SOAP 기반의 메시지 Broker 시스템 개발

김용수[0] 주경수
순천향대학교 공과대학 전산학과
{xml[0], gsoojoo}@sch.ac.kr

Developing Message Broker System based on SOAP

Kim Yong-Soo[0], Joo Kyung-Soo
Dept. of Computer Science College of Engineering Soonchunhyang University

요 약

SOAP(Simple Object Access Protocol)은 분산 환경에서의 정보 교환에 사용되는 분산 컴퓨팅 프로토콜로 분산 시스템간에 메시지를 전달하는 방법, 원격 프로시저 호출/응답을 처리하는 방법이 정의하고 있다. 이러한 SOAP 는 텍스트 기반 XML을 프로토콜로 사용하고 있기 때문에 하드웨어 플랫폼, 운영체제, 프로그래밍 언어에 독립적으로 사용할 수 있다. 이러한 이유 때문에 전자상거래 표준인 ebXML 에서도 메시지 전송을 위해 SOAP을 사용하고 있다. 본 논문에서 SOAP을 기반으로 한 메시지 Broker 시스템을 구현하였다. Broker 시스템을 통한 메시지 전송으로 많은 비즈니스 파트너를 통합관리 할 수 있을 것이다.

1. 서론

현재 인터넷은 급속히 발전하였으며 데이터 중심의 마크업 언어인 XML이 전사상거래에서 표준으로 자리를 잡아가고 있다. 전자상거래에서는 수많은 문서들을 XML로 작성하고 이를 기업간에 주고받으려는 노력을 하고 있다. 이로 인해 XML 문서의 표현에 표준을 만들기 위해 많은 단체와 기업들이 노력하고 있는 것이 현실이다[5]. 이러한 현실은 인터넷을 통해 기업과 기업, 기업과 고객의 장벽이 없어진 상황에서 서로 다른 인프라를 바탕으로 하는 시스템을 연결하는 방법이 절실한 현실이다. 그러나 JAVA, CORBA, COM 등의 주요 분산 컴퓨팅 기술은 독자적인 인프라와 기술을 사용하므로 상호 운용 성이 크게 떨어지게 마련이다. 즉, 현재의 분산 컴퓨팅을 구현하는 JAVA, CORBA, COM 등은 목적은 비슷하지만 목적을 구현하는 방법은 매우 다르므로 호환성을 기대하기 어렵다. JAVA는 RMI(Remote Method Invocation)를, CORBA는 IIOP(Internet Inter-ORB Protocol)를, COM은 DCOM(Distributed Component Object Model)을 사용한다. 이들은 지원하는 플랫폼이 다르고 통신 수단 또한 다르기 때문에 이 기종 구성요소간의 호환이 어렵다. SOAP은 분산 환경에서의 정보 교환에 사용되는 분산 컴퓨팅 프로토콜로 분산 시스템간에 메시지를 전달하는 방법, 원격 프로시저 호출/응답을 처리하는 방법이 정의하고 있다[2][3].

본 논문에서 SOAP을 기반으로 한 메시지 Broker 시스템을 구현하였다. Broker를 통한 메시지 전송으로 많은 비즈니스 파트너를 통합관리 할 수 있을 것이다. 2절에서 Broker 시스템에 대해 살펴보고, 3절에는 Broker 시스템을 분석하며, 4절에서는 구현을 살펴본다. 마지막으로 5절에서는 결론 과 향후 연구 과제를 제시한다.

본 연구는 정보통신부의 ITRC 사업에 의해 수행된 것임

2. SOAP 기반의 Broker 시스템

2.1 SOAP 기반 XML 메시지 구조

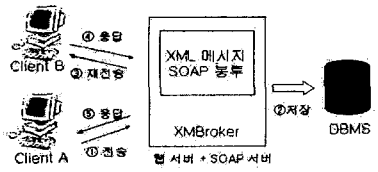
SOAP 메시지는 두 개의 데이타 구조인 SOAP 머리글(header)과 SOAP 본문(body)을 둘러싸는 SOAP 봉투(envelope)로 구성된다. 봉투는 머리글 및 본문을 포함하는 것 외에도 이러한 데이타 구조의 정의에 사용되는 이름 공간을 정의하는 정보를 포함한다. 머리글은 생략할 수 있지만 SOAP 본문에 정의된 요청에 대한 정보를 전달하려는 경우에 사용된다. 예를 들어, 머리글은 트랜잭션, 보안, 컨텍스트 또는 사용자 프로필 정보를 포함할 수 있다. 본문에는 Web Services 요청이나 요청에 대한 응답 XML 형식으로 포함된다. 그림 1은 SOAP 메시지의 상위 레벨 구조를 나타낸 것이다[1][2].

(그림 1) SOAP 메시지 구조

2.2 메시지 Broker 시스템 구조

메시지 Broker 시스템은 Broker 시스템을 통해 다른 클라이언트에 메시지를 전송할 수 있도록 한다. 그림 2는 메시지 Broker 시스템의 구조를 보여준다. Client A가 Client B에 XML문서를 전송하기 위해 Broker 시스템에 메시지를 전송하면 Broker 시스템은 이를 저장하고 Client B에 XML 문서를 전송한다. 전송의 응답이 다시 Broker를 통해 Client A에 전달된다. 이와 같은 방법은 많은 거래 파트너를 클라이언트가 모두 관리해야 하는 일을 줄일 수 있다 또한 서로 다른 XML 문서의 형태를 가지고 있을 경우에 Broker를 통해 적절한 형태로 변형

되어 전송할 수 있다.



(그림 2) Broker 시스템 구조

2.3 메시지 Broker 시스템의 기능

메시지 Broker 시스템에서 서버 기능을 수행하며, 받은 메시지에 대하여 헤더처리, 암호화/복호화 및 보안, 에러/예외처리, 라우팅, 호출, 5가지 기능을 수행한다 [4][6].

2.3.1 헤더처리

메시지를 수신했을 때 메시지 브로커에서 가장 먼저 수행되는 기능 가운데 하나이다. 헤더처리는 수신된 메시지의 헤더 영역을 확인하고, 거기에 포함된 기능 수행을 포함한다. 헤더처리에 특정번호를 헤더에 추가하여 추적 가능하게 하거나 헤더정보가 유효하게 구성됐는지를 검증할 수 있다. XML 메시지 내의 수신자가 적절한지를 처리 과정에서 사전에 검사할 수 있다.

2.3.2 보안

보안 관점에서 메시지 브로커는 보안의 가장 기본적인 조건인 신원확인(authentication), 인증(authorization), 암호화(encryption), 부인 방지(nonrepudiation)를 보장해야 한다. 메시지가 수신되면 메시지 브로커는 우선 디렉토리 서비스나 데이터베이스에 저장된 자료로 신원을 확인한다. 해당 자격을 가진 사용자로 확인이 되면 메시지 브로커는 메시지에 포함된 기능이나 처리에 인증을 받는다.

2.3.3 오류와 예외처리

오류와 예외처리는 메시지 브로커가 수행하는 중요한 기능 중 하나다. 클라이언트가 수신한 메시지가 유효하지 않거나, 요청을 수행할 수 없는 경우에는 에러 메시지를 클라이언트에 보내야만 한다. 그리고 메시지 브로커 시스템의 문제로 인해 서비스를 제공할 수 없는 경우에도 해당 메시지를 클라이언트에 보낸다.

2.3.4 라우팅

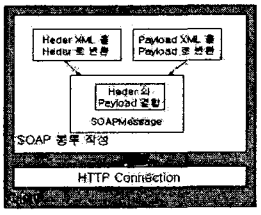
메시지 라우팅은 두 단계로 이루어진다. 하나는 헤더 라우팅으로, 수신된 메시지가 어느 응용 프로그램에서 처리돼야 할지를 결정한다. 다른 하나는 페이로드 라우팅으로 해당 응용프로그램에서 어떤 프로세스나 메쏘드가 사용돼야 할 지를 결정한다.

2.3.5 호출

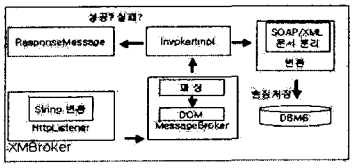
호출 단계에서는 실제 수신 메시지에 있는 페이로드의 자료를 가지고 메쏘드를 호출하게 된다. 여기서는 메쏘드 호출을 통해 브로커에서 클라이언트로 반환할 수행 결과가 만들어진다.

3. Broker 시스템 분석

Broker 시스템의 처리과정은 그림 3, 그림 4와 같다. 클라이언트 측에서는 사용자가 작성한 XML 문서를 SOAPHeder와 SOAPPayload를 이용하여 헤더와 페이로드를 작성한다. 또한 이렇게 Heder와 Payload를 작성한 후 SOAPMessage를 이용하여 Heder와 Payload를 SOAP 봉투화 한다. 이렇게 작성된 메시지를 HTTP 를 이용하여 전송한다. 이렇게 클라이언트가 메시지를 만들어서 Broker 에 보내게 되면, Broker는 다시 최종 수신하는 클라이언트에 전송하여 HttpListener가 받고 이 메시지로 어떤 작업을 처리해야 할지를 결정하는 MessageBroker, 그리고 헤더에 요청된 작업을 실제로 처리하는 Invoker로 이루어진다. 그 이외에 전송 성공 여부를 되돌려 주는 ResponseMessage 가 있다.



(그림 3) 클라이언트 측 메시지 처리 과정



(그림 4) Broker 에서 XML 메시지 처리과정

3.1 SOAPMessage

SOAPHeder와 SOAPPayload를 이용하여 작성한 것을 SOAPMessage에서 구성하게 된다. 즉, 두 개의 서로다른 Heder 파일과 Payload 파일을 하나로 구성하는 기능을 가진다. 이곳에서 작성한 메시지는 connection에게 넘겨준다.

3.2 SOAPConnection

SOAPMessage에서 받은 파일을 HTTP 를 이용하여 전송하게 된다. 전체 메시지를 구성하는 각각의 메시지의 특성들을 Content-Type과 Type, Bounder, Version 등을 통하여 설정할 수 있도록 구성한다.

3.3 HttpListener

HttpListener 은 클라이언트에서 POST 메소드로 넘어온 XML 메시지를 문자열로 받는 서블릿 클래스이다. HttpListner는 XML 문서를 Stream 값으로 받은 다음 Reader로 변환한 후, 스트링으로 읽고, 그 코드를 MessageBroker에 넘겨준다.

3.4 MessageBroker

HttpListner에서 얻어온 값을 이용하여, 파싱해서 DOM을 만들고, invoke()로 DOM을 넘겨서 XML 문서에서 <type> 태그에 있는 텍스트 값을 읽는다.

3.5 InvokerImpl

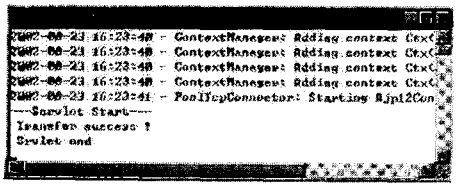
Invoker 인터페이스를 구현한 것인 InvokerImpl 클래스이다. 여기에서는 파일 시스템에 송장을 저장하고, ResponseMessageMaker에서 응답 메시지를 만든다. ResponseMessage Manager는 요청이 성공적으로 이루어 졌을 때는 클라이언트에 보낼 '성공' 메시지나, 실패했을 경우의 '에러' 메시지를 XML 문서로 만든다.

4. Broker 시스템 구현

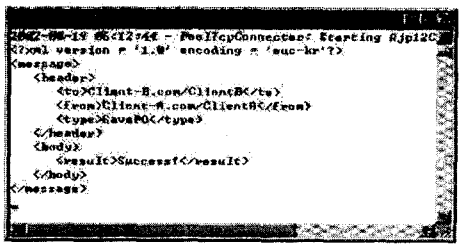
Broker 시스템의 환경으로 운영체제는 Windows 2000 Server, 웹 서버로는 Jakarta Tomcat V3.1.3을 사용하였으며, JDK1.3.1, XML Parser V2를 이용하여 구축하였다. 다음 그림 5는 XML 문서를 이용하여 작성된 송장을 나타낸다. 송장에서 헤더 부분과 페이로드 부분으로 나뉘는 것을 볼 수 있을 것이다. OrderNo를 이용하여 각 송장에 일련번호를 부여하였으며, 송장은 Broker 에 전달되어 저장 시스템에 저장되고 최종 클라이언트에 다시 전송된다.

(그림 5) sample 송장

그림 6은 최종 클라이언트가 메시지를 받을 때 나타나는 화면이다. 메시지 전송이 시작되면 서블릿이 시작되고, 전송이 완료되면 서블릿이 끝나며, 전송 결과는 Broker 전달되어 그림 7과 같은 전송 내용과 결과를 웹 서버의 실행 창에 보여 주게된다.



(그림 6) 최종 Client 의 전송결과



(그림 7) Broker 시스템에 보이는 전송결과

5. 결론 및 향후 연구방향

e비즈니스를 위해 다양한 분야의 기업간 협력이 필요하다. 그러나 실질적으로 기업간의 의사소통은 온라인상의 문서전달을 이용하여 이루어지고 있기 때문에, 기업간의 의사소통을 위해서는 온라인 메시지를 전달할 수 있는 기능을 포함한 시스템이 필요하다. 또한, 기업간 거래는 여러 가지 종류의 교류와 예측하기 힘든 양의 데이터 변화를 필요로 한다. 따라서 많은 플랫폼과 시스템은 서로 중립적인 데이터 교류에 필요한 표준을 필요로 하는데, 이 요구를 만족시켜 줄 수 있는 것이 XML 이다. 데이터를 XML로 표현하고 전송수단을 HTTP 환경의 SOAP을 이용한다면 지금 까지 가지고 있던 여러 가지 문제를 한번에 해결 수 있는 것이다.

본 논문에서는 현재 다양한 방향으로 연구가 진행중인 XML 문서 교환용 Broker 시스템을 설계 및 구현하였다. 이 Broker 시스템에서 전송되는 메시지의 형태는 SOAP에서 정의한 메시지 봉투를 적용하였다. 이를 이용하면 기업간의 문서를 전달하는데 좀더 효율적인 방법을 제공한다. 향후 연구방향으로는 Broker 시스템을 좀더 발전시켜 HUB 시스템을 구현하고자 한다. XML HUB 시스템은 XML 메시지 전달, 메시지 변환, 저장, 비동기전송 등을 지원하며, 기업간의 메시지 전송에서 가장 핵심이 되는 부분으로 발전할 것이다.

참고문헌

[1] Simple Object Access Protocol(SOAP)1.1, http://www.w3.org/TR/SOAP
[2] Simple Objedt Access Protocol(SOAP), http://www.microsoft.com/korea/msdn/workshop/xml/general/SOAP_White_Paper.asp
[3] Bequet, Henry. Professional Java SOAP, Wrox, 2001
[4] Alexander Nakhimovsky, Tom Myers, Professonal Java XML Programming, WORX, 2000
[5] 김채미, 최학열, 글로벌 e비즈니스 리더를 위한 ebXML, 대청미디어, 2001
[6] 김채미, 전문가와 함께가는 XML Camp, 아이트Press, 2001