

Article

The Semantics of Web Services: An Examination in GIScience Applications

Xuan Shi

Department of Geosciences, University of Arkansas, Fayetteville, AR 72701, USA;
E-Mail: xuanshi@uark.edu; Tel.: +1-479-575-7906; Fax: +1-479-575-3469

*Received: 20 July 2013; in revised form: 19 August 2013 / Accepted: 11 September 2013 /
Published: 23 September 2013*

Abstract: Web service is a technological solution for software interoperability that supports the seamless integration of diverse applications. In the vision of web service architecture, web services are described by the Web Service Description Language (WSDL), discovered through Universal Description, Discovery and Integration (UDDI) and communicate by the Simple Object Access Protocol (SOAP). Such a divination has never been fully accomplished yet. Although it was criticized that WSDL only has a syntactic definition of web services, but was not semantic, prior initiatives in semantic web services did not establish a correct methodology to resolve the problem. This paper examines the distinction and relationship between the syntactic and semantic definitions for web services that characterize different purposes in service computation. Further, this paper proposes that the semantics of web service are neutral and independent from the service interface definition, data types and platform. Such a conclusion can be a universal law in software engineering and service computing. Several use cases in the GIScience application are examined in this paper, while the formalization of geospatial services needs to be constructed by the GIScience community towards a comprehensive ontology of the conceptual definitions and relationships for geospatial computation. Advancements in semantic web services research will happen in domain science applications.

Keywords: semantic; web services; GIScience

1. Introduction

In English, service means “*work done for others*” [1], “*contribution to the welfare of others*” [2] or “*work done for somebody*” [3]. For any *others* who want to consume “services”, the prerequisite is that they can find the required services. If such a prerequisite cannot be satisfied, then there is no service at all for the *others*. In 2007, Michael Brodie wrote, “How will services discover the right services that meet specific requirements in such an environment? We’re far from that scale at the moment.” [4]. If we are far from discovering the right services, then how can we use them in varied service-related research and development of the past few years? Earlier, in 2005, Ian Foster commented in *Science* (“Service-Oriented Science,” 6 May 2005) that “(Web) services have little value if others cannot discover, access and make sense of them” [5], which also means, service discovery seems to be the first order for others to deploy useful services. In essence, service discovery is the key and starting point in service computation.

Why are we still far from discovering the right services? When developing so-called web services technology, it was promised that when a service is defined in Web Service Description Language (WSDL), a service provider can publish the service into a service registry (such as Universal Description, Discovery and Integration (UDDI)), and a service requester can find the service through UDDI and consume and invoke the service through the Simple Object Access Protocol (SOAP). World Wide Web Consortium (W3C) (2004) proposed three approaches for service discovery, which include service registry, index and peer-to-peer (P2P) [6]. P2P is impossible, because under the existing web service technology, a service only has an interface definition specified in WSDL, while a message can be exchanged through the SOAP protocol. There is no mechanism for a “service” to query its neighbors in search of a suitable web service. The index approach is not feasible either and encounters varied challenges and problems [7]. When the UDDI Business Registry (UBR) was closed in early 2006 [8], there was no successful way for others to discover the services, let alone the right services. In practice, there may be certain progress in discovering data. In this paper, however, a service is a software module by definition. Data discovery may not be the focus of this paper.

For a variety of reasons, service description having little semantics should be the key issue. However, what the semantics of web services are, a fundamental scientific question, has not been clarified for a long time. This paper starts the discussion about the concept of web services, trying to mitigate the gap between SOAP-based and Representational State Transfer (REST) web services. A unified service discovery can be synthesized by targeting the crux of web services as the functional modules of software. It can be explained (1) why the semantics of web services are neutral and independent from the syntactic application programming interface (API) definitions; (2) why the semantics of web services have no relationship with who develops the services; and (3) why not every concept in the semantic definitions has a corresponding syntactic counterpart. Such general rules establish the scientific principles for constructing the semantics of web services and can be examined and validated by different domain science applications. A prototype is proposed to explore approaches for the formalization of geospatial services towards a comprehensive ontology of the conceptual definitions and relationships for geospatial services and computational modules that will also be useful to GIS software engineering and GIScience education. An ontology-based geospatial knowledge framework will help to organize software modules and tools in a more meaningful, or semantic, way in

GIS software products. As a result, meaningful GIS software will help students to understand the concepts in GIScience and to find appropriate tools and modules to complete different tasks for data analysis and processing. The proposed rules and the formalized geospatial services can be implemented through a sustained registry system that has a cybernetic mechanism to enforce the syntactic and semantic specifications, while the proposed methodology can be extended to other domain science applications.

2. The Chaos in the Concept and Description of Web Service

In the vision of web service architecture, or service-oriented architecture, there should be a mechanism or component that allows service providers to publish their services and enables service requesters to find the required services. The concept of web service, however, could be interpreted with different facets, while the description of web services has been criticized as not semantic. Although ontology and semantic research is about the concepts and relationships between the concepts, in service-oriented computing, if the core concepts and description of web services are not clear, what can be discovered could be a puzzle. Understanding the chaos in the conceptual framework of web services is helpful to establish a common ground for discussion and to explore potential solutions. This section first reviews the different concepts of web services to clarify the central theme discussed in this paper. The chaos in the syntactic definition and description of web services is then elaborated upon to help understand why service discovery may not be successful based on the syntactic objects and modules defined in WSDL. Such a discussion is fundamental to examine what the semantics of web services are and is elaborated upon in the following section.

2.1. The Concept of Web Service

Scientific research should have a precise and consistent conceptual framework. Although “service”-related research and development would have been the fashion in academia and industry, such research may not have much academic rigor when it is started from a tricky, but imprecise, concept. Throughout the course of technology development and evolution, what has to be stressed is that web service, as a descendant of Remote Procedure Call (RPC), originally was a solution to software interoperability, taking the place of Common Object Request Broker Architecture (CORBA) and Distributed Component Object Model (DCOM). Unfortunately, the concept itself is ambiguously related to “web”, although the technology might even have no relationship to the web at all, because we can deploy the technology in standalone applications without the need for the web.

Roy Fielding (2000) [9] proposed the concept of Representational State Transfer (REST) to describe an architectural style of networked systems. He explained—“Representational State Transfer is intended to evoke an image of how a well-designed web application behaves: a network of web pages (a virtual state-machine), where the user progresses through an application by selecting links (state transitions), resulting in the next page (representing the next state of the application) being transferred to the user and rendered for their use.”[9]. In science, REST is about web application or web pages, according to Roy’s dissertation. REST service is invoked through URL definition using HTTP GET|POST methods.

Inevitably, the result of service-related research and development is doomed to be ambiguous and is full of chaos or paradoxes without a precise concept. We have been seeing the debate between SOAP and REST services since the early years of the decade. Later on, the concept of “web services” was transformed as services on the web. Then, such “services” could be web sites and web applications, since they offered “services” to the users who use the web for various purposes, although SOAP-based services, the descendant of RPC, could be included in such a broad category of “services”. Furthermore, we can find many other categories of “services” [10] without surprise.

The concept of “web services” has been chaotic for more than a decade when SOAP-based and REST services originated in early 2000. Besides the different meaning in SOAP-based and REST services, web applications could be called “web services” [11]. It is not unusual to see the argument that a web-based service is not a real web service or a web service can be interpreted as a service on the web [12]. In the world of “semantic web services”, “semantic-web service” is obviously different from “semantic web-service” when the hyphen is placed differently. In scientific research, however, if we start from an ambiguous concept, how can we understand what each other would have been talking about and derive a clear and correct conclusion?

When World Wide Web or W3C defined the concept of “web services”, in contrast to many other things, W3C [13] had to claim:

- There are many things that might be called “web services” in the world at large. However, for the purpose of this working group and this architecture, and without prejudice toward other definitions, we will use the following definition:
- A web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically, WSDL). Other systems interact with the web service in a manner prescribed by its description using SOAP-messages, typically conveyed using HTTP with an XML serialization in conjunction with other web-related standards.

In fact, W3C’s statement means those many things may *not* be web services at all. The chaos at the conceptual level increases the difficulties in service discovery. *Considering both SOAP-based and REST services can trigger the same functions that implement (geospatial) computation over the computer network, there is a need to unify the service discovery process that can cover both kinds of services.* In fact, if the service is examined as a functional module of GIS software, for example, then the gap and debate between SOAP-based and REST services can be mitigated and synthesized. No matter what protocol a service is implemented in, *a service*, or functional module, has a syntactic framework following the software engineering lifecycle to define the inputs, outputs, preconditions and effects (IOPEs) [14,15] of the function or service. Obviously, the syntactic IOPE definition is utilized for service invocation, not for service discovery.

2.2. The Chaos in the Syntactic Description of Web Services

Although WSDL is called the description language for web services, it describes the syntactic definition for the application programming interfaces (APIs) of the services covering the information of IOPEs for certain functional modules within the service. Since WSDL has direct correlation with

the object-oriented programming paradigm, the definitions in WSDL are correspondent to the data, data type, function, objects, *etc.*, in the functional modules. Since such definitions can be named randomly, it can be concluded that the *same* service provider can use *different* service interfaces defined in WSDL documents even when developing the *same* kind of service. On the other hand, it is possible that a *different* service provider can use the *same* service interfaces defined in WSDL documents even when developing *different* kinds of services. For this reason, varied syntactic descriptions of services are actually full of chaos that prevents successful service discovery. When such chaotic service description documents were freely and randomly published into a service registry, such as UDDI, which is directly correspondent to the WSDL definitions, the failure of service discovery was inevitable.

In GIScience practice, ESRI, the world leader in GIS software and technology development, provides geocoding web services in multiple versions. Initially, its address geocoding web services (<http://arcweb.esri.com/services/v2/AddressFinder.wsdl>) had a function named *findAddress*, which returns an object called *LocationInfo*. In a newer version, the exactly same function (<http://www.arcwebservices.com/services/v2006/AddressFinder.wsdl>) was renamed *findLocationByAddress*, which returns an object called *GeocodeInfo*. This can be good evidence to support the above conclusion that the *same* service provider can use *different* service interfaces defined in WSDL documents even when developing the *same* kind of service.

Given the other example, while *geocoder.us* also provides geocoding web services in different versions, such as <http://rpc.geocoder.us/dist/eg/clients/GeoCoder.wsdl> and <http://geocoder.us/dist/eg/clients/GeoCoderPHP.wsdl>, its service interface definition is different from ESRI's address geocoding web services. Such a fact demonstrates that *different* service providers can generate *different* service interfaces defined in WSDL documents when developing the *same* kind of service.

In the case of REST services, structured URLs can be composed to invoke a service through the HTTP GET method. In this case, the URL is followed by the input data or information defined within or without the JavaScript Object Notation (JSON) format, for example. The variables used in composing the URL can be randomly defined, as well, which frequently result in confusion unless detailed documentation is provided. When the same function is implemented by using the HTTP POST method, the input data may not be attached in composing the URL. The same chaotic situation described in the above WSDL/SOAP-based services can be repeated in REST services. In conclusion, the content in the syntactic definition of either SOAP-based or REST service has nothing to do with the semantics of web services.

3. The Semantics of Web Services in Semantic Web Services

What are the semantics of web services? This question seems untimely after the research initiative of semantic web services (SWS) has even been evolving with a large amount of publications for more than a decade. This section first proposes a new definition that is completely different from prior works and is followed by a brief examination over prior initiatives in semantic web services in the past decade to help understand the distinctions. It can be clarified that dynamic service invocation may be not relevant to the semantic issue, though it was claimed that semantic web service will help dynamic service invocation.

3.1. The Semantics of Web Services

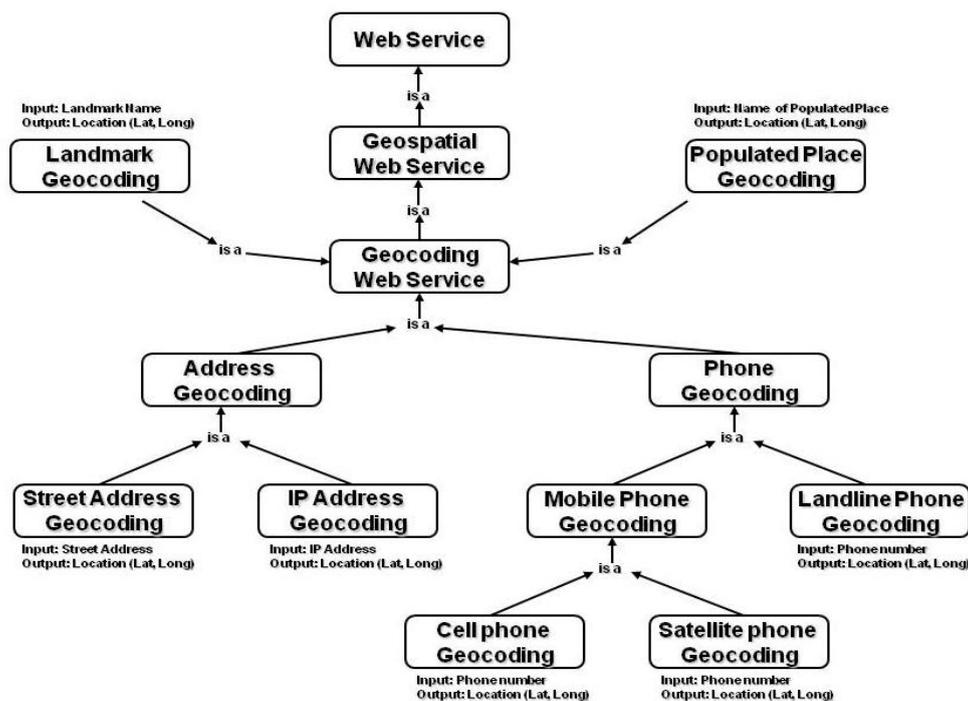
Today, the question about what the semantics of web services are can be answered and interpreted in a completely different vision. In fact, the answer is pretty straightforward. If *the semantics in the semantic web* are about *the concepts and relationship between the concepts*, then in web services, *the semantics of web services* should be about *the concepts of services and the relationship between services*. In GIScience applications, the concepts of services are those common terms that are used to call a GIS function as the common knowledge developed by GIS scientists and GIS professionals. Since many concepts have been developed for decades, but not well-organized, the same GIS function may be called by different terms or the same term may have different interpretations. The research on semantics of web services will explore a formal way to organize the concepts in a more comprehensive framework to understand the relationship between the concepts of geospatial functions and services.

This interpretation is different. because in the previous works, the semantics of web services are about the *components* in the syntactic definition of the service described in WSDL, including the IOPEs, along with other information, such as the service provider's profiles. At the beginning of this initiative, SWS research in computer science and mainstream IT had mainly focused on varied business transaction models, such as "buying books", "buying airline tickets", "reserving hotels and cars", *etc.* When such services were used in (semantic) web applications, the API or IOPE definitions in WSDL were meaningless. As a result, how to annotate the service APIs or IOPEs by semantic web technology was critical in the composition process in order to chain services to accomplish the online business transactions.

Obviously, few researchers, if anybody, care about the relationship between the web services of "buying books", "buying airline tickets" and "reserving hotels and cars". Indeed, if only such business scenarios are involved, a few discrete business concepts are enough in these simple and monotonous use cases. However, if we examine the SWS framework within a systematic knowledge domain, such as GIScience, we can drive a completely different conclusion. Figure 1 is intuitive and straightforward to explain some *general rules* to understand and re-construct the conceptual framework to formalize the ontology and semantics of web services:

1. Service semantics are neutral and independent of the syntactic API definitions (IOPEs).
2. The semantics of web services have no relationship with who develops the services.
3. Not every concept in the semantic definitions has a corresponding syntactic counterpart.

Figure 1 describes a taxonomic architecture of the semantics of geocoding web services. Geocoding is a common function in GIS. This taxonomy contains hierarchical concepts of web services that perform the *geocoding* function, which derives the location (latitude and longitude) information from different input objects that can be a *landmark*, a *populated place*, a *street address* or an *IP address* or a *phone number*, while such concepts can be classified into many sub-categories.

Figure 1. The taxonomic architecture of the semantics of geocoding web services.

The *syntactic* API definition is obviously different from the *semantic* concept of services, because the APIs are about the IOPEs, such as input and output variables, data formats and function names. As discussed above, ESRI has offered several versions of street address geocoding services. All versions of services perform exactly the *same* function, but each of them has a *different* API [16,17]. Meanwhile geocoder.us offers the *same* geocoding service using a *different* API [18], while Google and Yahoo! offer this geocoding function online through the REST approach.

It can thus be concluded that *the semantics of web service are neutral and independent of the syntactic API definitions (and IOPEs) and have no relationship with who develops the services*. No matter how the geocoding service is developed and no matter who develops the geocoding service, the semantics of geocoding web services are the same in the taxonomy described in Figure 1, although the APIs or IOPEs can be different. Furthermore, *not every concept in the taxonomy of the service semantics has a corresponding syntactic counterpart associated with it*. In fact, the concept of geocoding web service should refer to a series of web services within this semantic taxonomy. The concept of geocoding web service itself does not have a syntactic API associated with this semantic concept, while how to develop web services (APIs) is not dependent upon the semantic definition.

When it is easy to understand that inputs and outputs (IO) can be treated as part of API definitions and, thus, not related to the semantics of web services, preconditions and effects (PE) should not be related to the semantics of web services either. A precondition is examined to ensure all input requirements are satisfied before service invocation. Effect, or post-effect, is about the results after service invocation. Given the example of “unsupervised image classification”, to invoke such a service, the precondition includes four variables (*i.e.*, image source file, number of classes, maximum number of iteration and the threshold ratio to terminate the function) that must be defined. If any of the preconditions cannot be satisfied, the function or service will not be invoked. In the case of the effect of service invocation, one possible result could be the failure of the service call, due to memory

problems, for example. No matter what the precondition (P) or effect (E) could be, it has nothing to do with the concept of this service (*i.e.*, “unsupervised image classification”). The concept remains the same no matter if service invocation is successful or not. In conclusion, PE is not relevant to “the semantics of web services”, as we never change the concept of a service, because the service requester cannot satisfy the precondition to invoke a service or because the service fails for some reason.

Geocoding is a professional concept in GIS. Alternatively, the same concept can be called in similar or relevant terms by other application or service developers, such as “locate” [19], “locator” or “locating” service. All these terms are about the *concept of the service*, but are different at the *semantic level*. Thus, they can be defined as the *equivalentClass* of geocoding service through the formalization of the ontology and semantics of web services. Obviously, the semantics of web services, *i.e.*, the concepts of services and the relationship between services, have no relationship with the APIs or IOPEs.

The sample taxonomy in Figure 1 is a very small segment in the domain knowledge of GIScience, since GIS software may have more than 600 functions, which theoretically can be decomposed into services. Vector *overlay* computation services may perform the operations of *union*, *difference*, *exclusive OR (XOR)* or *intersection* on the input datasets. A variety of raster computation services may include *interpolation*, *hillshade*, *viewshed* and *reclassification*. The *networking* services may cover the functions for different kinds of *networks*, such as *transportation*, *hydrology*, *electricity*, *water* and *sewer* facilities. The concepts for geospatial *statistic* services are obviously related to two knowledge domains.

All these words in *italic* are relevant to the semantics of geospatial web services, which means they are in the conceptual framework of geospatial web services. Within this conceptual framework, the relationship between thousands of concepts of services are defined and formalized. This conceptual framework is the ontology and semantics of hundreds of geospatial functions and services that are neutral and independent from the APIs or IOPEs. No matter who develops any of such geospatial web services in any API with varied IOPEs, the semantics of geospatial web services are in the same conceptual framework. The result of service discovery based on the semantics of web services will contain the URIs of the services, which could be the identifiers of service providers or developers. In this case, the details of the semantic specification of services could be further investigated. The impact of the semantic specification and service discovery would have a broader impact on the research of the quality of services, which may have impacts on users’ behavior, as well.

3.2. Prior Works on Semantic Web Services

While web services were criticized, as they only had syntactic definitions without semantics, it has long been expected that semantic web services research could enable the automatic and dynamic service discovery, matchmaking, composition and invocation of web services. However, it has to be clarified that service invocation is based on the syntactic definition of the services, not the semantics of services. The semantic definition or specification is fundamental to the service discovery process, while it helps the matchmaking and composition processes that are based on the results of service discovery. Unfortunately, three member submissions to the W3C [20–22], including Semantic Annotation to WSDL (SAWSDL), Web Ontology Language for Web Services (OWL-S) and Web

Service Modeling Ontology (WSMO), could not help successful service discovery, because these prior works were all based on or relevant to WSDL, while it is proven that service semantics are neutral and independent of the syntactic API definitions (IOPEs) in WSDL.

3.2.1. SAWSDL and SAREST

While it was claimed [23] that “SAWSDL is the first step to infusing semantics into services and SOA”, it has been demonstrated that *the semantics of web service are neutral and independent from the service interface definition specified within the WSDL document*, because the *same* service semantics can be implemented through a *different* service interface definition specified in WSDL, while the *same* interface definition specified in WSDL can have *different* service semantics. Moreover, adding semantic annotation into WSDL allows every different service provider to annotate the service interface definition in a different way—the result may lead to *semantic chaos*, although the authors did not have the awareness that the semantics of web services have no relationship with WSDL. Given the example of geocoding services discussed above, annotating the syntactic definitions of “*findAddress*”, “*LocationInfo*”, “*findLocationByAddress*” and “*GeocodeInfo*” cannot establish the semantic equivalence between the same geocoding services developed in different versions. Similarly, the so-called SAREST tries to add semantic annotation to a web page that describes the REST services, but such an approach has little value to enable meaningful service discovery for the same reasons.

3.2.2. OWL-S and WSMO

Both OWL-S and WSMO [20,21] are full of *paradoxes and assumptions*, because they mix the concept of web service with web site. “It’s not easy to understand whether they are talking about a web service or a web site when they discuss the role of varied service providers in their submission and tutorial” [24]. Both approaches focus on service aggregation through varied logical models. Indeed, OWL-S and WSMO might not be relevant to service discovery, as both were mainly about composing services on the (semantic) web. At last, both approaches are themselves proprietary and, thus, not compatible with each other, let alone any solution of semantic interoperability.

3.2.3. Dynamic Invocation of Web Service

The dynamic invocation of web services had been claimed as one of the far-reaching goals of SWS. In contrast to the static service invocation, *dynamic service invocation* means the client application has no link to the pre-uploaded service stubs that implement those server classes defined and specified in the WSDL document. Beyond such a *programming* aspect, *dynamic service invocation* was envisioned as follows: “*without any reprogramming, a software system could have the flexibility to use various services that do the same kind of job but have different APIs*” [25]. In case the service requester can find required multiple services, the requester can invoke any of the discovered services *without the need to reprogram*.

Given the example of geocoding web services, the same service can be developed in different ways with different APIs and IOPEs. No matter how semantic concepts are annotated onto such APIs or IOPEs, dynamic or automatic service invocation is *not* possible. Given the example of ESRI’s multiple

versions of the same geocoding service, ESRI offers a detailed guide to tell the service consumers about how to *migrate* from the old version to the new version. Dynamic service invocation is not possible no matter how you markup different versions of APIs or IOPEs with semantic tags, although the same service provider offers the same service in different versions. Between different providers, such as ESRI vs. geocoder.us, dynamic service invocation is not possible, because the APIs and IOPEs are different.

In summary, the dynamic invocation of web service has no relationship with the semantics of web services. *Service invocation is determined by the syntactic specifications, not the semantic definition.* Dynamic invocation of web service is possible if the syntactic specification of the service can be standardized [26]. Only in this way, the requester can invoke any of the discovered services *without the need to reprogram* when standardized APIs are applied on services with the same semantic specification.

GIS scientists had discussed the interoperability issues in the 1990s [27]. At the technical (syntactic) level, interoperability is relatively easy to achieve. At the semantic level, it is much more difficult. The most challenging task is at the institutional level if nobody would like to compromise and agree. If syntactic and semantic specifications are about legislation, we then need law enforcement for implementation. Although the world will never become uniform, there has been a sort of uniformity in the cyber-world.

In a recent publication [28], it was addressed that “from the very beginning, the Internet/web technology had adopted a top-down approach that defined the rules and protocols first. Application development over the Internet/web has been under the strict control of varied Internet protocols. For example, we know that such image types as TIF, BMP, GIF, JPG and PNG are supported on the Internet. While many other image types, such as IMG, GRID, MrSID, BIL, BIP, BSQ, and some others, are common for satellite images, aerial photos and digital elevation models, they cannot be published on the Internet directly.” Here, the web browser is actually a cybernetic, or law enforcement, tool that implements the HTTP/HTML protocols, or otherwise, the web browser cannot decode the content in the right format. Dynamic service invocation can be realized under certain conditions when the standardized syntactic interface can be enforced in practice. In the GIScience community, a variety of OGC web service (OWS) standards have been adopted and implemented successfully for a long time, such as Web Mapping Services (WMS), Web Feature Services (WFS) and Web Coverage Services (WCS), that specified the syntactic structure to format the HTTP request and response.

4. The Formalization of Geospatial Web Services for Sustained Service Registry

If the semantics of web services are about the concepts of services and the relationship between services, the formalization of geospatial web services can be a challenging task to GIS scientists who have to first identify all concepts of the functional modules as services in GIS software and build the taxonomy to organize the concepts of geospatial computation into hierarchical order. This paper promotes such an initiative by prototyping a few examples to help understand the complexity and challenges of constructing the complete ontology and taxonomy of geospatial computation and services. If the purpose of semantic specification is to enable service discovery, a sustained service

registry has to be developed to enforce the implementation of semantic and syntactic specifications in order to realize the goals of semantic web services.

4.1. The Formalization of Geospatial Web Services

The sample taxonomy of geocoding web services described in Figure 1 is only one segment in the domain knowledge of GIScience, since a GIS may have more than 600 functions, which theoretically can be decomposed into services. Since different service providers can develop the same service in different ways, the same service has different API or IOPEs in its service description document. As it has been proven that the service semantics are neutral and independent of the syntactic API definitions (IOPEs), when different service developers publish their services into a service registry or repository, such as UDDI, service discovery is not feasible, because traditional service registries only manipulate the syntactic components in service APIs or IOPEs. Inevitably, service discovery is not possible based on those meaningless and discrete elements of APIs or IOPEs.

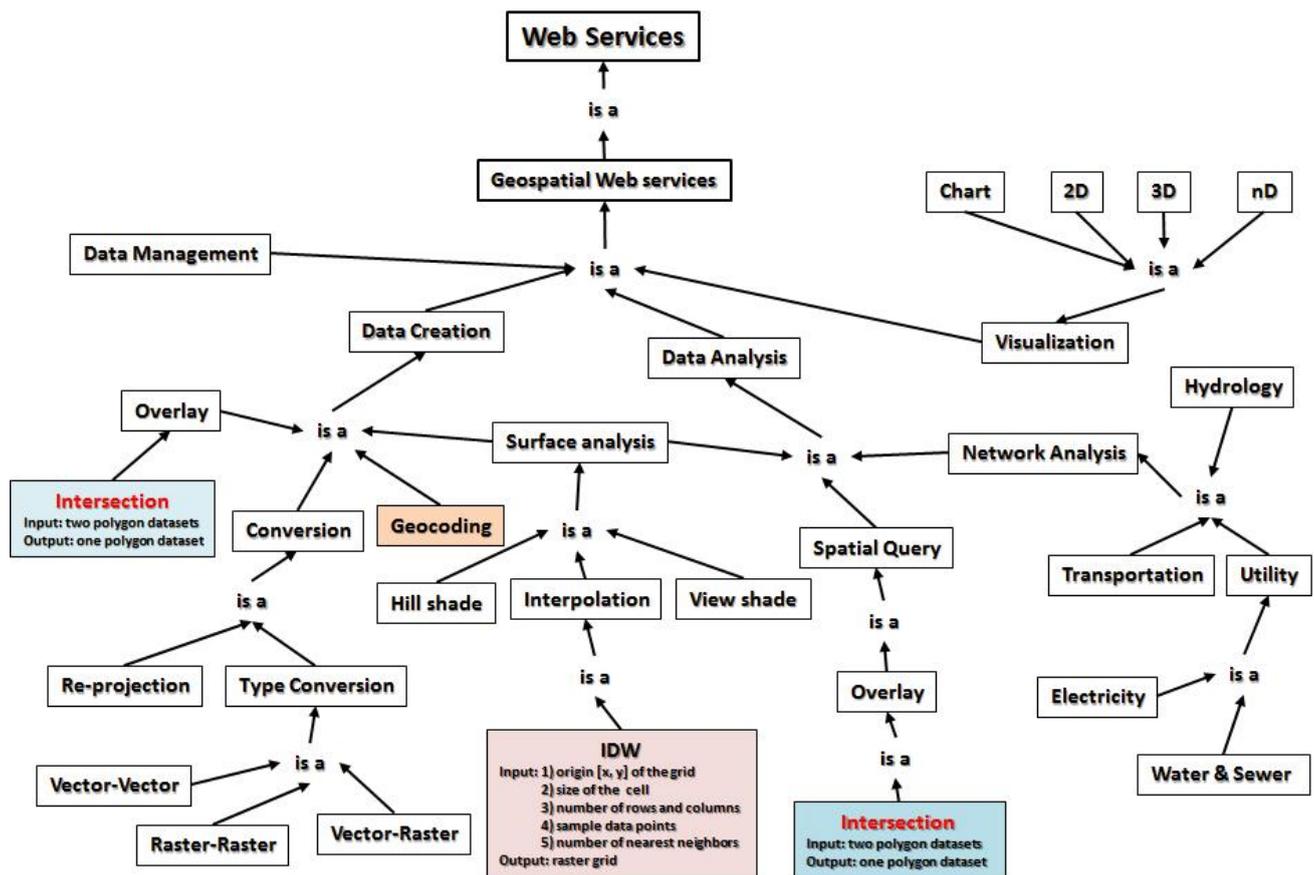
In nature, web services technology decomposes the whole software packet into individual functional modules and components. Thus, the ontology or taxonomy framework of geospatial services in the domain science application can be established based on the GIS software architecture to enable efficient and meaningful service discovery. Due to the complexity of GIS software and functionalities, this paper only discusses a top-level taxonomy of GIS software as a starting point to lead and expand further discussion and substantiation of the ontology of GIS software and geospatial semantic web services.

To enable a meaningful and successful discovery of geospatial web services, the formalization of hundreds of geospatial web services is a prerequisite. According to its definition, a GIS integrates hardware, software and data for *capturing, managing, analyzing* and *displaying* all forms of geographically referenced information [29]. For this reason, geospatial web services can be formalized into four hierarchical sub-categories as *data management services, data creation services, data analysis services* and *data visualization services*. Figure 2 describes a sample taxonomy of geospatial web services covering partial concepts of geospatial computation. The taxonomy of geocoding web services illustrated in Figure 1 is only one node under *data creation services* in Figure 2.

Data creation means creating new geospatial data from scratch or empty files, such as digitization, creating an index grid through the Fishnet function or creating a new feature class. Data creation also includes functions to create geospatial data from non-spatial data, such as the above-mentioned geocoding functions. Moreover, data creation may be a result of other geospatial functions that generate new data as the output, including, but not limited to, the concepts of data conversion, transformation, calculation, modification (add, update, delete features and attribute fields), re-projection or creating new data from existing datasets, such as building road network from a highway data layer.

Data storage and management covers the concepts regarding the manipulation of data types, spatial reference systems, data definition, spatial index, topology, relationship and other database-related terminology.

Figure 2. Sample taxonomy of geospatial web services.



Data analysis includes the concepts of spatial (overlay) analysis, temporal analysis, non-spatial attribute analysis, network analysis, spatial modeling, 3D analysis and statistical analysis. The boundary between the concept of data creation and data analysis can be identified as whether new data are created. For example, an overlay analysis will only select and highlight features from existing feature layers by implementing the overlay operator, while overlay operation for data processing will create new datasets. In many cases, geospatial computation can be classified as a subclass under both categories of data creation and data analysis. For example, terrain surface analysis implements the data analysis task, but also generates new data as the output result.

Data visualization includes the concepts of spatial visualization through 2D or 3D maps, non-spatial visualization, such as varied charts and graphics, and statistical description. Subcategories of the concepts for visualization may include the ontology of symbology, color, font and other cartographic concepts, which are used in specifying the properties of IOPEs for visualization services.

This sample taxonomy of geospatial web services, in essence, reinforces the general rules enumerated in Section 4. Given the example of *Intersection* service, the syntactic API definitions or IOPEs could be the *same* for either Boolean operation or topological operation in spatial overlay computation, because the *input* is two polygon datasets and the *output* is one polygon dataset. The semantics of these two services, however, can be different. When the Boolean operator is used, the *Intersection* service does not create any new features, but only selects features from *one* input dataset (which is called the base layer in professional GIS terminology) as the output features. When the topological operator is used, the *Intersection* service creates new polygons derived from *two* input

datasets. For this reason, the same concept of *Intersection* service with the same service APIs or IOPEs can be classified under two different categories of geospatial web services. Tables 1 and 2 describe some of the Boolean and topological operators in the polygon overlay computation. The concept of “intersect” or “intersection” could have different meaning in Boolean and topological operations. Such a difference may not be easily elaborated by the syntactic IOPE definitions, but can be clarified by the conceptual taxonomy described in Figure 2.

Table 1. Boolean operators for overlay computation (no new feature created).

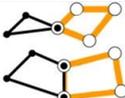
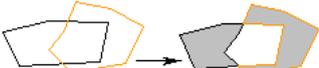
Operator	Description	Operator	Description
<i>Equal</i>		<i>Touch</i>	
<i>Disjoint</i>		<i>Contain (black contains yellow)</i>	
<u><i>Intersects</i></u>		<i>Within (black is within yellow)</i>	

Table 2. Topological or logical operators for overlay computation (new feature(s) created). *Symmetric Difference* implements the XOR (exclusive OR) operation.

Operator	Description	Operator	Description
<i>Intersect</i>		<i>Union</i>	
<i>Difference</i>		<i>Symmetric Difference (XOR)</i>	

Given another example, interpolation is a kind of spatial analytics that interpolates a raster surface by using the values of neighboring observations to estimate the value of properties at un-sampled sites. Inverse Distance Weighted (IDW), spline and kriging are three commonly used algorithms for interpolation computation. When the IDW algorithm is utilized for interpolation calculation, the syntactic API definitions or IOPEs include: the origin (x,y coordinates) of the output grid, the output cell size, the number of rows and columns, a dataset of sampled observations and how many *nearest* neighbors are used in the calculation. The syntactic definition does not reveal the semantics of how to find the *nearest* neighbors. Applying Euclidean distance vs. great circle distance to find the *nearest* neighbors will generate a different result in interpolation, particularly around the polar area and areas across the 180 longitude degree. This means the service API or IOPE description has to clarify whether Euclidean distance or great circle distance is used to find the *nearest* neighbors, so that the user can utilize the appropriate service to complete the task based on the spatial extent of the dataset. However, such a parameter or option in the API or IOPE description of how to find nearest neighbors has no impact on the name or concept of the “interpolation” services. It is called “interpolation” no matter whether Euclidean distance or great circle distance is used to do the “interpolation” calculation. For this reason, it is critical to differentiate the concepts used to define the “services” from those used in the IOPE definitions. We will not change the service name or concept, due to the different parameters

in IO or PE, and thus, service semantics are indeed neutral and independent of the syntactic API definitions (IOPEs).

In general, all those concepts in Figure 2 are relevant to the semantics of geospatial web services, which means they are in the *conceptual* framework of geospatial web services. Within this conceptual framework, the *relationship* between thousands of concepts of geospatial services are defined and formalized. This conceptual framework is the ontology and semantics of hundreds of geospatial functions and services that are neutral and independent from the APIs or IOPEs. No matter who develop any of such geospatial web services in any API with varied IOPEs, the semantics of geospatial web services are in the same conceptual framework. Without understanding the semantics or the meaning of geospatial computation, manipulating the syntactic API definitions or IOPEs of a web service is helpless for derive the semantics of geospatial web services. As a result, discovering the right services is not possible.

Although the focus of this paper is to differentiate the syntactic definition from the semantics of web services in order to establish and prove the general rules about the semantics of web services, it has to be indicated that a variety of challenges or conflicts may occur in the process of constructing a shared conceptualization about geospatial computation and services. It seems not everybody may easily agree on how to organize and formalize the domain knowledge in a hierarchical framework. For this reason, it can be expected that the conceptual framework for geospatial services and computation may have to be developed and optimized by the GIScience community if the lack of a consensus can be a serious obstacle, even if the definition of the semantics of web services can be clarified.

Early in the 1990s, interoperability problems in the GIS community were addressed at three levels, as technical, semantic and institutional [27]. Technical interoperability could be the easiest to achieve, such as WSDL, which represents an “agreement” among different parties. Semantic differentiation or problems can be identified in a given knowledge domain, such as the concept of geocoding services. The biggest problem limiting semantic interoperability is at the institutional level, because the unwillingness of institutions to adhere to common standards or the debate among researchers could prevent consensus on the formalization of the domain knowledge of geospatial computation.

4.2. Sustained Registry for Semantic Service Discovery and Dynamic Service Invocation

Meaningful service discovery is based on the domain knowledge that is represented by a conceptual framework containing the semantic definition of the services and relationships between the concepts of services, which has little relationship with the IOPEs or the syntactic components of a service. Since the semantics of web services have little relationship with the syntactic definition, service registration will have no dependency on the syntactic specification of web services, such as WSDL. Prior works on service publication and discovery failed, such as UDDI, because the designers had no awareness about the distinction and relationship between the service semantics and the IOPEs definitions.

The key for a sustained service registry and successful service discovery is to build the knowledge base to control and regulate the operation of the registry system, following the scientific principles in systems science, cybernetics and cognitive psychology [28]. If the registry system is envisioned as a warehouse of services, such a warehouse should be categorized first before the input items are placed into it, so that we can find the necessary components efficiently. UDDI and prior works failed, because

they did not have any feasible management plan to organize the resources, let alone to find any meaningful services. On the contrary, such an open system allowed service providers to publish services into the system freely and randomly, which resulted in chaos and failure inevitably, as interpreted by systems science and information theory [28].

The so-called SAWSDL approach asked the service developers to markup their service IOPEs before they dump the service into the registry warehouse. Unfortunately, as it has been evidenced that (1) the same service provider can use different service interfaces defined in WSDL documents even when developing the same kind of service, (2) different service provider can use the same service interfaces defined in WSDL documents, even when developing different kinds of services and (3) different service providers can generate different service interfaces defined in WSDL documents when developing the same kind of service, the SAWSDL approach may only aggravate the chaos, because the IOPEs have no direct relationship with the semantics of the services.

Since service semantics have no relationship with the syntactic definition of services, service discovery based on the semantic definition and knowledge framework will be different from the prior works. That is to say, a service provider will register in the registry to offer a kind of service based on the semantic definition of this service. Consequently, service discovery will find which service providers will offer a given type of service. In this case, different providers may develop the same service in different ways, such as using either the SOAP or REST approach, or the IOPEs of the same service can be different. The outcome of service discovery is a list of service providers that provide the corresponding services along with the URLs as the location of the service. How to invoke such services will be dependent upon the syntactic definition of the IOPEs, which is not relevant to the semantics of the services and service discovery. Given the example of Geocoding services, service discovery should be able to find such services offered by ESRI, geocoder.us, Yahoo! and Google, no matter whether such services are based on SOAP or REST approaches or whether the services are named as findAddress, findLocationByAddress, GeoCoder, locator or something else in the syntactic definition.

However, if we remember that the dynamic service invocation is the last and final goal of semantic web services discussed in Section 5, we also need to standardize the IOPEs or syntactic definition for web services. In this way, without any reprogramming, a software system could have the flexibility to use various services that do the same kind of job and have the same APIs, no matter whether it employs the SOAP-based or REST approaches. The ultimate goal of a sustained service registry will thus be operated upon two sets of specifications. The semantic specification will be the guideline for service registry and discovery. The syntactic specification will ensure that the dynamic invocation of services will be applicable.

5. The Expansion of the Established Scientific Principles to Other Domain Knowledge

The three general rules discussed in Section 4 have established the scientific principles in constructing the ontology or conceptual framework to enable the development of semantic web services, not only for GIScience, but also for any domain knowledge. Given the example in the scientific domain of *statistics*, computational modules in well-known software products, like R, MATLAB, SAS and SPSS, can be exposed as web services. Although the syntactic API definitions or

IOPEs of a statistical web service can be different, the semantics of the service should be identical. For example, a service that calculates the standard deviation of a dataset or implements the regression analysis for modeling and analyzing several variables can be developed by different service providers with different APIs, but each type of service (*standard deviation calculation* or *regression analysis*) has the same service semantics. In the domain science of *bioinformatics*, the syntactic API definition, or IOPEs, for WebServiceBlast [30] is meaningless. Without constructing a taxonomy in the domain knowledge of bioinformatics, nobody knows the semantics of bioinformatics services by those components defined in APIs or IOPEs.

Within domain applications initiated by government agencies, such as web services developed by US EPA [31], NOAA [32], the US Department of Agriculture [33], NASA [34] or by industry, such as Microsoft's TerraService [35], and those from ESRI [16,17], all of the syntactic descriptions of such services may be not meaningful, or semantic, in the context of the service-oriented architecture for service discovery. In fact, such organizations developed specific web sites to interpret the services in detail to help users or consumers to understand the meaning of services. Such endeavors might have proven that service discovery through the registry or index approaches could not help users to find and understand the right services based on the syntactic definition and description of such services, because their syntactic API definitions or IOPEs had no relationship to the service semantics and were not meaningful to enable service discovery. For example, NOAA's National Digital Forecast Database (NDFD) web services [32] have nine functions, including NDFDgen(), NDFDgenLatLonList(), LatLonListSubgrid(), LatLonListLine(), LatLonListZipCode(), LatLonListSquare(), CornerPoints(), NDFDgenByDay() and NDFDgenByDayLatLonList(). The concepts of such services are not clear, and the relationships between these services are not defined. Actually, NDFD services, as well as NASS/USDA's CropScape and VegScape web services [33] are developed for the purpose of data dissemination; thus, they could be placed and organized under the category of data services.

Since individual applications or services only target certain specific use cases, the concepts of standalone services could hardly be organized to establish a taxonomy with a hierarchical framework to clarify the relationships between the services. For this reason, developing a well-organized domain knowledge base is fundamental to formalize the domain-specific computational modules to enable a successful service discovery. For example, Microsoft's TerraService provides sixteen functions, including ConvertLonLatPtToNearestPlace(), ConvertLonLatPtToUtmPt(), ConvertPlaceToLonLatPt(), ConvertUtmPtToLonLatPt(), CountPlacesInRect(), GetAreaFromPt(), GetAreaFromRect(), GetAreaFromTileId(), GetPlaceList(), GetLatLonMetrics(), GetPlaceFacts(), GetPlaceListInRect(), GetTheme(), GetTile(), GetTileMetaFromLonLatPt() and GetTileMetaFromTileId(). Obviously, some of these functions have a close relationship to the domain knowledge of GIScience, such as those data conversion functions that can be placed under the categories of re-projection (e.g., between Universal Transverse Mercator coordinate system and Lat/Long) and type conversion (between non-spatial and spatial data, such as ConvertLonLatPtToNearestPlace() and ConvertPlaceToLonLatPt()) under the subclass of conversion in Figure 2. Those GET functions could be an effort for data dissemination, as well.

In the case of NASA's Global Change Master Directory (GCMD) [34], concepts relevant to Earth science data and services have been categorized by hierarchical key words to build and organize the knowledge base to refine the search query over GCMD resources. However, the concept of "service" may not be clear when data, web site and web applications, SOAP-based and REST services are all

mixed together. For example, “The Harvard Geospatial Library” or the “European Forest Fire Information System (EFFIS)” can be found under the category of “Services/Tools”. However, as defined in this paper, a service is a software module that can be accessed by SOAP-based or REST protocols. A mixture of everything seems not to be helpful for finding the right services.

6. Conclusions

In 2007, Michael Brodie wrote, “How will services discover the right services that meet specific requirements in such an environment? We’re far from that scale at the moment.” [4]. If we are still far from discovering the right services, it is not surprising, because there was not even a correct understanding of the semantics of web services, while most of the existing services were not organized in any domain science. Service discovery should be based on the semantics of web services, which have no relationships with APIs or IOPEs. As a result, manipulating APIs or IOPEs is useless to derive the semantics of web services and to find any right service.

Service computing in computer science and the IT mainstream might only deal with some simple business transaction models without essential domain knowledge. On the other hand, the needs of ontology and semantic research are based on the requisites that there are too many ambiguous concepts that have to be formalized into a taxonomical or conceptual framework to study the relationship between the concepts.

While the business transaction models have been focusing on buying something, e.g., book, ticket, hotel, *etc.* the deficiency of concepts in business transaction models has prevented such research from exemplifying the value of semantics in service computing. When we switch the focus to domain science applications from business transactions, a substantial difference can be identified, as discussed in this paper. If semantic web services can advance, it will happen in domain science applications, not from IT applications based on business transactions.

The established three principles about the semantics of web services will transform the web service architecture, or SOA, and web service applications, in general, into a new paradigm. Service registry has to be re-constructed based on the ontology and semantic framework of a domain science. Such an ontology covers the concepts of services and the relationships between the services. Service publication has nothing to do with the syntactic components of APIs or IOPEs, but will be based on the semantic classification of the service. That is to say, service providers will first find a semantically defined service in the registry and, then, register their services by its unique URL. Service discovery can be successful based on the ontology framework. When a service can be discovered, the registry will identify a series of service providers that offer such a service. Since services under the same semantic definition can be developed by different providers with different syntactic components defined in APIs or IOPEs, dynamic service invocation can be realized only when the syntactic specifications can be standardized.

Besides its importance in service-oriented computing, research on the semantics of computational modules in constructing GIS is also significant in GIS software engineering and education. Without such research and established theory, many GIS software products are only a pile of functional modules and tools, as the software architecture and structure cannot present the domain knowledge of GIS. Inevitably, duplicated modules and ambiguous concepts can be identified in GIS software

products. This creates a problem for GIS education, as students may be confused by those duplicated modules positioned in different places in the software, as such modules perform exactly the same functions, while in other situations, similar or the same kind of functional modules may use different names. Essentially, web services are computational modules of software products. For this reason, the conceptual framework constructed for semantic geospatial services can be a common infrastructure to GIS software engineering in order to develop a more meaningful system to improve the learning process. Since ontology is the formalization of the shared conceptualization of a domain knowledge, it is expected that the conceptual framework for geospatial computation can eventually be developed and optimized by the GIScience community. In general, the formalization of the semantics of web services could include an “is-a” relation, taxonomies of classes and properties, class-instance relationships, and so on. This paper only discussed two types of relationships in formalizing the concepts of geospatial web services. However, this does not mean there are only two kinds of relationships, since many other types of relationships (e.g., intersection, union, complement, disjoint, and so on) between services can be considered in this context.

Acknowledgments

This paper was supported partially by the National Science Foundation through the award OCI-1047916.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Service. Available online: <http://www.thefreedictionary.com/Service> (accessed on 13 September 2013).
2. Service. Available online: <http://www.merriam-webster.com/dictionary/service> (accessed on 13 September 2013)
3. Service. Available online: http://www.ldoceonline.com/dictionary/service_1 (accessed on 13 September 2013).
4. Brodie, M. Semantic technologies: Realizing the services vision. *IEEE Intell. Syst.* **2007**, *22*, 13–15.
5. Foster, I. Service-oriented science. *Science* **2005**, *308*, 814–817.
6. W3C. Web Services Architecture. 2004. Available online: <http://www.w3.org/TR/ws-arch/> (accessed on 13 September 2013).
7. Al-Masri, E.; Mahmoud, Q.H. Discovering web services in search engine. *IEEE Internet Comput.* **2008**, *12*, 74–77.
8. Universal Description Discovery and Integration. Available online: http://en.wikipedia.org/wiki/Universal_Description_Discovery_and_Integration (accessed on 13 September 2013).
9. Fielding, R.T. Architectural Styles and the Design of Network-Based Software Architectures. Ph.D. Thesis, University of California, Irvine, CA, USA, 2000.
10. Singh, M.P.; Huhns, M.N. Chapter 1 Computing with Services. In *Service Oriented Computing—Semantics, Processes, Agents*; John Wiley & Sons, Ltd.: West Sussex, UK, 2005; pp. 11–12.

11. Tu, S.; Abdelguerfi, M. Web services for geographic information systems. *IEEE Internet Comput.* **2006**, *10*, 13–15.
12. Petrie, C. Practical web services. *IEEE Int. Comput.* **2009**, *13*, pp. 93–96.
13. W3C. Web Services Glossary. 2004. Available online: <http://www.w3.org/TR/ws-gloss/> (accessed on 13 September 2013).
14. Lara, R.; Roman, D.; Polleres, A.; Fensel, D. A conceptual comparison of WSMO and OWL-S. *Lect. Note. Comput. Sci.* **2004**, *3250*, 254–269.
15. Cabral, L.; Domingue, J.; Motta, E.; Payne, T.; Hakimpour, F. Approaches to semantic web services: An overview and comparisons. *Lect. Note. Comput. Sci.* **2004**, *3053*, 225–239.
16. ESRI. ArcWebServices Version 2. Available online: <http://arcweb.esri.com/services/v2/AddressFinder.wsdl> (accessed on 13 September 2013).
17. ESRI. ArcWebServices Version 2006. Available online: http://www.arcwebservices.com/services/v2006_1/AddressFinder?wsdl (accessed on 13 September 2013).
18. Geocoder.us. Available online: <http://geocoder.us/dist/eg/clients/GeoCoder.wsdl> (accessed on 13 September 2013).
19. BlackBerry. Available online: <http://us.blackberry.com/developers/platform/locateservice/> (accessed on 13 September 2013).
20. W3C. OWL-S: Semantic Markup for Web Services. 2004. Available online: <http://www.w3.org/Submission/OWL-S/> (accessed on 13 September 2013).
21. W3C. Web Service Modeling Ontology (WSMO). 2005. Available online: <http://www.w3.org/Submission/WSMO/> (accessed on 13 September 2013).
22. W3C. Semantic Annotations for WSDL. 2006. Available online: <http://www.w3.org/2002/ws/sawsdl/spec/SAWSDL.html> (accessed on 13 September 2013).
23. Verma, V.; Sheth, A. Semantically annotating a web service. *IEEE Internet Comput.* **2007**, *11*, 83–85.
24. Shi, X. Semantic web services: An unfulfilled promise. *IT Prof.* **2007**, *9*, 42–45.
25. Burstein, M.H. Dynamic invocation of semantic web services that use unfamiliar ontologies. *IEEE Intell. Syst.* **2004**, *19*, 67–73.
26. Shi, X. Semantic Request and Response for Standardized Web Services. Available online: <http://www.ibm.com/developerworks/library/ws-semantic/> (accessed on 13 September 2013).
27. Goodchild, M.F.; Egenhofer, M.J.; Fegeas, R. Interoperating GISs. Available online: <http://www.ncgia.ucsb.edu/conf/interop97/report.html> (accessed on 16 September 2013).
28. Shi, X.; Nellis, M.D. Semantic web and service computation in GIScience applications: A perspective and prospective. *Geocarto Int.* **2013**, in press.
29. ESRI. *Understanding GIS: The ARC/INFO Method*; Environmental System Research Institute: Redlands, CA, USA, 1990.
30. Altunay, M.; Colonnese, D.; Warade, C. Web Services for Bioinformatics, Part 1. Available online: <http://www.ibm.com/developerworks/webservices/library/ws-bioinfo/index.html> (accessed on 13 September 2013).
31. EPA STORET/WQX Web Services. Available online: http://www.epa.gov/storet/web_services.html (accessed on 13 September 2013).

32. NOAA National Digital Forecast Database (NDFD) Simple Object Access Protocol (SOAP) Web Service. Available online: <http://graphical.weather.gov/xml/> (accessed on 13 September 2013).
33. USDA National Agricultural Statistics Service. Available online: <http://www.nass.usda.gov/> (accessed on 13 September 2013).
34. NASA Global Change Master Directory. Available online: <http://gcmd.nasa.gov/> (accessed on 13 September 2013).
35. Microsoft's TerraService. Available online: <http://teraserver-usa.com/TerraService2.asmx> (accessed on 13 September 2013).

© 2013 by the authors; licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution license (<http://creativecommons.org/licenses/by/3.0/>).