Chaining Geographic Information Web Services

Client-coordinated, static, or mediated chaining of GIS Web services enables easier on-demand access to customized geographic information.

Nadine Alameh
Global Science & Technology

Geographic information systems (GISs) are "computer-based information systems that enable capture, modeling, manipulation, retrieval, analysis, and presentation of geographically referenced data."[1] These systems process queries about spatial data, such as what is at a particular location, which locations satisfy certain requirements, what the spatial relationship is between certain objects, and what spatial patterns a certain data set supports. To address such questions, the GIS research field combines various special-interest computer-science topics, such as databases, graphics, systems engineering, and computational geometry.

Over the past decade, GIS technologies have evolved from the traditional model of stand-alone systems, in which spatial data is tightly coupled with the systems used to create them, to an increasingly distributed model based on independently provided, specialized, interoperable GIS Web services.[2] This evolution is fueled by factors such as GIS's growing role in today's organizations, spatial data's increasing availability and inherent conduciveness to reuse, the maturity of Web and distributed computing technologies, and GIS's key role in the promising location-based services market. Furthermore, most users of traditional GIS systems use only a small percentage of their systems' functionalities; the services model provides users with just the services and data they need, without having to install, learn, or pay for any unused functionalities.

The services-based GIS model is rapidly materializing, owing in part to advancements in general Web service technologies, and in part to focused efforts by the Open GIS Consortium (OGC; www.opengis.org) to sponsor consensus-based development of interoperable GIS Web service interfaces. Soon, it will be possible to dynamically assemble

Published by the IEEE Computer Society 1089-7801/03/$17.00

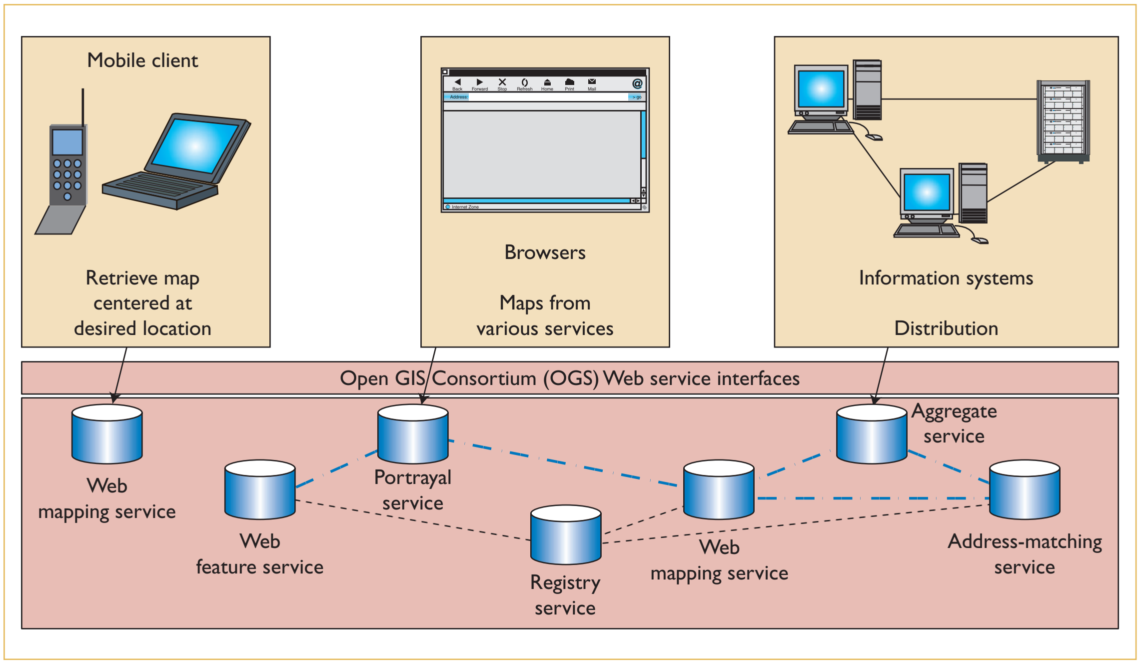

Figure 1. Simplified view of the GIS Web services architecture. Clients and services can access other services either directly or through referral by a registry service (dotted lines). Services will respond either by locally performing all the tasks required by the client (the Web mapping service the mobile client uses) or by chaining to complementary services (blue dashed lines).

applications from multiple GIS Web services for use in a variety of client applications. The mechanism for such assembly of services is often referred to as *service chaining*, the process of combining or pipelining results from several complementary services to create customized applications. GIS services have specific middleware requirements that current Web service technologies can only partially meet.

GIS Web Services Architecture

Figure 1 presents a simplified view of the GIS Web services architecture, in which a range of client applications are chaining a variety of standards-based GIS Web services.

GIS Web Services

Web services are self-contained, self-describing, modular applications that clients can publish, locate, and dynamically invoke across the Web.[3] They provide access to sets of operations through one or more standardized interfaces. GIS services can be grouped into three categories:

- *Data services* typically are tightly coupled with specific data sets and offer access to customized portions of that data. Examples of data services include the Web Mapping Service (WMS), which produces maps as two-dimensional visual portrayals of geospatial data; the Web Coverage Service (WCS), which provides access to unrendered geospatial information as needed for client-side rendering; and the Web Feature Service (WFS), which lets a client retrieve geospatial data encoded in Geography Markup Language (GML).[4]
- *Processing services* provide operations for processing or transforming data in a manner determined by user-specified parameters.[2] Such services can provide generic processing functions such as projection and coordinate conversion, rasterization and vectorization, map overlay, imagery manipulation, or feature detection and imagery classification. They are not associated with specific data sets.
- *Registry*, or *catalog services* allow users and applications to classify, register, describe, search, maintain, and access information about Web services (see www.opengis.org/ogcSpecs.htm). They store information on data types, online data instances, service types, and online service instances.

Because interoperability is critical to the GIS Web services architecture, the GIS community, with OGC sponsorship, is focusing on defining interoperable interfaces to the basic services listed above. Current OGC initiatives are studying how to offer

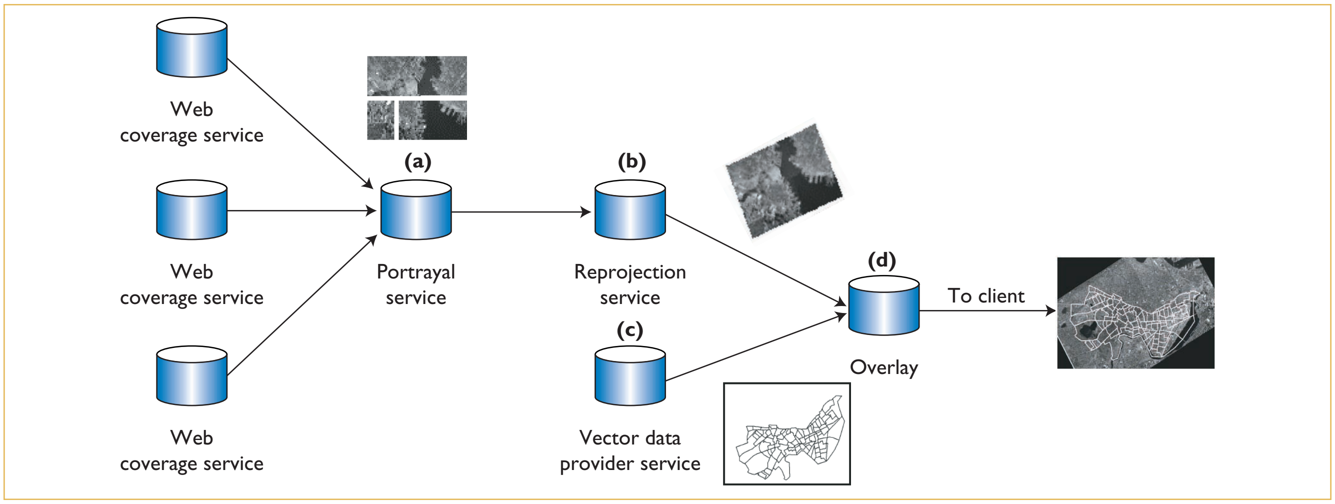

Figure 2. A simple service-chaining example of a map of Cambridge, Mass. (a) A portrayal service assembles and portrays an orthoimage from several imagery services. (b) A reprojection service reprojects the image from one coordinate system to another. (c) A vector data provider service provides access to the vector data layer at the level of detail the client requests. (d) An overlay service overlays the input image and the vector data and sends the overlay to the client.

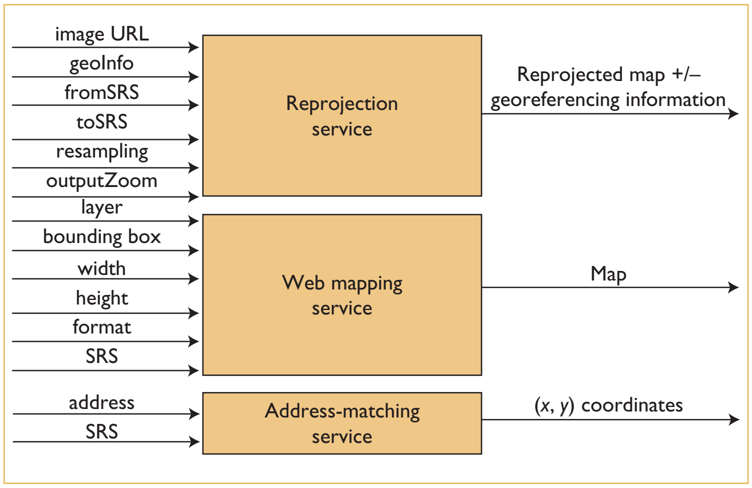

Figure 3. Key parameters of the address-matching, mapping, and reprojection services. The example services transform input variables into output products the client can use.

access to these services over `HTTP GET`, `HTTP POST`, and SOAP protocols.

Client Applications

Once GIS Web services are deployed, service providers can build client applications more flexibly by mixing and matching available services. The clients in Figure 1 cover a range of possibilities: thin clients such as Web browsers, emerging clients such as handheld devices, and traditional GIS and information systems.

A typical GIS application will require the combining or chaining of multiple GIS and non-GIS Web services. Figure 2 shows a typical service-chaining scenario involving GIS coverages, which are multidimensional metaphors for phenomena found on or near a portion of the Earth's surface.[5] GIS coverages include the special case of Earth images. A coverage portrayal service (CPS) fetches several GIS coverages from WCS services, then mosaics them and portrays the resulting composite image of Cambridge, Massachusetts. A processing service reprojects the resultant coverage to another spatial reference system (SRS). An overlay service then supplements the coverage with feature data it extracts from a WFS and sends the result to the client as a rendered map. The OGC Web site (www.opengis.org) offers more information on WCS, CPS, and WFS.

We can achieve the Figure 2 scenario using client-coordinated, static, or mediated service chaining. Each option weights coordination complexity, metadata tracking ability, and error-handling propagation differently.

Service Chaining

To compare the three main varieties of service chaining, let us employ each to execute a simple but nontrivial task: answering a user's request for a geo-referenced map of certain pixel dimensions, centered at a given location. By limiting the GIS data types to raster imagery in the example, we avoid the complexities of heterogeneous semantics and topology representations. Nevertheless, the example is still rich enough to expose the tradeoffs in various service-chaining approaches.

The example assumes the availability of the following service types:

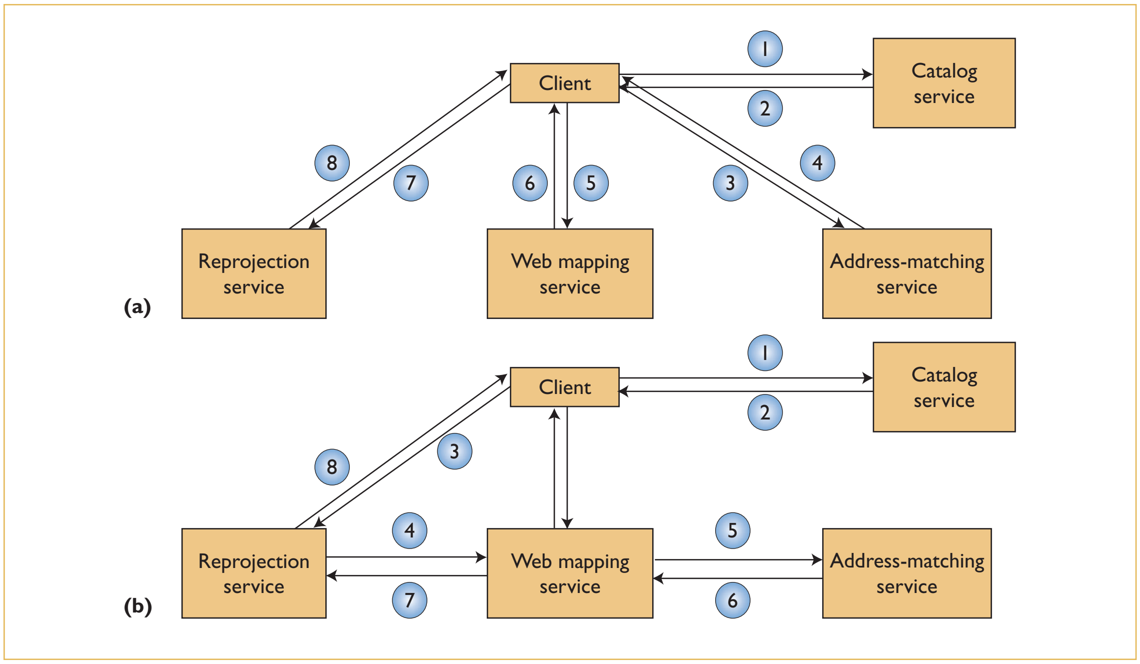

Figure 4. Client-coordinated service chaining. (a) The client can contact all the necessary services individually. (b) The client can also nest calls within the service chain, making the chaining more opaque.

- An *address-matching service* transforms a phrase or term that uniquely identifies a feature, such as a place or address, into applicable geometry (usually in the form of either a coordinate x,y or a minimum bounding rectangle).[6] For simplicity, we assume that the service can provide (x,y) coordinates in any SRS. Address-matching services often return additional information such as normalized address and location-matching precision, but we assume that it is possible to filter out additional information so that the client receives only the coordinates in response to a request. In some cases, the user might need to pick a desired location out of a list of locations matching a given address. Etak (www.etak.com) can be used as the basis for one such service.
- A *Web mapping service* returns a map corresponding to prespecified rectangular geographic extent and pixel dimensions for a given area.
- A *reprojection service* transforms a raster image from one SRS to another. This service becomes necessary when the native projection of a data set differs from the one the client requested.

Figure 3 illustrates the inputs and outputs for these three types of services. Note that the client in our example is not limited to a particular service instance from each type. Instead, the client will use registry services to find instances of the service types it needs.

The following discussion assumes that the services authenticate the client by using available authentication technologies and manage billing for services and data using existing e-commerce technologies.

Key Analysis Issues

Several issues surround the design and implementation of efficient and scalable service chaining. Chaining GIS Web services creates its own subset of these issues, such as

- *Transparency.* How much should the client see of the service-chaining complexities? A related issue is how much the client should have to do to construct, execute, and manage service chains.
- *Tracking.* How should the service chain track and relay to the client the sources of geographic data and the transformations various services apply to it along the chain? Keeping track of metadata is particularly important in GIS because users often cannot trust the data unless they have some information about its resolution, orthorectification parameters (to guide the process of correcting the relief and tilt distortions in aerial photographs), remote-sensing origin, and so forth.
- *Error-reporting.* How should services handle errors and report them along a chain to the

> client? Precise error reporting is critical in the type of synchronous chains described here, when one or more services breaks or returns an exception.

The service-chaining example provides ample opportunity to explore these issues.

Client-Coordinated Service Chaining

In the simplest case, service chaining is fully transparent to the client, who defines and controls the order of execution for individual services in a chain. Of course, the client must have prior knowledge of the different service types to be used in the chain. In Figure 4a, the client first searches a catalog service to find service instances to use.[7] The catalog service returns a pointer to an instance for each service type needed. The client then uses the address-matching service to find the coordinates of the address the user provides. The client forwards the coordinates to a Web mapping service, which returns a map. If the returned map's spatial reference system differs from the one the client used, a reprojection service reprojects the map; the client might access the catalog service again to find a reprojection service instance. In this process, the client handles all intermediate results that the services in the chain return.

Although nesting calls simplifies some of the client's coordination responsibilities, it also introduces complexities.

Figure 4b shows how the client can directly embed a request to one service in the input to the prior service. Instead of downloading the image itself, the client provides the reprojection service with the URL of the Web mapping service it needs to retrieve the map. The client specifies the bounding box the reprojection service will send to the Web mapping service as a function of the result of a request sent to the address-matching service. This approach of nesting calls within a service chain is consistent with the OGC's initial CPS and Styled Layer-Descriptor (SLD)[8] chains.

Although nesting calls simplifies some of the client's coordination responsibilities, it introduces complexities in relaying back to the client information about errors and metadata propagation, as well as the client's ability to control certain details. Consider, for example, what would happen if the address-matching service issued an exception in response to an erroneous call from the client. This exception would in turn trigger the Web mapping service to signal an "invalid input" exception. A domino effect would ensue as the same problem occurred over the Web mapping/reprojection link.

To prevent this requires a mechanism for embedding detailed information about the source or cause of every exception delivered to a client. One alternative is to let a service automatically forward the exception input to the next service in the chain, appending any of its own error messages to the forwarded exception. In this context, representing exceptions in XML is particularly useful because it makes it easier for services to detect and add to incoming exceptions.

We can extend this approach to handling metadata propagation. Services can thus append metadata, such as billing information from individual services, to normal data as it passes between services.

Finally, consider the issue of an unexpected delay in one of the chain's services. The chain's serial nature implies that the delay propagates all the way back to the client. In a scenario where the client directly accesses each service, it can abort the operation and find a substitute if a specified time-out period expires for a service.

In the nested-calls model, however, the client must control a global time-out and communicate it to every service. If a service takes longer than this global time-out at any point, the service aborts and returns an appropriate exception to the preceding service in the chain.

An alternative that bypasses the delays inherent in serial chains is to apply a message-based approach instead of (or in addition to) the request–response approach we have described thus far. The messaging approach enables services to exchange well-structured XML messages and supports a mechanism for notifying clients (or other services) when a service completes an operation.

Client-coordinated service chaining in a decentralized setting requires the client's deep involvement. Static chaining by aggregate services is an alternative that hides such chaining complexities from the client.

Static Chaining Using Aggregate Services

Aggregate services, which third-party providers usually supply, bundle static (predefined) chains of services and present them to the client as one. As Figure 5 shows, chaining becomes totally opaque to the client, which sees only a single aggregate service that coordinates the individual services in the chain.

Despite their benefits, aggregate services have some drawbacks. By having a single access point to the chain, the client loses some flexibility and control over parameters of the individual services. In the example in Figure 5, the client has no control over the reprojection step. In fact, the client might not even be aware that the image is being reprojected. Not being able to control the projection parameters can present problems for GIS users, as their applications often depend on specific data quality and resolution. To allow clients to differentiate between basic and aggregate services, each OGC service's capabilities response includes a flag that indicates whether the service is basic or aggregate (also known as cascading).

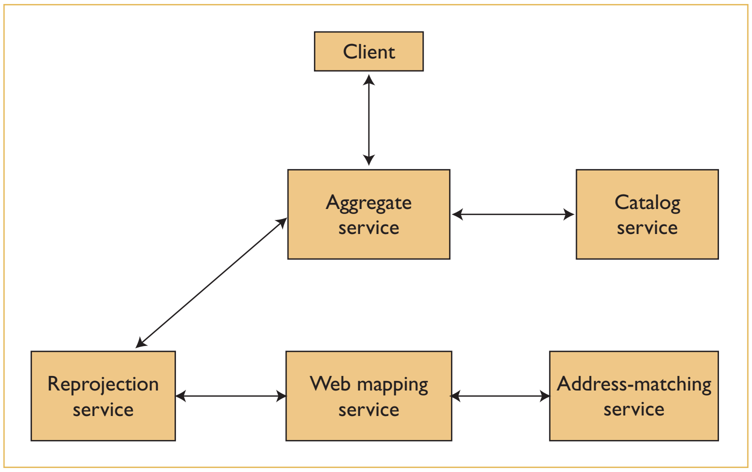

Figure 5. Static chaining with aggregate services. The aggregate service coordinates individual services so the client doesn't have to.

Workflow-Managed Service Chaining

Workflow-managed service chaining strikes a balance between opaque aggregate-service chaining and transparent client-coordinated chaining. It replaces aggregate services with smarter mediating services that act as gateways. These services offer access to data and processes, but they do not necessarily serve that data themselves. They retrieve it from other services. The client might execute the chain itself or just select a chain for the mediating service to execute.

The concept of mediating services is borrowed from the database arena,[9] where mediating elements — also known as facilitators, brokers, and dispatchers[10] — dynamically convert multidatabase queries into smaller subqueries that they dispatch to the various databases. The mediating element then integrates the subquery results and returns them to the client.

Correspondingly, in a distributed geoprocessing infrastructure, mediating services dynamically construct and manage chains of GIS Web services. Based on a client's requirements, mediating services determine appropriate data sources and services, retrieve and process the data, and assemble the final response. In the process, a mediating service might consult with available catalogs, search engines, or metasearch tools. It might also keep its own indexed lists of useful services, which are likely to be biased toward certain providers or domains. For efficiency purposes, mediating services might also provide commonly used functions, such as format, coordinate, or vector-to-raster conversions.

Mediating services use prespecified client preferences to search for data and processing services that best meet the requirements. Such preferences might include information about service time-outs, price ceilings, accuracy and mapping-quality requirements, or preferences for particular service providers. To achieve performance or monetary efficiencies, the client might also impose a constraint that all services used in a particular session come from the same provider.

With the wide range of possible GIS applications and the different semantics needed in different fields, the mediating service's internal rules should cater to specific application domains (such as transportation) or processes (such as image processing). The need for domain or process-tuned services presents excellent market-entry opportunities for third-party players with significant expertise in a domain but no ability to singlehandedly offer and maintain all the data and transformations that the domain demands.

Despite their promise for simplifying the service-chaining process, mediating services are inherently complex and challenging to design, especially in light of the continuous evolution of underlying technologies such as XML-based service description and orchestration languages.

To ensure that they properly construct and execute the chains according to client-specified requirements, mediating services need to use carefully designed mechanisms to obtain and refresh client configuration parameters. This raises the question of how to specify such parameters and how frequently to refresh them. Options include having each client register its preferences with a mediating service a priori or having each client store a local version of its preferences that the mediating service can easily access.

Moreover, the fact that mediating services create chains dynamically can result in inconsistencies in the responses the client receives. Dynamic service-chain formation means that a given request might not always garner the same response. For instance, using different reprojection services in two different executions of the same request could result in maps with different resolution and quality attributes. Whether this is acceptable depends on the nature of the client and its application. One way to overcome such a problem involves storing a cookie at the client with the details of a previously executed service chain. The mediating service can use this cookie and its enclosed service chain to fulfill future requests.

Mediating services promise to reduce the transparency of service chaining to the client because they handle most of the work required to assemble, manage, and execute the service chains. However, because they require a certain degree of intelligence and continuous updates, they could be challenging to design and implement. Existing XML-based process-definition and service-composition technologies such as the Web Services Flow Language[11] and XLANG[12] might help.

> **Many XML-based technologies can facilitate or support GIS Web services and pave the way for their automated chaining.**

Enabling XML Technologies

Many XML-based technologies can facilitate or support GIS Web services and pave the way for their automated chaining. This section highlights some of these technologies with an emphasis on how they can be used and their limitations, with respect to the geospatial community's requirements.

Web Services Description Language

WSDL[13] describes Web services as a set of endpoints operating on messages that contain either document or procedure-oriented information. WSDL provides a way to describe the messages and operations in an abstract manner and bind them to a concrete protocol and message format.

Describing a Web service with WSDL enables programs known as proxy generators to automatically construct requests to that service. By not requiring the calling party (whether a client or mediating service) to know a priori the interface to a service, WSDL makes both transparent and workflow-managed chaining easier to implement.

However, describing the service interfaces is often not enough. A data-centric field such as GIS needs a mechanism for describing the data characteristics that various GIS services can serve or process. The OGC currently achieves this by requiring each GIS Web service to support a `getCapabilities` operation that returns, among other information, details about the data or data types that the service supports.

Universal Description, Discovery, and Integration

The UDDI specification (www.uddi.org) lets businesses quickly and dynamically find and transact with each other. It can serve as an alternative technology for implementing GIS Web catalog services.

The major obstacle to the geospatial community's adoption is that UDDI registries do not currently support any type of spatial queries, which are at the heart of any GIS application. Not being able to search for services or data by bounding box constitutes a real limitation for users. It remains to be seen whether future versions of UDDI will support such functionality.

SOAP

SOAP[14] provides a simple, lightweight mechanism for exchanging structured and typed information between peers in a decentralized distributed environment.

Of particular relevance to service chaining are SOAP's model for exchanging messages among intermediaries and its `actor` attribute. This attribute in the SOAP header provides a mechanism for applications to specify a message's ultimate destination as well as any intermediary services that can partially process the message and forward it to another actor. Such a mechanism could assist mediating services in creating dynamic service chains.

DAML-Based Web Service Ontology

DAML-S (www.daml.org/services/) is a DAML-based Web service ontology that supplies Web service providers with a core set of markup language constructs for describing their service's properties and capabilities in unambiguous, computerinterpretable form.

DAML-S's support for automatic selection, composition, and interoperation of Web services is particularly relevant to service chaining. DAML-S provides a way to declaratively specify the prerequisites and consequences of individual service use, which are necessary for automatic service composition and interoperation. Mediating services can use these specifications to dynamically identify which services can be chained to each other and which can be substituted for one another to answer specific requests.

Conclusions

The Web services model lets users freely combine services to create customized solutions with minimal programming, integration, and maintenance efforts. Such a model, supported by efficient service chaining, is key to extending GIS beyond its traditional boundaries of mapping to embrace a broader community of users and a wider scope of services.[15] The development of this model should leverage general Web services technologies and extend them to address any requirements unique to the geospatial community.

Under the sponsorship of the OGC, the Web services model is rapidly manifesting within the geospatial community. Ongoing consensus-based efforts are focusing on the development of a unifying OGC service framework that can support the chaining options described so far. The framework is essential to the harmonization of the various OGC services and to the sustainability of this model. Correctly positioning this framework with respect to the evolving general Web services and IT architectures is a challenge; the Web services domain is relatively young and the underlying technologies are still immature. OGC members are currently undertaking experiments with these technologies, with a focus on highlighting the geospatial-specific requirements and producing practical recommendations to the appropriate standards groups. Other issues, such as semantics, binary compression of XML, and dynamic service chaining, have yet to be resolved.

References

1. M. Worboys, *GIS: A Computing Perspective*, Taylor & Francis, 1995.
2. N. Alameh, *Scalable and Extensible Infrastructures for Distributing Interoperable Geographic Information Services on the Internet*, doctoral dissertation, MIT Libraries, Cambridge, Mass., 2001.
3. "OpenGIS Abstract Specification, Topic 12: OpenGIS Service Architecture," document 02-112, OpenGIS Consortium, Wayland, Mass., Jan. 2002; www.opengis.org/techno/abstract/02-112.pdf.
4. "OpenGIS Geography Markup Language (GML) Implementation Specification," document OGC 02-023r4, OpenGIS Consortium, Jan. 2003; www.opengis.org/techno/documents/02-023r4.pdf.
5. "OpenGIS Abstract Specification, Topic 6: The Coverage Type and Its Subtypes, document 00-106, OpenGIS Consortium," Mar. 1999; www.opengis.org/techno/abstract/00-106.pdf.
6. M. Mabrouk et al., "OpenGIS Location Services (OpenLS): Core Services," document OGC 03-006r1, OpenGIS Consortium, Apr. 2003; www.opengis.org/techno/03-006r1.pdf.
7. D. Nebert, "Interoperable Spatial Data Catalogs," *Photogrammetric Eng. and Remote Sensing*, vol. 65, no. 5, 1999, pp. 573–575.
8. "Styled Layer Descriptor Implementation Specification," document OGC 02-070, OpenGIS Consortium, Sept. 2002; www.opengis.org/techno/specs/02-070.pdf.
9. W. Litwin, L. Mark, and N. Roussopoulos, "Interoperability of Multiple Autonomous Databases," *ACM Computing Surveys*, vol. 22, no. 3, 1990, pp. 267–293.
10. G. Wiederhold, "Mediation to Deal with Heterogeneous Data Sources," *Proc. Interop '99: Interoperating Geographic Information Systems 2nd Conf.*, Springer-Verlag, 1999, pp. 1–16.
11. F. Leymann, "Web Services Flow Language (WSFL 1.0)," IBM, May 2001; www-3.ibm.com/software/solutions/webservices/pdf/WSFL.pdf.
12. S. Thatte, "XLANG: Web Services for Business Process Design," Microsoft, 2001; www.gotdotnet.com/team/xml%5Fwsspecs/xlang%2Dc/.
13. E. Christensen et al., "Web Services Description Language (WSDL) 1.1," W3C note, Mar. 2001; www.w3.org/TR/2001/NOTE-wsdl-20010315.
14. D. Box et al., "Simple Object Access Protocol (SOAP) 1.1," W3C note, May 2000; www.w3.org/TR/2000/NOTE-SOAP-20000508/.
15. D. Abel et al., "Towards Integrated Geographical Information Processing," *Int'l J. Geographical Information Science*, vol. 12, no. 4, 1998, pp. 353–371.

Nadine Alameh is a senior computer engineer at Global Science & Technology. Her research interests focus on the interoperability and distribution of geographic information systems (GIS) and on the standardization of GIS services and data models. Alameh holds two MSs and a PhD from MIT. She is an architect in the OpenGIS Consortium's Interoperability Program, a director in IEEE's Women In Engineering DC chapter, and the vice chair of IEEE's Geoscience and Remote Sensing Society DC chapter. Contact her at alameh@gst.com.