Available online at www.sciencedirect.com

JOURNAL OF Web Semantics

www.elsevier.com/locate/websem

Web Semantics: Science, Services and Agents on the World Wide Web 1 (2004) 309–320

Review

Web services: problems and future directions

Hongbing Wang[a,b,c,*], Joshua Zhexue Huang[c], Yuzhong Qu[b], Junyuan Xie[a]

[a] *Department of Computer Science and Technology, Nanjing University, Nanjing 210093, China*
[b] *Department of Computer Science and Engineering, Southeast University, Nanjing 210096, China*
[c] *E-Business Technology Institute, The University of Hong Kong, Hong Kong, China*

Received 3 September 2003; received in revised form 29 January 2004; accepted 5 February 2004

Abstract

Recently, Web services have generated great interests in both vendors and researchers. Web services, based on existing Internet protocols and open standards, can provide a flexible solution to the problem of application integration. With the help of WSDL, SOAP, and UDDI, Web services are becoming popular in Web applications. However, the current Web services architectures are confronted with a few stubborn problems, for instance, security. In this paper, we shall give an overview of these problems. We believe that solving these problems will become crucial to success of Web services. In the end, we predict distinct advances in semantic Grid services.

Keywords: Web services; Security; Composition; Semantics; Grid services

1. Introduction

In the past few years, ad hoc approaches have been used in business-to-business applications to take advantage of the basic Internet infrastructure. Recently, Web services are emerging as a systematic and extensible framework for application-to-application interaction, built on top of existing Web protocols and open XML standards.

Web services are a new breed of Web applications. They are self-contained, self-describing, modular applications that can be published, located, and invoked across the Web. Web services perform functions that can be anything from simple requests for information to creating and executing complicated business processes. Once a Web service is deployed, it can be discovered and invoked by other applications (or other Web services).

The key advantage of using Web services is the ability to create applications on the fly through the use of loosely coupled, reusable software components. This has fundamental implications in both technologies and business applications. Software can be redelivered and paid for as fluid streams of services as opposed to packaged products. It is possible to achieve automatic and dynamic interoperability between systems to accomplish business tasks. Business services can be completely decentralized and distributed over the Internet and accessed by a wide variety of communications devices. Businesses can be released from

* Corresponding author. Tel.: +86-25-83792465; fax: +86-25-83794838.
E-mail addresses: hbw@seu.edu.cn (H. Wang), jhuang@eti.hku.hk (J.Z. Huang), yzqu@seu.edu.cn (Y. Qu), jyxie@nju.edu.cn (J. Xie).

the burden of complex, low, and expensive software integration and focus instead on the value of their offerings and mission critical tasks. Then, the Internet will become a global common platform where organizations and individuals communicate with each other to carry out various commercial activities and to provide value-added services. The barriers to providing new offerings and entering new markets will be lowered to enable access for small and medium-sized enterprises. The dynamic enterprises and dynamic value chains become achievable and may be even mandatory for competitive advantages [24].

The Web services framework is divided into three areas—communication protocols, service descriptions, and service discovery—and specifications are being developed for each. In this article, we look at the following specifications that are currently most salient and stable in each area:

1. The simple object access protocol (SOAP) that enables communications among Web services;
2. The Web Services Description Language (WSDL) that provides a formal, computer-readable description of Web services; and
3. The universal description, discovery and integration (UDDI) directory that is a registry of Web services descriptions.

SOAP is fundamentally a stateless, one-way message exchange paradigm that enables applications to create more complex interaction patterns (e.g., request/response, request/multiple responses, etc.) by combining one-way exchanges with features provided by an underlying protocol and/or application-specific information. Although SOAP provides a solid framework for information exchange, it lacks semantics on the application-specific data it conveys, such as the routing of SOAP messages, reliable data transfer, firewall traversal, etc. Also, SOAP provides a full description of the required actions taken by a SOAP node on receiving a SOAP message [78].

At its core, a SOAP message has a very simple structure: an XML element with two children elements, one containing the header and the other the body. The header contents and body elements are also represented in XML.

SOAP messages can be transported over HTTP for the runtime invocation. The HTTP protocol plays the bridging role for interactions between computer systems.

WSDL provides a model and an XML format for describing Web services [79]. WSDL defines services as collections of network endpoints or ports. In WSDL, the abstract definition of endpoints and messages is separated from their concrete network deployments or data format bindings. This allows the reuse of abstract definitions of messages that describe the data being exchanged and port types that represent collections of operations. The concrete protocol and data format specifications for a particular port type constitute a binding. A port is defined by associating a network address with a binding. A collection of ports defines a service.

UDDI provides a mechanism for clients to find Web services [80]. Web services are meaningful only if potential users may find information sufficient to permit their execution. The focus of universal description discovery & integration is the definition of a set of services supporting the description and discovery of (1) businesses, organizations, and other Web services providers, (2) the Web services they make available, and (3) the technical interfaces which are used to access those services. Based on a common set of industry standards, including HTTP, XML, XML Schema, and SOAP, UDDI provides an interoperable, foundational infrastructure for a Web services-based software environment for both publicly available services and services only exposed internally within an organization.

A UDDI registry is similar to a CORBA trader and can be considered as a DNS service for business applications. A UDDI registry has two kinds of clients: businesses who want to publish a service description (and its usage interfaces) and clients who want to obtain services descriptions of a certain kind and bind the services programmatically (using SOAP).

The UDDI information contains four levels. The top level is the business entity that provides the general data about a company, such as its address, a short description, contact information, and other general identifiers. This kind of information can be seen as the white pages of UDDI. Associated with each business entity is a list of business services, including the description of each service and the categories of the service, for instance, purchasing, shipping, etc. This can be considered as the yellow pages of UDDI. Within a business service, one or more binding templates define the green pages that provide more technical information about a Web service [18].

The rest of this paper is organized as follows. First, we compare briefly Web services with other distributed component models such as CORBA. Then, we discuss the current status of Web services, including problems and potential solutions. Finally, we identify some key strategic directions for further development.

2. Web services: a better rpc and distributed component model

Web services were designed to tackle the problem of integration of heterogeneous sources and make heterogeneous systems interoperable. Technologies such as CORBA, RPC, and EDI had the same objectives, but each having its own infrastructure. Therefore, solutions to the heterogeneous system integration problem with these technologies are very expensive.

One difficulty in using the distributed technologies such as CORBA is dealing with languages. CORBA's solution to this problem is to move the language into the background through an Interface Definition Language (IDL) [81]. Although this is a good idea, the IDL language looks much like C++ and developers must also understand language bindings in order to use CORBA. If there was a way to replace IDL-like languages with higher-level specifications, surely the communication could be simplified using meaningful messages. Web services, defined as self-contained, self-describing modular applications that can be published, located, and invoked across the Web, have achieved this goal and can automatically get into the Web-wide scope, unlike CORBA that was primarily used in the enterprise-wide scope (at least initially).

Web services are a good choice for loosely coupled architectures [16]. The CORBA architecture was more suitable for intra-enterprise environments, while the technical features and choices of Web services make Web services more reusable and thus more appropriate for inter-enterprise and global environments.

We can thus explain the success of Web services by viewing them as a technology based on maximal decoupling (and thus maximal reusability) available over the existing economic infrastructure (the Internet). Their power is not so much in their technology (the idea of RPC is nothing new) but rather that they offer a Web-native XML-based solution [11], so we can rapidly design, implement, and deploy Web services on the Internet.

SOAP is a lightweight protocol intended for exchanging structured information in a decentralized, distributed environment. It has been designed to be independent of any particular programming model and other implementation specific semantics. Because it is based on the ubiquitous HTTP protocol in conjunction with XML as the message syntax, computer system boundaries are easy to overcome and the notion of the "better RPC" or even "better distributed component management technology" was born, leading to the wide awareness of Web services in the developer and researcher community. Because SOAP messages can be transported over HTTP for the runtime invocation, interaction between different applications becomes easier due to the HTTT protocol. If only packaged applications like the enterprise resource planning systems, such as Oracle or SAP, were to expose WSDL interfaces, then integrating them with Web services would be easy, fast, cheap and manageable (better along all possible dimensions). In such a scenario, Web services would be a "better enterprise application integration solution," overcoming the current costly integration technology deployments.

On the other hand, we must look into the opposite side of the problem. The standard protocol that most services use currently is SOAP over HTTP, though this requirement is not necessary. Furthermore, SOAP alone can not address some crucial problems, such as reliability and security. All these deserve serious considerations in the research community. Currently, Web services activities include efforts to upgrade SOAP to provide better security and reliability.

3. Some problems of Web services and solutions

SOAP, WSDL, and UDDI are important technologies to enable Web services. However, to fully satisfy the requirements of business applications, the current technologies have shortcomings. In this section we discuss three major problems and research directions to upgrade the existing technologies.

3.1. Security problems and solutions

Let us use a simple travel scenario to illustrate the security problem of Web services. More than three pieces of the Web services framework are required to interact properly to complete the travel scenario. At the very least, we have to ensure that transactions like the electronic check-ins were conducted in a secure environment and that messages were reliably delivered to the destinations. Why must we build additional security when we have technologies such as secure multipurpose internet mail extensions (S-MIME), HTTP secure (HTTPS), and Kerberos available? The answer lies in the difference between end-to-end and single-hop usage. Business messages typically originate from one application and then is transferred to another one. Mechanisms such as secure sockets layer are great for securing (for confidentiality) a direct connection from one machine to another, but they are of no help if the message has to travel over more than one connection.

It is well known in the penetration testing community that attacks to modern systems are usually not at the network level but within the application protocols (e.g., HTTP in the case of Web systems). This means that the firewall will simply pass the attack traffic along with any legitimate HTTP requests as it looks for port 80 traffic only, and does not concern the malformed HTTP traffic or application-specific attacks (such as SQL injection). In many cases where SSL is used, the firewall cannot see into the traffic stream. In some respects, Web services have adopted the HTTP's tunneling idea, by allowing all systems, both internal and external, to communicate on HTTP ports so flexibility is obtained. What is removed is the control the firewall may have, and ultimately the application servers are opened up to "application level" attacks in exactly the same way as insecure and vulnerable Web servers today.

Basically, the security problems that are likely to affect Web services are the same as those that have affected the conventional Web-based systems. Many of these were discussed at length in [22,36,42, 52,62,65,66]. Here, we summarize the current situation as follows: security is critical to the adoption of Web services by enterprises, but, as it stands today, the Web services framework does not meet basic security requirements.

The fact that Web services involve exchange of messages means that securing the message exchange is an important issue to consider when building and using Web services. In the Web services context, security means that the recipient of a message should be able to verify the integrity of the message and to make sure that it has not been modified. The recipient should have received a message confidentially so that unauthorized users could not read it, know the identity of the sender and determine whether or not the center is authorized to carry out the operation requested in the message. These are usually met through encrypting messages [75].

On the other hand, because Web services allow all systems, both internal and external, to communicate on HTTP ports, the application servers are inevitably opened up to "application level" attacks.

A few standards have come out to alleviate the message security problem, including WS-security and various other initiatives (mostly from the major vendors and PKI suppliers) towards enabling digital signatures on XML messages and transactions. But the "application level" attacks were hardly concerned. These new standards are briefed below.

The XML signature specification [84] was a joint effort of W3C and IETF. It aims to provide data integrity and authentication (both message and signer authentication) features, wrapped in the XML format.

W3C's XML encryption specification [85] addresses the issue of data confidentiality using encryption techniques. Encrypted data is wrapped inside the XML tags defined by the XML encryption specification.

WS-security [86] from OASIS defines the mechanism to include integrity, confidentiality, and single message authentication features within a SOAP message. WS-security makes use of the XML signature and XML encryption specifications and defines how to include digital signatures, message digests, and encrypted data in a SOAP message.

Security Assertion Markup Language (SAML) [87] from OASIS provides a means for partner applications to share user authentication and authorization information. This is essentially the single sign-on (SSO) feature being offered by all major vendors in their e-commerce products. In the absence of any standard protocol on sharing authentication information, vendors normally use cookies in the HTTP communication to implement SSO. With the advent of SAML, the same data can be wrapped in XML in a standard way, so that cookies are not needed and interoperable SSO can be achieved.

eXtensible Access Control Markup Language (XACML) [88] presented by OASIS allow you to express your authorization and access policies in XML. XACML defines a vocabulary to specify subjects, rights, objects, and conditions—the essential bits of all authorization policies in today's e-commerce applications.

All standards mentioned above do not address the "application level" attacks [56,68,69,70,75]. Provided some more detailed descriptions of Web services security. Generally speaking, "application level" attacks were not concerned enough by current standards and researches

3.2. *Composition problems and solutions*

Complex business interactions require support for higher levels of business functionality. Business interactions are typically long execution processes and involve multiple interactions between partners. To deploy and effectively use these types of services, we must be able to represent business processes and states of services and to create service compositions (complex aggregations) in a standardized and systematic fashion. Several proposals for accomplishing this task exist; see, for example, Web Services Flow Language [82], XLANG [72], and BPEL4WS [83].

The industry has used a number of terms to describe how components can be connected together to build complex business processes. Workflow and document management systems have existed as a means to handle the routing of work between various resources in an IT organization. These resources might include people, systems, or applications, and typically involve some human intervention. Business process management systems (BPMS) have also been used to enable a business to build a top-down process design model, consisting of various integration activities (e.g., integration to a legacy system). BPMS systems [49,50] would typically cover the full lifecycle of a business process, including modeling, executing, monitoring, management, and optimization tasks. With the introduction of Web services, terms such as "Web services composition" and "Web services flow" were used to describe the composition of Web services in a process flow. More recently, the terms *orchestration* and *choreography* have been used to describe this too. Orchestration describes how Web services can interact with each other at the message level, including business logic and execution order of the interactions. These interactions may span applications and/or organizations, and result in a long-lived, transactional, multi-step process model.

3.2.1. *Early work*

Early work in Web services composition included eCo, WSCL (Web Services Conversation Language), XLANG, and Web Services Flow Language(WSFL).

CommerceNet initially created the eCo framework [90] to demonstrate the value of integrating e-commerce services, with a focus on the document exchanges required for B2B integration. The specification had a vague notion of orchestration, showing how a process can be built from Web services.

The Web Services Conversation Language outlined a simple conversation language standard [89], focusing on modeling the sequence of interactions between Web services. This was somewhat analogous to Web services choreography.

Microsoft initially developed the XLANG specification for the Microsoft BizTalk Server. XLANG was focused on the creation of business processes and the interactions between Web service providers. The specification provided support for sequential, parallel, and conditional process control flows. It also included a robust exception handling facility, with support for long-running transactions through compensation. XLANG used WSDL as a means to describe the service interface of a process.

The Web Services Flow Language was an IBM proposal to describe both public and private process flows. WSFL defines a specific order of activities and data exchanges for a particular process. It defines both the execution sequence and the mapping of each step in the flow to specific operations, referred to as flow models and global models. The flow model represents the series of activities in the process, while the global model binds each activity to a specific Web service instance. A WSFL definition can also be exposed with a WSDL interface, allowing for recursive decomposition. WSFL supports the handling of exceptions but has no direct support for transactions.

3.2.2. BPEL4WS

The Web services workflow specifications outlined by XLANG and WSFL have recently been superseded by a new specification from IBM, Microsoft and BEA, called Business Process Execution Language for Web Services (BPEL4WS). BPEL4WS is a specification that models the behavior of Web services in a business process interaction [76]. The specification provides an XML-based grammar for describing the control logic required to coordinate Web services participating in a process flow. This grammar can then be interpreted and executed by an orchestration engine, which is controlled by one of the participating parties. The engine coordinates the various activities in the process, and compensates for the system when errors occur.

BPEL4WS is essentially a layer on top of WSDL, with WSDL defining the specific operations and BPEL4WS defining how the operations can be sequenced.

BPEL4WS provides support for both executable and abstract business processes. An executable process models the behavior of participants in a specific business interaction, essentially modeling a private workflow. Abstract processes, modeled as business protocols in BPEL4WS, specify the public message exchanges between parties. Business protocols are not executable and do not convey the internal details of a process flow. Essentially, executable processes provide the orchestration support described earlier while the business protocols focus more on the choreography of the services.

3.2.3. WSCI

The Web Services Choreography Interface (WSCI) is a specification from Sun, SAP, BEA, and Intalio that defines an XML-based language for Web services collaborations [8]. It defines the overall choreography describing the messages between Web services that participate in a collaborative exchange. The specification supports message correlation, sequencing rules, exception handling, transactions, and dynamic collaboration.

A key aspect of WSCI is that it only describes the observable or visible behavior between Web services. WSCI does not give the definition of executable business processes as defined by BPEL4WS. Furthermore, a single WSCI document only describes one partner's participation in a message exchange. In WSCI, there is no single controlling process managing the interaction.

WSCI can also be viewed as a layer on top of the existing Web services stack. Each action in WSCI represents a unit of work, which would typically map to a specific WSDL operation. WSCI can be considered as the glue around WSDL, describing how the operations can be choreographed. In other words, WSDL can be used to describe the entry points for each service available and WSCI describes the interactions among these WSDL operations. This is very similar to how BPEL4WS leverages WSDL.

3.2.4. BPML

The Business Process Management Language (BPML) is a meta-language for describing business processes [91]. Business Process Management Initiative, an independent organization chartered by Intalio, Sterling Commerce, Sun, CSC, and others, developed the specification. BPML was initially designed to support business processes that could be executed by a BPMS system. However, the first draft of BPML also incorporated the WSCI protocol. WSCI could be used to describe the public interactions and choreographies and the private implementations could be developed with BPML. Both BPML and WSCI share the same underlying process execution model and similar syntaxes. The specification can also be loosely compared to BPEL4WS, providing similar process flow constructs and activities. Basic activities for sending, receiving, and invoking services are available, along with structured activities that handle conditional choices, sequential and parallel activities, joins, and looping. BPML also supports the scheduling of tasks at specific times.

3.2.5. A brief analysis of emerging standards in Web services composition

We have presented an overview of the emerging standards [74] in the Web services composition arena in this subsection. Each standard has taken a different approach to composition. BPEL4WS primarily focuses on the creation of executable business processes, while WSCI is concerned with the public message exchanges between Web services. WSCI takes a collaborative and choreographed approach, requiring each participant in the message exchange to define a WSCI interface. BPEL takes an "inside-out" perspective, describing an executable process from the perspective of one of the partners. BPML has some complimentary components to BPEL4WS, both providing capabilities to define a business process. WSCI is now considered as a part of BPML, with WSCI defining the interactions between the services and BPML defining the business processes behind each service.

BPEL4WS has recently been submitted to OASIS (OASIS is a nonprofit, global consortium that drives the development, convergence and adoption of e-business standards, see http://www.oasis-open.org/home/index.php) and is emerging as the dominant orchestration standard.

3.2.6. *A brief overview of research work in Web services' composition*

Besides above efforts to constitute Web services composition standards, many researches have been carried out in Web services' composition. Narayanan and McIlraith [57] proposed an approach to automated compositions of Web services. They took as the starting point the DAML-S DAML + OIL ontology for describing the capabilities of Web services. They defined the semantics for a relevant subset of DAML-S in terms of a first-order predicate language. With the semantics in hand, they encoded their service descriptions in a Petri Net formalism and defined decision procedures for Web services simulation, verification, and composition.

Florescu et al. [26] presented an XML programming language XL specially designed for the implementation of Web services. XL is portable and fully compliant with W3C standards such as XQuery, XML Protocol, and XML Schema. One of the key features of XL is that it allows programmers to concentrate on the logic of their applications. XL provides high-level and declarative constructs for actions, which are typically carried out in the implementation of a Web service.

Su et al. [71] introduced a framework for modeling and specifying the global behavior of e-service compositions. This paper proposed conversation specifications as a formalism to define the conversations allowed by an e-service composition. By understanding properties of these conversations, this study provided a new approach to the design and analysis of "well-formed" e-service compositions.

Zeng et al. [77] provided a model of Quality Driven Web Services Composition. In this approach, individual Web services are federated into composite Web services whose business logic is expressed as a process model. Usually, several component services are able to execute a given task. In this paper, they advocated that the selection of component services should be carried out during the execution of a composite service, rather than at design-time. Accordingly, the paper proposed a global planning approach to optimally selecting component services during the execution of a composite service. Service selection was formulated as an optimization problem, which could be solved using efficient linear programming methods.

Interested readers can refer to [48,57,63,67] for more details.

3.3. *Semantic problems and solutions from semantic Web*

The current Web services technology basically provides a syntactical solution and still lacks the semantic part. A Web service is described in WSDL, outlining what input the service expects and what output it returns. To exploit their potentials (beyond the enterprise application integration), Web services must be able to orchestrate themselves into more complex services. Thus, we need methods to combine individual Web services into a distributed, higher-level service. The Web Service Flow Language (WSFL), which can express the sequencing of individual services, is taking the first steps. WSFL lets the user decide which Web services to combine and in what order. However, we still need a framework that semantically describes services so that software agents can locate, identify, and combine these services.

Many researchers believe that the Semantic Web vision of the next-generation Web, that enables computers unambiguously interpreting the Web content, addresses precisely this problem [33,39,54]. The Semantic Web project is Tim Berners-Lee's brainchild, seeking to create a machine processable Web. Semantic Web has advocates predominantly from the more research-oriented members of the Web community. Due to commercial interests, industrial player, including Microsoft, IBM, and BEA, on the other hand, have largely driven the development of Web Services.

In his opening keynote at the Twelfth International World Wide Web conference, the director of the World Wide Web Consortium explained how to make the two main thrusts of the development of the Web not compete, but work together. Berners-Lee claimed that Web Services meet immediate technology needs, while the Semantic Web has the potential for future exponential growth. There are many ways in which the two areas could interact in the future, and the W3C does not intend to limit their work to one area or the other.

Current Web services standards, such as SOAP, WSDL, XLANG, WSFL, BPEL4WS, WSCI, and BPML; all describe Web service content in terms of XML syntax. Unfortunately, XML alone lacks both a well defined semantics and sufficient expressive power to realize the vision of diverse Web services having wide-scale interoperability. Seamless interoperability between services that have not been pre-designed to work together requires programs to describe their own capabilities and understand other services' capabilities. To realize this vision, Web content, particularly Web service content and capabilities, may need to be described in a language that goes beyond XML. This problem is well addressed in the Semantic Web vision of the next-generation Web.

3.3.1. *Semantic Web*

The Semantic Web is not a separate Web but an extension to the current one, in which information is given well defined meaning, enabling computers and people to work in cooperation [12].

A key element to realizing the Semantic Web is developing a suitably rich language for encoding and describing the Web content. Such a language must have a well defined semantics, be sufficiently expressive to describe the complex interrelationships and constraints between the Web objects, and be amenable to automated manipulation and reasoning with acceptable limits on time and resource requirements. RDF [92], RDF Schema [93], DAML + OIL [94], and OWL [95] are relevant languages.

Resource Description Framework (RDF) is a foundation for processing metadata and provides interoperability between applications that exchange machine-understandable information on the Web. DAML + OIL is a semantic markup language for Web resources. It builds on top of earlier languages such as RDF and RDF Schema, and extends these languages with richer modeling primitives. OWL is a Web Ontology Language and can be used to explicitly represent the meaning of terms in vocabularies and the relationships between those terms. OWL has more expressive power of semantics than XML, RDF, and RDF-S, and thus goes beyond these languages in its ability to represent machine readable content on the Web. It is a revision of the DAML + OIL Web ontology language incorporating lessons learned from the design and application of DAML + OIL [10].

More detailed descriptions of Semantic Web are given in [2,3,13,19,23,35,37–41,43,44,46,47,58,60].

3.3.2. *Semantic Web services*

The Web services standards described above allow parties to easily exchange information in a standardized manner. These standards solve many problems on the technical level but the semantics of Web services and they do not address Web services descriptions as a whole. The Semantic Web services address this problem.

The Semantic Web services is to describe Web services' capabilities and content in a computer-interpretable language and improve the quality of existing tasks, including Web services discovery, invocation, composition, monitoring, and recovery.

In [20,25,54,61], authors discussed two major ongoing efforts to advance the World Wide Web. One is the Semantic Web research and the other is the Web services research. Both activities aim to make content on the Web accessible and usable not only for humans but also for computers. They considered that these two efforts are complementary and the ultimate goal is a unification of the two.

Authors of [44,59,53] presented some approaches to services discovery. Klein and Bernstein [44] proposed an ontology-based approach that employed the characteristics of one process taxonomy to increase the recall without sacrificing precision and computational complexity of the service retrieval process. Paolucci et al. [59] adopted DAML-S [4] as the service description language, and then discussed a matching algorithm between advertisements and requests described in DAML-S that recognizes various degrees of matching.

Mandell et al. [53] used a bottom-up approach to integrating the Semantic Web technology into Web services.

Initial attempts have already been made to apply the semantic Web services technology to a few applications [7,34,51].

A key element of semantic Web services is the creation of a description language. DAML-S, an ontology created by DAML-S Coalition (see http://www.daml.org/services/members.html) with support from DARPA, is such a description language. Ankolekar et al. [5] described the general structure of the ontology. The service profile provides advertisement of a service, and the service model provides enough information for an agent to make use of the service.

Comprehensive discussions on DAML-S can be found from [1,6,9,14,15,17,32,45,48,55,63,64,67].

4. Towards Grid services

Complex applications of Web services need a powerful computing infrastructure to support. The Grid computing provides such an infrastructure. The Open Grid Services Architecture (OGSA) [27] represents an evolution towards a Grid system architecture based on Web services concepts and technologies.

The OGSA integrates key Grid technologies [28–30] with Web services mechanisms to create a distributed system framework based on the Open Grid Services Infrastructure (OGSI) [73]. A Grid service instance is a service that conforms to a set of conventions, expressed as Web Service Definition Language (WSDL) interfaces, extensions, and behaviors, for such purposes as lifetime management, discovery of characteristics, and notification. Grid services provide for the controlled management of the distributed and often long-lived services that is commonly required in sophisticated distributed applications. OGSI also introduces the standard factory and registration interfaces for creating and discovering Grid services.

Grid service instances are made accessible to (potentially remote) client applications through the use of a Grid Service Handle (GSH) and a Grid Service Reference (GSR) [27,31]. A Grid Service Handle can be considered as a permanent network pointer to a particular Grid service instance. The GSH does not provide sufficient information to allow a client to access the service instance. The client needs to “resolve” a GSH into a Grid Service Reference. The GSR contains all the necessary information to access the service instance. A client application can use a Grid Service Reference to send requests directly to the specific instance at the specified network-attached service endpoint identified by the Grid Service Reference. OGSI provides a mechanism, the HandleResolver (see [73]) to support the client resolution of a Grid Service Handle into a Grid Service Reference. Besides, OGSI does not dictate the particular service provider–side implementation architecture.

Another important issue is how OGSI interfaces are likely to be invoked from client applications. OGSI exploits an important component of the Web services framework: the use of WSDL to describe multiple protocol bindings, encoding styles, messaging styles, and so on for a given Web service.

Recently, Grid services have been included in many international conferences such as SC2002 (the international conference for high performance computing and communications), Grid2002 etc.

Another interesting issue is semantic Grid services [21]. The state of play of the Grid today is reminiscent of the Web some years ago: there is limited deployment, largely driven by enthusiasts in the scientific community, with emerging standards and a degree of commercial uptake. The same might also be said of the Semantic Web. Meanwhile, the Web has seen a shift from machine-to-human communications (HTML) to machine-to-machine (XML). This is precisely the infrastructure needed for the Grid. It is appealing to infer from these similarities that Grid deployment will follow the same exponential model as the Web growth.

The visions of the Grid and the semantic Web have much in common but can perhaps be distinguished by a difference of emphasis. The Grid is traditionally focused on high performance computing, while the ambitions of the Semantic Web take it towards inference, proof and trust. The Grid we are now building is heading towards what we term the Semantic Grid: as the Semantic Web is to the Web, and the Semantic Grid to the Grid.

5. Conclusion

In this paper, we have presented Web services, an emerging technology for the Web. Three aspects of Web services were presented: the service security, the service composition, and the service semantics. They are critical to the successful deployment of Web services. Finally, we give a view of Grid services. We believe that the weaving of Semantic Web and Grid services will become an important technical trend.

Acknowledgements

This work is jointly supported by NSFC with grant no. 60173036 and JSNSF with grant no. BK2003001.

References

[1] J.-i. Akahani, K. Hiramatsu, K. Kogure, Coordinating Heterogeneous Information Services based on Approximate Ontology Translation, in: B. Burg, J. Dale, T. Finin, H. Nakashima, L. Padgham, C. Sierra, S. Willmott (Eds.), Agentcities: Challenges in Open Agent Environments, Springer-Verlag, 2003.

[2] D. Allsopp, P. Beautement, J. Carson, M. Kirton, Toward semantic interoperability in agent-based coallition command systems, in: Proceedings of the International Semantic Web Working Symposium (SWWS), July 30–August 1 2001, Stanford University, California, USA.

[3] B. Amann, C. Beeri, I. Fundulaki, et al., Ontology-based integration of XML web resources, in: Proceedings of the First International Semantic Web Conference (ISWC), Sardinia, Italy, 9–12 June 2002.

[4] A. Ankolekar et al., DAML-S: Semantic markup for web services, in: Proceedings of the International Semantic Web Working Symposium (SWWS), Stanford University, California, USA, July 30–August 1 2001.

[5] A. Ankolekar et al., DAML-S: Web service description for the semantic web, in: Proceedings of the First International Semantic Web Conference (ISWC), Sardinia, Italy, 9–12 June 2002.

[6] A. Ankolekar, F. Huch, K. Sycara, concurrent execution semantics for DAML-S with subtypes, in: Proceedings of the First International Semantic Web Conference (ISWC), Sardinia (Italy), June, 2002.

[7] L. Ardissono, A. Goy, G. Petrone, Enabling conversations with web services, in: Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems, Melbourne, Australia, 14–18 July 2003.

[8] A. Assaf et al, Web Service Choreography Interface 1.0, http://www.sun.com/software/xml/developers/wsci/wsci-spec-10.pdf, 2002.

[9] B.-M. Walter, J. Padget, M. Aird, Brokerage for mathematical services in MONET, in: AAMAS Workshop on Web Services and Agent-Based Engineering, 2003.

[10] S. Bechhofer, C. Goble, I. Horrocks, DAML+OIL is not enough, in: Proceedings of the Symposium on International Semantic Web Working (SWWS), Stanford University, California, USA, July 30–August 1 2001.

[11] V. Richard Benjamins, Web services solve problems, and problem-solving methods provide services, IEEE Intell. Syst. 18 (1) (2003) 76–77.

[12] T. Berners-Lee, J. Hendler, O. Lassila, The semantic web, Sci. Am. 284 (5) (2001) 34–43.

[13] J.J. Bryson, D.L. Martin, S.A. McIlraith, L.A. Stein, Toward behavioral intelligence in the semantic web, IEEE Comput. 35 (11) (2002) 48–54.

[14] P.A. Buhler, J.M. Vidal., Semantic web services as agent behaviors, in: B. Burg, J. Dale, T. Finin, H. Nakashima, L. Padgham, C. Sierra, S. Willmott (Eds.), Agentcities: Challenges in Open Agent Environments, Springer-Verlag, 2003.

[15] P.A. Buhler, J.M. Vidal, Towards the synthesis of web services and agent behaviors, in: B. Burg, J. Dale, T. Finin, H. Nakashima, L. Padgham, C. Sierra, S. Willmott (Eds.), Agentcities: Challenges in Open Agent Environments, Springer-Verlag, 2003.

[16] C. Bussler, A. Maedche, D. Fensel, Web services: quo vadis? IEEE Intell. Syst. 18 (1) (2003) 80–82.

[17] I. Constantinescu, S. Willmott, B. Faltings, Abstract behavior representations for service integration, in: B. Burg, J. Dale, T. Finin, H. Nakashima, L. Padgham, C. Sierra, S. Willmott (Eds.), Agentcities: Challenges in Open Agent Environments, Springer-Verlag, 2003.

[18] F. Curbera, M. Duftler, R. Khalaf, W. Nagy, N. Mukhi, S. Weerawarana , Unraveling the web services web, IEEE Internet Comput. 6 (2) (2002) 86–93.

[19] S. Decker, S. Melnik, F. V. Harmelen, D. Fensel, M. Klein, J. Broekstra, M. Erdmann, I. Horrocks, The semantic web: the roles of XML, and RDF, IEEE Internet Comput. 4 (5) (2000) 63–74.

[20] G. Denker et al, Accessing information and services on the DAML-enabled web, in: Proceedings of the Second International Workshop on the Semantic Web (SemWeb'), Workshop at WWW10, Hongkong, May 1, 2001.

[21] D. De Roure, N.R. Jennings, N.R. Shadbolt, The Semantic Grid: A Future e-Science Infrastructure, 2003, http://www.semanticgrid.org.

[22] D. Farber, Balancing security and liberty, IEEE Internet Comput. 5 (6) (2001) 96–96.

[23] J. Farrugia, Model-theoretic semantics for the web, in: Proceedings of the Twelfth International World Wide Web Conference (WWW), Budapest, Hungary, 20–24 May 2003.

[24] D. Fensel, C. Bussle, Web services modeling framework, Electron. Commerce Res. Appl. 1 (2002) 113–137.

[25] D. Fensel, C. Bussler, A. Maedche, Semantic web enabled web services, in: Proceedings of the first International Semantic Web Conference (ISWC), Sardinia, Italy, 9–12 June 2002.

[26] D. Florescu, A. Grünhagen, D. Kossmann, XL: An XML programming language for web service specification and composition, in: Proceedings of the Eleventh International World Wide Web Conference (WWW), Honolulu, Hawaii, USA, 7–11 May 2002.

[27] I. Foster, C. Kesselman, J. Nick, S. Tuecke, The physiology of the grid: an open grid services architecture for distributed systems integration. Open Grid Service Infrastructure WG, Global Grid Forum, June 22 2002. http://www.globus.org/research/papers/ogsa.pdf.

[28] I. Foster, C. Kesselman, S. Tuecke, The anatomy of the grid: enabling scalable virtual organizations, Int. J. Supercomputer Appl. 15 (3) (2001) 200–222.

[29] I. Foster, C. Kesselman (Eds.), The Grid: Blueprint for a New Computing Infrastructure, Morgan Kaufmann, San Francisco, 1999.

[30] I. Foster, The grid: a new infrastructure for 21st century science, Phys. Today 55 (2) (2002) 42–47.

[31] I. Foster, C. Kesselman, J. Nick, S. Tuecke, Grid services for distributed system integration, IEEE Comput. 35 (6) (2002) 37–46.

[32] S. Gaio, A. Lopes, L. Botelho, From DAML-S to Executable Code, in: B. Burg, J. Dale, T. Finin, H. Nakashima, L. Padgham, C. Sierra, S. Willmott (Eds.), Agentcities: Challenges in Open Agent Environments, Springer-Verlag, 2003.

[33] N. Gibbins, S. Harris, N. Shadbolt, Agent-based Semantic Web Services, WWW 2003, Budapest, Hungary, 20–24 May 2003.

[34] N. Gibbins, S. Harris, N. Shadbolt, Agent-based semantic web services, in: Proceedings of the Twelfth International World Wide Web Conference (WWW), Budapest, Hungary, 20–24 May 2003.

[35] C. Goad, Describing computation within RDF, in: Proceedings of the International Semantic Web Working Symposium (SWWS), Stanford University, California, USA, July 30–August 1 2001.

[36] G. Goth, Securing the internet against attack, IEEE Internet Comput. 7 (1) (2003) 8–10.

[37] B.N. Grosof, I. Horrocks, R. Volz, S. Decker, Description logic programs: combining logic programs with description logic, in: Proceedings of the Twelfth International World Wide Web Conference (WWW), Budapest, Hungary, 20–24 May 2003.

[38] R. Guha, R. McCool, E. Miller, Semantic search, in: Proceedings of the Twelfth International World Wide Web Conference (WWW), Budapest, Hungary, 20–24 May 2003.

[39] J. Hendler, Agents and the semantic web, IEEE Intell. Syst. 16 (2) (2001) 30–37.

[40] I. Horrocks, S. Tessaris, Querying the semantic web: a formal approach, in: Proceedings of the First International Semantic Web Conference (ISWC), Sardinia, Italy, 9–12 June 2002.

[41] I. Horrocks, P.F. Patel-Schneider, Three theses of representation in the semantic web, in: Proceedings of the Twelfth International World Wide Web Conference (WWW), Budapest, Hungary, 20–24 May 2003.

[42] A. Householder, K. Houle, C. Dougherty, Computer attack trends challenge Internet security, IEEE Comput. 35 (4) (2002) 5–7.

[43] M. Klein, Ontology versioning on the semantic web, in: Proceedings of the International Semantic Web Working Symposium (SWWS), Stanford University, California, USA, July 30–August 1 2001.

[44] M. Klein, A. Bernstein, Serching for services on the semantic web using process ontologies, in: Proceedings of the International Semantic Web Working Symposium (SWWS), Stanford University, California, USA, July 30–August 1 2001.

[45] H. Kuno, A. Sahai, my agent wants to talk to your service: personalizing web services through agents, in: B. Burg, J. Dale, T. Finin, H. Nakashima, L. Padgham, C. Sierra, S. Willmott (Eds.), Agentcities: Challenges in Open Agent Environments, Springer-Verlag, 2003.

[46] M.S. Lacher, S. Decker, On the integration of topicmaps data with RDF data, in: Proceedings of the International Semantic Web Working Symposium (SWWS), Stanford University, California, USA, July 30–August 1 2001.

[47] O. Lassila, F. van Harmelen, I. Horrocks, J. Hendler, D.L. McGuinness, The semantic Web and its languages, IEEE Intell. Syst. 15 (6) (2000) (November/December) 67–73.

[48] M. Laukkanen, H. Helin, Composing workflows of semantic web services, in: AAMAS Workshop on Web Services and Agent-Based Engineering, 2003.

[49] L. Juhnyoung, J. Yang, J. Chung, Winslow: A Business Process Management System with Web Services. IBM Research Report, November 2002.

[50] L. Frank, D. Roller, M. Schmidt, Web services and business process management, IBM Syst. J. 41 (2) (2002) 198–212.

[51] L. Li, I. Horrocks, A software framework for matchmaking based on semantic web technology, in: Proceedings of the Twelfth International World Wide Web Conference (WWW), Budapest, Hungary, 20–24 May 2003.

[52] G. McGraw, Managing software security risks, IEEE Comput. 35 (4) (2002) 99–101.

[53] D.J. Mandell, S.A. McIlraith, A bottom-up approach to automating web service discovery, customization, and semantic translation, in: Proceedings of the Twelfth International World Wide Web Conference Workshop on E-Services and the Semantic Web (ESSW), Budapest, 2003.

[54] S. McIlraith, T.C. Son, H. Zeng, Semantic web services, IEEE Intell. Syst. 16 (2) (2001) 46–53.

[55] S. McIlraith, D. Martin, Bringing semantics to web services, IEEE Intell. Syst. 18 (1) (2003) 90–93.

[56] M. Naedele, Standards for XML and web services security, IEEE Comput. 36 (4) (2003) 96–98.

[57] S. Narayanan, S. McIlraith, Simulation, verification and automated composition of web services, in: Proceedings of the Eleventh International World Wide Web Conference (WWW), Honolulu, Hawaii, USA, 7–11 May 2002.

[58] J.Z. Pan, I. Horrocks, Metamodeling architecture of web ontology languages, in: Proceedings of the International Semantic Web Working Symposium (SWWS), Stanford University, California, USA, July 30–August 1 2001.

[59] M. Paolucci, T. Kawamura, T.R. Payne, K. Sycara, Semantic matching of web services capabilities, in: Proceedings of the First International Semantic Web Conference (ISWC), Sardinia, Italy, 9–12 June 2002.

[60] P.F. Patel-Schneider, D. Fensel, Layering the semantic web: problems and directions, in: Proceedings of the First International Semantic Web Conference (ISWC), Sardinia, Italy, 9–12 June 2002.

[61] J. Peer, Bringing together semantic web and web services, in: Proceedings of the First International Semantic Web Conference (ISWC), Sardinia, Italy, 9–12 June 2002.

[62] W.T. Polk, N. E Hastings, A. Malpani, Public key infrastructures that satisfy security goals, IEEE Internet Comput. 7 (4) (2003) 60–67.
[63] D. Richards, S. van Splunter, F.M.T. Brazier, M. Sabou, Composing web services using an agent factory, in: AAMAS Workshop on Web Services and Agent-Based Engineering, 2003.
[64] M. Sabou, D. Richards, S. van Splunter, An experience report on using DAML-S, in: Proceedings of the Twelfth International World Wide Web Conference Workshop on E-Services and the Semantic Web (ESSW), Budapest, 2003.
[65] D. Scott, R. Sharp, Abstracting application-level web security, in: Proceedings of the Eleventh International World Wide Web ConferencE (WWW), Honolulu, Hawaii, USA, 7–11 May 2002.
[66] D. Scott, R. Sharp, Developing secure web applications, IEEE Internet Comput. 6 (6) (2002) 38–45.
[67] M. Sheshagiri, M. desJardins, T. Finin, A planner for composing services described in DAML-S, in AAMAS Workshop on Web Services and Agent-Based Engineering, 2003.
[68] B. Siddiqui, Web Services Security, Part 1, http://webservices.xml.com/pub/a/ws/2003/03/04/security.html.
[69] Bilal Siddiqui, Web Services Security, Part 2, http://webservices.xml.com/pub/a/ws/2003/04/01/security.html.
[70] B. Siddiqui, Web Services Security, Part 3 http://webservices.xml.com/pub/a/ws/2003/05/13/security.html.
[71] J. Su, R. Hull, T. Bultan, X. Fu, Conversation specification: a new approach to design and analysis of E-service composition, in: Proceedings of the Twelfth International World Wide Web Conference (WWW), Budapest, Hungary, 20–24 May 2003.
[72] S. Thatte, XLANG-Web Services for Business Process Design, http://www.gotdotnet.com/team/xml_wsspecs/xlang-c/default.htm.
[73] S. Tuecke, K. Czajkowski, I. Foster, J. Frey, S. Graham, C. Kesselman et al. (Eds.), Open Grid Services Infrastructure (OGSI), 2003, http://www-unix.globus.org/toolkit/draft-ggf-ogsi-gridservice-33_2003-06-27.pdf.
[74] W. van der Aalst, Don't go with the flow: web services composition standards exposed, IEEE Intell. Syst., January/February (2003) 72–76.
[75] E. Wales, Web services security, Comput. Fraud Security 18 (1) (2003) 15–17.
[76] W. Sanjiva, C. Francisco, Business processes: understanding BPEL4WS, Part 1. IBM developer Works, August 2002.
[77] L. Zeng, B. Benatallah ,M. Dumas, Quality driven web services composition, in: Proceedings of the Twelfth International World Wide Web Conference (WWW), Budapest, Hungary, 20–24 May 2003.
[78] http://www.w3.org/TR/2003/REC-soap12-part0-20030624/-L1161.
[79] http://www.w3.org/TR/2003/WD-wsdl12-20030611/.
[80] http://uddi.org/pubs/uddi-v3.00-published-20020719.htm.
[81] http://www.omg.org/technology/documents/spec_catalog.htm.
[82] http://www-3.ibm.com/software/solutions/webservices/pdf/WSFL.pdf.
[83] http://www-106.ibm.com/developerworks/webservices/library/ws-bpel/.
[84] http://www.w3.org/Signature/.
[85] http://www.w3.org/Encryption/2001/.
[86] http://www.oasis-open.org/committees/wss/.
[87] http://www.oasis-open.org/committees/security/.
[88] http://www.oasis-open.org/committees/xacml/.
[89] http://www.w3.org/TR/wscl10/.
[90] http://www.commerce.net/.
[91] http://www.bpmi.org/specifications.esp.
[92] http://www.w3c.org/RDF/.
[93] http://www.w3.org/TR/2003/WD-rdf-schema-20030123/.
[94] http://www.daml.org/2001/03/daml+oil-index.html.
[95] http://www.w3c.org/TR/owl-features/.

Hongbing Wang is an associate professor at the Department of Computer Science and Engineering, Southeast University, PR China. He is a PhD candidate in computer software from Nanjing University, PR China. His current research interests include Semantic Web, Web services, electronic commerce, grid computing, and software engineering. He has been involved in two projects on Semantic Web, which are supported by NSFC and JSNSF respectively. He has published over 20 research papers.

Joshua Zhexue Huang received his PhD from the Royal Institute of Technology, Stockholm, Sweden in 1993 and is now the assistant director of The E-Business Technology Institute (ETI) and honorary professor of the Department of mathematics, The University of Hong Kong. He was a senior consultant at The Management Information Principles Australia from 1998 to 2000 and a research scientist at The Mathematical and Information Sciences Division (CMIS) in The Commonwealth Science and Industry Research Organization (CSIRO) of Australia from 1994 to 1998. At CSIRO, he developed two clustering algorithms k-modes and k-prototypes, which have been frequently cited by peers and used in research organizations and companies around the world. His research interests include data mining, clustering algorithms, knowledge-based business intelligence systems, topic-driven Web crawling, text mining, and Grid computing.

Yuzhong Qu is a professor at Department of computer science and Engineering,Southeast University, PR China. He received a master degree in mathematics from Fudan University, PR China, in 1988, and a PhD in Computer Software from Nanjing University, PR China, in 1995. His current research interests include Semantic Web, grid computing and software engineering. He has published two books and more than 30 papers. Currently, he is the principle investigator of two projects on Semantic Web, which are supported by NSFC and JSNSF respectively.

Junyuan Xie is a professor at the Department of computer science and technology, Nanjing University, PR China. His current research interests include artificial intelligence, information security, electronic commerce, and Web services. He has been a principle investigator on a few more china funded researches projects. He has published over 40 research papers.