SOA와 CBD를 이용한 웹서비스 시스템 설계 및 구현

신혜원° 윤수진
고려대학교 컴퓨터과학기술대학원
{hester°, bani23}@korea.ac.kr

A Design and Implementation of Web Services System Using SOA and CBD

Haewon Shin° , Soojin Yun
Graduate School of Computer Science & Technology, Korea University

요 약

시스템의 요구사항이 복잡해 짐에 따라, 분산 환경의 시스템 간에 일부 기능 모듈을 컴포넌트로 개발하여 공유하는 형태로 시스템이 발전하고 있다. 그러나 컴포넌트는 특정 기술에 종속적이면서 상대적으로 연관도가 높아 동일한 환경에서의 재사용성은 뛰어나지만, 플랫폼과 구현 언어에 의존성이 강하여 상호운용의 문제가 발생한다. 이를 보완하기 위한 아키텍쳐 중 하나인 SOA는 구현 기술로부터의 독립성 및 유연성(Flexibility)이 높아 이기종 시스템간 통합 구축에 적합하다.

본 논문에서는 컴포넌트의 상호운용 능력을 높이기 위한 방안으로 SOA와 CBD를 이용한 웹서비스(Web Services) 모델을 제안하였다. 또한 기존 어플리케이션에 존재하는 컴포넌트를 래핑(Wrapping)하여 재사용 가능한 서비스를 생성하여 제공하고, 이기종 환경에서 재사용하는 구현 사례를 제시함으로써 높은 상호운용 능력이 있음을 보였다.

1. 서 론

정보 기술 환경이 발전하고 시스템의 요구사항이 복잡해 짐에 따라 분산 환경의 시스템 간에 일부 기능을 공유하는 형태로 시스템이 발전하고 있으며, 근래에는 CBD 방법론을 통하여 소프트웨어가 개발됨에 따라 시스템의 기능 모듈들이 컴포넌트로 개발되고 있다[1][2]. 이러한 컴포넌트는 특정 기술에 종속적이면서 상대적으로 연관도가 높아(Tightly-Coupled) 동일한 환경에서의 재사용성은 뛰어나지만, 서로 다른(Heterogeneous) 환경에서 개발된 기존 컴포넌트 (Legacy Component)들을 재사용하고자 할 때 상호운용의 문제가 발생한다는 단점이 있다[3]. 그래서 서로 다른 기술간의 상호운용성(Interoperability)을 위한 많은 방안들이 연구되어 왔으며, 이중에서 서비스(Service)를 이용한 개발은 구현 기술로부터의 독립성 및 유연성이 높아 이기종 시스템간 통합 구축에 적합하다.

본 논문에서는 컴포넌트의 상호운용 수준을 높이기 위한 방안으로 서비스를 기반으로 한 아키텍처와 CBD를 이용한 웹서비스 모델을 제안하고, 기존의 컴포넌트 기반 시스템을 래핑(Wrapping)하여 이기종 환경에서 재사용함으로써 사례 연구를 제시한다.

2. 관련 연구

2.1 SOA(Service-Oriented Architecture)

서비스는 독립적인 비즈니스 기능을 구현한 소프트웨어 컴포넌트로서, 어플리케이션이나 다른 서비스가 외부에서 그 서비스를 찾을 수 있고 공개된 인터페이스를 통해 접근하며, 주로 메시지 기반의 비동기 커뮤니케이션 방식으로 사용하도록 구현되는 소프트웨어 개체이다.

SOA는 이러한 서비스를 기반으로 어플리케이션을 구축하거나 서비스 자체를 구축하도록 제공되는 아키텍쳐 혹은 소프트웨어를 설계하는 방법이다.

SOA는 기본적으로 서비스를 구현하여 제공하는 서비스 제공자Service Provider)와 서비스를 요청하여 사용하는 서비스 요청자(Service Requester), 서비스 제공자가 제공하는 서비스 명세(Service Description)를 저장소에 공개 등록하여 서비스 요청자가 필요한 서비스를 찾아 쓸 수 있도록 중개해주는 서비스 중개자(Service Broker)로 구성된다. 서비스 요청자는 서비스 제공자를 호출(Bind)하여 사용하는 소프트웨어 개체이다. 흔히 클라이언트(Client)라고 하는 사용자 어플리케이션이 될 수도 있고, 다른 서비스가 될 수도 있다. 서비스 제공자는 인터페이스로 정의되는 서비스 명세를 실제로 구현한 소프트웨어 개체를 말한다. 서비스 중개자는 일종의 서비스 제공자로서, 서비스와 서비스 명세를 찾아볼 수 있는 레지스트리 및 중간자적 역할을 해준다.

SOA를 구성하는 세 가지 요소는 서로 느슨하게 연결되어 있으며 완전히 독립적이다. SOA가 다른 모델과 다른 특징은 비즈니스 프로세스를 위주로 정보시스템 인프라를 구축한다는 것과 느슨하게 결합된(Loosely Coupled) 어플리케이션 개발 및 관리 아키텍처라는 점을 들 수 있다. 느슨하게 결합되어 있다는 것은 필요할 때만 연결이 이루어진다는 이야기이며 또한 서로 독립적으로 운영되고 있음을 뜻한다[4].

2.2 CBD(Component-Based Development)

CBD는 재사용 가능한 소프트웨어 모듈 컴포넌트를 생성, 조립 생산, 선택 평가 및 통합으로 구성하여 더 큰 컴포넌트를 생성하거나 완성된 어플리케이션 소프트웨어를 구축하는

개발 기법이다. 프로그래밍 위주의 전통적인 개발 방법론과는 달리 테스트가 완료된 이미 만들어진 컴포넌트를 기본 아키텍처와 설계도에 따라 조립하는 방식의 개발 형태로써, 1)컴포넌트의 오퍼레이션 및 상호 오퍼레이션을 정의하는 명세 개발, 2)객체나 컴포넌트로부터 컴포넌트 구축, 3)컴포넌트들을 이용해 어플리케이션 조립 등 세가지 활동을 필요로 한다. 기존의 소프트웨어 컴포넌트를 사용자가 원하는 모양으로 재조립해서 새로운 어플리케이션을 만들 수가 있으므로 개발 생산성과 경제성을 높일 수 있다.

컴포넌트 기반 시스템은 개발하려는 시스템 전체를 컴포넌트화 할 수 있는 여러 개의 단위로 분리해 개발하므로 복잡성을 단순화 시킬 수 있고, 컴포넌트는 캡슐화 되어있어 로직상의 에러나 런타임 에러 등의 범위를 컴포넌트로 한정할 수 있어 유지보수가 용이하다[5].

3. 제안 시스템 설계

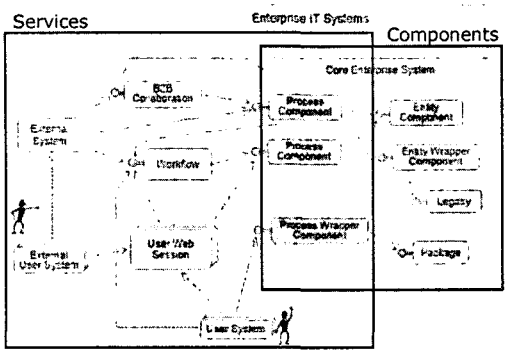
3.1 기존 시스템의 문제점

최근에는 CBD 방법론을 통하여 소프트웨어가 개발됨에 따라 시스템의 기능 모듈들이 컴포넌트 형태로 개발되고 있다.

컴포넌트를 이용한 개발의 경우 특정 기술에 종속적이면서 상대적으로 연관도가 높아 동일한 환경에서의 재사용성은 뛰어나지만, 서로 다른 환경에서 개발된 기존 컴포넌트들을 재사용하고자 할 때 상호운용의 문제가 발생한다는 단점이 있다. 이러한 문제를 해결하기 위하여 프로그램 코드 자체를 사용하고자 하는 환경에 맞게 마이그레이션(Migration) 하거나 재구축(Rebuilding) 하는 경우도 있으나, 프로그램의 개발 생산성 및 컴포넌트의 재사용성을 높이는 근본적인 해결책은 되지 못한다.

3.2 제안 시스템 구성도

제안 시스템은 [그림 1]와 같이 컴포넌트들로 이루어진 컴포넌트 레이어(Component Layer)와 리소스들을 관리하기 위한 서비스 레이어(Service Layer)로 구성되어 있다.



[그림 1] 컴포넌트와 서비스

Core Enterprise System은 컴포넌트들로 구성되며, 엔티티 컴포넌트들과 레거시 시스템(Legacy System)의 래퍼 컴포넌트들로 컴포넌트 레이어를 형성한다. 또한 컴포넌트 레이어에 속해있는 컴포넌트들을 활용하여 프로세스 단위로 외부에 서비스를 제공할 수 있도록 서비스 레이어를 형성한다.

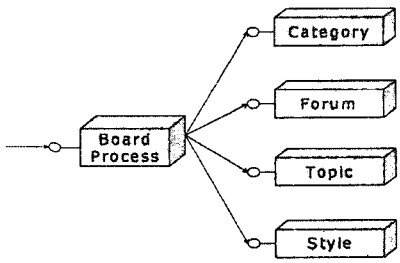
내부 사용자 어플리케이션(User System)은 서비스 제공자와 동일한 환경을 가진 서비스 요청자로써 컴포넌트 인터페이스를 통해 직접 컴포넌트를 호출하고, 외부 사용자 어플리케이션(External User System)은 서비스 제공자와 서로 다른 환경을 가진 서비스 요청자로써 서비스 레이어를 통해 서비스를 호출한다. 이것은 서비스 레이어 영역에 웹서비스 기술이 적용되고 SOA를 구성하기 때문에 가능하다. 이렇게 함으로써, 서비스와 컴포넌트는 서로 독립적인 개념이 아닌 상호보완적 의미를 가지며, 컴포넌트 기반 시스템에서도 서비스의 특징인 느슨한 연결(Loosely Coupled), 성긴 인터페이스(Coarse Grained), 비동기성(Asynchronous)을 가지게 된다.

4. 구현(Implementation)

4.1 제안 시스템 적용

SODA(Service Oriented Development of Applications)는 서비스 단위의 개발, 조립, 유지보수를 지원하는 아키텍처, 프로세스 및 기술의 총체적 개념이다. SODA에 있어서 CBD는 다양한 형태로 적용될 수 있는데, pre-built 컴포넌트와 패턴들을 이용해서 개발하거나, CBD 개발 플랫폼 상에서 신규로 개발하거나, 기존 시스템을 컴포넌트로 래핑하는 형태, 컴포넌트 제공자들이 제공하는 서비스나 컴포넌트화된 솔루션, 비즈니스 솔루션을 기반으로 개발하는 형태, 패키지 솔루션을 그대로 사용하는 형태 등 개발 속도와 유연성을 조건으로 다양한 적용이 가능하다[6][7][8].

본 논문에서는 기존의 컴포넌트 기반 시스템에 대하여, 컴포넌트를 사용하는 프로세스 컴포넌트(Process Component)를 서비스로 래핑하는 방식으로 구현하였다. 즉, [그림 2]와 같이 비즈니스의 실체를 구현한 EJB 기반 엔티티 컴포넌트들(Category, Forum, Topic, Style)이 위치하는 컴포넌트 레이어와 이러한 컴포넌트들을 사용하는 프로세스를 구현한 컴포넌트(Board Process)가 위치하는 서비스 레이어로 구성된다. Board Process 컴포넌트는 실제로 구현될 서비스 기술에 따라 서비스를 제공하는 역할을 담당할 수 있도록 쉽게 변경될 수 있으며 기능 모듈 변경 시 해당 모듈만 수정하여 프로그램 코드의 유지 보수성을 높임으로써 프로그램의 개발 생산성을 향상시킬 수 있다.



[그림 2] 프로세스와 엔티티 컴포넌트들

서비스 제공자는 게시판의 분류를 관리하는 Category 컴포넌트, 게시판을 관리하는 Forum 컴포넌트, 게시물을 관리하는 Topic 컴포넌트, 게시물의 화면 스타일을 관리하는 Style 컴포넌트로 구성된다. Style 컴포넌트는 웹 기반 환경에서 프로그램 소스를 수정하지 않고도 게시판 관리 화면에서 화면 UI를 직접 지정해 줌으로써 화면 UI 변경을 용이하게 한다.

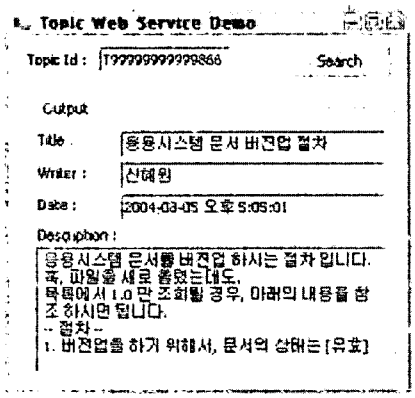
내부 사용자 어플리케이션의 경우 플랫폼과 구현 언어가 동일하므로 서비스 레이어를 거치지 않고 엔티티 컴포넌트들(Entity Components)을 직접 호출하여 사용하며, 외부 사용자 어플리케이션의 경우 Board Process를 통해서 서비스를 호출하면 서비스 명세서(Description)를 통해 서비스 제공자로부터

서비스를 제공받을 수 있다.

```
// Topic 컴포넌트를 호출하는
// 서비스 제공자 EJB 소스의 일부
.....
TopicHome topicHome=
  (TopicHome)ServiceLocator.lookup(
      JndiNames.EJB_TOPIC,TopicHome.class);
.....
//Topic 객체 생성
Topic topic=getTopic();
String topicId=topic.getTopicId();
.....
```

제안 시스템이 상호운용성이 높음을 보이기 위하여 서비스 요청자를 EJB와 이기종 환경인 .NET 환경으로 구성하였으며, 서비스 제공자 측에는 BBS 기능을 가능 모듈들이 구성되어 있고 서비스 요청자는 웹서비스를 이용하여 제공되는 서비스를 사용하는 시나리오로 구현하였다. 서비스를 호출하기 위하여 HTTP를 이용해 SOAP(Simple Object Access Protocol) 메시지를 전송하는 방식으로 접근하였다. 서비스 요청자는 WSDL(Web Services Description Language)로부터 프록시 클래스(Proxy Class)를 생성하고 생성된 프록시 클래스를 이용하여 서비스를 제공받을 수 있게 된다.

```
//wsdl을 통해 생성된
//Topic proxy class의 객체를 생성
Topic topic = new Topic();
.....
TopicListDT value =
topic.getTopic(txtTopicId.Text);
txtTitle.Text = value.title.ToString();
txtWriter.Text = value.writerName.ToString();
txtDesc.Text = value.topicId.ToString();
.....
```



[그림 3] 외부 사용자 어플리케이션 서비스 요청자

4.2 구현 결과

컴포넌트와 서비스를 이용하여 시스템을 구현해 본 결과, 서비스 제공자와 동일한 환경을 가진 서비스 요청자는 컴포넌트를 재사용 함으로써 개발 생산성을 향상시킬 수 있었으며, 이기종 환경에서의 서비스 요청자는 시스템간의 유연성 및 플랫폼 독립성이 보장되어 인터넷, 익스트라넷, 모바일 등 다양한 환경으로 확장하여 사용 가능하다는 점에서 뛰어난 상호운용 능력을 보였다. 즉, 컴포넌트와 서비스를 함께 사용하여 컴포넌트가 가진 장점은 수용하고 단점은 서비스를 통하여 보완함으로써, 시스템을 무한히 확장할 수 있다.

4.3 시스템 환경

제안 시스템은 [표 1]의 기술을 이용하여 구현하였다.

환 경	구현 기술
OS	Unix, Windows XP Professional
Program Language	Java(JDK 1.4), .NET C#
DB	Oracle 8i
WAS Server	WebLogic 6.1

[표 1] 구현 기술

5. 결론 및 향후 연구

본 논문에서는 시스템을 기능 모듈별로 컴포넌트 형태로 구성하고 컴포넌트를 사용하는 프로세스 컴포넌트를 서비스로 래핑하는 방식으로 플랫폼과 구현 언어에 대한 의존성을 제거하여 시스템의 기능을 확장시켰다.

SOA와 CBD를 이용한 웹서비스 모델은 다양한 프로그래밍 언어와 플랫폼을 기반으로 시스템을 구성하는 경우 효과적으로 사용될 수 있다. 그러나 웹서비스는 트랜잭션 처리 규약이 미흡하고, 서비스 제공자와 서비스 요청자 간의 보안이 취약하며, 속도 또한 느리다. 이런 문제로 향후에는 자주 요청되는 서비스에 대해 캐쉬에 담아두어 중복된 서비스 요청을 제거할 수 있는 캐쉬 어댑터를 아키텍처에 추가하여 서비스 응답 속도를 향상시킴으로써 시스템의 퍼포먼스(Performance)를 높일 수 있도록 보완해야 할 것이다.

6. 참고 문헌

[1] George Coalouris, et al, "Distributed Systems", Addison-Wesley, 2000
[2] C.C Chiang, "The use of adapters to support interoperability of components for Reusability", 2002 Elsevier Science B. V., pp. 149 ~ 156
[3] Barrett, Kaplan, Wiledn, "Automated Support for Seamless Interoperability in Polylingual Software Systems", SIGSOFT 1996
[4] David Sprott and Lawrence Wilkes, "Understanding SOA", Sep. 2003
[5] Keith Short, "Component-Based Development and Object Modeling", Sterling Software, Feb., 1997
[6] Allan Brown, "Using Service-Oriented Architecture and Component-Based Development to Build Web Service Applications, white paper published on the Rational Software", 2003
[7] Ali Arsanjani, "Developing and Integrating Enterprise Components and Services," Oct. 2002
[8] "Component-Based Development Helps Enable SODA", Gartner Research, 16 December 2002
[9] Oliver Slim, "Service Oriented Architecture - Part 1, CBDi Journal" 12 March 2003
[10] "Predicts 2003 : SOA Is Changing Software", Gartner Research, 9i December 2002
[11] "Service-Oriented Development : SODA and the Web Service Producer Platform", Gartner Symposium/Itxpo, October 2002