="0"
ref="tns:DistanceTolerance"/>
                       <xsd:element maxOccurs="1" minOccurs="0"
ref="tns:EdgeFilter"/>
                       <xsd:element maxOccurs="1" minOccurs="0"
ref="tns:FeatureDetector"/>
                       <xsd:element maxOccurs="1" minOccurs="0"
ref="tns:GridX"/>
                       <xsd:element maxOccurs="1" minOccurs="0"
ref="tns:GridY"/>
                       <xsd:element maxOccurs="1" minOccurs="0"
ref="tns:MaxArea"/>
                       <xsd:element maxOccurs="1" minOccurs="0"
ref="tns:MinAngle"/>
                       <xsd:element maxOccurs="1" minOccurs="0"
ref="tns:ModelTin"/>
                       <xsd:element maxOccurs="1" minOccurs="0"
ref="tns:SmoothFilter"/>
                       <xsd:element maxOccurs="1" minOccurs="0"
name="GramEndpointReference" type="wsa:EndpointReferenceType"/>
                       <xsd:element maxOccurs="1" minOccurs="1"
ref="rpns0:FixedTopicSet"/>
                       <xsd:element maxOccurs="unbounded" minOccurs="1"
ref="rpns0:Topic"/>
                       <xsd:element maxOccurs="1" minOccurs="1"
ref="wsrl:TerminationTime"/>
                       <xsd:element maxOccurs="unbounded" minOccurs="1"
ref="rpns0:TopicExpressionDialects"/>
                       <xsd:element maxOccurs="1" minOccurs="1"
ref="wsrl:CurrentTime"/>
                   </xsd:sequence>
```

```
                </xsd:complexType>
            </xsd:element>
        </xsd:schema>
    </types>
    <message name="Execute_detectBreaklinesRequest">
        <part name="parameters" element="tns:DetectBreaklinesParameters"/>
    </message>
    <message name="DescribeProcessResponse">
        <part name="Body" element="wps:ProcessDescriptions"/>
    </message>
    <message name="ExecuteResponse">
        <part name="Body" element="wps:ExecuteResponse"/>
    </message>
    <message name="Execute_createGridRequest">
        <part name="parameters" element="tns:CreateGridParameters"/>
    </message>
    <message name="DescribeProcessRequest">
        <part name="parameters" element="wps:DescribeProcess"/>
    </message>
    <message name="GetCapabilitiesResponse">
        <part name="Body" element="wps:Capabilities"/>
    </message>
    <message name="Execute_createTinResponse">
        <part name="Body" type="tns:CreateTinResponseType"/>
    </message>
    <message name="ExecuteRequest">
        <part name="parameters" element="wps:Execute"/>
    </message>
    <message name="Execute_createTinRequest">
        <part name="parameters" element="tns:CreateTinParameters"/>
    </message>
    <message name="Execute_createGridResponse">
        <part name="Body" type="tns:CreateGridResponseType"/>
    </message>
    <message name="Execute_detectBreaklinesResponse">
        <part name="Body" type="tns:DetectBreaklinesResponseType"/>
    </message>
    <message name="GetCapabilitiesRequest">
        <part name="parameters" element="ows:GetCapabilities"/>
    </message>
    <portType name="Gaja3dPortType"
wsrp:ResourceProperties="tns:Gaja3dResourceProperties">
        <operation name="GetCapabilities">
            <documentation>This standard web method is automatically
                generated to give a full description of this operation. The
response
                will be an embedded WPS XML GetCapabilities Response. Please
consult
                the OGC Web Site for the WPS 1.0.0 standard.</documentation>
            <input name="GetCapabilitiesRequest"
message="tns:GetCapabilitiesRequest"/>
            <output name="GetCapabilitiesResponse"
message="tns:GetCapabilitiesResponse"/>
        </operation>
        <operation name="DescribeProcess">
            <documentation>This standard web method is automatically
                generated to give a full description of all processes provided by
                this operation. The response will be an embedded WPS XML
                DescribeProcess Response. Please consult the OGC Web Site for the
                WPS 0.4.0 specification.</documentation>
            <input name="DescribeProcessRequest"
message="tns:DescribeProcessRequest"/>
            <output name="DescribeProcessResponse"
message="tns:DescribeProcessResponse"/>
```

```
        </operation>
        <operation name="Execute">
            <documentation>This is the generic Execute operation
            </documentation>
            <input name="ExecuteRequest" message="tns:ExecuteRequest"/>
            <output name="ExecuteResponse" message="tns:ExecuteResponse"/>
        </operation>
        <operation name="Execute_createGrid">
            <input name="Execute_createGridRequest"
message="tns:Execute_createGridRequest"/>
            <output name="Execute_createGridResponse"
message="tns:Execute_createGridResponse"/>
        </operation>
        <operation name="Execute_detectBreaklines">
            <input name="Execute_detectBreaklinesRequest"
message="tns:Execute_detectBreaklinesRequest"/>
            <output name="Execute_detectBreaklinesResponse"
message="tns:Execute_detectBreaklinesResponse"/>
        </operation>
        <operation name="Execute_createTin">
            <input name="Execute_createTinRequest"
message="tns:Execute_createTinRequest"/>
            <output name="Execute_createTinResponse"
message="tns:Execute_createTinResponse"/>
        </operation>
        <operation name="GetResourceProperty">
            <input name="GetResourcePropertyRequest"
message="wsrpw:GetResourcePropertyRequest" wsa:Action="http://docs.oasis-
open.org/wsrf/2004/06/wsrf-WS-ResourceProperties/GetResourceProperty"/>
            <output name="GetResourcePropertyResponse"
message="wsrpw:GetResourcePropertyResponse" wsa:Action="http://docs.oasis-
open.org/wsrf/2004/06/wsrf-WS-ResourceProperties/GetResourcePropertyResponse"/>
            <fault name="InvalidResourcePropertyQNameFault"
message="wsrpw:InvalidResourcePropertyQNameFault"/>
            <fault name="ResourceUnknownFault"
message="wsrpw:ResourceUnknownFault"/>
        </operation>
        <operation name="Subscribe">
            <input message="wsntw:SubscribeRequest"
wsa:Action="http://docs.oasis-open.org/wsn/2004/06/wsn-WS-
BaseNotification/Subscribe"/>
            <output message="wsntw:SubscribeResponse"
wsa:Action="http://docs.oasis-open.org/wsn/2004/06/wsn-WS-
BaseNotification/SubscribeResponse"/>
            <fault name="TopicNotSupportedFault"
message="wsntw:TopicNotSupportedFault"/>
            <fault name="SubscribeCreationFailedFault"
message="wsntw:SubscribeCreationFailedFault"/>
            <fault name="InvalidTopicExpressionFault"
message="wsntw:InvalidTopicExpressionFault"/>
            <fault name="ResourceUnknownFault"
message="wsntw:ResourceUnknownFault"/>
            <fault name="TopicPathDialectUnknownFault"
message="wsntw:TopicPathDialectUnknownFault"/>
        </operation>
        <operation name="GetCurrentMessage">
            <input message="wsntw:GetCurrentMessageRequest"
wsa:Action="http://docs.oasis-open.org/wsn/2004/06/wsn-WS-
BaseNotification/GetCurrentMessage"/>
            <output message="wsntw:GetCurrentMessageResponse"
wsa:Action="http://docs.oasis-open.org/wsn/2004/06/wsn-WS-
BaseNotification/GetCurrentMessageResponse"/>
            <fault name="TopicNotSupportedFault"
message="wsntw:TopicNotSupportedFault"/>
```

```
            <fault name="InvalidTopicExpressionFault"
message="wsntw:InvalidTopicExpressionFault"/>
            <fault name="ResourceUnknownFault"
message="wsntw:ResourceUnknownFault"/>
            <fault name="NoCurrentMessageOnTopicFault"
message="wsntw:NoCurrentMessageOnTopicFault"/>
        </operation>
        <operation name="Destroy">
            <input message="wsrlw:DestroyRequest" wsa:Action="http://docs.oasis-
open.org/wsrf/2004/06/wsrf-WS-ResourceLifetime/Destroy"/>
            <output message="wsrlw:DestroyResponse"
wsa:Action="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-
ResourceLifetime/DestroyResponse"/>
            <fault name="ResourceUnknownFault"
message="wsrlw:ResourceUnknownFault"/>
            <fault name="ResourceNotDestroyedFault"
message="wsrlw:ResourceNotDestroyedFault"/>
        </operation>
        <operation name="SetTerminationTime">
            <input message="wsrlw:SetTerminationTimeRequest"
wsa:Action="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-
ResourceLifetime/SetTerminationTime"/>
            <output message="wsrlw:SetTerminationTimeResponse"
wsa:Action="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-
ResourceLifetime/SetTerminationTimeResponse"/>
            <fault name="ResourceUnknownFault"
message="wsrlw:ResourceUnknownFault"/>
            <fault name="UnableToSetTerminationTimeFault"
message="wsrlw:UnableToSetTerminationTimeFault"/>
            <fault name="TerminationTimeChangeRejectedFault"
message="wsrlw:TerminationTimeChangeRejectedFault"/>
        </operation>
        <operation name="SetResourceProperties">
            <input name="SetResourcePropertiesRequest"
message="wsrpw:SetResourcePropertiesRequest" wsa:Action="http://docs.oasis-
open.org/wsrf/2004/06/wsrf-WS-ResourceProperties/SetResourceProperties"/>
            <output name="SetResourcePropertiesResponse"
message="wsrpw:SetResourcePropertiesResponse" wsa:Action="http://docs.oasis-
open.org/wsrf/2004/06/wsrf-WS-
ResourceProperties/SetResourcePropertiesResponse"/>
            <fault name="InvalidResourcePropertyQNameFault"
message="wsrpw:InvalidResourcePropertyQNameFault"/>
            <fault name="InvalidSetResourcePropertiesRequestContentFault"
message="wsrpw:InvalidSetResourcePropertiesRequestContentFault"/>
            <fault name="SetResourcePropertyRequestFailedFault"
message="wsrpw:SetResourcePropertyRequestFailedFault"/>
            <fault name="ResourceUnknownFault"
message="wsrpw:ResourceUnknownFault"/>
            <fault name="UnableToModifyResourcePropertyFault"
message="wsrpw:UnableToModifyResourcePropertyFault"/>
        </operation>
        <operation name="QueryResourceProperties">
            <input name="QueryResourcePropertiesRequest"
message="wsrpw:QueryResourcePropertiesRequest" wsa:Action="http://docs.oasis-
open.org/wsrf/2004/06/wsrf-WS-ResourceProperties/QueryResourceProperties"/>
            <output name="QueryResourcePropertiesResponse"
message="wsrpw:QueryResourcePropertiesResponse" wsa:Action="http://docs.oasis-
open.org/wsrf/2004/06/wsrf-WS-
ResourceProperties/QueryResourcePropertiesResponse"/>
            <fault name="UnknownQueryExpressionDialectFault"
message="wsrpw:UnknownQueryExpressionDialectFault"/>
            <fault name="QueryEvaluationErrorFault"
message="wsrpw:QueryEvaluationErrorFault"/>
```

```
            <fault name="InvalidQueryExpressionFault"
message="wsrpw:InvalidQueryExpressionFault"/>
            <fault name="InvalidResourcePropertyQNameFault"
message="wsrpw:InvalidResourcePropertyQNameFault"/>
            <fault name="ResourceUnknownFault"
message="wsrpw:ResourceUnknownFault"/>
        </operation>
        <operation name="GetMultipleResourceProperties">
            <input name="GetMultipleResourcePropertiesRequest"
message="wsrpw:GetMultipleResourcePropertiesRequest"
wsa:Action="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-
ResourceProperties/GetMultipleResourceProperties"/>
            <output name="GetMultipleResourcePropertiesResponse"
message="wsrpw:GetMultipleResourcePropertiesResponse"
wsa:Action="http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-
ResourceProperties/GetMultipleResourcePropertiesResponse"/>
            <fault name="InvalidResourcePropertyQNameFault"
message="wsrpw:InvalidResourcePropertyQNameFault"/>
            <fault name="ResourceUnknownFault"
message="wsrpw:ResourceUnknownFault"/>
        </operation>
    </portType>
</definitions>
```

B.8 `Gaja3d_bindings.wsdl` (Gaja3dService)

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="Gaja3dService"
targetNamespace="http://org.kalypso.gaja3d.service/bindings"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:porttype="http://org.kalypso.gaja3d.service"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">
    <wsdl:import namespace="http://org.kalypso.gaja3d.service"
location="Gaja3d_flattened.wsdl"/>
    <wsdl:binding name="Gaja3dPortTypeSOAPBinding"
type="porttype:Gaja3dPortType">
        <soap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
        <wsdl:operation name="GetCapabilities">
            <soap:operation
soapAction="http://org.kalypso.gaja3d.service/Gaja3dPortType/GetCapabilitiesReq
uest"/>
            <wsdl:input>
                <soap:body use="literal"/>
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal"/>
            </wsdl:output>
        </wsdl:operation>
        <wsdl:operation name="DescribeProcess">
            <soap:operation
soapAction="http://org.kalypso.gaja3d.service/Gaja3dPortType/DescribeProcessReq
uest"/>
            <wsdl:input>
                <soap:body use="literal"/>
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal"/>
            </wsdl:output>
        </wsdl:operation>
        <wsdl:operation name="Execute">
            <soap:operation
soapAction="http://org.kalypso.gaja3d.service/Gaja3dPortType/ExecuteRequest"/>
```

```
            <wsdl:input>
                <soap:body use="literal"/>
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal"/>
            </wsdl:output>
        </wsdl:operation>
        <wsdl:operation name="Execute_createGrid">
            <soap:operation 
soapAction="http://org.kalypso.gaja3d.service/Gaja3dPortType/Execute_createGrid
Request"/>
            <wsdl:input>
                <soap:body use="literal"/>
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal"/>
            </wsdl:output>
        </wsdl:operation>
        <wsdl:operation name="Execute_detectBreaklines">
            <soap:operation 
soapAction="http://org.kalypso.gaja3d.service/Gaja3dPortType/Execute_detectBrea
klinesRequest"/>
            <wsdl:input>
                <soap:body use="literal"/>
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal"/>
            </wsdl:output>
        </wsdl:operation>
        <wsdl:operation name="Execute_createTin">
            <soap:operation 
soapAction="http://org.kalypso.gaja3d.service/Gaja3dPortType/Execute_createTinR
equest"/>
            <wsdl:input>
                <soap:body use="literal"/>
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal"/>
            </wsdl:output>
        </wsdl:operation>
        <wsdl:operation name="GetResourceProperty">
            <soap:operation soapAction="http://docs.oasis-
open.org/wsrf/2004/06/wsrf-WS-ResourceProperties/GetResourceProperty"/>
            <wsdl:input>
                <soap:body use="literal"/>
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal"/>
            </wsdl:output>
            <wsdl:fault name="InvalidResourcePropertyQNameFault">
                <soap:fault name="InvalidResourcePropertyQNameFault" 
use="literal"/>
            </wsdl:fault>
            <wsdl:fault name="ResourceUnknownFault">
                <soap:fault name="ResourceUnknownFault" use="literal"/>
            </wsdl:fault>
        </wsdl:operation>
        <wsdl:operation name="Subscribe">
            <soap:operation soapAction="http://docs.oasis-
open.org/wsn/2004/06/wsn-WS-BaseNotification/Subscribe"/>
            <wsdl:input>
                <soap:body use="literal"/>
            </wsdl:input>
            <wsdl:output>
```

```
                <soap:body use="literal"/>
            </wsdl:output>
            <wsdl:fault name="TopicNotSupportedFault">
                <soap:fault name="TopicNotSupportedFault" use="literal"/>
            </wsdl:fault>
            <wsdl:fault name="SubscribeCreationFailedFault">
                <soap:fault name="SubscribeCreationFailedFault" use="literal"/>
            </wsdl:fault>
            <wsdl:fault name="InvalidTopicExpressionFault">
                <soap:fault name="InvalidTopicExpressionFault" use="literal"/>
            </wsdl:fault>
            <wsdl:fault name="ResourceUnknownFault">
                <soap:fault name="ResourceUnknownFault" use="literal"/>
            </wsdl:fault>
            <wsdl:fault name="TopicPathDialectUnknownFault">
                <soap:fault name="TopicPathDialectUnknownFault" use="literal"/>
            </wsdl:fault>
        </wsdl:operation>
        <wsdl:operation name="GetCurrentMessage">
            <soap:operation soapAction="http://docs.oasis-
open.org/wsn/2004/06/wsn-WS-BaseNotification/GetCurrentMessage"/>
            <wsdl:input>
                <soap:body use="literal"/>
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal"/>
            </wsdl:output>
            <wsdl:fault name="TopicNotSupportedFault">
                <soap:fault name="TopicNotSupportedFault" use="literal"/>
            </wsdl:fault>
            <wsdl:fault name="InvalidTopicExpressionFault">
                <soap:fault name="InvalidTopicExpressionFault" use="literal"/>
            </wsdl:fault>
            <wsdl:fault name="ResourceUnknownFault">
                <soap:fault name="ResourceUnknownFault" use="literal"/>
            </wsdl:fault>
            <wsdl:fault name="NoCurrentMessageOnTopicFault">
                <soap:fault name="NoCurrentMessageOnTopicFault" use="literal"/>
            </wsdl:fault>
        </wsdl:operation>
        <wsdl:operation name="Destroy">
            <soap:operation soapAction="http://docs.oasis-
open.org/wsrf/2004/06/wsrf-WS-ResourceLifetime/Destroy"/>
            <wsdl:input>
                <soap:body use="literal"/>
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal"/>
            </wsdl:output>
            <wsdl:fault name="ResourceUnknownFault">
                <soap:fault name="ResourceUnknownFault" use="literal"/>
            </wsdl:fault>
            <wsdl:fault name="ResourceNotDestroyedFault">
                <soap:fault name="ResourceNotDestroyedFault" use="literal"/>
            </wsdl:fault>
        </wsdl:operation>
        <wsdl:operation name="SetTerminationTime">
            <soap:operation soapAction="http://docs.oasis-
open.org/wsrf/2004/06/wsrf-WS-ResourceLifetime/SetTerminationTime"/>
            <wsdl:input>
                <soap:body use="literal"/>
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal"/>
```

```
           </wsdl:output>
           <wsdl:fault name="ResourceUnknownFault">
               <soap:fault name="ResourceUnknownFault" use="literal"/>
           </wsdl:fault>
           <wsdl:fault name="UnableToSetTerminationTimeFault">
               <soap:fault name="UnableToSetTerminationTimeFault"
use="literal"/>
           </wsdl:fault>
           <wsdl:fault name="TerminationTimeChangeRejectedFault">
               <soap:fault name="TerminationTimeChangeRejectedFault"
use="literal"/>
           </wsdl:fault>
       </wsdl:operation>
       <wsdl:operation name="SetResourceProperties">
           <soap:operation soapAction="http://docs.oasis-
open.org/wsrf/2004/06/wsrf-WS-ResourceProperties/SetResourceProperties"/>
           <wsdl:input>
               <soap:body use="literal"/>
           </wsdl:input>
           <wsdl:output>
               <soap:body use="literal"/>
           </wsdl:output>
           <wsdl:fault name="InvalidResourcePropertyQNameFault">
               <soap:fault name="InvalidResourcePropertyQNameFault"
use="literal"/>
           </wsdl:fault>
           <wsdl:fault name="InvalidSetResourcePropertiesRequestContentFault">
               <soap:fault
name="InvalidSetResourcePropertiesRequestContentFault" use="literal"/>
           </wsdl:fault>
           <wsdl:fault name="SetResourcePropertyRequestFailedFault">
               <soap:fault name="SetResourcePropertyRequestFailedFault"
use="literal"/>
           </wsdl:fault>
           <wsdl:fault name="ResourceUnknownFault">
               <soap:fault name="ResourceUnknownFault" use="literal"/>
           </wsdl:fault>
           <wsdl:fault name="UnableToModifyResourcePropertyFault">
               <soap:fault name="UnableToModifyResourcePropertyFault"
use="literal"/>
           </wsdl:fault>
       </wsdl:operation>
       <wsdl:operation name="QueryResourceProperties">
           <soap:operation soapAction="http://docs.oasis-
open.org/wsrf/2004/06/wsrf-WS-ResourceProperties/QueryResourceProperties"/>
           <wsdl:input>
               <soap:body use="literal"/>
           </wsdl:input>
           <wsdl:output>
               <soap:body use="literal"/>
           </wsdl:output>
           <wsdl:fault name="UnknownQueryExpressionDialectFault">
               <soap:fault name="UnknownQueryExpressionDialectFault"
use="literal"/>
           </wsdl:fault>
           <wsdl:fault name="QueryEvaluationErrorFault">
               <soap:fault name="QueryEvaluationErrorFault" use="literal"/>
           </wsdl:fault>
           <wsdl:fault name="InvalidQueryExpressionFault">
               <soap:fault name="InvalidQueryExpressionFault" use="literal"/>
           </wsdl:fault>
           <wsdl:fault name="InvalidResourcePropertyQNameFault">
               <soap:fault name="InvalidResourcePropertyQNameFault"
use="literal"/>
```

```
            </wsdl:fault>
            <wsdl:fault name="ResourceUnknownFault">
                <soap:fault name="ResourceUnknownFault" use="literal"/>
            </wsdl:fault>
        </wsdl:operation>
        <wsdl:operation name="GetMultipleResourceProperties">
            <soap:operation soapAction="http://docs.oasis-
open.org/wsrf/2004/06/wsrf-WS-
ResourceProperties/GetMultipleResourceProperties"/>
            <wsdl:input>
                <soap:body use="literal"/>
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal"/>
            </wsdl:output>
            <wsdl:fault name="InvalidResourcePropertyQNameFault">
                <soap:fault name="InvalidResourcePropertyQNameFault"
use="literal"/>
            </wsdl:fault>
            <wsdl:fault name="ResourceUnknownFault">
                <soap:fault name="ResourceUnknownFault" use="literal"/>
            </wsdl:fault>
        </wsdl:operation>
    </wsdl:binding>
</wsdl:definitions>
```

B.9 `Gaja3d_service.wsdl` (Gaja3dService)

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="Gaja3dService"
targetNamespace="http://org.kalypso.gaja3d.service/service"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:binding="http://org.kalypso.gaja3d.service/bindings"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">

    <wsdl:import namespace="http://org.kalypso.gaja3d.service/bindings"
location="Gaja3d_bindings.wsdl"/>
    <wsdl:service name="Gaja3dService">
        <wsdl:port name="Gaja3dPortTypePort"
binding="binding:Gaja3dPortTypeSOAPBinding">
            <soap:address location="http://localhost:8080/wsrf/services/"/>
        </wsdl:port>
    </wsdl:service>
</wsdl:definitions>
```

Annex C

XML Example Documents

C.1 Trajectory Service process (WPS Execute request)

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:m0="http://www.opengis.net/wps/1.0.0"
xmlns:m1="http://www.opengis.net/ows/1.1"
xmlns:m2="http://www.w3.org/XML/2001/namespace"
xmlns:m3="http://www.w3.org/1999/xlink">
    <SOAP-ENV:Body>
        <m:Execute_TrajService xmlns:m="http://www.stfc.ac.uk">
            <m0:Execute service="WPS" version="1.0.0" language="en-GB">
                <m1:Identifier codeSpace="RunTraj"/>
                <m0:DataInputs>
                    <m0:Input>
                        <m1:Identifier codeSpace="">Datetime</m1:Identifier>
                        <m0:Data>
                            <m0:LiteralData
dataType="timestamp">2008111112</m0:LiteralData>
                        </m0:Data>
                    </m0:Input>
                    <m0:Input>
                        <m1:Identifier codeSpace="">Length</m1:Identifier>
                        <m0:Data>
                            <m0:LiteralData dataType="integer">2</m0:LiteralData>
                        </m0:Data>
                    </m0:Input>
                    <m0:Input>
                        <m1:Identifier codeSpace="">Time_step</m1:Identifier>
                        <m0:Data>
                            <m0:LiteralData
dataType="integer">1800</m0:LiteralData>
                        </m0:Data>
                    </m0:Input>
                    <m0:Input>
                        <m1:Identifier codeSpace="">Lat</m1:Identifier>
                        <m0:Data>
                            <m0:LiteralData
dataType="float">54.1</m0:LiteralData>
                        </m0:Data>
                    </m0:Input>
                    <m0:Input>
                        <m1:Identifier codeSpace="">Lon</m1:Identifier>
                        <m0:Data>
                            <m0:LiteralData dataType="float">-
5.1</m0:LiteralData>
                        </m0:Data>
                    </m0:Input>
                    <m0:Input>
                        <m1:Identifier codeSpace="">Lev</m1:Identifier>
                        <m0:Data>
                            <m0:LiteralData dataType="float">900</m0:LiteralData>
                        </m0:Data>
```

```
                  </m0:Input>
              </m0:DataInputs>
              <m0:ResponseForm>
                  <m0:ResponseDocument storeExecuteResponse="true"
lineage="false" status="true">
                      <m0:Output uom="" mimeType="" encoding="" schema=""
asReference="true"/>
                  </m0:ResponseDocument>
              </m0:ResponseForm>
          </m0:Execute>
          <m:Username>xxx</m:Username>
          <m:Password>xxxxxx</m:Password>
      </m:Execute_TrajService>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

C.2 Trajectory Service process (WPS Execute response)

n/a

C.3 Plume Service process (WPS Execute request)

```
<?xml version="1.0" encoding="UTF-8"?>
<Execute xmlns="http://www.opengis.net/wps/1.0.0" version="1.0.0">
    <Identifier
xmlns="http://www.opengis.net/ows/1.1">org.n52.wps.server.algorithm.simulation.
PlumeTimeSeriesService</Identifier>
    <DataInputs>
        <Input>
            <Identifier
xmlns="http://www.opengis.net/ows/1.1">TRAJECTORY</Identifier>
            <Data>
                <LiteralData
dataType="xs:string">http://glue.badc.rl.ac.uk/ndg/traj.xml</LiteralData>
            </Data>
        </Input>
        <Input>
            <Identifier
xmlns="http://www.opengis.net/ows/1.1">WIDTH</Identifier>
            <Data>
                <LiteralData dataType="xs:int">600</LiteralData>
            </Data>
        </Input>
        <Input>
            <Identifier
xmlns="http://www.opengis.net/ows/1.1">HEIGHT</Identifier>
            <Data>
                <LiteralData dataType="xs:int">600</LiteralData>
            </Data>
        </Input>
        <Input>
            <Identifier
xmlns="http://www.opengis.net/ows/1.1">IndividualPhysicalMemory</Identifier>
            <Data>
                <LiteralData dataType="xs:int">536870912</LiteralData>
            </Data>
        </Input>
        <Input>
            <Identifier
xmlns="http://www.opengis.net/ows/1.1">IndividualCPUTime</Identifier>
            <Data>
```

```
                <LiteralData dataType="xs:int">1800</LiteralData>
            </Data>
        </Input>
        <Input>
            <Identifier
xmlns="http://www.opengis.net/ows/1.1">TotalCPUCount</Identifier>
            <Data>
                <LiteralData dataType="xs:int">2</LiteralData>
            </Data>
        </Input>
    </DataInputs>
    <ResponseForm>
        <ResponseDocument storeExecuteResponse="false" status="true">
            <Output asReference="true">
                <Identifier
xmlns="http://www.opengis.net/ows/1.1">IMAGE_0</Identifier>
            </Output>
        </ResponseDocument>
        <ResponseDocument storeExecuteResponse="false" status="true">
            <Output asReference="true">
                <Identifier
xmlns="http://www.opengis.net/ows/1.1">IMAGE_1</Identifier>
            </Output>
        </ResponseDocument>
        <ResponseDocument storeExecuteResponse="false" status="true">
            <Output asReference="true">
                <Identifier
xmlns="http://www.opengis.net/ows/1.1">IMAGE_2</Identifier>
            </Output>
        </ResponseDocument>
        <ResponseDocument storeExecuteResponse="false" status="true">
            <Output asReference="true">
                <Identifier
xmlns="http://www.opengis.net/ows/1.1">IMAGE_3</Identifier>
            </Output>
        </ResponseDocument>
        <ResponseDocument storeExecuteResponse="false" status="true">
            <Output asReference="true">
                <Identifier
xmlns="http://www.opengis.net/ows/1.1">IMAGE_4</Identifier>
            </Output>
        </ResponseDocument>
        <ResponseDocument storeExecuteResponse="false" status="true">
            <Output asReference="true">
                <Identifier
xmlns="http://www.opengis.net/ows/1.1">IMAGE_5</Identifier>
            </Output>
        </ResponseDocument>
        <ResponseDocument storeExecuteResponse="false" status="true">
            <Output asReference="true">
                <Identifier
xmlns="http://www.opengis.net/ows/1.1">IMAGE_6</Identifier>
            </Output>
        </ResponseDocument>
        <ResponseDocument storeExecuteResponse="false" status="true">
            <Output asReference="true">
                <Identifier
xmlns="http://www.opengis.net/ows/1.1">IMAGE_7</Identifier>
            </Output>
        </ResponseDocument>
        <ResponseDocument storeExecuteResponse="false" status="true">
            <Output asReference="true">
                <Identifier
xmlns="http://www.opengis.net/ows/1.1">IMAGE_8</Identifier>
```

```
            </Output>
        </ResponseDocument>
        <ResponseDocument storeExecuteResponse="false" status="true">
            <Output asReference="true">
                <Identifier
xmlns="http://www.opengis.net/ows/1.1">IMAGE_9</Identifier>
            </Output>
        </ResponseDocument>
    </ResponseForm>
</Execute>
```

C.4 Plume Service process (WPS Execute response)

```
<?xml version="1.0" encoding="UTF-8"?>
<ns:ExecuteResponse xmlns:ns="http://www.opengis.net/wps/1.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/wps/1.0.0
http://geoserver.itc.nl:8080/wps/schemas/wps/1.0.0/wpsExecute_response.xsd"
serviceInstance="http://giv-bandog.uni-
muenster.de:8080/wps/WebProcessingService?SERVICE=GetCapabilities&amp;SERVICE=W
PS" xml:lang="en-US" service="WPS" version="1.0.0" statusLocation="http://giv-
bandog.uni-muenster.de:8080/wps/RetrieveResultServlet?id=1240838485333">
    <ns:Process ns:processVersion="2">
        <ns1:Identifier
xmlns:ns1="http://www.opengis.net/ows/1.1">org.n52.wps.server.algorithm.simulat
ion.PlumeTimeSeriesService</ns1:Identifier>
        <ows:Title xmlns:wps="http://www.opengis.net/wps/1.0.0"
xmlns:ows="http://www.opengis.net/ows/1.1"
xmlns:xlink="http://www.w3.org/1999/xlink">Plume Forecast</ows:Title>
    </ns:Process>
    <ns:Status creationTime="2009-04-27T15:21:25.240+02:00">
        <ns:ProcessSucceeded>Process successful</ns:ProcessSucceeded>
    </ns:Status>
    <ns:ProcessOutputs>
        <ns:Output>
            <ns1:Identifier
xmlns:ns1="http://www.opengis.net/ows/1.1">IMAGE_0</ns1:Identifier>
            <ows:Title xmlns:wps="http://www.opengis.net/wps/1.0.0"
xmlns:ows="http://www.opengis.net/ows/1.1"
xmlns:xlink="http://www.w3.org/1999/xlink">One of the rendered image
files.</ows:Title>
            <ns:Reference schema="" href="http://giv-bandog.uni-
muenster.de:8080/wps/RetrieveResultServlet?id=1240838485333IMAGE_0"/>
        </ns:Output>
    </ns:ProcessOutputs>
</ns:ExecuteResponse>
```

C.5 Rainfall Interpolation process (WPS Execute request)

```
<?xml version="1.0" encoding="UTF-8"?>
<ns:Execute version="1.0.0" xmlns:ns="http://www.opengis.net/wps/1.0.0"
xmlns:rainfall="http://www.52north.org/ows-6/debris-flow/rainfall">
    <ns1:Identifier
xmlns:ns1="http://www.opengis.net/ows/1.1">org.n52.wps.server.algorithm.interpo
lation.RainfallDataInterpolation</ns1:Identifier>
    <ns:DataInputs>
        <ns:Input>
            <ns1:Identifier
xmlns:ns1="http://www.opengis.net/ows/1.1">RAINFALL</ns1:Identifier>
            <ns:Data>
                <ns:ComplexData xmlns:rainfall="http://www.52north.org/ows-
6/debris-flow/rainfall">
                    <rainfall:ObservationStation id="1">
```

```
                        <rainfall:Station id="1" rainFall="10" XCoordinate="13"
YCoordinate="20"/>
                        <rainfall:Station id="2" rainFall="99" XCoordinate="17"
YCoordinate="15"/>
                        <rainfall:Station id="3" rainFall="78" XCoordinate="20"
YCoordinate="22"/>
                        <rainfall:Station id="4" rainFall="11" XCoordinate="17"
YCoordinate="27"/>
                        <rainfall:Station id="5" rainFall="37" XCoordinate="24"
YCoordinate="25"/>
                        <rainfall:Station id="6" rainFall="0" XCoordinate="27"
YCoordinate="28"/>
                    </rainfall:ObservationStation>
                </ns:ComplexData>
            </ns:Data>
        </ns:Input>
        <ns:Input>
            <ns1:Identifier
xmlns:ns1="http://www.opengis.net/ows/1.1">X1</ns1:Identifier>
            <ns:Data>
                <ns:LiteralData dataType="xs:double">10</ns:LiteralData>
            </ns:Data>
        </ns:Input>
        <ns:Input>
            <ns1:Identifier
xmlns:ns1="http://www.opengis.net/ows/1.1">Y1</ns1:Identifier>
            <ns:Data>
                <ns:LiteralData dataType="xs:double">30</ns:LiteralData>
            </ns:Data>
        </ns:Input>
        <ns:Input>
            <ns1:Identifier
xmlns:ns1="http://www.opengis.net/ows/1.1">X2</ns1:Identifier>
            <ns:Data>
                <ns:LiteralData dataType="xs:double">30</ns:LiteralData>
            </ns:Data>
        </ns:Input>
        <ns:Input>
            <ns1:Identifier
xmlns:ns1="http://www.opengis.net/ows/1.1">Y2</ns1:Identifier>
            <ns:Data>
                <ns:LiteralData dataType="xs:double">10</ns:LiteralData>
            </ns:Data>
        </ns:Input>
        <ns:Input>
            <ns1:Identifier
xmlns:ns1="http://www.opengis.net/ows/1.1">CELLWIDTH</ns1:Identifier>
            <ns:Data>
                <ns:LiteralData dataType="xs:double">5</ns:LiteralData>
            </ns:Data>
        </ns:Input>
        <ns:Input>
            <ns1:Identifier
xmlns:ns1="http://www.opengis.net/ows/1.1">TOPN</ns1:Identifier>
            <ns:Data>
                <ns:LiteralData dataType="xs:int">3</ns:LiteralData>
            </ns:Data>
        </ns:Input>
    </ns:DataInputs>
    <ns:ResponseForm>
        <ns:RawDataOutput mimeType="text/plain">
```

```
            <ows:Identifier xmlns:ows="http://www.opengis.net/ows/1.1"
xmlns:wps="http://www.opengis.net/wps/1.0.0"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">INTERPOLATION</ows:Identifier>
        </ns:RawDataOutput>
    </ns:ResponseForm>
</ns:Execute>
```

C.6 Rainfall Interpolation process (WPS Execute response)

```
<?xml version="1.0" encoding="UTF-8"?>
<ns:ExecuteResponse xmlns:ns="http://www.opengis.net/wps/1.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/wps/1.0.0
http://geoserver.itc.nl:8080/wps/schemas/wps/1.0.0/wpsExecute_response.xsd"
serviceInstance="http://giv-bandog.uni-
muenster.de:8080/wps/WebProcessingService?SERVICE=GetCapabilities&amp;SERVICE=W
PS" xml:lang="en-US" service="WPS" version="1.0.0">
    <ns:Process ns:processVersion="2">
        <ns1:Identifier
xmlns:ns1="http://www.opengis.net/ows/1.1">org.n52.wps.server.algorithm.interpo
lation.RainfallDataInterpolation</ns1:Identifier>
        <ows:Title xmlns:wps="http://www.opengis.net/wps/1.0.0"
xmlns:ows="http://www.opengis.net/ows/1.1"
xmlns:xlink="http://www.w3.org/1999/xlink">Rainfall Interpolation
Service</ows:Title>
    </ns:Process>
    <ns:Status creationTime="2009-04-27T15:19:24.911+02:00">
        <ns:ProcessSucceeded>The service succesfully processed the
request.</ns:ProcessSucceeded>
    </ns:Status>
    <ns:ProcessOutputs>
        <ns:Output>
            <ns1:Identifier
xmlns:ns1="http://www.opengis.net/ows/1.1">INTERPOLATION</ns1:Identifier>
            <ns1:Identifier
xmlns:ns1="http://www.opengis.net/ows/1.1">INTERPOLATION</ns1:Identifier>
            <ows:Title xmlns:wps="http://www.opengis.net/wps/1.0.0"
xmlns:ows="http://www.opengis.net/ows/1.1"
xmlns:xlink="http://www.w3.org/1999/xlink">An HTTP URL to a file that includes
the interpolated rainfall data for each cell.</ows:Title>
            <ns:Data>
                <ns:LiteralData dataType="xs:string">http://giv-bandog.uni-
muenster.de:8080/wps/data/1240838367208/result.csv</ns:LiteralData>
            </ns:Data>
        </ns:Output>
    </ns:ProcessOutputs>
</ns:ExecuteResponse>
```

C.7 Geophone Analysis process (WPS Execute request)

```
<?xml version="1.0" encoding="UTF-8"?>
<Execute xmlns="http://www.opengis.net/wps/1.0.0" version="1.0.0">
    <Identifier
xmlns="http://www.opengis.net/ows/1.1">org.n52.wps.server.algorithm.interpolati
on.WaveletTransformation</Identifier>
    <DataInputs>
        <Input>
            <Identifier
xmlns="http://www.opengis.net/ows/1.1">WAVELET</Identifier>
            <Data>
```

```
                <LiteralData dataType="xs:string">http://giv-bandog.uni-
muenster.de:8080/wps/1630.d02</LiteralData>
            </Data>
        </Input>
    </DataInputs>
    <ResponseForm>
        <RawDataOutput mimeType="text/plain">
            <Identifier xmlns="http://www.opengis.net/ows/1.1">TRANSFORMED
WAVELET</Identifier>
        </RawDataOutput>
    </ResponseForm>
</Execute>
```

C.8 Geophone Analysis process (WPS Execute response)

```
<?xml version="1.0" encoding="UTF-8"?>
<ns:ExecuteResponse xmlns:ns="http://www.opengis.net/wps/1.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/wps/1.0.0
http://geoserver.itc.nl:8080/wps/schemas/wps/1.0.0/wpsExecute_response.xsd"
serviceInstance="http://giv-bandog.uni-
muenster.de:8080/wps/WebProcessingService?SERVICE=GetCapabilities&amp;SERVICE=W
PS" xml:lang="en-US" service="WPS" version="1.0.0">
    <ns:Process ns:processVersion="1">
        <ns1:Identifier
xmlns:ns1="http://www.opengis.net/ows/1.1">org.n52.wps.server.algorithm.interpo
lation.WaveletTransformation</ns1:Identifier>
        <ows:Title xmlns:wps="http://www.opengis.net/wps/1.0.0"
xmlns:ows="http://www.opengis.net/ows/1.1"
xmlns:xlink="http://www.w3.org/1999/xlink">Wavelet Transformation
Service</ows:Title>
    </ns:Process>
    <ns:Status creationTime="2009-04-27T15:20:28.693+02:00">
        <ns:ProcessSucceeded>The service succesfully processed the
request.</ns:ProcessSucceeded>
    </ns:Status>
    <ns:ProcessOutputs>
        <ns:Output>
            <ns1:Identifier
xmlns:ns1="http://www.opengis.net/ows/1.1">TRANSFORMED WAVELET</ns1:Identifier>
            <ns1:Identifier
xmlns:ns1="http://www.opengis.net/ows/1.1">TRANSFORMED WAVELET</ns1:Identifier>
            <ows:Title xmlns:wps="http://www.opengis.net/wps/1.0.0"
xmlns:ows="http://www.opengis.net/ows/1.1"
xmlns:xlink="http://www.w3.org/1999/xlink">The location (HTTP-URL) of the
transformed wavelet file.</ows:Title>
            <ns:Data>
                <ns:LiteralData dataType="xs:string">http://giv-bandog.uni-
muenster.de:8080/wps/data/1240838433130/wavelet595969242953044l004-
L3.txt</ns:LiteralData>
            </ns:Data>
        </ns:Output>
    </ns:ProcessOutputs>
</ns:ExecuteResponse>
```

Bibliography

[1] Baranski, B. (2008). Grid Computing Enabled Web Processing Service. GI-Days 2008, Münster, Germany.

[2] Lan-Kun Chung, Bastian Baranski, Yao-Min Fang, Yin-Huei Chang, Tien-Yin Chou and Bing Jean Lee (2009). A SOA based debris flow monitoring system - Architecture and proof-of-concept implementation. Geoinformatics 2009, Fairfax, USA (not published yet)

[3] Di, L., Chen, A., Yang W., & Zhao, P. (2003). The Integration of Grid Technology with OGC Web Services (OWS) in NWGISS for NASA EOS Data . GGF8 & HPDC12. 24 – 27 June at Seattle.

[4] Erl, T. (2005). Service-Oriented Architecture : Concepts, Technology, and Design. Upper Saddle River, Prentice Hall PTR.

[5] Foster (2002). What is the Grid? A Three Point Checklist. GRIDToday **1**(6), 2002

[6] Foster and Kesselman (1998). The Grid: A Blueprint for a New Computing Infrastructure. Morgan Kaufmann

[7] A. S. Grimshaw, M. M. Morgan, D. Merrill, Hiro Kishimoto, Andreas Savva, Chris Smith, Dave Berry, "An Open Grid Services Architecture Primer," IEEE Computer, vol. 42, no. 2, pp. 27-34, February, 2009.

[8] I. T. Foster, C. Kesselman, G. Tsudik, and S. Tuecke, "A security architecture for computational grids," in ACM Conference on Computer and Communications Security, 1998, pp. 83-92. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.57.5657

[9] Hartig. K (2008) What is Cloud Computing? The cloud is a virtualization of resources that maintains and manages itself. .NET Developers Journal, SYS-CON Media.

[10] Hazel, T., Toma, L., Vahrenhold, J. & Wickremesinghe, R. (2008). Terracost: Computing least-cost-path surfaces for massive grid terrains. J. Exp. Algorithmics 12 (Jun. 2008), 1-31.

[11] Kiehle, C., Greve, K. & C. Heier (2007). Requirements for Next Generation Spatial Data Infrastructures - Standardized Web Based Geoprocessing and Web Service Orchestration. In: Transactions in GIS. 11(6), p. 819-834.

[12] Kurzbach, S. & E. Pasche (2009), A 3d Terrain Discretization Grid Service for Hydrodynamic Modeling. In: Proceedings of the 8th International Conference on Hydroinformatics. Concepción, Chile, Jan. 2009

[13] Lanig, S., Schilling, A., Stollberg, B. & Zipf, A. (2008). Towards Standards-based Processing of Digital Elevation Models for Grid Computing through Web Processing Service (WPS). The 2008 International Conference on Computational Science and its Applications (ICCSA2008), Perugia, Italy.

[14] John Lee and Ron Ben-Natan. Integrating Service Level Agreements: Optimizing Your OSS for SLA Delivery. John Wiley & Sons, Inc., New York, NY, USA, 2002.

[15] Masser, Ian (2005). GIS worlds : Creating spatial data infrastructures. 1st ed. Redlands, California: ESRI Press.

[16] Nash, Edward (2008). WPS Application Profiles for Generic and Specialised Processes, GI-Days 2008, Münster, Germany.

[17] Weerawarana, Sanjiva (2006). Web services platform architecture: SOAP, WSDL, WS-policy, WS-addressing, WS-BPEL, WS-reliable messaging and more. 4. printing ed. Upper Saddle River, N.J. u.a.: Prentice Hall/PTR.

[18] Woolf, A (2006). Wrappers, portlets, resource-orientation and OGC in Earth-System Science Grids, Grid ad-hoc, OGC TC Edinburgh, June 2006 [http://portal.opengeospatial.org/files/?artifact_id=15966]