
개선된 SMETA 시스템과 암호화 적용에 관한 연구

황인문* · 유남현* · 손철수* · 김원중**

A Study on Improved SMETA System and Applying Encryption Function

In-Moon Hwang*, Nam-Hyun Yoo*, Cheol-Su Son*, Won-Jung Kim**

이 논문은 2006년도 순천대학교 공과대학학술재단 연구비에 의하여 연구되었음

요 약

XML이 다양한 분야에서 정보의 표현 및 교환을 위한 표준 포맷으로 사용되면서 SVG가 무선 인터넷 기반의 모바일 폰과 같은 임베디드 시스템의 사용자 인터페이스나 표현 도구로 사용되고 있다. SVG 파일은 실제 사용되는 데이터에 비하여 문서의 구조를 위한 부가적인 정보를 유지하여야 하기 때문에 전송시간이 많이 소요되는 문제점을 가지고 있다. SMETA(Svg transmission Method using Semantic meTAdata) 시스템[9]은 SVG 파일을 의미부여가 가능한 최소단위로 분할한 후, 각각의 분할된 부분에 의미가 부여된 메타데이터를 할당하여 실제 전송되는 SVG 파일의 크기를 줄이기 위한 연구이다. 본 논문에서는 기존의 SMETA 시스템에서 교환되는 파일의 크기를 줄이기 위하여 사용한 메타데이터 교환 기법 대신에 서버시스템에 사용자 별로 전송되어 저장소에 보관된 메타데이터를 분석한 후, 사용자에게 필요한 데이터를 전송하는 개선된 SMETA 시스템에 대하여 연구하였다. 또한 암호화를 적용하더라도 기존 시스템에 비하여 더 나은 성능을 제공할 수 있음을 시뮬레이션을 통하여 입증하였다.

ABSTRACT

As the XML is used as the standard format for information delivery and exchange in various fields, the SVG is frequently used as a user interface or expression tool for the embedded system like an internet based mobile phone. The SVG file must contain additional information specifying the structure of the document and it consumes more transmission time than the actual data sent. The SMETA(Svg transmission Method using Semantic meTAdata) system[9] is a study to reduce the size of the SVG file by partitioning the SVG file to a minimal size and assigning meaningful meta data to each portion of the data. In this paper, instead of the meta data exchange method for reducing the size of the file exchanged in the existing SMETA system, we studied an improved version of SMETA system that analyzes the data for each user in the server system and only transmits the data that a user needs. In addition, through our simulation, we verified that it provides better performance than the existing system even if encryption is used.

키워드

Semantic Metadata, Encryption, SVG, Embedded System, Mobile Device

* 순천대학교

** 교신저자

I. 서론

기존의 모바일 폰과 같은 임베디드 시스템에서는 대부분 사용자 인터페이스를 구현하거나 콘텐츠 서비스를 표현하기 위하여 BMP(BitMaP)와 같은 비트맵 이미지를 이용하였다. 그러나 다양한 화면 사이즈 및 색깔이 지원되는 모바일 폰들이 출시되면서 이미 제작되어 있는 사용자 인터페이스나 콘텐츠 서비스를 화면 사이즈가 다른 모바일 폰에 적용하기 위해서는 다시 제작하여야 하는 문제가 존재한다. 이와 같은 문제를 해결하기 위하여 매크로미디어사의 Flash Lite[1]와 W3C의 SVG(Scalable Vector Graphic)[2] 같은 벡터 기반의 이미지 파일 포맷 등이 모바일 폰 등과 같은 임베디드 시스템에 사용되기 시작하였으며, 3GPP(3rd Generation Partnership Project)의 MMS (Multimedia Message Service)를 위한 이미지 파일로 SVG가 이용되었다.

SVG는 XML[3]의 한 종류로서 XML이 가지는 장점뿐만 아니라 단점도 함께 가지고 있다. 특히 XML의 경우 파일 전체의 용량에 비하여 실제 데이터가 차지하는 비율이 상대적으로 적기 때문에 모바일 기기와 같은 임베디드 시스템에서 데이터 교환용으로 사용하기 위해서는 XML 파일의 사이즈를 효과적으로 줄이는 연구가 필요하다. XML 사이즈를 줄이기 위한 기존 연구로는 압축 방법을 이용하는 XMill[4], XGrind[5] 등이 있다. 그러나 SMETA(Svg transmission MEthod using Semantic meTadata)[9] 시스템은 기존의 연구와 다른 형태로 제안된 방식이다. 기존 연구의 경우 XML 파일 자체 사이즈를 줄이는데 중점을 두었지만, SMETA는 XML 파일은 그대로 유지한 상태에서 메타데이터를 이용하여 전송되는 XML 파일의 용량을 줄이는 방식이다. 또한, SMETA 시스템의 장점은 기존의 연구결과들을 그대로 적용시킬 수 있다는 것이다.

본 논문에서는 효율적인 SVG 파일 전송을 위한 개선된 SMETA 시스템을 제안하고, 시뮬레이션을 통하여 개선된 SMETA 시스템의 전송 성능이 향상되었음을 확인할 수 있었다. 또한 암호화를 적용하더라도 SMETA를 이용하지 않는 기존의 시스템보다 훨씬 뛰어난 결과를 얻을 수 있었다. 본 논문의 구성은 다음과 같다. 2장에서 SMETA에 대해서 설명하고, 3장에서는 개선된 SMETA 시스템과 암호화를 적용한 SMETA 시스템을 설계하였다. 4장에서는 프로토 타입을 구현하고, 구현한 후 실제

데이터를 이용하여 전송 성능 및 암호화 적용을 테스트하여, 개선된 SMETA 시스템이 기존의 시스템에 비하여 효과적인 성능 향상을 이루어졌음을 보였다.

II. 관련 연구

2.1 SMETA

XML 문서는 문서 교환을 위한 산업표준으로서, 대부분의 어플리케이션 프로그램에서 문서의 기본 포맷으로 사용하고 있다. XML 문서는 교환하기 쉽고, 사람이 읽을 수 있는 정보이지만, 실제 정보 이외에도 문서를 구성하는데 필요한 구조적인 정보를 보관하고 있기 때문에 파일 크기가 커지는 단점이 있다. 특히 모바일 기기와 같은 임베디드 시스템에서는 제한된 네트워크 환경과 시스템 성능 때문에, 교환되는 파일 크기가 큰 경우 파일 전송에 너무 오랜 시간이 걸리고, 시스템에 많은 부하가 된다. 이와 같은 문제를 해결하기 위하여 어플리케이션 사이에 교환되는 XML 문서의 크기를 줄이기 위하여 XMill, XGRIND, 압축 방법 등이 사용된다. XMill과 XGRIND는 XML 파일의 구조를 변경하는 방식이며, 압축 방법은 XML 자체를 압축하는 것이다. 특히 XML 문서 중에서 그래픽 파일을 위한 SVG의 svgz[6]와 같이 별도의 기술이 적용된 경우도 있다.

SMETA는 기존의 방법과는 다르게 문서의 구조 정보를 이용하는 방식이다. SMETA에서는 교환되는 XML (or SVG) 파일의 크기를 줄이기 위하여, 문서의 구조 정보를 표현하는데 사용되는 <g> 엘리먼트의 속성에 의미가 할당된 메타데이터를 부여한다. 부여된 메타데이터들은 실제 데이터를 전송하기 전에 비교되어, 필요한 데이터 부분만을 전송하는 방식이다. SMETA는 네트워크나 커널 단계에서 교환되는 SVG 파일의 크기를 줄이지 않고, 응용 프로그램의 캐시 메모리 단계에서 작업이 수행된다.

(1) SMETA에서 사용하는 메타데이터 구조

그림 1.은 SMETA에서 제안한 SVG 파일의 구조이다. SVG 파일은 각각의 그룹화 된 구성 요소로 이루어졌으며, 그룹화 된 구성 요소들은 <g> 엘리먼트의 'id' 속성에 부여된 의미가 할당된 메타데이터에 의하여 구별된다.

SMETA에서 <g> 엘리먼트의 'id' 속성에 부여하는 메타 정보의 구성은 다음과 같다. 첫 번째에서 세 번째까지

필드에 포함되는 정보는 현재 이미지를 구성하는 요소의 타입을 나타내는 약어이며, 네 번째에서 마지막까지 필드는 <g> 엘리먼트의 버전 정보를 의미한다. SVG 파일의 <g> 엘리먼트에 부여되는 메타 데이터의 크기는 약 22바이트이며, 메타 데이터로만 구성되는 SVG 파일은 포함되어 있는 <g> 엘리먼트의 수에 비례하여 파일 사이즈가 커진다.

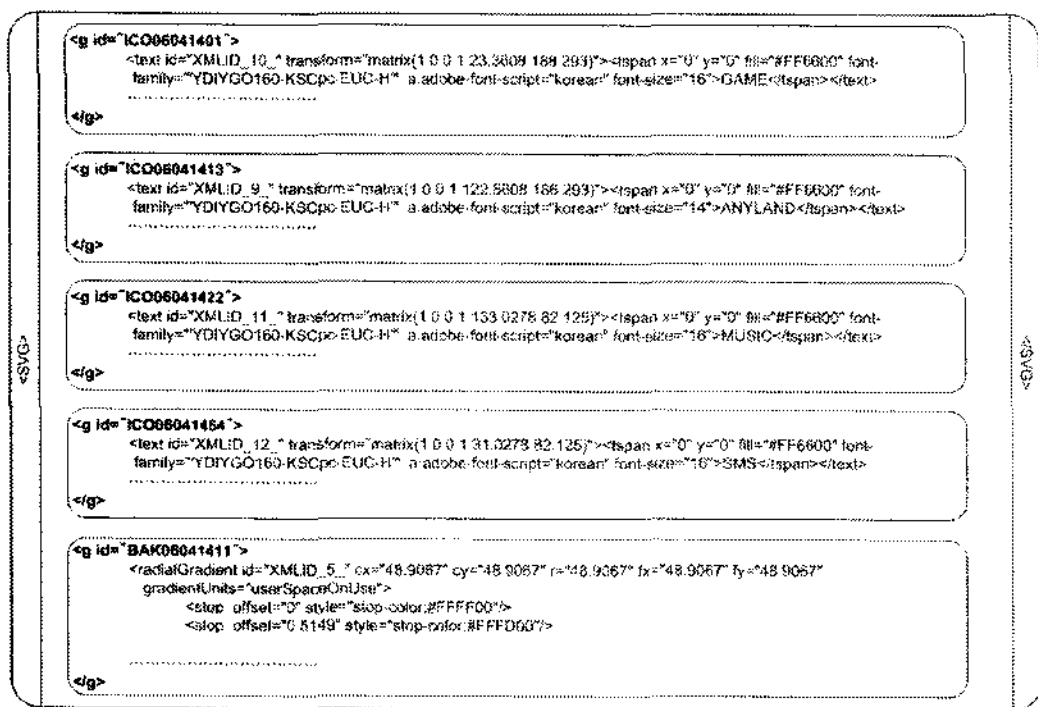


그림 1. 의미있는 속성이 부여된 <g> 엘리먼트를 이용한 SVG 파일 구조
 Fig. 1 The Structure of SVG using <g> Element assigned Semantic Attribute

(2) SMETA를 이용한 SVG 파일 전송 방법

SMETA를 이용한 SVG 파일의 전송 방법은 [그림 2]와 같이 시퀀스 다이어그램으로 나타낼 수 있다.

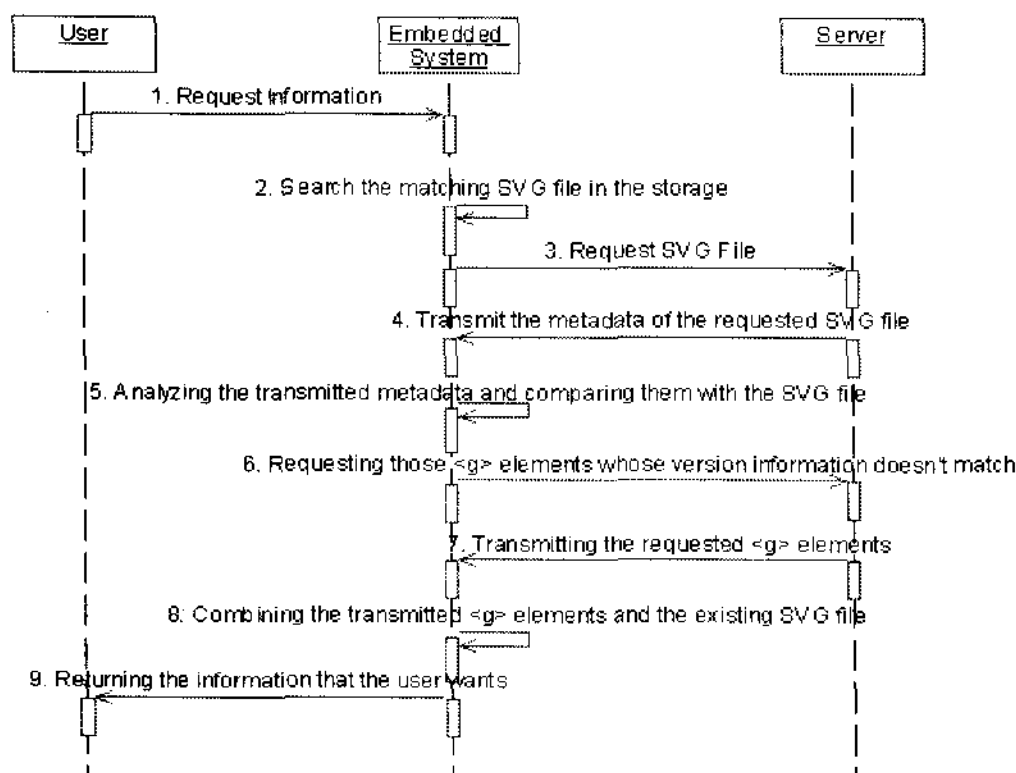


그림 2. SMETA의 메타데이터를 이용한 전송 방법의 시퀀스 다이어그램
 Fig. 2 Sequence Diagram of transmission method using metadata of SMETA

사용자는 UI, MMS, GIS 등과 콘텐츠 정보를 모바일 기기와 같은 임베디드 시스템에 요청한다. 임베디드 시스템은 사용자가 요청한 정보가 무선 모바일 단말기의 저장 공간에 저장되어 있는지를 확인한다. 사용자가 원하는 정보가 없는 경우 서버에 해당 정보를 표현하기 위한 SVG 파일을 요청한다. 서버는 사용자가 원하는 SVG 파일을 구성하는 <g> 엘리먼트 그룹들의 메타데이터로만 구성된 메타데이터용 SVG 파일을 모바일 기기와 같은 임베디드 시스템에 전송한다. 임베디드 시스템에서는 전송받은 메타 데이터를 분석한 후, 현재 자신이 가지고 있는 메타 데이터와 비교한다. 임베디드 시스템은 가지고 있지 않거나 버전 정보가 일치하지 않을 경우에는 <g> 엘리먼트를 서버에 요청한다. 서버는 임베디드 시스템에서 요청한 <g> 엘리먼트 그룹들만으로 구성된 부분 SVG 파일을 전송한다. 임베디드 시스템은 전송받은 부분 SVG 파일을 이용하여 기존의 <g> 엘리먼트들을 조합한 후 사용자가 원하는 정보에 해당하는 SVG 파일을 생성한 후 그 결과를 사용자에게 보여주는 기능을 수행한다.

이와 같은 과정 중에서 사용자가 원하는 정보를 구성하기 위하여 해당 정보의 전체에 해당하는 SVG 파일을 전송하지 않고, 사용자의 임베디드 시스템에 이미 가지고 있는 SVG 파일들과 비교한 후 변동되거나 추가되는 부분만을 전송하기 때문에 네트워크 사용량을 감소시킬 수 있으며, 네트워크 사용량이 감소되는 만큼 사용자의 정보를 구성하는 시간이 단축되는 효과를 가진다.

III. 암호화가 적용된 개선된 SMETA의 설계

3.1 개선된 SMETA의 기본 개념

SMETA 시스템은 SVG를 사용하는 모바일 기기에 불필요한 데이터 교환을 하지 않게 함으로써 시스템의 전송 성능, 정보의 생성 및 응답 성능을 향상시켰다.

일반적으로 모바일 기기에 사용되는 CPU와 메모리는 개인용 컴퓨터에서 사용되는 것보다 성능이 떨어진다. 또한, 네트워크 속도도 일반 유선환경에 비하여 매우 느리다. 이와 같은 제약성으로 인하여, 사용자의 모바일 환경에서 데이터를 처리하는 것보다 정보를 제공하는 서버 쪽에서 모든 데이터 처리를 수행한 후, 사용자의 모바일 기기에 사용자가 필요로 하는 정보만을 전송한다

면, 네트워크 전송 속도, 정보의 생성 및 응답 성능을 크게 향상시킬 수 있다. 본 논문에서는 기존의 SMETA 시스템의 성능을 향상시키기 위하여 사용자의 모바일 기기와 서버 사이에 메타데이터를 전송하지 않고, 서버 측에서 각각의 클라이언트에 대한 저장소를 별도로 제공하는 새로운 방식을 제안하였다. SMETA 시스템은 사용자의 모바일 기기에 전송된 모든 메타데이터 정보를 저장하고 있으며, 사용자가 요청하는 정보를 생성한 후, 사용자에게 정보를 전송하기 이전에 저장소에 저장되어 있는 메타데이터와 비교하여 사용자에게 필요한 정보만을 추출하여 전송한다.

3.2 암호화가 적용된 개선된 SMETA의 설계

그림 3.은 암호화 기능이 적용된 개선된 SMETA 시스템의 구조로서 기존의 SMETA 시스템과의 가장 큰 차이는 Metadata Repository for Mobile Device 이다. 이 저장소는 사용자의 모바일 기기에 전송되는 각 정보들의 메타데이터 값들을 별도로 보관하고 있는 저장소로서, 사용자에게 정보를 전송할 때마다 그 정보를 구성하는 메타데이터만을 별도로 구성하여 저장하고 있다. License Server는 사용자가 원하는 형태의 다양한 암호화 방법을 지원할 수 있는 표준 서버이다. 공개키 및 대칭키 암호화 알고리즘을 기반으로 하여 정보를 암호화하고, 복호화할 때 필요한 정보를 제공한다. 또한 암호화 작업을 수행하는 경우에는 빠른 실행 속도를 구현하기 위하여 XML-Based canonization을 사용하지 않고 텍스트 기반의 암호화 방식을 사용한다[7].

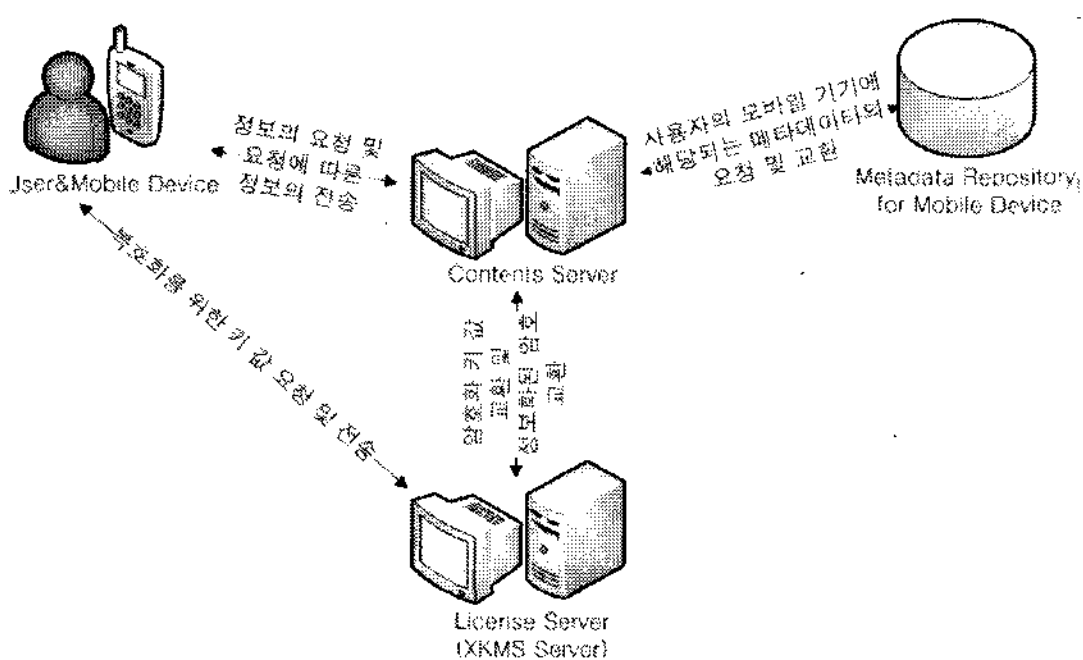


그림 3. 암호화 기능이 적용된 개선된 SMETA 시스템의 구조

Fig. 3 The Structure of Improved SMETA applying Encryption Function

그림 4.는 개선된 SMETA 시스템의 시퀀스 다이어그램이다. 기존 SMETA 시스템의 시퀀스 다이어그램과 다른 점은 Contents Server와 사용자의 Mobile Device 사이에서 정보 전송을 위한 메타데이터의 교환을 수행하지 않고, Contents Server에서 사용자의 Mobile Device와 관련된 메타 데이터를 비교한 후에, 필요한 정보만을 파일로 생성하여 사용자의 Mobile Device로 전송한다. 또한 암호화에 따른 License Server 시스템과의 암호화 작업에 따른 시퀀스들이 추가되었다.

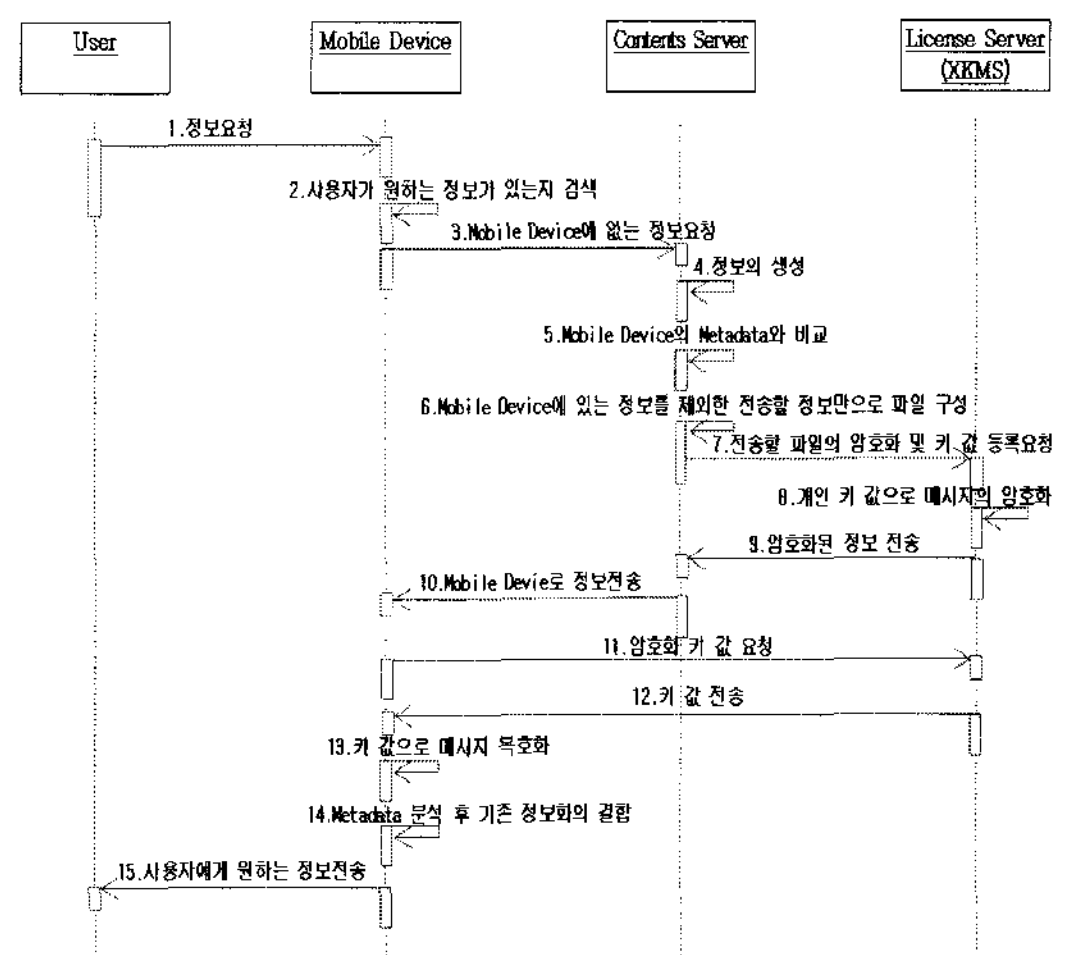


그림 4. 암호화 기능이 적용된 개선된 SMETA 시스템의 시퀀스 다이어그램

Fig. 4 The Sequence Diagram of Improved SMETA applying Encryption Function

IV. Performance Analysis

4.1 암호화가 적용된 개선된 SMETA의 구현

개선된 SMETA 시스템은 개발 환경은 JDK(Java Development Kit) 1.5.x, Sun Java(TM) Wireless Toolkit 2.5.1 for CLDC(Connected Limited Device Configuration)를 이용하였다. Target Platform은 MSA(Mobile Service Architecture)이며, Profiles는 MIDP(Mobile Information Device Profile) 2.1을 Configuration은 CLDC 1.1을 사용하였다. 사용된 확장된 함수로는 SASTA-CRYPTO(JSR 177)과 Web Service API(JSR 171), JAXP(Java API for

XML Processing) XML Parser(JSR 172), Scalable 2D Vector Graphics API(JSR 226)을 사용하였다. 암호화 작업은 JCA(Java Cryptography Architecture) API를 사용하였으며, 암호화 작업에 따른 성능 분석을 위하여 사용된 암호화 방식은 대칭키 암호화 방식의 DESede 방식을 사용하였다. 시스템의 성능 테스트에 사용한 중앙처리장치(CPU)는 Intel Mobile Core 2 Duo T7200 @2.00Ghz에 메모리는 2.5GB를 사용하였다.

4.2 시스템 성능 분석

개선된 SMETA 시스템이 기존의 시스템보다 우수한 성능을 제공함을 보이기 위하여 기존의 SMETA 시스템에서 서버 시스템과 사용자의 Mobile Device 사이에 메타데이터를 상호 교환하여 처리한 후 정보를 구성하는 것과 서버 시스템에서 모든 메타데이터와 관련된 처리를 한 후 정보를 구성하는 것을 비교하였다. 시스템의 성능을 측정하기 위하여 WTK 2.5.1에서 제공하는 Default Emulator의 Network throughput emulation을 CDMA 2000 1x의 최대 대역폭인 14,400bit/sec으로 하였으며, MTU (Maximum Transmission Unit), 스토리지 공간은 1,024KB로 하였다. 또한 암호화 시스템을 적용하더라도 SMETA 시스템의 성능이 우수하다는 것을 증명하기 위해 사용한 보안 정책은 WTK 2.5.1의 MSA를 사용하였다. 암호화 작업에 따른 속도 저하를 고려하여 VM speed emulation은 적용하지 않았다.

(1) SMETA 시스템과 개선된 SMETA의 성능 비교

SMETA 시스템과 개선된 SMETA의 차이는 정보 구성에 필요한 메타데이터의 교환 여부이다. 기존 시스템의 시퀀스 다이어그램인 그림 2.의 4,5,6 번의 처리 과정과 개선된 시스템의 시퀀스 다이어그램인 그림 4.의 5,6 번의 처리 과정의 처리 속도를 비교하였다.

시스템의 성능 테스트를 위하여 25개의 그룹 엘리먼트로 구성된 102,598바이트 크기의 SVG 파일을 이용하여 테스트 하였으며, 선택되는 그룹 엘리먼트는 랜덤 함수를 이용하여 생성하였다. 로컬 시스템에서 서버 시스템과 클라이언트 시스템을 설치 한 후 1,000회의 전송 성능 테스트를 시험하였으며, 전송 테스트에 사용된 메타데이터의 크기는 표 1.과 같다.

표 1. 재전송시 사용되는 메타데이터의 파일 사이즈
Table. 1 File size of Metadata when re-sending

Ratio	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
Size	42 ~ 53	75	97 ~ 108	130	152 ~ 163	185	207 ~ 218	240	262 ~ 273	295

(단위 byte)

그림 5.의 결과 값을 분석하게 되면 개선된 SMETA 시스템에서는 실제 전송해야 할 SVG 파일을 추출하기 위하여 필요한 전송해야 될 메타데이터를 클라이언트에 전송하지 않고 서버 시스템의 Metadata Repository for Mobile Device에 저장되어 있는 메타데이터와 비교하기 때문에 자체 처리 시간의 경우 거의 소요되지 않는다.

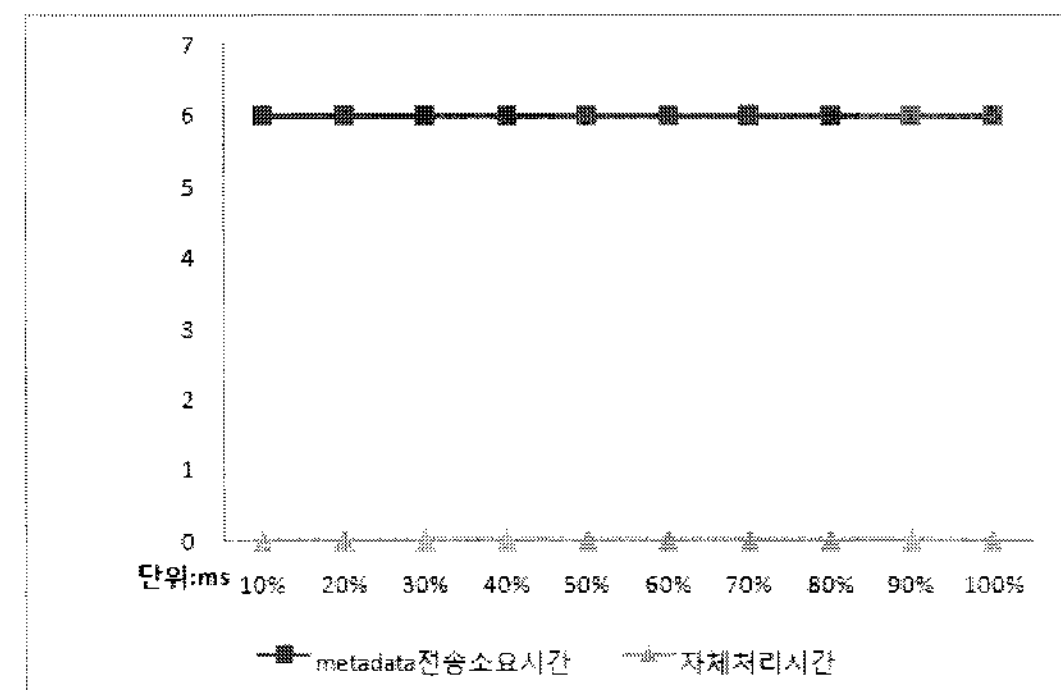


그림 5. 기존 SMETA 시스템과 개선된 SMETA 시스템의 처리 성능 비교

Fig. 5 The Comparison of Performance Capacity between Existing SMETA system and Improved SMETA

기존 시스템에서는 메타데이터의 교환으로 처리되는 시간의 경우, 메타데이터의 비교에 소요되는 시간보다 메타데이터의 교환에 따른 네트워크 접속에 따른 처리시간 및 메타데이터 전송 소요 시간이 대부분을 차지한다. 다시 말하면 기존 시스템과 개선된 SMETA 시스템에서 가장 큰 관건은 메타데이터를 처리하는 시간보다 메타데이터를 교환하는 시간임을 의미 한다. 특히 테스트 환경의 경우 로컬 컴퓨터에 서버와 클라이언트가 함께 존재하기 때문에 전송 시간이 비교적 적게 소요되지만, 실제 운용 환경에 적용하는 경우 해당 메타데이터의 교환에 따른 소요 시간은 증가할 수밖에 없다. 이와

같이 개선된 SMETA의 시스템의 성능은 기존의 SMETA 시스템보다 나은 성능을 제공함을 알 수 있다.

(2) 암호화 적용에 따른 성능 개선 효과

암호화 시스템을 적용하는 경우에는 그림 4.의 7,8,11,12,13번의 과정이 요구된다. 모바일 기기에서 암호화와 관련되어서는 [10],[11],[12],[13]와 같은 다양한 연구 방법들이 제시되었다. 본 논문에서는 암호화 기법을 적용하더라도 SMETA 방식이 기존의 시스템보다 나은 성능을 제공한다는 것을 증명하기 위해서, 가장 일반적인 대칭키 암호화 알고리즘인 DESede를 이용하였다. 암호화 할 때의 키 값은 라이센스 서버에 사용자 아이디와 함께 저장하고, 복호화시 인증된 사용자의 요구에 의한 경우 키 값을 모바일 기기에 전송한다. 실제 모바일 환경에서 사용되는 공개키 기반의 방식을 사용하는 경우 본 논문에서 소요된 시간보다 더 오랜 시간이 소요될 것이다[8]. 본 논문에서는 DESede를 이용하여 테스트용 SVG 파일을 암호화 하는 경우, 테스트용 SVG 파일의 크기와 관계없이 55바이트에서 56바이트의 암호화된 파일이 생성된다. 이 파일을 클라이언트에 전송하는 경우, 암호화된 파일의 사이즈가 MTU보다 작기 때문에 테스트 환경에서는 전송 시간이 거의 소요되지 않는다. 그러므로 실제 파일을 전송하는 경우보다 암호화해서 전송하는 경우 전송되는 파일 사이즈가 줄어들기 때문에, 키 값을 요구하고 전송 받는 과정을 추가하더라도 그림 6.과 같이 더 나은 성능을 제공할 수 있다. 그림 6.의 결과 값에서 볼 수 있듯이 모바일 기기와 서버 시스템 사이의 성능에서 가장 중요한 것은 전송되는 파일 사이즈임을 알 수 있다.

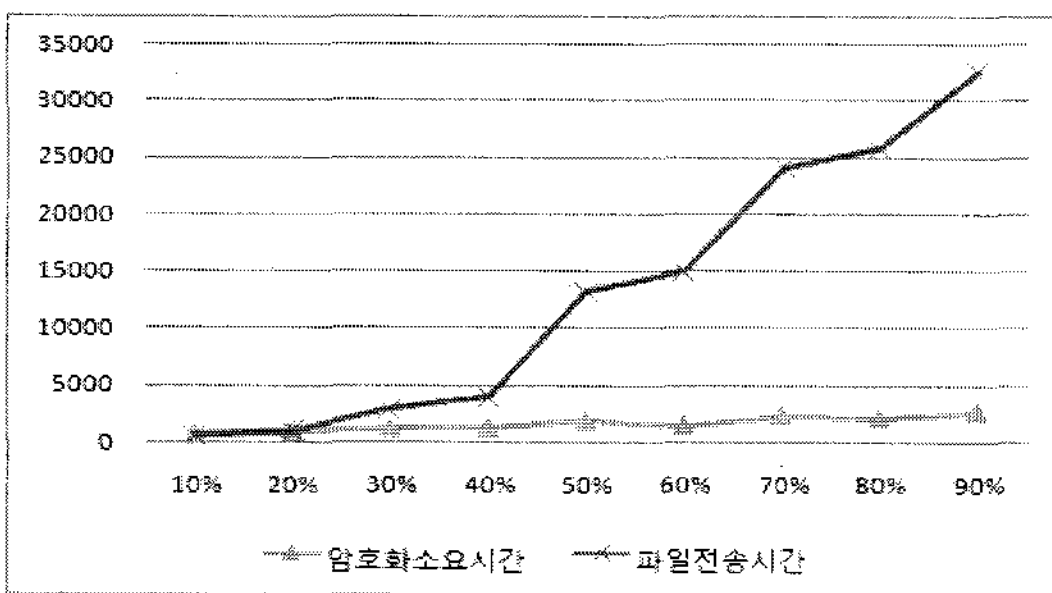


그림 6. 암호화 소요 시간과 암호화 되지 않는 실제 파일 전송시간의 비교
Fig. 6 Average

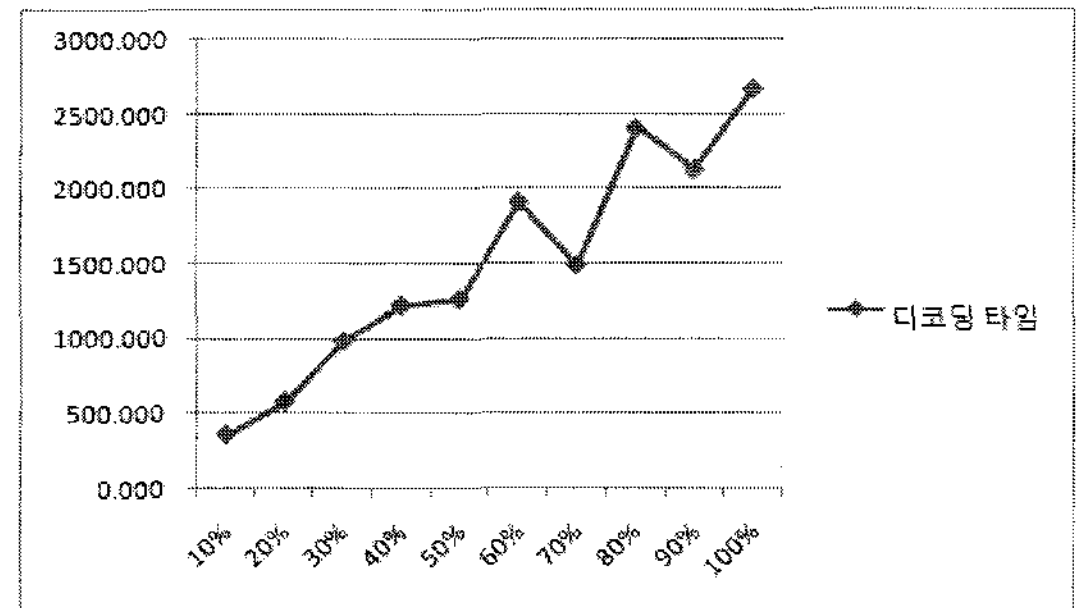


그림 7. SVG 파일 크기에 따른 복호화 소요 시간
Fig. 7 Average

또한, 사용자가 요청하여 전송되어지는 테스트용 SVG 파일의 크기는 102,598바이트의 크기를 가진다. 이 파일을 암호화 하여 전송한 경우, 모바일 기기에서 복호화 할 때 소요되는 시간은 약 2,700ms 정도 소요된다. 그림 7.의 결과 값을 참조하더라도 SMETA를 이용하여 전송되는 파일 크기를 줄이는 것이 전체적인 시스템의 성능 향상을 가져올 수 있음을 알 수 있다.

V. 결론

XML이 정보를 표현하거나 교환하기 위한 표준 포맷으로 자리를 잡아가면서, XML과 관련된 다양한 연구들이 진행되고 있다. 모바일 환경에서는 3GPP(3rd Generation Partnership Project)에서는 MMS(Multi media Messaging Service)의 2D 그래픽 이미지 포맷으로서 SVG를 표준으로 채택하였다. 또한 2007년 11월 6일에는 OHA(Open Handset Alliance)에서는 개방형 모바일 플랫폼인 안드로이드를 개발한다고 발표했다. OHA에는 Motorola, HTC, 삼성전자, LG 전자등과 같은 휴대폰 제조사와, KDDI, Sprint, T-Mobile, NTT Docomo, 텔레포니카 등과 같은 이동통신업체, 브로드컴, 인텔, TI, 퀄컴 등과 모바일 칩셋 제조사들이 다양하게 포함되어 있다. OHA에서 배포한 안드로이드 SDK(Software Development Kit)에는 기본적으로 XML 문서를 처리할 수 있는 라이브러리가 포함되어 있으며, SVG 문서를 처리하고 표현할 수 있는 라이브러리 및 Viewer들을 내장하고 있다.

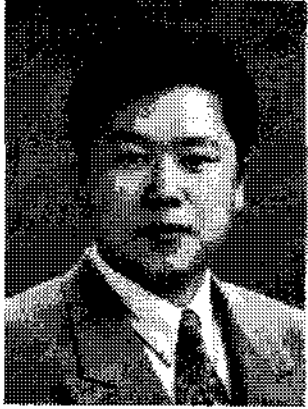
XML 문서는 구조적인 형태를 가지고 있는 문서이므로 이진 파일에 비하여 구조를 표현하기 위한 다양한 정

보들을 표현하고 있다. 또한 이런 구조를 표현하기 위한 다양한 정보들은 사람이 인식할 수 있도록 의미있는 단어들로 구성되어 있는 경우가 대부분이다. 이와 같은 XML의 특징으로 인하여 모바일 환경이나 임베디드 시스템에서는 교환되는 XML 파일을 줄이기 위한 다양한 연구[4,5,6]들이 진행되어 왔다. [9]에서는 기존의 연구와 같이 병행해서 사용이 가능한 SMETA 시스템을 제안하였다. 이 시스템은 SVG 파일의 구조 정보를 이용하여 교환되는 SVG 파일의 크기를 줄이고, 파일 크기가 줄어들므로써 전송 속도를 향상시킨 방법이다. 본 논문에서는 SVG 파일 전송에 효과적인 기존의 SMETA 시스템을 개선하여, 전송되는 SVG 파일의 크기를 줄이기 위한 메타데이터의 교환을 수행하지 않고 서버 시스템에 별도로 구성되어 있는 저장소를 이용하여 메타데이터를 비교하고, 그 결과에 따라 필요한 SVG 파일을 전송하게 함으로써 전송 속도 성능의 향상을 가져왔으며, 전체적으로 시스템의 수행 속도를 향상시켰다. 또한 암호화가 적용되더라도 SMETA 시스템을 사용하는 것이 훨씬 더 나은 성능을 제공함을 알 수 있다.

References

- [1] Flash Lite, <http://www.macromedia.com/soft-ware/flashlite>
- [2] Jon Ferraiolo, Fuisawa Jun, Dean Jackson, "Sc alable Vector Graphics(SVG) 1.1 Specification, "W3C Recommendation, World Wide Web Consortium, January 2003.
- [3] Tim Bray, Jean Paoli, C. M. Sperberg-Mc Que en, and Eve Maler, "Extensible Markup Langauge(XML) 1.0(Second Edition)," W3C Recommendation, World Wide Web Consortium, October 2000.
- [4] H. Liefke and D. Sucie, "XMill: An Efficient Compressor for XML Data," Proc. of the 2000 ACM SIGMOD International Conference on Management of Data, pp. 153-164, 2002.
- [5] P. M. Tolani and J. R. Haritsa, "XGRIND: A Query-friendly XML Compressor," Proc. of 18th International Conference on Database Engineering, pp. 225-234, 2002.
- [6] SVGZ, <http://www.adobe.com/svg/illustrator/compresse-dsvg.html>
- [7] Mark Bartel, John Boyer, Barb Fox, Brian LaMacchia, Ed Simon, "XML-Signature Syntax and Processing," W3C Recommendation, 2002. 02.
- [8] Manish Verma, "XML Security: Implement security layers," IBM DeveloperWorks, 2003.10.
- [9] 유남현, 손철수, 김원중, "SMETA를 이용한 효과적인 SVG 파일 전송에 관한 연구", 한국해양정보통신학회논문지 제 11 권 제 1 호 pp14-19, 2007.01.
- [10] 김남윤, 황기태, "모바일 환경에서 XML 기반 전자서명 시스템의 성능 평가", 한국통신학회논문지 '04-4 Vol.29 No.4C pp570-580.
- [11] 최병선, 김상국, 채철주, 이재광, "모바일 단말기 상에서 안전한 인증을 위한 자바 기반의 PKI 시스템 연구", 정보처리학회논문지C 제14-C권 제 4 호 pp331-340, 2007.08.
- [12] 김후종, 나승원, "무선 인터넷 환경에서 디지털 콘텐츠 저작권 보호를 위한 모바일 보안 시스템의 설계 및 구현", 정보처리학회논문지 C 제10-C권 제 6 호 pp695-704, 2003.10.
- [13] 황선태, "디지털 콘텐츠의 부분 암호화 기법", 정보과학회논문지 : 컴퓨팅의 실제 제 8 권 제 1 호 pp82-87, 2002.02.

저자소개



황인문(In-moon, Hwang)

1996년 한국방송통신대학교
컴퓨터과학과 졸업(이학사)
2001년 순천대학교정보과학대학원
컴퓨터과학과 졸업(이학석사)

1991년~현재 순천제일대학 근무

※관심분야: 소프트웨어공학, WSN, 유비쿼터스,
무선계측제어

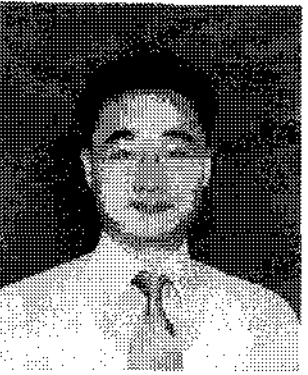


유남현(Nam-hyun, Yoo)

1999년 순천대학교 컴퓨터과학과
졸업(이학사)
2001년 순천대학교 대학원 컴퓨터과
학과 졸업(이학석사)

2007년 순천대학교 컴퓨터과학과 이학박사

※관심분야: Semantic Web, SVG



손철수(Cheol-su Son)

1994년 순천대학교 전자계산학과
졸업(이학사)
2007년 순천대학교 컴퓨터과학과
이학박사

1994~2002년 (주)포스데이타 근무

2008년 한국공학기술연구원 원장

※관심분야: 위치측정, LBS, RFID, WSN



김원중(Won-Jung Kim)

1987년 전남대학교 계산통계학과
(이학사)
1989년 전남대학교 대학원 전산 통계
학과(이학석사)

1991년 전남대학교 전산통계학과 이학박사

1992년~현재 순천대학교 정보통신공학부 교수

※관심분야: 소프트웨어공학, 시스템 모델링, 객체지
향 시스템, 인터넷 서비스, RFID 응용