International Journal of
Emerging Research in Management &Technology
ISSN: 2278-9359 (Volume-4, Issue-6)

Research Article

June 2015

SOA based Approach and Technologies Issues on Distributed OLTP Environment

Abu Sarwar Zamani
Research Scholar,
Pacific University, Udaipur, India

Pankaj Kumar Gupta
Associate Professor, Tirupati College of
Technical Education, Jaipur, India

Abstract-

Cloud computing – a relatively recent term, defines the paths ahead in computer science world. Being built on decades of research it utilizes all recent achievements in virtualization, distributed computing, utility computing, and networking. SOA is a style of design in which applications are composed in whole or in part of reusable services. SOA aligns information systems technology well with business objectives by modeling an application as a composition of reusable services.This paperreviews technologies and approaches that unify the principles and conceptsof SOA with those of event-based programing.The paper proposes an approach to extend the conventional SOA to cater for essential Enterprises Service Bus (ESB) requirements that include capabilities such as service orchestration.

Keywords--Cloud, SOA, OLTP, ESB, Distributed Computing, Utility Computing.

I. INTRODUCTION

Distributed transactions are required when an application needs to update resources that span multiple instances or servers that are managed by different resource managers. A distributed transaction must be synchronized (that is provide atomicity, consistency, isolation, and durability [ACID]) among multiple participating databases that are distributed among different instances or servers. A transaction within a single instance of the SQL Server Database Engine that spans two or more databases is actually a distributed transaction. However, the instance manages the distributed transaction logic internally; to the user, the transaction operates as a local transaction. At the application level, a distributed transaction is managed much the same as a local transaction. At the end of the transaction, the application requests that the transaction be either committed or rolled back. A distributed commit must be managed differently by the transaction manager to avoid the risk that any kind of failure may result in some resource managers successfully committing while others roll back the transaction. This is achieved by managing the commit process in two phases: the prepare phase and the commit phase. Together, the prepare phase and the commit phase are known as a two-phase commit (2PC) protocol.

A service-oriented architecture is essentially a collection of services, among which the communication can involve either simple data passing or it could involve two or more services coordinating some activity, requiring means of connecting services to each other.A service is a function that is well defined, self-contained, and does not depend on the context or state of other services. The technology of Web services is the most likely connection technology of service-oriented architectures. Web services essentially use XML to create a robust connection. An SOA is designed to allow developers to overcome many distributed enterprise computing challenges including application integration, transaction management, security policies, while allowing multiple platforms and protocols and leveraging numerous access devices and legacy systems. The driving goal of SOA is to eliminate these barriers so that applications integrate and run seamlessly. In this way an SOAcan deliver the flexibility and agility that business users require, defining coarse grained services, which may be aggregated
and reused to facilitate ongoing and changing needs of business, as the key building blocks of enterprises. An SOA provides a flexible architecture that unifies business processes by modularizing large applications into services. A client from any device, using any operating
system, in any programing language, can access an SOA service to create a new business process. An SOA creates a collection of services that can communicate with each other using service interfaces to pass messages from one service to another, or coordinating
an activity between one or more services. Thus, with SOA, an enterprise can create, deploy and integrate multiple services and choreograph new business functions by combining new and existing application assets into a logical flow. Accordingly, for well defined and semantically unambiguous applications an SOA can serve as an enabler of just-in-time integration and interoperability of legacy applications; a key consideration for enterprises that are seeking to deploy demand driven computing environments. Services in an SOA exhibit the following main characteristics.

A. *SOA are defined as services in all functions. This includes pure business functions, business transactions composed of lower-level functions, and system service functions as well.*
B. *All services are autonomous. Service opaqueness guarantees that external componentsneither know nor care how services perform their function, they merely anticipate that they return the expected result. The implementation and execution space of the application providing the desired functionality is encapsulated behind the service interface.*

C. In the most general sense, the interfaces are invocation. This implies that it is irrelevant whether services are local or remote, the interconnect scheme or protocol to effect the invocation, nor which infrastructure components are required to establish the connection.

SOA aligns information systems technology well with business objectives by modeling an application as a composition of reusable services. In contrast to the object-oriented (OO) paradigm, services are designed to model functions rather than things. They are a natural abstraction of the concept of business services; that is, services that a business provides to its customers and partners. A service can be implemented using an object, but it need not be.

II. SERVICE ROLES IN SOA

The SOAs and Web services solutions support two key roles: a service requestor (client) and service provider, which communicate via service requests. SOA strives to meet services and business needs much more effectively. In the service-oriented architecture (SOA) of Web services, three distinct actors - the *Provider*, the *Requestor*, and the *Broker* interact to help an organization make a choice among five possible business roles.

A. Service Requester

For a business to identify with this SOA role, it must find some commonality between their business activity and the actions of a requestor. There are two clear business activities that would allow a business to benefit from implementing the role of a service requestor - Content Aggregation and Service Aggregation. Content Aggregation is an activity where a business entity interacts with a variety of content providers to process or reproduce such content in the desired presentation format of its customers (such as Internet portal or information service provider). Service Aggregation is an activity where a business entity interacts with a variety of service providers to re-brand, host, or offer a composite of services to its customers.

B. Service Provider

For a business to identify with this SOA role, it must view itself as performing some degree of an electronic service. Whether that service is defined as the processing of data or the act of carrying out a specific task, the business entity must believe it is performing work for others as an occupation or a business.

C. Registry

If a business entity finds itself collecting and cataloging data about other businesses and then selling that data to others, it may identify well with a registry, a form of SOA Broker. Usually, a registry would collect data such as business name, description, and contact information. In UDDI terms, this SOA role is often referred to as the *White Pages*.

D. Broker

Building on the concept of a registry, business entities may also be able to identify with the notion of a broker , which in UDDI terms is often referred to as *Yellow Pages*. Brokers usually extend the value proposition of a registry by offering intelligent search capability and business classification or taxonomy data.

E. Aggregator and Gateway

Any business entity that provides Broker capabilities plus the ability to describe actual policy, business processes and binding descriptions would be able to identify itself as *Green Pages*.

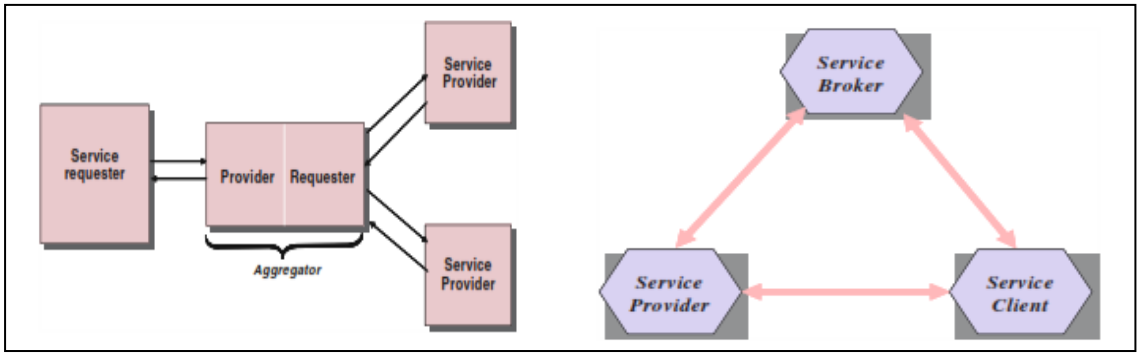

Fig. 1 The role of Service aggregator

Fig. 2 Service brokering

The process of a service requester having to directly interact with a service provider exposes service requesters to the potential complexity of discovering, exploring, negotiating, and reserving services between different service providers. An alternative approach is for an organization to provide this combined functionality directly to the service requester. This service role combines the role of service requester and provider, and is labeled as a service aggregator. The service aggregator thus performs a dual role. First, it acts as an application service provider as it offers a complete "service" solution by creating composite, higher-level services, which it provides to the service client. Service aggregators can accomplish this composition using specialized composition languages like BPEL and BPML. Second, it acts as a service requester as it may need to request and reserve services from other service providers. This process is shown in Fig. 1.

A service broker maintains an index of available service providers. The service broker is able to "add value" to its registry of application service providers by providing additional information about their services. This may include differences about the reliability, trustworthiness, the quality of the service, service level agreements, and possible compensation routes to name a few.

In Figure 2 shows an SOA where a service broker serves as an intermediary that is interposed between service requesters and service providers. Figure 2 falls under this category with the service registry (UDDI operator) being a specialized instance of a service broker. Under this configuration, the UDDI registry serves as a broker where the service providers publish the definitions of the services they offer using WSDL and where the service requestors find information about the services available.

III. MANAGING OLTP APPLICATION IN THE CLOUD COMPUTING

OLTP systems are used for order entry, financial transactions, and Customer Relationship Management (CRM) and retail sales. Such systems have a large number of users who conduct short transactions. Database queries are usually simple, require sub-second response times and return relatively few records. An important attribute of an OLTP system is its ability to maintain concurrency. To avoid single points of failure, OLTP systems are often decentralized. A system designed to handle these data requirements for a typical sales cycle may not have enough compute power, memory, or bandwidth to accommodate a surge in demand due to a special promotion or the introduction of a high-demand product. These systems lack scalability or the ability to expand IT resources as needed to meet the demand for those resources. In order to have the scalability needed to maintain service levels with customers at all levels of the sales cycle, retailers running OLTP applications would like to leverage cloud technology. The cloud can help organizations to scale up (add more compute resources in a single node) and to scale out (add more compute resources using many nodes). However, many retailers have found it difficult to transition to the cloud. One of the key challenges they face in moving to the cloud is due to the bottlenecks that occur within their data architecture. These bottlenecks are created when access to memory and databases are insufficient to meet the real time performance requirements for complex transactions.

IV. REQUIREMENTS FOR DELIVERING OPTIMAL OLTP PERFORMANCE

Companies encounter many performance challenges when dealing with multiplenodes and multiple databases required for mission-critical transactions. Thesituation becomes even more complicated when organizations themselvesare highly distributed and they are interacting with business partners thatare located across the globe. In this type of situation, traditional "distributedtransactions" can be slow and unreliable. This is primarily due to the fact thatdistributed transaction capabilities were largely created for smaller demandloads on fewer computer nodes. In other situations, a large-scale ecommerce site mayneed to scale up and out to accommodate higher transaction volumes at holiday time. In addition, enterprises need to have seamless transaction management with their partners.

With this vast range of transactions coming from a multitude of systems, ServiceOriented Architecture (SOA) services, devices, and users, the negative impact ondata sources is significant. Accommodating this demand, and the irregularity ofthe demand, requires new algorithms and techniques to be successful.

V. ESB DISTRIBUTED NATURE

Essentially, Web services denote an important technologyfor implementing SOAs; however, other moreconventional programing languages or middleware platforms may be adopted as well. Therequirements to provide an appropriately capable andmanageable integration infrastructure for Web servicesand SOA are coalescing into the concept of the ESB. The ESB exhibits two prominent features.Firstly, it promotes loose coupling of the systems takingpart in integration. Secondly, the ESB can breakup the integration logic into distinct easily manageablepieces.

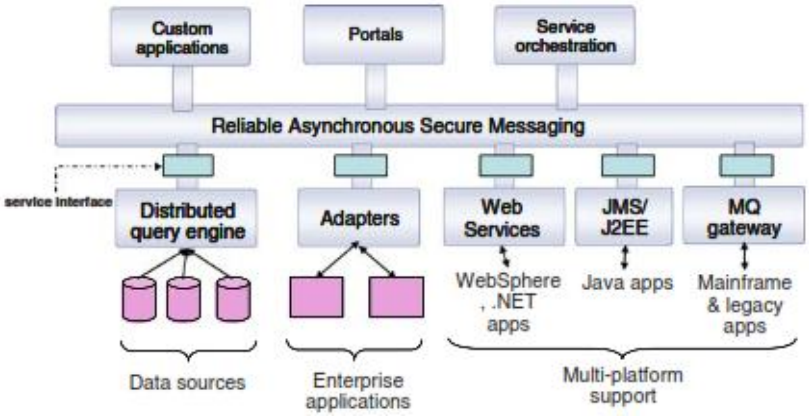

Fig. 3 ESB connecting diverse application and technologies

The ESB is an open, standards-based message busdesigned to enable the implementation, deployment,and management of SOA-based solutions with a focuson assembling, deploying, and managing distributedSOA. The ESBprovides the distributed processing, standards-basedintegration, and enterprise-class backbonerequired by the extended enterprise. The ESB isdesigned to provide interoperability between large grained applications and other components via standards-

basedadapters and interfaces. The bus functionsas both transport and transformation facilitator to allowdistribution of these services over disparate systems andcomputing environments.Conceptually, the ESB has evolved from the storeand-forwardmechanism found in middleware products.

VI. ENABLING TECHNOLOGIES AND APPLICATION SERVER

In this section, we will review the technological under pinning of ESBs in some more detail. Fundamentally,ESBs fuse the following four types of technologies: integrationbrokers, application servers, business processmanagement, and adapters.Application servers are widely used to develop anddeploy back-end server logic. Application servers enablethe separation of application (or business) logic andinterface processing and also coordinate many resourceconnections. The most prominent features of applicationservers include secure transactional execution environment,load balancing, application-level clustering acrossmultiple servers, failover management should one of these servers break down. In addition, applicationservers provide application connectivity and thus accessto data and functions associated with EIS applications also. Application servers were created forWeb-based transactions and application developmentand because of their ability to provide componentbasedintegration to back-end applications, they are particularly useful as support framework for integrating business processes.

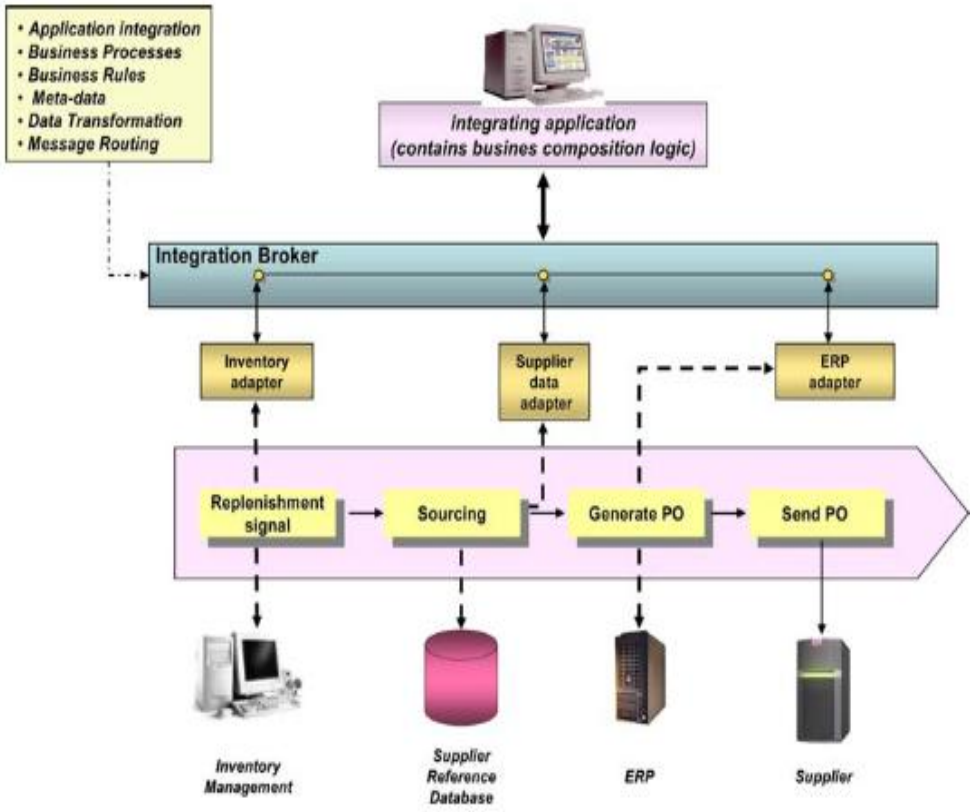

Fig.4 Integration broker integrating disparate back end system

VII. CONCLUSIONS

This challenge in automatedbusiness integration has driven major advancesin technology within the integration software space. As aresult the SOAhas emerged recently, essentially addressingthe requirements of service requesters, providers andservice brokers, regarding loosely coupled, standardsbased,and protocol-independent distributed computingand offering ways to achieve the desired levels ofbusiness integration effectively, mapping IT implementationsmore closely to the overall business process flow.This paper has surveyed approaches, technologies,and research issues related to services architectures andunderlying technologies.

REFERENCES

[1] Alonso, G., Casati, F., Kuno, H., Machiraju, V.: Web Services: Concepts, Architectures and Applications. Springer, Heidelberg (2004).

[2] Candadai, A.: A dynamic implementation framework forSOA-based applications. Web Logic Dev. J. WLDJ **September/October**,6–8 (2004)

[3] Dhesiaseelan, A., Ragunathan, V.: Web Services ContainerReference Architecture (WSCRA). In: Proceedings of the International Conference on Web Services, IEEE, pp. 806– 805, 2004.

[4] Krafzig, D., Banke, K., Slama, D.: Enterprise SOA: ServiceOriented Architecture Best Practices. Prentice-Hall, Englewood Cliffs (2005)

[5] Marcia Kaufman, COO and Principal Analyst" The Challenge of Managing On-line Transaction Processing Applications in the Cloud Computing World.

[6] Mike P. Papazoglou · Willem-Jan van den Heuvel" ervice oriented architectures: approaches, technologies and research issues".

[7] Arora, A., et al.: Web services for management (WSManagement).Technical report, Advanced Micro Devices, Dell, Intel, Microsoft Corporation and Sun Microsystems, October 2004.

[8] Booth, D., et al.: Web Service Architecture. http://www.w3.org/tr/ws-arch/, W3C, Working Notes, 2003/2004.

[9] Box, D., et al.: Simple Object Access Protocol (SOAP), Version1.1. W3C Note, W3C, May 2000. http://www.w3.org/TR/SOAP/

[10] Chappell, D.: Enterprise Service Bus. O'Reilly Media, Inc.,Sebastopol (2004).

[11] Vladimir Tosic, David Mennie, and Bernard Pagurek, Software Configuration Management Related to the Management of Distributed Systems and Service-Oriented Architectures. Network Management and Artificial Intelligence Lab Department of Systems and Computer Engineering, Carleton University 1125 Colonel By Drive, Ottawa, Ontario, K1S 5B6, Canada{vladimir,bernie}@sce.carleton.ca

[12] Abu SarwarZamani, Pankaj Kumar Gupta, SOA-Based Distributed System in Online Transaction Processing, Journal of Information Sciences and Computing Technologies(JISCT) ISSN: 2394-9066

[13] Abu SarwarZamani, Mohammad Jawed Miandad, Shakir Khan" Data Center –Based, Service Oriented Architecture (SOA) in Cloud Computing" *International Journal of Computing Science and Information Technology*,2013, Vol. 01 (01), 33-37 ISSN: 2278-9669, January 2013 (http://ijcsit.org)

[14] Arsanjani, A.: Introduction to the special issue on developingand integrating enterprise components and services. Commun. ACM **45**(10), 30–34 (2002)

[15] Atkinson, B., et al.: Web Services Security (WS-Security).Technical report, Microsoft, IBM and Verisign, April 2002

