대용량 E-Mail 처리를 위한 분산 Mail Client의 설계 및 구현

박긴[0] 조동섭
이화여자대학교 과학기술대학원 컴퓨터학과
{kiny, dscho}@ewha.ac.kr

Design and Implementation of Distributed Mail Client for Processing a Great Quantity of Data

Kin Park[0] Dong-Sub Cho
Dept. of Computer Science and Engineering, Institute of Science and Technology in Ewha Womans University

요 약

XML의 등장 이후 인터넷을 이용한 다량의 데이터의 분산 처리 방식이 보다 가속화되고 있다. 이에 본 논문에서는 이러한 분산 처리의 개념을 다수의 수신자에게 같은 내용의 이메일을 보내는 경우에 적용을 하고자 한다. 사용자는 한 사람의 수신자에게 메일을 보내는 것과 같이 메일 에이전트를 통해 메일을 보낼 수 있으나 메일 에이전트는 수신자의 수와 내용의 크기를 토대로 분산되어 있는 메일 시스템인 APEX-MAIL (Application Program Embedded XML-MAIL) 시스템의 각 서버로 메일의 내용과 수신자들의 목록을 XML문서로 만들어 보내며 XML문서를 전송 받은 메일 서버들은 각각의 수신자에게 메일을 보내는 것이다. 따라서 본 논문에서 제안하는 APEX-MAIL 시스템은 대용량의 메일을 보냄에 있어 드는 비용을 효과를 줄여주는 효과를 보인다.

1. 서 론

로컬 네트워크를 이용한 방대한 데이터의 분산 처리 방법에는 여러 가지가 있다. 기본적인 모델로서 서로 다른 머신 오브젝트간에 메시지를 주고받는(Message Passing) 방법이 있는데, 이 방법은 각각의 분산 응용 프로그램을 만들어야 하며 이것은 서로 통신을 할 수 있는 프로토콜의 표준이 없어 로우-레벨의 통신 처리가 반드시 필요하여 그 오버헤드가 크다[1]. 이러한 응용 프로그램 간 데이터 처리 요구는 로컬 영역에서만 그 의미가 있고 수행할 수 있는 것이 아니라, 웹을 이용하여 클라이언트와 클라이언트 간, 클라이언트와 서버 간에도 데이터 처리 요구가 있을 수 있다. 아직 표준화되지는 않았지만 XML 프로토콜은 응용 프로그램 간 데이터 교환을 지원한다. WDDX, XML- RPC와 SOAP등은 XML 프로토콜 표준안의 후보로서 컴포넌트 기반의 모델이다[2].

이 후보안들은 대체로 데이터를 교환할 때 해당 데이터만을 태그로 캡슐화 하여 전달하게 된다. 이 때 태그를 이용하여 쌍방간의 데이터를 직렬화하여 전달하게 되고 데이터를 받은 쪽의 시스템은 태그의 의미 구조를 바탕으로 데이터를 재구성하게 된다. 재구성된 데이터는 시스템에 이미 존재하는 응용 프로그램을 통해 처리가 된다.

그러나 데이터만 전달하는 것이 아니라 그에 적합한 처리를 할 수 있는 응용 프로그램까지 전달 할 수 있다면 데이터 처리 비용 측면에서 볼 때 큰 가치가 있을 것이다. 본 논문에서는 이것을 다수의 수신자에게 동시에 메일을 보내고자하는 웹메일 시스템에 적용하여 APEX-MAIL (Applica -tion Program Embedded XML- MAIL)시스템을 설계하고 구현하고자 한다.

이 논문은 2001년도 두뇌한국21사업에 의하여 지원되었음.

2. 관련 연구

2.1 컴포넌트의 개념

소프트웨어를 개발함에 있어 미리 구현된 블록을 사용하여 소프트웨어 개발비용과 시간을 단축할 수 있는데, 이 블록을 컴포넌트라고 한다. 이 컴포넌트는 실행단위로 개발자에게 인터페이스만을 제공하여 내부의 상세한 부분을 숨기게 되어 보다 쉽고 빠르게 소프트웨어를 개발할 수 있게 해준다[3].

널리 사용되고 있는 컴포넌트로는 Microsoft의 COM /DCOM, OMG의 CORBA, JAVA의 BEAN 등이 있는데 아직은 각각의 컴포넌트 간에 통합된 표준 사양이 존재하지 않아 서로 상호 운용성을 얻을 수 없다[4].

2.2 XML 프로토콜

XML은 HTML의 확장으로 설계되었는데, 서로 다른 이종의 컴포넌트 기반의 시스템을 통합하는 기술로 사용된다. 이는 XML이 플랫폼과 프로그래밍 언어에 독립적이라 최소한의 표준만 갖추어지면 상호 운용성을 획득할 수 있기 때문에 가능한 것이다. XML 프로토콜(XP)은 응용 프로그램 간에 데이터를 교환하는 컴포넌트 모델로 데이터 교환 시 데이터를 태그를 이용하여 캡슐화하고 외부함수를 호출할 수 있게 하며 비구조적 데이터에 대한 직렬화를 지원하며 HTTP 프로토콜을 그대로 사용하여 목적을 달성한다[2].

2.2.1 XML-RPC

XML 프로토콜의 후보안인 XML-RPC(XML- Remote Procedure Call)는 리모트 머신에 있는 메소드를 호출하는 방식이다. XML을 이용하여 리모트 머신에 대한 함수 호출을 캡슐화한 후 리모트 머신에 XML 문서를 전달하고 그 결과값을 XML 문서로 되돌려 받는다. XML-RPC는 기존의 HTTP 프로토콜의 POST 방식을 사용하며 XML 표준을 그대로 따르므로 프로그래밍 언어에 독립적일 수 있다 [5].

이진 데이터의 전송은 Base64 인코딩 방식을 이용하며, 데이터 타입 또한 XML 태그를 이용하여 표현한다. 기존의 HTTP 프로토콜을 XML 바인딩으로 이용할 수 있음으로써, XML-RPC는 방화벽에서의 특정 프로토콜에 대한 거부를 피하면서 분산 환경을 지원할 수 있는 특징을 가진다[6].

그러나 XML-RPC는 자체 데이터 타입을 사용함으로써 데이터 타입 레벨의 상호운용성을 보장할 수 없다.

2.2.2 WDDX

WDDX(Web Distributed Data exchange)는 프로그래밍 언어 레벨에서 복잡한 데이터 구조를 교환하기 위한 방법으로 제안된 XML 프로토콜이다. WDDX는 XML을 이용하여 데이터를 데이터 타입, 변수 이름, 변수 값을 포함하여 직렬화 한다.

하지만 WDDX는 데이터의 직렬화에 대한 방법만을 정의하여 리모트 머신의 메소드를 호출하는 기능이 없어서 XML 프로토콜에서 요구하는 조건을 완전히 만족시키지 못한다. 그리고 XML 프로토콜의 바인딩이 HTTP로만 한정되어 있고 지원하는 데이터 타입 역시 한정되어 있는 단점을 지니고 있다.

2.2.3 SOAP

현재 응용 프로그램을 만드는 데 사용할 수 있는 플랫폼은 많이 있다. 각 플랫폼은 전통적으로 시스템 간 통합을 위해 이진 특성을 갖는 자체 프로토콜을 사용했기 때문에 플랫폼 내의 응용 프로그램들만 데이터를 공유할 수 있다는 제한이 있다. 이러한 제한을 인식하게 되면서 데이터 형식과 데이터 교환의 표준화 작업이 엄청난 속도로 진행되고 있다. 이러한 추이는 전통적인 하드웨어와 소프트웨어 장벽이 자연스럽고 웹 사용 가능한 서비스 통합의 새로운 컴퓨팅 패러다임으로 빠르게 진화할 것이라는 시각에서 유래한다.

이러한 시각의 중심에는 상호 운용성의 개념 또는 종류가 다른 시스템 간에 데이터를 자연스럽게 공유하고 통신할 수 있는 기능이 자리하고 있습니다. 이것이 웹 서비스의 목표이다. 웹 서비스는 표준 인터넷 프로토콜을 사용하여 액세스할 수 있는 프로그램 가능 응용 프로그램 논리 또는 시스템 간 통신과 응용 프로그램 간 통신을 투명하게 하기 위한 웹 지원 표준의 구현이다. SOAP은 XML을 기반으로 하며 시스템 간 통신에 대한 메시징 형식을 설명한다. 또한 메소드 호출(RPC)을 설명하고 HTTP를 통한 SOAP 메시지 보내기를 자세히 설명하는 여러 선택 사항이 있다.

3. Application Program Embedded XML System

3.1 APEX 시스템의 개념

최근의 전자상거래 시스템을 포함한 웹 응용 프로그램들은 사용자에게 많은 정보를 제공한다. 많은 양의 상품 데이터를 여러 가지 기준으로 분류하여 분산된 데이터베이스에 저장을 하고자 할 때, 상품 데이터를 모든 데이터베이스 서버에 전송해야 하고 그 분류를 위한 응용 프로그램 또한 모든 서버에 필요하다. 하지만 이를 위해 일일이 서버에 응용 프로그램을 설치하고 따로 구동한다면 그 오버헤드가 크게 되므로, 응용 프로그램의 바이너리 레벨의 코드와 함께 데이터를 서버에 전송함으로써 그 목적을 달성하고자 하는 것이다. 이러한 데이터 및 응용 프로그램의 전달은 따로 프로토콜을 마련하지 않고 기존의 HTTP 프로토콜을 이용하여 웹 수준에서 이루어질 수 있도록 한다. 본 논문에서는 이러한 시스템을 APEX(Application Program Embedded XML System)이라 명한다.

3.2 APEX-Mail 시스템의 설계

APEX-Mail 시스템의 개념은 그림 1과 같다. 클라이언트가 다수의 수신자에게 메일을 보내는 요구를 한 경우 메일 에이전트는 메시지의 내용과 수신자의 목록을 LB에 전달하고 LB는 그 수신자 수와 메시지 길이를 고려하여 여러 개의 작은 로드로 쪼갠다. LB는 이것을 내부 네트워크로 연결되어 있는 다른 DMS(Distributed Mail Server)에

로드를 분산시키고, 각각의 DMS들은 전달받은 메일 전송 명령을 수행하고 그 전송 현황과 결과를 다시 LB로 전달한다.

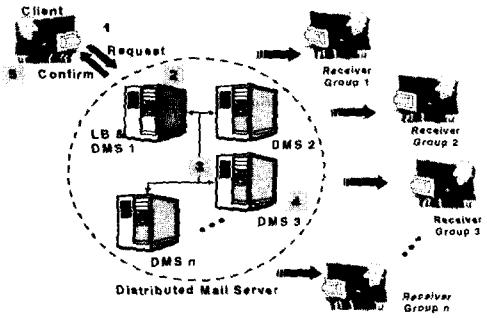

그림 1 APEX-Mail 시스템의 구조

DMS의 상세 구조는 그림 2와 같다. LB-DMS는 DM 서버와 Surfer 에이전트, XML 문서 생성기로 이루어지고, 단말 DMS는 DM 서버와 Surfer 에이전트, XML 문서 분석기로 이루어진다.

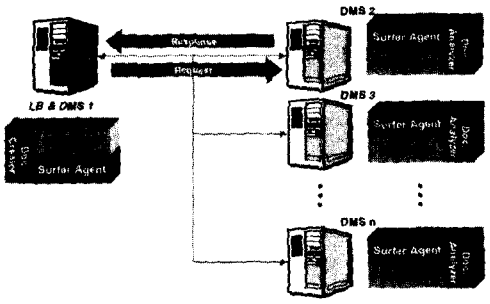

그림 2 DMS(Distributed Mail Server)의 구조

3.2 DMS의 처리 과정

DMS의 처리 과정은 그림 3과 같다.

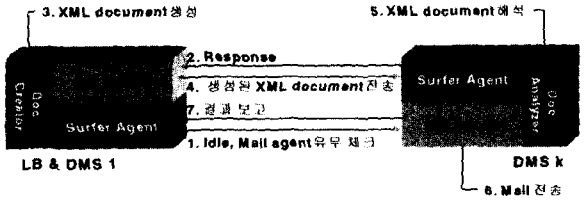

그림 3 DMS의 처리 과정

LB의 Surfer 에이전트가 DMS k의 DM 서버에게 현재 진행하고 있는 작업이 있는지 여부와 특히 메일 에이전트가 존재하는지 여부를 체크하는 메시지를 요구하면 DMS k의 DM 서버는 그 결과를 LB의 DM 서버에게 보내주고 LB에서 XML 문서를 만들어서 다시 DMS k의 Surfer 에이전트에게 전달한다. 전달받은 XML 문서는 Doc 분석기를 통해 해석하고 메일을 수신자에게 전송한 후 그 결과를 다시 LB에 전달하게 된다.

4. 결 론

APEX 시스템은 방대한 양의 데이터를 웹을 이용하여 분산되어 있는 각 분산메일서버(DMS)에서 사용자에 의해 제공된 응용 프로그램이나 DMS에 있던 기존의 응용 프로그램을 통해 처리하여 그 결과 값을 원 데이터와 응용 프로그램과 함께 데이터베이스에 저장하고 또한 사용자에게 HTML 문서 혹은 XML 문서로 되돌려주는 시스템이다.

사용자 제공 응용 프로그램을 이용하여 데이터를 처리하는 경우, SURF 에이전트에서 응용 프로그램의 바이너리 코드와 데이터는 각각 <APEX_code>, </APEX_code>와 <APEX_data>, </APEX_data> 태그로 캡슐화 된다. 전달받은 응용 프로그램의 바이너리 코드는 DMS에서 다시 디코드하여 실행코드를 만들어 데이터에 적용한 후 그 결과를 사용자에게 알리고, 데이터베이스에 저장한다.

APEX-MAIL 시스템은 APEX 시스템을 기반으로 다수의 수신자에게 메일을 보내고자할 때 분산 메일 서버를 이용하여 비용과 시간을 줄이고자 하였다.

[참 고 문 헌]

[1] Dan Grigoras, Stefan Mihaila, "A Framework for Component-Based Distributed Applications Design The CODE: Component Oriented Distributed Environment," Proceeding of the International Conference on Parallel Computing in Electrical Engineering (PARELEC '00), 2000.

[2] W3 Consortium, "XML Protocol Working Group Charter," http://www.w3.org/2000/09/XML-Protocol-Charter, 2000.

[3] Eduardo Pelegri-Lopart, Laurence P.G. Cable, "How to be a Good Bean," Sun Microsystems, Inc. Sep. 1997

[4] Don Box, DevelopMentor Inc, "Lessons from the Component Wars: An XML Manifesto," http://msdn.microsoft.com/xml/articles/xmlmanifesto.asp, Sep. 1999.

[5] Dave Winer, Userland Software Inc, "XML-RPC Specification," http://www.xml-rpc.com/spec

[6] 이경하, 이규철, "XML 프로토콜," 대한정보과학회지 제19권 제1호 통권 140호, 2000년 1월.

[7] Eric Prud' hommeaux, "XML Protocol Comparisons," http://www.w3.org/2000/03/29-XML -protocol-matrix, 3, 2000.