Methods and Tools for More Efficient Working With OGC Web Processing Services

Andreas Abecker, Roman Wössner
Disy Informationssysteme GmbH
Karlsruhe, Germany
firstname.lastname@disy.net

Dorian Alcacer-Labrador, Felix Bensmann, Rainer Roosmann
University of Applied Sciences Osnabrück
Osnabrück, Germany
d.alcacer|f.bensmann|r.roosmann)@hs-osnabrueck.de

Abstract—We start with the observation that the OGC Web Processing Service (WPS) is by far less widespread in public administrations than the OGC download and visualization services WMS, WFS, WCS. Based on this observation, it was the goal of the RichWPS research project to develop methods and tool which should facilitate the use of WPS in practice. Now, after the end of the RichWPS project, we briefly review most of its methodological and technological outcomes and try to conclude to which extent the project goals have been achieved.

Keywords—OGC Web Processing Service, Service-Oriented SDI

I. INTRODUCTION

By the INSPIRE directive, public authorities in Europe are obliged to contribute to the realization of an EU-wide, service-oriented Spatial Data Infrastructure (SDI) for provision and use of geospatial data. It has been shown that OGC Web Services (OWS) provide an excellent basis for realizing such service-oriented SDIs. However, while OGC Web Feature Service (WFS), Web Coverage Service (WCS) and Web Map Service (WMS) are already in wide practical use as download and visualization services, the OGC Web Processing Service (WPS) is by far less widespread in the reality of public administrations.

Although it seems obvious that processing services could (and should?) be the next evolution step of service-based Spatial Data Infrastructures (SDIs) and that distributed geoprocessing has to offer manifold advantages, its usage – since years – is to a large extent limited to academic prototypes. Besides non-technical barriers for a wider use of WPS in administrative practice (like the question of “payment” if one administrative unit develops and others re-use a WPS service), at least four technical issues related to WPS were identified:

- Composition: simple tools for building complex WPS processes from existing ones
- Retrieval: finding reusable components based on comprehensive semantic metadata
- WPS dialects: data types and encoding of I/O parameters are not equal for all WPS implementations
- Output processing: the data-type agnostic protocol does not tell the client how to further process WPS results

Hence, the aim of the RichWPS research project1 was to design and prototypically develop methods and tools which support end users (who are no IT experts) in employing WPS for their work.

In the meanwhile, the RichWPS project has been finished, all implementation works have been concluded, and the project results have been validated and evaluated by domain experts from nature conservation and marine-data management. Hence, this paper shall (a) summarize the totality of RichWPS developments and their play-together and (b) draw some conclusions about the usability and usefulness as well as critical success factors and required pre-conditions of the WPS use in public administrations in environmental informatics.

In former publications and talks, (i) the RichWPS overall solution approach, (ii) the RichWPS case studies from environment monitoring and marine-data management, and (iii) the RichWPS orchestration approach and the respective tools for service modeling and orchestration have already been presented (cp. [1], [2]). So, this paper will lay its emphasis on the presentation of the overall solution (Section II) and on the (mostly client-side) components not yet published so far (Section III). Section IV concludes with a summary and short discussion of the project results.

II. OVERVIEW OF RICHWPS COMPONENTS

First, we sketch the overall RichWPS architecture with the main software elements shown in Figure 1 which play together for enabling orchestration of WPS processes by domain experts – instead of IT experts. These main elements are:

The RichWPS ModelBuilder (cp. Figure 2) serves as interface for domain experts. It allows graphical modelling of complex processes from existing ones. To this end, the ModelBuilder automatically discovers existing processes and data by using the SemanticProxy. An easy graphical notation is used to build and configure new geospatial workflows. Those can be tested, optimized and finally published and managed using a RichWPS Server. The ModelBuilder itself is not considered a major innovation of RichWPS – indeed, it resembles, of course, very much similar tools from commercial or open source WPS suites. However, the ModelBuilder is the

1 See also http://richwps.github.io/
central point of interaction with the RichWPS system where the knowledge of the SemanticProxy and the functionalities of the RichWPS Monitor can be combined in order to achieve an easy-to-use, yet powerful WPS design-time support.

Figure 1: Main Elements of RichWPS Architecture

Figure 2 shows the ModelBuilder with its different parts: the ResourcesView at the left-hand side lists the OGC Web Services (on local machine and in the Web) available for use; the ModelEditor in the middle ground is the canvas for graphical process composition; the StatusView below the ModelEditor gives actual feedback and status information about model and server (error messages etc.); the ButtonBar at the top allows starting different functionalities (testing, profiling, deployment); the PropertyView at the right-hand side presents characteristics of the resources under consideration (input/output parameters of services, etc.).

The RichWPS Server manages the centralized execution of workflows by a custom orchestration engine. Further, it enables domain experts to test, optimize and manage workflows at runtime. Therefore, the RichWPS Orchestration Language (ROLA), a custom language for OWS orchestration, has been developed. The RichWPS Server manages the centralized execution of workflows by an orchestration engine that interprets ROLA scripts. The RichWPS server is an extension of the 52°North WPS server with renovated functions for WPS lifecycle management (WPS-T based on [4]). The RichWPS server also allows for testing and optimization of workflows by delivering intermediate results and profiling information to the ModelBuilder at design-time. Further, the RichWPS server provides information about the data types it works with, and it supports the WPS Presentation Directives [3].

Figure 3 illustrates the RichWPS server elements and interfaces. Based on the 52°North implementation, a ROLA interpreter and orchestration engine for local and Web-based services is added, as well as a WPS client API for programmatic access. Besides the standard WPS1.0.0 operations and the operations for WPS-T service-lifecycle management (deploy/undeploy), the operations TestProcess and ProfileProcess are implemented which allow monitoring of processes during design-time. Further, the operation GetSupportedTypes allows inspecting the server for the input/output-datatypes supported. Since in our assessment of WPS usage barriers, the incompatibility of different WPS frameworks with respect to IO datatypes was a severe problem.
for true interoperability between servers, we consider this small extension an important thing.

The RichWPS SemanticProxy covers the field of OWS service description and discovery. Services (WPS, WFS) can be registered manually or discovered at runtime. The service- and content description takes place using the Resource Description Framework (RDF). Based on that, a custom vocabulary enables the interface- and content-specification beyond OWS service descriptions.

The RichWPS Monitor enables the scheduled metering of WPS-services and contained processes. Based on prepared queries the Monitor delivers QoS-information to the Semantic-Proxy. Linked systems, such as the ModelBuilder and the Server, are enabled to optimize workflows when needed.

These components that focus on the upper two issues listed in the introductory section (retrieval, composition), are complemented by client-side extensions dealing with the lower two issues in the list above (WPS dialects, output processing).

III. CLIENT- AND SERVER-SIDE RUNTIME COMPONENTS

A number of further, mostly client-side, developments has been made for facilitating truly interoperable WPS-based infrastructures. Their usage is mostly not necessarily restricted to a purely RichWPS-based environment, but could support any WPS environment. These developments include:

An agent-based intermediate communication layer for mobile WPS Clients aims at supporting efficiently dealing with weak or unreliable Internet connections for outdoor work with mobile devices [5]. The idea of this communication layer is illustrated in Figure 4: Two agents establish an intermediate (proxy) layer between stable and potentially unstable parts of the connection and “buffer” network irregularities. The intermediate pair of proxy agents has been prototypically implemented for an iOS and Android mobile WPS client (part of the Cadenza Mobile solution). The agent pair implements a generic and extensible approach able to deal with manifold different optimization mechanisms for different kinds of network irregularities. At the moment, in particular mechanisms for keeping small the network load have been implemented, namely compression algorithms and vector tiling.

An Adapter Layer for WPS Clients has been implemented that allows working with numerous widespread data types served by different WPS-server implementations - including Esri ArcGIS Geoprocessing Services accessed through the ArcGIS REST interface (cp. Figure 5) [6].

This development addresses an issue already mentioned above: Although there is a number of powerful and efficient, commercial and open source WPS frameworks they all differ in details of the input-/output-datatypes they support and how the data are transferred. This is what we call “WPS dialects”. This represents a serious impediment for true interoperability and reuse of WPS processes offered by different server implementations – what we consider a necessary precondition for real WPS-based distributed geoprocessing architectures. Hence we have prototypically developed the adapter layer illustrated in Figure 5: It is a generic and extensible framework where different server and client implementations can be included and the supported datatypes are mapped and transformed in order to achieve interface interoperability. In the prototype, I/O-data from RichWPS, GeoServer and ArcGIS
Geoprocessing have been included at the server-side and the Disy Legato WPS client has been supported at the client side.

A further-development and a prototypical implementation of client- and server-side for the Disy WPS Presentation Directives (WPS-PD) [3] has also been integrated. WPS-PD allows the server to communicate to the WPS-client the kind of WPS-result data types and the way how to present them in the GIS. Currently, the following result types are included: Link; Message; Viewport; Marker; Group; StyledFeatureCollection.

For each of these WPS-output types, the client “knows” how to visualize them. The WPS-PD protocol also makes possible to transmit additional presentation-specific details. Figure 6 shows the simple example of a “Marker” result, transmitted through the WPS-PD protocol and visualized in the Cadenza Web WPS-client as a marker at a specified x-y-coordinate in the map, together with a text message.

IV. RICHWPS PILOT APPLICATIONS

Two pilot installations have been undertaken in the RichWPS project. One of them belongs to the tasks of the implementation of the Marine Strategy Framework Directive (MSFD) and the Water Framework Directive (WFD) by Schleswig-Holstein’s Government-Owned Company for Coastal Protection, National Parks and Ocean Protection (LKN). Together with further North Sea resident countries the status of the North Sea is recorded, evaluated and reported regularly. For reasons of transparency, traceability and comparability this shall be realized using WPS.

The scenario deals with the assessment of macrophytes in the Wadden Sea of Schleswig-Holstein. The spreading of sea weed and algae is a quality factor involved in assessing the eutrophication state for the WFD. The procedure is described in more detail in [7], but in order to provide an impression of the task, the applied data-processing steps are outlined here again:

- Statements are made for a defined reference area.
- Statements about parameters can refer to different, overlapping spaces; they are mapped to each other using a geometrical intersection.
- Statements about spaces are made binary (true/false), quantitatively or gradually – often in relation to the area.
- Exceedances or shortfalls of thresholds mark changes in quality of the environmental status.
- Often multiple factors are incorporated in the calculation. The factors need to be harmonized and mapped to a common scale.
- For the determination of assessment levels a qualitative classification is done.

The use case had already been implemented exemplarily as tightly integrated WPS process for the 52°North WPS server [8]. For the test the process is subdivided into several smaller processes that are deployed on a WPS server. Using RichWPS the processes should be recomposed to a workflow. Since new interfaces originated with new data types at the processes, additional parsers and generators had to be implemented.

The resulting processes are:

- SelectReportingArea
- MSRLDSSelection
- SelectTopography
- Intersect
- Characteristics

Figure 7: Workflow Diagram for the Macrophyte Assessment Scenario in the RichWPS ModelBuilder

Figure 8: Execution order of Sub-Processes
They are recomposed to an equivalent workflow. SelectReportingArea and Intersect is used multiple times. Figure 7 shows the resulting workflow created with the RichWPS Modelbuilder.

The translation of the model into ROLA creates a sequential workflow script as shown in Figure 8. The workflow is published as a WPS process. Figure 9 shows the result of the example application.

Figure 9: Result of Macrophyte Assessment

V. CONCLUSIONS

The software environment presented has been fully implemented and is partially available as Open Source on GITHUB. Two pilot applications have been realized in the context of the German Federal Waterways Engineering and Research Institute (BAW) and the Schleswig-Holstein Government-Owned Company for Coastal Protection, National Parks and Ocean Protection (LKN). The piloting experience shows that the implemented tools can facilitate the use of WPS significantly. Nevertheless, it was probably too optimistic to expect a solution fully applicable by non-IT people in all respects. Instead, some consulting will probably still be necessary, but more seldom and easier. In particular, the reuse of implementations and the interoperability of independent nodes will be supported.

The RichWPS environment is not “yet another WPS workbench”, but extends existing Open Source code (especially the Legato client and the 52°North server, http://52north.org/communities/geoprocessing/wps/) by elements not yet provided by other solutions. Parts of the solution could easily be integrated into other WPS implementations. RichWPS addresses the issues identified in the Introduction: (1) ModelBuilder, OrchestrationEngine and ROLA resolve the composition issue; (2) SemanticProxy shows a way how to address the retrieval issue; (3) the client-side AdapterLayer exemplarily solves part of the WPS dialect problem, in a generic and extensible manner; (4) output processing is facilitated through the WPS presentation Directives.

We do not expect that these contributions are already sufficient to solve “all WPS problems”. But we do think that besides the non-technical issues, WPS has significant inherent weaknesses and problems where the approaches presented could provide solution approaches. Based on practical experience, we also consider these problems (especially the interoperability with respect to data types of I/O parameters) severe blockers that must be eliminated before WPS can unfold its enormous application potential.

It should also be noted that – according to our knowledge regarding the upcoming WPS2.0 specification – although WPS2.0 will offer a couple of new and interesting features, we do at least not expect that it will address the problems of composition, WPS dialects and output presentation.

REFERENCES