Toward A Service Web: Integrating the Semantic Web and Service Orientation

John Domingue, Knowledge Media Institute, The Open University

Dieter Fensel, University of Innsbruck

By a variety of metrics, the Semantic Web is doing very well. For example, the 2006 Gartner report included the “Corporate Semantic Web” as one of three key technology themes, and in 2007, Sir Tim Berners-Lee briefed a hearing of the US Congress subcommittee on the topic. The current state of the Semantic Web is said to be comparable with that of the early days of the Web. Until very recently, growth has been linear, but signs of exponential growth are beginning to emerge.

Given this promising situation, we should ask ourselves where and how we expect the Semantic Web to continue to grow in the near- to mid-term future. Our view is that the Semantic Web will come to the fore through a comprehensive integration with service orientation. In particular, we envisage that the combination of Semantic Web and SOAs will lead to the creation of a “service Web” – a Web where billions of parties are exposing and consuming billions services seamlessly and transparently and where all types of stakeholders, from large enterprises to SMEs and singleton end users, engage as peers consuming and providing services within a network of equals.

Service orientation today

A Web service encapsulates discrete functionality through an interface that’s accessible via standard Internet protocols. As such, Web services have proven very popular in industry. The technology abstracts over legacy hardware and software platforms and enables the exposure of commercial services to a potentially wide audience. Web services have also proven to be useful in B2B scenarios. Consequently, service-oriented frameworks such as SAP’s NetWeaver (see www.sap.com/platform/netweaver) now dominate IT solutions for large enterprises.

Web services have, however, failed to make a significant impact on the Web. Current estimates are that the Web currently contains 30 billion pages, with 10 million new static pages added each day. In contrast, only 12,000 true Web services—Web services with active endpoints described with a Web Services Description Language (WSDL) file—currently exist on the Web. SOA solutions are thus restricted in their application context to being an in-house solution for corporations. Realizing service orientation’s full potential will require integrating SOA with both semantic and Web technologies. Here, we briefly outline several principles that must be followed to successfully achieve this.

Enhancing SOA with semantic technologies

Current standards for describing Web services use syntactic (XML-based) notations such as WSDL. Because these descriptions aren’t machine readable, IT staff must carry out all of the tasks associated with creating and maintaining Web service-based applications. The requirement for specialist workers to be involved in all points in the Web service life cycle causes numerous problems, the overriding one being the lack of scalability. Maintaining millions of services, let alone billions, to cope with environmental and context changes solely through human effort isn't feasible.

Tasks such as Web service discovery, composition, and invocation can be automated to a certain extent, so we can overcome this problem by applying semantic technologies. For example, semantics enable programs to access services through a description of offered capability rather than as an end point. The use of semantics thus forms a scalable access layer over Web service data and processes. Key to this is the creation of a semantic service bus that supports service usage through semantic formulations.

More generally, combining service orientation with semantics will require developing a comprehensive framework and architecture following several principles.

Service-oriented principle

This principle captures a distinct approach to analysis, design, and implementation and introduces specific subprinciples that govern aspects of communication, architecture, and processing logic. These include service reusability, loose coupling, abstraction, composability, autonomy, statelessness, and discoverability. With respect to service orientation, which enables a service-level view of an organization, there's a need to further distinguish services along several dimensions.

First, services can be distinguished according to the functionality they provide within an architecture, namely business and middleware services. Business services are services that various service providers supply through their back-end systems. Additionally, they're the subject of integration and interoperation within the architecture and can provide a certain value for users (such as booking a hotel room). On the other hand, middleware services are the main facilitators for the integration and interoperation of business services (such as discovery and mediation).

Second, services can be differentiated according to their abstraction level within an architecture – namely, Web services and services. A Web service is a general service that might take several forms when instantiated (such as purchasing a flight), whereas a service is an actual instance of the Web service that is consumed by and provides a concrete value for a user (such as the purchase of a particular flight from Innsbruck to Vienna). This distinction would be used for business services in the architecture.

Semantic principle

Combining semantics with service orientation lets you define scalable, semantically rich, formal service models founded on ontologies. This allows the total or partial automation of tasks such as service discovery, contracting, negotiation, mediation, composition, and invocation. A semantic service-oriented approach to the modeling and implementation of service-based applications will enable the scalable and seamless interoperation, reusability, discovery, and composition of the components to be used or reused.

Distributed principle

The distributed principle is the process of aggregating several computing entities' power to collaboratively run a single task, transparently and coherently, so that those entities appear as a single centralized system. Applying this principle to the architecture middleware system will allow the transparent distribution of components over the network so that executing processes running in a middleware can be scaled across numerous physical servers over a network. The distributed principle would also apply to business services, letting running processes span across enterprises distributed over a network.

User-centric principle

The user-centric principle puts the user in the center of the architecture. It refers to concepts such as personalizing business services, facilitating service usability, promoting multichannel access and service delivery, building trust, and achieving efficiency, accountability, and responsiveness according to user requirements. It will also facilitate the seamless implementation of business processes across organizational boundaries.

SOA and the Web

The Web can be understood as a collection of principles and highly scalable means for electronic publication. We believe that applying these principles to service orientation will lead to a global, dynamically changing environment of services accessible for third-party usage. Within this environment, services will undergo many changes; for instance, they will

- appear, disappear, and change location;
- initially be free, then transform to pay-per-use; and
- occasionally be blocked, out of service, or inspected for antitrust violations.

The provision of Web-based lightweight integration infrastructures will facilitate openness and easy adoption for both the service provider and consumer. Moreover, the adoption of Web principles will open service orientation beyond the boundaries of single organizations. We advocate several main Web principles for widespread acceptance.

Openness

In principle, anybody can contribute to the Web as a provider or consumer of information. Openness is a major necessity for a Web-scale environment. Using the infrastructure must be as simple, smooth, and unrestricted as possible for both service providers and users.

Interoperability

Interoperability should be provided through the integration of heterogeneous proprietary and legacy solutions through a common interface. Interoperability on the Web is platform and vendor neutral to let all providers and requesters of information to participate on an even basis.

Decentralized changeability and dynamicity

Content can appear, be modified, or disappear in an ad hoc fashion. That is, the provisioning and modification of content is under the distributed control of peers rather controlled by a central authority. Central control would hamper access and therefore scalability. In this respect, an element of chaos or "messiness" should be tolerated.

Automation of central mechanisms to route requests or responses

Manually generated repositories are inherently nonscalable and costly, and they quickly become outdated. You could argue that Web sites such as Google centralize control or access. However, what they actually implement is an abstraction process for accessing and caching information. In the Web's early days, sites were accessed by magic numbers (and later by magic names), and lists of bookmarked pages were considered valuable intellectual property. Through search engines, literal numbers and names have been replaced by keyword retrieval and ranking based on relevance factors. So, we can view the Google search form as an abstraction over the browser address bar. (Still, search engines such as Google can be misused if central control is used to manipulate access to implement censorship or promote commercial interests.)

Enabling n:m relationships to maximize interaction

In contrast to email, where the content is targeted at specific receivers, the Web is based on anonymous distribution through publication. In principle, information is disseminated to any potential reader, which email can achieve only through spam-like processes.

Integrating these principles into SOA will require

- A worldwide addressing schema. A simple form for this is a unique name and, more elaborately, a description of the capability of a service (that is, the degree to which it can be used to achieve a certain goal).
- A transport layer (a protocol) to transmit requests for and the results of service invocations.
- A platform-independent interface to process (client) service requests and access to service implementations. From a purely technological viewpoint, the mechanisms used in Web 2.0 are similar to the "standard" Web. However, Web 2.0 brings numerous Web-related concerns to the fore that can facilitate the creation of a Service Web, including

- blurring the distinction between content consumer and content provider,
- providing a lightweight semantic mechanism through tagging,
- blurring the distinction between service consumer and service provider, and
- blurring the distinction between machine- and human-based computation.

Web 2.0 technology can thus provide a means to generate and access the semantic service layer outlined above. Incorporating human interaction and cooperation in a comprehensive fashion creates a route to solving tasks such as service ranking or mediation, which otherwise remain computationally infeasible. In several scenarios, Web 2.0 and human computing approaches, together with their underlying social consensus-building mechanisms, have proven the potential of combining human-provided services with services provided via automated reasoning. In our view the transparent provisioning of services abstracting over whether the underlying provider is a human or machine will significantly increase the overall quality of services available to the end-user.

A promising future

From our point of view, the Semantic Web's future is very promising. Moreover, we believe that the successful integration of Semantic Web and service-oriented technologies will form the main pillar of the software architecture of the next generation of computing infrastructure. We envision a transformation of the Web from a Web of static data to a global computational resource that truly meets the needs of its billions of users, placing computing and programming within a services layer to facilitate computing's real goal: placing problem solving in the hands of end users through a truly balanced cooperative approach.

John Domingue is the deputy director of the Knowledge Media Institute at The Open University, UK. Contact him at j.b.domingue@open.ac.uk.

Dieter Fensel is the director of the Digital Enterprise Research Institute at the Leopold Franzens University of Innsbruck, Austria. Contact him at dieter.fensel@deri.at.