Developing Web Services and Challenges

Anshu Kumar

Student, PIT, Parul University

Shivangi Patel

Asst. Prof. PIT, Parul University

Department of Computer Science & Engineering

anshuukumar889@gmail.com

***Abstract –***Web services have recently attracted considerable interest from vendors and researchers. They provide a flexible solution for application integration by leveraging open standards and existing Internet protocols. The adoption of WSDL, SOAP, and UDDI has made web services increasingly common in web applications. Despite their growing popularity, current web service designs encounter persistent challenges, particularly regarding security. This essay will explore these issues, as we believe that addressing them is crucial for the future success of web services. Additionally, we anticipate significant advancements in semantic Grid services, which could lead to more intelligent and efficient data processing. The integration of semantic technologies may also enhance interoperability among diverse systems, paving the way for more seamless interactions.

*Keywords – Web Services, Data processing, Semantic Grid Services, Simple Object Access Protocol, Open Standards, Application Integration.*

Introduction

A web service is essentially an autonomous software component identified by a specific URI (Uniform Resource Identifier), which utilizes Internet technology, particularly the Web. According to the fundamental concept, web services have both providers and (ideally) consumers or subscribers. The service provider is responsible for developing the service, creating a specification, and publishing it in a directory for potential clients to access. Users can search this service directory to find available services. To locate a single service that meets their needs or a combination of services that can work together effectively, service clients or users often browse through such directories.

It is obviously clear that web services need to be interoperable. Furthermore, they should be compatible with all web browsers and operate independently of the operating system on which they are installed.

Service engines should be capable of communicating with each other, regardless of their programming languages. To achieve this, web services are often built on standards, with the most popular ones currently being XML-based specifications such as SOAP(Simple Object Access

Protocols such as UDDI (Universal Description, Discovery, and Integration) and WSDL (Web Services Description Language) are commonly used, although the landscape is continually evolving. The fundamental steps for publishing and utilizing a software component developed as a web service are as follows:

The provider of a web service registers their service in a UDDI directory, which acts as a central repository of metadata for all registered services and maintains information about them.

Examples of metadata include the author, service category, and technical specifications. UDDI data is typically stored across multiple remote servers, and UDDI defines a query language, authorization for authors, and replication methods. This setup ensures that services can be quickly located in response to a request, with the request being fulfilled by providing instructions on how to access and utilize the service at the provider's server.

A WSDL file, which is distinct from the UDDI directory, provides the guidelines for communication between clients and web services. Instead of being referenced by the details a client requests from the UDDI directory, this document specifies the requirements for input and output, ensuring seamless and effective interaction.

The WSDL file specifies both the data and the communication protocol. To send and receive SOAP messages with the actual web service, a proxy is created using the WSDL specification[1,2].

In recent years, ad hoc methods have been utilized in business-to-business applications to leverage the fundamental architecture of the Internet.

Recently, web services have been developed on top of existing web protocols and open XML standards, emerging as a systematic and scalable framework for application-to-application interaction.

A web service is a new type of web application. These applications are self-contained, self-describing, and modular, allowing them to be published, discovered, and utilized.

Web services can be utilized across various platforms to execute complex business operations or simply handle straightforward information requests.

Once deployed, a web service can be discovered and utilized by other applications or web services.

The primary advantage of using web services is the ability to rapidly develop applications by utilizing loosely coupled, reusable software components.

This has significant implications for both technological and commercial applications. Instead of being sold as packaged products, software could be delivered and paid for as an ongoing stream of services. Automatic and dynamic interoperability between systems can be achieved to complete business tasks. The Internet enables fully decentralized distribution of business services, allowing them to be accessed by a wide audience. This shift allows businesses to focus on the value of their services and other critical tasks, rather than dealing with the complexities, high costs, and low quality of traditional software integration.

The Internet will evolve into a universal platform for communication between businesses and individuals, facilitating various commercial operations and providing value-added services. Barriers to entry will be lowered, making new offerings and markets more accessible to small and medium-sized businesses. Dynamic businesses and value chains will become feasible and potentially essential for maintaining competitive advantages.

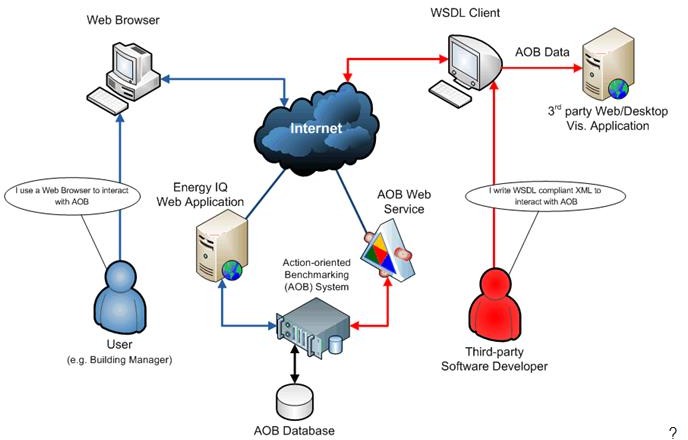


Fig. (1): Web Services Map.

Web services difficulties and challenges

Technologies such as SOAP, WSDL, and UDDI are essential for enabling web services. However, to fully meet the needs of business applications, these technologies are not without flaws. This section will discuss the three main issues and potential research directions to enhance the current technology.

Let's use a simple travel scenario to illustrate the security issues associated with web services. The web services framework requires more than three components to appropriately interrelate in order to complete the travel scenario.

At a minimum, we must ensure that operations such as electronic check-ins are conducted in a secure environment and that messages are reliably transmitted to their intended destinations. Why is additional security necessary when technologies like Kerberos, HTTPS, and S-MIME are already available? The answer lies in the difference between end-to-end and single-hop usage. Business messages often originate in one application and then transition to another.

In order to secure (for confidentiality) a direct connection between two machines, mechanisms like secure sockets layer are excellent.

No assistance is provided if the message must cross multiple connections.

Fundamentally, the security issues that could impact Web services are the same ones that have troubled traditional Web-based systems.

Numerous of these were thoroughly covered in[1,2]. Here, we offer the following summary of the current scenario: Enterprise adoption of Web services depends on security, yet

The current Web services architecture does not adhere to fundamental security standards. Since Web services involve the exchange of messages, protecting these exchanges is a crucial consideration in their development and use.

Security in the context of Web services refers to the recipient's ability to verify the integrity of the message and ensure it has not been altered. The communication should be delivered confidentially, with the recipient aware of the sender's identity and whether the sender is authorized to perform the requested operation. Typically, messages are encrypted to meet these requirements.

Web services facilitate communication between various systems, both internal and external, using HTTP ports. This makes application servers susceptible to "application-level" attacks. To enhance message security, standards like WS-Security have been established, along with numerous initiatives from major manufacturers and PKI providers. These efforts focus on implementing digital signatures for XML messages and transactions. Despite these advancements, "application-level" attacks continue to pose a significant threat.

Web services typically rely on HTTP-based application servers to enable communication between various systems, both internal and external. This reliance creates an "application-level" attack surface. To enhance message security, standards such as WS-Security have been established, along with numerous initiatives from major manufacturers and PKI providers. These efforts aim to implement digital signatures for XML messages and transactions. Despite these advancements, "application-level" attacks continue to be a significant concern.

The OASIS standard Security Assertion Markup Language (SAML) provides a method for partner programs to exchange user authentication and authorization data.

Essentially, all major vendors' e-commerce products include single sign-on (SSO) functionality. In the absence of a standardized protocol for transmitting authentication data, companies often use cookies in HTTP communication to implement SSO. With the advent of SAML, the same data can be encapsulated in XML in a standardized way, eliminating the need for cookies and enabling interoperable SSO.

Composition Difficulties and Solutions

To support complex business interactions, greater levels of business functionality are required. Business transactions often involve lengthy execution processes and various relationship interactions. To successfully deploy and use these services, we need to express business processes and service states, as well as develop service compositions (complex aggregations) in a consistent and methodical way. Several solutions have been proposed for this task, including Web Services Flow Language, XLANG, and BPEL4WS.

Various terminologies have been used in the industry to describe how components can be connected to construct intricate business processes. Workflow management systems and document movement systems within IT organizations have been around for a while. These systems often involve human interaction and can include people, systems, or applications. Business Process Management Systems (BPMS) have also been used to help organizations create top-down process design models that include various integration tasks, such as integration with legacy systems. BPMS typically cover the entire lifecycle of a business process, including modeling, execution, monitoring, management, and optimization. These systems ensure that business processes are streamlined and efficient. They also provide tools for continuous improvement and adaptation to changing business needs.

The composition of Web services in a process is now referred to by terms like "Web services composition" and "Web services flow." More recently, the terms "orchestration" and "choreography" have been used. Orchestration refers to the process of defining how Web services can communicate with each other at the message level, including the business logic and sequencing of interactions. These interactions, which may involve multiple applications, companies, or both, result in a long-lasting, transactional, multi-step process model.

.

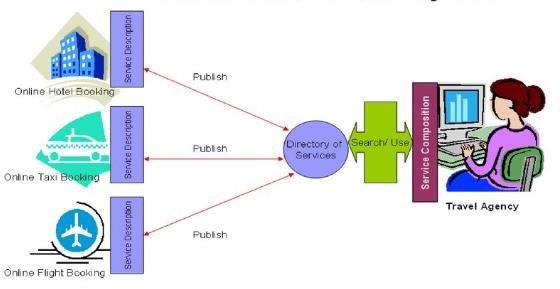


Fig.(2): Web Services Composition Sample.

Solution for Some Difficulties and

Challenges

Current Web services technology primarily offers a syntactical solution, lacking a semantic component. In WSDL, a Web service is defined by specifying the input data it requires and the output it produces. To unlock their full potential beyond simple business application connections, Web services need the capability to orchestrate themselves into more complex services.

To achieve this, we need methods to integrate distinct Web services into distributed, higher-level services. The Web Service Flow Language (WSFL) has made initial strides by allowing the sequencing of separate services. WSFL enables users to select and order Web services for combination. However, to allow software agents to discover, understand, and integrate services, we still need a framework that semantically characterizes these services.

Many researchers believe that the Semantic Web concept, which aims to enable computers to understand Web content without ambiguity, addresses this issue. Tim Berners-Lee, the creator of the Semantic Web project, envisions a web that can be processed by machines. The Semantic Web is mainly supported by the research-focused members of the Web community.

Conversely, industrial players like Microsoft, IBM, and BEA have significantly influenced the development of Web Services due to business interests. In his opening address at the Twelfth International World Wide Web Conference, the director of the World Wide Web Consortium discussed how to integrate these two approaches. The primary focus of Web development should be on cooperation rather than competition. According to Berners-Lee, addressing urgent technology needs is crucial, while the Semantic Web could experience exponential expansion in the future

.

Grid services

To handle complex Web service applications, a robust computing infrastructure is essential, and grid computing provides this foundation. The Open Grid Services Architecture (OGSA) represents an evolution toward a grid system architecture that leverages Web services concepts and technologies.

OGSA creates a distributed system framework by building on the Open Grid Services Infrastructure (OGSI), integrating key Grid technologies with Web services mechanisms. A Grid service instance adheres to standards for managing its lifecycle, discovering its characteristics, and handling notifications. These standards are defined through Web Service Definition Language (WSDL) interfaces, extensions, and behaviors. Grid services facilitate the controlled management of distributed, often long-lived services, which are crucial for complex distributed applications.

Additionally, OGSI introduces standard interfaces for creating and locating Grid services, ensuring a consistent approach to developing and managing these services.

Client applications, which may be located remotely, can access grid service instances using a Grid Service Handle (GSH) and a Grid Service Reference (GSR). Think of a GSH as a persistent network pointer to a specific Grid service instance. However, the GSH alone does not provide enough information for a client to access the service instance.

The client must resolve the GSH into a GSR, which contains all the necessary details to reach the service instance. This resolution process ensures that the client has the correct and complete information to interact with the service. By using GSR, clients can establish direct communication with the service instance, facilitating efficient and reliable interactions. This mechanism is crucial for maintaining the integrity and accessibility of distributed services. Furthermore, it allows for seamless integration of services across different platforms and environments. This approach enhances the scalability and flexibility of the system, making it adaptable to various business needs. Ultimately, the use of GSRs contributes to a more robust and resilient service infrastructure.

OGSI does not mandate a specific implementation architecture for service providers. A crucial concern is how client programs will use OGSI interfaces. OGSI leverages a key feature of the Web services architecture: the use of WSDL to specify protocol bindings, encoding schemes, messaging schemes, and other aspects for a given Web service.

Recently, many international conferences, such as SC2002 (the international conference for high-performance computing) and Grid2002, have featured discussions on grid services and communications. Semantic Grid services are another intriguing topic. The current state of Grid development is reminiscent of the early days of the Web, with small deployments primarily driven by scientific community enthusiasts, growing standards, and limited commercial uptake. This situation is comparable to the Semantic Web's development. Meanwhile, machine-to-machine communication (XML) on the Web has replaced machine-to-human communication (HTML), providing the precise infrastructure needed by the Grid. The parallels suggest that Grid adoption may follow a similar exponential growth pattern as the Web.

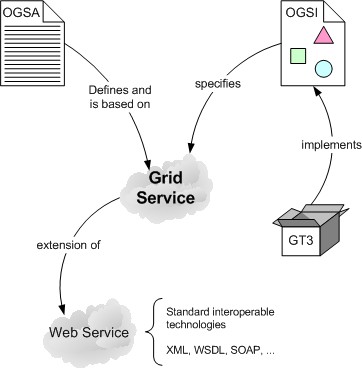


Fig.(3): Grid service.

Conclusion

We have discussed the difficulties and challenges of Web services as a new Web technology. In this research, Web services were examined from various perspectivessuch as semantics and service composition. These elements are essential for the successful deployment and development of Web services. We conclude by providing a glimpse of Grid services, believing that the integration of Grid and Semantic Web services will represent a significant technical advancement.

References

1. Keidl, M., Kreutz, A., Kemper, A., Kossmann, D. (2002). A Publish & Subscribe Architecture for Distributed Metadata Management. In Proc. 18th IEEE Int. Conf. on Data Engineering

(ICDE), San Jose, CA, February 2002. IEEE Computer Society, pp. 309–320.

1. Pilioura, T., A. Tsalgatidou (2001). E-Services: Current Technology and Open Issues. In Proc. 2nd Int. Workshop on Technologies for E-Services, Rome, Springer-Verlag, Berlin, pp. 1–15.
2. D. Fensel, C. Bussle, Web services modeling framework, Electron. Commerce Res. Appl. 1 (2002) 113–137.
3. D. Farber, Balancing security and liberty, IEEE Internet Comput. 5 (6) (2001) 96–96.
4. G. Goth, Securing the internet against attack, IEEE Internet Comput. 7 (1) (2003) 8–10.
5. A. Householder, K. Houle, C. Dougherty, Computer attacktrends challenge Internet security, IEEE Comput. 35 (4) (2002) 5–7.
6. G. McGraw, Managing software security risks, IEEE Comput. 35 (4) (2002) 99–101.
7. W.T. Polk, N. E Hastings, A. Malpani, Public key infrastructures that satisfy security goals, IEEE Internet Comput. 7 (4) (2003) 60–67.
8. D. Scott, R. Sharp, Abstracting application-level web security, in: Proceedings of the Eleventh International World Wide Web ConferencE (WWW), Honolulu, Hawaii, USA, 7–11 May 2002.
9. D. Scott, R. Sharp, Developing secure web applications, IEEE Internet Comput. 6 (6) (2002) 38–45.
10. E. Wales, Web services security, Comput. Fraud Security 18 (1) (2003) 15–17.
11. [http://www.oasis-open.org/committees/security/.](http://www.oasis-open.org/committees/security/) [13][http://www3.ibm.com/software/solutions/webservices/pd](http://www3.ibm.com/software/solutions/webservices/pdf/)

[f/](http://www3.ibm.com/software/solutions/webservices/pdf/) WSFL.pdf.

1. S. Thatte, XLANG-Web Services for Business Process Design, [http://www.gotdotnet.com/team/xml](http://www.gotdotnet.com/team/xml%20wsspecs/xlangc/) [wsspecs/xlangc/](http://www.gotdotnet.com/team/xml%20wsspecs/xlangc/) default.htm.
2. <http://www-106.ibm.com/developerworks/webservices/> library/ws-bpel/.
3. L. Juhnyoung, J. Yang, J. Chung, Winslow: A Business Process Management System with Web Services. IBM Research Report, November 2002.
4. L. Frank, D. Roller, M. Schmidt, Web services and business process management, IBM Syst. J. 41 (2) (2002) 198–212.
5. N. Gibbins, S. Harris, N. Shadbolt, Agent-based Semantic Web Services, WWW 2003, Budapest, Hungary, 20–24 May 2003.
6. J. Hendler, Agents and the semantic web, IEEE Intell. Syst. 16 (2) (2001) 30–37.
7. S. McIlraith, T.C. Son, H. Zeng, Semantic web services, IEEE Intell. Syst. 16 (2) (2001) 46–53.
8. I. Foster, C. Kesselman, J. Nick, S. Tuecke, The physiology of the grid: an open grid services architecture for distributed systems integration. Open Grid Service Infrastructure WG, Global Grid Forum, June222002[.http://www.globus.org/research/papers/ogs](http://www.globus.org/research/papers/ogsa.pdf) [a.pdf.](http://www.globus.org/research/papers/ogsa.pdf)
9. S. Tuecke, K. Czajkowski, I. Foster, J. Frey, S. Graham,

C. Kesselman et al. (Eds.), Open Grid Services Infrastructure(OGSI), 2003, [http://www-](http://www-unix.globus.org/toolkit/draft-ggfogsi-) [unix.globus.org/toolkit/draft-ggfogsi-](http://www-unix.globus.org/toolkit/draft-ggfogsi-) gridservice-33 2003-06-27.pdf.

1. I. Foster, C. Kesselman, J. Nick, S. Tuecke, Grid services for distributed system integration, IEEE Comput. 35 (6) (2002)37–46.
2. D. De Roure, N.R. Jennings, N.R. Shadbolt, The Semantic Grid: A Future e-Science Infrastructure, 2003,[http://www.semanticgrid.org.](http://www.semanticgrid.org/)