International Journal of Computing Science and Information Technology, **2013, Vol. 01 (01), 33-37**
ISSN: 2278-9669, January 2013 (http://ijcsit.org)

Data Center –Based, Service Oriented Architecture (SOA) in Cloud Computing

Abu Sarwar Zamani
Dept. of Computer Science, College of Science & Humanity, Al Quwaiyyah, Shaqra University
Kingdom of Saudi Arabia
Sarwar_zamani @ yahoo.com

Mohammad Jawed Miandad
Dept. of Mathematics, Al Jouf University, Kingdom of Saudi Arabia
miandad07 @ gmail.com

Shakir Khan
Dept. of E-Learning, King Saud University, Kingdom of Saudi Arabia
shakirkhan2006 @ gmail.com

Abstract

SOA and cloud computing are complementary activities; both will play important roles in IT planning for senior leadership teams for years to come. Cloud computing does not replace SOA or the use of distributed software components as an integration technology. SOA inspired componentization eff orts, where soft ware leverages other network based soft ware using standards-based interfaces. In this paper we discussed what it takes to build a cloud network, evolution from the managed service provider model to cloud computing roles as well as importance of collaboration, service-oriented architectures as an intermediary step and the basic approach to data center-based SOAs, and the role of open source software in data centers and where and how it is used in cloud architecture.

Keywords:: Cloud Computing, Cloud N/w, SOA and Data Center

1. Introduction

Cloud computing is the next generation in computation. Maybe Clouds can save the world; possibly people can have everything they need on the cloud. Cloud computing is the next natural step in the evolution of on-demand information technology services and products. The Cloud is a metaphor for the Internet, based on how it is depicted in computer network diagrams, and is an abstraction for the complex infrastructure it conceals. It is a style of computing in which IT-related capabilities are provided "as a service", allowing users to access technology-enabled services from the Internet (i.e., the Cloud) without knowledge of, expertise with, or control over the technology infrastructure that supports them.

This is a technology that uses the internet and central remote servers to maintain data and applications. Cloud computing allows consumers and businesses to use applications without installation and access their personal files at any computer with internet access. This technology allows for much more efficient computing by centralizing storage, memory, processing and bandwidth.

A simple example of cloud computing is Yahoo email, Gmail, or Hotmail etc. You don't need software or a server to use them. All a consumer would need is just an internet connection and you can start sending emails. The server and email management software is all on the cloud (internet) and is totally managed by the cloud service provider Yahoo, Google etc. The consumer gets to use the software alone and enjoy the benefits. The analogy is, 'If you need milk, would you buy a cow?' All the users or consumers need is to get the benefits of using the software or hardware of the computer like sending emails etc. Just to get this benefit (milk) why should a consumer buy a (cow) software /hardware?

Cloud computing is broken down into three segments: "application" "storage" and "connectivity." Each segment serves a different purpose and offers different products for businesses and individuals around the world. In June 2011, a study conducted by Version One found that 91% of senior IT professionals actually don't know what cloud computing is and two-thirds of senior finance professionals are clear by the concept, highlighting the young nature of the technology. In Sept 2011, an Aberdeen Group study found that disciplined companies achieved on average a 68% increase in their IT expense because cloud computing and only a 10% reduction in data center power costs.

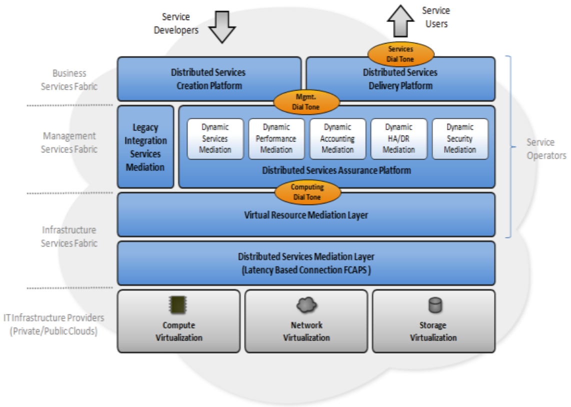

Figure: 1. Architecture Model for Next Generation Cloud Computing Infrastructure

2. How SOA Contributes to Cloud Computing

SOA means more than just thinking in terms of services. It implies a particular way of constructing enterprise business systems using IT. An SOA system includes software services and an SOA platform and supporting infrastructure, just as a column in a classic Greek temple has a capital, a shaft and a base. The software services are supported by the SOA platform, which typically includes components such as an enterprise service bus (ESB) and a service registry. There are established standards for the components of the SOA platform, such as the Web services description language (WSDL) and WS-messaging, and there are commercial off-the-shelf products that support these standards. The SOA platform is in turn supported by the enterprise IT infrastructure of computers, data stores and networks. These elements of the SOA style also relate to the different kinds of Cloud service. The software services relate directly to SaaS, the infrastructure relates directly to IaaS, and the SOA platform relates, but not yet directly, to PaaS.

From the consumer's perspective, the SaaS provider's applications are software services that can replace some internally-provided software services and integrate with others, using the internal SOA platform extended over the Internet. This implies no radical architectural change. Decisions on use of externally-provided services can be made on a case-by-case basis, depending on cost, ease of deployment and other factors for the particular services concerned.

The software deployed over IaaS can include part or all of the SOA platform, with some or all of the software services that run on it. Again, there is no radical architectural change. The decision depends on the same factors as a decision to set up an in-house data center: equipment and support costs, return on investment, security, capability for disaster recovery, and so on.

The correspondence is however not quite as close for PaaS. Today, the popular Cloud platform services are running mostly at the operating system and programming language level rather than at the level of the SOA platform. They support standards such as Linux and Java, rather than WS-Messaging and WSDL. Of course, a SOA platform can be implemented on top of Linux and Java, for example, more easily than directly on the raw infrastructure, but this is not as good as being able to obtain the SOA platform itself as a Cloud service. To be really useful for SOA, Cloud platforms should include enterprise service buses, service registries and other SOA platform components—in other words, "SOA as a Service."

3. Approach to a Data Center-Based SOA

A service-oriented architecture is essentially a collection of services. A service is, in essence, a function that is well defined, self-contained, and does not depend on the context or state of other services. Services most often reflect logical business activities. Some means of connecting services to each other is needed, so services communicate with each other, have an interface, and are message-oriented. The communication between services may involve simple data passing or may require two or more services coordinating an activity. The services generally communicate using standard protocols, which allows for broad interoperability. SOA encompasses legacy systems and processes, so the effectiveness of existing investments is preserved. New services can be added or created without affecting existing services. Service-oriented architectures are not new. The first service-oriented architectures are usually considered to be the Distributed Component Object Model (DCOM) or Object Request Brokers (ORBs), which were based on the Common Object Requesting Broker Architecture (CORBA) specification. The introduction of SOA provides a platform for technology and business units to meet business requirements of the modern enterprise. With SOA, your organization can use existing application systems to a greater extent and may respond faster to change requests. These benefits are attributed to several critical elements of SOA:

a. Free-standing, independent components
b. Combined by loose coupling
c. Message (XML)-based instead of API-based
d. Physical location, etc., not important

3.1 Planning for Capacity

It is important to create a capacity plan for SOA architecture. To accomplish this, it is necessary to set up an initial infrastructure and establish a baseline of capacity. Just setting up the initial infrastructure can be a challenge. That should be based on known capacity requirements and vendor recommendation for software and hardware. Once the infrastructure is set up, it is necessary to establish a set of processing patterns. These patterns will be used to test capacity and should include a mix of simple, medium, and complex patterns. They need to cover typical SOA designs and should exercise all the components within the SOA infrastructure.

3.2 Planning for Availability

Availability planning includes performing a business impact analysis (BIA) and developing and implementing a written availability plan. The goal is to ensure that system administrators adequately understand the criticality of a system and implement appropriate safeguards to protect it. This requires proper planning and analysis at each stage of the systems development life cycle (SDLC). A BIA is the first step in the availability planning process. A BIA provides the necessary information for an administrator to fully understand and protect systems. This process should fully characterize system requirements, processes, and interdependencies that will determine the availability requirements and priorities. Once this is done, a written availability plan is created. It should define the overall availability objectives and establish the organizational framework and responsibilities for personnel. Management should be included in the process of developing availability structure, objectives, roles, and responsibilities to support the development of a successful plan.

3.3. Planning for SOA Security

The foundations of SOA security are well known and are already widely used in the IT industry. SOA practitioners have come to realize they also must understand these foundations in order to provide adequate security for the systems being developed. The foundations include public key infrastructure (PKI), the common security authentication method Kerberos, XML (Extensible Markup Language) encryption, and XML digital signatures. Three main areas of concern are widely accepted as part of the SOA security arena. First, message-level security provides the ability to ensure that security requirements are met in an SOA environment, where transport- level security is inadequate because transactions are no longer point to- point in SOA. Second, Security-as-a-Service provides the ability to implement security requirements for services. Third, declarative and policy- based security provides the ability to implement security requirements that are transparent to security administrators and can be used to quickly implement emerging new security requirements for services that implement new business functionalities.

3.4. Message level security

The OASIS set of WS-Security standards addresses message-level security concerns. These standards are supported by key vendors including IBM, Microsoft, and Oracle. The standards provide a model describing how to manage and authenticate message exchanges between parties (including security context exchange) as well as establishing and deriving session keys. The standards recommend a Web service endpoint policy describing the capabilities and constraints of the security and other business policies on intermediaries and endpoints including required security tokens, supported encryption algorithms, and privacy rules. Furthermore, a federated trust model describing how to manage and broker the trust relationships in a heterogeneous federated environment, including support for federated identities, is described. The standards include a Web service trust model that describes a framework for trust models that enables Web services to operate securely. There is also an authorization model describing how to manage authorization data and authorization policies. Finally, the standards include a Web service privacy model describing how to enable Web services and requesters to state subject privacy preferences and organizational privacy practice statements.

3.5. Security as a Service (SAAS)

Security-as-a-Service can be accomplished by collecting an inventory of service security requirements throughout the enterprise architecture (EA) and specifying the set of discrete security services that will be needed for the enterprise. Next, the organization must complete the process of designing and implementing these security services as services themselves. Often, a toolkit approach can help specify the set of typical security services that may be used to provide most of the requirements and accelerate the establishment of Security-as-a-Service in an organization.

3.6. Declarative & Policy based security

Implementation of declarative and policy-based security requires tools and techniques for use at the enterprise management level and at the service level. These tools and techniques should provide transparency for security administrators, policy enforcement, and policy monitoring. When policy violations are detected, alerts should be issued. Traceability of such violations, both for data and users, should be included as a critical element.

4. Open Source Software in Data Centers

The Open Source Initiative uses the Open Source Definition to determine whether a software license can truly be considered open source. The definition is based on the Debian Free Software Guidelines, [8] written and adapted primarily by Bruce Perens. [9] Under Perens's definition, the term open source broadly describes a general type of software license that makes source code available to the public without significant copyright restrictions. The principles defined say nothing about trademark or patent use and require no cooperation to ensure that any common audit or release regime applies to any derived works. It is considered as an explicit "feature" of open source that it may put no restrictions on the use or distribution by any organization or user. It forbids this, in principle, to guarantee continued access to derived works even by the major original contributors. Over the past decade, open source software has come of age. There has always been a demand for software that is free, reliable, and available to anyone for modification to suit individual needs. Open source distributions such as Red Hat, OpenSuSE, and BSD, coupled with open source applications such as Apache, MySQL, and scores of others have long been used to power databases, web, email, and file servers. However, something that has as much impact as the applications used in a data center has caused many implementers to hesitate to adopt open source software—until now. Recently, more than just a few users have become strong advocates that open source can and does work in the data center environment. In an online article, Robert Wiseman, chief technology officer at Sabre Holdings (a travel marketing and distribution technology company in Southlake, Texas, that uses open source software on over 5,000 servers) stated:

It's true that with open-source products, users generally forfeit the security of professional support teams to help resolve their problems quickly. But in our environment, we almost always purchase support for our open-source products from high-quality vendors. This, of course, reduces some of the cost advantages of using open source, but the advantages are big enough that there's still plenty left over, and the security we get from a service contract lets us sleep better at night. [10] Sabre Holdings uses an enterprise service bus for message transformation, routing, and other tasks. An enterprise service bus (ESB) refers to a software architecture construct that is typically implemented by technologies seen as a type of middleware infrastructure. ESBs are usually based on recognized standards and provide fundamental services for complex architectures via an event-driven and standards-based messaging engine (called the bus since it transforms and transports the messages across the architecture).

One example of open source ESB, Apache Synapse, is an easy-to-use and lightweight ESB that offers a wide range of management, routing, and transformation capabilities. With support for HTTP, SOAP, SMTP, JMS, FTP, and file system transports, it is considered quite versatile and can be applied in a wide variety of environments. It supports standards such as WS-Addressing, Web Services Security (WSS), Web Services Reliable Messaging (WSRM), efficient binary attachments (MTOM/XOP), as well as key transformation standards such as XSLT, XPath, and XQuery. Synapse supports a number of useful functions out of the box, without programming, but it also can be extended using popular programming languages such as Java, JavaScript, Ruby, and Groovy.

Another example is a project called Open ESB, which implements an enterprise service bus runtime with sample service engines and binding components. Open ESB allows easy integration of enterprise applications and web services as loosely coupled composite applications. This allows an enterprise to seamlessly compose and recompose composite applications, realizing the benefits of a true service-oriented architecture. Today, most users of open source agree that these products have now reached a level of maturity equal to and, in some cases, better than their commercial counterparts. Open source products have forced commercial vendors to compete on price and quality of service. Because open source code is open and transparent, developers can troubleshoot problems and learn how other developers have addressed issues. Users gain the freedom to use these products across their organizations, all over the world, without worrying about tracking client licenses.

5. Conclusion

Cloud Computing describes a system architecture period. This particular architecture assumes nothing about the physical location, internal composition or ownership of its component parts. This is about scalable web applications and data processing needed to make apps interesting. In this topic we discussed to build a cloud network, evolution from the managed service provider model to cloud computing and the role and importance of collaboration, service-oriented architectures as an intermediary step and the basic approach to data center-based SOAs, and the role of open source software in data centers and where and how it is used in cloud architecture.

REFERENCES

[1] Cloud Confusion Amongst IT Professionals. VersionOne (June 6, 2011).

[2] Business Adoption of Cloud Computing. AberdeenGroup (Sept 9, 2011).

[3] Abu Sarwar Zamani, Md. Mobin Akhtar and ," Emerging Cloud Computing Paradigm", IJCSI, International Journal of Computer Science Issues, Vol. 8, Issue 4, No 2, July 2011 ISSN (Online): 1694-0814 in Mauritius.

[4] Abu Sarwar Zamani, Dr.Nassir S. Al-Araifi, Dr. Ashit K. Dutta," Emerging cloud computing", International Journal of Enterprise Computing and BusinessSystems, ISSN (Online) : 2230-8849, Vol. 1 Issue 2 July 2011 in India.

[5] Bruce Perens, "Debian's 'Social Contract' with the Free Software Community," 1997/msg00017.html, http://lists.debian.org/debian- announce/debian-announce- retrieved 08 Feb 2009.

[6] Bruce Perens, "The Open Source Definition,"http://opensource.org/docs/osd, 1999, retrieved 08 Feb 2009.

[7] Julie Sartain, "Open-Source Software in the Data Center—There Is a Place for It, but It Won't Do Everything," (Computerworld, 25 Jan 2008), retrieved 08 Feb 2009.

[8] Service Oriented Architecture and the Cloud By Chris Harding | CIO Insights Article Published February 8,2010.

Author Biography

Abu Sarwar Zamani was born on 15th Jan 1982 in Bihar, India. He is working as a Lecturer in Department of Computer Science in the College of Science & Humanity in Shaqra University, Kingdom of Saudi Arabia. He worked in Department of Computer Science in King Saud University, KSA. He has received his Master of Science in Computer Science from Jamia Hamdard (Hamdard University), New Delhi, India in the year of 2007, and later he did Master of Philosophy in Computer Science from Vinayak Mission University, Chennai, India in 2009. As well as did PhD. from Chandra Mohan Jha University, Meghalaya (India). He is Reviewer Board Member of International Journal of Computer Science and Security- IJCSS (Malaysia), Member of International Association of Computer Science and Information Technology- IACSIT, (Singapore, Member No: 80341225), Member of International Journal of Computer Science & Emerging Technology-IJCSET (UK), Member of International Association of Engineers-IAE (Hong Kong, Member No: 113797) and Member of IEEE. He has actively attended and published various research papers in National as well as International.