TODAY'S IT CHALLENGES: QUALITY WEB SERVICES AND SERVICE ORIENTED ARCHITECTURES

Anca Alexandra PURCĂREA[1], Lucia SANDOVICI[2], Dan DUMITRIU[3]

World Wide Web-ul evolueaza intr-un mediu care ofera o paleta foarte larga de e-commerce, business-to-business, business-to-consumer si alte servicii bazate pe informatie. In tehnologia Arhitecturii Orientata pe Servicii (SOA), serviciile Web evolueaza intr-o tehnologie care aliniaza sisteme decuplate din mai multe platform, limbaje de programare si aplicatii. Beneficiile imbunatatirii calitatii serviciilor IT cand se folosesc servicii Web si SOA vin cu pretul introducerii unor noi nivele de complexitate in mediile unde aceste servicii sunt prestate. Aceasta complexitate este alcatuita din libertatea de a compune Servicii Web care sa adreseze necesitati cum ar fi calitatea serviciilor, disponibilitatea, securitatea,si costul. Serviciile Web imbinate cu SOA permit o noua paradigma pentru design-ul si operarea aplicatiilor in afaceri.

The World Wide Web is evolving into a medium providing a wide array of e-commerce, business-to-business, business-to-consumer, and other information-based services. In Service Oriented Architecture (SOA) technology, Web Services are emerging as the enabling technology that bridges decoupled systems across various platforms, programming languages, and applications. The benefits of improving IT services' quality when using Web Services and SOA come at the expense of introducing new level of complexity to the environments where these services are deployed. This complexity is compounded by the freedom to compose Web Services to address requirements such as quality of service (QoS), availability, security, reliability, and cost. Web Services deployed in SOA enable a new paradigm for the design and operation of business applications.

Keywords: Quality, Service Oriented Architectures, QoS, IT Services

1. Introduction: today's IT Challenges

Complexity is a fact of life in information technology (IT). Dealing with the complexity while building new applications, replacing existing applications, and keeping up with all the maintenance and enhancement requests represents a major challenge.

[1]Professor, Department of Management, University "Politehnica" of Bucharest, ROMANIA
[2] Professor, Department of Management, University "Politehnica" of Bucharest, ROMANIA
[3]Assistant, Department of Management, University "Politehnica" of Bucharest, ROMANIA

The pressures on IT executives have strongly increased during the past years: pressures to cut costs while at the same time to innovate IT-infrastructure, to improve customer services, to be more competitive and to respond quickly to the business's strategic priorities.

Two fundamental reasons are behind all of this hassle. The heterogeneous system and application landscape and the rapidly changing market requirements.

Most enterprises cannot take a single-vendor approach to IT, because available application suites are not sufficient and flexible enough, which potentially leads to higher costs. Embracing heterogeneity therefore often is the better practice, if a best-of-breed approach is the more efficient way. Interoperability obviously is the only way to overcome these inconsistencies and may even drive new IT investment.

Increasing market changes are the second issue. Business must rapidly adapt in today's dynamic competitive environment, and the IT infrastructure must follow. Change is an ever-present issue in today's IT world for several reasons:

Broad economic forces including globalization and e-business are accelerating the pace of change. Global competition leads to shortening product cycles, as companies try to gain advantage over their competition. Customer needs and requirements change more quickly in response to this cycle of competitive improvements in the quality of products and services.

Finally improvements in technology continue to accelerate, feeding the increased pace of changing customer requirements.

Executives need to find ways to solve today's problems. Such problems for example are:

- Costly, inflexible integration technologies that cause unacceptable risks to the enterprise.
- Increasing complexity of IT environment.
- Monolithic business applications that require expensive customization and maintenance.
- Getting locked into vendors products.
- Complex automation of business processes that involve partners and customers with inadequate security.
- Lack of visibility and control into automated business processes for line-of-business management.
- Limited ability to participate in value networks because of complexity and insufficient security.

Business is calling upon IT more than ever to respond quickly and efficiently to these multiple and serious challenges. In this tough environment enterprises highly welcome an evolutionary, standards-based architectural approach, which addresses the issues mentioned above.

This approach is a Web services based technologies known as Service Oriented Architecture (SOA).

2. The Web Services Paradigm and SOA

The Web services paradigm refers to SOA, an architecture that constitutes a distributed computing environment in which applications call functionality from other applications either locally or remotely over an internal network or an IP-network in a loosely-coupled way.

Web services are self-contained, self-describing modular and autonomous applications, encapsulated and invokable via standardized XML based interfaces. On the foundation of the XML standard the modular nature of XML Web services can be seen as the next logical step in application development.

Although advanced component models such as J2EE/EJB and Microsoft's .NET Object Model have enjoyed widespread acceptance within companies, extended component models that enable components to work across networks have been less successful. That is mainly due to two factors:

- The incompatibility of the two camps' frameworks (Java and Microsoft). Both component-based frameworks use different platforms and a different scheme to communicate. A network application using J2EE cannot communicate easily with one based on the .NET architecture.
- As companies' security policies usually require that their firewalls block network services such as those used by software components, both of the component-based frameworks cannot be used successfully on an inter-enterprise basis.

The overcome of these deficiencies through SOA and Web services means either being deployed intra-enterprise as an easier way to do distributed computing or being published, located and invoked across the Internet. Web services published on the Web (software as a service) can be used by any application at any time. The Web services components will remain consistently regardless of various deployments.

The goal of Web services is to enable application-to-application communication. This is achieved by a document-oriented way of distributed computing. The following aspects describe what makes up Web services:

- The Service is represented by a software that is first of all capable to process an XML document it has received from an invoking application via any internal interface or IP network. The internal structure of this software as well as the realization of the requested functionality does not play any role. Whether it is based on object-oriented techniques or it operates as a stand-alone process or is a thin layered front-end of an existing enterprise application, is not of any importance. This software

only needs to be capable of processing well-defined XML documents and performing the functionality requested in these documents.

- The XML document is the keystone of a Web service, as it contains all the application-specific information. This document is described using an XML scheme, and the two applications that are engaged in the conversation need to have knowledge of the same description in order to interpret and validate the document correctly. For these descriptions the Web Services Description Language (WSDL) is used.
- The Address defines where the service can be found, i.e. a transport protocol (e.g. TCP, HTTP etc.) combined with a network address.

The Envelope encapsulates the XML document and some system information the two communicating applications may want to exchange in the message. This system information allows, for example, routing and security information to be added to the message without the need to modify the XML document. The message protocol that is used for almost all Web services is SOAP (Simple Object Access Protocol). The SOAP message consists of two possible elements: a soap-header, in which all the system information is kept, and a soap-body, which contains the XML document that is to be processed by the Web service.

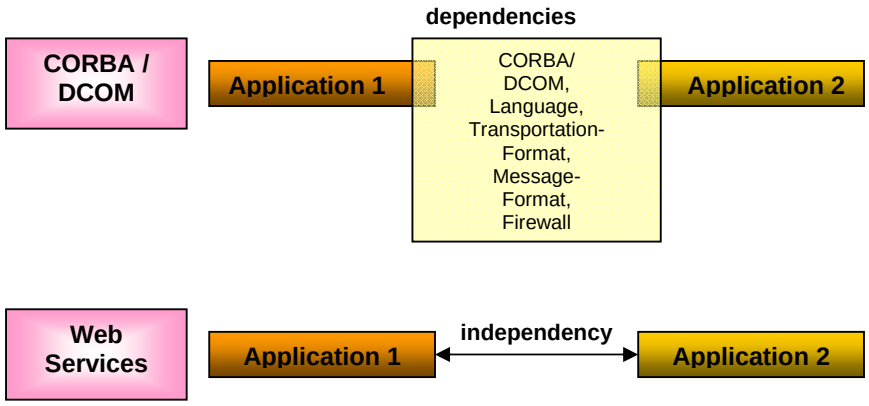

Fig. 1 – Web Services' Independency of Platforms

Moreover the passing through firewalls using HTTP is enabled, WSDL is bound to the service it describes and the service is accessed electronically at a location, which may not be known to the requester but can automatically be discovered.

SOA based on Web services is characterized by the ability of publishing a service on an IP-network in a registry.

The network is either public (Internet) or private (enterprise). The registry is standardized, called UDDI (Universal Description, Discovery and Integration) and may be either public or private

UDDI consists of an XML schema that defines four data structures: business definition, service definition, binding specification and programmatic interface as well as a set of APIs that operate on those structures.

The WSDL listing in the UDDI is comprised of three elements: the white pages, which contain basic information about the providing company and its services, the yellow pages, which organize services by industry, service type or geography, finally the green pages, which include the technical mechanisms, i.e. how to find (e.g. URL) and execute a Web service.

The SOA provides in addition to the publishing two other significant capabilities, that are taken advantage of at runtime of an application that invoke a Web service: the discovery, the ability to find a required functionality (service) and the binding, the ability to connect automatically to this functionality.

Figure 2 shows the three corresponding roles:

- the Web service provider
- the Web service requester and
- the Web service broker (UDDI).

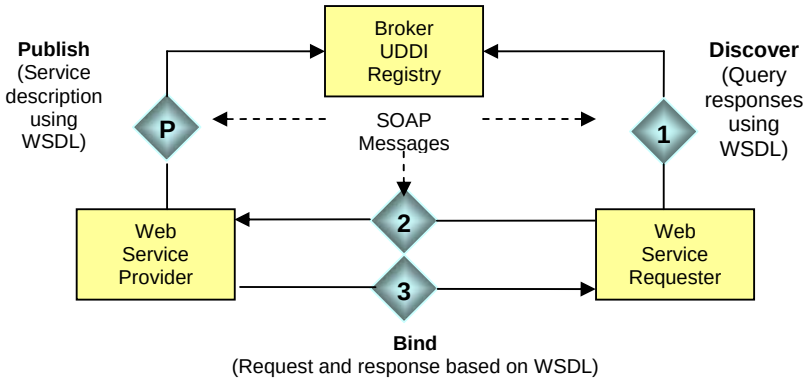

Fig. 2 – SOA based on Web Services

After a Web services is built by a Web service provider and published to companies' or broker's UDDI registry (P in the diagram) any invocation via the network can occur. A typical transaction flow comprises 3 steps. The Web service requester searches the UDDI registry and finds the WSDL description with the location of the desired service (1). The Web service requester then connects to the

Web service provider via the SOAP (Simple Object Access Protocol) to obtain and activate the Web service during several interactions (2). After execution of the Web service the deliverables are sent back again via SOAP to the Web service requester (3), where they are processed in the requester's application.

3. Advantageous aspects of SOA

The most important new aspect about Web services and SOA is its ubiquitous support by all major vendors. Most likely this is not at least due to some convincing advantages such as:

- Flexibility in new software design.
- Reuse of business components in networks.
- Interoperability and integration capability.
- Ease of assembling new business processes.

Recent IT developments have pushed SOA forward. For instance with personalized, portal-style user interfaces over multiple wired and wireless channels an increasing number of solutions require the reuse of application components. Different users such as customers, employees, managers using many devices (laptops, PDAs, smart phones) in different situations (home, office, hotel) all may request access to the same set of business applications. The loosely coupled SOA provides the natural basis for reuse of applications to multiple categories of clients. Thus, the transition to multi-client and multi-channel applications pushes forward the Web services based software design.

No doubt, SOA will imply more competition and innovation of application development. Best-of breed business components can be encapsulated as autonomous Web services and published to UDDI registries as reusable services with the accessibility of the whole world. This will not only become an excellent business opportunity for smart software start-ups, but also established vendors will tend to modularize their application packages and offer components as Web services via the Internet as modularized functions or processes to more customers.

Another fundamental need is the easier interoperability of application platforms of different vendors such as IBM, Microsoft and SAP as well as the integration of enterprise applications from different vendors or with home-grown applications. Many projects have created programmatic interfaces to wrap legacy and other existing external functionality for assembly into heterogeneous composite transactions. Composition and integration of existing and new real-time transaction patterns are a natural fit of SOA.

4. Limits and critical aspects

SOA is neither a universal remedy for the multiple existing shortcomings in today's mix and match IT-architectures nor is it the solution for all upcoming challenges. In the following, areas where SOA is not of benefit as well as some present critical aspects are mentioned:

- Applications, of which business logic components are used in a closed application domain or will be neither reused nor changed do not benefit from SOA because of the additional design and development effort.
- SOA requires an environmental framework i.e. a state-of-the-art application platform, such as Microsoft .NET, SAP NetWeaver, IBM WebSphere or BEA WebLogic and unfortunately there is not yet achieved a seamless platform interoperability and tool independency.
- The most critical aspects of SOA and Web services are still some pending security issues. A more or less extensive framework of security-standards has been worked out, not yet finished, and applicable products are available on the market now. But considering mission critical processes with embedded external Web services via the Internet, the existing security measures are not sufficient for the most cases.
- Furthermore SOA and Web services deployment are still critical where an application requires the transactional handling of sequential request-response exchanges. In many cases existing applications cannot be wrapped as Web services that match the high level of ACID (Atomicy, Consistency, Isolation, Durability) transactional processing. Extended transactions therefore will become very important in the design phase. They define how non- ACID transactions should be managed acceptable, if any events cause that a transaction cannot be finished successfully. As an example, long- running activities may be structured as many independent, short-duration transactions to form a logical long-running transaction.
- Another critical aspect to be considered if deploying Web services are the coordinated collaboration of Web services with old applications as well as the provisioning of complex functionality achieved by the aggregation of various Web services. Business processes usually comprises various coordinated activities, where contexts, dependencies and events have to be regarded. This requires a coordination framework with appropriate coordination protocols to address those issues. Specifications concerning such protocols exist already, but there is little experience how these challenges can be successful responded.
- Finally there are some issues concerning the deployment of published Web services in mission critical enterprise application, such as trust, guaranteed availability and quality levels, and the ability to negotiate contracts with sufficient flexibility.

5. Frameworks and standardization

Frameworks, services and standards concerning Web services and SOA have been pushed mainly by Microsoft and IBM. Other software vendors such as Ariba, SAP, BEA, and Sun as well as security specialists, such as RSA Security and Verisign have supported and contributed to those standardization works. Bodies like OASIS (Organization for the Advancement of Structured Information Standards), and the standardization organization W3C (World Wide Web Consortium) are continuing to drive these technologies until a standard is concluded.

An important step of the Web services standardization development was Microsoft's publication of a specification called Global XML Web services Architecture (GXA) at the end of 2001. This specification was then extended with the assistance of other vendors during the course of the last years with concrete standards and protocols being added. IBM and Microsoft formed the Web services Interoperability Organization (WS-I) in 2002 in a bid to ensure the interoperability of Web services implementations from different vendors.

The aim of the WS-I organization is to link various Web services approaches with each other. WS-I provides guidelines, best-practice examples, tools and test environments for ensuring that Web services can communicate with each other across platform and vendor boundaries. Furthermore, WS-I provides a so-called basic profile, which contains implementation guidelines and recommendations as to how interoperable Web services can be created on the basis of the core technologies and protocols.

The standardization is an ongoing process with frequently extended versions and additional discovered new fields where standardization is needed. Examples are Web services security, policy, provisioning, long-running transaction, federation, composition, orchestration and management. Some of the mentioned standard proposals are vendor proposals but not yet taken up by standardization organizations (e.g. WS-Trust). Unfortunately concerning standards, involved players have different business interests that cause overlapping or disagreement of standard proposals preventing unified deployment and interoperability.

As a present status, the framework of specifications, standards, protocols and extended services can be summarized as shown in the figure below:

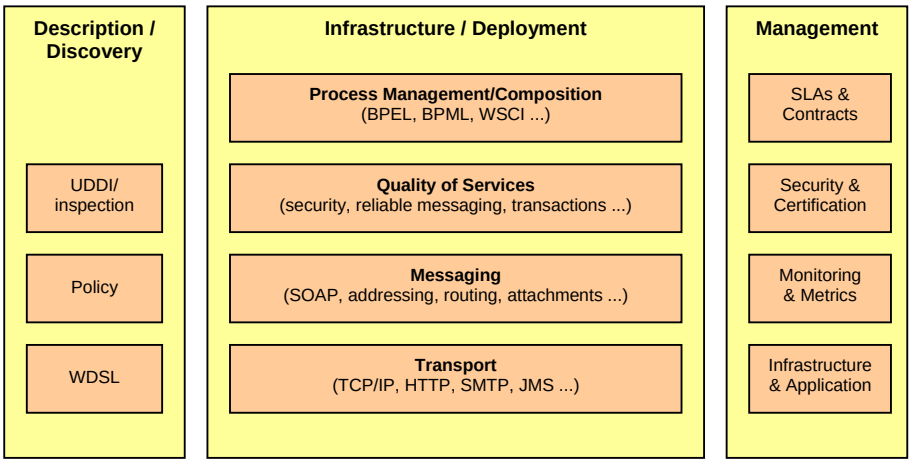

Fig. 3 – SOA and Web Services Framework

6. Complexities of testing web services

Currently, Web Services are generally managed using tools supplied by platform vendors. This makes testing a vendor-dependent activity. Service management includes configuration, accounting, QoS, policy and fault identification, containment, and repair.

Passive or active testing mechanisms are widely used as tools for fault detection. Active testing techniques require the generation and the application of test cases in order to detect faults. Passive testing techniques use observers to track the interaction between the entities being tested. Observers can be inserted directly on-line in the data flow or can be off-line and with access to log files.

Current proposed Web Services strategies are either based on active testing techniques or require Web Services to participate in their management through the support of a management interface to active testers. These solutions assume that a Web Service will participate in its management by providing specific interfaces that are based on active testers.

Requiring Web Services to provide their own management interfaces adds complexity to Web Services architecture and may impact performance. In addition, there are security risks associated with these interfaces.

7. Conclusions

The use of Web Services in IT is becoming more relevant as the key technology for enabling the foundation of a loosely coupled, language-neutral, platform-independent way to link applications across multiple organizations.

Despite the heterogeneity of the underlying platforms, Web Services enhance interoperability and are thus able to support business applications composed of chains of Web Services. Interoperability among heterogeneous systems is a key promise of Web Service technology and therefore notions such as Web Service composition and technologies like workflow systems are being investigated and developed.

Interoperability and security play an important role in positioning Web Services as the industry choice for realizing Service Oriented Architectures.

B I B L I O G R A P H Y

[1] *D.A. Menascé*, QoS Issues in Web Services, IEEE Internet Computing, vol. 6, no. 6, 2002.
[2] *E.Bertino, A.C. Squicciarini, L.Martino, F. Paci*, An Adaptive Access Control Model for Web Services, International Journal of Web Service Research, (3), 27-60 July-September 2006.
[3] http://sys.sbs.de/e/techn_direction/index.htm
[4] *Nadalin, A., Kaler C., Hallam-Naker, P., Monzillo R.*, Web Services Security: SOAP Message Security 1.0, (WS-Security 2004), OASIS, 2004.
[5] *Monzillo R., Kaler C., Nadalin A., Hallam-Naker, P.*, Web Services Security: Rights Expression Language (REL) Token Profile 1.1, OASIS, February 2006.
[6] *Demchenko, Y.*, Web Services and Grid Security Vulnerabilities and Threats Analysis and Model, Grid Computing Workshop, 2005.
[7] *D. Tidwell, J. Snell, P. Kulchenko*, Programming Web Services with SOAP, Ed. O'Reilly, 2001.
[8] *Douglas K. Barry*, Web Services and Service Oriented Architecture, Ed. Morgan Kaufmann 2003.