한국컴퓨터정보학회 동계학술대회 논문집 제20권 제1호 (2012. 1)

안드로이드 애플리케이션을 위한 XML 파서 성능비교

정길현°, 이종진**, 이진원***
장안대학교 인터넷정보통신과
e-mail:{khjeong°, jjin**}@jangan.ac.kr, salsalyi2@naver.com

Performance Comparison of XML Parsers for Android Applications

Kil Hyun Jeong°, Jong Jin Lee**, JinWon Lee***
Dept. of Internet Communication, Jangan University

● 요 약 ●

모바일 애플리케이션의 다양한 개발 방법 중에서 파서는 중요한 요소로써 쓰이고 있으며 그 종류에는 여러 가지가 있다. 또 각 파서마다 데이터를 파싱하는 방법이 모두 다르며 구현되는 구조 또한 다르다. 본 논문에서는 이렇게 여러 가지 형태로 구현되는 파서를 분석하여 웹 서버를 거쳐 데이터베이스에 접근하는 방식을 좀 더 신속하고 효율적인 구조로 구현하고자한다. 구현방법으로는 파서의 성능 비교를 웹 전송부분을 제외한 파싱 속도를 측정한 방법과 웹 전송부분을 포함한 파싱 속도를 측정하는 두 가지 방법을 통하여 비교하였다.

그 결과, 웹 전송부분을 제외한 방법에서는 DOM 파서가 가장 좋은 성능을 보여주었고 웹 전송부분을 포함한 방법에서는 SAX 파서가 가장 좋은 성능을 보여주었다. 이러한 결과는 안드로이드 애플리케이션에서 웹 서버를 경유하여 데이터를 가져와 파싱하는데 사용할 파서를 선택하는데 도움을 줄 수 있다.

키워드: XML 파서(Parser), 성능비교(Performance Comparison), 안드로이드 애플리케이션(Android Applicition)

I. 서론

현대는 스마트폰의 등장으로 모바일 시장이 급속도로 성장 하고 있는 상황이다. 스마트폰의 각종 애플리케이션의 등장과 그 파급 효과로 인해 기업과 개인의 애플리케이션 개발에 대한 관심 및 수요가 증가 하고 있으며 애플리케이션 개발에는 다양한 기술들이 쓰이고 있다. 특히, 데이터베이스 접근에 있어서 애플리케이션에서 직접적으로 데이터베이스에 접근하는 방법으로 Android에서 JDBC를 사용하는 것을 설계적 관점에서 잘못된 접근 방식으로 보며 수행성, 가용성, 관리성, 신뢰성, 보안성 등을 보장할 수 없다고 하기 때문에 현재는 중간의 웹 서버를 거쳐 데이터를 가져오는 방식을 사용하고 있다. 이 때 웹 서버에서 가져온 데이터를 애플리케이션 쪽에서 사용하기 위한 목적으로 쓰이는 핵심적인 요소 중 하나가 Parser라는 것이다.

현재 다양한 방식의 파서들이 존재하며. 파서 별로 여러 가지의 성능과 구현 방법을 가지고 있다. 이렇게 다양한 파서들 중에 웹 서버를 경유하여 데이터베이스에 접근하는 방법이 안드로이드 애플리케이션 개발 시 응용 환경에 가장 적합한 파서를 적용하는 것이 애플리케이션 전체 시스템 성능을 좌우하는 중요한 요소로써 작용한다[1~2]. 이런 이유로, 본 논문에서는 이러한 파서를 비교하여 애플리케이션에서 최적의 성능을 낼 수 있는 파서를 찾기위해서 안드로이드 애플리케이션과 JSP를 사용하는 웹 서버와 MySQL서버를 연동하여 게시판을 생성하고 그 게시판에서 DOM(Document Object Model) , SAX(Simple API for XML), JSON((JavaScript Object Notation) 파서들을 사용하여 데이터를 데이터베이스에서 가져와 파싱하는데 까지 걸리는 시간을 비교하고 그 결과를 분석함으로써 각각 파서의 장점과 단점을 파악하고 각각의 상황에서 최적의 성능을 낼 수 있는 Parser를 제시하고자 한다.

II. 관련 연구

1. DOM

DOM은 객체 지향 모델로써 구조화된 문서를 표현하는 형식으로 플랫폼/언어 중립적으로 구조화된 문서를 표현하는 W3C의 공식 표준이다. DOM은 또한 W3C(World Wide Web Consortium)가 표준화한 여러 개의 API의 기반이 된다. DOM은 HTML 문서의 요소를 제어하기 위해 웹 브라우저에서 처음 지원되었다. DOM은 동적으로 문서의 내용, 구조, 스타일에 접근하고 변경하는 수단이었다. 브라우저 사이에 DOM 구현이 호환되지 않음에 따라, W3C에서 DOM 표준 규격을 작성하게 되었다. DOM은 문서의 기반이 되는 데이터 구조에 제한을 두지 않는다. 잘 구조화된 문서는 DOM을 사용하여 트리 구조를 얻어낼 수 있다. 이 같은 구현에

서는 문서의 전체 내용이 해석되어 메모리에 저장되어야 한다. 이런 특성때문에 DOM은 문서 요소가 임의적으로 접근되고 변경할 수 있어야 하는 응용 프로그램에 가장 적합하다.

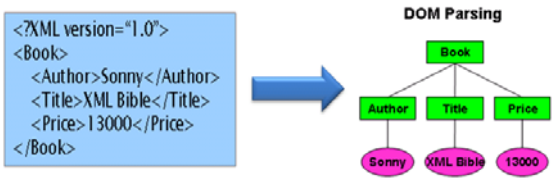


그림 1. DOM 구조
Fig. 1. DOM Architecture

2. SAX

SAX는 XML 파일을 해석하기 위해 DOM 대신 사용되는 것으로 XML문서를 애플리케이션에서 사용하기 위한 API이다. SAX는 DOM에 비해 단순한 인터페이스를 갖고 있으며, 처리해야할 파일이 많거나 큰 경우에 적합하다. 그러나 데이터 내용을 조작할 수 있는 기능은 상대적으로 적다고 볼 수 있다. SAX는 이벤트 중심의 인터페이스이다. 프로그래머가 일어날 수 있는 이벤트를 설정해 놓으면, 그 이벤트가 일어났을 때 제어권을 가지고 상황을 처리한다. 그러므로 XML 파서와 함께 배포되는데 예를 들어서 Apache의 Xerces도 SAX 클래스를 포함하고 있다[3]. SAX는 다음과 같은 상황에 적용이 가능하다. 큰 문서를 핸들링 하고자 할 때, 이벤트 스트림에서 문서를 필터링하고자하거나 문서 자체에 대한 변경이 필요하지 않을 때 이런 경우에 SAX Parser를 적용하여 구현할 수 있다.

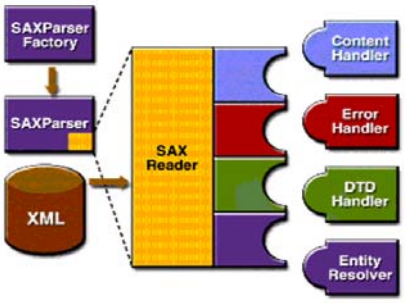


그림 2. SAX 구조
Fig. 2. SAX Architecture

3. JSON

자료의 종류에 큰 제한은 없으며, 특히 컴퓨터 프로그램의 변수 값을 표현하는 데 적합하다. 그 형식은 자바스크립트의 구문 형식을 따르지만, 프로그래밍 언어나 플랫폼에 독립적이므로 많은 언어에서 이용할 수 있다. JSON은 텍스트로 이루어져 있으며 사람과 기계 모두 읽기 쉽고 쓰기 쉬워서 프로그래밍 언어와 플랫폼에 독립적이다. 또한, 서로 다른 시스템 간에 객체를 교환하기에 좋으며 자바스크립트의 문법을 채용했기 때문에, 자바스크립트에서 eval 명령으로 곧바로 사용할 수 있다. 이런 특성은 자바스크립트를 자주 사용하는 웹 환경에서 매우 유리하다. 파이어폭스, 인터넷 익스플로러, 오페라, 사파리, 구글 크롬 등 대부분의 최신 웹 브라우저는 JSON 전용 파서 기능을 내장하고 있으므로 이런 기능을 사용하는 것이 더 안전할 뿐만 아니라 빠른 방법이다[4].

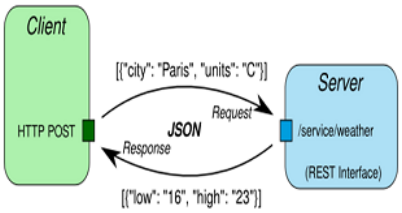


그림 3. JSON 구조
Fig. 3. JSON Architecture

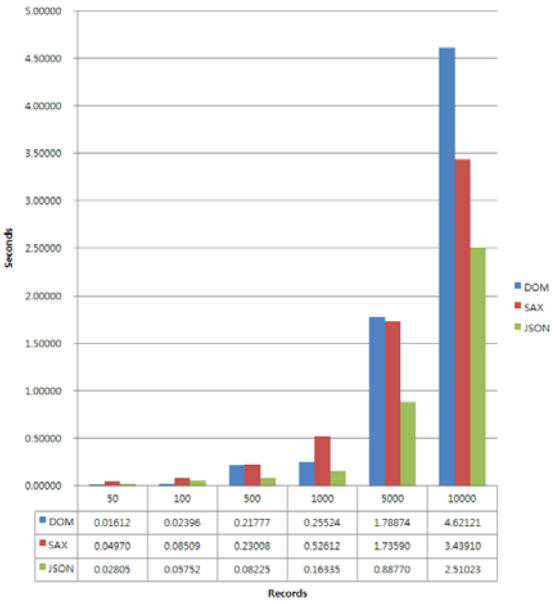
III. 본론

성능 측정은 두 가지 방법으로 진행하였다.

첫 번째 방법은 웹 서버에서 전송되는 시간을 제외하고 XML 형태의 데이터를 파싱하여 ArrayList형태로 변환하는데 까지 걸리는 시간을 파서 별로 측정한 방법이다.

두 번째 방법은 웹 서버로 부터 받은 XML형태의 데이터를 파싱하여 ArrayList형태로 변환하는데 까지 걸리는 총 소요 시간을 파서 별로 측정하였다.

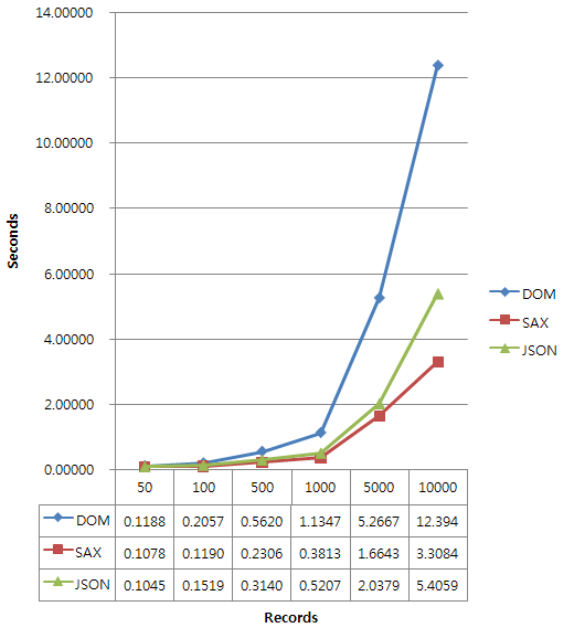
성능을 측정하는 두 가지 방법은 데이터의 레코드수를 증가시키면서 데이터 양의 변화에 따른 시간을 측정하였다. 측정 데이터의 신뢰성을 높이기 위하여 측정을 여러 번 수행하여 그 값을 평균을 구하여 데이터를 산출하였다.



	50	100	500	1000	5000	10000
DOM	0.01612	0.02396	0.21777	0.25524	1.78874	4.62121
SAX	0.04970	0.08509	0.23008	0.52612	1.73590	3.43910
JSON	0.02805	0.05752	0.08225	0.16335	0.88770	2.51023

그림 4. 웹 전송부분을 제외한 파싱 소요시간 측정

<그림4>에서와 같이 웹 전송부분을 제외한 결과, 초기에 레코드 수가 적을 경우에는 다른 파서보다 DOM 파서가 가장 빠른 시간에 파싱을 하는 것으로 측정되었지만 레코드수가 증가하면 할수록 성능이 급격하게 저하되어 다른 파서들 중 가장 낮은 성능을 보여준다. 이것은 DOM 파서는 XML을 파싱하여 문서의 내용을 트리구조로 생성하기 때문에 내용이 증가하면 사용되는 메모리가 증가되어 성능이 급격하게 떨어지는 것으로 보인다. 이렇게 레코드수가 많은 작업을 하는 경우에는 XML형태가 아닌 JSON 파서의 자체 포맷을 이용하여 데이터를 파싱하는 것이 가장 좋은 성능을 낼 수 있는 방법으로 사료된다.



	50	100	500	1000	5000	10000
DOM	0.1188	0.2057	0.5620	1.1347	5.2667	12.394
SAX	0.1078	0.1190	0.2306	0.3813	1.6643	3.3084
JSON	0.1045	0.1519	0.3140	0.5207	2.0379	5.4059

그림 5. 웹 전송부분을 포함한 파싱 소요시간 측정

<그림5>에 웹 전송부분까지 모두 포함한 파싱 소요시간을 측정한 결과를 나타내었다. 그림에서 보는 바와 같이 DOM파서는 레코드 수가 증가 할수록 급격한 성능저하를 보여주었고, 가장 좋은 성능을 보여주는 파서는 SAX 파서라는 결과가 나왔다. SAX 파서는 XML문서를 순서대로 읽으면서 이벤트를 발생시키는 방식으로 데이터를 처리하여서 메모리를 거의 사용하지 않기 때문에 기동 속도가 빠른 특징이 있다.

Ⅳ. 결론

본 논문에서는 두 가지 방법을 사용하여 파싱 기술의 성능을 비교하여 애플리케이션 구현에 적합한 파서에 대하여 알아보았다.

안드로이드에서 파서를 이용하여 애플리케이션을 구현할 때는 대부분 웹 서버를 연동하여 구현하는 형태를 취하기 때문에 SAX 파서를 이용하여 기능을 구현하는 것이 최적의 성능을 낼 수 있는 방법이다. 하지만 SAX 파서는 XML문서에 순차적인 접근을 하기 때문에 문서 일부분에 대한 임의접근이 불가능 하다. 이렇게 본 논문에서 도출한 결과는 향후 안드로이드 애플리케이션에서 웹 서버를 경유하여 데이터를 가져와 파싱하는데 사용할 파서를 선택하는데 도움을 주어 시스템의 전체 성능을 향상 시킬 수 있는 지표로써 안드로이드 애플리케이션 개발에 도움을 줄 것이다.

참고문헌

[1] developer.com, “Android XML Parser Performance”, *http://www.developer.com/xml/article.php/10929_3824221_2/Android-XML-Parser-Performance.htm*

[2] “A Performance Evaluation of XML Parsers Supporting Java API”, J. Ind. Sci., Cheongju Univ. Vol. 27, No. 1, 2009.

[3] Li Ya Dong, “Performance Analysis of XML Parsers for Developing XML Applications”, 2009.

[4] “JSON vs XML Parsing”, *http://bolder00.tistory.com/2*