PERFORMANCE IN SOA CONTEXT

Sudheer Doodi[1], Mohd Fahimuddin[2]

[1]Bangalore, India
[2]Bangalore, India

Abstract

Service Oriented Architecture (SOA) is a new software development paradigm where application is developed in distinct pieces, providing functionality as a service to other applications, internal and external. Though SOA has many benefits like Loose coupling, incremental implementation, high business agility, reduced lead time, reduces cost, reduced risk and new opportunities to deliver values, some of these benefits also introduce performance overhead. For smaller systems with small number of services and small number of users, this performance hit may be minimal. However, maintaining required performance levels for large SOA implemented systems is big challenge. In this paper, we will discuss the SOA concepts, its architectural style, attributes, performance advantages and disadvantages of SOA and techniques to improve the performance of SOA.

Keywords: *Service Oriented Architecture (SOA), service-level-agreements (SLA), Extensible markup language (XML), JavaScript object Notation (JSON), Representational State Transfer (REST), Simple Object Access Protocol (SOAP).*

---***---

1. INTRODUCTION

SOA is an architecture for building application as a set of loosely coupled components called services. SOA is been evolved in all fields of engineering, for example, in a traditional music player everything is tightly integrated, where as in latest music players each part – speakers, cassette player, DVD player, amplifier are independent, they are assembled to form a music player, each of this parts play as service in assembled player. Evolutions in software engineering begin from procedural to structure to object-oriented to component based programming and now to service oriented over the years. Each evolution comes with an abstraction building on the previous and SOA applies best of object and component development thus gives higher level of abstraction.

A SOA application mainly comprises of a service provider and a service consumer, service consumer could be a web application or any other service as well, that depends on this service component to do its job.

Key idea behind SOA is to divide the application into small pieces so these components run on multiple servers as separate services. Combination of all services form a platform to serve internal and external users.

2. ATTRIBUTES OF SOA

A SOA service is an independent business logic, following are the attributes of a service in SOA.

Stateless: SOA services neither records last activity done nor care about the next. Services are concise to its functionality making it independent of the context or state of other services.

Discoverable: Needy consumers of the service must be able to discover it easily otherwise it is unlikely to be used. SOA service directory maintained by service providers is the best place to publish the service.

Self-describing: The SOA service interface must describe, expose and provide all aspects of a service i.e. an entry point. The interface camouflages all technical implementation of a service but providing information a consumer needs to discover and consume it.

Composable: SOA services are, by nature, composite. New business solutions can be achieved by composing one or more services in SOA.

Loose coupling: Loose coupling separates the concerns of an application into independent pieces. Which in turn provides a process for services to call one another without being tightly bound. Boundaries establishment help in separation of concerns, where a boundary is any separation (logical or physical) that serves a given set of responsibilities.

Governed by policy: Services are built by contract. Relationships between service domains and services are controlled by policies and service-level-agreements (SLAs), providing process consistency and reducing complexity.

Location, language, and protocol Independent: Services are engineered to be protocol, location and platform independent. Which makes them accessible by any authorized user, any platform and from any location.

3. SOA ARCHITECTURE

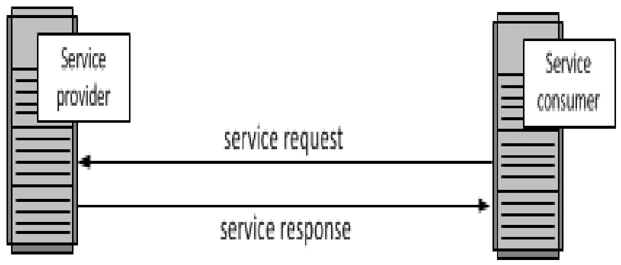

Fig 1 Basic SOA Architecture

The above figure illustrates a basic service oriented architecture. It shows a service provider on the left and service consumer on the right. Service consumer sending a service request message to service provider, service provider process the request based on input parameters then forms a response, which is sent back to service consumer. The request and response messages are understandable to both the service consumer and service provider.

4. NON-FUNCTIONAL PERFROMANCE ADVANTAGES OF SOA

High Scalability: Since each service component is independent to each other, they can easily clustered across many servers with appropriate load balancing, hence the SOA systems are highly scalable.

Higher Availability: High availability can be achieved by introducing redundancy in SOA components using clustering, with SOA's logical decoupling facility this can easily done, so SOA allows to design a very resilient system.

5. NON-FUNCTIONAL PERFORMANCE DISADVANTAGES OF SOA

Latency: Service Oriented Architecture involves distributed processing, hence it is network centric. Service provider and service consumer are usually in different servers, often on different machines and different data centers. The messages that exchange over network to provide service increases the overall response time. Networks used for SOA are mostly internet which do not guarantee latency. Therfore, SOA is not feasible for real-time systems and presents challenges for near real-time systems.

Bandwidth: SOA is prone to consume more bandwidth in terms of headers and additional tags which come as part of service responses.

Lookup: The service consumer sometimes requires an extra call to a directory of services to locate the desired service. This extra call is an overhead and increases the total time needed to perform the transaction.

Processing: The data transmission in services is done in forms of XML or JSON technology. These creates additional overhead in processing of data. Processing consists of at least three activities, parsing, validation and transformation which are CPU and memory intensive.

Security: Providing security for each service call is an additional overhead in SOA.

Connections: SOA is highly clustered architecture where each cluster has its own set of open connections to database or directory services, which in turn impacts the end application performance.

6. TECHNIQUES TO IMPROVE PERFORMANCE

Code for Performance: There are a various programming techniques that can improve SOA application performance.

a. Application should be designed to use appropriate calls. As client and service are not sitting on same computer or even in the same datacenter, there could be a great distance between them, so avoid making frequent calls to SOA service. Lesser the calls from client to service layer, the better the performance. Choose your calls in such a way that more work can be done in one call instead of multiple calls to do the same work.
b. Make asynchronous web method calls.
c. While forming response, data provided should be precise and compact, which means no irrelevant data should be sent.

Choose right interaction style: Representational State Transfer (REST) has better response times and throughput than Simple Object Access Protocol (SOAP)

Combining prudent loose coupling with calculated tight coupling: The Simplistic implementation of SOA is to develop as many loosely coupled services to provide required functionality. The fact is loose coupling introduces extra calls overhead, while tight coupling can smooth out bottlenecks.
So design your SOA in way of providing loose coupling only as much as it needs.

Apply **Caching**, pre-fetching and messaging techniques at the macro level.

7. CONLUSIONS

For web applications, decoupling along functional requirements may lead to an increase in network latency, decrease in throughput, and consequently a negative impact on performance. In other words, a service-oriented architecture may not be seen as a performance oriented architecture.

If you want to have high performing application, then a loosely coupled SOA, while beneficial in terms of system maintenance, management and evolution, may turn out to have unsatisfactory performance. One might argue that performance is lacking because the system was not decomposed in the best possible way, but the more you draw lines to make it service oriented the more it is going to impact the performance.

To achieve performance oriented architecture with service oriented mindset then it is good idea to tune the critical subsystems with a performance oriented mindset. We do not suggest to replace SOA with POA across but pay attention to isolated functionality that would benefit from performance oriented architecture.

REFERENCES

[1]. Liam O'Brien, Len Bass, Paulo Merson, Quality Attributes and Service-Oriented Architectures, CMU,SEI-2005-TN-014
[2]. Understanding Service-Oriented Architecture by David Sprott and Lawrence Wilkes,http://msdn.microsoft.com/en-us/library/aa480021.aspx
[3]. How service-oriented architecture (SOA) impacts yourIT infrastructure, http://public.dhe.ibm.com /software/solutions/soa/pdfs/wp_how-soa-impacts-your-it-infrastructure.pdf
[4]. Service-Oriented Architecture (SOA) Definition, http://www.service-architecture.com/articles/web-services/service-oriented_architecture_soa_definition.html
[5]. Six Strategies for Building High Performance SOA Applications by Uwe Breitenb cher, Oliver Kopp, Frank Leymann, Michael Reiter, Dieter Roller, and Tobias Unger,ceur-ws.org, vol 847,paper 16.
[6]. S.Neelavathi and K.Vivekanandan , An Innovative Quality of Service (QOS) based Service Selection for Service Orchrestration in SOA, An Innovative Quality of Service (QOS) based Service Selection for Service Orchrestration in SOA
[7]. Vikas Anand, Oracle, ODTUG, Kscope11, Tuning Your SOA Infrastructure for Performance and Scalability.