Services Mashups

The New Generation of Web Applications

Djamal Benslimane
Lyon University

Schahram Dustdar
Vienna University of Technology

Amit Sheth
Wright State University

The Internet and related technologies have created an interconnected world in which we can exchange information easily, process tasks collaboratively, and form communities among users with similar interests to achieve efficiency and improve performance. Web services are emerging as a major technology for deploying automated interactions between distributed and heterogeneous applications, and for connecting business processes, which might span companies' boundaries.[1] Various standards support this deployment, including, for enterprises, the Web Services Description Language (WSDL), UDDI, and SOAP. These standards support the definition of Web services and their advertisement to the potential user community, binding for invocation purposes, and reuse. At the same time, use of "lighter-weight" approaches to services, especially for Web applications, is increasing. Here, the Web APIs and RESTful (Representational State Transfer) reign supreme.

Service Mashups

Recently, in the context of the Web, the *mashup* concept has emerged, and researchers have developed a huge number of Web 2.0 applications. But what exactly does mashup mean? It simply indicates a way to create new Web applications by combining existing Web resources utilizing data and Web APIs. Mashups are about information sharing and aggregation to support content publishing for a new generation of Web applications. By extension, service mashups — the theme of this special issue — aim to design and develop novel and modern Web applications based on easy-to-accomplish end-user service compositions. Combining Web service technologies with fresh content, collaborative approaches (such as Web 2.0 technologies, tags, and microformats), and possibly Web data management and semantic technologies (RSS, RDFa, Gleaning Resource Descriptions from Dialects of Languages, and the Sparql Protocol and Rdf Query Language) is an exciting challenge for both academic and industrial researchers building a new generation of Web-based applications. Researchers have created different mashup tools and platforms, letting developers and end-users access and compose various data that Web applications can provide. IBM's QEDWiki (http://services.alphaworks.ibm.com/qedwiki/), Yahoo! Pipes (http://pipes.yahoo.com), Google Mashup Editor (http://code.google.com/gme/), and Microsoft's Popfly (www.popfly.com) are some well-known examples of mashup platforms that users have largely adopted. Yet these platforms and associated tools represent only early and limited sets of capabilities that are sure to be followed by more powerful and flexible alternatives.

Key Research Issues in Service Mashups

Although many have already adopted the (service) mashup concept and recognized its value, realizing the concept is still challenging, and much work remains before we'll see mashup applications in a mature stage. Let's briefly discuss some key issues we must consider in the future to improve sharing (registration and publication), finding (search and discovery), reusing (invocation), and integrating (mediation and composition) services.

The first key challenge is that of semantic heterogeneity. Compared to data, services can present a broader form of heterogeneity. Correspondingly, the Web services research community has identified a broader form of semantics — data (I/O), functional (behavioral), nonfunctional (quality of service, policy), and execution (runtime, infrastructure, exceptions).[2] Several research projects have looked at semantics for traditional (WSDL or SOAP) Web services to help address heterogeneity and mediation challenges, and the community took a step toward supporting semantics for Web services by adopting Semantic Annotation for WSDL (SAWSDL) as a W3C recommendation in August 2003 (www.w3.org/2002/ws/sawsdl/). Now, attention has shifted to using semantics for community-created content, as with the Semantic MediaWiki,[3] and for WebAPIs and RESTful services, such as hRESTS,[4] SA-REST (Semantic Annotation of RESTful Services), and smart mashups.[5] We believe that existing mashup approaches and tools must move one step further in order to use semantics approaches to deal with service interoperability and integration (including mediatability[6]). To do so, we must have an open eye on how we might build new solutions upon existing semantic Web technologies using appropriate Web 2.0 and Semantic Web approaches and technologies that complement each other.[7]

Most service mashup solutions assume that the needed services are known and available somewhere on the Web. Programmableweb.com and APIHut[8] are current approaches developers can use to share, find, and reuse Web APIs. These approaches are building a nice ecosystem in which people can reuse Web APIs and build mashups. For complex applications to meet enterprise needs, we must also develop advanced capabilities leading to dynamic configuration and composition.

In this Issue

The four articles in this issue address some of the challenges inherent in developing mashup-services-based advanced Web applications.

3-line pull quote 3-line pull quote 3-line pull quote 3-line pull quote 3-line pull quote 3-line pull quote 3-line pull quote

In "Mashing Up Search Services," Daniele Braga, Stefano Ceri, Davide Martinenghi, and Florian Daniel propose a visual service mashup language for graphically composing and automatically executing queries over search services. They define search services as services with a variable number of ranked data. The proposed language lets users declaratively specify a query and mash up registered services in a drag-and-drop fashion to compose that query. The authors build service compositions as directed acyclic graphs whose nodes are service invocations and whose arcs are connections between services. The authors propose a physical service access plan for the Web service composition's execution needs, such as generating a schedule of series or parallel service invocations, orchestrating such invocations, and joining data from different services into a ranked output. They argue that their language enables interesting runtime environments capable of deriving different executable service invocation strategies. Researchers are still working to meet other challenges, such as mastering more complex dependencies among services and supporting reliable and transactional Web services.

In "Composing RESTful Services and Collaborative Workflows: A Lightweight Approach," Florian Rosenberg, Fancisco Curbera, Matthew J. Duftler, and Rania Khalaf propose the extensible Bite language based on a lightweight process composition model for both Web data-driven applications and Web workflow composition. Bite combines SOA process composition principles with REST architectural requirements and workflow functionalities. It lets users implement RESTful service composition and interactive workflows.

In "An Online Platform for Web APIs and Service Mashups," E. Michael Maximilien, Ajith Ranabahu, and Karthik Gomadam propose an online mashup platform that enables the construction, reuse, sharing, deployment, and management of Web APIs and service mashups. The proposed platform's main characteristic resides in its domain-specific language, which is introduced to explicitly represent the activities that a mashup designer must fulfill, such as data mediation and service protocol mediation. The authors have deployed the IBM sharable code platform on IBM alpha works services.

Finally, In "Understanding Mashup Development," Jin Yu, Boualem Benatallah, Fabio Casati, and Florian Daniel provide an overview of some popular and representative mashup development tools and frameworks. Mashup in this article refers to Web applications comprising data, application logic, and UIs of existing applications or services. The authors compare and discuss these tools and frameworks by considering four dimensions: the *component model*, which describes the mashup components' characteristic properties; the *composition model*, which specifies how components are glued to create a mashup application; the *development environment*, and the *runtime environment*. In their discussion, the authors conclude that mashup tools' main characteristics are simplicity, usability, and ease of access. They also identify some perspectives that could improve mashup tools, such as describing UIs as components that can be reused and integrated like services.

Service mashups are becoming very important as Web applications and Web data grow. Efforts are still needed before we'll be able to easily semantically connect existing Web applications, and we must take into account the challenges we've discussed here. We're convinced that the research community will soon provide new solutions and tools that have real commercial impact.

References

1. M.P. Papazoglou and D. Georgakopoulos, "Service Oriented Computing," *Comm. ACM*, vol. 46, no. 10, 2003, pp. 24–28.
2. K. Sivashanmugam et al., "Adding Semantics to Web Services Standards," *Proc. Int'l Conf. Web Services* (ICWS), CSREA Press, 2003, pp. 395–401.
3. M. Krötzsch, D. Vrandecic, and M. Völkel, "Semantic Mediawiki," *Proc. Int'l Semantic Web Conf.*, 2006, Springer, pp. 935–942.
4. J. Kopecky, K. Gomadam, and T. Vitvar, *hRESTS: An HTML Microformat for Describing RESTful Web Services*, Kno.e.sis tech. report, Wright State Univ., 2008; http://knoesis.org/research/srl/hRESTs/.
5. A.P. Sheth, K. Gomadam, and J. Lathem, "SA-REST: Semantically Interoperable and Easier-to-Use Services and Mashups," *IEEE Internet Computing*, vol. 11, no. 6, 2007, pp. 91–94.
6. K. Gomadam et al., "Mediatability: Estimating the Degree of Human Involvement in XML Schema Mediation," to appear in *Proc. IEEE Int'l Conf. Semantic Computing*, IEEE Press, 2008.
7. A. Ankolekar et al., "The Two Cultures: Mashing up Web 2.0 and the Semantic Web," *Proc. Int'l Conf. World Wide Web*, ACM Press, 2007, pp. 825–834.
8. K. Gomadam et al., "A Faceted Classification-Based Approach to Search and Rank Web APIs," to appear in *Proc. IEEE Int'l Conf. Web Services*, IEEE Press, 2008.

Djamal Benslimane is a full professor of computer science at Lyon University, France, and a member of the LIRIS (Laboratoire d'InfoRmatique en Image et Système d'information) research laboratory. He has a PhD in computer science from Blaise Pascal University. Contact him at djamal.benslimane@liris.cnrs.fr; www710.univ-lyon1.fr/~dbenslim/.

Schahram Dustdar is a full professor of computer science with a focus on Internet technologies heading the Distributed Systems Group at the Vienna University of Technology. He is also director of the Vita Lab. Contact him at dustdar@infosys.tuwien.ac.at; www.infosys.tuwien.ac.at/staff/sd/.

Amit Sheth is the LexisNexis Ohio Eminent Scholar and the director of the Kno.e.sis Center at Wright State University. He is a fellow of the IEEE. Contact him via http://knoesis.wright.edu/amit.