모바일 환경에서 효율적인 SVG 전송 방법

*이성재° **유남현 **김원중
*순천제일대학, **순천대학교 컴퓨터과학과
kwj@sunchon.ac.kr

The Efficient SVG Transmission Method on Mobile Environment

*Sungjae Lee° **Namhyun Yoo **WonJung Kim
*Suncheon First College, **Dept. of Computer Science, Sunchon National University

요 약

모바일 기기가 가지는 다양한 화면 사이즈 등을 고려한 정보 서비스를 제공하기 위해서는 비트맵 기반의 JPG나 GIF 보다는 벡터 기반의 SVG, Flash Lite와 같은 이미지 포맷이 더 적합하다. 대부분의 모바일 기기의 경우 작은 메모리, 협소한 네트워크 대역폭, 불안전한 네트워크 연결 등과 같은 제약성을 가지고 있기 때문에 모바일 기기와 서버 간의 정보나 데이터의 양과 전송 시간을 단축시키기 위한 다양한 연구들이 진행되고 있다. 이에 본 논문에서는 모바일 기기에서 정보를 표현하기 위하여 사용되는 SVG와 같은 XML 파일들을 효과적으로 전송하기 위한 방법을 제안한다.

1. 서 론

모바일 기기에서 정보 서비스를 제공하기 위해서 사용되는 그래픽 파일 포맷은 다양한 화면 사이즈에 맞는 UI의 설계, 콘텐츠의 제작, 입력 도구의 불편함 등을 고려해야 한다. 따라서 PC 환경에서 주로 사용되고 있는 라스터 이미지인 BMP, JPG, GIF보다는 벡터 기반의 이미지가 더 적합하다. 모바일 기기에서 사용 가능한 벡터 이미지로는 매크로미디어사의 Flash Lite[1]와 W3C에서 제안한 SVG의 서브셋인 SVGT가 있다. SVG는 XML의 한 종류로서 W3C에서 제안한 2D 그래픽 표준 이미지 포맷이다[2]. 바이너리 형태의 Flash Lite와 달리 SVGT는 텍스트 형태로 구성되어 있다. 특히 일본의 KDDI에서는 SVGT를 이용한 LBS(Location Based Service) 기반의 서비스 등 다양한 서비스를 현재 제공하고 있다.

모바일 기기의 경우 PC 기반의 환경과 달리 협소한 네트워크 대역폭, 불안정한 네트워크 연결성, 작은 메모리 등의 다양한 제약이 있기 때문에 모바일 기기와 서버 간에 실행 파일 등을 교환하는 경우 [3],[4],[5],[6]과 같은 방법을 이용한다. 또한 데이터를 교환하는 경우에는 대부분 압축하거나 컴팩트하게 변환하여 전송하는 방법[7]에 대한 연구가 대부분이다.

본 논문에서는 모바일 기기의 정보 표현 도구로 많이 이용되는 SVG 파일을 압축하거나 컴팩트하게 변환하는 방식과 호환이 가능하면서도 네트워크 대역폭 및 전송 시간을 적게 사용하는 효과적인 전송 방법을 제안한다.

2. 관련연구

모바일 기기가 가지는 여러 가지 제약점으로 인하여 네트워크 사용량을 줄이기 위한 기존의 연구는 대부분 실행 코드를 효과적으로 전송하는 방식에 중점을 두었으며, 그와 연관된 데이터의 경우 대부분 압축 방식을 이용하였다. 특히 모바일 기기에서서의 XML의 한 종류인 SVG 활용에 대한 적용 사례가 많아지면서 제안된 연구가 SVGZ 라는 파일 압축 기법이다[8].

2.1 실행 코드 개선 방식

자바를 이용하여 모바일 기기 기반의 시스템을 구축한 경우 실행 코드 전송으로 발생하는 네트워크 사용량을 줄여주는 방법은 이미 자바의 JVM에 구현되어 있다. 자바로 구현된 바이트 코드는 원격지에서 실행되는 경우, 모든 바이트 코드가 전송이 완료된 후 실행되는 방식이 아닌 클래스 파일별로 요구가 발생하였을 때 전송되어 수행되는 방식을 사용함으로서 네트워크 사용량을 감소시킨다[3]. 자바의 JVM에서 제공하는 방식을 개선한 것이 [4]에서 제안한 방식이다. [4]는 바이트 코드의 실행 지연 시간을 줄이기 위한 방법으로서 사용자가 작성한 클래스 파일들을 분류하여 자주 사용되는 그룹과 덜 사용되는 그룹으로 구분하여, 자주 사용되는 그룹의 클래스 파일을 전송하여 실행시키는 방식이다. 이와 같은 방식을 이용하는 경우 먼저 전송된 클래스 코드만으로도 프로그램의 실행이 가능하기 때문에 실행을 위한 전송되어야 할 바이트 코드의 양이 줄어든다. 전송되어야 할 바이트 코드가 줄어들기 때문에 사용자는 이전보다 프로그램 시작 지연 시간을 단축시킬 수 있는 효과를 가질 수 있다. [5]에서 제안한 방식은 각 클래스 파일을 전역 데이터와 메소드로 구분한 후 클래스 파일의 요청이 발생하게 되면 전역 데이터를 먼저 전송하여 프로그램을 수행시킨다. 프로그램이 수행되는 동안 다른 메소드들은 백그라운드로 전송되는 방식이다. 이 방식을 이용하게 되면 프로그램의 전체 바이트 코드들이 전송되지 않더라도 프로그램을 실행시킬 수 있으므로 프로그램의 시작

본 연구는 중소기업청의 2005년도 지역혁신컨소시엄 사업의 지원에 의해 수행되었음.

지연 시간이 단축되는 장점이 있다. 그러나 변형된 JVM에서만 실행이 가능하기 때문에 다른 시스템과 호환이 불가능하다는 단점이 있다. 이외에도 클래스 파일 분할 방식과 코드 prefetch 기능을 결합한 연구[6] 등이 있으며 바이트 코드들을 압축하여 전송하는 연구도 있다. 바이트 코드들을 압축하는 방식은 기존의 다른 방식과 혼합하여 효과를 배가시킬 수 있는 장점이 있다.

2.2 데이터 전송 방식

모바일 기기에서 실행 코드들이 전송될 때 네트워크 대역폭과 점유 시간을 줄여주는 연구는 많이 진행되어 왔다. 그러나 데이터의 경우 대부분 바이너리 형태인 경우가 많기 때문에 데이터를 압축하는 방식을 많이 이용한다. 또 XML이 데이터 교환용으로 많이 사용되기 시작하면서 XML 데이터를 모바일 기기에서 전송하기 위한 연구로 [7],[8]과 같은 단순 압축이나 구조적인 압축 기법에 대한 연구가 대부분이었다.

3. 모바일 환경에서 효율적인 SVG 전송 방법

본 논문에서는 모바일 기기에서의 효과적인 SVG 전송 방법을 위한 SVG 파일 구조를 다음과 같이 제안한다.

첫째, <g>를 이용하여 그룹화 하여 SVG를 구성한다.
둘째, <g>의 'id' 속성 값에 <g>가 표현하고자 하는 것과 연관된 의미 정보와 버전 정보를 부여한다.
셋째, SVG 파일 내에 <svg> 바로 다음에 각 <g> 들의 'id' 속성 값들을 저장하기 위한 <g>를 선언한다. 메타데이터용 <g>는 정보를 표현하기 위한 그래픽 관련 엘리먼트들은 전혀 포함하지 않으며 그림 1.과 같이 구성한다.

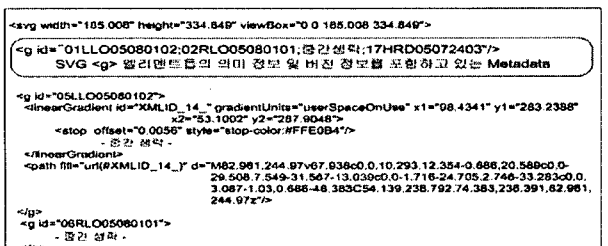

그림 1. SVG 파일 내에 선언한 메타데이터

그림 1.과 같이 맨 처음에 나오는 <g>의 'id' 속성 값에는 <svg> 내의 모든 <g>의 버전 정보를 저장하고 있는 id의 속성 값들을 ';' 구분자에 의해 저장하고 있다. 본 논문에서 제안한 SVG 구조는 기존에 사용되던 XML 파서들을 그대로 사용할 수 있으며, SVG 파일이 표현하고자 하는 정보에도 전혀 영향을 미치지 않기 때문에 기존 시스템에서 그대로 사용이 가능하다는 장점이 있다.

3.1 <g>에 부여되는 'id' 속성 값의 구조

표 1. <g>의 'id' 속성 값

0	1	H	A	R	0	5	0	9	0	4	0	1

01	<svg>내의 순서
HAR	캐릭터를 구성하는 부분을 의미하는 약어
05090401	<g> 버전 번호

<g>의 'id' 속성에 부여되는 값의 구조는 표 1과 같다. 1번째에서 2번째 속성 값은 <g>의 <svg> 내의 순서를 나타낸 것이다. 3번째에서 5번째 속성 값은 <g>가 표현하고자 하는 정보의 의미를 가지고 있는 약어이다. 표 1.의 'HAR'의 경우 "Hair"를 의미한다. 6번째에서 마지막까지의 속성 값은 <g>의 버전 정보를 의미하는 것으로서 앞의 여섯 자리는 최초로 작성된 날짜를 의미하며, 나머지 2자리는 수정된 버전을 의미한다.

3.2 메타 데이타를 이용한 SVG 파일 전송 방법

본 논문에서 제안한 SVG 구조를 이용하여 모바일 기기와 서버 시스템 간의 네트워크 사용량 및 전송 시간을 감소시키기 위하여 제안하는 전송 방식은 다음과 같다.

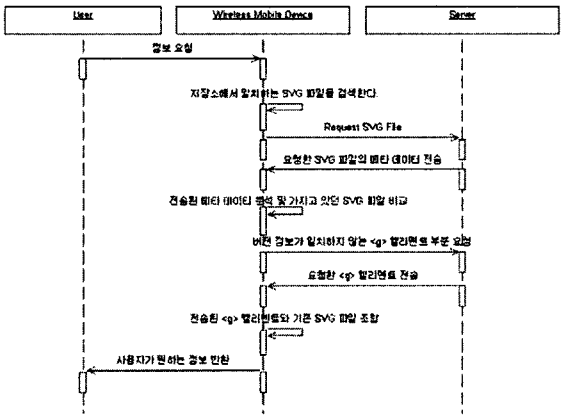

그림 2. 본 논문에서 제안한 전송 방법의 Sequence Diagram

사용자는 LBS, UI, Cartoon 서비스 등과 같은 정보를 모바일 기기에 요청한다. 모바일 기기는 사용자가 요청한 해당 정보를 모바일 기기의 저장 공간에 저장되어 있는가를 확인한다. 사용자가 원하는 정보가 없는 경우 Server에 해당 정보를 표현하기 위한 SVG 파일을 요청한다. Server는 사용자가 원하는 SVG 파일을 구성하는 <g>그룹들의 버전 정보가 포함된 메타 데이터를 모바일 기기에 전송한다. 모바일 기기에서는 전송받은 메타데이터의 정보를 분석한 후, 현재 자신이 가지고 있는 <g>의 버전 정보와 비교한다. 모바일 기기는 일치하지 않은

버전 정보의 <g>를 Server에 요청한다. Server는 모바일 기기가 요청한 <g> 그룹들을 전송한다. 모바일 기기는 전송받은 엘리먼트 그룹들과 기존의 엘리먼트들을 조합한 후 사용자가 원하는 정보에 해당하는 SVG 파일을 생성한 후 사용자에게 전송하는 기능을 수행한다. 그림 2.는 본 논문에서 제안한 전송 방식의 시퀀스 다이어그램이다.

본 논문에서 제안한 전송 방식은 사용자가 원하는 정보를 구성하기 위하여 해당 정보의 전체에 해당하는 SVG 파일을 전송하지 않고, 사용자가 이미 가지고 있는 SVG 파일들의 메타데이터와 서버에서 전송된 메타데이타를 비교한 후 변동되거나 추가되는 부분만을 전송하기 때문에 그에 따른 네트워크 사용량을 감소시킬 수 있으며, 네트워크 사용량이 감소되는 만큼 사용자에게 정보를 표현하기 위한 지연 시간이 단축되는 효과를 가질 수 있다.

4. 성능분석

본 논문에서 제안한 SVG 전송 방법에 대한 성능 분석을 위하여 다음과 같은 환경에서 테스트하였다. 개발 환경은 서버의 경우 J2SDK 1.4.x를 이용하여 구성하였으며, 임베디드 시스템의 경우 J2ME Wireless Toolkit 2.2를 이용하여 구현하였다. J2ME Wireless Toolkit 2.2의 구성은 "Target Platform"은 'JTWI', 'Profiles'는 "MIDP 2.0", 'Configuration'은 "CLDC 1.1"과 같이 하였다. 테스트 환경 중 'Performance'의 경우 "Network throughput emulation"은 '19200bits/sec'으로, 'VM speed Emulation'은 'Disable' 하였다.

테스트를 위하여 구성된 SVG 파일은 아바타를 표현하기 위하여 구성하였다. 대부분의 SVG 파일의 전체 사이즈는 '15KB'에서 '20KB'까지이다. 테스트의 범위는 실제로 전송되는 SVG 파일의 전송 시간을 측정하였다. 메타데이터의 내용과 일치하지 않아 재전송되는 SVG 파일의 사이즈는 100% 재전송, 75% 재전송, 50% 재전송, 25% 재전송으로 구분하였다. 네트워크 전송량의 경우 SVG 파일의 전송 시간과 비례하기 때문에 별도로 테스트하지 않았다.

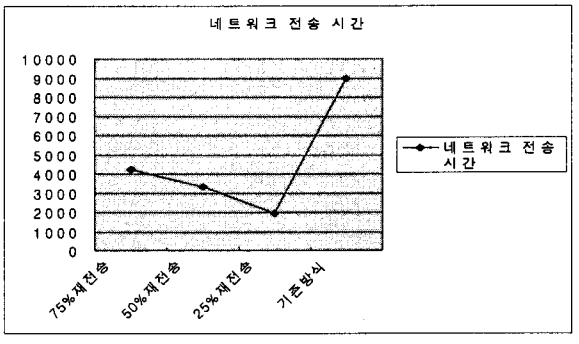

그림 3. 네트워크 전송 시간

4.1 네트워크 전송 시간 분석

그림 3.은 SVG 파일을 모바일 기기에 전송하는 필요한 소요 시간을 나타내는 것으로서 모바일 기기에 전송되는 SVG 파일 사이즈와 비례하여 네트워크 전송 시간이 비례함을 알 수 있다. 본 논문에서 제안한 전송 방법을 사용하는 경우 SVG 파일을 75%를 재전송하더라도 네트워크 전송 시간을 최대 32%까지 감소시킬 수 있다.

5. 결론

본 논문에서 제안한 SVG 파일 구조를 사용한 SVG 파일 전송 방법을 사용하는 경우 사용자의 모바일 기기와 서버 간의 전송 시간을 76%에서 24%로 감소시킬 수가 있었다. 그러나 사용자의 모바일 기기에 연관된 SVG 파일이 전혀 존재하지 않는 경우에는 약 200ms의 서비스 지연 시간이 발생한다. 그러나 대부분 서버에서 모바일 기기로 정보를 전송하는 경우 완전히 새로운 정보가 제공되기 보다는 LBS 기반의 교통 정보, 주식 시세, 날씨 정보 등과 같이 부분적으로 변경되는 경우가 대부분이다. 그러므로 본 논문에서 제안하는 SVG 전송 방식을 적용하면 네트워크 전송 시간 및 전송되는 SVG 파일의 크기을 효과적으로 감소시킬 수 있다.

참고문헌

[1] http://www.macromedia.com/software/flashlite/

[2] http://www.w3c.org/Graphics/SVG

[3] C. Krintz, B. Calder, and U. Holzle, "Reducing Transfer Delay Using Java Class File Spiliting and Prefetching," In Proceedings of the 1999 ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languates, and Applications, pp.276-291, 1999.

[4] E. G. Sirer, A. J. Gregory, and B. N. Bershad, "A Practical Approach for Improving Startup Latency in Java Applications," In Workshop Compiler Support for System Software, pp. 47-55, 1999.

[5] C. Krintz, B. Calder, H.B. Lee, and B. G. Zorn, "Overlapping Execution with Transfer Using Nonstrict Execution fro Mobile Programs," In Proceedings of the 8th International Conference on Architectural Support for Programming Languages and Operating Systems, pp. 159-169, 1998.

[6] D. Lee, J.-L. Baer, B. Bershad, and T. Anderson, "Reducing Startup Latency in Web and Desktop Applications," In Proceedings of the 3rd USENIX Windows NT Symposium, pp. 165-174, 1999.

[7] http://www.ftponline.com/xmlmag/2003_02/magazine/columns/netdomain/default.aspx

[8] http://www.adobe.com/svg/illustrator/compressedsvg.html

[9] http://www.idealliance.org/papers/xml2001/papers/html/05-05-05.html