OGC Web Processing Service Interface for Web Service Orchestration

Aggregating geo-processing services in a bomb threat scenario

Beate Stollberg[1] and Alexander Zipf[1]

[1] i3mainz - Institute for Spatial Information and Surveying Technology,
University of Applied Sciences FH Mainz,
Holzstraße 36, Mainz, Germany

{stollberg, zipf}@geoinform.fh-mainz.de

Abstract. The aggregation of web services in order to achieve a common goal is a basic concept in *Service Oriented Architectures* (SOA). This paper demonstrates *OGC Web Services* (OWS) aggregation based on a simulated bomb threat scenario which is a possible disaster management example. Web services are aggregated based on the OGC *Web Processing Service* (WPS) interface. Here the concept of a "Composite-WPS" is introduced and used to invoke all other services involved. The presented approach is compared to the use of BPEL which is an orchestration language with some technical pitfalls when applied to the orchestration of OWS.

Keywords: geo-processing, geographical web services, orchestration, Web Processing Service (WPS), service chain, composition, SDI, disaster management

1 Introduction

Spatial Data Infrastructures (SDIs) support discovery, access, and use of geographic information in the decision-making process [1]. Traditionally, *Geographic Information Systems* (GIS) have been used for the extraction of information from spatial data in order to answer spatial questions. Today, the transition from the traditional monolithic GIS to an interoperable *Service Oriented Architecture* (SOA) is taking place. Increasingly research on the consequences of this paradigm shift is carried out, e.g. in resolving some of the key interoperability problems among GI services [2] or the chaining of these services [3,4]. SDIs are based on the assumption that users are usually not interested in data but in a piece of information that can be generated by Geographic Information (GI) services which are main components of SDIs [5]. Therefore, Spatial Data Infrastructures provide an important basis in the field of disaster management where geographic information must be discovered, processed and visualized quickly to provide critical assistance in emergencies and in support decision makers and rescue workers [6].

This is also one of the main goals within our project "OK-GIS". OK-GIS is the acronym for "Open Disaster Management using free GIS" (http://www.ok-gis.de). Within OK-GIS we aim at the development of a flexible toolbox for the administration, application, visualization and mobile creation of geodata using standard based services in a disaster management environment. Basic principles are derived from the *Open Geospatial Consortium* (OGC) standardized *OGC Web Services* (OWS) such as *Web Map Service* (WMS), *Web Feature Service* (WFS), *Web Coverage Service* (WCS), and *Catalog Service for the Web* (CSW). Furthermore, OK-GIS incorporates services according to the OpenLS OGC initiative, including the *Route Service* (RS) [7], which was extended into a 3D Emergency Route Service (3D-ERS) [8]. The *OpenLS Presentation Service* and the OpenLS *Utility Service* (Geocoder / Reverse Geocoder) were also implemented and the *OpenLS Directory Service* is currently under development.

Visualization, vector and raster data access, along with the ability to search for spatial data is mostly covered by the standards mentioned above. However, sophisticated and recognized standards for distributed spatial data processing, leading to user specific information preparation, are still missing [9] and were taken into consideration in an OGC discussion paper that recommended a *Web Processing Service* (WPS) specification via a standardized interface.

The described WPS interface specification is also used within the OK-GIS project for realizing example scenarios in the field of disaster management. This will be demonstrated in this paper on a specific example - finding a bomb. A first version of this bomb scenario which was based on [10] has already been discussed by [11,12] concerning the orchestration of existing OWS using the *Business Process Execution Language* (BPEL). In this article the scenario was elaborated and focuses on the geo-processing aspect using the OGC WPS interface. While [11,12] only presented this aspect in form of a "black box" through calls of generic WPS processes that were specified in detail, we will discuss the interaction of several processes and present the implementation of the scenario using the existing WPS framework [13] of the *deegree* project (http://www.deegree.org). Since [12] uncovered some technical problems when chaining OCG Web Services with BPEL we will introduce and discuss a new approach for composing OWS without an orchestration language but rather by using the WPS interface itself.

2 OGC Web Processing Service

The idea behind a WPS is to offer any kind of GIS processing functionality. It 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 based on the WPS interface.

A WPS is able to handle more than a single process and there are three mandatory operations performed by a WPS, namely *GetCapabilities*, *DescribeProcess* and *Execute*. The response to a *GetCapabilities* request is an XML-document containing metadata of the WPS and all available processes. A detailed process description as well as input and output parameters are provided for every process as response to a

DescribeProcess request, also in form of a XML-document. The final process execution is carried out when an *Execute* request is send to the WPS.

3 Example: Search for Adequate Evacuation Shelters in a Bomb Threat Scenario

In our scenario a bomb was found. This can be either an "old" bomb from World War II or a “ticking” bomb from a current attack. All persons within a specified radius around the bomb, namely within the “danger zone”, must be evacuated. A second, larger buffer must also be defined around the bomb, namely the “security zone”: Within this security zone only persons inside buildings of special interest (such as kindergartens, schools, etc.) will be evacuated. The scenario is illustrated in figure 1.

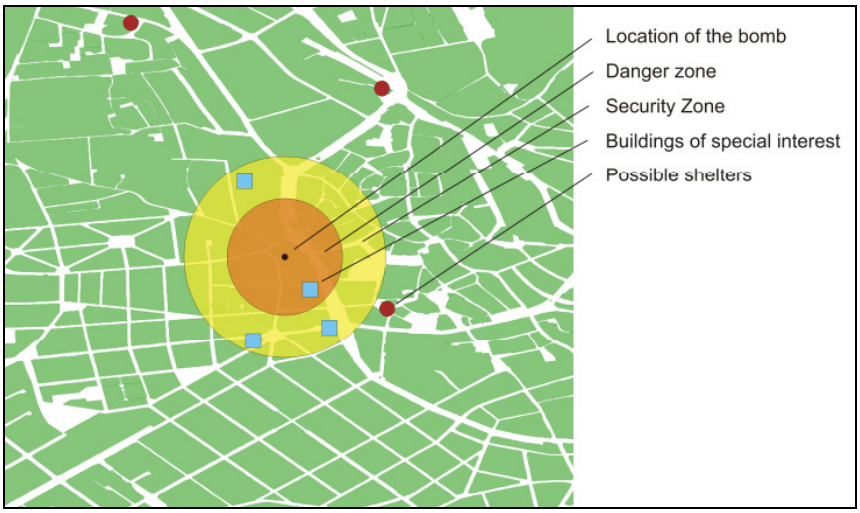

Fig. 1. Controlled Areas around a discovered Bomb

First the danger and the security zones must be defined. This is done by using the well-known GIS base function "Buffer", which is available as a WPS process. In order to calculate the number of persons to be evacuated, population numbers are available per building block. The building block information has to be clipped with the danger zone and then the population information has to be summed up. This is achieved by using the WPS process *PolygonIntersectsPolygonJoinAggregation*. Additionally the number of persons within the security zone who are inside buildings of special interest must be calculated. These building and respectively the number of people inside are provided by a WFS. Using the WPS process *PointInPolygonJoin* all the buildings of special interest inside the security buffer zone are determined. Then, by using the WPS process *Aggregation*, the number of persons inside those buildings is calculated [14] (we assume here that the needed data is available or can be found through a search in the CSW and retrieved from a WFS). On top of this, a WFS

provides information on possible shelters, where the evacuees can be safely assembled together, with all relevant attributes for these shelters such as address and building capacity in order to find the best suitable shelter. A buffer around the bomb area is calculated and then a decision can be made where to send the affected people by determining which shelters are closest for the calculated number persons seeking refuge. If no shelter is found, the bounding box is automatically extended until at least one appropriate shelter is found. In case there is more than one shelter within the bounding box, a route will be calculated between the location of the bomb and all possible shelters by an OpenLS RS. The resulting shortest route is then suggested to the decision makers. Finally all routes between the buildings of special interest and the allocated shelter are calculated by an *Emergency Route Service* (ERS) [15]. The ERS is a RS that conforms to the OpenLS specification but automatically by-passes restricted zones or streets. These restrictions need to be made available within the disaster management SDI, e.g. by a WFS, and automatically considered by the ERS. Here the danger zone is defined as an area to be avoided and is by-passed in the route calculation.

For representing the whole bomb scenario as a single application all of the listed services (WPS, RS, ERS, WFS) must be combined to one aggregated service, also referred to as a composition. For the description of business processes in general the orchestration model can be used. The background of *Web Service Orchestration* (WSO) and the option of using the WPS interface itself as possible alternative of using the BPEL standard is presented in the following sections.

4 Orchestration of OGC Web Services

The aggregation of Web Services is the composition of a set of services to achieve a common goal [15]. The possibility of compositing different services is often perceived as the greatest value of the web service paradigm [16] and it is a central concept in the framework of *Service Oriented Architectures* (SOA). Various complex tasks can be solved by compositing and reusing several “simple” services. In order to represent the complete process logic orchestration can be used which means that the sequence and implementation terms for activating external or internal services is presented through the eyes of a single participant [17].

4.1 Using BPEL for the Orchestration of OGC Web Service

An established standard concerning WSO is the use of 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 [12]. The WSDL interface offers the possibility of providing a defined functionality for a client where the underlying implementation is not transparent. This leads to one of the problems related to the use of BPEL for the orchestration of OGC Web Services, namely the lack of WSDL documents describing the OWS. These documents have to be created manually for every service used in order to orchestrate them using BPEL. Furthermore the *Simple Object Access*

Protocol (SOAP) support is missing in OWS. At the moment OGC Web Services are invoked via HTTP POST and/or GET which is not supported by all Orchestration Engines tested or at least not directly supported. The OGC is aware of these problems and there are ongoing discussions which will end in concrete solutions in the near future.

On the other hand there is also a technical problem with the OE and the transfer of raw binary data. This kind of data is served in response to a WMS *GetMap* or WCS *GetCoverage* request which cannot for this reason be orchestrated using the BPEL approach. All these difficulties are described in detail in [12]. The mentioned pitfalls forced us to look for alternatives and lead us to the use of the WPS interface itself which is an alternative to the *World Wide Web Consortium* (W3C) compliant WSDL interface in OWS architectures.

4.2 Using the WPS Interface itself – Three different Approaches

The OGC WPS interface provides the technical possibilities for a orchestration of web services, although this may be in contrast to the original WPS idea. Therefore our focus is on the implementation and aggregation of WPS processes. At the same time, the possibility of using an WPS interface to aggregate complex geo-processing services is investigated. In principle there are no restrictions on what can be implemented using the WPS interface. The specification provides plenty of room for interpretation because it does not describe the problem clearly. The specification primarily describes the concrete implementation of geo-processing methods and does not entirely exclude its use as an orchestration service. Therefore it is possible to use a WPS for aggregating participating OWS. The interface is specified in such a way that any kind of processing can be hidden, if desired.

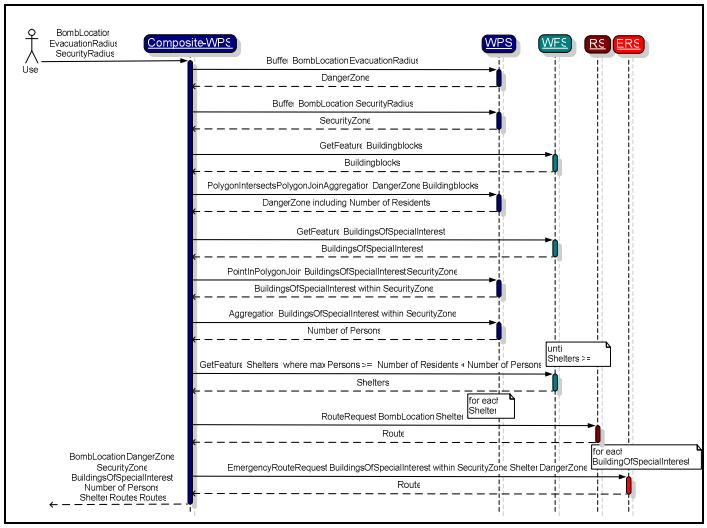

Fig. 2. Composite-WPS calling all other Services in the Bomb Threat Scenario

In the first approach of the implementation of our scenario we define a "Composite-WPS" as a WPS that can serve as an orchestration service for activating other services. The Composite-WPS activates all OWS needed for our scenario, including other WPSs as well as RS, ERS (which is an RS with special behavior) and WFS according to the defined application logic. The Composite-WPS can also be addressed as an independent service. The sequence diagram of the bomb threat scenario is presented in figure 2.

Such a solution requires large amounts of data to be passed between the different services. E.g. the security zone is calculated by the WPS process *Buffer*, sent back to the Composite-WPS and then sent again to the process *PointInPolygonJoin*. This can be avoided by calling specific services directly from other services. The *PointInPolygonJoin* process calls the *Buffer* process directly and the *Aggregation* process then calls the *PointInPolygonJoin* process. This possibility is explicitly enabled through the WPS interface by anticipating a *Key-Value-Pair* (KVP) encoded *Execute* request transported by the HTTP GET method. The use of "nested" KVP encoded *Execute* requests is a second option for the implementation of our bomb threat scenario and is shown in figure 3 for a part of the whole scenario.

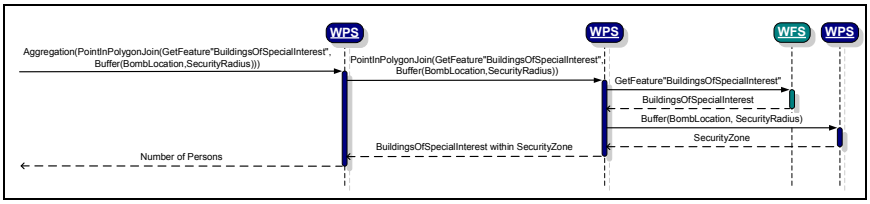

Fig. 3. Example of possible nested Requests in the Bomb Threat Scenario

These two alternatives show that there are in general two different concepts concerning the chaining of services: „*Centralized Service Chaining*" and „*Cascading Chaining*" according to R. Lucchi [personal communication, May 07, 2007]. *Centralized Service Chaining* means that a single central service invokes all other services, one after the other and controls the entire work flow. This approach is state-of-the-art and supported by common technologies. On the other hand, the *Cascading Chaining* method exchanges data directly because the individual services communicate with each other. Both concepts are presented in figure 4 and figure 5.

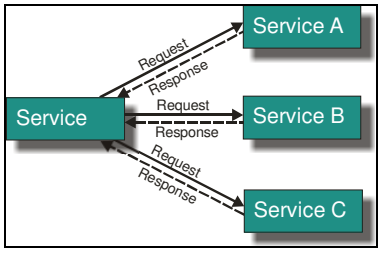

Fig. 4. Illustration of the Centralized Service Chaining Concept

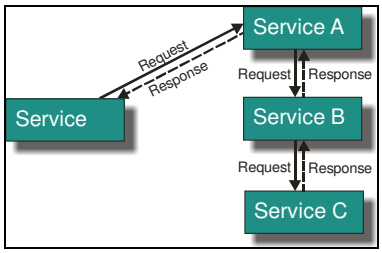

Fig. 5. Illustration of the Cascading Service Chaining Concept

The WPS interface is one of the few specifications which support *Cascading Chaining* and the concept is described by looking closer at a KVP encoded *Execute* request and calling other services out of a called service. We assume for our simplified bomb threat scenario that the location of the bomb is stored in a GML (*Geograpy Markup Language*) encoded file "bomblocation.gml" which is accessible at a server and the danger zone will be calculated for a radius of 1000 m around this bomb location. Here is a possible implementation of the KVP encoded *Execute* request for invoking the WPS process *Buffer*:

```
http://localhost:8080/wps?
request=Execute&
service=WPS&
version=0.4.0&
Identifier=Buffer&
DataInputs=
InputGeometry,http://localhost:8080/bomblocation.gml,
BufferDistance,1000
```

The information about buildings of special interest (e.g. schools, kindergartens etc.) is provided by a WFS and can be requested in the following way:

```
http://localhost:1979/geoserver/wfs?
service=wfs&
version=1.0.0&
request=GetFeature&
typename=buildings_of_special_interest
```

These two KVP encoded requests deliver the data inputs for the WPS process *PointInPolygonJoin*, namely the buildings of special interest (point data) and the danger zone (polygon data). Both can be directly integrated in the request for invoking the WPS process *PointInPolygonJoin* which could then look like that:

```
http://localhost:8080/wps?
request=Execute&
service=WPS&
version=0.4.0&
Identifier=PointInPolygonJoin&
DataInputs=
PointFeatures,
http://localhost:1979/geoserver/wfs?
service=wfs&
version=1.0.0&
request=GetFeature&
typename=os:buildings_of_special_interest,
PolygonFeatures,
http://localhost:8080/wps?
request=Execute&
service=WPS&
version=0.4.0&
Identifier=Buffer&
DataInputs=InputGeometry,http://localhost:8080/bombloca
tion.gml,
BufferDistance,1000
```

This complete request could then be integrated into an WPS *Execute* request to invoke the process *Aggregation.*

Such nested requests become quite complex, however the advantage is obvious: data is requested were it is processed and not passed unnecessarily. When large amounts of GML encoded data are passed around, a performance increase is expected. On the other hand, it is more difficult to trace the outputs of the various services which were called. Therefore, tracking errors is difficult because they are not easily matched to a particular service.

KVP encoded *Execute* requests are optional according to the WPS interface specification and unfortunately, to the best of our knowledge, are not yet implemented within the existing *deegree* WPS framework nor any other framework.

A third approach would be to combine multiple WPS functionalities into a single implementation by calling only classes and the respective methods of the classes in order to run the complete workflow. The whole functionality would be exposed as a single WPS process but at the same time the single building blocks are still available externally, as independent processes. This alternative is purely a question of implementation and shown in figure 6.

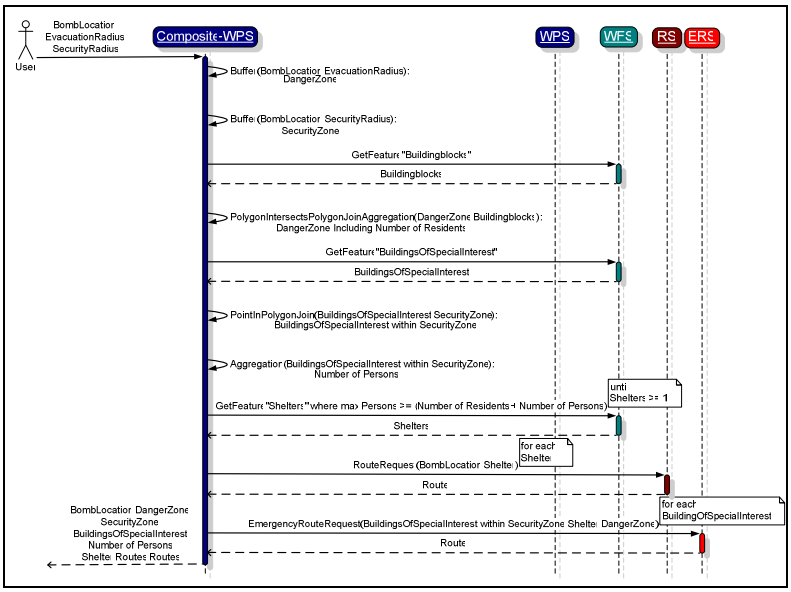

Fig. 6. A third Alternative of invoking services in the Bomb Threat Scenario

The advantage of this alternative is the performance boost because no data is exchanged between individual services more than once. The disadvantage is that it is not possible to replace WPS process instances by other instances which offer the same functionality and this does not make sense in terms of a distributed architecture. Thus, it is assumed that only one WPS instance is used to provide the whole scenario functionality as even though another service provider may have developed a different WPS instance with e.g. a process *Buffer* which is faster and achieves a higher accuracy. It would not be possible to address this WPS process within the third presented alternative. In reality however, the described scenario should use distributed WPS processes provided by different organizations. And this would be possible in the both alternatives described above, namely introducing a Composite-WPS or using nested KVP encoded requests. In these two approaches it is quite easy to replace WPS instances and this demonstrates the strength of distributed architectures.

4.3 Towards a Composite-WPS

Currently, we are implementing the scenario by using a Composite-WPS that invokes all other services and respective processes in our scenario in a sequential manner. This is in accordance with the *Centralized Service Chaining* concept. We did not inspect the concept of *Cascading Service Chaining* further because the support of KVP encoded *Execute* requests is at the moment still missing in the existing WPS implementations as mentioned above. We are demonstrating successfully that it is possible to represent complex scenarios completely in accordance with OGC standards, which was our main aim in the first place. The question which of the three approaches described above is preferable for a particular scenario cannot be answered

in general. It depends on the number of services involved and especially on the amount of data in every single case.

However, all three presented alternatives offer the possibility to provide the final application as an independent OGC compliant service through the WPS interface. In OK-GIS, these aggregated services are called "virtual services". In comparison to the generic OGC services, the idea within OK-GIS is, that these “virtual services” are scenario or domain specific similar to the described bomb threat scenario. They represent the end-user view who is not interested in technical implementations of OGC service chains, but rather on the task to be performed in order to manage the disaster situation.

Furthermore, it would be possible to create a WSDL document describing this domain specific service in order to achieve W3C compliance. This may be relevant for a developer who is not familiar with OWS and who does not expect an OGC compliant interface but instead, a in the general IT community well established standard like WSDL.

4.4 Using an Accessibility Analysis Service

A part of the described service chain, namely the finding of an optimal evacuation centre could also be realized by using the *Accessibility Analysis Service* (AAS) [18]. This would avoid re-implementing this part of the service chain and we shortly discuss this alternative even though the AAS is not a standardized OWS or even a discussion paper, but its interface is modeled according the OpenLS specification. Therefore it has to be initially hidden behind a WPS facade for being OGC conform.

Based on a street network and a defined time distance, the AAS calculates accessible areas and other information.

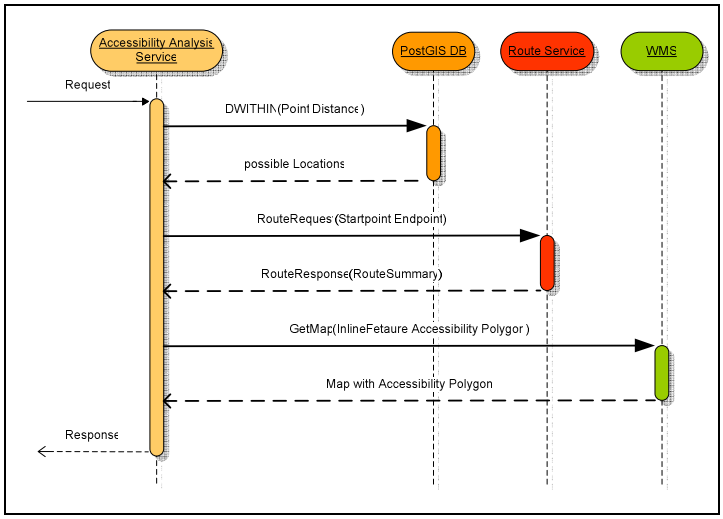

Fig. 7. UML Sequence Diagram of the AAS interacting with other Services

For using the AAS in the bomb scenario, the evacuation shelters need to be added and respectively the different "types" of places have to be made selectable. This means that the AAS needs slightly to be extended. Additionally, the already existing AAS has several other options, which are not needed in our scenario. However, the use of this already available service would save some implementation work but the advantages and disadvantages of using the AAS need to be assessed. The current AAS implementation is shown in figure 7. Instead of a WFS, currently a PostGIS database is accessed for performance reasons. This database uses the *Simple Feature Specification* for SQL (SFS4SQL), which offers analog functionalities like a WFS and therefore it could be easily replaced by a WFS in principle.

An example of an AAS response is illustrated in figure 8. The convex polygon represents the area which is reachable within three minutes from the marked “Location”.

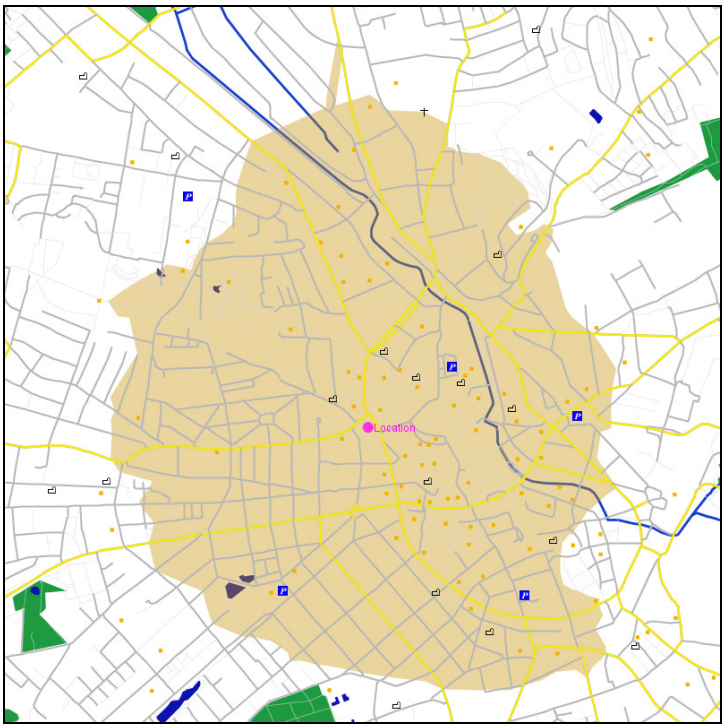

Fig. 8. Example Response Map of the AAS

As mentioned above the AAS is not an OGC standard but it would be possible to hide the service behind the WPS interface. This makes clear that the quite open WPS specification offers also the possibility to use any kind of web service “packed” as a Web Processing Service.

5. Implementation

For the implementation of the described bomb threat scenario we use the existing WPS framework of the *deegree2* project (www.deegree.org) [19]. This and other projects such as the *52°North* project (http://52north.org) and the *Agriculture and Agri-Food Canada* project (www.agr.gc.ca) are in accordance with the current version 0.4.0 of the WPS specification from the year 2005 [20].

The *deegree2* 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 has to be created for the process and integrated into the framework along with the process class. After that the new process is then accessible via the WPS interface.

While the implementation of a WPS from scratch would be time-consuming, simple processes can be rapidly implemented when using frameworks such as the *deegree* WPS. The frameworks relieve programmers of much of the overhead and they can focus on implementing custom processes.

Because the WPS interface is that open, there is no restriction to the process type and process complexity. Therefore, separate processes can also be aggregated into one new WPS process. We have taken advantage of this in our approach by orchestrating all scenario steps via the Composite-WPS but still providing this WPS as an independent WPS process.

The first implementation step based on the Composite-WPS concept is to realize the aggregation of the involved services by direct coding. This will be replaced in a second step by designing a more flexible process class which is fed by configuration files. A future goal is the aggregation of several services within one WPS without touching the implementation itself but only configuring some XML documents.

6. Discussion

Modularization and the reuse of existing modules are the key to Service Oriented Architectures and to object oriented programming in general. Reusing services is supported by applications like BPEL designer which make it possible to orchestrate single services using graphical tools. Unfortunately, difficulties exist with respect to the orchestration of OGC Web Services based on BPEL which compelled us to explore alternatives such as the OGC WPS interface for orchestrating OWS. Consequently, programming languages must be used instead of the simpler graphical configuration tools provided by BPEL designer. The use of BPEL would be more likely in case of existing WSDL documents for all OWS, but with respect to the WPS interface, such a WSDL document is redundant because the response to a *DescribeProcess* requests contains more or less the same information as a WSDL description and would only be necessary for technical reasons, namely the use of BPEL for service orchestration.

The use of the WPS interface itself for the orchestration of OGC Web Services is a “misuse” of the interface and contrary to the original idea. But only the lack of

restrictions and the openness of the interface gave us the idea to “abuse it” in order to aggregate several processes to achieve our goal. It is no secret that the current WPS concept is not fully developed and any complex service can be realized, no matter if this service provides geo-processing functionality or not. Alternatively, it would seem to make more sense, if it offered a well known amount of relevant geo-processing procedures which could serve as building blocks for more complex geo-processing functions. This was taken into account for the new 1.0 WPS interface specification which is currently developed within a working group. In this specification, *Profiles* are introduced which should be further developed by user communities to agree on defined WPS processes. Within particular domains, it is imperative that standardized geo-processing functionalities are agreed upon in order to solve interoperability problems. It is important to remember that final aggregated services (like the bomb threat scenario) are usually domain specific and do not really belong to an unspecified domain OGC service. Luckily, other existing OWS are mostly domain-neutral. On the other hand, these processes do represent geo-processing functions and this is what the WPS was actually designed for. That is why the term "complex" needs to be further defined considering aspects such as: what can still be categorized under geo-processing base functions and when do they start becoming domain specific functions which then may not belong to a WPS? However, the WPS interface closes an open gap in the area of OWS and respectively SDIs and this paper demonstrated that complex scenarios can be completely represented in accordance with OGC standards if needed. But neither the use of the WPS interface nor using WSDL for orchestrating OWS give an answer to interoperability problems in SDIs and this challenged has yet to be solved.

References

1. GSDI (2004). Developing Spatial Data Infrastructures: The SDI Cookbook. Version 2.0. Retrieved May 29, 2007, from: http://www.gsdi.org/gsdicookbookindex.asp.
2. Granell, C., Gould, M. and Poveda, J. (2004): Incremental Composition of Geographic Web Services: An Emergency Management Context. Proceedings 7th AGILE Conference on Geographic Information Science, Heraklion, Greece.
3. Alameh, N. (2003): Chaining Geographic Information Web Services, In: IEEE InternetComputing, Volume: 7, Issue: 5 p. 22-29.
4. Yue Peng, Liping Di, Wenli Yang, Genong Yu and Peisheng Zhao (2006): Path planning for chaining geospatial Web services. In: J.D. Carswell and T. Tezuka (Eds.): W2GIS 2006 (4-5 December 2006, Hong Kong, China), LNCS 4295. Springer-Verlag Berlin Heidelberg. pp. 214–226.
5. Bernard, L., Einspanier, U., Haubrock, S., Hübner, S., Kuhn, W., Lessing, R., Lutz, M. & Visser, U. (2003) Ontologies for Intelligent Search and Semantic Translation in Spatial Data Infrastructures. In: Photogrammetrie - Fernerkundung - Geoinformation (PFG), pp. 451—462.
6. Li, J., Zlatanova, S., Fabbri, A. (eds) (2007): Geomatics Solutions for Disaster Management. Springer, Berlin, Heidelberg, New York, 444 p.
7. Neis, P**.** (2006): Routenplaner für einen Emergency Route Service auf Basis der OpenLS Spezifikation. Diploma Thesis. University of Applied Sciences FH Mainz.

8. Neis, P., Schilling, A. and Zipf, A. (2007): 3D Emergency Route Service (3D-ERS) based on OpenLS Specifications. GI4DM 2007. 3rd International Symposium on Geoinformation for Disaster Management. Toronto, Canada.
9. Kiehle, C., Greve, K. and Heier, C. (2006). *Standardized Geoprocessing – Taking Spatial Data Infrastructures one step further.* Proceedings of the 9th AGILE International Conference on Geographic Information Science. Visegrád, Hungary.
10. Biermann, J., Gervens, T. and Henke, S. (2005): Analyse der Nutzungszenarien und Aktivitäten der Feuerwehr . Internes Projektdokument. Projekt OK-GIS.
11. Weiser, A., Neis, P. and Zipf, A. (2006): Orchestrierung von OGC Web Diensten im Katastrophenmanagement - am Beispiel eines Emergency Route Service auf Basis der OpenLS Spezifikation. In: GIS - Zeitschrift für Geoinformatik. 09/2006. pp. 35-41.
12. Weiser, A. and Zipf, A. (2007): *Web Service Orchestration (WSO) of OGC Web Services (OWS) for Disaster Management*. Joined CIG/ISPRS Conference on Geomatics for Disaster and Risk Management. Toronto, Kanada.
13. Heier,C. and Kiehle, C. (2005): Geodatenverarbeitung im Internet - der OGC Web Processing Service. GIS 2005, 6: 39-43.
14. Stollberg, B. (2006): Geoprocessing in Spatial Data Infrastructures - Design and Implementation of a Service for Aggregating Spatial Data. Diploma Thesis. University of Applied Sciences FH Mainz.
15. Dustdar, S. and Schreiner, W. (2005): A Survey on Web Services Composition. International Journal of Web and Grid Services, 1(1), Inderscience. January 2005.
16. Einspanier, U., Lutz, M., Senkler, K., Simonis, I. and Sliwinski, A. (2003): Toward a process model for gi service composition. In GI-Tage (GI Days) 2003, Münster, Germany.
17. Reichert, M. and Stoll, D. (2004): *Komposition, Choreographie und Orchestrierung von Web Services – Ein Überblick*. EMISA Forum, Band 24, Heft 2, 2004, S. 21-32.
18. Neis, P., A. Zipf (2007): A Web Accessibility Analysis Service based on the OpenLS Route Service. AGILE 2007. International Conference on Geograpic Information Science of the Association of Geograpic Information Laboratories for Europe (AGILE). Aalborg, Denmark.
19. Fitzke, J; Greve, K; Müller, M. and A. Poth (2004): Building SDIs with Free Software - the deegree Project. In: Proceedings of GSDI- 7, Bangalore, India.
20. OGC (2005). *Web Processing Service*. OGC Discussion Paper, Document Reference Number 05-007r4, Version 0.4.0.