Multidisciplinary Research on Geographical Information in Europe and Beyond
Proceedings of the AGILE'2012 International Conference on Geographic Information Science, Avignon, April, 24-27, 2012
ISBN: 978-90-816960-0-5
Editors: Jérôme Gensel, Didier Josselin and Danny Vandenbroucke

Building Standards-Based Geoprocessing Mobile Clients

Alain Tamayo
Institute of New Imaging Technologies
University Jaume I
Castellón de la Plana, Spain
atamayo@uji.es

Carlos Granell
Institute for Environment and Sustainability
EU-Joint Research Center
Ispra, Italy
carlos.granell@jrc.ec.europa.eu

Laura Díaz and Joaquín Huerta
Institute of New Imaging Technologies
University Jaume I
Castellón de la Plana, Spain
laura.diaz@uji.es, huerta@uji.es

Abstract

The adoption of geoprocessing service clients in mobile devices seems to be still rare, even when the communication protocol provided by the Web Processing Service (WPS) seems to fit the philosophy of accessing computation-intensive processes from resource-constrained devices such as mobile phones. One of the reasons to such a low use of WPS services is that input and output data used in geospatial processes are often encoded in some XML-based format, which demands large processing capabilities for mobile devices. In order to deal with this problem we present the WPS Mobile Framework. This framework provides light-weight libraries to communicate with WPS servers and it carries out automatic generation of XML data binding code for mobile devices tailored to specific application needs.

Keywords: WPS; Geoprocessing; Mobile Computing; Android applications; XML Data Binding.

1 Introduction

The success of service interface specifications defined by the Open Geospatial Consortium (OGC) is witnessed by the large number of service instances available online [4]. The number of client and server implementations for these geospatial specifications registered in the OGC website is counted by hundreds.

Whereas geospatial client-side developments have been mostly focused in desktop and web environments, mobile enviroments have received a reduced attention. As the processing capabilities of mobile devices is in continuous growth, the idea of accessing geospatial data in mobile devices has been recently popularised by applications such as Google Maps for Mobile[1]. However, the burden related to XML processing and the large size of geospatial data are a serious obstacle to the massive use of OGC Web Services (OWS) in mobile computing contexts.

The Web Processing Service (WPS) [6] specification aims to expose any kind of (geospatial) processes as web services. In doing so, the WPS specification describes clearly a simple communication protocol that allows clients to interact with exposed processes at the server side. In contrast to other OGC specifications for data services, WPS presents an additional challenge as WPS providers may define their own data encodings and formats of input and output parameters of the available processes. This implies that the development of WPS clients is a very specific task, since such clients should potentially manage a great variety of data encodings and formats. This challenge is even bigger when WPS clients are developed for mobile devices as they present serious constraints in terms of memory and processing capabilities.

In order to deal with the previous issues we present in this paper the *WPS Mobile Framework* (WMFW). This framework is targeted to build WPS clients for mobile devices. It provides built-in code to support the WPS communication protocol and also generates XML processing code for Java-based mobile devices. The usefulness of the framework is demonstrated by building two WPS-compliant client applications with different requirements.

The remainder of this paper is structured as follows. The following section introduces the XML processing topic. Section 3 presents related work. In Section 4, the description of the framework is provided. Section 5 describes two sample applications built using the framework. Section 6 discusses challenges and open issues in the topic of geoprocessing mobile clients. Last, conclusions are presented.

2 XML Processing Code

XML processing can be implemented using low-level APIS such as DOM[2], SAX[3], and StAX [4]; or using high-level solutions such as *XML data binding* tools where XML data is mapped into application-specific concepts. The first option is recognized to be difficult and error-prone producing code that is hard to maintain. The second option is usually favoured as developers can focus on the semantics of the data they are manipulating and the productivity is improved through the use of code generators.

The increasing number of XML data binding code generators based in XML Schema[5] (and other schema languages) emerged in the last decade has facilitated the task of producing XML processing code. However, XML processing code generated from large and complex schemas cannot always be built in an automatic way. This is the case of OWS schemas that with its large size provokes that the amount of generated code is often too large

[1] http://www.google.com/mobile/maps/

[2] http://www.w3.org/DOM/

[3] Simple API for XML (http://www.saxproject.org/)

[4] http://jcp.org/en/jsr/detail?id=173

[5] http://www.w3.org/TR/xmlschema-1

to be executed in devices with memory and/or processing limitations [7].

3 Related Work

Due to the nature of WPS itself, a generic protocol to execute remote processes, it is very difficult to build a generic client that suits to every process because of the great degree of freedom in defining input and output parameters [6]. Among the few WPS clients and supporting WPS libraries, a prominent example is Geotools[6], an open source library written in Java that includes a wide set of tools for handling geospatial information. Geotools provides support for several OWS specifications, WPS included, and benefits from other contained tools to process several common geospatial information formats. Another set of clients for WPS is developed by *52°North - Initiative for Geospatial Open Source Software GmbH*[7], which include clients for uDig[8][1] and JUMP[9].

Literature about geoprocessing web services based on the WPS specification rarely mentions the topic of mobile or embedded devices. The only exception known by the authors is [9], where a GPS navigation system to access a geoprocessing server is presented.

Commercial products such as ArcGIS Server[10] integrates geoprocessing capabilities, although not compliant to the WPS standard. This product allows users to define a process containing logical sequences of geoprocessing operations supported by the server. This can be done using the user interface of tools such as ArcGIS Desktop[11] or by writing scripts that automate the execution of these operations. These geoprocesses can be published at the server and executed later by desktop, web and mobile clients. ESRI offers versions of ArcGIS for the most popular mobile platforms, including SDKs for developers to build their own applications.

Geoprocessing in ArcGIS Server has had more success than WPS-based implementations. In our opinion, one reason is that these geoprocesses are built using set of operations existing in the server that are well-known, tested and documented. Another reason is that they can be accessed using a simple, well-defined and documented REST API[12]. In contrast, consumers of WPS-based services cannot rely on the existence of any operation or well-documented interface. In addition, the input and output formats are usually described using complex schemas that are not easy to process or understand.

4 Framework Description

In this section we describe the WPS Mobile Framework (WMFW) designed to build WPS clients for mobile devices. WMFW aids developers by supplying a light-weight network communication library, as well as a code generator to build *adapters*. Adapters are in charge of transforming WPS messages, encoded in XML, into application objects in the business layer.

Figure 1 shows the structure of WMFW in the context of a full-fledged layered application, which contains layers for *user interface*, *business logic* and *communication*. It shows the main adapters generated for a common WPS application (*Capabilities adapter*, *Process descriptions adapter*, *GML adapter*), although adapters for other formats, e.g., raster formats, can be added independently. Note that as the figure shows the structure of an application that has been built using the framework it does not include the generator itself between its components. The first version of the framework has been targeted to the Android platform[13].

4.1 Network Communication Library

The network communication library has been reduced to the minimum size to make applications as light-weigth as possible. In order to do this, some trade-offs have been made:

- *Support only mandatory bindings of operations*: The WPS specification allows operation requests to be encoded using HTTP GET with KVP or HTTP POST with XML payload, but for each operation only one of them is required. For this reason we chose to implement only mandatory request encodings.
- *Prefer RawDataOutput to ResponseDocument as output response type*: Responses to *Execute* request can be returned as raw data or wrapped in an XML response document. By preferring the use of raw responses the size of the returned data is smaller, and the burden of parsing the XML response is also minimized.
- *Customized outgoing message packaging*: We have produced customized code to generate complex *Execute* requests (*GetCapabilities* and *DescribeProcess* requests are straightforward). *Execute* requests allow passing parameters as *literal data* (values that can be represented as strings), or *complex data* (complex application-specific data structures by value or by reference).

As a result, a very compact library with the needed functionality has been produced with a size of only 15 Kb.

4.2 Code Generator

The second component of the framework is a code generator, called *DBMobileGen*. We have utilised a generator designed to optimise code for resource-constrained platforms and targeted initially to Android-based devices [8]. *DBMobileGen* is a generic tool that has not been built specifically to be used by this framework. For this reason we only mention briefly its main features:

- *Use of XML documents to determine how schemas are used by the application*: This feature allows the simplification of XML schemas based on specific application needs and the use of techniques used to generate code a smaller size and memory footprint [7]. The main goal is to lower the number

[6]http://geotools.org/
[7]http://52north.org/
[8]http://udig.refractions.net
[9]http://www.openjump.org
[10]http://www.esri.com/software/arcgis/arcgisserver/index.html
[11]http://www.esri.com/software/arcgis/about/gis-for-me.html
[12]http://help.arcgis.com/en/arcgisserver/10.0/apis/rest/index.html
[13]http://www.android.com/

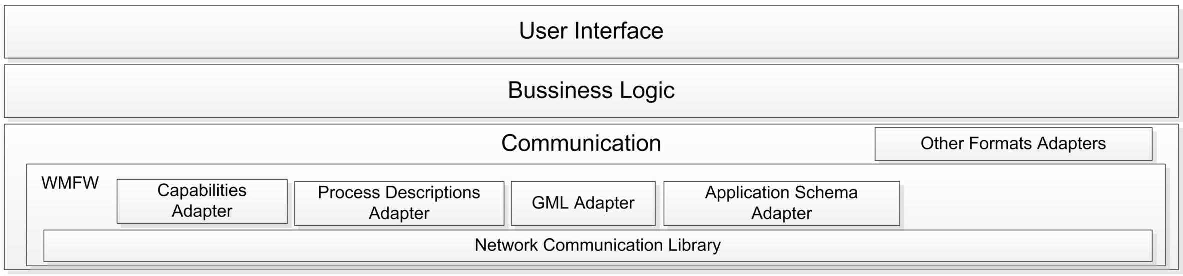


Figure 1: WPS Mobile Framework in the context of a layered application.

of classes in the final code, which will allow that it could be loaded faster by the JVM and it will consume less memory resources [2].

- *Disabling parsing or serialization operations as needed*: By selectively disabling unneeded operations final generated code can be optimized.
- *Ignore simple types facets*: Facets allow to restrict the values of simple types by specifying a set of constraints. This at the cost of having to check for valid values every time these values are handled.
- *Ignoring sections of the instance files*: Ignoring the unneeded portions of the file reduce memory and processing requirements of the applications.
- *Use of streaming APIs for low-level XML manipulation*: the use of streaming parsers offers the best performance for networking applications [3]. In our case we use *kXML*[14], an XML parser optimized for constrained environments.

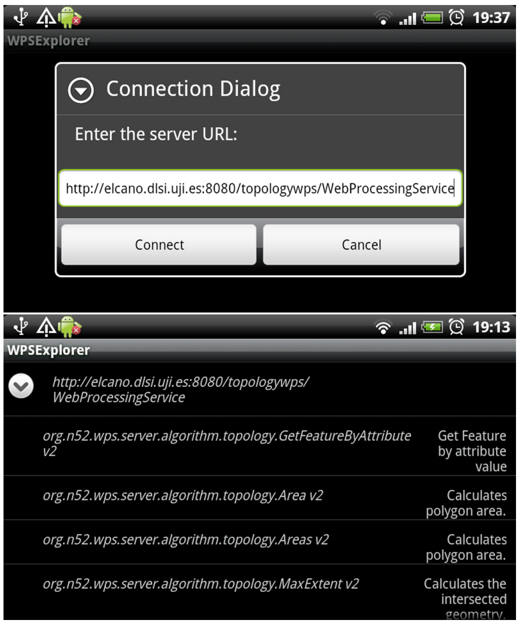
5 Sample Applications

To illustrate the use of WMFW, we present two applications developed for Android devices. Each application has different requirements and uses different parts of the WPS specification schemas. The first example is a *WPS Explorer*, which allows users to connect to a WPS server and to browse through the set of process descriptions published on the server. The second application is a simple geoprocessing client based on Google Maps that allows users to enter basic geometries and to execute different remote WPS operations to them.

5.1 WPS Explorer

The WPS Explorer allows users to browse the process descriptions of a WPS service. Figure 2 shows screenshots of the application. At the top, it shows the connection dialog where the user specifies the URL of the WPS server. At the bottom, after a connection is established, the list of processes published on the server is shown. The user may request detailed information about a given process by selecting it in the list.

To generate XML processing code we gathered a set of capabilities and process description files for the processes included by default in the following WPS servers: 52 North WPS Server[15],


19:37
WPSExplorer
Connection Dialog
Enter the server URL:
http://elcano.dlsi.uji.es:8080/topologywps/WebProcessingService
Connect
Cancel

19:13
WPSExplorer
http://elcano.dlsi.uji.es:8080/topologywps/WebProcessingService

org.n52.wps.server.algorithm.topology.GetFeatureByAttribute v2	Get Feature by attribute value
org.n52.wps.server.algorithm.topology.Area v2	Calculates polygon area.
org.n52.wps.server.algorithm.topology.Areas v2	Calculates polygon area.
org.n52.wps.server.algorithm.topology.MaxExtent v2	Calculates the intersected geometry.

Figure 2: WPS Explorer screenshots

Deegree WPS Server[16] and ZOO WPS[17]. Using these files as input, *DBMobileGen* determines that only 50 out of the 99 complex types in the schemas of WPS 1.0.0 are used by the application. The final size of the whole application in *Dalvik Executable (.dex) format* is only 163 KB.

5.2 Simple Geoprocessing Client

The second application is a client based on Google Maps that allows clients to enter simple GML (Geography Markup Language) [5] geometries that will being used as input to remote processes. The operations tested were *buffer*, *intersection*, and *area*, which are executed in 52 North WPS server. Figure 3 shows screenshots for the calculation of the area (top) and buffer (bottom) of geometries introduced by touching the device screen. The geometries are sent to the server on user request and the result is displayed as a different layer on the map.

If we capture some responses of these operations and use them as input to *DBMobileGen*, the resulting schemas are reduced in

[14]http://kxml.sourceforge.net/
[15]http://52north.org/maven/project-sites/wps/52n-wps-webapp/
[16]http://wiki.deegree.org/deegreeWiki/deegree3/ProcessingService
[17]http://www.zoo-project.org/

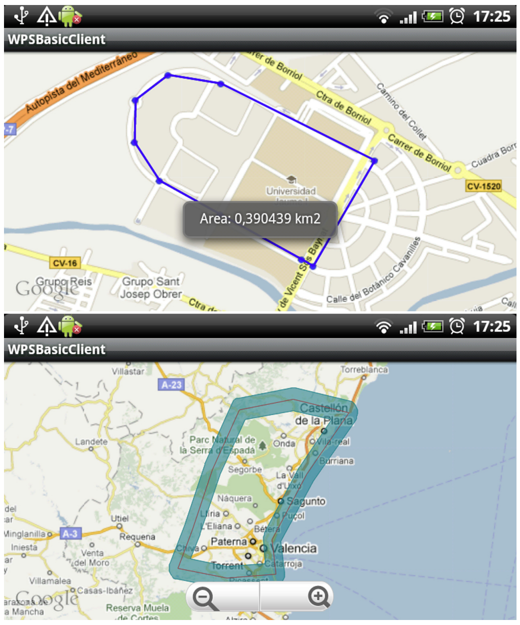


Figure 3: WPS Google Maps-based client

Table 1: #CT in full schemas vs. #CT used by the application

	Full schemas	Used	%
WPS	99	11	11.11
52NAS	395	18	4.55

a large degree. Being pragmatic in the case of the first two operations we supported only an application schema based in version 3.1.1 of GML (52NAS). Table 1 shows a comparison between the number of complex types (#CT) in the full schemas in which the application is based and the number that its really needed for its implementation.

The WPS schemas can be largely simplified. We must only process the subset needed to parse a simple *Execute* response document with a single literal value as result. The 52 North WPS application schema (52NAS) imports explicitly all the GML 3.1.1 schemas, but only references directly types *gml:AbstractFeatureType* and *gml:SurfacePropertyType*. In the XML documents returned by the server the type *gml:Polygon* was always used to describe the result of the operations. The final application size is similar than the previous one with around 160 KB.

6 Limitations, Challenges and Open Issues

WMFW, as currently implemented, presents some limitations:

- In the current implementation we only offer customized packaging for complex geospatial data based on the *GML-Packet schema* supported by the 52 North WPS server implementation.
- Code generation for serialization is not still supported.
- Only synchronous requests are supported.

During the development of WMFW and the sample applications we faced several challenges. The most important one was the size of XML schemas used to describe the encoding of the input and output data. For example, in the WPS basic client, the overall number of complex types for the whole schemas was almost 500. Considering the limited application functionality, this number is rather large. This problem was solved by extracting only the section of the schemas that are needed for the application using *DBMobileGen*. Although the process of using only sections of the schema can be accomplished using a manual approach, using a dedicated tool is a more time and effort saving approach. Another challenge was writing a light-weight communication library, which was accomplished by implementing only the required behaviour, and when several approaches for doing a task were available, by selecting the simpler one.

In addition to these challenges, we have identified some open issues in the topic of geoprocessing for mobile devices. The first issue is the integration of raster formats in WPS client mobile applications. The question of how to handle raster formats, which by general rule occupy a large disk space and require large processing capabilities is still an open issue. These images must also be handled in a way that allow users to change between different zooms levels without noticing a significant delay. The second issue is related with the use of code generation techniques for sections of the applications in the business logic and user interface layer. During the development of the second sample application common coding patterns for drawing geometries on the screen were noted, so a further look into these topics in order to generate larger sections of the final application might be interesting. The last open issue is the lack of a mechanism to now beforehand is we have enough memory or processing capabilities to handle a server response. When a request is issued to a server, there is no way to know whether the response will be too large or not to be handled by the device.

7 Conclusion

Development of WPS client applications has been mostly focus in desktop and web environment, with a much more modest insertion in mobile computing. For this reason, we have presented in this article the WPS Mobile Framework (WMFW), with the aim of building customized WPS clients targeted to mobile devices. The framework provides a light-weight communication layer and customized XML data binding code generation. XML processing code is generated in an application-specific manner, producing binary code with a low memory footprint and acceptable performance. At the communication level, the code is optimized by following a set of guidelines such as supporting only mandatory bindings of operations, processing server *Execute* responses in its raw form, and customizing the packaging of outgoing messages.

Acknowledgment

This work has been partially supported by the “España Virtual” project (ref. CENIT 2008-1030) through the Instituto Geográfico Nacional (IGN).

References

[1] T. Foerster and B. Schaffer. A Client for Distributed Geoprocessing on the Web. In J. Ware and G. Taylor, editors, *Web and Wireless Geographical Information Systems*, volume 4857 of *LNCS*, pages 252–263. Springer Berlin / Heidelberg, 2007.

[2] V.-M. Hartikainen, P.P. Liimatainen, and T. Mikkonen. On mobile java memory consumption. In *14th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing, 2006. PDP 2006.*, page 7 pp., feb. 2006.

[3] T.C. Lam, J.J Ding, and I.C. Liu. XML Document Parsing: Operational and Performance Characteristics. *Computer*, 41:30–37, 2008.

[4] F.J. López-Pellicer, R. Béjar, A.J. Florczyk, P.R. Muro-Medrano, and F.J. Zarazaga-Soria. A review of the implementation of OGC Web Services accross Europe. *International Journal of Spatial Data Infrastructures Research*, 6:168–186, 2011.

[5] OGC. OpenGIS Geography Markup Language (GML) Implementation Specification 3.1.1. *OGC Document*, (03-105r1), 2004.

[6] OGC. OpenGIS Web Processing Service 1.0.0. *OGC Document*, (05-007r7), 2007.

[7] A. Tamayo, C. Granell, and J. Huerta. Dealing with large schema sets in mobile SOS-based applications. In *Proceedings of the 2nd International Conference and Exhibition on Computing for Geospatial Research and Application (COM.Geo 2011)*, pages 16:1–16:9, New York, NY, USA, 2011. ACM.

[8] A. Tamayo, C. Granell, and J. Huerta. Instance-based XML data binding for mobile devices. In *Proceedings of the 3rd International Workshop on Middleware for Pervasive Mobile and Embedded Computing*, M-MPAC'2011, 2011.

[9] D.H. Zhang, B. Xie, H. Chen, Y. Ly, and L. Yu. Using geodata and geoprocessing web services in embedded device. In *2nd International Conference on Education Technology and Computer*, volume 1, pages 272–275, 2010.