RESTful Web Processing Service

Theodor Foerster[1], Andre Brühl[2], Bastian Schäffer[3]

[1]Institute for Geoinformatics (ifgi) - University of Muenster, Germany
[2]dynport GmbH, Hamburg, Germany
[3]52°North GmbH, Muenster, Germany
theodor.foerster@uni-muenster.de, andre.bruehl@dynport.de,
schaeffer@52north.org

ABSTRACT

User-generated content is currently stored and shared by so-called RESTful Web Services. To allow users to process their data in a seamless fashion a RESTful Web Service interface for web-based geoprocessing is required, which is presented in this article. The presented service applies a model to represent geoprocesses as resources. The service is applied in a comprehensive RESTful Web Service architecture to accomplish a humanitarian relief case. The implementation of the service is based on Free and Open Source tools.

1 INTRODUCTION

Web-based geoprocessing has created a lot of attention in research and industry over the past years as for instance documented by Brauner et al. (2009). It enables users to retrieve information on the Web instead of plain data. The Web Processing Service (WPS) is an attempt by the Open Geospatial Consortium (OGC) to provide a web service interface for web-based geoprocessing (OGC 2007).

Currently, Representational State Transfer (REST) (Fielding 2000) evolves as a new paradigm on the Web and is currently used for designing web services providing geographic data and maps (e.g. Google, GeoCommons). At OGC, REST only received little attention (OGC 2010). A service for processing this kind of data based on REST has not been proposed yet. Designing such a service is linked to the question of how to represent processes as resources as identifying the resources is a key concept in REST.

In this article, we will describe such a RESTful WPS, which will be designed along the interface functionality of OGC´s WPS specification, as a popular representative for providing web-based geoprocessing functionality. The article demonstrates the applicability of the RESTful WPS based on a RESTful architecture analyzing user-generated content for a humanitarian relief case in Haiti. The presented implementation uses software components available through Open Source licenses (Ruby on Rails and PostGIS).

In Section 2 we will describe related work about web-based geoprocessing and REST. Section 3 will present the interface of the RESTful WPS. Its implementation is described in Section 4. The implementation is applied to the use case of humanitarian relief in Haiti (Section 5). The article ends with a conclusion and a discussion regarding future standardization work and further research.

2 RELATED WORK

This section describes web-based geoprocessing and the principles of REST as applied in this article.

2.1 Web-based Geoprocessing

Geoprocessing is the application of functionality representing real-world processes (e.g. hydrological runoff models) or transformation of geodata (e.g. generalization, (coordinate) transformation). Providing these models and functionality on the web is a relevant topic in research and industry. Kiehle, Greve, and Heier (2006) identified web-based geoprocessing as the next step towards information in Spatial Data Infrastructures.

To support interoperability of available web-based geoprocessing functionality, the OGC specified the Web Processing Service interface (OGC 2007). The WPS interface describes three operations: *GetCapabilities* for service metadata retrieval, *DescribeProcess* for process metadata retrieval and *Execute* for performing the specific process with the desired parameters. Moreover, the interface provides mechanisms to process data supplied as web-accessible references and to perform the process asynchronously. The communication is based on HTTP GET with a key value pair encoding and HTTP POST with an XML encoding. Different client applications are available such as the desktop-based client described by Schaeffer and Foerster (2008).

2.2 Representational State Transfer

REST is not considered to be a standard, but a style of Web Service design. Fielding (2000) describes REST as:

"... a hybrid style derived from several of the network-based architectural styles [...] and combined with additional constraints that define a uniform connector interface."

In particular, properties of REST are addressability, statelessness, connectedness (architectural styles) and a uniform interface. A Web Service, which implements these properties, is called RESTful (Richardson and Ruby 2007). These properties are realized by using Hypertext Transfer Protocol (HTTP) and Uniform Resource Identifiers (URIs) as common protocols of the Web.

Besides, the resource is a central aspect of REST. A resource is a specific entity in the system, which is represented by a URI. Designing the resource correctly regarding the granularity and the purpose of the service is important. All operations of a RESTful Web Services are represented by the combination of a resource and HTTP. The vocabulary of HTTP describes already some operational semantics, which can be used to interact with the resources:

- GET - retrieve a resource
- POST - create a new resource
- PUT - update a resource
- DELETE - remove a resource.

So instead of conventional Web Services, which are based on SOAP, or OGC XML encoding and are framed by a Service-Oriented Architecture (SOA) (Alonso et al. 2004), no arbitrary names for specific methods are required. So the combination of HTTP verbs and URIs is considered to be a clear advantage over these conventional Web Services, as HTTP verbs contain already a lot of semantics, which provide a common understanding throughout the user community. In case of a RESTful Web Service, the user does not need to read additional specifications for a particular service. Therefore, RESTful Web Services can be considered to be light-weight regarding their interface.

Clients can retrieve these resources as specific representations identified by the MIME-type. The type of representation can be negotiated with the service through HTTP functionality, which makes the resource flexible to the client. Examples of representations are HTML, XML, JSON, ATOM or any arbitrary format.

Additionally, these resources are connected through links (represented by URIs) for browsing through the resources of a RESTful Web Service. This so-called hypermedia approach[1] allows users to perceive the functionality of the specific service faster. An example of a resource and its representation in ATOM with additional links to other resources is depicted in (Listing 1, Section 3).

2.3 REST for Geographic Applications

There are various providers and implementations of RESTful Web Services available. Examples of implementations are FeatureServer, GeoREST, PostGIS RESTful Web Service, REST GeoServer extension.

Also geographic content is available through RESTful Web Services such as: Google, Yahoo (both providing maps used as base layers), OpenStreetMap (providing maps and data about roads and POIs), GeoCommons (providing any geographic data). In particular, GeoCommons[2] allows users to store their data and to share it with others regarding any geographic theme. GeoCommons provides an interface to query the available data through keywords or geographic extent.

[1] sometimes also called hypermedia as the engine of application state (HATEOAS).
[2] GeoCommons website: http://geocommons.com/.

Also research about RESTful Web Services for geographic applications has been carried our lately. For instance Janowicz et al. (2010) describe an URI encoding for publishing sensor data as Open Linked Data. The URI scheme is used in a RESTful web service for publishing sensor data. Mazzetti, Nativi, and Caron (2009) discuss REST for publishing coverages in comparison to the existing OGC Web Coverage Service interface. They come to the conclusion, that geographic applications (mostly SOA-based) can largely benefit from a RESTful web services by integrating them.

The OGC adopted REST as one part in the Web Map Tiling Service (WMTS) specification (OGC 2010). The WMTS interface provides a mechanism to access discrete pre-rendered map tiles in a standardized way for caching of tiles and performance reasons. Similar approaches but proprietary approaches are used by Google and Yahoo for their RESTful services. The WMTS is also a complement to the existing Web Map Service. The specification is split in two parts defining a) the resources and b) the concrete exchange mechanisms (REST or SOAP).

Lately, ESRI published a whitepaper on a comprehensive list of interfaces for the so-called REST GeoServices (ESRI 2010). This whitepaper features different services of which one is the so-called GPService for offering web-based geoprocessing functionality through REST. The whitepaper specifies light-weight interfaces, which are adopted by client and service providers and thereby establish a de-facto standard (comparable to shapefile). The interface of the GPService uses for instance ESRI´s own interpretation of JSON data encoding. Moreover, the interface is not completely following the concept of resources, but uses names for operations (e.g. GPServer/{process_ID}execute). Additionally, the whitepaper does not describe how to link different resources (i.e. the process list contains the process names, instead of the links to the process).

3 DESIGN OF A RESTFUL WEB PROCESSING SERVICE

This section presents the design of the RESTful WPS.

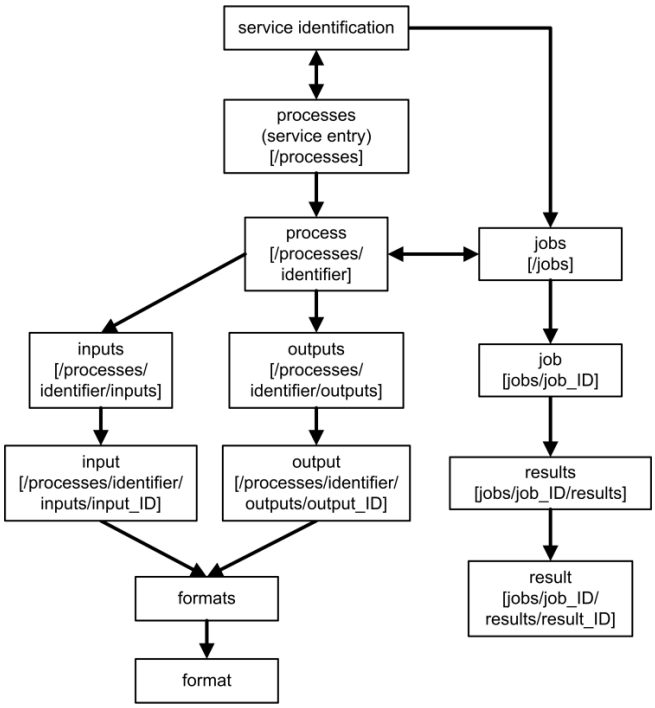

Figure 1: The resources offered by the RESTful WPS (links between the resources are indicated by arrows along which the user can navigate).

As expressed in Section 2.2, a RESTful Web Services consists of a set of resources. They are the points of interaction for the clients. The structure of the resources for the RESTful WPS is depicted in Figure 1. The resources are interconnected with links for browsing the service (indicated by the

arrows in the figure) and to help the user to explore the service. The different resources are presented to the client in an applicable format based on content negotiation.

The entry point for the service is the list of all available processes. Each process is a resource and has inputs and outputs as resources, which contain specific formats. Besides that, to support asynchronous communication, also jobs are available as resources. A particular job contains a set of results.

On the web, these resources are represented as URIs. Based on the presented structure of resources some URIs are described exemplary:

- /processes represents the list of processes
- /processes/{identifier} represents the process for the given identifier (process name)
- /jobs represents all processed and running jobs (available only to logged in users)
- /processes/{identifier}/jobs represents all processed and running jobs for one particular process (available only to logged in users).

Describing production rules for these URIs in EBNF are not required, as the resources are linked for service discovery and interaction.

An example of such a resource for representing a buffer process is described in Listing 1. The resource is represented in ATOM and contains links as URIs to other resources such as the input. The representation of the resource depends on the client (e.g. HTML, ATOM, OGC WPS encoding).

```
<entry xmlns="http://www.w3.org/2005/Atom">
 <identifier>buffer</identifier>
 <title>Buffer - ...</title>
 <abstract>Returns a geometry that represents ...</abstract>
 <tags>GeometryProcessing</tags>
 <link href="http://api.gisonrails.de/processes/buffer" rel="self" type="application/atom+xml"/>
 <link href="http://api.gisonrails.de/processes/buffer" rel="alternate" type="text/html"/>
 <link href="http://api.gisonrails.de/processes/buffer/jobs" rel="jobs" type="application/atom+xml"/>
 <link href="http://api.gisonrails.de/processes/buffer/inputs" rel="inputs" type="application/atom+xml"/>
 <link href="http://api.gisonrails.de/processes/buffer/outputs" rel="outputs" type="application/atom+xml"/>
</entry>
```

Listing 1: Example of a process resource represented in ATOM retrieved from a RESTful WPS.

Based on the resources and their URIs identified, different HTTP verbs are attached to model the RESTful WPS. The resources describe the information, whereas the HTTP verbs define the interaction with the resource (Section 2.2).

Modeling processes as distinct resources identifiable as a URI (i.e. without additional key value pairs) is difficult, as the number of potential process configurations (i.e. the resources) is unlimited and unknown prior to execution. Therefore, requesting the URI of a process resource through HTTP GET returns its metadata. The specific configuration is then represented via key value pairs (parameters) and sent via HTTP GET (/processes/{identifier}?{parameters}).

To perform a process asynchronously, a job resource needs to be created via HTTP POST on /processes/{identifier}/jobs. The new job resource can then be queried (/processes/{identifier}/jobs/{job_id}) via HTTP GET. When finished, this job resource contains links to the results, which can be retrieved via HTTP GET. Finally, the job or its results can be removed from the service using HTTP DELETE.

The presented design reflects the interface functionality of the OGC WPS. For instance, the processes are discovered stepwise in the presented design following the linked resources (Figure 1). The identified resources are inline with the elements of OGC WPS specification. Contrarily, the transition between retrieving service metadata and process metadata is achieved through URIs linking the resources explicitly. In OGC WPS, retrieving service and process metadata is modeled as distinct operations (GetCapabilities and DescribeProcess). The client needs to know the specification and the workflow of process discovery.

4 IMPLEMENTATION

The RESTful WPS is implemented with free and open source software. In particular, it is implemented with Ruby on Rails using the Model View Controller paradigm (Gamma, Helm, and Johnson 1995) as shown in Figure 2. The model uses a database (PostGIS) to perform the geoprocessing functionality and to store the created resources (the data is always provided by the client as in a WPS). The view creates the different representations (e.g. ATOM, OGC WPS XML and

HTML) of these resources (Section 3). The transition between the model (resources) and the view (representation) are handled by the controller as well as the incoming requests.

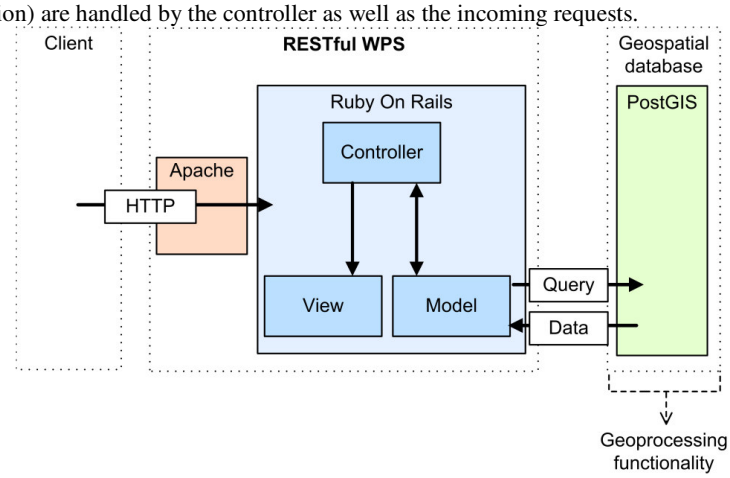

Figure 2: The implemented architecture using Model View Controller paradigm.

To integrate resources of different RESTful Web Services, a browser-based client has been created. The client uses RESTful Web Services of GeoCommons for data discovery and retrieval, mapping functionality (also RESTful) from Google or OpenStreetMap respectively and the RESTful WPS for performing processing (Figure 3). Based on the seamless integration, it is possible to create transparent workflows (Alameh 2003) by invoking step-wise different web services and use their output as input for other web services (see also the use case in Section 5). During this course of action, the data resides on the different web services and is shipped between the services in an optimal way (client and services can choose the most suitable format for a specific resource using content negotiation).

The overall user experience with the browser-based client is based on drag & drop interaction. The resources of GeoCommons are queried on the left side of the browser-based client (Figure 3) through keywords and geographic extent. The queried resources can directly be added to the map view. The user queries the available processes and configures a specific process with the resources, which have been added to the map view (searched and integrated from GeoCommons). Finally, the processed resources are available for external applications (such as Google Earth) using the different representations provided by the RESTful WPS (e.g. KML, GML).

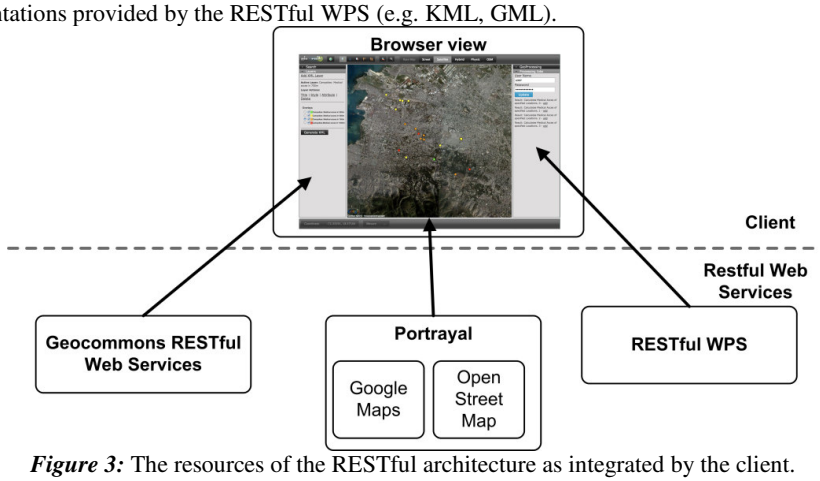

Figure 3: The resources of the RESTful architecture as integrated by the client.

5 HAITI USE CASE FOR HUMANITARIAN RELIEF

Regarding the use case, after the earthquake in Haiti in 2010, crisis mappers played a significant role in supporting humanitarian aid by collecting geographic data and maps (Zook et al. 2010). Such content is available in GeoCommons through RESTful Web Services. For creating relief maps, processing functionality as available in a RESTful WPS is required. In the presented workflow, a map

is created analyzing the campsites regarding their medical access based on the user-generated data such as available in GeoCommons. Additionally, the workflow is inspired by Foerster and Schaeffer (2010).

The workflow is performed using the browser-based client (Section 4). In particular, the workflow uses data for medical facilities and available camp sites in Port-au-Prince, Haiti. Using processes for buffer, intersection and symmetric difference, it is possible to rate the available camp sites according to their accessibility of medical facilities. The workflow and its intermediate results are depicted in Figure 4.

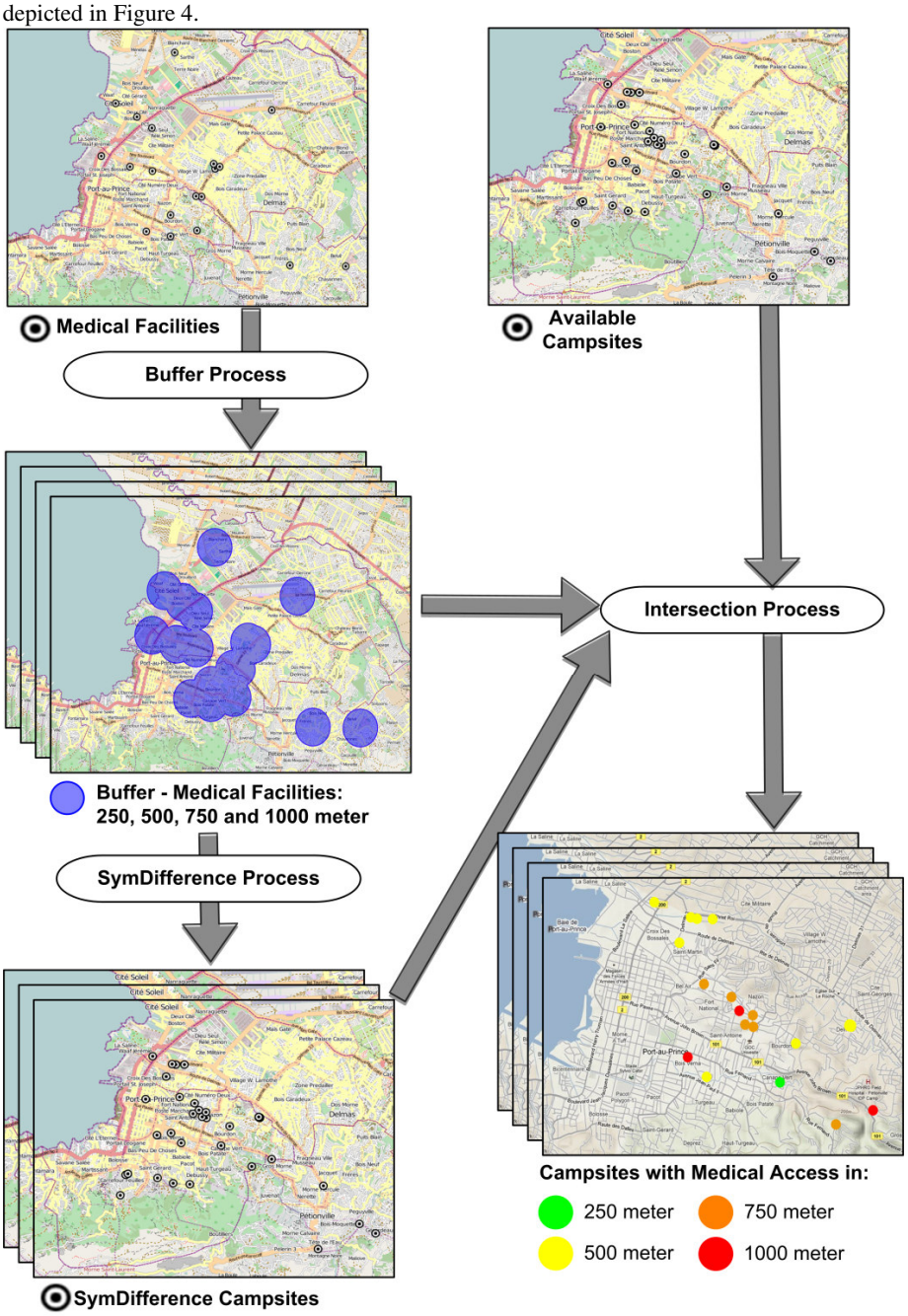

Figure 4: Workflow to extract medical sites close to Port-au-Prince, Haiti.

6 DISCUSSION AND CONCLUSION

This article presents a RESTful WPS to enhance existing RESTful architectures for geographic applications with geoprocessing functionality. The RESTful WPS can be integrated seamlessly into the RESTful architecture as the added service did not require the existing architecture to change using the common architectural style of REST. Thereby, the design of the RESTful WPS (Section 3) can be considered RESTful regarding the following properties (Section 2.2):

- addressability - each resource is addressed through a URI and can be used later (e.g. completed processing jobs).
- statelessness - the service does not need to maintain the state of the client, while the user is interacting with the resources.
- connectedness - the resources are linked and it is possible to browse through the available resources.
- uniform interface - RESTful WPS interface uses HTTP and URIs.

However, it became clear that the presented RESTful WPS breaks the current OGC model (Section 3). Metadata as well as the process execution are modeled as resources and thereby different from the WPS specification. To overcome these limitations, the OGC implementation specifications have to be created in a more modularized way to reflect different architectural styles (e.g. SOAP, REST) in a common model. This paper presents a first attempt how to model geoprocessing functionality as resources and a valuable input for future specifications at OGC. Overall, the design of the RESTful WPS is as powerful as the OGC specification, but with no well-defined response structure, as this is handled by the different representations. So it appears to be more flexible regarding encoding requirements of different applications.

The implementation (Section 4) has been successfully applied as proof-of-concept to a real-world scenario for creating maps supporting humanitarian relief of Haiti (Section 5). The presented browser-based client is able to perform transparent workflows. Moreover, the implementation is based on state-of-the-art Open Source frameworks.

Future research will investigate a transactional RESTful WPS using the full capability of HTTP verbs (e.g. use HTTP PUT to upload processes). Such a transactional version of a geoprocessing service has been initially described by Schaeffer (2008) for the example of OGC WPS. Additionally, the proposed structure of resources for processing services can be applied to research in Open Linked data, as for instance applied for the case of sensors (Janowicz et al. 2010).

ACKNOWLEDGEMENTS

The presented work has been performed as part of the Diploma thesis research by Andre Brühl at the University of Muenster.

REFERENCES

Alameh, Nadine. 2003. Chaining Geographic Information Web Services. *IEEE Internet Computing* 07, no. 5 (September): 22-29. doi:10.1109/MIC.2003.1232514.

Alonso, Gustavo, Fabio Casati, Harumi Kuno, and Vijay Machiraju. 2004. *Web Services*. 1. ed. Springer Verlag.

Brauner, Johannes, Theodor Foerster, Bastian Schaeffer, and Bastian Baranski. 2009. Towards a Research Agenda for Geoprocessing Services. In *12th AGILE International Conference on Geographic Information Science*, ed. Jan Haunert, B. Kieler, and J. Milde. Hanover, Germany: IKG, Leibniz University of Hanover. http://www.ikg.uni-hannover.de/agile/fileadmin/agile/paper/124.pdf.

ESRI. 2010. *GeoServices REST specification*. White paper. New York, USA: ESRI Inc. http://www.esri.com/library/whitepapers/pdfs/geoservices-rest-spec.pdf.

Fielding, Roy Thomas. 2000. Architectural Styles and the Design of Network-based Software Architectures. UNIVERSITY OF CALIFORNIA, IRVINE. http://www.ics.uci.edu/fielding/pubs/dissertation/top.htm.

Foerster, Theodor, and Bastian Schaeffer. 2010. *OWS-7 Feature and Statistical Analysis Engineering Report*. OGC Engineering Report. OGC.

Gamma, Erich, Richard Helm, and Ralph E. Johnson. 1995. *Design Patterns. Elements of Reusable Object-Oriented Software*. Addison-Wesley Longman.

Janowicz, K., A. Broering, C. Stasch, and T. Everding. 2010. Towards Meaningful URIs for Linked Sensor Data. In *Proceedings of the Workshop "Towards Digital Earth: Search, Discover and Share Geospatial Data 2010"*, ed. A. Devaraju, A. Llaves, Patrick Maue, and Carsten Kessler, 6. Berlin, Germany. http://ceur-ws.org/Vol-640/paper2.pdf.

Kiehle, Christian, Klaus Greve, and Christian Heier. 2006. Standardized Geoprocessing - taking spatial data infrastructures one step further. In *9th AGILE International Conference on Geographic Information Science*, 273-282. Visegrad, Hungary.

Mazzetti, P., S. Nativi, and J. Caron. 2009. RESTful implementation of geospatial services for Earth and Space Science applications. *International Journal of Digital Earth* 2: 40–61.

OGC. 2007. *OpenGIS Web Processing Service*. OGC implementation specification. Open Geospatial Consortium. http://www.opengeospatial.org/standards/wps.

OGC. 2010. *Web Map Tile Service Implementation Standard*. Implementation standard. Open Geospatial Consortium Inc. http://www.opengeospatial.org/standards/wmts.

Richardson, L., and S. Ruby. 2007. *RESTful web services*. O'Reilly Media, Inc.

Schaeffer, Bastian. 2008. Towards a Transactional Web Processing Service (WPS-T). In *Proceedings of the 6th Geographic Information Days*, ed. E. Pebesma, M. Bishr, and Thomas Bartoschek, 32:91-116. IfGI prints. Muenster, Germany: Institute for Geoinformatics.

Schaeffer, Bastian, and Theodor Foerster. 2008. A Client for Distributed Geo-Processing and Workflow Design. *Journal for Location Based Services* 2, no. 3 (September): 194-210. doi:10.1080/17489720802558491.

Zook, Matthew, Mark Graham, Taylor Shelton, and Sean Gorman. 2010. Volunteered Geographic Information and Crowdsourcing Disaster Relief: A Case Study of the Haitian Earthquake. *World Medical & Health Policy* 2, no. 2: 7-33. doi:10.2202/1948-4682.1069.