A Service-Oriented Framework for Real-time and Distributed Geoprocessing

Bastian Schaeffer[1], Bastian Baranski[1], Theodor Foerster[2] & Johannes Brauner[3]

[1] Institute for Geoinformatics, Muenster, Germany
{schaeffer,baranski}@uni-muenster.de

[2] International Institute for Geo-information Science and Earth Observation (ITC), Enschede, the Netherlands
foerster@itc.nl

[3]Technische Universität Dresden, Dresden, Germany
johannes.brauner@tu-dresden.de

Abstract

With increasing availability of network bandwidth, server side processing capacity and advancements regarding the standardization of Web Service technologies, it becomes feasible to process geodata over the Web. As a result, web-based geoinformation is generated, which supports in particular spatial decision making. However, existing Spatial Data Infrastructures (SDI) mainly address data retrieval and portrayal, but to provide web-based geoinformation through SDIs web-based geoprocessing is required. This was the starting point to develop a framework which integrates existing desktop Geographical Information System functionality in SDIs. Moreover, an approach will be introduced to improve complex and large-scale geoprocesses in two dimensions: On the vertical scale, by significantly increasing the performance of single processes by means of Grid Computing technology and by improving other Quality of Service aspects. On the horizontal scale, by chaining this highly efficient processing services in order to automate these business processes. The relevance of this work will be demonstrated in a real-world scenario focusing on air quality assessment based on real-time data with the help of open source components.

1. Introduction

With the advent of the Service-Oriented Architecture (SOA) paradigm (Erl 2005) and advancements in Web Services technology (Weerawarana et al. 2006), Geographical Information Systems (GIS) underwent a substantial change from stand-alone applications to distributed service architectures manifested in Spatial Data Infrastructures (SDI) (Masser 2005). While existing SDIs mainly address geodata retrieval and portrayal (Kiehle et al. 2007) the gap between available geodata and required web-based geoinformation increases, especially in the context of increased network and processing capacities.

To overcome this gap, it is a common approach in mainstream IT to generate Web Services from existing functionality by adapting and wrapping it (Papazoglou and van den Heuvel 2007). This approach is also promising for geospatial applications. It allows the service provider to integrate stand-alone functionality, such as provided by GRASS GIS[1], into an SDI. Additionally, the integration of existing functionality is sustainable because it prevents reimplementation. Thus, geoprocesses can be exposed through distributed Web Services. Since interoperability is a key requirement of the SOA paradigm, which is ensured through standardization, open standards such as developed by the Open Geospatial Consortium (OGC) for geodata retrieval and portrayal create a strong foundation to establish Geoprocessing Services in existing SOAs.

The OGC Web Processing Service (WPS) interface specification[2] became an official standard in mid 2007 and is a major attempt to publish and perform geoprocesses on the web. Such a process can range from a simple geometric calculation (for example a simple spatial intersect operation) to a complex simulation process (for example calculating rain water discharge).

---

[1] GRASS GIS: http://grass.osgeo.org

[2] OpenGIS Web Processing Service Standard: http://portal.opengeospatial.org/files/?artifact_id=24151

However, 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. To improve the computational performance of processing such large amounts of real-time geodata, Grid Computing provides appropriate tools. Additionally, Cloud Computing (for example realized by Grid Computing) is a promising approach to improve the service availability and to support a specific level of Quality of Service (QoS). Although the application of Grid Computing is not novel to the mainstream IT-world (as shown e.g. by Foster and Kesselman (1998)), in the context of geospatial applications and OGC Web Services only little research has been conducted; see Baranski (2008), Lanig et al. (2008) and OWS-6 (2009).

Applying Grid Computing improves the performance of a complex application only on a vertical scale, but on a horizontal scale whole business processes can be optimized by composing geoprocessing workflows incorporating different types of distributed services (for retrieval and processing) (Alameh 2003). This enables outsourcing of specific tasks to highly specialized Web Services. As a result of this approach, workflows become flexible and are able to quickly adopt to changing requirements. Therefore, cross-enterprise workflows can be composed as virtual business processes (van der Aalst and van Hee 2004) since they integrate several processes from different partners. These composed workflows appear as a single process from an external point of view. Outsourcing of specific tasks to specialized services is still missing for geoprocessing workflows (Alameh 2003). The combination of the two aspects (Grid Computing and workflow) leads to a framework for high-performance and fully automated value-added geoprocessing workflows.

The introduced framework is applied to a real-world scenario for real-time and distributed geoprocessing involving live sensor data and high-resolution geodata, both served through standardized Web Services. In the scenario a geoprocessing workflow has to be modelled to identify areas with a high Particulate Matter pollution. To accomplish this scenario, computationally extensive processes are required. The required steps in the scenario such as modelling, execution of geoprocesses and portrayal of the process results will be achieved through an integrated client application, which provides an easy access to the technologies described in this paper.

It is important to note, that the proposed framework is fully based on components available through open source licenses (GPL). Additionally, all Web Services listed in the framework are compliant to open standards. This openness of our approach is an essential aspect, since components can be replaced and extended for future requirements. We want to stress that the openness results finally in a sustainable approach for building SOAs and prevents from vendor lock-in (license and interface-wise). The presented framework is published as open source by the Geoprocessing Community at the 52°North Open Source initiative. All the implementations are part of the 52°North WPS framework (in the following referred to as 52°North WPS)[3]. A thorough performance evaluation of the framework is outside the scope of this paper, but e.g. Scholten et al (2006) review the performance of distributed geoprocessing applications.

At first, the paper provides a review of the basic concepts applied in the framework: OGC WPS interface specification, Grid Computing and geoprocessing workflows. After that the paper presents the combined approach of Grid Computing and geoprocessing workflows. To build sustainable workflows, existing geoprocessing functionality incorporated in stand-alone desktop GIS has to be exposed by Web Services. In this context the wrapping of GIS functionality with a WPS interface is discussed based on the experiences with GRASS GIS. In Section 4 the proposed framework is applied to a real-world scenario for proving the introduced concepts. The paper ends with a discussion and a conclusion about the framework.

2. Technical Background and Related Work

This section provides a review of related work in the context of OGC Web Processing Service, Grid Computing and Web Service Orchestration.

[3] The Geoprocessing community at 52° North: http://52north.org/wps

2.1 OGC Web Processing Service

The OGC Web Processing Service interface specification describes a standardized method to publish and execute web-based processes for any type of geoprocess. According to the WPS interface specification, a process is defined as any calculation operating on spatially referenced data.

In detail, the WPS interface specification describes three operations, which are all handled in a stateless manner: *GetCapabilities*, *DescribeProcess* and *Execute*. *GetCapabilities* is common to any type of OGC Web Service and returns service metadata. In case of WPS, it also returns a brief description of the processes offered by the specific service instance. To get more information about the hosted processes, the WPS provides process metadata through the *DescribeProcess* operation. This operation describes all parameters, which are required to run the process. Based on this information the client can perform the *Execute* operation upon the designated process. As every OGC Web Service, the WPS communicates through HTTP-GET and HTTP-POST based on an OGC-specific XML-message encoding.

Besides this basic communication pattern, the WPS interface provides functionality for scalable processing such as asynchronous processing (implemented using the pull model), storing of process results and processing of data references encoded as URLs. The application of URL references as input for specific processes is a promising feature, as it limits the volume of data sent between client and service and allows the service to apply specific caching strategies. The service retrieves the data once and reuses it multiple times, by using the reference as an identifier for data.

2.2 Grid Computing

The term *Grid Computing* is a diffuse phrase and there are many definitions available. This lack of a single and precise definition leads to a broad variety of understandings of this term. However, the term Grid Computing is frequently used when linking computational resources to solve time- and resource-intensive computations. Following the argumentation of Foster and Kesselmann (1998), the term Grid Computing must be evaluated in „terms of the applications, business value, and scientific results that it delivers, not its architecture". Moreover, Foster characterizes in his famous three-point-checklist[4] a system as a Grid that:

1. coordinates resources that are not subject to centralized control
2. is using standard, open, general-purpose protocols and interface
3. delivers nontrivial qualities of service.

The emerging term *Cloud Computing* overlaps with some concepts of distributed and Grid Computing (Hartig 2008). It uses a cloud metaphor to represent the internet or other large networking infrastructures. The paradigm behind the buzzword foresees a future in which data and computations will be handled by distributed facilities operated by third-party resources (Foster et al. 2008). The key characteristics of the cloud are the ability to scale and to provide computing power on-demand in a cost efficient way and the ability of the consumer to benefit from these resources 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[5]. In essence, Cloud Computing is not a completely new concept. Moreover, it combines a family of well known and established methods and technologies (for example SaaS as a model for software packaging and deployment, Virtualization as an efficient hosting platform and Grid Computing as a backbone for providing sufficient computational power). Besides, it describes a paradigm of outsourcing applications and specific tasks to a scalable infrastructure and therefore consequently enabling new business models with less up-front investments.

In the context of linking geospatial applications and SDIs with Grid Computing only little research has been conducted. Baranski (2008) and Lanig et al. (2008) extended the work of Di et al. (2003) and accomplished first experiments using Grid Computing technology for improving processing performance by distributing and parallel execution of processes. In the OWS-6 testbed (OWS-6 2009) the adoption of Open Grid Forum (OGF) standards for the WPS interface specification

[4] Ian Foster, "What is the Grid? A three point checklist": http://www.gridtoday.com/02/0722/100136.html

[5] The Open Cloud Manifesto homepage: http://www.opencloudmanifesto.org

was reviewed. Recently, the GDI-GRID[6] project has started focusing on the integration and processing of geodata 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 data enter hosting geospatial services running on the UK National Grid Service[9]. The SLA4D-GRID[10] project 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. A SLA-enabled WPS is one important use case in this research project.

2.3 Web Service Orchestration and Workflows

Complex business processes often require the functionality of multiple tasks. These tasks could be often realized as Web Services which can be composed to one workflow to represent the desired business process. Alonso et al. (2003) speak of a composite service and entitle the act of combining Web Services as *Web Service Composition*. The interaction of the Web Service in the workflow can be handled by *Web Service Orchestration*, which is a description of how combined Web Services interact on the message level (Pelz 2003). This description includes the business-logic and execution order. Thereby it can be distinguished from *Web Service Choreography*, which has a more collaborative character and is more focused on the public message exchange between multiple parties. Since *Web Service Choreography* specifies an explicit message interaction protocol, each service knows its predecessor and successor.

This differs from *Web Service Orchestration*, which is often realized by a central orchestration engine coordinating the Web Service interaction according to a predefined workflow description. Thereby, this approach allows the Web Services to be held loosely coupled (Weerawarana et al. 2006).

In this respect, the Business Process Execution Language (BPEL) has advanced to a de-facto standard in the mainstream IT-world. BPEL is an OASIS standard[11] for describing workflows based on elementary tasks implemented as Web Services using an XML-encoding. In particular, these so-called BPEL scripts describe the roles involved in the message exchange, supported port types and orchestration information of a process. Additionally, specific binding and deployment issues remain at the Web Service and not at the BPEL process itself. All these aspects make BPEL an appropriate candidate to represent geoprocessing workflows in SOAs.

3. An Approach towards Real-Time and Distributed Geoprocessing

This section presents the developed approach towards real-time and distributed geoprocessing. At first, the wrapping of stand-alone geoprocessing functionality as a Web Service is explained. Secondly, an approach for the gridification OGC Web Services is presented. Finally, both concepts are combined into an approach for geoprocessing workflows.

3.1 GIS Wrapping

A wrapper is responsible for transforming incoming service requests into adequate calls for the underlying (legacy) system without any modifications of the wrapping interface (i.e. WPS interface) (Erl 2005, Papazoglou and van den Heuvel 2007). In this section, requirements to wrap geoprocessing functionality are described. Additionally, an architecture for such a wrapper will be presented based on the popular stand-alone open source GRASS GIS and the 52°North WPS framework.

Table 3.1 provides an overview of requirements for successfully wrapping geoprocessing functionality. To wrap a large number of existing functionality, automation is a key aspect. In particular, the automated extraction of the description of the

[6] GDI-Grid project homepage: http://www.gdi-grid.de/

[7] SEE and SAW-GEO project homepage: http://edina.ac.uk/projects/seesaw/index.html

[8] JISC national academic data centre based at the University of Edinburgh: http://edina.ac.uk/

[9] UK National Grid Service (NGS): http://www.grid-support.ac.uk/

[10] SLA4D-GRID project homepage: http://www.sla4d-grid.de

[11] Web Services Business Process Execution Language, Version 2.0: http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html

functionality and the automated forward of the service call is a key requirement. This is achieved through a programmable interface (*shell scripting* in GRASS GIS), which is accessible for the wrapping framework. All other requirements are optional.

Table 3.1 *Mandatory and optional characteristics for successful wrapping GIS functionality.*

Requirement	Type of requirement
Programmable interface for unsupervised processing	mandatory
Multitasking	optional
Machine-readable interface descriptions	optional (for automatic creation of DescribeProcess documents mandatory)
Determination of coordinate reference systems	optional (technical level) / mandatory (regarding correct results)

Workspaces are common for importing, processing data and finally storing the results in GIS such as for instance in GRASS. However, this concept does not match the stateless fashion of WPS. Therefore, a temporary workspace has to be created. This implies that the data are not usable to the process afterwards and should be deleted after sending the response. This leads to the architecture shown in Figure 3.1.

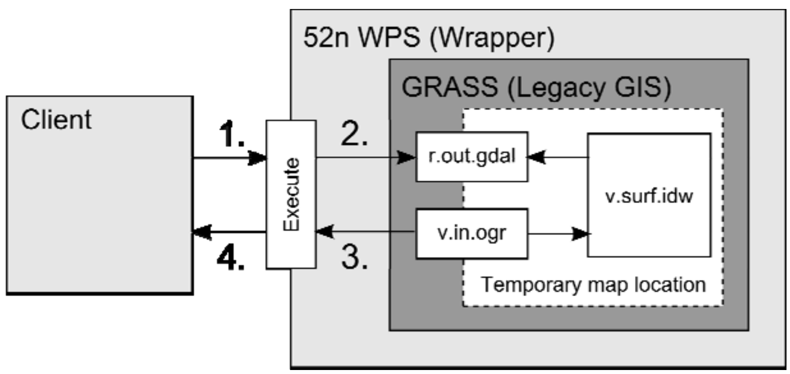

Fig. 3.1 *Architecture for wrapping GRASS with a WPS interface. An Execute request for an Inverse Distance Weighting (IDW) interpolation functionality of GRASS is sent and processed.*

The architecture consists of three components: the client, the 52°North WPS (wrapper) and the GRASS GIS (legacy GIS). An Execute request is received from the client (Figure 3.1, Step 1). In this example Inverse Distance Weighting (IDW) functionality is requested, which will also be used in the scenario (Section 4). The wrapper examines the request and creates an adequate shell script to invoke the underlying GRASS GIS (Step 2).This includes pre-configuring of GRASS, importing the input data, invoking the desired functionality and exporting the result. A reference to a temporary file is returned back to the wrapper (Step 3), and handed back to the client (Step 4). With minor modifications this architecture is applicable for other GIS as well. The type of programming interface might change from shell scripts to for example a programmable Python interface, but the architecture remains the same.

If the GIS provides machine-readable interface descriptions for its functionality, it is possible to transform these descriptions into DescribeProcess documents automatically. This is especially convenient if a full set of GIS functionality (in GRASS there are about 400 different pieces of functionality) should be offered by a WPS. To transform the XML-based descriptions of GRASS the eXtensible Stylesheet Language (XSL) is used. An XSL-transformation (XSLT) filter creates the DescribeProcess document elements via if-then-else statements, case differentiation and other information (e.g. MIME type for complex parameters).The XSLT-filter has to be implemented only once and can be applied to extract any process description fully-automated in the future.

Regarding GRASS, there are two semantic problems which constrain this fully-automated approach. Firstly, GRASS uses a non-typed parameter input system. This has certain drawbacks for WPS, as datasets and their formats are unknown during runtime. Hence, the machine controlled XSLT-filter is unable to detect which parameter has to be supplemented with additional information about data types and formats. User intervention may be required to decide which parameter is a complex one, especially in the case of multiple input parameters.

Secondly, not all GIS functionality available in a GIS is applicable to be exposed as a WPS process. For example, data visualization is not considered to be a designated process but more an integral part of a client application. The same applies to interactive functionality: the stateless nature of WPS does not permit interactions during processing. Again, user intervention is required to exclude unsuitable functionality from being offered through the WPS.

Nevertheless, this semi-automatic approach to create process descriptions is much more convenient than creating a full set of DescribeProcess documents manually.

Further details about the architecture and implementation can be found in Brauner and Schaeffer (2008).

3.2 Grid Computing

Two potential ways can be identified to adapt OGC Web Services (OWS) to domain-specific requirements of a Grid Computing environment and to make use of Open Grid Forum (OGF) and related specifications, concepts and their implementations.

At first, the OWS interacts with distributed computing resource in the backend (encapsulating other resources). In this simple approach the OWS could use other grid services, for example, to send calculation jobs to compute resources or for accessing distributed databases based on well defined mechanisms. In the course of this work this approach will be referred to as "encapsulation" as the clients of the OWS are not exposed to any changes of the OWS interface. The encapsulation fosters the performance of distributed time- and resource-intensive processes and the access of large and distributed datasets.

Secondly, the OWS is integrated within other Grid Computing environments. Therefore, the service has to be fully embedded - mostly as a stateful web service - into the grid middleware. This approach offers opportunities to realize benefit from existing middleware solutions such as intelligent workflow management, reliability, security and other QoS. In the course of this work this approach is referred to as "integration" as the OWS becomes a service that clients may utilize alongside others in the Grid Computing environment. Current OGC standards define typically a stateless message exchange via plain HTTP-GET and HTTP-POST protocol, but most of common grid middleware applications are based on the stateful Web Service Resource Framework (WSRF). Therefore, regarding the integration, the gridified OWS must be accessed via an OGC-compliant proxy component.

The presented implementation of an encapsulating WPS (in the course of this work referred as WPS-G) is based on the 52°North WPS and the open source UNICORE grid middleware[12]. A brief architecture overview is shown in Figure 3.2.

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 or assessed automatically. After preparing each computational 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.

[12] UNICORE grid middleware: http://www.unicore.eu

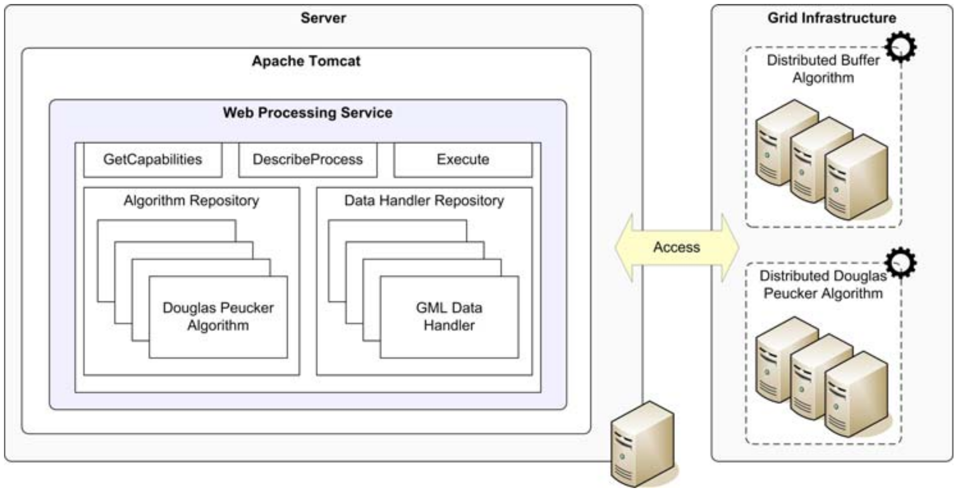

Fig. 3.2 *An architectural overview of the gridified WPS implementation.*

3.3 Scalable Geoprocessing Workflows

This section describes the integration of the GRASS-based (Section 3.1) and gridified (Section 3.2) WPS into a geoprocessing workflow. Since the interfaces of the gridified and GRASS-based WPS instances were not changed. Hence, the WPS-G and the GRASS-based WPS can be exactly treated like any OGC compliant WPS. Furthermore, a WPS is a Web Service, which can be used to create geoprocessing workflows in SOAs (Section 2). For instance, Weiser and Zipf (2007) showed this for simple OGC Web Services without a desktop GIS backend or grid technology. The Orchestra project developed similar approaches for the orchestration of geospatial web services, but neither incorporated grid technologies nor used standard OGC web services as frontend to existing geoprocessing functionality (Usländer et al. 2007). Besides, a separate class of geospatial web services was developed for the Orchestra project, which has not been adopted by international standardization bodies yet.

Our approach goes beyond that and allows the reuse of existing geoprocessing workflows by exposing the geoprocessing workflow as a simple WPS process using the WPS-T approach (T stands for transactional) (Schaeffer 2008). This approach enables the dynamic deployment of arbitrary geoprocessing workflows. In our work the approach applies workflows based on BPEL scripts describing web service chains of WPS instances enhanced with GRASS and grid technologies. This approach is utilized in the scenario (Section 4) and depicted in Figure 4a. Following this approach, the complexity of grid technology and stand-alone GIS functionality is encapsulated from the user.

4. A Scenario for Real-Time and Distributed Geoprocessing

This section presents a real-world scenario, which illustrates the presented concepts incorporated in the developed framework and acts as a proof-of-concept. Ambient air quality has been identified by the European Commission as one of the most critical environmental aspects. Therefore, it is one of the major objectives in the Sixth Environmental Action Programme[13]. Through legislation, the European Union has also been active in this field for several decades. With the Council Directives 1999/30/EC and 1996/62/EC strict air quality limits for several air quality parameters were specified, which have to be adapted by national law by all member states.

This scenario addresses Particulate Matter (PM10) as one of the most hazardous aerosols. Several studies have researched the correlation of PM10 and health effects. For instance, Pope et al. (2002) found that for each rise of 10 µg/m^3 the lung cancer rate increases by 4-8%. Hence, the real-time monitoring of PM10 is a crucial task to support decision makers in

[13] The Sixth Environment Action Programme of the European Community: http://ec.europa.eu/environment/newprg/index.htm

protecting public health and to comply with the EU regulations. In order to automate this business process, a model realized as a geoprocessing workflow will be developed to monitor real-time spatial distributions of PM10 by applying the technical concepts introduced in this paper.

Figure 4a presents the deployed workflow. It was modelled with the 52°North Workflow Modeler, which automatically creates WSDL documents for the workflow partners and describes the workflow via BPEL. While for instance Weiser and Zipf (2007) also showed the orchestration of OGC Web Services with BPEL, they had to build the WSDL manually. Our approach goes beyond that and exposes the deployed workflow as a simple WPS process using the WPS-T approach (Section 3.3). The client on the left-hand side (Figure 4a) is now capable to invoke the workflow like any other WPS process (Step 1). Only the reference to the sensor data has to be included in the request. In our case, the 52°North WPS uDig client application (Schaeffer and Foerster 2008) can be used to invoke the service and visualize the results (Figure 4b).

After invoking the workflow represented by a WPS process, hosted on the WPS-T, the underlying workflow engine is triggered in the backend (Step 2). Upon then, the workflow engine orchestrates the workflow and invokes each of the workflow partners. At first, a WPS filter process, which takes a URL from an OGC Sensor Observation Service (SOS) as input (Step 3) is triggered. The SOS instance is located on top of a sensor network and allows the client to query the observed phenomena in a standardized format. In this scenario real-time air quality data from the state of North-Rhine Westphalia, located in the western part of Germany, is available through the SOS instance. The SOS instance provides up-to-date data of 56 monitoring stations distributed through out this region. Internally, the WPS queries the SOS for its latest available sensor data regarding this specific region (Step 4). Next, the retrieved sensor data is converted into the common Geography Markup Language (GML) format (Step 5), which can be used for further processing. In particular, the geometry and observed values are extracted and new GML point features are created. This process is performed by a gridified WPS, which delegates the conversion task to its underlying grid middleware due to high computational complexity. The following step (Step 6) takes the resulting point data from the second process as input and interpolates them with the GRASS Inverse Distance Weighting (IDW) method to a resulting GeoTIFF raster. In the last step (Step 7) the GeoTIFF raster becomes visualized according to the GRASS color scheme for air pollution is applied, in which the color red represents values above a specific threshold (40 µg PM10 /m³ air as the current EU legislation limit). The output is visualized as a new layer in the open source uDig client (Figure 5b).

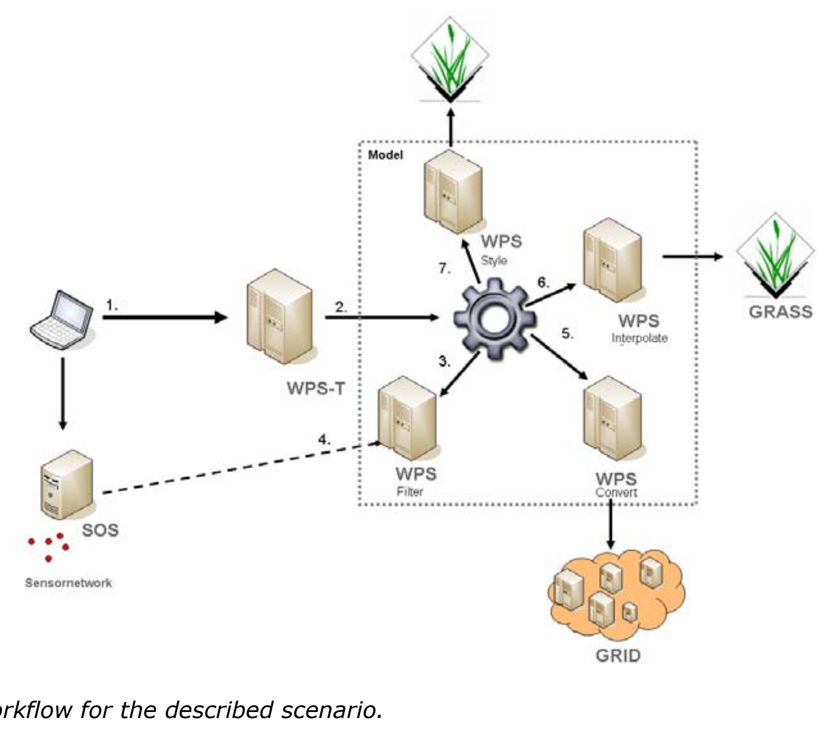

Fig. 4a *Deployed workflow for the described scenario.*

As the workflow analyses latest data in (almost) real-time and can be triggered on-demand, it allows the decision makers to monitor the current air quality and thereby to take actions accordingly based on the latest information in real-time. Additionally, it is possible to compare the phenomena over time and thereby get analyze its behaviour more thoroughly.

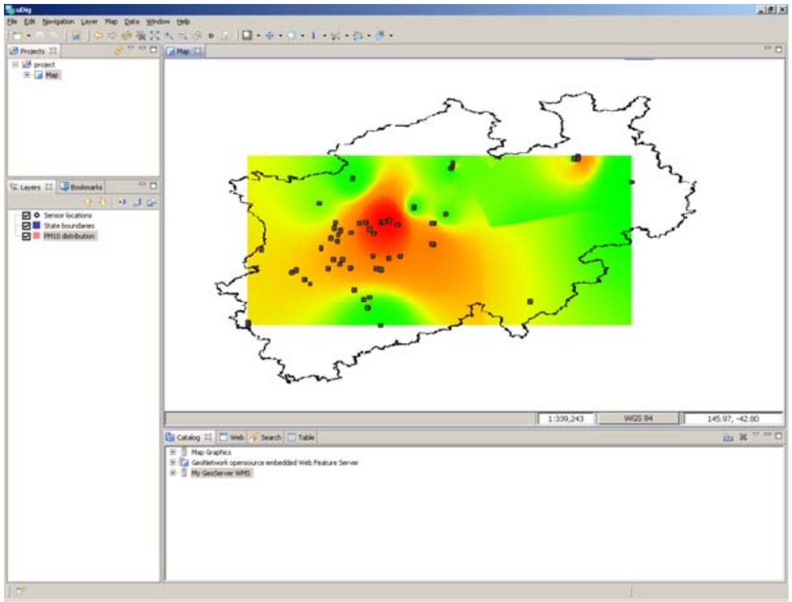

Fig. 4b *Workflow results visualized with states boundary of North-Rhine Westphalia and the sensor location. The red colored areas imply a high amount of PM10 (40 µg PM10 /m³ air), which violates current EU regulations.*

5. Discussion & Conclusion

This paper describes an open source framework for building geoprocessing workflows using real-time geodata based on distributed Web Services. First, existing geoprocessing functionality of stand-alone GIS is exposed as Web Services. Second, computational intense processes are delegated to a Grid Computing environment. Since both approaches do not change the standardized WPS interface, they can be easily applied in a geoprocessing workflow. Moreover, the framework includes a mechanism to expose geoprocessing workflows as single processes through WPS interface (WPS-T). All the mechanisms of the framework are utilized to realize a real-time scenario for air quality monitoring (Section 4).

Besides the general geoprocessing research topics elaborated in Brauner et al. (2009), this paper shows that wrapping geoprocessing functionality with a WPS interface is a promising concept to reuse existing and reliable functionality in a Web Service context. However the semantic gap between the user and the process description still remains. In particular, taxonomies for geoprocesses to classify and identify functionality and its parameters, independently of the GIS package, have to be created such as described in Lemmens et al. (2006). Using WPS Profiles as stated in the WPS specification could be a first step in this direction.

The presented approach of encapsulating distributed Grid Computing resources improves the availability by outsourcing computational tasks to a failsafe infrastructure and increases computational performance by using shared distributed computational abilities of common Grid Computing methods. But the encapsulated gridification approach results in two major problems. At first, not all geospatial problems can be divided into smaller sub-problems. At second, the dynamic stage-in of input data and application binaries is a bottleneck. The presented integrated gridification approach might be a feasible solution for such problems, but a lot of research work together with common standardization bodies has to be carried out. Still various theoretical and practical relevant questions remain unsolved (e.g. how to integrate OGC Web Services into stateful WSRF and how to handle different grid-specific security issues). From the authors' point of view, Grid Computing in general is the most promising technology to increase performance in large scale resource-intensive geoprocessing tasks.

The deployment of workflows using real-time data from distributed services allows the service provider to model such workflows only once but execute them anytime by a client application. This is highly relevant for complex workflows and significant for real-time risk management applications as demonstrated by the scenario. Additionally, the workflow can be integrated as another building block into a higher level workflow since it is exposed as a single WPS process (WPS-T). Finally, this results in nested workflows, which are again based on workflows. Even though in our case only WPS instances are incorporated, the presented approach for establishing workflows can be applied to any type of Web Service interface because the approach is based on the mainstream IT standard BPEL. Asynchronous processing and the use of references have to be taken into account to improve the performance of the designed workflows in the future.

References

Alameh, N (2003) Chaining Geographic Information Web Services. IEEE Internet Computing, 07:22-29, doi:10.1109/MIC.2003.1232514

Alonso G, Casati F, Kuno H (2003) Web Services. Concepts, Architectures and Application. Springer, Berlin

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

Brauner J, Schaeffer B (2008) Integration of GRASS Functionality in Web based SDI Service Chains. Academic paper at FOOSS4G 2009, Cape Town, South Africa

Brauner J, Foerster T, Schaeffer B, Baranski B (2009) Towards a Research Agenda for Geoprocessing Services. 12th AGILE International Conference on Geographic Information Science, Hannover, Germany

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, Seattle

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

Foster I, Kesselman C (1998) The Grid: A Blueprint for a New Computing Infrastructure. Morgan Kaufmann Publishers, San Francisco

Foster I, Zhao Y, Raicu I, Lu S (2008) Cloud computing and grid computing 360-degree compared. http://arxiv.org/abs/0901.0131. Accessed 26 May 2009

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

Kiehle C, Greve K, Heier C (2007) Requirements for Next Generation Spatial Data Infrastructures-Standardized Web Based Geoprocessing and Web Service Orchestration. Transactions in GIS 11(6):819–834, doi:10.1111/j.1467-9671.2007.01076.x

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, doi:10.1007/978-3-540-69848-7_17

Lemmens R, Wytzisk A, de By R, Granell C, Gould M, van Oosterom P (2006) Integrating Semantic and Syntactic Descriptions to Chain Geographic Services. IEEE Internet Computing 10:42-52, doi:10.1109/MIC.2006.106

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

OWS-6 (2009) OWS-6 WPS Grid Processing Profile Engineering Report. OGC 09-041 (Forthcoming)

Papazoglou MP, van den Heuvel W-J (2007) Service oriented architectures: approaches, technologies and research issues. The VLDB Journal 16 (3), 389-415, doi:10.1007/s00778-007-0044-3

Pelz, C (2003) Web services orchestration and choreography. IEEE Computer 36(8):46–52, doi:10.1109/MC.2003.1236471

Pope C, Burnett R, Thun M, Calle E , Krewski D (2002) Lung Cancer, Cardiopulmonary Mortality, and Long-term Exposure to Fine Particulate Air Pollution. Med. Assoc., 287:132-1141, doi:10.1001/jama.287.9.1132

Schaeffer B (2008) Towards a Transactional Web Processing Service (WPS-T). In: Pebesma E, Bishr M, Bartoschek T (ed.) Proceedings of the 6th Geographic Information Days June 16-18, 2008, Muenster, Germany, Institute for Geoinformatics, Muenster. 91-117

Schaeffer B, Foerster T (2008) A Client for Distributed Geo-Processing and Workflow Design. Journal for Location Based Services 2, no. 3 (September 2008): 194-210, doi:10.1080/17489720802558491

Scholten M, Klamma R, Kiehle C (2006) Evaluating performance in spatial data infrastructures for geoprocessing. IEEE Internet Computing 10, no. 5 (October 2006): 34-41, doi:10.1109/MIC.2006.97

Usländer T (ed) (2007) Reference Model for the ORCHESTRA Architecture. http://www.eu-orchestra.org/download.php?file=docs/RM-OA/RM-OA-V2-Rev-2.0.pdf. Accessed 26 May 2009

Weerawarana S, Curbera F, Leymann F, Storey T, Ferguson D (2006) Web services platform architecture : SOAP, WSDL, WS-policy, WS-addressing, WS-BPEL, WS-reliable messaging and more. 4. printing ed. Prentice Hall/PTR, Upper Saddle River, N.J.

van der Aalst W, van Hee K (2004) Workflow management : models, methods and systems, MIT Press, Cambridge

Weiser A, Zipf A (2007) Web Service Orchestration (WSO) of OGC Web Services (OWS) for Disaster Management. In: Li J, Zlatanova S, Fabbri A (ed) Lecture Notes in Geoinformation and Cartography:239–254 Springer, New York