Serendipitous Interoperability

Ora Lassila*

Abstract

This article will discuss issues of interoperability of Web Services. Semantic Web technologies, when applied to current Web Service architectures, will enable true automation and interoperation, and will "futureproof" systems unlike *a priori* standardization approaches. We will extend the Semantic Web approach to Web Services to Ubiquitous Computing: By abstracting device functionality via software agents, and using Semantic Web technologies to describe agent services and capabilities, we can achieve a rich and deep form of service discovery. By discovering partially matching services, and piecing these together into "virtual value chains", we can automatically form device coalitions which operate within a dynamically changing environment.

1 Introduction

"Web Services" – functionality that can be invoked remotely over the Web – is a strong recent trend in the development of World Wide Web -based systems. Several industry standards have emerged in this area, including SOAP [2], an invocation protocol, and WSDL [7], a formalism for describing the interfaces of Web Services. In addition, mechanisms have been proposed to facilitate the discovery of Web Services – one of these is UDDI [19]. All these specifications are being offered with the promise of greater opportunity for automation of tasks and improved interoperability of information systems.

In this paper we will argue that albeit we see progress in the right direction, these attempts will fall short of the goals of improved automation and interoperability, because they are based on heavy *a priori* standardization and they ultimately retain humans in the loop. The maintenance and management of all the emerging vocabularies will result in a phenomenon we can only compare to the biblical story of the "Tower of Babble" [22]. We believe that our true goal should be "serendipitous interoperability", the ability of software systems to discover and utilize services they have not seen before, and that were not considered when the systems were designed. To realize this, qualitatively stronger means of representing the service semantics are required, enabling fully automated discovery and invocation, and complete removal of unnecessary interaction with human users.

*Nokia Research Center, 5 Wayside Road, Burlington MA 01803, USA

The Semantic Web [5] offers means of advancing beyond the current proposed architectures for Web Services. Characterized by the exposure of declarative formal semantics of information and services, the Semantic Web allows information systems to *reason* about data sources and the functionality of other systems, and consequently allows them to better take advantage of these. The Semantic Web will also enable – finally – the emergence of *intelligent agents*, systems based on autonomously operating goal-oriented software entities.

In the context of Web Services, the application of the Semantic Web to representing information and its semantics will enable the following:

- Description of semantics of services to allow their automatic discovery, even if the services offered only partially match the needs of the requester.
- Automatic composition of multiple services – possibly partially matching ones – into a "super-service" satisfying the needs of the requester (whether the requester be a human or an artificial agent).

2 About Representation and Ontologies

The *ontological approach* characteristic of the Semantic Web is predicated on the existence and use of *ontologies*: documents or files that formally define the relationships between terms for any particular domain of discourse – a widely cited definition of an ontology is Gruber's "a specification of a conceptualization" [8], and in that sense we depart from the abstract philosophical notion of ontology defined as "a branch of metaphysics concerned with the nature and relations of being" [16].

People, as well as artificial agents, typically have a notion or conceptualization of the meaning of terms. Just as the specification inputs and outputs of a software program could be used as a specification of the program itself, ontologies can be used to provide a concrete specification of term names and meanings. If we consider ontologies as specifications of the conceptualizations of terms, there is much room for variation, and the spectrum of Web ontologies typically range from simple controlled vocabularies through informal concept hierarchies to something where arbitrarily complex logical relationships can be specified between defined concepts. In practical terms, we would expect the following properties to hold in order to consider something an ontology [14]:

1. Finite controlled (extensible) vocabulary
2. Unambiguous interpretation of classes and term relationships
3. Strict hierarchical subclass relationships between classes

The following properties for ontologies are typical but not mandatory:

4. Property specification on a per-class basis
5. Inclusion of individuals (i.e., instances) in the ontology
6. Value restriction specification on a per-class basis

2.1 Representing Semantics of Web Services

In the context and environment of the World Wide Web, "Web-friendly" knowledge representation formalisms DAML+OIL [9, 23] and its foundation, W3C's RDF [12, 15, 4] will be used[1] to describe services for the purposes of discovery. Just as the success of the deployment of the Semantic Web will largely depend on whether useful ontologies will emerge, so will discovery services benefit from mechanisms that allow shared agreements about vocabularies for knowledge representation. Sharing vocabularies allows *automated* interoperability; given a base ontology shared by agents, each agent can extend this ontology while achieving partial understanding of the others; this is analogous to OOP systems, where a base class defines "common" functionality.

One attempt to represent service semantics is DAML-S [1, 6], a Web Service ontology expressed in DAML+OIL. It gives Web Service providers a core set of markup language constructs for describing the properties and capabilities of their Web Services in an unambiguous, computer-interpretable form. DAML-S markup of Web Services will facilitate the automation of the following Web Service tasks:

1. **Automatic discovery** involves the automatic location of services that provide a particular function and adhere to requested constraints.

2. **Automatic invocation** involves the execution of a discovered service by a computer program or agent. Execution of a service can be thought of as a collection of function calls. DAML-S markup of Web Services provides a declarative, computer-interpretable interface for executing these function calls. A software agent should be able to interpret the markup to understand what input is necessary to the service call, what information will be returned, and how to execute the service automatically.

3. **Automatic composition and interoperation** involves the automatic construction of a plan to use a number of services to perform some task, given a high-level description of an objective. With DAML-S markup of Web Services, the information necessary to select and compose services will be encoded at the service Web sites. Software can be written to manipulate these representations, together with a specification of the objectives of the task, to achieve the task automatically.

4. **Automatic execution monitoring**: Individual services and, especially, compositions of services, will often require some time to execute completely. Users (human or artificial ones) may want to know during this period what the status of their request is. Their plans may also have changed requiring alterations in the actions the service provider takes.

In comparison with *service profiles* – the DAML-S characterization of services – the other industry standards are limited, first and foremost, in that they cannot

[1] DAML+OIL will be succeeded by the emerging Web ontology language OWL [26].

express logical statements. Input and output types are supported to varying extents. From the DAML-S standpoint, services can be simple (or primitive); they invoke only a single Web-accessible computer program that does not rely upon another Web Service, and there is no ongoing interaction between the user and the service, beyond a simple response. Alternately, services can be complex, composed of multiple primitive services, often requiring an interaction or dialogue between the consumer and the provider of the services, so that choices can be made or information provided conditionally. DAML-S is meant to support both categories of services, but complex services have provided the primary motivation for the features of the language.

DAML-S provides an upper ontology for services, including the properties normally associated with all kinds of services. The upper ontology does not address what the particular subclasses of the base service class should be, or even the conceptual basis for structuring this taxonomy – this may take place according to functional and domain differences, or market needs. A service profile provides a high-level description of a service and its provider, and is used as a request, or as an advertisement, within a service discovery process.

Service profiles consist of three types of information: a human readable description of the service, a specification of the functionalities that are provided by the service (represented as a transformation from the inputs required by the service to the outputs produced), as well as a host of functional attributes which provide additional information and requirements about the service that assist when reasoning about several services with similar capabilities (these include guarantees of response time or accuracy, as well as the cost of the service). Furthermore, a more detailed perspective on services is to view them as processes. DAML-S representation of processes draws upon established work in a variety of fields, such as automated planning and workflow automation, and will support the representational needs of a very broad array of services on the Web.

3 Semantic Web Meets Ubiquitous Computing

Ubiquitous Computing is an emerging paradigm of personal computing, characterized by the shift from dedicated computing machinery (that requires the user's attention – e.g., PCs) to pervasive computing capabilities embedded in our everyday environments [20, 21]. Characteristic to Ubiquitous Computing are small, handheld, wireless computing devices. The pervasiveness and the wireless nature of devices require network architectures to support automatic, *ad hoc* configuration [10].

A key functionality of true *ad hoc* networks is *service discovery*, by which functions offered by various devices on the network can be described, advertised, and discovered by others. Several frameworks and formalisms for this type of service discovery and capability description have already emerged – examples include Sun's "Jini" and Microsoft's Universal Plug and Play (UPnP) [24, 25, 17] as means of describing services and invoking them, as well as World Wide Web Consortium's (W3C) Composite Capability/Preference Profile (CC/PP) [11]

as a means of describing device characteristics. The current service discovery mechanisms are based on *ad hoc* representation schemes and rely heavily on standardization (i.e., on *a priori* identification of all those things one would want to communicate or discuss). "Jini" also relies on the Java object system and instance serialization, and UPnP uses its own flavor of HTTP.

Our goal is to represent the functionality of Ubiquitous Computing devices as services in a multiagent framework, and apply the Semantic Web -based Web Service techniques to facilitate the interoperation of these devices – more specifically, we aim at enabling the discovery and utilization of services by other agents without human guidance or intervention, thus enabling the automatic formation of device coalitions through this mechanism. We call these devices, capable of semantic discovery and coalition formation, *semantic gadgets* [13, 5].

3.1 Semantic Discovery

Semantic gadget technology begins with the discovery of functionality and services. As mentioned before, a number of mechanisms for low-level service discovery have emerged; these mechanisms attack the problem at a syntactic level, and rely heavily on the standardization of a predetermined set of functionality descriptions. Standardization, unfortunately, can only take us halfway toward our goal of intelligent automated behavior *vis-a-vis* discovery, as our ability to anticipate all possible future needs is limited. By elevating the mechanisms of service discovery to a "semantic" level, a more sophisticated description of functionality is possible, and the shared understanding between the consumer and the provider can be reached via the exchange of ontologies which provide the necessary vocabulary for a dialogue.

Semantic discovery mechanisms will undoubtedly be layered on top of existing, lower-level services. These services involve *ad hoc* networking technologies and other mechanisms that are beyond the scope of this article. In our approach, physical devices and their functionality will be abstracted as software agents. These agents will advertise services and/or will query for services they need. Both the advertisements and the queries will be abstract descriptions of functionality – in fact, there is no difference between the two, as Sycara has pointed out [18]. Through a matchmaking process (either by the provider, by the consumer, or by a third party "matchmaker") we are able to associate compatible advertisements with queries. The match might be perfect – in which case a service will exactly meet the need of the consumer – or partial – in which case the consumer might have to combine the service with some additional functionality (it can do this by itself, or – as we will demonstrate later – continue to discover the "missing pieces" and finally compose a service that meets its need).

The Semantic Web plays two key roles in the discovery process. First, Semantic Web techniques provide a rich mechanism for describing functionality: ontologies will describe the concepts and vocabulary needed to discuss functionality and services. Second, the Semantic Web provides a unifying layer of naming and distribution, an addressing mechanism that can encompass virtual and physical entities, and a way for various pieces of the "puzzle" to reside on

various servers, devices etc. For example, a device description can refer to an ontology elsewhere, which in turn can be a specialization or extension of another ontology, again somewhere else. The polymorphic nature of ontology extension will allow broader interoperation through partial understanding and agreement.

3.2 Services and Contracting

Once the services we want to use have been discovered and identified, there are several rather "bureaucratic" issues to be dealt with, related to "contracting" the use of the services. These include (but are not limited to):

- Assuring security, privacy, and trust
- Compensating the service provider
- Determining execution locus

The Semantic Web promises to be useful when it comes to matters of security, privacy and trust, as these are largely issues of representation. Generally, the Semantic Web rests heavily on a framework of trust partially constructed using digital signatures and other types of cryptographic certificates. Any agent can expose its reasoning about trust using the same mechanisms by which everything else is represented on the Semantic Web. These representations of reasoning – we might call them "proofs" – can themselves be exchanged between agents as persuasive communication. We anticipate the emergence of services specific to assisting agents with their security, privacy, and trust issues (e.g., there might be a service that rates the "reputation" of other services: reliability, trustworthiness, or some other relevant metric). Again, these services themselves can be discovered and their use contracted.

Given a functional security and trust framework, we can introduce the notion of payments. This is required for the purpose of compensating providers for services rendered. Generally, we anticipate third-party services (which, again, are discoverable) to facilitate payments or other types of compensation. We are not trying to imply that everything on the Semantic Web will cost something, but some services will emerge that are not free. Advertising (of products and services for humans this time) will no longer be a viable revenue model once automated agents take care of a large part of the information exchange, reasoning and service utilization. Furthermore, some type of compensation mechanism might be used to provide stability to the operation of a system of self-interested agents [3].

3.3 Composition of Services

The discovery of services based on some description of a requirement of new functionality might result in a partial match [18]. In this case the requester can attempt to provide the missing parts itself or continue the discovery process and identify other services. They can then be pieced together to form an aggregate service that fulfills the original requirement.

Composition of exact required functionality from partially matching services should be viewed as a process of goal-driven assembly, achievable by the use of automated planning and/or configuration techniques. In the context of Ubiquitous Computing and semantic gadgets, contracting the use of services should always be viewed as a goal-driven activity because of the "volatile" nature of Ubiquitous Computing environments (not only can any device or service fail or be removed from the environment at any time, but new ones can be added to it; opportunistic exploitation of services might thus be beneficial).

The goal-driven approach takes information system interoperability beyond what mere standardization or simple interface sharing enables, since it is based on "deeper" descriptions of service functionality and can be performed *ad hoc* and on demand. In fact, the dynamically composed aggregate services are the Semantic Web's virtual equivalent of "real-life" value chains: large quantities of information (i.e., "raw material") may be obtained, and at each step the value of information increases and its volume decreases [5].

Given that the services discovered represent actual physical functionality of devices, aggregate services could be seen as device coalitions. Not only can individual devices extend their functionality, but the device coalitions effectively form task-specific "super-devices".

4 Conclusions

We have presented how Semantic Web technologies can be used in the context of Web Services to ensure "serendipitous" forms of interoperation. The Semantic Web encompasses efforts to populate the Web with content that has formal semantics; thus the Semantic Web will enable automated agents to reason about Web content, and produce an intelligent response to unforeseen situations. Our vision is to overlay the Semantic Web on a Ubiquitous Computing environment, making it possible to represent and interlink devices, their capabilities, and the functionality they offer. By abstracting device functionality as software agents, and then using Semantic Web technologies to describe agent services, we can build a "semantic" discovery service capable of function beyond *a priori* standardization. Through the composition of discovered, partially matching services into virtual value chains we are able to form device coalitions which opportunistically exploit a dynamically changing Ubiquitous Computing environment.

The interoperability of physical devices is an objective worth pursuing: it is unlikely that a single company will be engaged in the manufacture of all the different types of devices that will make up the "web of semantic gadgets" (for example, Nokia manufactures wireless communication devices but it does not make thermostats; yet consumers might benefit from wireless devices that communicates with thermostats). In fact, applying Semantic Web technologies to Web Services in the Ubiquitous Computing context may have more compelling business models than the application of the same technology elsewhere, since the device manufacturers are not primarily in the Semantic Web business.

Acknowledgements

The author owes deep gratitude to the following individuals for discussions related to this article and its topics: Mark Adler (Nokia Research Center), Michael Mahan (Nokia Research Center), Deborah McGuinness (Stanford University), Katia Sycara (Carnegie Mellon University), the entire DAML-S coalition, and the ATMOS and PHEAR project teams of the Agent Technology Group at the Nokia Research Center. The research described in this paper was supported in part by Nokia Research Center, Nokia Ventures Organization, Nokia Mobile Phones, and Nokia Venture Partners.

References

[1] Ankolenkar, A., Burstein, M, Hobbs, J.R., Lassila, O., Martin, D., McIllraith, S.A., Narayanan, S., Paolucci, M., Payne, T., Son, T.C., Sycara, K. and Zeng, H.: DAML-S: A Semantic Markup Language for Web Services, Proceedings of the First Semantic Web Working Symposium (SWWS'01), Stanford University, July 2001

[2] Box, D., Ehnebuske, D., Kakivaya, G., Layman, A., Mendelsohn, N., Nielsen, H.F., Thatte, S. and Winer, D.: Simple Object Access Protocol (SOAP) 1.1, W3C Note, World Wide Web Consortium, Cambridge (MA), May 2000; available as http://www.w3.org/TR/2000/NOTE-SOAP-20000508

[3] Brainov, S. and Sandholm, T.: Power, Dependence and Stability in Multiagent Plans, Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99), Orlando (FL), AAAI Press, 1999

[4] Brickley, D. and Guha, R.V.: Resource Description Framework (RDF) Schema Specification 1.0, W3C Candidate Recommendation 2000-03-27

[5] Berners-Lee, T., Hendler, J. and Lassila, O.: The Semantic Web, Scientific American 284(5):34–43 (May 2001)

[6] Burstein, M., Hobbs, J., Lassila, O., Martin, D., McIllraith, S., Narayanan, S., Paolucci, M., Payne, T., Sycara, K: DAML-S 0.6 Draft Release, draft document of the DARPA Agent Markup Language Program, 2001; available as http://www.daml.org/services/daml-s/2001/06/

[7] Christensen, E., Curbera, F., Meredith, G. and Weerawarana, S.: Web Services Description Language (WSDL) 1.1, W3C Note, World Wide Web Consortium, Cambridge (MA), March 2001; available as http://www.w3.org/TR/2001/NOTE-wsdl-20010315

[8] Gruber, T.R.: A translation approach to portable ontologies, Knowledge Acquisition 5(2): 199–220, 1993

[9] Hendler, J. and McGuinness, D.L.: The DARPA Agent Markup Language, IEEE Intelligent Systems 15(6): 67–73 (November/December 2000)

[10] Huitema, C.: Plug and Play, in IPv6: the New Internet Protocol, Prentice-Hall, Upper Saddle River (NJ), 1998

[11] Klyne, G., Reynolds, F., Woodrow, C. and Ohto, H.: Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies, W3C Working Draft 15 March 2001, World Wide Web Consortium, Cambridge (MA); available as http://www.w3.org/TR/CCPP-struct-vocab/

[12] Lassila, O.: Web Metadata: A Matter of Semantics, IEEE Internet Computing 2(4): 30–37 (1998)

[13] Lassila, O. and Adler, M.: Semantic Gadgets – Ubiquitous Computing Meets the Semantic Web, in: Fensel, D. et al (eds.): Spinning the Semantic Web, MIT Press, Boston (MA); to appear

[14] Lassila, O. and McGuinness, D.L.: The Role of Frame-Based Representation on the Semantic Web, Electronic Transactions on AI (to appear); available as a Stanford KSL technical report KSL-01-02

[15] Lassila, O. and Swick, R.R.: Resource Description Framework (RDF) Model and Syntax Specification, W3C Recommendation 1999-02-22

[16] Merriam-Webster's Collegiate Dictionary, Merriam-Webster, 1998

[17] Richard, G.G.: Service Advertisement and Discovery: Enabling Universal Device Cooperation, IEEE Internet Computing 4(5): 18–26 (September/October 2000)

[18] Sycara, K., Klusch, M., Widoff, J. and Lu, J.: Dynamic Service Matchmaking Among Agents in Open Information Environments, ACM SIGMOD Record 28(1): 47–53 (March 1999)

[19] UDDI Technical White Paper, UDDI.org, September 2000; available from http://www.uddi.org/whitepapers.html

[20] Weiser, M.: The Computer for the Twenty-First Century, Scientific American 265(3): 94–104 (September 1991)

[21] Weiser, M.: Some Computer Science Problems in Ubiquitous Computing, Communications of the ACM 36(7): 75–84 (July 1993)

[22] "The Dispersion of the Nations at Babel", Genesis 11:1–9

[23] http://www.daml.org/2001/03/daml+oil-index.html

[24] http://www.sun.com/jini/

[25] http://www.upnp.org/

[26] http://www.w3.org/2001/sw/WebOnt/