XML-RPC를 이용한 웹서비스 프로그래밍

진 현 수

◈ 목 차 ◈

1. XML이란?

확장 마크업 언어(Extensible Markup Language)의 약자인 XML은 월드 와이드 웹 컨소시엄(W3C)에서 개발했으며,데이터와 문서를 위한 인터넷에 최적화된 형식이다. '마크업'이라 함은 문서 자체에서 문서의 구조를 표현하는 방법을 의미한다. XML은 주로 배포하는 데 쓰이며, 이러한 특성들을 HTML과 공유하는 표준 일반 마크업 언언(SGML,Standard Generalized Language)라고 불리는 표시 언어데 그 근원을 두고 있다. HTML이 인간이 인식할 수 있는 문서이듯, XML은 웹에서 기계가 인식할 수 있는 문서를 작성하기 위해 만들어졌다. 즉, 상호간에 동의한 문법을 사용함으로써, XML 형태를 프로세싱하는 것을 일반적으로 상품화 할 수 있기 때문에 모든 사용자가 이 문서에 접근할 수 있게 되었다.

HTML과 달리, XML은 사전 정의가 거의 없다. HTML 개발자는 요소들(예를 들면 '문법')을 표현하기 위해, 그리고 각 요소들의 이름(head,body와 같은)을 표현하기 위해<,>와 같은 산형 괄호를 사용하는 데 익숙해져 있다. XML은 단지 전자의 특성(즉, 요소를 표현하는 산형 괄호를 사용하는 형식)을 공유할 뿐이다. HTML과 달리, XML은 사전에 정의된 요소들이 없으며, 단지 HTML처럼 다른 언어들을 사용하도록 일련의 규칙을 제공할 뿐이다. XML은 이처럼 사전 정의가 거의 없기 때문에 개발자들이 XML문법을 사용하여, 그 위에 애플리케이션을 쉽게 만들 수 있다. 특별한 알파벳 문자를 사용하고 구두점 기호들을 사용할 수도 있지만, 어떤 언어를 사용할 지는 약간 다른 문제다. 하지만 개발자들이 HTML 개발 경험을 가지고 XML을 접했다면, 자신이 개발한 태크를 어떻게 부를지 선택해야 하는 충격에 미리 대비해야 할 것이다.

XML의 기원이 SGML이라는 사실은 개발자 여러분이 XML의 특성과 설계에 대한 결정을 이해하는 데 많은 도움이 된다. SGML이 핵심적으로는 문서 중심의 기술이지만, XML의 기능은 이를 뛰어넘어 XML-RPC를 포함하는 데이터 중심의 기술까지 확장함을 잘 기억해두기 바란다. 일반적으로 데이터 중심의 애플리케이션은 XML이 제공하는 유연성과 표현성 기능을 모두 필요로 하지는 않으며, XML의 기능 중 일부만 사용하도록 제한한다.

2. RPC란?

RPC(Remote Procedure Call)는 웹보다 훨씬 오래된 기술이다. 네트워크를 통해서 멀리 떨어져 있는 다른 컴퓨터의 함수를 요청한다는 개념은 네트워크가 만들어졌던 역사만큼이나 오래되었지만, 썬은 네트워크를 통해 프로시저를 요청하고 그 결과를 전달받는 일반적인 공식 메커니즘을 만들었다는

* 천안대학교 정보통신학부

공적을 항상 칭찬받고 있다. RPC는 1990년대까지 주된 프로그래밍 방법이었던 프로시저 접근 방법에 가장 잘 맞았다.

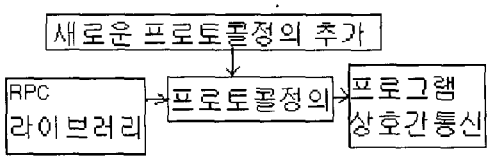
우리들이 물리 운동량을 측정하는 프로시저를 개발한다고 가정해보자. 이 함수는 한 물체의 이름과 속도를 알고 있다. 하지만 운동량을 측정하기 위해서는 그 물체의 질량을 알아야 한다. 따라서 그 물체의 질량을 리턴해주는 프로시저를 호출해야 한다. 이 작업은 '로컬 프로시저 호출'에서는 상당히 간단하다. 서로간에 호출하는 프로시저(또는 함수나 메소드)로 프로그램을 분리해주면 된다. 구문은 다르지만, 일반적으로 매개변수 값을 넘기고 그 결과를 전달받는 형식으로 이루어진다.

```
Mass = getMass(objectID)
```

이제 원격 시스템에서 수행되는 getMass()함수를 생각해보자. 이 경우, 프로시저를 요청한 측에서 복잡한 프로세스에 대해 더 많은 사항을 알아야 한다. 어떤 원격 시스템을 연결할 것인지, 어떻게 패키징하고 어떻게 매개변수를 넘길 것인지, 결과는 어떤 방법으로 전달받을 것인지, 그리고 그 값을 요청했던 루틴에게 결과 값을 어떻게 패키지화하지 않고 전달할 것인지 등을 프로그램이 알도록 해야 한다.

RPC 접근 방법은 네트워크를 통해서 연결되기 때문에 지연 가능성뿐만 아니라, 연결되는 양측이 메시지를 생성해서 프로세싱하는 라이브러리를 가져야 하는 많은 오버헤드를 가지고 있지만, 대신에 분산 처리 및 정보의 공유를 가능하게 해준다.

RPC 접근 방법은 프로그래머들이 하위 레벨의 프로토콜, 네트워킹, 그리고 다양한 구현 과정을 상세하게 알아야 하는 부담을 덜어주기 때문에 개발 과정이 쉬워진다. RPC 라이브러리는 비교적 투명하게 설계되고, 복잡한 API보다는 간단히 하나의 함수 호출로 동작하는 경우가 많다. RPC 구현 개념은 개발자들에게 또 하나의 이점을 준다. RPC 시스템에서 동작하기 위해서는 정의된 프로토콜을 사용해야 하는데, 서로 다른 환경을 지원하도록 프로토콜을 새로 정의해 만들 수 있다. 따라서 서로 다른 업체의 메인 프레임, 컴퓨터, 워크스테이션,



(그림 1) RPC 구현 개념

PC 등에서 개발된 프로그램들은 공유되는 네트워크만 있으면 상호간에 통신할 수 있다.

실제로 RPC는 네트워크를 통해서 호출되도록 인터페이스를 정의하는 메커니즘을 제공한다. 이 인터페이스는 하나의 함수를 요청하는 것처럼 간단할 수도 있고, 커다란 API처럼 복잡할 수도 있다. RPC는 유용한 메커니즘으로서, 네트워크 오버헤드와 구조적인 면을 고려해야 하는 점을 제외하고는 개발자들이 원하는 많은 장점을 가질 수 있다.

3. XML-RPC와 웹 서비스

XML-RPC는 개발자가 선택할 수 있는 유일한 것이 아니고, 또한 아닐 것이라는 사실은 분명한 일이지만, 오랫동안 솔루션으로써 지속될 것 또한 분명해 보인다. XML-RPC가 가장 유연하거나 효율적인 XML 프로토콜은 아니지만, 간편성과 기존의 아키텍처와의 통합, 그리고 안전성 면에서는 다른 프로토콜과 비교되는 중요한 이점을 가지고 있다. XML-RPC는 현재 개발된 지 3년 정도 되었으며, 앞으로 커다란 변화가 있을 것 같지는 않다. 새로운 프로토콜을 창안하고 있는 W3C와 IETF 활동 위원회는 간편성에는 별로 신경을 쓰지 않으면서 몇몇 상황에서 좀 더 유용하게 사용할 수 있는 툴을 생성하고 있지만, XML-RPC와 필수적인 경쟁 관계에 있지는 않은 것 같다. XML-RPC는 현재 우리가 사용할 수 있는 솔루션이며, 안정적이고 앞으로도 오랫동안 특정한 상황의 문제를 해결하는 데 이용될 것이다.

XML-RPC가 모든 문제점을 풀 수 있다고 주장하지 않기 때문에, XML-RPC를 사용할 것인지 판단하는 것은 꽤 간단하다. 일반적으로 어떤 종류의 서비스가 XML-RPC에 맞는지 알아보는 것은 그리

어렵지 않다. 이 프로토콜은 네트워크의 기능을 보강하는 단순하며 간단한 계층이므로 XML-RPC 솔루션을 구현한다고 해서 다른 가능성을 방해하지는 않는다. 많은 프로젝트에서 개발 필요성에 의해 다른 프로토콜(SOAP, RMI나 CORBA)을 사용하면서, 여러 종류의 라이브러리가 곳곳에 있고 사용의 편리성 때문에 XML-RPC를 같이 구현하고 있다. 같은 노력을 반복할 필요 없이(대부분이 우리가 이 책에서 보았던 패키지들을 이용해서 줄일 수 있기 때문에) 각 작업에 맞는 툴을 선택함으로써 여러 언어로 프로그램을 작성하는 것은 그리 어렵지 않다.

여러 개의 패키지가 존재하게 되면, 그것도 XML-RPC처럼 약식으로 쓰여진 규약에 대한 패키지들이라면 당연히 상호 운영성에 대한 의심이 생기게 된다. 그래서 XML-RPC 웹사이트는 테스팅에 대한 기반을 제공하고자 http://validator.xmlrpc.com/에서 XML_RPC 유효검사기(validator)기능을 제공하고 있다. XML-RPC 메일링 리스트 (http://groups.yahoo.com/group/xml-rpc)에서는 개발자들이 상호 운영성 이슈를 논의할 수 있는 포럼을 제공하고 있다. 일반적으로 개발자는 그들이 추가한 여분의 특성을 문서화했으며, 이러한 특성이 잠재적으로 다른 구현물과 상호 호환되지 않을 수 있다고 표시했다. 이 문서를 읽고 중요 특성을 사용한다면, XML-RPC를 사용한 프로그램 상호간 호환성을 지킬 수 있을 것이다.

또한 어떤 프로시저를 이용할 수 있고, 그 프로시저가 어떤 매개변수를 취하는지 설명하는 방법으로 기본적인 프로시저 호출 메커니즘을 보강함으로써 XML-RPC서비스의 표준 라이브러리를 만들 수 있는 여지가 있다. WSDL은 웹 서비스를 설명할 수 있는 수단을 제공하지만 현재 XML_RPC보다는 SOAP과 HTTP GET및 PUT방식, 그리고 MINE 인코딩 전송에 초점을 맞추고 있다. XML-RPC의 PHP구현물은 XML-RPC에서 위와 같은 서비스를 유사하게 제공하는 내부검색 서비스를 포함하고 있지만, 이러한 메소드들은 현재 PHP 구현물에서만 작동한다.

XML-RPC 규약은 상대적으로 간단한 요구 사항을 충족시켜주고 있으므로, XML-RPC 규약은 앞으로도 거의 변화하지 않을 것 같다. 하지만 그렇게 좀처럼 변하지 않는 규약임에도 불구하고 새로운 XML-RPC 라이브러리가 계속해서 나오고 있으며, 새로운 환경에서 XML-RPC를 사용하기 위해 많은 작업들이 진행 중이다.

4. 결론 - 웹 서비스 전망

현재 WWW(World Wide Web)은 이해하기 쉬우며, 인간에게 컨텐츠를 보여주고, 자료를 찾아 항해할 수 있고, 어느 정도 상호 작용 할 수 있는 저렴한 방법을 제공하고 있다. 웹 기반 구조인 TCP/IP와 HTTP, 그리고 HTML은 폭넓게 개발되어 왔고, 현재까지 지원받고 있으며, 이해되고 있다. 이러한 기반 구조에 약간의 제약 사항이 있지만, 웹의 폭 넓은 이용 가치로 인해 저렴하게 재활용할 수 있는 필수품이 되었고, 웹 서비스 커뮤니티에서는 이것을 XML-RPC가 선도하고 있는 컴퓨터간 통신의 모델로 도입하여 좀 더 폭넓고 다양한 아키텍처와 프로젝트로 확장할 계획이다.

컨퍼런스에서 발표자가 네트워크를 거쳐 로직을 배포하는 해결책이라고 떠들어대고, 애플리케이션 서비스 제공자를 위한 새로운 사업 모델의 기반이라고 강조되어온, 웹 서비스는 그 심장부에 매우 간단한 툴 셋을 지니고 있다. 그 기반은 XML과 HTTP이며, 이 두 개의 성분은 컴퓨터간 통신을 위해 결합되었다. 웹 서비스의 비전이 XML-RPC가 제공한 기본 통합 개념을 넘어섬에 따라 웹 서비스와 관련된 많고 다양한 종류의 규약 또한 발전하고 있다. 현재의 규약들은 객체 정보를 컴퓨터간에 전달하는 프로토콜 및 서비스를 검색할 수 있는 API, 그리고 서비스를 설명하는 언어등을 포함하고 있다.

XML-RPC는 오늘날 실제적으로 사용 가능한 애플리케이션에 적용할 수 있는 통합 솔루션이다. 웹 크로싱은 스레드 기반의 그룹 토의가 가능하도록 설계된 또 다른 컨텐츠 관리 시스템이다. 이것은 또한 서로 사이트에 있는 웹 크로싱 서버간에 정보와 사용자 인증서를 교환할 수 있는 방법을 제

공하고 있다. 웹 크로싱의 사용자 인증서 전달 메커니즘을 이용해 간단한 애플리케이션을 작성하면, 한 사이트에 로깅한 사용자가 다른 웹 크로싱 사이트로 자신의 인증서를 전달할 수 있게 된다. 또한 트러스트 관계를 애플리케이션 수준에서 맺을 수 있다. 이 인터페이스를 흥미롭게 이용하는 또 하나의 방법은 다른 웹 크로싱 사이트와 정보를 주고받는 일정 관리 시스템으로 사용하는 것이다.

컨텐츠 관리 시스템만이 XML-RPC 서비스를 사용한 것은 아니다. 오라일리의 미어캣 와이어 서비스(Meerkat Wire Service)또한 5장에서 자세히 설명되었던 웹 서비스 인터페이스를 제공하고 있다. 간략하게 말해, 미어캣은 Rich Site Summary 파일을 많은 웹사이트로부터 모아, 사용자가 연속된 웹 인터페이스를 통해 모아진 컨텐츠를 볼 수 있도록 한다. 이 애플리케이션에서 흥미 있는 점은 이것의 고안자인 라엘 돈 페스트가 다른 애플리케이션이 미어캣의 데이터에 접근하길 원한다는 것을 이해했다는 것이다. 돈페스트는 전통적인 방법인, 미어캣으로 페이지를 요청하여 HTML을 파싱함으로써 원하는 데이터를 추출하는 방식으로 이 애플리케이션을 만드는 대신, 이 애플리케이션의 데이터를 좀 더 친근한 프로그램적인 방법으로 나타내도록 URL 행의 매개변수를 만들고 문서화하였다. 라엘은 XML-RPC를 학습한 후 바로 이 기능을 웹 서비스 프로토콜로 전환하였다. 미어캣의 데이터는 읽기 전용이기 때문에, 인증이나 보안 문제도 이 공용 서비스에 있어 특별한 주의를 필요로 하지 않는다. 명확하게 공용이라 말할 수는 없지만, KDE 데스크탑 환경 버전 2.0은 KDE애플리케이션에서 XML-RPC 스크립팅을 지원하고 있다. 마이크로소프트가 애플리케이션 기능을 COM 객체에 노출시킨 방법과 매우 유사하게, KDE에서는 컴포넌트 모델을 DCOP이라고 부르며 비슷한 방법으로 사용한다. 사용자는 C++로 XML-RPC 서버를 작성해 어떤 XML-RPC 클라이언트에서도 DCOP으로 호출할 수 있도록 할 수 있다. 대부분의 데스크탑 사용자는 임의의 인터넷 사용자가 정해지지 않은 시간에 자신의 컴퓨터에 있는 Konquor를 열기를 원하지 않지만, 이러한 XML-RPC/DCOP브리지를 구현한 알맞은 애플리케이션을 이용한다면 작업 그룹간에 달력을 일치시킬 수 있을 것이다. 또한 KDE사용자는 KOrganizer 애플리케이션을 사용해 자신의 웹 크로싱 달력을 업그레이드할 수 있을 것이다. 이러한 가능성들은 참으로 흥미진진하다.

소프트웨어 개발자들이 점차적으로 웹 서비스를 인식함과 동시에 공용 XML-RPC 애플리케이션 인터페이스의 수는 증가하게 될 것이다. XML_RPC.com에서 웹 서비스가 탑재된 애플리케이션의 최신 리스트 정보를 확인할 수 있다.

참 고 문 헌

[1] 윤성우 "TCP/IP 소켓 프로그래밍"
[2] 주종면 "오라클 9i SQL & PL/SQL"
[3] 김세환 "최적화를 생각하는 PHP프로그래밍"

◐ 저 자 소 개 ◑

진 현 수

1986년 서울시립대학교 전자공학과(학사)
1990년 서울시립대학교 전자공학과(석사)
2000년 서울시립대학교 전자공학과(박사)
1990년~1995년 서울시청 총무과 근무
1996년~2000년 안산공과대 조교수
2000년~2001년 한국과학기술원 박사후 과정
2001년~현재 천안대학교 정보통신학부 근무
2001년~현재 한국인터넷정보학회 학술지 편집위원 근무