Web Service Orchestration of OGC Web Services for Disaster Management

A. Weiser and A. Zipf
i3mainz, Institute for Spatial Information and Surveying Technology
University of Applied Sciences FH Mainz

Summary

Flexibility and reusability are major goals when developing complex applications based on OGC Web Services (OWS). Within the project OK-GIS (www.ok-gis.de) we evaluate the suitability of the Web Service Orchestration (WSO) technology as possible solution for disaster management scenarios. We present an example of a part of an evacuation scenario after a bomb has been found. This scenario includes in particular the need for emergency route planning. We evaluate how the actions to be performed by the system supporting the rescue workers can be mapped onto a service chain of basic OWS. The service chain is represented as a BPEL document and can be executed in a web service orchestration engine, such as Oracle BPEL-engine. BPEL is a standard for service orchestration and means Business Process Execution Language.

Free and open GIS for disaster management - the project OK-GIS

"OK-GIS" is the acronym for Open Disaster Management using free GIS (http://www.ok-gis.de). One of the main goals of the OK-GIS project is the development of a flexible toolbox of software components and services based on a Spatial Data Infrastructure (SDI) that uses open standards (e.g. by OGC, ISO, W3C etc.) This framework should be applicable for various scenarios and requirements in case of a disaster situation. The approach should be independent from specific scenarios. This shall be achieved by a highly configurable system. Technically this freedom can be gained by us-

ing profile dependent configurations on the one hand side and through the aggregation of high-class functionality by using more fine-grained base services, in particular standardised OGC Web Services (OWS) on the other hand side. The latter will be combined to specific workflow chains according to the demands of a corresponding profile. Research was carried out in order to test the possibility for OWS to use the technique of Web Service Orchestration (WSO).

We do present the idea of Web Service Orchestration of OWS and the practical experiences we have made implementing this. We discuss how OWS and in particular an Emergency Route Service (ERS) as well as an OpenLS Route Service (RS) can be used within a service chain orchestrated through BPEL scripts within a scenario where a bomb has been found and people need to be evacuated. The work of the firemen and rescue team that plans this evacuation shall be supported through this. The basic workflow as defined in the rules of the firemen is shown in figure 1.

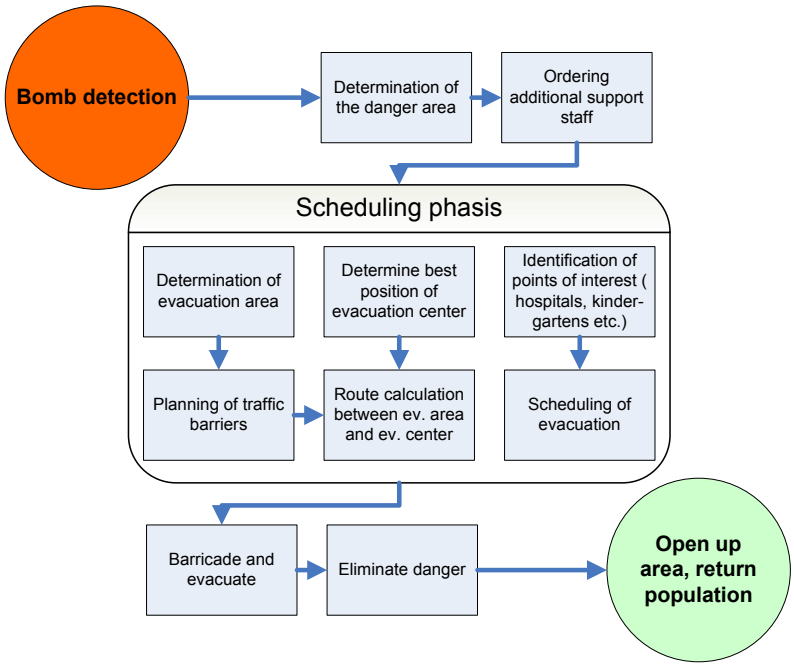

Fig. 1: Simplified evacuation scenario in case of finding a bomb

The above figure shows the workflow of a bomb finding and evacuation scenario. A part of that scenario will be mapped onto a BPEL use case in a later paragraph.

Orchestrating OGC Web Services with BPEL

The aim of OK-GIS project is to develop a modular architecture which ensures flexible reaction to a range of demands in emergency management. Complex functions and workflows of a certain granularity will be achieved through the skilled aggregation of standardized base services (OWS). These new aggregated functionalities can then also be used as web services on their own. So far the middleware performing the aggregation usually needs to be developed using conventional programming languages. This way all the advantages of object oriented languages can be used. The price to pay, however, is that every modification and compilation has to be made within the programming platform. The promise of the alternative WSO is to provide an easy and flexible way to link services in given patterns, especially through the configuration. This is done using so called orchestration scripts. By using WSO (Peltz 2003), it should be possible to map workflows as service chains through XML based languages such as BPEL (Business Process Execution Language), which can be configured with graphical tools instead of hard-coded programming. These BPEL scripts will be executed in corresponding WSO engines. Similar ideas and methods have been discussed by Kiehle et al. (2006) and Lemmens et al. (2006). The suitability of this concept will be reviewed based on real experiences and tests.

In OK-GIS, OWS offer the base for all further functionalities, as OK-GIS focuses on the spatial aspect of disaster management - i.e. finding, handling, managing, visualizing and analysing spatial data. With respect to WSO OWS unfortunately are subject to certain constraints: One of the main problems is the missing SOAP support (Simple Object Access Protocol) of OWS (OGC 03-014). Instead the services communicate through the http-protocol. However, the WSO engines usually use the SOAP-protocol. As a result we need to answer the question whether or not the orchestration of OGC web services is possible at all with BPEL. To summarize the result in short we can say, that our tests have shown that it is generally possible to orchestrate OWS using BPEL. Some technical problems were encountered because OGC Services (OWS) are not SOAP Web Services but XML-RPC[1] via HTTP-Get/Post. These findings will be discussed in the following in more detail.

[1] RPC: Remote Procedure Call - protocol for calling operations and procedures of a service/application from a remote point. Uses mostly the http protocol.

BPEL and what else? - Dependencies and Severities

BPEL and its extension for Web Services BPEL4WS is based on technologies like WSDL (Web Service Description Language), XML Schema and XPath. The highest influence on BPEL has WSDL, because the BPEL process model is built upon the WSDL 1.1 Specification. How do all these parts fit together?

BPEL itself defines the syntax for an XML based programming language (Juric et al. 2004). Although with some limitations, it is nevertheless an adequate programming language because of some properties like:

- loops (while)
- algorithm branch (switch)
- definition of variables

The result of a complete process definition is a BPEL script that will be interpreted by an orchestration engine like Oracle BPEL Process Manager or Active BPEL (Active Endpoints). The engine can be seen as a runtime environment that interprets the BPEL script. But a complete process definition does not only consist of the BPEL-based program sequence, but of the so-called "endpoint" references that reference the services used within the service chain. This is the part that WSDL plays in that concert: WSDL is the interface definition to the involved services. It keeps both- the information of the abstract operational structure of the services and the information of the concrete binding information like service-URL, servlet mapping path and binding protocol. Because WSDL defines the base for defining a BPEL-process, it has also the most critical potential of danger to its breakdown.

Due to the differences of OGC web services (OWS) to "normal" web services there are some incompatibilities between the W3C-built WSDL and OWS. One belongs to the special Key Value Pair encoding (KVP) of the WMS GetMap and the WCS *GetCoverage* requests; the other belongs to a missing raw binary format in XML Schema definitions of WSDL that would be necessary to accept a returned WMS GetMap- or WCS *GetCoverage*- binary (OGC 04-060r1). These limitations and some more issues are part of our evaluation and proof of concept tests.

Orchestration exemplified through the use of Emergency Routing during a bomb discovery

A possible emergency scenario could look like the one we want to discuss now.

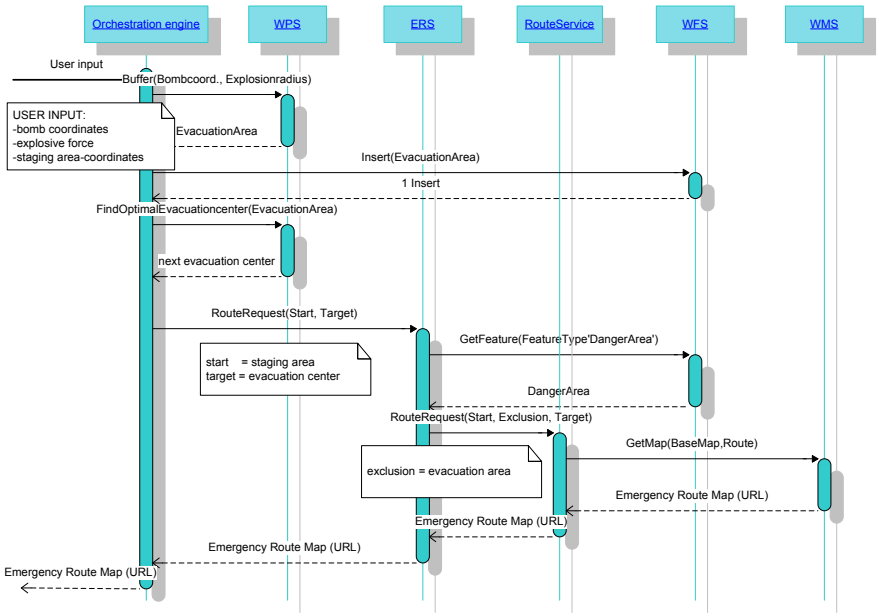

Fig. 2: UML Interaction diagram of involved OWS in the evacuation scenario (simplified)

The following scenario is given: A bomb was found at some place. Now it is necessary to evacuate the people living within a given distance. A part of the scenario includes that it is necessary to calculate an emergency route between the staging area, where all the people from the danger zone (evacuation area) are being concentrated and the shelter. The emergency route needs to bypass the danger zone(s). This will be used as an example to explain how to orchestrate the OWS base services within a service chain, using a WSO engine, such as the Oracle BPEL orchestration engine.

The BPEL script executed by the orchestration engine (OE) integrates a WPS (OGC Web Processing Service), a WFS (OGC Web Feature Service) and an (Emergency Route Service) ERS using an OpenLS Route Service (Neis 2006). The OE acts as a web service on its own that is wrapping the complete process chain. This aggregated ("virtual") service needs as input parameters from the user merely the coordinates of the bomb, as well as its explosive force and finally the coordinates of the staging area.

The evacuation area can then be determined in several different ways. One option is that the explosive force (available within an OE internal XML look up table) can be mapped to a certain evacuation radius. With the bombs coordinates and the radius, the WPS can identify the evacuation

area by using a simple buffer function (cp. Heier and Kiehle 2006). Of course also more complex versions can be implemented such as information about the affected population in the area.

In the next step the OE starts a WPS process called „*FindOptimalEvacuationCenter*“, which defines the nearest shelter from the evacuation area. That WPS process implements typical GIS functionality like buffer and intersection (Stollberg 2006). Using the acquired geometries of the danger zone(s), the staging area and the shelter locations, a *RouteRequest* is sent to the ERS. The ERS gets all the dangerous and closed areas from the WFS using the *GetFeature* function. These are integrated into the request and sent to the RouteService as *AvoidAreas*, along with start- and destination points. The OpenLS RS then calculates the route including the *RouteGeometry*. This is then transferred through a *GetMap*-Request by the route service as a *UserLayer-SLD* (which is possible with the version 1.1.0 of the SLD standard) along with the parameter for a base map. The client receives a map with the specified emergency route as well as further relevant information regarding this route as resulting output.

Feasibility

The question that arises when discussing the scenario that was explained above: “Is it actually possible to build a service chain like postulated above”? What are possible limitations of OWS using such a technology like WSO?

In order to answer the questions we performed a range of test. The answer in short is that we successfully orchestrated several OGC Web Feature Services (WFS) and OGC Web Catalogue Services (CS-W). These services can be taken as examples for the feasibility of all http-POST based OWS.

In particular the OGC WMS GetMap request with the Oracle BPEL Process Manager (OPM) 10.1.2 and 10.1.3 was tested. The promise from Oracle was to also support transportation of binary raster data through the engine. The previous version supported only base64Binary, a lexical encoded XML binary type based on the 64 numeration. The latter version also supports hexBinary which bases on the hexadecimal numeration. But our tests showed, that none of them grant success. A raw, uncoded binary type would be necessary to accept a GetMap response, returning maps as binary raster data.

A possible solution would be a proxy server acting as binary transcoding service. It would transcode the uncoded binary bitmap to a base64Binary. That would of course increase the amount of traffic on the

wire and the necessary computing power by the factor 3, because the bitmap would have been decoded between WMS and BPEL process and also recoded between BPEL process and client.
A qualified question may be: is piping of large binary data like a WMS-map or a WCS-coverage through orchestration engines sensible at all? It is very likely that it would slow down the response time of the whole system. But displaying spatial data is essential in disaster management. Street maps have to be shown, directions have to be displayed, hazard zones to be mapped.

There is another solution to that problem: Instead of maps being exchanged, references to the maps can be communicated:

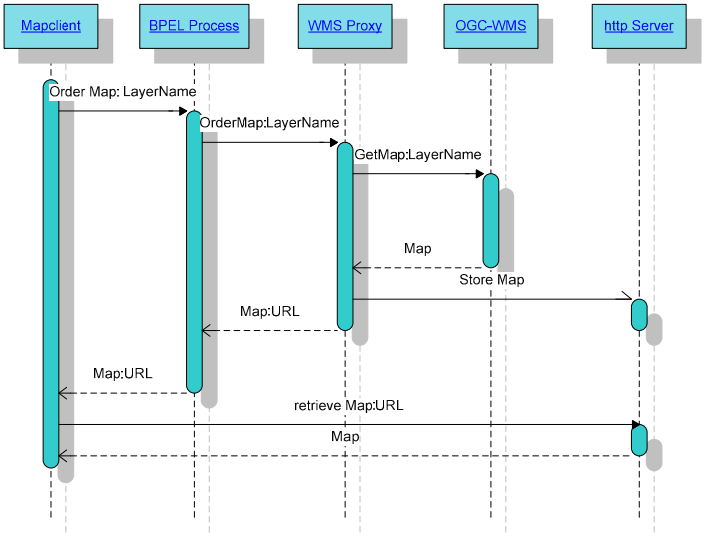

Fig. 3: UML sequence diagram of an alternative WMS orchestration using WMS-proxy and a map client

The above figure shows a possibility to embed an OGC WMS (or alternatively WCS) in a BPEL process chain.
The map-generation was initiated by the map client. The map client is not completely OGC compliant, as it handles the response slightly differently as explained here: First the client requests a map in an OGC compliant way by passing the desired LayerName (and all other required GetMap parameter) to the BPEL process (instead of the WMS directly). The process hands the parameters over to the WMS proxy-server. The proxy launches an OGC compliant GetMap request to the WMS and receives the GetMap response from the WMS. Subsequent the proxy stores the physical part of

the map on a simple http-server directory and returns the URL of the stored map back to the BPEL process. The process gives the information back to client. The map client then collects the stored map from the specified URL.

The main difference to an OGC compliant map retrieval of a WMS client is that the WMS-map generation is a synchronous operation. In this example calling the map and displaying it was divided in two (partly asynchronous) operations.

The OpenLS Route Service (RS) described above does this alike. It works as a WMS proxy server that stores the map instance on an http-server and gives back the RouteMap which contains the route summary, route instructions, URL to the map-instance etc. (Neis 2006).

Another possibility totally different to the one above is the usage of an SLD inline Feature. It is an SLD 1.1 concept where the SLD contains a collection of simple features "inline" its body. Fig. 4 shows the service chain:

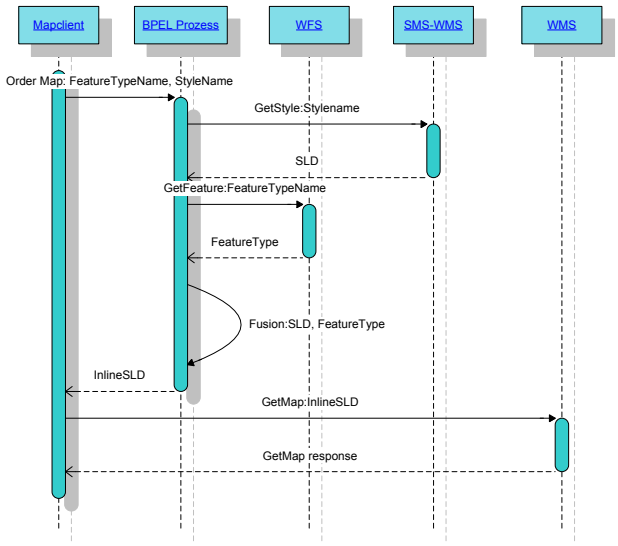

Fig. 4: UML sequence diagram of an alternative WMS orchestration using WFS, SMS-WMS and a map client

A map client is the base of operations. It passes a request to the BPEL process. The process requests the current style of a special Style Management Service-WMS (SMS-WMS) (OGC 04-040). The SMS-WMS responds an SLD to the process. Subsequent the process requests a Feature Type from WFS. The BPEL process merges the responded *FeatureCollec-*

tion and the SLD to an *InlineFeature*-SLD and passes it to the calling map client. The map client sends the inline-SLD to a compliant WMS and displays the resulting map.
There are currently only a few WMS implementations able to compute an inline-SLD. One is the geoserver WMS 1.4.0. It supports WMS 1.2.0 and SLD 1.1.0.

These two examples show how a WMS can be integrated into an orchestration process in spite of the current limitations. This means that the handling of maps can be incorporated in a larger orchestration process like presented in fig. 2.

Web service allocation security - BPEL ensured find-bind

The availability of all necessary services is one of the most critical points of view for service oriented architecture (SOA) in an emergency response situation. Particularly in a supraregional natural disaster the electrical and electronic infrastructure may suffer and therefore support of a disaster management system might fail partially when it is really indispensable.

Therefore an effective emergency support system (ESS) needs a high level redundancy in its participating services.
If the SOA has information about several redundant services on infrastructural independent locations, it can choose between these services.
This means that it is necessary to implement an algorithm that enables the system to choose automatically the necessary service that is currently up and running.

Fig. 5 shows the whole “secure Web Service Allocation” BPEL-process.

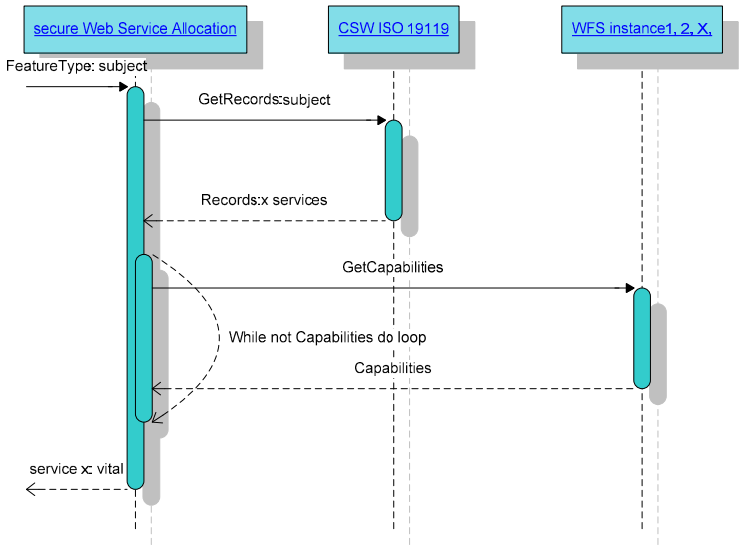

Fig. 5: UML sequence diagram of the secure Web Service Allocation

As a simple example the process starts with the request of a special *FeatureType*. That may be a client-application where this feature was selected in a dropdown list. The BPEL process launches a semantically tolerant *GetRecords* request to an ISO 19119 catalogue service with the specified subject. All records with according ISO *subject* elements will be returned to the process. These records represent different instances of redundant Web Feature Services. They will be tested in a case-expression for the ISO element *MD_DataIdentification.SupplementalInformation*. If the service appellation of this element correlates with a service appellation of a service defined in the BPEL process, it will try to invoke the WFS. For these purposes the WFS GetCapabilities operation is being requested from that service.

If it answers with a capabilities response, the process exits the loop, because a vital WFS instance that serves the desired subject was found. The calling client was sent the information “service X is up” so that the client in the next step can request the desired feature from the service.

The aim of the process is a necessary and integral part of a successful working disaster management system and hence can become a future module of the OK-GIS platform. The only constraint in this way of service allocation is that the services cannot be defined during runtime. It is not necessarily the case that all available services are found in a GetRecords

request, but all possible services need to be previously defined in the process to be available if necessary. All endpoints must be well known in the run-up to the original definition.

The above BPEL process can be combined with other processes like the FeatureType Retrieval we already implemented:

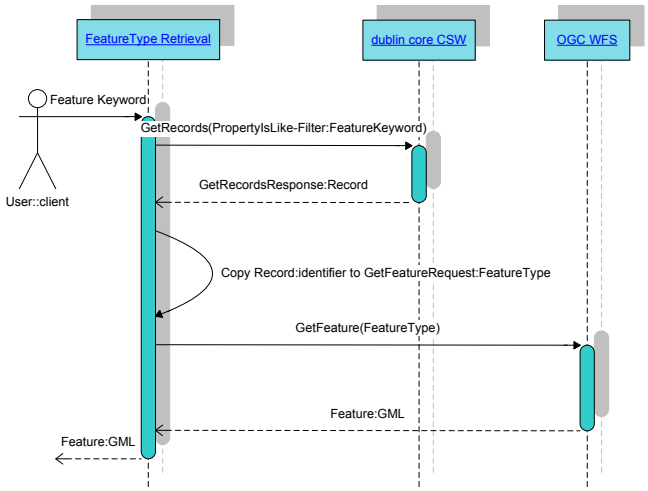

Fig. 6: FeatureType Retrieval in the OK-GIS SDI

For the OK-GIS GDI it was necessary to join the metadata with the relating geodata. To manage this there are (at least) two possibilities:

One was to use the ISO 19115 metadata schema. There are 2 possible elements to contain the relation to the geodata:

- *MD_Metadata.identificationInfo/MD_(Data)Identification.citation/CI_Citation*, which contains the ID of the geodataset
- *MD_Metadata.distributionInfo/MD_Distribution.transferOptions/MD_DigitalTransferOptions.onLine*, where the link to the http-GET request can be placed

In this way a “normal” web service could send a request to the WFS after obtaining the record from the previous *GetRecords* request.

The other possibility to join metadata with geodata was to use a Dublin Core (DC) CSW and BPEL, because in conjunction with BPEL the metadata information of the http-link is not really necessary. All the necessary information (like the reference to the relating geodata-service) can be found in the corresponding WSDL.

As a BPEL process for a valid *GetFeature* request only the *FeatureTypeName* is necessary. All further information like service, etc. can be

initiated in the BPEL script. The DC element *Identifier*[2] was used for containing the *FeatureTypeName*. The *FeatureTypeName* (plus Namespace-prefix) establishes the relationship to the concerning dataset, hence the usage is compliant to the Dublin Core (DC) metadata standard on the one hand and on the other hand it allows the usage in a *GetFeature* request.
This means that the process flow works very similar to that described in the paragraph on Web Service Allocation Security.
A difference to the example above is that the catalogue used is a Dublin Core compliant CS-W. It returns the records found in exchange to a requested *Feature* keyword. This keyword can be any name of the *FeatureType* the user is looking for. If there is an according name in the DC *description* element, the record will be found.
Another difference concerns the handling of the records. The requestor wants one special *FeatureType*. For this purpose the response is tested on its element *SearchResults* that contain the attribute *numberOfRecordMatched*. If the attribute contains more than one hit, the result is ambiguous and will be terminated after informing the requestor about this condition. If the attribute contains no hit, the name of the *FeatureType* could not be resolved. If its exactly one hit, a *GetFeature* request will be send to the corresponding WFS asking for the FeatureType with that special FeatureTypeName. In a final step the *Feature* will be returned to the requestor.

We made some test series to get an average response time. The medial for the *FeatureType Retrieval*-process was 3 seconds on a not too new desktop computer. Before every single test the browser cache and history was cleared. Most measured values ranged from approximately 2 to 3 seconds (if all endpoint services were up and unemployed). Always at the first test after startup of the engine we got some overlong process durations (from approximately 8 until 22 seconds).
So we infer that the engine is caching internal connection parameters.

Current Work

Orchestration of OGC web services (OWS) is not a brand-new topic although it is still highly interesting as only recently real experiences using technologies like BPEL have been reported. Within OGC discussions on service oriented architectures with OWS have started at least since the OWS2 common architecture discussion paper (OGC 04-060r1) from 2004. Although there were some earlier considerations to adjust the architecture

[2] According to the Dublin Core Metadata initiative the purpose of the element *identifier* is: "An unambiguous reference to the resource within a given context"

towards compatibility to common web services (OGC 03-014) the OWS2 paper offered the first helpful suggestions in this direction.
Hence there are some first trials to provide "real" SOAP-based web services technology for OWS in contrast to the earlier XML-RPC. For example some WSDL-files can be found at the official OGC schema sites for Web Feature Service (…/wfs/1.1.0/wsdl). The WSDL are readily configured for http-GET-, –POST- and SOAP-binding.
When testing the WSDL with an Oracle BPEL designer 10.1.2, the WSDL were accepted from the engine internal XML processor, but no WSDL structure was displayed and hence it was not possible to be used within the orchestration engine. Some results of our investigations about the reasons of the failure follow in the next section:

On the other hand the XML schema files for *GetFeature* request and response provided by OGC worked well inside a self-made WSDL. That's a great relief, because the challenge in creating WSDL isn't the WSDL itself, but the XML schema type definitions of the very complex OWS messages.

Recently, within the internal OGC discussion paper "OWS4 Workflow IPR" (OGC 06-187) has been written that presents some results from the OWS4 initiative. It contains some BPEL use cases for/related to the OGC Geo Processing Workflow. Within the defined service chains there are WFS and Web Processing Services (WPS) used.
There are also examples of BPEL workflows used in complex ESA and NASA Spatial Data Infrastructures (SDIs). The NASA use case uses the grid operation of a WPS, triggered by a Web Coverage Service. The ESA use case is a bit more complicated and much more interesting: There, a Sensor Planning Service (SPS) of the OGC Sensor Web Enablement (SWE), a Web Coordinate Transformation Service (WCTS) and a Web Coverage Service are being aggregated to a service chain. The SPS catches raw level 1a (not geocoded) SPOT satellite images and transforms them via WCTS to a geocoded ortho image. The BPEL process passes the reference to that ortho image back to the calling client. Unfortunately within the paper there is no explanation of the interaction between client and process, but the use of the WCS seems to be similar like the use of WMS described in the paragraph "feasibility of Orchestration through the use of Emergency Routing during a bomb discovery".

Some initiatives react on the increasing demand of W3C compliant web service interfaces. So did 52°North, Münster, Germany for their recently released Sensor Observation Service (SOS) and Web Notification Service (WNS). They offer a few already configured WSDL for the current version

of their service (the pre standards of SOS and WNS are still heavily under construction) (Claussen 2007).

Conclusion

BPEL and Web Service Orchestration attract more and more interest within the GI community due to their increasing popularity in main-stream computer science. The proof-of-concept evaluation presented here that it is possible to create an added-value by combining and aggregating OGC Web services. Of course the use of BPEL must be reasonable and purposeful. It only makes sense to use orchestration where a continuous service chain without human intervention is possible. But in particular in many cases within disaster management there are a lot of unforeseeable factors influencing the service chain in a dynamic way. This means that it is on the one hand necessary to find relatively stable standard chains (small parts of a larger workflow that can act as modular building blocks) and on the other side it is necessary to research how such BPEL scripts can be assembled in an even more dynamic way even for non-technical users. This needs work on relevant user interface concepts in order to ease the highly dynamic orchestration of OWS on the fly.

Because of the big benefit when using standards to develop a platform for disaster management is that this results in a highly adjustable toolkit.

In OK-GIS we develop a spatial data infrastructure and the necessary server and client applications to support some typical use cases for the fire department. But the fire brigade is here only a typical example representative for the variety of disaster task forces. The aim of this work is to make it possible to add new use cases to the overall system, without the need for difficult changes within the code or even the program structure. Therefore we decided to test new technology, use standards like BPEL and standards-based software: With that middleware technology it is possible to create loosely coupled service networks with a high reliability. These can use and integrate other (OGC-) web technologies like the OGC Sensor Web Enablement (SWE). For the use case of a flood scenario we test the integration of SWE services into our OK-GIS SDI (Claussen 2007) in order to implement a web based flood early warning system.

Literature

Biermann J, Gervens T, Henke S (2005): *Analyse der Nutzungszenarien und Aktivitäten der Feuerwehr*. Internal Project Report. Projekt OK-GIS.

Botts M, Robin A, Davidson J, Simonis I (2006): *OpenGIS® Sensor Web Enablement Architecture Document.* V. 1.0. OGC Ref. Nr. 06-021r1. D. P.

Chen L, Wassermann B, Emmerich W, Foster H (2006): *Web Service Orchestration with BPEL.* Dept. of Computer Science, University College London

Claussen K (2007 in work): *Exemplarischer Aufbau einer Service Chain mit Diensten aus dem OGC Sensor-Web unter Verwendung von Open-Source-Software* (OGC Sensor Web for disaster management). Master Thesis. University of Applied Sciencens FH Mainz.

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.

Friis-Christensen A, Bernard L, Kanellopoulos I, Nogueras-Iso J, Peedell S, Schade S, Thorne C (2006): *Building service oriented applications on top of a spatial data infrastructure – a forest fire assessment example.* AGILE 2006.

Juric M, Mathew B, Sarang P (2004). *Business Process Execution Language for Web Services.* Packt publishing Ltd., Birmingham, 2006

Kiehle C; Greve K & 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.

Lemmens R, Granell C, Wytzisk A, de By R, Gould M, van Oosterom P (2006): *Semantic and syntactic service descriptions at work in geo-service chaining.* Proc. of the 9th AGILE Int. Conference on Geographic Information Science. Visegrád, Hungary

Neis P. (2006): *Routenplaner für einen Emergency Route Service auf Basis der OpenLS Spezifikation.* Dipl. Thesis. University of Applied Sciences, Mainz.

OASIS. http://www.oasis-open.org/apps/org/workgroup/wsbpel/

Open Geospatial Consortium Inc. *OWS2 Common Architecture.* Hrsg. OGC. RefNum. OGC 04-060r1; Vers. 1.0.0; Status: OGC Discussion Paper.

Open Geospatial Consortium Inc. *OWS 1.2 SOAP Experiment Report.* Hrsg. OGC. RefNum. OGC 03-014; Vers. 0.8; Status: OGC Discussion Paper.

Open Geospatial Consortium Inc. *OWS-4 Workflow IPR.* Hrsg. OGC. RefNum OGC 06-187; Vers. 1.0.0; 2007-02-15 Status: internal OGC Discussion Paper.

Open Geospatial Consortium Inc. *Style Management Services for Emergency Mapping Symbology.* Hrsg. OGC. RefNum OGC 04-040 Vers. 1.0. Status: OGC Discussion Paper

Peltz C. (2003): *Web services orchestration and choreography.* IEEE Computer.

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