Chapter XXII
Web Services Management: Toward Efficient Web Data Access

Farhana H. Zulkernine
Queen's University, Canada

Pat Martin
Queen's University, Canada

ABSTRACT

The widespread use and expansion of the World Wide Web has revolutionized the discovery, access, and retrieval of information. The Internet has become the doorway to a vast information base and has leveraged the access to information through standard protocols and technologies like HyperText Markup Language (HTML), active server pages (ASP), Java server pages (JSP), Web databases, and Web services. Web services are software applications that are accessible over the World Wide Web through standard communication protocols. A Web service typically has a Web-accessible interface for its clients at the front end, and is connected to a database system and other related application suites at the back end. Thus, Web services can render efficient Web access to an information base in a secured and selective manner. The true success of this technology, however, largely depends on the efficient management of the various components forming the backbone of a Web service system. This chapter presents an overview and the state of the art of various management approaches, models, and architectures for Web services systems toward achieving quality of service (QoS) in Web data access. Finally, it discusses the importance of autonomic or self-managing systems and provides an outline of our current research on autonomic Web services.

INTRODUCTION

The Internet and the World Wide Web have gradually become the main source of information with regard to extent, versatility, and accessibility. Products and services are being traded over the Internet more than ever before. Due to the cost of building and maintaining functionality in a service, outsourcing and acquiring services from other service providers are becoming increasingly popular. Web services are a leading Internet-based technology and a perfect implementation of service-oriented computing (SOC; Casati, Shan, Dayal, & Shan, 2003; Curbera, Khalaf, Mukhi, Tai, & Weerawarana, 2003). It has great potential for being an effective gateway to information accessible on the Web. Web services follow specific standards to ensure interoperability and are accessible on the World Wide Web. In a service-based system, all applications are considered as services in a large distributed network. Web services, which have features like fine-grained functionality, interoperability, and Web accessibility, hold great potential for Web data access and business-to-business (B2B) communication (Hogg, Chilcott, Nolan, & Srinivasan, 2004; Seth, 2002) via cross-vendor service composition.

Efficient management is indispensable to provide good service quality, especially for complex Web service hosting systems that provide services around the clock over the Internet. Quality of service (QoS) is an increasingly important feature of Web data access. It is generally represented by a statistical metric of the system performance, such as the average response time for queries or the level of availability of a service that symbolizes a certain quality of system performance. In order to guarantee the QoS for business and legal aspects, the service provider and consumer should first agree upon a specific service level. This contractual agreement, which is called a service-level agreement (SLA), is a primary economic aspect of Web services management in a corporate environment.

Researchers are working on the architecture, policies, specifications, and enhancement of different standards to facilitate the development of Web services management systems (Farrell & Kreger, 2002). Some of the main management goals are ensuring QoS (Sheth, Cardoso, Miller, Kochut, & Kang, 2002), negotiating SLAs (Liu, Jha, & Ray, 2003), load balancing or resource provisioning (Chung & Hollingsworth, 2004), dynamic reconfiguration (Anzböck, Dustdar, & Gall, 2002), error detection (Sahai, Machiraju, Ouyang, & Wurster, 2001), recovery from failure (Birman, Renesse, & Vogels, 2004), and security (Chou & Yurov, 2005). Most of the management-related research conducted by industry contributes to areas like the architecture and implementation of a management infrastructure for Web services (Catania, Kumar, Murray, Pourhedari, Vambenepe, & Wurster, 2003), the specification of event subscription and notification (*WS-Eventing*, 2004), security and trust relationships (WS-Trust, 2005) in a federated Web services architecture, and the automation of ensuring SLA negotiation (Dan et al., 2004) among coordinating Web services. There are yet many open problems in the area of Web services management that need to be addressed.

This chapter surveys the state of the art of Web services management to facilitate efficient Web data access. It specifically focuses on the importance of an effective management framework for providing reliable, efficient, and secure data access on the Web. The rest of the chapter is organized as follows. The next section provides background information about the architecture and basic standards of Web services. Then the chapter presents an overview and comparison of Web service management frameworks. It explains the criteria used to compare the frameworks, describes frameworks found in the research literature, and then gives a comparison of the frameworks. Finally, it discusses the open problems in Web service management, summarizes the chapter, and draws some conclusions.

BACKGROUND

Web Services

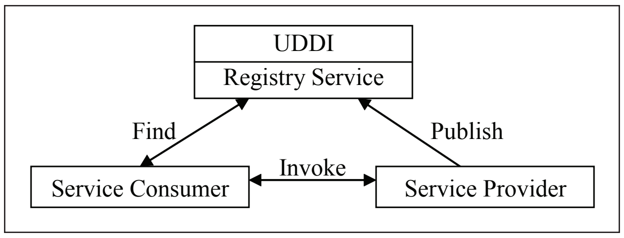
Web services are software applications that offer specific services to client applications and have Web-based interfaces to provide user access over the Internet through standard communication protocols. Information about accessibility, protocols, and the functionality of a Web service is advertised in standard-based registry services like Universal Description, Discovery, and Integration (UDDI; Organization for the Advancement of Structured Information Standards [OASIS] UDDI, 2005) hosted by well-known organizations. Client applications and users can look for required services in the UDDI. If a desired service is found, an SLA (Sahai, Machiraju, Sayal, Van Moorsel, & Casati, 2002) can be negotiated between the client and the Web service, if necessary, and then the service can be invoked (as shown in Figure 1). Client applications communicate with Web services by passing messages over the Internet using standard protocols.

Web services technology is based on standards and specifications in order to ensure interoperability. Two main consortiums, OASIS and the World Wide Web Consortium (W3C), work on standardizing various specifications for Web services. The eXtensible Markup Language (XML; W3C, 2004a) is the basis of all languages used for the communications or specifications of Web services. An XML schema describes data types used in an XML representation. The most frequently used message-passing protocol for Web services is the simple object access protocol (SOAP; W3C, 2003) over the hypertext transport protocol (HTTP). Web services are published in the UDDI using the Web Services Description Language (WSDL; W3C, 2005), which is an XML-based specification language for service interfaces and accessibility. XML, SOAP, and WSDL are required for publishing, discovering, and invoking Web services. Numerous other standards and specifications have been published and are still being worked on for Web services in order to enable automated service discovery, composition, and management.

Several components are needed to host a Web service on the Internet. Management of the Web service system implies managing all these components to ensure satisfactory performance. The components are typically HTTP or the Web server, application server, SOAP engine, Web-service interfaces (WS1 and WS2 in Figure 2), and associated back-end applications. The back-end applications may in turn include database systems and legacy systems (as shown in Figure 2). These components can reside on the same server machine or be distributed among multiple interconnected servers.

SOAP messages to Web services are received by the Web server. There can be one or more instances of the application server acting as containers to host single or multiple Web services. Messages received by the Web server are forwarded to the application servers. These messages are basically XML data bounded by headers and footers containing the messaging protocol. In most cases, the protocol is SOAP and the interpreter used is called the SOAP engine. The engine

Figure 1. Web service life cycle



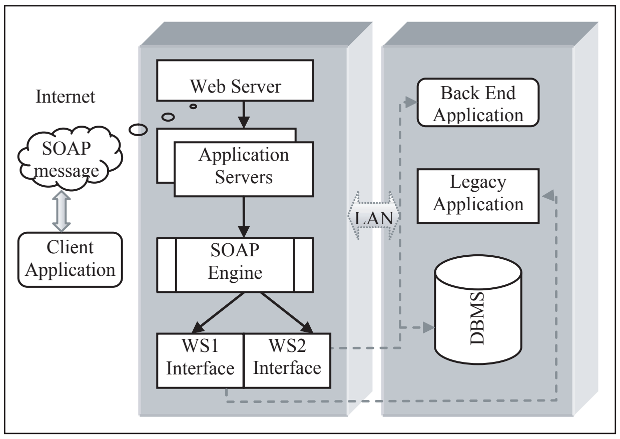
translates the envelope of the message and, after necessary preprocessing, passes the message to the appropriate Web service. Web services are applications built with technologies such as Java Servlets (Sun, 2004) that process the messages and compose replies to send back to the client. A more complex service can require connections to back-end applications, or legacy or database systems. These back-end applications may reside on separate servers connected to the HTTP server via a local area network (LAN).

In a more complex setup, the Web service may invoke other Web services provided by multiple enterprises and thus create a chain process constituting a large cross-vendor distributed, composite system. The client invoking a service may be a user using a Web browser, a client application, or another service. The services in a composite system can be selected and bound either before execution of the system (static composition) or on the fly (dynamic composition), in which case the system is built gradually with each service call. In either case, the structure is based on messages communicated over the Internet between the service requester and the service provider. A Web service can be a service requester and a service provider at the same time. The final response can reach the root or initiator of the call through intermediate service providers or via direct messaging.

WEB SERVICE MANAGEMENT FRAMEWORKS

The management of Web services in its simplest form refers to monitoring, configuring, and maintaining Web services. Web service management is a specific case of network and systems management, which considers issues such as performance, configuration, accounting, fault detection and correction, and security (Cisco Systems, 2005). The various systems management tools and software perform monitoring, reconfiguration, updating, access control, error reporting, and repair. Web services have largely evolved during the last few years from standard-based, loosely coupled, cross-vendor compatible service components to an industry-wide technology having the potential for constructing complex, multivendor composite systems. It is important to the continued growth and acceptance of this technology that the service providers ensure the QoS desired by the customers. This calls for the specification of the desired

Figure 2. Components of a Web service hosting system



level of service through an SLA on the customer's side (Jin, Machiraju, & Sahai, 2002), and efficient management of the Web service system (Farrell & Kreger, 2002) to ensure the required service level on the service provider's side.

The unpredictable nature of the workloads of Web media and the need for high levels of interoperability and accessibility make the management of Web service systems a difficult and challenging task. Web services are also hosted and supported by a number of components, and these components must be managed harmoniously to render a desired service level. This obviously is far more challenging than managing a single application. Moreover, composite Web services can be created by a collaboration of multivendor Web services. The management of such systems, therefore, requires a collaborative service management framework (Lipton, 2004). For Web services, the management system particularly needs to address issues such as the Web media, varying workloads, distributed and heterogeneous clients and system components, and dynamic service composition.

In this section, we first discuss the criteria used to compare Web service management frameworks. A number of management frameworks are then described using a categorization based on their implementation infrastructure. The frameworks are then analyzed based on the given criteria.

Evaluation Criteria

Most of the proposed management approaches by the researchers focus on one or more specific aspects, such as maintaining QoS, SLAs, providing secure and controlled access, ensuring automatic recovery, and provisioning the system resources. These are the most important aspects as they directly influence the business and revenue. They are, therefore, used as criteria for analyzing the various management approaches discussed below.

We also identify two additional criteria for comparing the management frameworks, namely, support for service composition and the implementation model used to develop the framework. The support for service composition can basically be classified as either static or dynamic, where static composition means that Web services are composed prior to run time, and dynamic composition means that Web services can be composed at run time. The implementation model is the main paradigm used to develop the framework, for example, agents, middleware, or reflective programming.

QoS and SLAs

The SLA specifies the required service properties as demanded by the service consumers, while QoS indicates how well a service is functioning. The most common QoS parameter is response time. Depending on the type of the system, other parameters such as memory usage, network delay, and queuing time can be monitored. A statistical QoS data set can also serve as a valuable guide for automatic service selection in a dynamic service composition (Zeng, Benatallah, Ngu, Dumas, Kalagnanam, & Chang, 2004). In most cases, the process of ensuring QoS is carried out by monitoring and analyzing performance data to build a predictive or reactive QoS model. The values of the configuration parameters are then determined to achieve a desired QoS based on the model, and those parameters are modified accordingly to reconfigure the system (Cardoso, Miller, Sheth, & Arnold, 2002; Liu et al., 2003; Sheth et al., 2002). Predictive models are becoming more important from the research perspective than reactive models. Ludwig (2003) discusses the challenges in delivering QoS as specified by SLAs for Web services because of the heterogeneity and network QoS, particularly in the case of composite services.

An SLA may typically define the purpose of the service, authorized parties or customers, the period of validity, the scope, restrictions, penalties, and service-level objectives like availability, performance, and reliability (Jin et al., 2002). The service provider is legally bound to provide the QoS as agreed through the SLA. Monitoring applications are deployed as part of the management application suite usually at both the customer's and service provider's ends to ensure the validity of the SLA throughout the service duration. Automating the process of creating and maintaining the SLA, especially for composite Web service systems, is one of the most challenging problems in the area of Web service management and dynamic service composition.

Recovery

An efficient management system should implement mechanisms to recover from minor and possibly major pitfalls in the least amount of time. Depending on the nature of the failure, the recovery can require a simple restart of the failed resource, or a complex error-tracking process (Sahai, Machiraju, Ouyang, et al., 2001) to identify and replace the failed resource. In a composite Web service system, the detection and replacement of a failed service with a similar service using automated service discovery and selection offers an interesting research problem. Resource provisioning also provides quick recovery from failure using replication with automatic resource replacement.

Security

Security is a major issue for Web services because of the nature of the network and interoperability (Chou & Yurov, 2005). However, currently, no standard security framework ensures all the security features such as authorized access, integrity, authenticity, and confidentiality of the information content. Some of the common approaches to implement security features are the use of access-control lists, security tokens, XML signatures, and message encryption and decryption techniques (Coetzee & Eloff**,** 2004; Wang, Huang, Qu, & Xie, 2004). New standards to support Web service security are being worked on such as the Security Assertion Markup Language (SAML; OASIS SAML, 2005), eXtensible Access Control Markup Language (XACML; OASIS XACML, 2005), and Web services security (WSS; OASIS WSS, 2004). SAML leverages core Web services standards like XML, SOAP, transport layer security (TLS), XML signatures, and XML encryption. It enables the secure exchange of authentication, attribute, and authorization information between disparate security domains, making vendor-independent, single sign-on, secure e-business transactions possible within federated networks on the Web. XACML is an XML-based language for expressing well-established ideas in the field of access-control policy. It provides profiles for SAML 2.0, XML digital signatures, privacy policies, hierarchical or multiple resources, and role-based access control (RBAC). WS-Security provides an industry standard framework for secure Web service message exchanges.

Security policies may be defined and managed at a higher level as part of the management framework. Communicating parties can use these policies to establish and maintain a trust relationship. Furthermore, for specific business partners, a federation can be created and maintained (Wang et al., 2004) where members can share and distribute trust relationships.

Resource Provisioning

The typical "bursty" and varied nature of client requests to a Web service means that the optimum allocation of resources to a Web service is difficult to predict and will vary over time. As a result, without a resource sharing and provisioning strategy, resources can either remain underutilized or may be insufficient to handle user requests at the peak hours. The effective use of resource management techniques has been shown to improve performance in cluster-based Web service environments (Chung & Hollingsworth, 2004). Much research is going on about automatic resource management these days. The automatic replacement of failed resources can also contribute to efficient service recovery.

Management Frameworks

Researchers have proposed many different strategies and approaches to address the various management aspects. There is also considerable activity on Web management in consortiums and standards organizations such as OASIS and W3C. We categorize a number of management approaches based on the infrastructure used in the approach, that is, whether it is centralized, distributed, or autonomic, and describe their main features. Finally, we present self-managing or autonomic management approaches, and thereby describe our research on autonomic Web service systems. Most of the management frameworks have been implemented as models, and therefore the comparative analysis presented in the last section highlights the main features of the models against the evaluation criteria.

Centralized Management

Centralized management schemes employ a single central unit to carry out the management activities. Cibrán, Verheecke, Suvee, Vanderperren, and Jonckers (2004) present the Web service management layer (WSML), a middleware shown in Figure 3, which facilitates the development and management of integrated service applications. WSML supports client-side service management and criteria-based service selection for dynamic service composition. The rich run-time environment of JAsCo, an Aspect-Oriented Programming (AOP) Language, is used to modularize the implementation of the management functionality within WSML (Verheecke, Cibrán, Vanderperren, Suvee, & Jonckers, 2004). WSML lies between the client application and the Web services, and receives client requests for specific service types. A new JAsCo aspect bean is defined dynamically as required holding specific policies for each management aspect like service selection and binding, service swapping, automated billing, caching, and monitoring. JAsCo connectors are

Figure 3. General architecture of the WSML (Cibrán, Verheecke, Suvee et al., 2004)

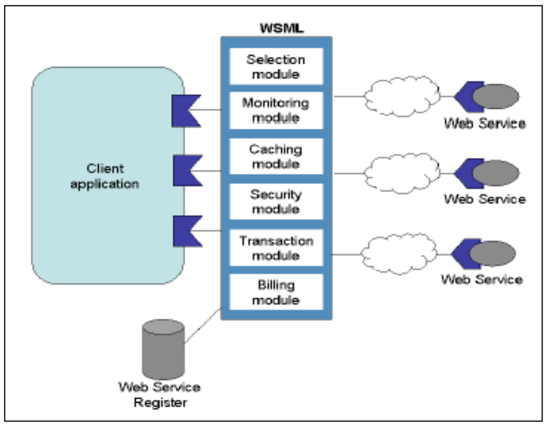
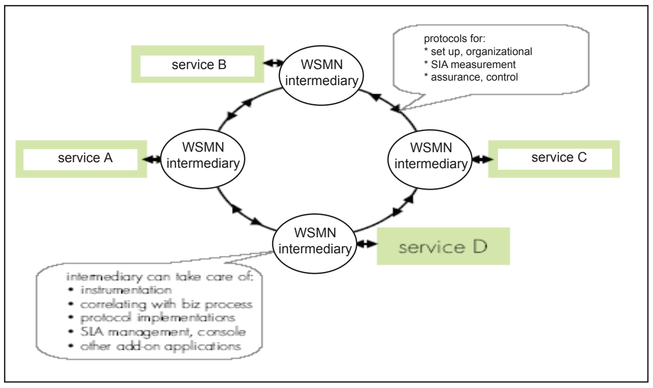


Figure 4. Web Services Management Network (WSMN; Sahai, Machiraju, Sayal et al, 2002)



also created dynamically to bind the client application with a specific Web service based on the policies defined in the aspect bean. Client-side management is implemented using AOP in client applications. By isolating the management tasks in WSML, the replication and maintenance of similar management-related code in the applications is avoided.

Sahai, Machiraju, and Wurster (2001) suggested a management service provider (MSP) model for the remote or outsourced monitoring and controlling of e-services on the Internet. An automated and distributed SLA monitoring engine using the Web service management network (WSMN) agent, as shown in Figure 4 (Sahai et al., 2002), was proposed in a later publication. A specification language for SLAs was also proposed to be used with WSMN. The framework uses a network of cooperating intermediaries as proxy components in between a service and the outside world to enable message tracking. Each intermediary is attached to the SOAP tool kits at each Web service site of a composite process and communicates with each other through a set of protocols specifically defined for this purpose. WSMN agents monitor the process flow defined using WSFL (Web Service Flow Language) to ensure SLA compliance for managing service relationships.

The concept of workflow and process QoS is investigated by a group of researchers at the Large Scale Distributed Information Systems lab (LSDIS) at the University of Georgia. Typically, Web services are composed in a workflow, and it is necessary to manage the QoS metrics for all the services in the composition. Sheth et al. (2002) describe an agent-based service-oriented middleware (SoM) that provides an upper level middleware over Web services-based middleware and leverages the development of multiorganizational applications. SoM uses a process QoS model (Cardoso et al., 2002) that can be used to automatically compute the QoS of a composite Web service workflow process from the QoS metrics of the component Web services.

Fuente, Alonso, Martínez, and Aguilar (2004) propose the reflective and adaptable Web service (RAWS), which is a Web service design model that is based on the concept of reflective programming. It allows the administrators to dynamically modify the definition, behavior, and implementation structure of a Web service during its execution without requiring a shutdown of the service. Technically, the design implements behavioral and structural reflection in a two-level architecture, made of the base level and meta level, to allow the modification of one level to be reflected in the other level.

The Web-services management framework (WSMF) version 2.0 (Catania et al., 2003) defines a general framework for managing different types of resources including Web services and was later published by OASIS as Web-services distributed management (OASIS WSDM, 2005). The framework defines and uses three main concepts: WSMF-Foundation specifies the basic mechanisms for management using Web services, WS-Events introduces a Web-services-based event subsystem, and WSMF-WSM (WSMF-Web service management) describes the management of Web services using WSMF. However, it only allows generic security measures such as the use of HTTPS (secure HTTP), SSL (secure sockets layer) certificates, and access-control mechanisms at the API (application programming interface) level to be implemented.

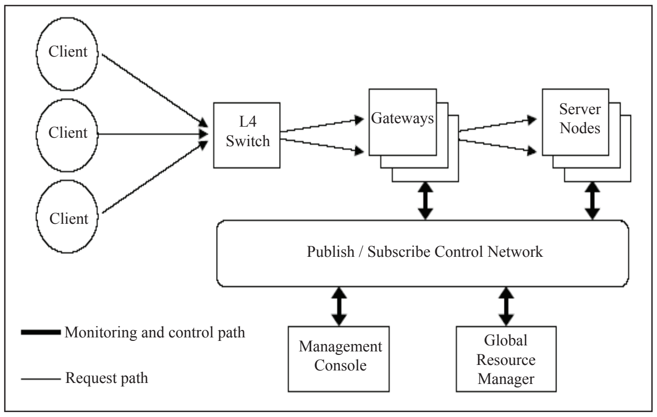
Tosic, Pagurek, Patel, Esfandiari, and Ma (2004) define the Web Service Offering Language (WSOL) to allow the formal specification of important management information such as classes of service, functional and accessibility constraints, price, penalties, and other management responsibilities. The Web service offering infrastructure (WSOI), in turn, demonstrates the usability of WSOL in the management and composition of Web services (Tosic, Ma, Pagurek, & Esfandiari, 2004).

Dobson, Hall, and Sommerville (2005) propose a container-based fault-tolerant system for a general service-oriented architecture (SOA). It implements proxies to pass each incoming service call to the proper service replica as selected by a set of prespecified XML policies, and thus increases the availability and reliability for outsourced services of different types.

Distributed Management

Web service management systems with a distributed framework contain multiple, distributed management components that cooperate to perform the management functions. Researchers at IBM (Levy, Nagarajarao, Pacifici, Spreitzer, Tantawi, & Youssef, 2003) propose an architecture and prototype implementation of a performance-management system for cluster-based Web services as shown in Figure 5. The system supports SLA and performs dynamic resource allocation, load balancing, and server overload protection for multiple classes of Web services traffic. It uses inner level management for the queuing and scheduling of request messages. Outer level

Figure 5. Performance management system for cluster-based Web services (Levy et al., 2003)



management implements a feedback control loop to periodically adjust the scheduling weights and server allocations of the inner level. The average response time is used as the performance metric for a given cluster utility function. The system supports multiple classes of Web-services traffic and allocates server resources dynamically to maximize the expected value of the utility function. However, it requires users to use a subscription interface to register with the system and subscribe to services.

Dan et al. (2004) define a framework that includes the Web Service Level Agreement (WSLA) Language, a system to provision resources based on SLAs, a workload-management system that prioritizes requests according to the associated SLAs, and a system to monitor compliance with the SLAs. The customers are billed differentially according to their agreed service levels.

Aggarwal, Verma, Miller, and Milnor (2004) present a Web service composition framework called METEOR-S (managing end-to-end operations for semantic Web services) to create and manage service composition. The framework allows automatic service selection, which facilitates the recovery and maintenance of composite Web service systems.

Coetzee and Eloff (2004) propose a logic-based access-control framework for single Web service-based systems. The framework implements authentication through identity verification and defines access-control policies to grant authorized access with the help of an authorization manager.

Autonomic Management

Manual management, especially the reconfiguration of numerous tunable system parameters of large heterogeneous complex systems, is becoming a nightmare for system administrators. Researchers are, therefore, seeking solutions to automate various tasks at different levels of system management. The autonomic-computing paradigm promises a completely new era of systems management (Ganek & Corbi, 2003).

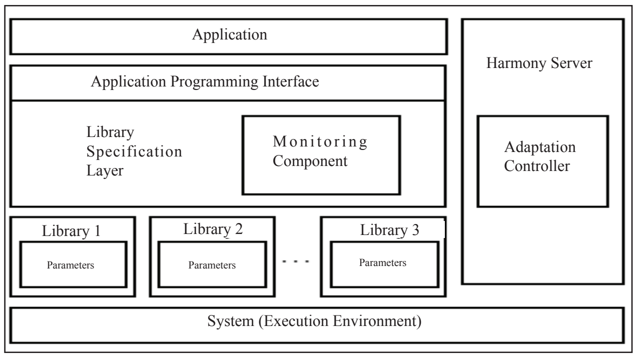
Autonomic systems are self-managing systems that are characterized by the following four properties.

- **Self-configuring:** Systems have the ability to define themselves on the fly to adapt to a dynamically changing environment.
- **Self-healing:** Systems have the ability to identify and fix failed components without introducing apparent disruption.
- **Self-optimizing:** Systems have the ability to provide optimal performance by automatically monitoring and tuning the resources.
- **Self-protecting:** Systems have the ability to protect themselves from attacks by managing user access, detecting intrusions, and providing recovery capabilities.

Autonomic systems are envisioned as imperative for next-generation highly distributed systems. Web services, in particular, can greatly benefit from autonomic computing because of their dynamic workloads and highly accessible communication media like the Internet (Birman et al., 2004). However, many problems need to be addressed in order to materialize the concept of autonomic Web services, rendering it as an interesting research topic. The approaches discussed below can also be categorized under centralized or distributed approaches according to their framework. However, for the self-management feature, they are described under this category.

The core part of an autonomic system is a *controller*, which is designed using either a performance feedback loop in a reactive manner or a feed-forward loop in predictive manner. Researchers propose different methodologies and frameworks to implement the controllers. Different search algorithms are proposed to search for the optimal values of the tunable configuration parameters. Various prediction logics are also followed by different approaches in case of

Figure 6. Active harmony automated tuning system (Chung & Hollingsworth, 2004)

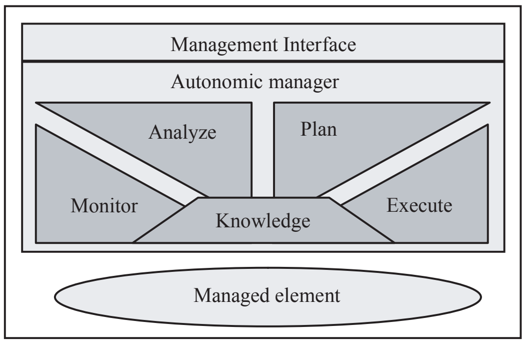


predictive controllers. Typically, a controller can dynamically manipulate the control switches, modify configuration parameters, or implement additional queues based on the performance data to provide optimal throughput.

An automated, cluster-based Web service performance-tuning infrastructure, which is shown in Figure 6, performs adaptive tuning with one or more active harmony servers using parameter replication and partitioning (Chung & Hollingsworth, 2004). The infrastructure contains a technique for resource sharing and distribution that allows active harmony to reconfigure the roles of specific nodes in the cluster during execution to further boost up the performance. In the active harmony framework, applications and services reveal tunable sets of parameters through API that are dynamically tuned based on the workload, past performance history as recorded in a database, and the current performance level. The core part of the server is a controller that implements optimization algorithms for determining the proper value of the tuning parameters.

We also propose an architecture for an autonomic Web-services environment that supports automatic tuning and reconfiguration in order to ensure that the predefined SLAs are met in Tian, Zebedee, Zulkernine, Powley, and Martin (2005). Each component in the system is assumed to be an autonomic element, as shown in Figure 7, that is managed by an autonomic manager (AM). The framework is based on a hierarchy of autonomic

Figure 7. Autonomic element



managers that coordinate with each other to provide an organized autonomic environment for Web services.

The autonomic manager consists of four functional modules that perform the following tasks: periodically monitor performance data; analyze the data based on past, current, and targeted performance; plan which parameters to tune in order to achieve the performance goal; and finally execute the action plan. We propose using reflection-based feedback controllers for our autonomic managers. A reflective system is one that can inspect and adapt its internal behavior in response to changing conditions. Typically, a reflective system maintains a model of self-representation, and changes to the self-representation are automatically reflected in the underlying system.

Control-theoretic approaches have been proposed by some researchers to implement the controllers. ControlWare is a middleware that embodies a control-theoretical methodology for QoS provisioning for software services (Abdelzaher, Stankovic, Lu, Zhang, & Lu, 2003). The feedback control mechanism guarantees QoS through the optimized allocation of system resources like various queues and cache sizes to different processes.

Bennani and Menascé (2004) create self-managing computer systems by incorporating mechanisms for self-adjusting the configuration parameters so that the QoS requirements of the system are constantly met. Their approach combines analytical performance models with combinatorial search techniques to develop controllers that run periodically to determine the best possible configuration for the system given its workload.

Birman et al. (2004) extend the general architecture of Web service systems to add high availability, fault tolerance, and autonomous behavior. The authors refer to the standards for Web services like WS-Events, WS-Transaction, and WS-Reliability while discussing failure protection, speedy recovery, and reliable messaging. The architecture includes server- and client-side monitoring, a consistent and reliable messaging system using information replication, a data-dissemination mechanism using multicasting, and an event notification system using WS-Events standards.

A framework for the deployment and subsequent autonomic management of component-based distributed applications is proposed by Dearle, Kirby, and McCarthy (2004). In their work, deployment goals are defined by specifying constraints over available resources, component mappings, and the interconnection topology. An autonomic deployment and management engine (ADME) is used to find a configuration that satisfies the goal, and the configuration is deployed automatically using the Cingal infrastructure (Dearle, Kirby, McCarthy, & Diaz y Carballo, 2004*)*. If a deviation or change in the goal is detected, the configuration finder and deployment cycle is repeated automatically to generate a revised deployment. The proposed framework mainly addresses the self-configuration and self-healing aspects of autonomic management.

Comparative Analysis of the Management Approaches

The various approaches discussed above address one or more of the management aspects but not all of them. The QoS aspect has been addressed by several researchers in different ways. Sheth et al. (2002) introduce an additional agent-based middleware for computing the QoS of individual Web services and use the model proposed by Cardoso et al. (2002) to compute and thereby monitor the process QoS of a composite service system. Dan et al. (2004) present a more comprehensive distributed framework that contains a service-offering unit, WSLA-based QoS monitoring and differentiated service provisioning using a workload-management unit, and a resource-provisioning unit. Sahai et al. (2002) propose a

Table 1.

Reference	Type*	QoS/SLA	Security	Recovery	Service Composition	Resource Provisioning	Implementation Model
Cardoso et al., 2002	Cent.	Computes work-flow QoS	N/A	N/A	Static	N/A	Mathematical model
Sheth et al., 2002	Cent.	Computes QoS of composite service	N/A	N/A	Static	N/A	Agent-based middleware
Dan et al., 2004	Dist.	WSLA-based differentiated QoS provisioning	N/A	By component replacement	Static with service-offering unit	Separate unit for dynamic provisioning	Distributed-unit-based framework
Sahai et al., 2002	Cent.	WSFL-based monitoring for SLA compliance	N/A	N/A	Static	N/A	Agent-based network of intermediaries
Tosic et al., 2004	Cent.	WSOL for service offering and SLA	Usual security measures	N/A	Static or dynamic	N/A	WSOI for implementation of WSOL
Levy et al., 2003	Dist.	Differentiated reactive weighted message scheduling	N/A	Resource replacement and overload protection	Static	Dynamic server allocation & load balancing	Two-level management
Dobson et al., in press	Cent.	Higher availability & quick recovery	N/A	Service replication	Unsuitable for composite systems	Policy-based replica selection	Fault-tolerant system
Aggarwal et al., 2004	Cent.	Work-flow QoS	N/A	Dynamic service selection and replacement	Static or dynamic	N/A	METEOR-S framework

* Cent.- Centralized, Dist.- Distributed, Auto. – Autonomic

Table 1. Continued

Reference	Type*	QoS/SLA	Security	Recovery	Service Composition	Resource Provisioning	Implementation Model
Fuente et al., 2004	Cent.	Undisturbed during update	N/A	Source-code modification	Static or dynamic	N/A	Bi-level reflective programming
Verheecke et al., 2004	Cent.	Both client- and server-side management	Access control	Dynamic service selection and replacement	Static or dynamic	N/A	Middleware (WSML) using AOP
Coetzee & Eloff, 2004	Cent.	N/A	Policy-based access control	N/A	Not supported	N/A	Authentication and authorization verification
Abdelzaher et al., 2003	Auto.	Dynamic tuning using feedback control	N/A	Parameter tuning and reconfiguration	Static or dynamic	Message queuing and scheduling	Control-theory-based ControlWare
Bennani & Menascé, 2004	Auto.	Dynamic tuning using analytical model	N/A	Parameter tuning and reconfiguration	Static with service-offering unit	None	Uses a performance evaluation model
Chung & Hollingsworth, 2004	Dist. & auto.	Dynamic tuning of only registered parameters	N/A	Reconfiguration, resource sharing and distribution	Static or dynamic	Message queuing and scheduling, resource sharing	Active harmony for cluster-based systems
Tian et al., in press	Auto.	Management of SLA and QoS	N/A	Dynamic parameter tuning and reconfiguration	Static or dynamic	N/A	Reflective-programming-based hierarchy of AMs

* Cent. – Centralized, Dist. – Distributed, Auto. - Autonomic

less comprehensive agent-based WSMN using intermediaries as proxies that enables message tracking to monitor SLA compliance. However, it requires additional protocols for the intermediaries to communicate within themselves securely. WSOI mainly focuses on the application and manipulation of WSOL, a language for the formal specification of service offerings, and hence does not serve as a complete infrastructure for Web service management (Tosic, Pagurek, Patel et al., 2004). The framework by Levy et al. (2003) uses a feedback controller for weighted scheduling, message queuing, and server allocation in a cluster-based system. However, the clients need to subscribe to the service prior to using it.

The recovery aspect is addressed by Dobson et al. (2005), Aggarwal et al. (2004), and Fuente et al. (2004) in different ways. Dobson et al. designed a fault-tolerant system. However, it is difficult to apply rollback or other fault-tolerance mechanisms in case of Web services, especially in composite systems. Aggarwal et al. suggest dynamic service selection and composition in a workflow process using the METEOR-S framework to allow quick recovery. The bi-level architecture by Fuente et al. implements reflective programming to enable dynamic source-code update for Web services and thus facilitates deployment and recovery.

The WSML middleware implements each management aspect using separate aspect beans in AOP (Verheecke et al., 2004). It facilitates the modular implementation of various management aspects such as dynamic service selection, monitoring, and access control, and enables client-side management. Coetzee and Eloff (2004) propose a policy-based access-control framework for only single Web service based systems. Abdelzaher et al. (2003), Bennani and Menascé (2004), Chung and Hollingsworth (2004), and Tian et al. (2005) propose autonomic management approaches that perform parameter tuning and reconfigurations to maintain QoS. Abdelzaher et al. use control theory to design the feedback control framework augmented by elements of scheduling and queuing theory for resource provisioning. Bennani and Menascé implement a performance evaluation model to search for the best configuration parameters to use with the controller, but the approach does not support resource provisioning. Chung and Hollingsworth use the active-harmony server to tune only the parameters that are registered with the server by the applications and services. In a cluster environment, it also performs resource sharing and distribution, and thereby supports quick recovery. Tian et al. use reflective programming with a hierarchy of autonomic managers to maintain QoS in a varying workload. However, none of the above approaches provide access control.

CONCLUSION

Open Problems

The above survey demonstrates that none of the proposed management frameworks addresses all the management aspects such as QoS, SLA negotiation, dynamic reconfiguration, security, recovery, and resource provisioning for dynamically composed systems. There has been considerable work done in the field of QoS and SLAs for Web services, but more research is needed to resolve the issues related to service dependency and providing differential services to different clients based on the SLA. The monitoring and maintenance of QoS and SLA for a composite service system is a challenging problem, especially when the QoS of the network is considered. The ranking of services based on specific service selection criteria is necessary to make automatic and run-time service selection. The publication and verification of QoS information that may be used as service selection criteria offer other interesting open problems.

Automatic service discovery and binding implemented by a management system can provide effective failure protection and recovery. However, this requires the use of well-defined semantics and specifications to enable automatic search and replacement of similar services.

Due to the nature and usage of Web media, they are vulnerable to malicious attacks. A good management framework should protect the system from such hazards. Standard specifications have been published for Web service security, but further research is needed to provide application-level security for Web services and to construct a secure management framework. End-user privacy is another important research area for Web services that is yet to be explored. The W3C is working on the Platform for Privacy Preferences Project (P3P) to provide a simple, automated way for users to gain more control over the use of personal information on Web sites they visit (W3C, 2004b). Privacy policies can be defined by service providers, which can be matched by service consumers before subscribing to a service. However, the monitoring of policy agreements at the provider's side is another problem.

Researchers are working on various policy specifications that may be incorporated into the management frameworks to implement policy-based management. The policies defined by service users need to be translated into system-usable formats. Further research is required to define strategies and implementation models to perform this translation effectively for different Web-services-based systems. SLA specification and QoS management for Web service systems have gained a lot of attention because of the wide range of user types and requirements. Resource provisioning for cluster-based Web services (Chung & Hollingsworth, 2004) is another important research problem.

With the rapid growth in the size and complexity of software systems, researchers are working toward designing autonomic systems featuring self-configuring, self-healing, self-optimizing, and self-protecting properties to reduce human intervention. Currently, fully autonomic systems do not exist. Researchers are now working on achieving partial autonomic behavior through various control mechanisms. Regardless of the additional system components and increased management overhead, autonomic computing promises huge potential benefits as a systems-management solution. The existing solutions, in most cases, do not consider composite systems. New specifications and standards can be used to update the existing frameworks to provide enhanced management functionality.

SUMMARY

The World Wide Web is currently perceived as an indispensable resource of information. Different search engines and Web sites provide us with tools to search, select, and retrieve necessary information from the Internet. Most of these tools have an application at the back end that processes user requests and generates the responses to be sent back to the user. Web services are independent software applications that provide specific services to users or client applications on the Web through standard protocols and user interfaces. Thus, a Web service can be designed to effectively provide customized Web data management and powerful information-retrieval services over the Internet. Web services are also an effective implementation of SOC. The immense potential of composing Web services over multiple enterprises and building platform-independent, complex, interacting software systems has elevated the importance of Web services as a research area.

The management frameworks described are mostly research models and do not yet have any practical implementations. Some of the large companies are working together on standardizing various protocols like WS-Eventing, WS-Transfer, and WS-Management for Web-services management. WSMF by HP was later published as WSDM by OASIS. Currently, only a few Web service management software are available in the market. Managed Methods has released a Web service management software called JaxView to monitor availability, throughput, response time, faults, policies, and messages to assist service management. Mike Lehmann (2004), in an article in Oracle Magazine, presents new Web-services management capabilities in Oracle Application Server. AmberPoint Express is another management software that assists Web service developers to incrementally measure, debug, and fine-tune the performance and functionality of their Web services.

During the last few years, Web service technology has rapidly grown to become a part of our everyday lives. Many popular Web sites like Amazon.com are using Web services to serve customer enquiries. The success of this interactive Web technology largely depends on customer satisfaction. An efficient and reliable management framework is indispensable to ensure around-the-clock availability and service quality of Web service systems. This chapter provides an introduction and background on Web service management and outlines some of the challenging research problems in the area. It presents the state of the art of various management approaches for Web-services management. Newer and more interesting management problems will continue to arise from the growing complexity and expansion of this technology. The ongoing research on service management is focused toward achieving automated solutions requiring minimum human intervention. We envision autonomic computing as the most efficient technique for the continuous monitoring and management of large, heterogeneous, and complex systems such as composite Web service based systems.

REFERENCES

Abdelzaher, T. F., Stankovic, J. A., Lu, C., Zhang, R., & Lu, Y. (2003). Feedback performance control in software services. *IEEE Control Systems Magazine, 23*(3), 74-90.

Aggarwal, R., Verma, K., Miller, J., & Milnor, W. (2004). Constraint driven Web service composition in METEOR-S, services computing. *Proceedings of the IEEE International Conference on Services Computing SCC'04* (pp. 23-30).

Anzböck, R., Dustdar, S., & Gall, H. (2002). Software configuration, distribution, and deployment of Web-services. *ACM International Conference Proceeding Series: Vol. 27***.** *Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering (SEKE'02): Web-Based Tools, Systems and Environments* (pp. 649-656).

Bennani, M., & Menascé, D. (2004). Assessing the robustness of self-managing computer systems under highly variable workloads. *Proceedings of the International Conference on Autonomic Computing (ICAC'04)*, 62-69.

Birman, K., Renesse, R. V., & Vogels, W. (2004). Adding high availability and autonomic behavior to Web services. *Proceedings of the 26th Annual International Conference on Software Engineering (ICSE'04)* (pp. 17-26).

Cardoso, J., Miller, J., Sheth, A., & Arnold, J. (2002). *Modeling quality of service for workflows and Web service processes* (Tech. Rep. No. 02-002 v2). University of Georgia, Department of Computer Science, LSDIS Lab, Athens.

Casati, F., Shan, E., Dayal, U., & Shan, M. (2003). Service-oriented computing: Business-oriented management of Web services. *Communications of the ACM, 46*(10), 55-60.

Catania, N., Kumar, P., Murray, B., Pourhedari, H., Vambenepe, W., & Wurster, K. (2003). *Overview: Web services management framework.* Hewlett-Packard Company. Retrieved June 20, 2006, from http://devresource.hp.com/drc/specifications/wsmf/WSMF-Overview.jsp

Chou, D. C., & Yurov, K. (2005). Security development in Web services environment. *Computer Standards & Interfaces, 27*(3), 233-240.

Chung, I., & Hollingsworth, J. K. (2004). Automated cluster-based Web service performance tuning. *Proceedings of IEEE Conference on High Performance Distributed Computing (HPDC'04)*, (pp. 36-44).

Cibrán, M. A., Verheecke, B., Suvee, D., Vanderperren, W., & Jonckers, V. (2004). Automatic service discovery and integration using semantic descriptions in the Web services management layer. *Journal of Mathematical Modelling in Physics, Engineering and Cognitive Studies, 11*, 79-89.

Cisco Systems. (2005). Network management basics. In *Internetworking technologies handbook* (ch. VI). Retrieved June 20, 2006, from http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/nmbasics.pdf

Coetzee, M., & Eloff, J. H. P. (2004). Towards Web service access control. *Computers & Security, 23*(7), 559-570.

Curbera, F., Khalaf, R., Mukhi, N., Tai, S., & Weerawarana, S. (2003). Service-oriented computing: The next step in Web services. *Communications of the ACM*, *46*(10), 29-34.

Dan, A., Davis, D., Kearney, R., Keller, A., King, R., Kuebler, D., et al. (2004). Web services on demand: WSLA-driven automated management. *IBM Systems Journal*, *43*(1), 136-158.

Dearle, A., Kirby, G., & McCarthy, A. (2004). A framework for constraint-based deployment and autonomic management of distributed applications. *International Conference on Autonomic Computing (ICAC'04)*, 300-301.

Dearle, A., Kirby, G. N. C., McCarthy, A., & Diaz y Carballo, J. C. (*2004).* A flexible and secure deployment framework for distributed applications. In W. Emmerich & A. L. Wolf (Eds.), *Proceedings of 2nd International Working Conference on Component Deployment (LNCS* 3083, *pp. 219-233). Edinburgh, UK*: Springer.

Dobson, G., Hall, S., & Sommerville, I. (2005, May). A container-based approach to fault tolerance in service-oriented architectures. *Proceedings of International Conference of Software Engineering (ICSE),* St. Louis, Missouri. Retrieved June 20, 2006, from http://digs.sourceforge.net/papers/2005-icse-paper.pdf

Farrell, J. A., & Kreger, H. (2002). Web services management approaches. *IBM Systems Journal, 41*(2), 212-227.

Fuente, J., Alonso, S., *Martínez*, O., & Aguilar, L. (**2004).** RAWS: Reflective engineering for Web services. *Proceedings of IEEE International Conference on Web Services (ICWS'04)*, (p. 488).

Ganek, A. G., & Corbi, T. A. (2003). The dawning of the autonomic computing era. *IBM System Journal, 42*(1), 5-18.

Hogg, K., Chilcott, P., Nolan, M., & Srinivasan, B. (2004). An evaluation of Web services in the design of a B2B application. *Proceedings of the 27th Conference on Australasian Computer Science,* **(vol. 26,** pp. 331-340).

Jin, L., Machiraju, V., & Sahai, A. (2002). *Analysis on service level agreement of Web services.* Software Technology Laboratory, HP Laboratories. Retrieved June 20, 2006, from http://www.hpl.hp.com/techreports/2002/HPL-2002-180.pdf

Lehmann, M. (2004). Web services management arrives. *Oracle Magazine*. Retrieved June 20, 2006, from http://www.oracle.com/technology/oramag/oracle/04-nov/o64web.html

Levy, R., Nagarajarao, J., Pacifici, G., Spreitzer, M., Tantawi, A., & Youssef, A. (2003, March). *Performance management for cluster based Web services* (IBM Tech. Rep.). In *Proceedings of the IFIP/IEEE 8th International Symposium on*

Integrated Network Management (IM 2003), (pp. 247-261). Norwell, MA: Kluwer.

Lipton, P. (2004). Composition and management of Web services. In *Service-oriented architecture* (White paper). Retrieved June 20, 2006, from http://webservices.sys-con.com/read/43567.htm

Liu, B., Jha, S., & Ray, P. (2003). Mapping distributed application SLA to network QoS parameters. *Proceedings of ICT*, (pp. 1230-1235).

Ludwig, H. (2003). *Web services QoS: External SLAs and internal policies or: How do we deliver what we promise?* Proceedings of the 1st Web Services Quality Workshop, Rome.

Organization for the Advancement of Structured Information Standards Extensible Access Control Markup Language (OASIS XACML). **(2005).** *OASIS Extensible Access Control Markup Language (XACML) technical committee specification, v2.0.* Retrieved June 20, 2006, from http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml

Organization for the Advancement of Structured Information Standards Security Assertion Markup Language (OASIS SAML). (2005). *OASIS security services specification, v2.0, 2005.* Retrieved June 20, 2006, from http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security

Organization for the Advancement of Structured Information Standards Universal Description, Discovery and Integration (OASIS UDDI). (2005). *OASIS universal description, discovery and integration technical committee specification, v3.0.2.* Retrieved June 20, 2006, from http://www.uddi.org/specification.html

Organization for the Advancement of Structured Information Standards Web Services Distributed Management (OASIS WSDM). (2005). *Web services distributed management (WSDM), v1.0.* Retrieved June 20, 2006, from http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsdm

Organization for the Advancement of Structured Information Standards Web Services Reliability (OASIS WS-Reliability). (2004). *OASIS Web services reliable messaging technical committee specification, v1.1.* Retrieved June 20, 2006, from http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsrm

Organization for the Advancement of Structured Information Standards Web Services Security (OASIS WSS). (2004). *OASIS Web services security (WSS) technical committee specification, v1.0.* Retrieved June 20, 2006, from http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss

Sahai, A., Machiraju, V., Ouyang, J., & Wurster, K. (2001). *Message tracking in SOAP-based Web services* (Tech. Rep.). HP Labs. Retrieved June 20, 2006, from http://www.hpl.hp.com/techreports/2001/HPL-2001-199.pdf

Sahai, A., Machiraju, V., Sayal, M., Van Moorsel, A., & Casati, F. (2002, October). Automated SLA monitoring for Web services. In *Proceedings of the 13th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management (DSOM'02)* (LNCS 2506, pp. 28-41), Montreal, Canada. Springer-Verlag.

Sahai, A., Machiraju, V., & Wurster, K. (2001). Monitoring and controlling Internet based e-services. *Proceedings of the 2nd IEEE Workshop on Internet Applications (WIAPP'01)*, 41.

Seth, M. (2002). *Web services: A fit for EAI* (White paper). Retrieved June 20, 2006, from http://www.developer.com/tech/article.php/10923_1489501_2

Sheth, A., Cardoso, J., Miller, J., Kochut, K., & Kang, M. (2002). QoS for service-oriented middleware. *Proceedings of the 6th World Multiconfer-*

ence on Systemics, Cybernetics and Informatics (SCI'02), 528-534.

Sun. (2004). *J2EE, Java Servlet technology.* Retrieved June 20, 2006, from http://java.sun.com/products/servlet/

Tian, W., Zebedee, J., Zulkernine, F., Powley, W., & Martin, P. (2005, May). Architecture for an autonomic Web services environment. *Proceedings of 7th International Conference on Enterprise Information Systems (ICEIS'05)*, Miami, Florida, (pp. 54-66). Retrieved from http://www.cs.queensu.ca/home/cords/wsmdeis.pdf

Tosic, V., Ma, W., Pagurek, B., & Esfandiari, B. (2004, April). Web services offerings infrastructure (WSOI): A management infrastructure for XML Web services (Tech. Rep.). *Proceedings of IEEE/IFIP Network Operations and Management Symposium (NOMS)* (vol. 1, pp. 817-830), Seoul, South Korea. Retrieved from http://www.sce.carleton.ca/netmanage/papers/TosicEtAlResRepAug2003.pdf

Tosic, V., Pagurek, B., Patel, K., Esfandiari, B., & Ma, W. (in press). Management applications of the Web Service Offerings Language (WSOL). *Information Systems.*

Verheecke, B., Cibrán, M. A., Vanderperren, W., Suvee, D., & Jonckers, V. (2004). AOP for dynamic configuration and management of Web services in client-applications. *International Journal on Web Services Research (JWSR), 1*(3), 25-41.

Wang, H., Huang, J., Qu, Y., & Xie, J. (2004). Web services: Problems and future directions. *Journal of Web Semantics*, *1*(3), 309-320.

Web services for management. (2005). Retrieved June 20, 2006, from http://developers.sun.com/techtopics/webservices/management/WS-Management.Feb.2005.pdf

Web service transfer. (2004). Retrieved June 20, 2006, from http://msdn.microsoft.com/library/en-us/dnglobspec/html/ws-transfer.pdf

WS-eventing. (2004). Retrieved June 20, 2006, from http://ftpna2.bea.com/pub/downloads/WS-Eventing.pdf

WE-Trust. (2005, February). *Web Service Trust Language (WS-Trust).* Retrieved from http://specs.xmlsoap.org/ws/2005/02/trust/WS-Trust.pdf

World Wide Web Consortium (W3C). (2003). *SOAP version 1.2 part 1: Messaging framework.* Retrieved June 20, 2006, from http://www.w3.org/TR/soap12-part1/

World Wide Web Consortium (W3C). (2004a). *Extensible Markup Language (XML).* Retrieved June 20, 2006, from http://www.w3.org/XML/

World Wide Web Consortium (W3C). (2004b). *The platform for privacy preferences project (P3P).* Retrieved from http://www.w3.org/P3P/

World Wide Web Consortium (W3C). (2005). *Web Services Description Language (WSDL) version 2.0* (Working draft). Retrieved June 20, 2006, from http://www.w3.org/2002/ws/desc/

Zeng, L., Benatallah, B., Ngu, A. H. H., Dumas, M., Kalagnanam, J., & Chang, H. (2004). QoS-aware middleware for Web services composition. *IEEE Transactions on Software Engineering, 30*(5), 311-327.