메시지 감소를 통한 SOA기반 시스템의 성능 개선방안

정현호*, 오수민, 이상범
*단국대학교 전자계산학과
e-mail:ilhyunli@nate.com

A SOA Based System of Strategy Improvement Performance by Message Reduction

Hyun-Ho Jung*, Su-Min Oh, San-Bum Lee
Dept of Computer Science, Dan-Kook University

요 약

최근 기업들의 정보 시스템들은 비즈니스 환경이 나날이 복잡해지고, 기업 운영에 요구되는 비즈니스 서비스들이 급격하게 변화되고 있다. 이러한 변화에 유연하고 민첩하게 대응하기 위한 해결책으로 서비스 지향 아키텍처(SOA : Service Oriented Architecture)에 대한 관심이 확대되고 있다. 특히 SOA가 제공하는 통합용이성, 재사용성, 확장성, 조직기민성 등의 실익으로 인해 많은 기업들은 SOA를 도입하고자 노력하고 있다. 하지만 SOA는 성능에 문제점을 가지고 있으며 대형 벤더들은 이를 해결하고자 노력해왔다.

본 논문에서는 SOA의 성능에 관한 문제점을 개선하기 위해 이전에 제시된 방법들에 대해 알아보고, SOA의 성능을 개선하고 서비스간의 메시지 전달횟수를 줄이기 위해 중계서비스를 사용하는 방법을 제안한다. 그리고 서비스를 탐색하고 서비스를 이용하는데 걸리는 시간을 줄이기 위해 서비스 리포지토리 캐쉬화 하는 방법을 제안한다.

1. 서 론

2000년 이후 웹의 확산으로 인해 기업들의 비즈니스 환경은 계속해서 변화하고 있다. 특히 조직의 업무환경이 웹으로 이전되었으며 이를 지원하는 정보시스템은 나날이 복잡해지고 있다.

더욱이 기업 운영에 필요한 비즈니스 서비스들은 고객의 요구사항과 동종 기업들과의 경쟁으로 인해 급격하고 복잡하게 달라지고 있다. 이러한 변화에 대처하기 위해 기업의 정보 시스템은 유연하고 민첩하게 대처할 수 있는 시스템에 대한 필요성이 증가하게 되었다. 몇 년 전부터 기업들과 대형 벤더들로부터 주목받기 시작했던 서비스 지향 아키텍처(SOA : Service Oriented Architecture)는 이러한 요구를 만족해줄 수 있는 충분한 대안으로 떠오르고 있다. 특히 SOA가 제공하는 통합용이성, 재사용성, 확장성, 조직기민성 등의 실익들은 많은 기업들로부터 SOA에 대한 관심을 더욱 확대시키고 있다.

SOA 환경에서 일반적으로 시스템 간의 연동 및 비즈니스 서비스의 활용을 위해 웹서비스 기술을 이용한다. 웹서비스를 이용한 SOA는 SOAP(Simple Object Access Protocol)이라는 XML 기반의 메시지 처리를 수행하는데 이로 인한 속도저하라는 성능의 문제점을 가지게 된다. 하지만 접근성, 호환성, 경제성 등의 웹서비스가 주는 많은 이점들 때문에 웹서비스의 사용은 필수불가결한 문제로 인식되고 있다. 때문에 대형 벤더들은 이 문제를 해결하기 위해 다른 여러 가지 대안을 내놓고 있다.

본 논문에서는 SOA의 성능에 관한 문제점을 개선하기 위해 이전에 제시된 방법들에 대해 알아보고, 서비스간의 메시지 전달횟수를 줄여 성능을 개선하는 방법으로 중계 서비스를 사용하는 방법과 서비스를 탐색하고 이용하는데 걸리는 시간을 줄이기 위해 서비스 리포지토리를 캐쉬화 하는 방법을 제안한다.

본 논문은 다음과 같이 구성되어 있다. 2장에서는 SOA의 구성, SOAP 및 XML 처리과정과 성능문제를 해결하기 위한 이전의 방법들에 대해 설명한다. 3장에서는 성능을 개선하기 위한 방법으로 중계 서비스를 사용하는 방법과 서비스 리포지토리를 캐쉬화 하는 방법에 대해 제안한다. 마지막으로 4장에서 결론을 맺는다.

본 과제는 한국소프트웨어진흥원의 SW공학 요소기술 개발과 전문인력 양성사업의 결과물임을 밝힙니다.

2. 관련연구

2.1 SOA의 개념

SOA는 애플리케이션 프론트엔드, 서비스. 서비스 리포지토리, 서비스 버스의 주요 개념에 바탕을 둔 소프트웨어 아키텍처이다. 여기서의 서비스는 계약, 하나이상의 인터페이스, 그에 대한 구현으로 이루어 진다[1].

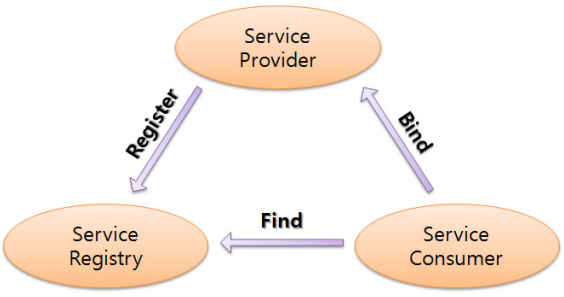
SOA는 비지니스 로직과 기술을 추상화하여, 도메인 간에 느슨한 결합을 유도한다. SOA는 과거 플랫폼의 진화물로서, 전통적인 아키텍처의 특징을 고스란히 가지고 있으며, 명확한 원칙을 가지고 SOE를 지원하며 서비스 지향을 촉진한다. SOA는 이상적으로 엔터프라이즈 환경을 표준화하지만, 치밀한 사전 계획에 기반한 이전 필요성과 현재에도 진화하고 있는 기술에 대한 지원만이 이러한 목적을 달성할 수 있다.

2.2 SOA 구성요소와 동작

SOA의 구성요소는 크게 서비스와 메시지로 구분된다. 첫 번째로 서비스(Service)는 명확하고 기능적인 의미를 지닌 소프트웨어 컴포넌트로, 고차원의 비즈니스 개념을 캡슐화 하고 있는 것을 말한다. SOA의 관점에서 서비스는 인터페이스를 통해 자신이 가진 비즈니스 프로세스를 처리할 수 있는 컴포넌트로 정의된다. 서비스는 인터페이스와 구현 부분으로 구성된다.

두 번째 구성요소는 메시지(Message)이다. 서비스 제공자와 서비스 사용자는 메시지를 통해 서로 통신한다. 서비스 제공자는 서비스 명세를 통해 자신이 가진 서비스의 인터페이스를 공개하는데, 이 명세 내에는 서비스가 제공하는 기능과 이를 이용하기 위해 사용자와 주고 받아야 하는 메시지의 형식이 정의되어 있다. SOA 관점에서 서비스는 플랫폼 독립적이어야 하므로, SOA에서 정의되는 메시지는 특정 기술에 독립적이어야 한다.

SOA에서 서비스는 분산환경에 배치되고 실행되는 것이 일반적인 특성이다. 서비스 제공자는 서비스 레지스트리에 서비스를 등록하고, 서비스 클라이언트는 서비스 레지스트리로부터 필요한 서비스를 발견하고 구성하여 애플리케이션 프론트엔드의 요구사항을 실행한다.[2,3,4]



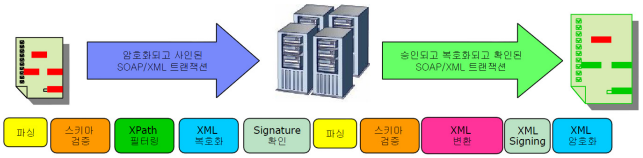
(그림 1) SOA 구성

SOA는 (그림 1)과 같이 동작한다. 이러한 일련의 과정들은 메시지 교환을 통해 이루어 지게 된다.

2.3 SOAP와 XML 메시지 처리

SOAP는 일반적으로 알려진 HTTP, HTTPS, SMTP 등을 사용하여 XML 기반의 메시지를 컴퓨터 네트워크 상에서 교환하는 형태의 프로토콜이다. SOAP는 웹 서비스에서 기본적인 메시지를 전달하는 기반이 된다.

SOAP에서 XML기반의 메시지를 처리하기 위해서는 일반적으로 (그림1)과 같이 파싱, 스키마 정의, 경로(PATH) 필터링, 복호화, 서명확인 등의 복잡한 과정을 거쳐야 한다.



(그림 2) 일반적인 XML처리과정

이러한 일련의 메시지 처리과정은 SOA의 서비스간 연동 시 발생하는 속도저하의 원인이 된다.

2.4 SOA의 성능문제

SOA가 가지는 성능 문제는 사용자 뿐만 아니라 SOA관련 소프트웨어 공급자들도 인정하고 있는 부분이다. 때문에 벤더들은 기업의 비즈니스 시스템에 SOA를 적용하면서 시스템의 성능을 향상하기 위해 여러 가지 대안을 내놓고 있으며 대표적인 방법들은 다음과 같다.

첫 번째, XML 포맷 데이터를 처리하는 특화된 하드웨어 어플라이언스를 사용하는 방법이다. 소프트웨어적인 방법으로 XML을 처리하는 것보다 일련의 처리과정에 소요되는 시간을 크게 단축시킬 수 있다.

두 번째, 바이너리 XML을 사용하는 방법이다. XML 포맷은 태그 형태로 보내지기 때문에 CORBA 같은 미들웨어 기술과 비교해서 상대적으로 느리다. MTOM(Message Transmission Optimization Mechanism) 메시지 인코딩을 사용하여 SOAP 메시지와 함께 큰 이진 첨부 파일을 원시 바이트로 전송함으로써 메시지의 크기를 작게 하여 성능을 높일 수 있다.[7]

세 번째, 미들웨어를 통한 SOA 구현방법이다. XML을 사용하는 SOAP 대신에 CORBA(Common Object Request Broker Architecture)같은 미들웨어를 사용하거나 IBM의 MQseries 같은 메시지 지향 미들웨어를 사용하여 SOA를 구현하여 성능문제를 개선 할 수 있다.

이러한 방법들은 대부분 서비스 간의 메시지 교환에서 생기는 속도 저하문제를 해결하기 위한 것들이 논의되고 있다. 하지만 서비스간 메시지 교환 횟수를 줄이게 된다면 위와 같은 효과를 얻을 수 있을 것이다.

3. SOA의 성능 개선방법 제안

본 논문의 제안은 서비스 구조의 변경으로 서비스 간의 접근성을 높여 메시지 전달 횟수를 줄임으로써 성능을 높이는 방법과 서비스를 이용하기 위해 탐색하는 과정을 보다 간소화 하여 이에 소요되는 시간을 줄이는 방법으로 나누어 진다.

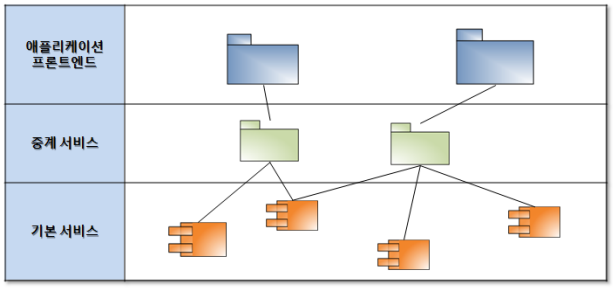
3.1 밀접한 관계를 가지는 서비스를 가상적으로 통합

SOA의 서비스는 명확하고 기능적인 의미를 가지며, 비즈니스 프로세스를 시작한 후 최종적으로 비즈니스 프로세스의 결과 값을 받는 애플리케이션 프론트엔드는 이러한 서비스들이 하나 이상으로 이루어진다.

서비스들의 수가 많아질수록 서비스간의 연동을 위한 메시지 교환이 증가하게 되고 이는 전체 시스템의 성능을 저하시킨다.

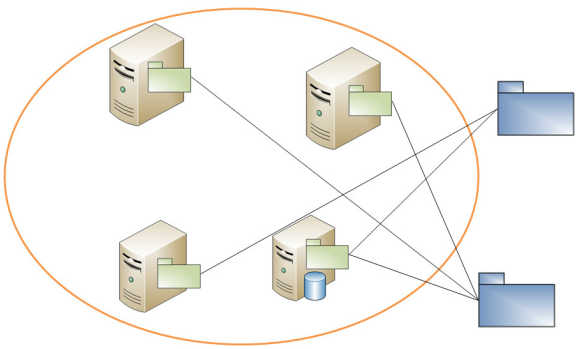
이러한 문제를 개선하기 위해서 일종의 게이트웨이 역할을 하는 가상적인 서비스를 사용하여 비즈니스 프로세스의 처리에 필요한 메시지 교환과정을 대신하도록 중계역할을 제공한다. 또한 분산처리 시스템 환경에서 백본으로 연결된 여러 대의 서버에 이러한 역할을 하는 서비스를 제공함으로서 네트워크 부하를 줄일 수 있다.

(그림 3)은 기존의 SOA환경에 제안한 방법을 적용하였을 경우 애플리케이션 프론트엔드와 서비스간의 연결형태를 보여준다.



(그림 3) 중계서비스를 이용한 SOA

또한 (그림 4)와 같이 분산처리 시스템 환경에서 각각의 서버마다 중계서비스를 통해 애플리케이션 프론트엔드가 서버 내에 있는 것처럼 동작하도록 하여 네트워크 효율을 높일 수 있다.

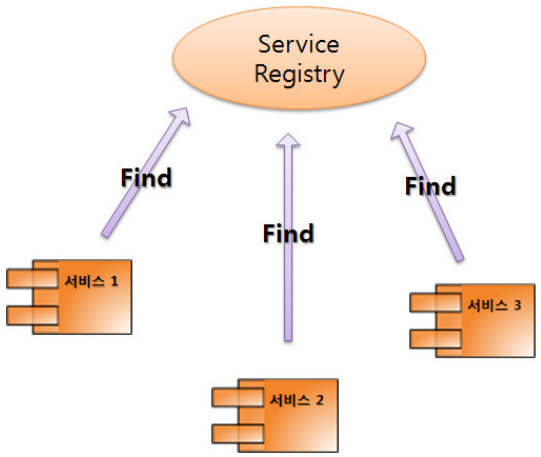


(그림 4) 애플리케이션 엔드포인트의 중계서비스 참조

3.2 서비스 리포지토리의 캐쉬화

SOA에서는 서비스를 탐색하고 서비스를 이용하기 위해서 서비스 리포지토리 기능을 사용한다. 이 기능은 서비스 등록을 통해 서비스의 실제 위치를 알지 못해도 등록정보를 통해 서비스에 접근할 수 있게 된다. 하지만 이 과정에 소요되는 시간은 모든 서비스에서 메시지 교환마다 필요하게 된다.

특히 정보 시스템이 클수록 서비스의 수가 증가하여 이 과정에 소요되는 시간도 늘어나게 된다. (그림 5)는 서비스 연동을 위해 메시지를 주고받을 때 상대 서비스의 위치를 알기위해 위치정보를 찾는 동작을 보여준다. 이러한 동작은 메시지 교환 시 계속해서 발생되므로 불필요한 메시지 교환이 일어나게 된다.

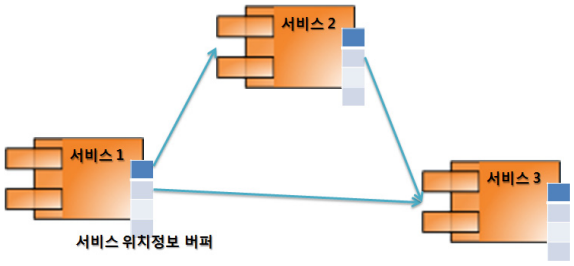


(그림 5) 상대 서비스의 위치를 찾는 과정

이러한 문제를 개선하기 위해 서비스마다 자주 연결되는 상대 서비스의 접근위치를 캐쉬에 저장하고, 이를 통해 상대 서비스에 접근하도록 한다.

서비스의 탐색은 시스템 유지보수가 생기거나 새로운 애플리케이션 엔드포인트가 생기기 이전까지는 유효하기 때문에 쉽게 바뀌지도 않으며 캐쉬를 통해 기억되는 접근위치는 자주 사용되는 몇 개의 서비스만 기억하도록 하면 된다.

(그림 6)는 서비스 리포지토리의 캐쉬화를 통해 서비스의 접근 방법을 보여준다.



(그림 6) 버퍼를 이용한 서비스 연동

위의 그림과 같이 서비스 내에 위치한 버퍼를 통해 추가적인 메시지 교환 없이 상대 서비스 위치를 알 수 있다. 버퍼에 저장될 서비스 위치정보는 처음의 서비스 연동시 서비스 리포지토리에 접근해 받아오게 되며 그 이후로부터는 서비스 리포지토리에 접근할 필요가 없어진다.

4. 결론 및 향후 연구

현재 많은 기업들이 SOA에 관심을 보이면서도 자사의 정보시스템에 적용을 꺼리는 큰 이유는 성공사례의 부족과 구현의 어려움, 보안과 성능 같은 문제점들 때문이다.

본 논문에서 우리는 서비스간의 메시지 전달횟수를 줄여 성능을 개선하는 방법으로 중계서비스를 사용하는 방법과 서비스를 탐색하고 서비스를 이용하는데 걸리는 시간을 줄이기 위해 서비스 리포지토리를 캐쉬화 하는 방법을 제안했다. 제안한 방법은 SOA가 가지는 기본 구조에 몇가지 기능을 추가하여 성능문제를 개선할 수 있도록 해준다.

하지만 제안한 방법이 성능문제를 얼마나 해결해 주는지 추가적인 테스트와 검증이 필요하다.

향후 연구로는 제안한 방법이 SOA 환경에서 성능문제를 얼마나 개선하는지 실제 SOA로 구현된 시스템을 기반으로 테스트를 진행하고, 서비스 사용 빈도가 증가함에 따라 얼마나 효과가 있는지 검증이 필요하다. 또한 제안한 방법이 성능문제로 인해 SOA 적용에 실패한 사례에서 얼마나 효과를 거둘 수 있는지 추가적인 연구가 필요하다.

참고문헌

[1] Dirk Krafzg, Karl Banke, Dirk Slama, “Enterprise SOA” Prentice Hall, 2005.

[2] Erl, T., Service-Oriented Architecture: Concepts, Technology, and Design, Prentice Hall, 2005.

[3] Brien, L., Bass, L., and Merson, P., Quality Attributes and Service-Oriented Architectures, Technical Note CMU/SEI-2005-TN-014, September, 2005.

[4] Woodall, P., et al. "Investigating service-oriented system performance: a systematic study", Software: Practice and Experience, Vol. 37, Issue.2, 2007.

[5] IBM. SOA and Web services. http://www.ibm.com/developerworks/webservices/

[6] HP, Service Oriented Architecture, https://h10078.www1.hp.com/cda/hpms/display/main/hpms_content.jsp?zn=bto&cp=1-11-130_4000_100__

[7] .NET Framework Developer Center, MTOM Encoding, http://msdn.microsoft.com/ko-kr/library/aa395209.aspx

[8] Wikipedia, Service-oriented architecture, http://en.wikipedia.org/wiki/Service-oriented_architecture