Reputation and Endorsement for Web Services

E. Michael Maximilien
North Carolina State University and IBM
maxim@us.ibm.com
and
Munindar P. Singh
North Carolina State University
singh@ncsu.edu

The web services set of standards promise the dynamic creation of loosely coupled systems, such as those that are required for e-commerce applications. However, current approaches for web services lack key functionality, especially to locate, select, and bind services meeting certain criteria of quality. We propose an approach wherein software agents assist in this task by disseminating reputations and endorsements through a specialized agency, which augments the capabilities of current standards.

General Terms: Security, Theory, Economics

Additional Key Words and Phrases: E-commerce, Web services, Software agents

1. WEB SERVICES

The WWW is transitioning from a network of information primarily suited to human access to a substrate for connection among arbitrary devices. In particular, for e-commerce applications, the WWW will allow seamless application to application (A2A) interoperation, regardless of differences in languages and platforms. Two basic technologies are enabling this new WWW. One, XML enables a common language for messages between software components. Two, web services standards such as WSDL [W3C 2001b], UDDI [UDDI 2000] and SOAP [W3C 2001a] allow dynamic, description, discovery, and communication between software components.

The above technologies, however, provide only the low-level plumbing needed to achieve dynamic A2A interoperation. In addition, we need a systematic approach through which clients can locate the right web services and bind dynamically to them. This binding would depend on client-specific criteria that may be set at run time or at design time. Briefly, our solution is to apply agents to develop a special reputation and endorsement service that helps improve the binding of service clients to appropriate web services.

Current web technologies, although powerful, have a severe limitation in that they force a fragmentation of potentially related activities. For example, Amazon's e-marketplace can interoperate neither with Yahoo's e-marketplace nor with eBay's auction services. The emerging web services standards are intended to facilitate global e-commerce through A2A transactions. These standards are based on XML and cover important aspects of building dynamically composed systems of services.

Following [IBM 2001], we "define a web service as an interface that describes a collection of operations that are network accessible through standardized XML messaging." To facilitate interoperability by making web services neutral as to vendor and platform, a service is described using an XML application standard called WSDL (Web Service Description Language) [W3C 2001b]. Service descriptions are cataloged using a standard registry definition called UDDI (Universal Discovery, Description and Integration) [UDDI 2000]. SOAP (Simple Object Access Protocol) is the most popular underlying messaging protocol [W3C 2001a]. Although these standards and assorted extensions are intended to realize the promises of web services [Colan 2001; Kirtland 2001], they do not provide a complete picture. We give a quick overview of typical client use cases with a view to pointing out potential problems with current approaches. Using current service standards, developers will typically have four main usage scenarios:

(1) Use as software components to implement portions of some application functionality. For instance, a personal money management application such as Quicken could integrate with stock brokerage firms such as Fidelity Investments to display stock and mutual fund information. This is now possible, however, since each investment firm expose their own private protocol, the application needs to understand each one.
(2) Build web interface front-ends to their applications. Again, as in the previous case, the interface may access the published service for their realization. The great benefit here is that since these functionalities are accessed via a published interface, the implementations can vary, potentially creating a market for common functionality or services.
(3) Build bridges between published services and other platforms or legacy systems. The client application cannot directly access the published service due to some constraint, but it can use a bridge that acts as an adaptor for the clients.
(4) Compose services to expose coarser-grained web software components.

In each of these scenarios, the clients will be either invokers or publishers of services. UDDI registries will be available on the Internet. Clients access these registries, querying for the specific services required to accomplish their tasks. Once actual binding templates for the WSDL interface are found via the *tModel* or service template model in the UDDI registry, the client will bind and submit SOAP calls formatted according to the WSDL operations formats. Figure 1 illustrates this generic distributed service architecture.

2. LIMITATIONS OF CURRENT APPROACHES

The key limitation of current approaches is that they lack fully dynamic discovery and binding processes. We need an architecture that allows services clients to automatically bind with the service that they need with minimal interactions from either the design and implementers or the end users. In addition, this lack of dynamism also presents itself in various other areas of service development, deployment, and usage.

—Service clients have no automatic memory of the binding connections. These clients cannot learn from their past interactions to improve future decisions.

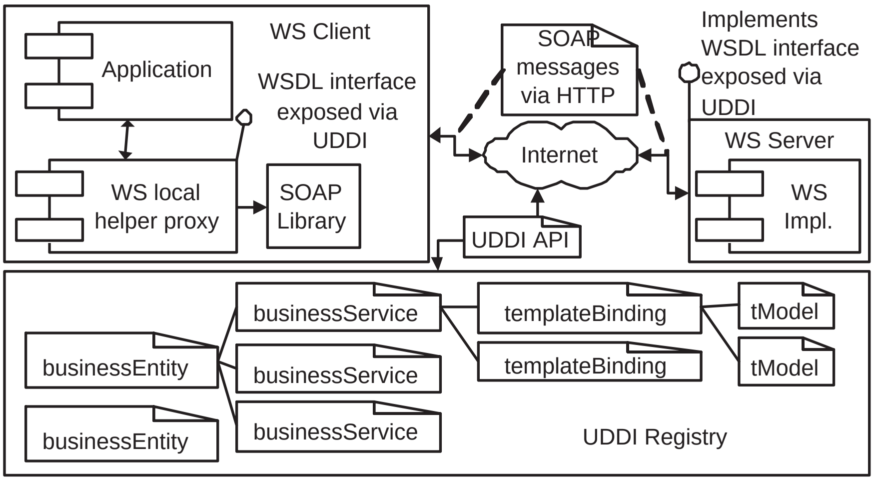

Fig. 1. Typical architecture for a service client.

—Service clients do not learn from interactions and cannot improve their selections from among the available implementations of a given service.

—To enable a true e-marketplace for services, there is a need for service clients to share their knowledge so as to help each other improve the quality of their decisions.

—Finally, an e-marketplace for software components will require ratings to attract new market opportunities and competition. It will support dynamic real-time advertising, which goes beyond what is now available through static registries such as UDDI.

The above problems illustrate the shortcomings of current service standards with respect to dealing with a large-scale, open, and ever-changing computational environment. Although much data handling and underlying protocols are automated through current standards, key aspects that support flexible decision-making with respect to the composition and selection of services are still treated in an ad hoc manner. Making these aspects systematic is the main motivation for our approach.

3. REPUTATION AND ENDORSEMENT AGENCY

As observed above, current approaches to web services lack some key aspects of dynamic binding of services. Such flexibility being critical for dealing with large-scale service environments, we look to an approach based on software agents working in a multiagent system. We propose a reputation and endorsement system (RES) agency to facilitate the tasks of the software agents.

The main idea is quite simple. We propose the addition of a Web Service Agent Proxy (WSAP) to access each service. This proxy facilitates the use of the service and collaborates with other agents to select good services. For our purposes, an agent is a software component that automates some tasks for its client. Agents communicate with other agents, accept requests from their users, and are typically autonomous. A WSAP is an agent that acts as a proxy for clients of web services. The WSAP is also knowledgeable about the various service standards.

The WSAP is configured by the client with information on the service it proxies. All service activities of the client—including requests and responses, and communications with UDDI registries and bindings—occur via the WSAP. The WSAP can monitor the activities and usages of the service by the client and help in future usages.

Service qualities that are subject to human perceptions are clearly harder, but they can be accommodated by our architecture. For such cases, the application should be designed so that it is possible to receive feedback from the human user after usage of the service. In this case, the WSAP will exploit this human feedback to learn the user's preferences.

A WSAP can also offer and obtain advice from other WSAPs. Specifically, a WSAP can also directly communicate with other WSAPs that participate in the RES agencies thereby allowing it to better realize the choices of its client.

3.1 Reputation and Endorsement

A distributed trust system consists of a set of *principals*, i.e., the parties involved either as service provider or requester. The principals engage with each other over a set of services. A *rating* of a service is a vector of attribute values. The *reputation* of a service aggregates the ratings of that service by other principals. An *endorsement* of a service by a principal is modeled as a Boolean scalar and a time limit on the validity of the endorsement.

Some existing systems that use a similar trust model are discussed in [Oram 2001]. For instance, the Pretty Good Privacy (PGP) "web of trust" system handles automated reputation collection to assist in verifying that public keys of principals are valid. The popular news site Slashdot, also discussed in [Oram 2001], uses a manual reputation gathering system to screen out and rank news posting thereby allowing clients to filter out certain posting that are judged to be of low value.

Reputation mechanisms are used in e-commerce web sites such as Amazon, eBay, and OnSale as a means of keeping track of ratings [Zacharia and Maes 2000]. A buyer or seller can capture his experience in specific transactions by rating the other party numerically or writing a text note about it. The marketplace makes this information available to other users so they can factor it in when deciding with whom to deal.

Like in the above marketplaces, we model reputation in terms of attributes reflecting a user's experiences with a given service. However, here the reputation data need not originate with humans, but may be provided by agents. WSAPs collect information on the services they bind to and convey this information to the RES agency, where it can be aggregated. The sequence diagram in Figure 2 illustrates how the WSAP is used in the RES agency. In addition, as stated above, some service qualities might require human intervention in order to capture the reputation attributes. Obviously, the human-oriented attributes should be limited to those that are essential to minimize the disruption to the user. However, our system must allow their existence since they are important and some qualities can only be determined based on the user's assessments.

An application contains one WSAP for each service that it uses. Each WSAP is configured to select the default implementation location for its service, as well as the default UDDI server and list of agencies that can be used to find alternative implementations. The required minimal reputation for each service is also listed in the configuration. Once activated, the application can make use of the WSAP as if it was a regular service. All interactions between the application and a service are monitored and added to the WSAP knowledge base, to be latter aggregated and forwarded to the various agencies that the agent participates in. Connection with the agencies can be done either automatically after a number of usage or certain amount of time or if the user decides to do so explicitly.

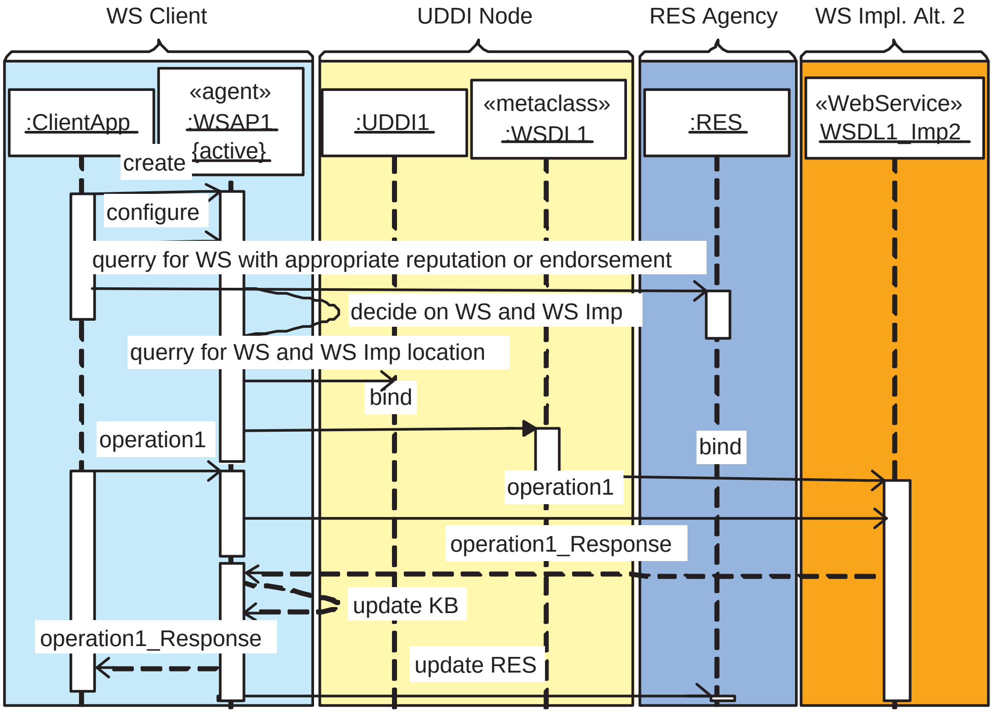

Fig. 2. Typical interaction sequence of WSAP and RES agency.

To get a better idea of the reputation information collected by the agents and the endorsement that a service can acquire, we will now give a sketch of the generic model of reputation data and endorsement. All reputation data, whether automatically collected by the WSAP or assisted by some human user, follows this model. The endorsements need to be communicated to the RES when new services are created and added to the RES.

3.2 Model of Reputation and Endorsement

The RES agency collects reputation and endorsement information for each service that it tracks. Endorsements are modeled as triples, each relating two provider principals and a time-limit of validity. Each principal is modeled using its public and private keys and certificate authority (CA). The time-limit is the length of time for which the endorsement is valid. A given service may be endorsed by multiple endorsers.

$$P_i = \langle K_{i\mathit{private}}, K_{i\mathit{public}}, CA_i \rangle$$
$$E_{WS_k} = \langle P_i, P_k, t \rangle$$

A reputation is modeled as a vector of attributes. The attributes are chosen by the WSAP from among a collection of reputation attributes that the RES makes available. By publishing a set of attributes, the RES ensures that the reputation data can be easily queried. However, doing so constrains the system as adding attributes requires reconfiguration of the agents if they are to collect information on these new attributes. Attributes naturally fall into some major categories, e.g., performance, compliance, reliability, and so on.

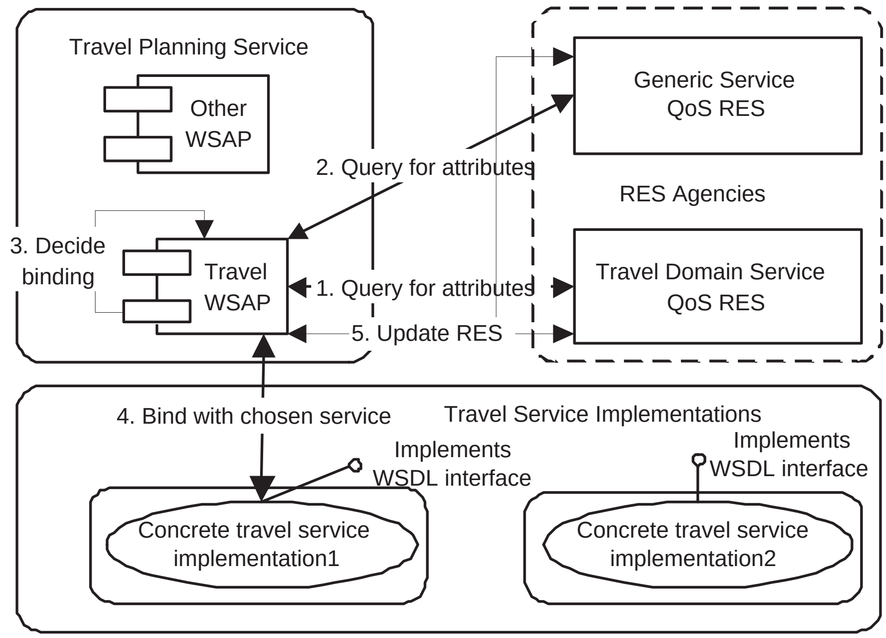

Fig. 3. Travel planning service example.

$$Attr_i = a_{i_{min}} \ldots a_{i_{max}}$$
$$AttrGroup_j = \langle Attr_i \rangle \text{ for } i = 0 \ldots n$$
$$R_{WS_k} = \langle Attr_{l\ Group_j} \rangle \text{ for } l, j = 0 \ldots n$$

Here $a_{i_{min}}$ and $a_{i_{max}}$ are the corresponding min and max value for the range of the attribute $Attr_i$. A particular service reputation is simply a vector of attribute values from different attribute groups.

3.3 Example Usage

As a classic example of service composition, let's consider an aggregate service that offers vacation packages on the web. Clients connect to this service and provide constraints on their preferences, such as travel dates, destinations, price range, and so on. The aggregated service then uses various other services for finding appropriate portions of the packages available that satisfy the given constraints. For instance, it might use an airline service that provides quotes for the clients destinations, available times and dates for departure and return, as well as price ranges for the air travel component of the package.

If the aggregated service uses a WSAP to access each of the individual services, reputation information could naturally be used to optimize the process and in turn to improve the end-user experience. The WSAP is configured to access two RES agencies: one that provides information on real-time service access and general *QoS* and another that provides information on the travel domain *QoS* in particular. The two agencies could be combined into one, but in general they could be independent and our approach leaves them like that.

—Using the generic RES, the travel service could capture information on its interactions with specific travel quoting services, knowing which have reputations for being responsive and reliable. In addition, information on how useful the interactions have been can be inferred if the WSAPs collect data on successful transactions for each travel service used. Finally, pricing information is generic enough that it is an attribute that can be collected for most services and used by the WSAP to optimize the real-time binding to travel services that are within the range specified by the user.

—The travel RES will collect domain-specific information about traveling. This information could include, e.g., the relative satisfaction that users felt, how on-time the travels arrangements have been, the quality of subservices such as food, safety information, schedule flexibility, and so on. This information could be accessed by WSAPs to make better decisions. This will be especially useful if the client—as part of its initial interaction with the travel planning service—can select its preferences in general and specifically for this trip. The WSAP can also be configured with various heuristics for determining how certain client requirements match the gathered information in the RES.

It is clear that without WSAPs and the RES agencies, a travel service must maintain its own information about the services it uses. This would make it difficult to exploit newly introduced services about whom local information is lacking. Using our approach, new services can be added and discovered dynamically, because the agencies are automatically updated for each usage of a new service. Finally, new services that have no historical data but are endorsed by trustworthy principals can get used even before their reputation is established. For the web client, it would be reasonable to entertain this endorsement, because they have a prior relationship with the endorser.

An endorsement can be thought of as a means to bootstrap the reputation of a new service. This is important, because merely by fact of being new, a new service would not have had any users and therefore any reputation information. Clients would discover this new service if they listed the principals that endorse this new service as one of the trusted principals that the client prefers. Of course, a question will be how to weigh reputation attributes in general for a client and specifically when comparing against an endorsed service.

We should note that other factors might also affect an endorsement, such as if a client uses other services provided by the principal that endorses the new service. It might be that the services have been tested to work well together therefore making the endorse service of high value.

4. DISCUSSION

In principle, service standards enable applications to connect to any suitable and accessible service provider. However, a key challenge is to select and locate a service provider who offers the best implementation of a particular service. It is this challenge that our approach addresses. We have only sketched the architecture of the WSAP and the RES agency here. We are building a prototype implementation with applications using WSAPs that are configured to proxy service implementations.

Our efforts open up some interesting directions for further research. A WSAP configuration should enable various run-time and design-time configuration alternatives to be captured in a declarative schema that is sufficiently expressive to cover realistic examples. That is a metamodel for reputation and endorsement will be critical and should be standardized in its own right. Our reputation model is intended to simplify the usage of WSAPs. It faces challenges such as fake reputation entries by impostor agents. The Sopras reputation model tries to address the latter problem by not only having barriers to reputation information but also preventing reputation ratings to increase arbitrarily and using a damping function for the reputation value that limits the memory of the system [Zacharia and Maes 2000].

Our approach presents an architecture for reputation and endorsement. It does not directly address the challenge of accountability, which besets all approaches for distributed trust systems [Oram 2001]. For instance, nothing would stop a WSAP from unfairly setting bad ratings for a service that competes with the owner's own service and thus compromise the integrity of the reputation data. Our present approach concentrates on the programming aspects of services given a reputation system, but it leaves the mechanisms of accountability to how the reputation system is realized. This general challenge can be addressed only by social approaches for reputation where the raters themselves are rated, e.g., [Yu and Singh 2002]. However, there is still the possibility of collusion by the raters.

Since the rating of a service is generally subjective, there needs to be a means to ensure that the WSAP client profile can associate different weights for different attributes. And as discussed above, endorsements need to also have some weight so they can be compared with reputation data. This is especially important for the bootstrapping of services.

Imagine a new service that is not endorsed or is endorsed by a principal that is not trusted. How is this new service to gain any reputation? Possibly, a separate advertising agency could be introduced that could collect new services along with their attributes as advertised by the service provider. The WSAP, if it could not find a perfect match in the RES for its configuration profile, could then query the advertising agency and—depending on the level of risk that the configured WSAP is willing to take—use one of these new services. This is similar to how human buyers may select a service that they have never used before, but decide to take a chance on. And similar to real-life better business bureaus, an agency responsible to rating service provider could be also used so that WSAP could avoid trusting service providers that have not yet collected much reputation data.

It would help to investigate variants of the RES agency, e.g., a referral agency. Since each WSAP collects information about the services that it proxies, we could add intelligence to the agent so that positive interactions are remembered and collected as part of a referral network. When other agents search for services, they could base their decisions on the referral information collected. Referral networks have been used previously for community-based service location [Singh et al. 2001]. Building a referral agency would bring the work on referral networks into our framework.

REFERENCES

COLAN, M. 2001. Dynamic e-business: Using web services to transform business. Tech. rep., IBM.

IBM. 2001. Web services conceptual architecture (WSCA 1.0). http://www.ibm.com.

KIRTLAND, M. 2001. A platform for web services. Tech. rep., Microsoft, http://msdn.microsoft.com.

ORAM, A., Ed. 2001. *Peer-to-Peer: Harnessing the Benefits of a Disruptive Technology*. O'Reilly.

SINGH, M. P., YU, B., AND VENKATRAMAN, M. 2001. Community-based service location. *CACM 44*, 49–54.

UDDI. 2000. UDDI version 2.0 API specification. http://www.uddi.org.

W3C. 2001a. SOAP version 1.2. http://www.w3.org/TR/2001/WD-soap12-20010709/.

W3C. 2001b. Web services description language (WSDL) 1.1. http://www.w3.org/TR/wsdl.

YU, B. AND SINGH, M. P. 2002. An evidential model of distributed reputation management. In *Proc. Intl Joint Conf. on Autonomous Agents and MultiAgent Systems*. To appear.

ZACHARIA, G. AND MAES, P. 2000. Trust management through reputation mechanisms. *Applied Artificial Intelligence 14*, 881–907.