비즈니스 프로세스 중심의 XML 기반 통합 브로커 설계 및 구현

한현철[0] 전자연 윤청
충남대학교 컴퓨터과학과
(alex, kanoe, cyoun)@cs.cnu.ac.kr

A design and Implemetation for XML-based and Business Process-centric Integration Broker

Hyun-chul, Han[0] Ja-Yon Chun Chung Youn
Dept. of Computer Science, Chungnam National University

요 약

기업의 업무환경은 끊임없이 변화하고 있다. 부서중심의 업무 시스템들이 전사(enterprise) 차원의 규모를 거쳐, 최근엔 기업 대 기업(Business-to-Business), 기업 대 소비자(Business-to-Customer) 등 과거에는 상상할 수 없던 규모와 속도로 증대되고 있다. 급속한 비즈니스 환경변화에 시스템의 추가 및 변경 등의 통합을 비즈니스 레벨에서 지원할 수 있는 솔루션으로서 EAI가 각광받고 있다.

본 논문에서는 비즈니스 프로세스 중심의 EAI 솔루션을 XML 기반으로 설계 및 구현해봄으로써, 기존에 존재하던 통합 솔루션들과 EAI와의 차이점을 알아본다. 또한, XML을 이용함으로써 얻을 수 있었던 통합에의 장점을 알아보고 향후 개선방안에 대하여 생각해 본다.

1. 서 론

기업의 업무환경은 끊임없이 변화하고 있다. 1980년대 부서중심의 업무 시스템들이 1990년대에는 전사(enterprise) 차원의 규모로 확대되었으며, 2000년대에 들어서는 기업 대 기업(Business-to-Business), 기업 대 소비자(Business-to-Customer) 등 과거와는 비교가 되지 않는 규모로 확대되어 가고 있다[1]. 이와 같은 업무환경 규모의 확대는 경제 활동의 범위 확장을 동반하고 있으며, 동시에 경제 활동의 속도 증대를 가져왔다.

부서 중심의 소규모 업무 시스템과는 다르게 전사 혹은 B2B 시스템의 경우엔 여러 종류의 다양한 시스템들이 유기적으로 통합되어 운영되는 경우가 많으며, 기업의 사업 목표 변경, M&A, 거래처 변경 등 환경 변화에 따른 다양한 형태의 시스템 상의 변경이 자주 요구된다. 이는 자연스럽게 업무 시스템 간의 유연한 통합에 대한 요구를 불러왔으며, EAI(Enterprise Application Integration)이라는 생소한 개념의 솔루션의 등장을 가져왔다.

2. EAI(Enterprise Application Integration)

시장 조사 전문기관 Ovum에 따르면 EAI의 정의는 다음과 같다.

"EAI는 자체 제작된 혹은 구매한 비즈니스 애플리케이션을 이용하여 상호 간에 이해하고 있는 형태와 맥락에 있는 비즈니스 레벨의 정보를 교환하기 위한 기술과 프로세스의 결합이다."[2]

여기서 잠시 짚고 넘어가야 할 사항은 EAI 이전에도 통합을 위한 시도들이 다양한 형태로 존재해왔다는 사실이다. 분산된 데이터베이스를 통합하여 관리하기 위한 데이터웨어하우스나 분산 객체 간 통합환경인 ORB 등에 대한 연구가 꾸준히 이루어져 왔으며, 이를 통한 성과도 뚜렷히 나타나고 있다.

기존의 통합 기술들과 EAI와의 차이는 통합하려는 대상의 차이에 있다. 기존의 통합기술들이 시스템 레벨의 정보와 프로세스를 통합하려는 시도였다면 Ovum의 정의에서도 나타나고 있듯이 EAI는 비즈니스 레벨의 정보와 프로세스를 교환하기 위한 기술이다.

즉, 기존의 시스템 레벨의 통합 솔루션을 통해서는 빠르게 변화하는 기업 환경의 변화를 시스템에 곧바로 적용시키기 어려움이 많았기 때문이다. EAI와 같은 비즈니스 레벨의 통합 솔루션을 통해 기업의 사업 목표가 바뀌고, 거래처가 변경되고, M&A를 통한 시스템 신규 도입되는 등의 경우에 비즈니스 차원에서의 대처가 가능해질 수 있었던 것이다.

본 논문에서는 비즈니스 차원에서의 통합 솔루션인 EAI를 비즈니스 프로세스 중심으로 설계 구현해 보았으며, 이를 위한 기반 기술로 XML을 사용하였다. 또한 통합할 시스템들 간에 사용할 프로토콜로는 HTTP를 사용하였다.

3. 관련 연구

3.1 워크플로우 관리 시스템

워크플로우 관리 시스템은 애초에 비즈니스 프로세스를 관리하기 위해 탄생한 시스템이다. 기존 시스템들에서는 시스템 구현 코드 내에 함축되어 녹아있던 비즈니스 프로세스 관리 부분을 따로 떼어내어 독립적으로 관리하기 위한 시스템이다. 이를 통해 비즈니스 프로세스의 변경에도 시스템이 유연하게 대처할 수 있게 되는 것이다[3].

EAI를 위해서 빠질 수 없는 것이 비즈니스 프로세스 관리 기술이라면, 이를 체계적으로 관리하기 위한 기술로서 워크플로우는 이전부터 존재해 왔으며, EAI가 각광받는 요즘에 있어 그 중요성이 더욱 부각되고 있다.

일반적으로 EAI환경에서 워크플로우 관리시스템은 프로세스 흐름 관리 서버(Process Flow Control Server), 혹은 프로세스 자동화 서버(Process Automation Server) 등으로 명명된다. 본 논문에서는 이 중, 프로세스 자동화 서버라는 용어를 사용하기로 한다.

3.2 SOAP

HTTP(Hyper Text Transfer Protocol)은 1990년대 이래 인터넷의 폭발적인 발전으로 전세계적으로 가장 보편화된 통신 프로토콜이다. 현재 대부분의 정보 자원들은 HTTP를 통해 접속할 수 있는 위치에 존재하고 있으며,

SOAP(Simple Object Access Protocol)은 기저에 이러한 HTTP를 통신 프로토콜로 사용하여, 원격지의 객체를 호출할 수 있도록 고안된 기술이다. 이를 통해 다양한 객체들을 손쉽게 호출하여 사용할 수 있고, 특히 기존에 방화벽으로 가로막혀 있던 기업 내부의 시스템까지도 HTTP를 통해 호출할 수 있다. SOAP 상에서는 XML 메시지를 통해 원격지의 객체를 실행시키며, 그 실행 결과 또한 XML 메시지로 되돌려받게 되어 있다.

본 논문에서는 SOAP과 거의 유사한 형태의 HTTP 프로토콜 기반 XML 메시징 시스템을 설계하여, 통신 환경을 예측할 수 없는 분산 시스템 간의 상호작용을 가능케 하였다.

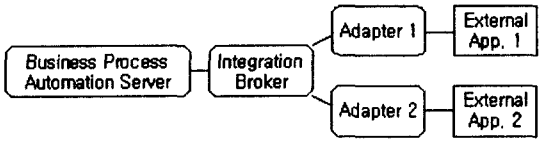
4. 통합 브로커 설계

통합 브로커는 기업 내 여러 시스템 통합을 가능케 하기 위해 프로세스 자동화 서버와 외부 시스템 사이의 호출을 중재해주기 위한 시스템들의 집합이다. 일차적으로 통합 브로커는 프로세스 자동화 서버 내의 비즈니스 프로세스 수행 중에 발생하는 외부 시스템 호출 요구 메시지를 라우팅 해주는 기능을 수행한다. 또한 외부 시스템과 내부 시스템 사이의 데이터 변환의 역할 또한 담당한다.

프로세스 자동화 서버와 통합 브로커는 서로 밀접한 관계에 놓여있기 때문에 EJB의 RMI를 이용하여 통신하며, 데이터의 구조 상으로 보면 XML형식의 메시지를 전달한다. 성능이나 향후 분산 환경 지원을 위해 EJB를 사용하였으나 실제로는 두 서버의 결합도는 높다.

통합 브로커와 외부 시스템 사이의 호출은 HTTP를 이용하여 이루어진다. 외부 시스템의 실제 API를 직접 호출하는 기존의 방식이 아닌 외부 시스템 측에 HTTP 요청을 처리하고 전송한 XML을 해석하여 실제 API를 호출할 수 있는 기능을 수행하는 부분이 추가되어야 한다. 그러한 기능을 수행하는 부분을 일반적으로 어댑터(Adapter)라고 칭한다.

이러한 구조는 일반적으로 EAI를 구축하는 솔루션들에서 일반적으로 채택하고 있는 구조 패턴(Architecture Pattern)으로서, 본 시스템 설계 시 그림1 과 같은 구조 패턴을 적용하였다.



[그림 1] EAI Architecture Pattern[4]

통합 브로커의 호출이 실제적으로 이루어지기 위해서는 호출될 대상 외부 시스템에 대한 기능에 대해 정의가 이루어져있어야 하며, 그 정의 정보를 바탕으로 비즈니스 프로세스를 정의하여야 한다. 이러한 정의작업이 끝나면 정의 정보를 바탕으로 비즈니스 프로세스가 실행되면서 외부 시스템이 실제 호출된다.

정의 시점과 수행 시점에 발생하는 일들을 좀더 상세히 기술해 보면 아래와 같다.

4.1 정의 시점 정보

4.1.1 통합 브로커의 외부 시스템 기능 정의

통합 브로커에서는 일반적인 객체지향 언어에서 클래스를 정의하는 방식과 유사하게 외부 시스템 기능을 정의할 수 있도록 하고 있다. 외부 시스템 기능 정의 정보는 객체와 그에 속하는 메소드, 그리고 각 메소드에 속하는 파라미터 정보들로 구성된다. 이 정의 정보들은 XML로 작성되어 통합 브로커의 정의 정보 관리 서버에 의해 ExecutableObject라는 이름으로 저장된다.

외부 시스템을 객체지향 관점에서 객체 단위로 정의함으로써, 기존에 구조적 프로그래밍 방법 등의 비 객체지향 시스템들 또한 객체지향 관점의 통합이 가능해질 수 있다는 장점이 있다.

4.1.2 프로세스 자동화 서버에서의 외부 시스템 호출 정보 정의 방법

4.1.1에서 정의한 외부 시스템의 기능 정의 정보를 바탕으로 비즈니스 프로세스를 정의할 때, 프로세스 흐름 내에서 외부 시스템을 호출하도록 정의하기 위한 추가적인 정보가 요구된다.

외부 시스템 호출을 위한 추가적인 정의정보 중 가장 중요한 것은 프로세스 자동화 서버 내에서 관리되고 있는 데이터를 외부 시스템에 전달하기 위한 방법이다. 본 논문에서 설계 구현한 프로세스 자동화 서버에서는 프로세스 데이터와 산출 데이터, 이렇게 두가지 정보를 외부 시스템으로 전달할 수 있도록 하고 있다.

4.2 수행 시점 정보

4.2.1 프로세스 자동화 서버와 통합 브로커 간의 호출 방법

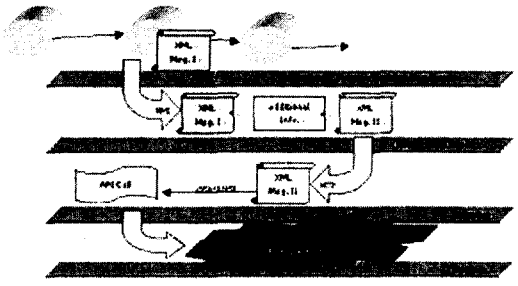
프로세스 자동화 서버는 프로세스 정의정보를 바탕으로 수행된다. 프로세스 정의정보에는 앞서 언급한 바와 같이 외부 시스템 호출을 위한 정의 정보가 포함되어 있는데, 이 정보가 수행시점에는 수행 상태에 맞게 실제 값들이 첨부되어 통합 브로커로 전달되어야 한다.

4.2.2 통합 브로커와 어댑터 간 호출 방법

통합 브로커와 어댑터 간 호출을 위해서는 프로세스 자동화 서버로부터 전달된 XML메시지 중, 필수적으로 요구되는 데이터만 전달한다. 또한, XML 메시지 전달에는 범용적인 통신 프로토콜인 HTTP를 이용하였으며, 따라서 어댑터 측에는 HTTP 요청(Request)를 해

석할 수 있는 웹서버가 존재해야 한다.

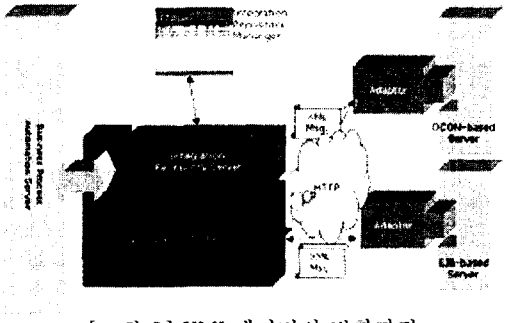
지금까지의 일련의 과정을 그림으로 도식화하면 [그림2]와 같다.



[그림 2] XML 메시지의 변환과정

5. 통합 브로커 구현

[그림 1]에서 기술한 EAI Architecture Pattern을 이용하여 시스템의 구조를 설계하였으며, 실제 시스템의 구조는 [그림3]과 같다.



[그림 3] XML데이터의 변환과정

통합 정의 서버(Integration Repository Server)는 통합할 외부 시스템에 대한 정의 정보를 관리한다. 통합 정의 클라이언트로부터 외부 시스템에 대한 정의 정보가 XML 문서로 산출되어 통합 정의서버로 전달되면, 통합 정의 서버에서는 이를 저장, 관리한다. 이후, 실제 비즈니스 프로세스 수행 시, 통합 브로커는 통합 정의 서버를 통해 외부 시스템 정의정보를 참조하여 프로세스 자동화 서버로부터 전달되는 외부 시스템 호출(XML메시지)를 라우팅한다.

통합 브로커, 통합 정의 서버, 그리고 프로세스 자동화 서버는 EJB(Enterprise JavaBeans) 기반 환경 하에 작성되었으며, EJB컨테이너로는 WebLogic 5.1 서버를 사용하였다.

통합 브로커의 통합 가능성 테스팅을 위하여 MS측 분산 객체 기술인 DCOM(Distributed COM)으로 구현된 서버와 Java 측 분산 객체 기술인 EJB로 구현된 서버를 외부 시스템으로 하여 통합하였다.

각각의 DCOM, EJB기반 서버를 HTTP기반으로 호출하여 통합 브로커로부터 전달한 XML메시지를 해석하여 실제 외부 시스템을 호출하는 기능을 수행하는 어댑터가 각각의 외부 시스템에 부착되어 있어야 한다. 이를 위해 DCOM기반 서버에는 IIS(Internet Information Server)기반 하의 웹 기술인 ASP(Active Server Page)를 이용하였으며, EJB기반 서버에는 Tomcat기반하의 웹 기술인 JSP(Java Server Page)를 이용하였다.

전체 시스템에서 빈번히 발생하는 XML 메시지의 작성과 해석을 위해 Apache의 공개 소프트웨어인 Xerces 1.3.1을 사용하였으며 각각의 서버들에서 데이터를 저장하여야 하는 경우에는 Oracle 8i 데이터베이스를 사용하였다.

6. 결론

본 논문에서는 기업 내 존재하는 다양한 시스템들을 비즈니스 프로세스 관점에서 통합할 수 있는 솔루션인 통합 브로커를 설계 구현하였다. 각 시스템들간 데이터 교환을 위하여 XML 메시지를 활용함으로써, 각 시스템의 기능과 데이터의 구조를 좀더 알기 쉬운 형태로 정의할 수 있었다. 또한 통합 브로커의 주요 기능인 외부 시스템 기능 호출 시에도 XML로 구성한 메시지를 주고 받고 그를 해석하여 실제 API를 호출하는 방식을 취함으로써, 통합 브로커와 외부 시스템 간 결합도를 낮출 수 있었다.

또한, XML 메시지 송수신에 일반적인 통신 프로토콜인 HTTP를 이용함으로써, 외부 시스템들의 각기 예상하기 힘든 환경에 대하여 대처할 수 있게 되었다.

그러나, XML 메시지 교환 방식으로 동기식(Synchronous)을 사용함으로써, 시간이 오래 걸리는 외부 시스템 호출의 경우, 프로세스 자동화 서버에서 불필요한 지체현상이 발생할 가능성이 있다. 이에 대한 해결책으로 제시되고 있는 비동기식(Asynchronous) 프로토콜에 대한 추가적인 연구가 요구된다. 또한, HTTP 뿐만 아닌 다른 통신 프로토콜을 이용해야 하는 경우도 있을 수 있으므로 그에 대한 다양한 대처방안에 대해서도 추가적인 연구가 필요하다.

참 고 문 헌

[1] Accenture, "EAI 시장전망과 ", 2001 EAI Solution Fair, 2001. 6

[2] inews24, "Premium Report : EAI 시장전망과 Solution Guide", inews24, September 1, 2000

[3] David Hollingsworth, "The Workflow Reference Model", WfMC, 1994

[4] Jeffrey C. Lutz, "EAI Architecture Patterns", EAI Journal, March 2000

[5] Thomas Puschmann, Rainer Alt, "Enterprise Application Integration-The case of the Rebert Bosch Group", Proceedings of the 34th Hawaii International Conference on System Sciences 2001, 2001