Development of a WPS Process Chaining Tool and Application in a Disaster Management Use Case for Urban Areas

B. Stollberg
Department of Geography, University of Bonn, Germany

A. Zipf
Department of Geography, University of Bonn, Germany

ABSTRACT: The recently approved OGC Web Processing Service (WPS) specification defines a generic interface for providing geo-processing functionality in a standardized way. Web Services in general can be composed in service chains for carrying out more complex tasks. This paper presents an approach for offering a chaining tool for WPS processes in a simple as possible way. Basic functionalities can be combined in extensive workflows without touching any source code, just by creating a XML file according to our developed XML schema. A proof of concept is given by applying the developed implementation in a use case of disaster management for urban areas. In conclusion our implementation makes it easy for any GIS user to provide new complex processes by our developed framework.

1 INTRODUCTION

Today, Spatial Data Infrastructures (SDIs) are primarily used to share and access distributed spatial data. However, the usefulness of SDIs can be improved by developing Web Services that are also capable of processing the shared data in order to tailor the results to the needs of the user. In this context the Open Geospatial Consortium (OGC) has recently approved the Web Processing Service (WPS) 1.0.0 as an OpenGIS standard (Schut 2007). This standard defines a generic Web Service interface to data processing functionality.

In general Web Services provide functions that can be used in diverse applications. They are platform-independent, self-contained and self-describing (Karastoyanova et al. 2003). Consequently, an application does not need to know a service's implementation in order to integrate it. Therefore, interoperable Web Services are reusable within other software applications regardless of the programming language and the platform. Service chains can be set up according to the user's requirements and services implementing the same interface can be replaced by each other. By building service chains it is possible to let services cooperate. That means one service uses the result of another one and complex tasks can be solved by calling particular services sequentially.

This paper presents an approach to offer this kind of chaining functionality for WPS processes in order to carry out a complex task in a simple as possible way. The idea behind this approach is to define a plain and intuitive schema for extensive service chains in a consistent representation. Our ideas are described in the following, starting with a description of the WPS interface and an introduction to Web Service orchestration in general. This is followed by the description of our developed schema and the realized implementation. Furthermore we give an use case example in the field of disaster management for urban areas and we finish with a conclusion and outlook.

2 THE OGC WPS INTERFACE

The WPS standard is quite open and such a service may provide simple calculations (e.g. the calculation of a buffer) as well as complex computations (e.g. the generation of a climate model). Thus, in principle there are no restrictions on what can be implemented using the WPS interface. There are three mandatory operations that must be performed by a WPS, namely *GetCapabilities*, *DescribeProcess* and *Execute*. When a *GetCapabilities* request is made to the WPS, it must send back an XML document describing the service capabilities. This XML response must contain metadata about the service itself and the processes it provides. When a *DescribeProcess* request is made to the WPS, it must reply sending back an XML document describing one or more of the available web processes in detail. The document must specify the input parameters and formats for executing a specific process and the output format of the process result. Finally the process is executed when an *Execute* request is sent to the service.

As the specification allows any kind of geoprocessing functionality it can be applied to a wide variety of domains and tasks. Some examples include disaster management (Stollberg & Zipf 2007), forest fire analysis (Friis-Christensen et al. 2007), generalization (Foerster & Stoter 2006), hydrological models (Diaz et al. 2008), housing market analysis (Stollberg & Zipf 2008), biodiversity research (Graul & Zipf 2008), time series analysis (Gerlach et al. 2008), precision farming (Nash et al. 2008) etc.

3 WEB SERVICE ORCHESTRATION

An established standard concerning Web Service Orchestration (WSO) is the use of the Business Process Execution Language (BPEL) which uses the Web Service Description Language (WSDL). WSDL is a standard for the description of a Web Service interface and acts as the link between the Orchestration Engine (OE) and the services involved. The WSDL interface offers the possibility of providing a defined functionality for a client where the underlying implementation is not transparent.

As the WPS standard defines the service itself but not the underlying processes provided by the implementation this means in consequence that there is no predefined WSDL description for a WPS. Rather a process developer is forced to define a WSDL description of each process together with a definition of the *DescribeProcess* document. In the case of a provided WSDL document for a WPS process it is then possible to address this process by means of BPEL and integrate it within a chain of services. This implies that an Orchestration Engine is set up which is able to execute the chain in turn.

BPEL and WSDL are mainstream IT standards and not yet well established within the Geographic Information (GI) community. The use of these standards is at the moment burdened with a lot of overhead (defining WSDL documents by hand, setting up and using an OE). For this reason we developed another approach by simple means for the easy chaining of WPS processes within the GI community.

4 DEVELOPMENT OF A SERVICE CHAIN SCHEMA

The idea behind our approach is the development of a simple XML schema defining a service chain. Such a service chain can then be presented according to the schema by a single XML document. This document in turn can be interpreted by a suitable implementation which is able to parse it and execute the involved processes in the defined order.

For the development of such a schema we made the considerations listed in the following section. It has to be pointed out that a WPS instance is able to provide several processes. For this reason a single service within a chain is synonymous to a WPS process. This means any service equates with any process of any WPS instance. In the following we are speaking of services for simplifying matters.

The following considerations were made:

- Each service chain consists of several services
- Each service has a web address (WPS Server) and an identifier (as a WPS can provide more than a single process)
- Each service needs a unique identifier within the service chain
- Each service defines any number of inputs and outputs
- Each input and output of a service has an identifier
- Each input of a service can be a predefined literal value, a predefined web-accessible resource providing complex spatial data (e.g. the address of a Web Feature Service (WFS) layer), a value which is passed within the service chain request or the output of a previously executed service within the chain (in this case the unique identifier of this service in the chain is required besides the corresponding identifier of the output of this service)
- The complete service chain consists of any number of outputs
- Each output of the service chain has an identifier
- Each output of the service chain is the output of a specific executed service within the chain and can be defined by the unique identifier of this service and the identifier of the output of this service

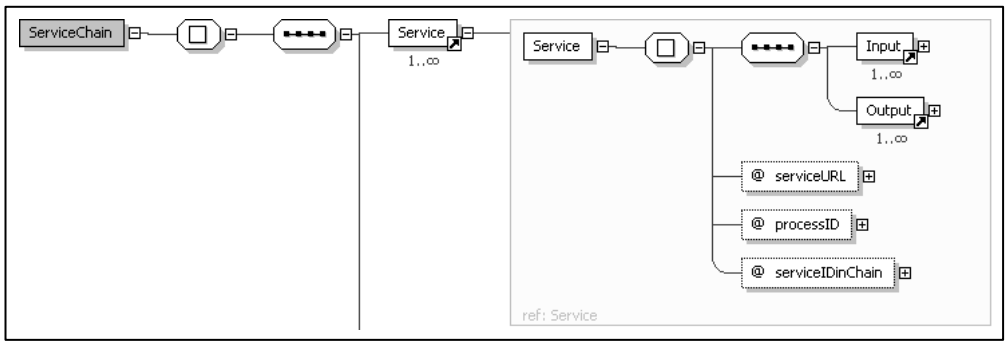

Figure 1. The element *Service* within the developed *ServiceChain* schema.

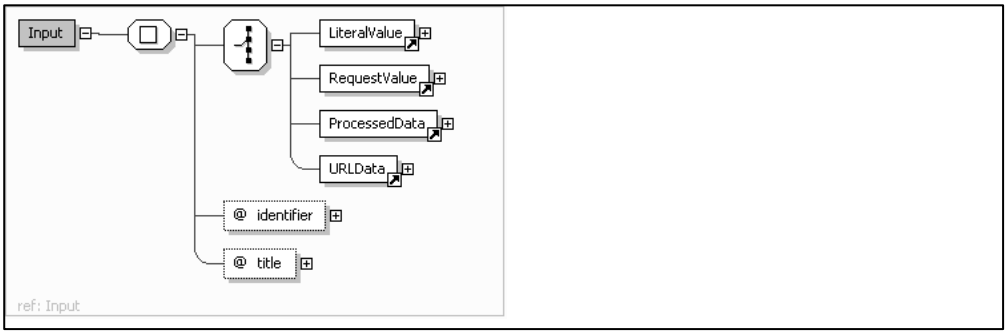

Figure 2. The element *Input* within the developed schema.

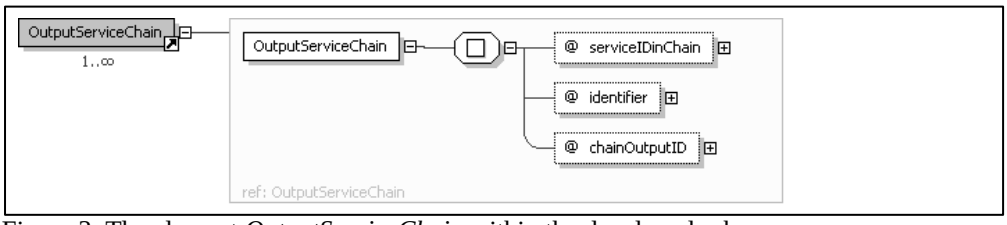

Figure 3. The element *OutputServiceChain* within the developed schema.

5 EXAMPLE SERVICE CHAIN IN A DISASTERMANAGEMENT USE CASE

The WPS *ServiceChain* implementation can be used in a wide area of applications and independent from a specific domain. Any functionality which is available as a WPS process (on any server instance) can be integrated into a more complex workflow. For proving our concept, we want to give an example in the field of disaster management in urban areas. Apart from this example application we want to emphasize that the concept is not confined to a specific theme but applicable in a wide range of domains.

In our use case we assume that a gas leakage was detected in a densely populated area. The toxic gas, which is a threat to human beings, is spreading. Furthermore we assume that there is a shared network of gauging stations available which provides measured values of air pollution within this area. The user in our scenario might be a staff member of the disaster management department of the affected city. He has to initiate further actions and has to answer three specific questions for this purpose:

- Are there any gauging stations providing real-time information about air pollution close to the location of the leakage?
- Approximately how many people have to be evacuated close to the location of the leakage?
- Are there any evacuation shelters close to the affected area where the evacuees can be safely assembled together?

All three questions imply several GIS operations and each can be provided as a new complex process. For answering the first question of close-by gauging stations two steps have to be executed. In the first step a buffer operation around the gas leakage location is carried out. This is followed by a polygon-contains-point analysis as a second step. Thereby the result of the first operation presents the input for the second operation within the workflow. The scenario is illustrated in figure 4. The complex process described above can be expressed by means of our *ServiceChain* schema as presented in figure 5.

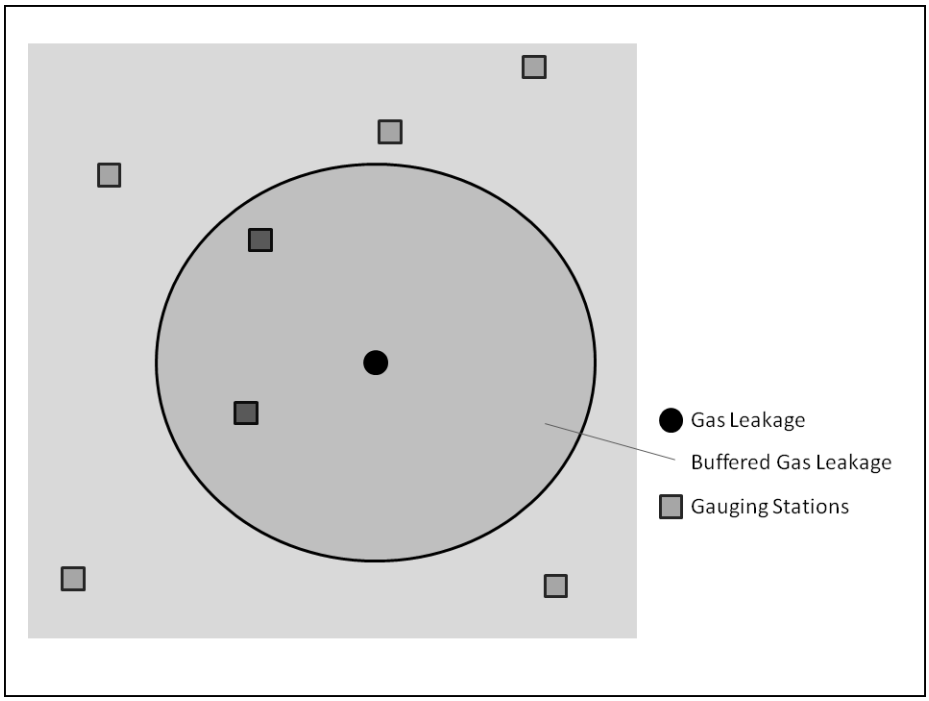

Figure 4. Finding gauging stations within a specific buffer around a gas leakage location.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <ServiceChain xmlns="http://www.geographie.uni-bonn.de/karto/wps/servicechain"
3    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4    xsi:schemaLocation="http://www.geographie.uni-bonn.de/karto/wps/servicechain/ServiceChain.xsd">
5    <Service serviceURL="http://karto.giub.uni-bonn.de:8080/deegreeWPS/all" processID="Buffer" serviceIDinChain="Service1">
6      <Input identifier="InputGeometry" title="Input Geometry to be buffered"><RequestValue identifier="InputGeometry"/></Input>
7      <Input identifier="BufferDistance" title="Buffer Distance"><RequestValue identifier="BufferDistance"/></Input>
8      <Output identifier="BufferedGeometry" title="BufferedGeometry"/>
9    </Service>
10   <Service serviceURL="http://karto.giub.uni-bonn.de:8080/deegreeWPS/all" processID="PolygonContainsPoints" serviceIDinChain="Service2">
11     <Input identifier="PolygonFeatures" title="PolygonFeatures"><ProcessedData serviceIDinChain="Service1" identifier="BufferedGeometry"/></Input>
12     <Input identifier="PointFeatures" title="PointFeatures"><RequestValue identifier="PointFeatures"/></Input>
13     <Output identifier="JoinedFeatures" title="JoinedFeatures"/>
14   </Service>
15   <OutputServiceChain serviceIDinChain="Service2" identifier="JoinedFeatures" chainOutputID="PointsInPoly"/>
16 </ServiceChain>
```

Figure 5. Service Chain example of finding gauging stations within a specific buffer by means of a XML document according to our developed schema.

The complex process consists of two services: *Service1* is defined in lines 5-9 within the XML document depicted in figure 5 and *Service2* is defined in lines 10-14. *Service1* takes two input parameters, namely the *InputGeometry* which is in our scenario the location of the gas leakage and secondly the desired *BufferDistance*. Both input parameter will be part of the request which has so be sent in order to execute the complex process. Furthermore *Service1* creates one output, namely the *BufferedGeometry*. *Service2* takes again two input parameters, namely *PolygonFeatures* which is in our scenario the buffered gas leakage location and respectively the output of *Service1*. For this reason the input parameter is set as *ProcessedData*. The second input parameter of *Service2* is *PointFeatures.* This is a dataset of all available gauging stations of the city. Furthermore *Service2* has one output called *JoinedFeatures*. These *JoinedFeatures,* which are the gauging stations located within the buffer of the gas leakage, are also the desired output for the complete complex process. This is expressed within the XML tag *OutputServiceChain*.

After deploying this complex process by means of our developed *ServiceChain* schema the new WPS process is available within our framework and the user is able to execute the process at any time with the relevant input parameters (gas leakage location, width of the buffer to calculate, location of the gauging stations dataset). It is now not necessary anymore for the user to execute the two separate processes one after another but he has the ability to carry out both steps within one operation. The result is a number of gauging stations within the specified buffer around the gas leakage location. These stations might deliver real-time data about air pollution and give information about how far the gas is spreading. Furthermore these stations might provide measured values about wind speed and wind direction for estimating how the gas will spread within the next hours.

The second question the member of the disaster management department has to answer is the number of people which have to be evacuated within a certain radius around the gas leakage. In order to calculate this number, population numbers are available on a building block level. This building block information has to be clipped with a buffer around the gas leakage location and the population information has to be summed up. This is achieved by the two processes *Buffer* and *PolygonIntersectsPolygonJoinAggreation* which are both available as independent processes and can be combined within a new complex process for fulfilling the task of the disaster management department.

The third question is dealing with the finding of shelters close to the area where the people have to be evacuated. Therefore four steps are necessary: First of all a *Buffer* operation around the gas leakage location has to be carried out. This buffer is defined as the area in which a possible shelter should be located and by the second step of a *PolygonConatinsPoints* analysis

these shelters are identified. As the final chosen shelter is not allowed to be located within the evacuation area another *Buffer* operation has to take place expressing the area where the people have to be evacuated. In the last step this buffer is together with the found shelters input for a *PointDisjointPolygon* analysis. The result is then a dataset of shelters located outside of the buffered area. The user of this complex process is now able to choose an appropriate shelter and arrange for the evacuation to be done.

We want to mention that the use of a buffer operation for identifying possible shelters close to the gas leakage is a quite simplified approach. Given that a low distance to an object does not imply the fast accessibility we would have in practice to pose a question like: Are there any evacuation shelters that can be reached within 20 minutes from the affected area? It becomes clear that a simple buffer operation is not sufficient enough. The accessibility of an object depends on the road network and not on the simple distance. So in practice the first buffer operation would be replaced by an accessibility analysis. Such an analysis was for example realised within an Accessibility Analysis Service (AAS) based on a service for routing (Neis & Zipf 2007).

6 IMPLEMENTATION AND OPEN ISSUES

For the implementation of our approach we use the existing WPS framework of the *deegree2* project (Fitzke et al. 2004). The framework offers the possibility to implement the application logic of new processes within a separate class. This class is derived from the *org.deegree.ogcwebservices.wps.execute.Process.java* class. In addition an XML configuration document defining process identifier, input and output parameters etc. has to be created and integrated into the framework along with the process class. After that the new process is accessible via the WPS interface. For our *ServiceChain* schema approach we implemented a *Chain* class which provides the process class for each complex service chain according to our schema. This class is parsing the associated defined *ServiceChain* XML file and all included services are requested in a sequential manner.

The implementation makes it therefore possible to provide any defined complex service chain by integrating two XML files: One defining the chain according to our schema and another one describing the resulting process itself including identifier, input and output parameters etc. This means that any complex process can be easily provided by our framework without touching any source code, just by integrating two XML files.

The next step will be the implementation of a web client which will make it possible to define such complex chains with a Graphical User Interface (GUI). The “ordinary” GI user will then be able to group his needed processes to such a chain through our GUI, register the chain at our WPS server and perform the new process through an Execute request.

An open issue of our developed implementation is the exception handling. Further work has to be done to ensure that the occurrence of an exception within the whole chain of processes can be clearly assigned to a specific process. Up to now it is quite hard to identify at which point an exception takes place and a mechanism has to be found which shows the user which process was not finished successfully. At the moment it is only obvious for the user that executing the complete chain was not successful but not which process is responsible for the exception.

Another open issue is the security aspect while offering new processes within the framework. The idea is to allow the deploying of new complex processes for any user. In consequence these new processes are available on the server and can be executed by everyone. This might be against the interest of a specific user who does not agree with the fact that anybody is able to execute his deployed processes. Preferable would be a solution where every user is only able to execute his “own” processes. This is especially interesting if a process is tailored to a specific dataset which is not free for the public. Services within our *ServiceChain* can take *URLData* as input parameter. This implies that the new complex process is always executed on this data. For this reason it would be preferable to develop a security concept including authentification/authorisation mechanisms in the future.

7 CONCLUSION AND FUTURE WORK

Modularization and the reuse of developed modules are the key to Service Oriented Architectures. One possibility for the orchestration of services is the well-know BPEL standard which allows the execution of complex workflows. But applying BPEL implies the use of an Orchestration Engine and in the case of the OGC WPS specification it requires the manual creation of WSDL documents for each process. These are two disadvantages of using BPEL for an “ordinary” GIS user who is not familiar with these mainstream IT standards.

In this paper we presented an approach to offer this kind of chaining functionality in a simple as possible way. The idea behind this approach was to define a plain and intuitive schema and offering the possibility to chain WPS processes just by creating a simple XML document according to the schema describing the service chain. Therefore we used the WPS interface itself for the orchestration of processes. The developed WPS ServiceChain implementation makes it easy to provide new complex processes by our framework. Basic functionalities can be combined in extensive workflows without touching any source code, just by creating an XML file along with an XML description of the new process.

Our future work will focus on the implementation of a web client which will make it possible to define such complex chains through a GUI. With the help of this client any GIS user will be able to easily compose his required processes in a chain, register the chain at our WPS server and perform the new process through an Execute request. But previously work has also to be done concerning exception handling and security issues as described in the chapter above.

However, the WPS interface closes an open gap in the field of OGC Web Services as it is now also possible to process spatial data in a standardized way. A disadvantage of the interface is the missing definition of well-know geo-processing operations. The specification is quite open and allows the provision of processing functionality in general, no matter if “geo” or not. For this reason profiles are introduced within the specification which shall define a well known amount of geo-processing functions. These profiles have to be developed by user communities which agree upon specific processes relevant to the applying domain. Not before a WPS process *Buffer* of provider A takes the same input and output parameter as the *Buffer* process of provider B another level of service interoperability will be reached.

REFERENCES

Díaz, L., Granell, C. and Gould, M. 2008 in press. Case Study: Geospatial processing services for web-based hydrological applications. In: J.T. Sample, K. Shaw, S. Tu and M. Abdelguerfi (Eds.): Geospatial Services and Applications for the Internet. Springer, 2008.

Fitzke, J; Greve, K; Müller, M. and Poth, A. 2004. Building SDIs with Free Software - the deegree Project. In: Proceedings of GSDI- 7, Bangalore, India.

Foerster, T. and Stoter, J. 2006. Establishing an OGC web processing service for generalization processes. Workshop ICA Commission on Map Generalisation and Multiple Representation. Portland, USA.

Friis-Christensen, A., Lutz, M., Bernard, L. and Ostländer, N. 2007. Designing Service Architectures for Distributed Geoprocessing - Challenges and Future Directions. Transactions in GIS, 11(6): 799-818.

Gerlach, R., Schmullius, C. and Nativi, S. 2008. Establishing a web processing service for online analysis of earth observation time series data. Geophysical Research Abstracts 10.

Graul, C. and Zipf, A. 2008. Putting Biogeographic Research on the Spatial Web: Towards Interoperable Analysis Tools for Global Change Based on the Web Processing Service (WPS). Digital Earth Summit on Geoinformatics. Potsdam, Germany.

Karastoyanova, D. and Buchmann, A. 2003. Components, Middleware and Web Services. Proceedings of the IADIS International Conference WWW/Internet, ICWI 2003, Algarve, Portugal.

Nash, E., Korduan, P. and Bill, R. 2007. Optimising data flows in precision agriculture using open geospatial web services. In Stafford, J. V. (ed.): Precision Agriculture '07: Proceedings of the 6th European Conference on Precision Agriculture, Wageningen, pp. 753-759.

Neis, P. and Zipf, A. 2007. A Web Accessibility Analysis Service based on the OpenLS Route Service. The 10th AGILE 2007 Conference on GI Science. Aalborg, Denmark.

Schut, P. 2007. OpenGIS Web Processing Service. Version 1.0.0. OGC 05-007r7.
Stollberg, B. and Zipf, A. 2007. OGC Web Processing Service Interface for Web Service Orchestration - Aggregating Geo-processing Services in a Bomb Threat Scenario. In: Web and Wireless Geographical Information Systems (W2GIS 2007). Cardiff, UK.
Stollberg, B. and Zipf, A. 2008. Geoprocessing Services for Spatial Decision Support in the Domain of Housing MarketAnalyses - Experiences from Applying the OGC Web Processing Service Interface in Practice. The 11th AGILE 2008 Conference on GI Science. Girona, Spain.