
임베디드 자료동기화 게이트웨이 구축

박주건* · 박기현* · 우종정**

Construction of Embedded Data Synchronization Gateway

Ju-Geon Pak* · Kee-Hyun Park* · Jongjung Woo**

이 논문은 2009년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된
기초연구사업임(No. 2009-0074021)

요 약

최근 모바일 단말기 제조사들이 자료동기화를 위한 솔루션을 제공하고 있지만 각 업체들은 특화된 자료동기화 방법을 사용하기 때문에 이기종간 데이터의 호환성을 보장할 수 없다. 이를 위하여 상호 다른 자료동기화 데이터 형식을 통일된 데이터 형식으로 변환해주는 게이트웨이가 필요하다. 기존의 고정식 자료동기화 게이트웨이는 다수의 단말기들이 동시에 자료동기화를 요청할 경우 병목현상과 수신한 메시지를 변환하는 과정에서 보안상 취약점이라는 문제점이 존재한다. 이를 해결하기 위해 본 연구에서는 모바일 단말기에 탑재되는 임베디드 방식의 자료동기화 게이트웨이를 제안한다. 임베디드 자료동기화 게이트웨이는 각 단말기에서 데이터를 변환하고 동기화 요청을 처리하므로 게이트웨이의 병목 현상 및 보안 문제를 해결할 수 있다. 개발한 시스템에 의하면 게이트웨이가 정상적으로 동작하며 동기화에 따른 병목현상이 심하지 않았다.

ABSTRACT

Nowadays, the major manufacturer of the mobile devices has offered its own solution for the data synchronization. But each manufacturer uses its proprietary data synchronization method so that the compatibility of data can not be guaranteed. To solve such problems, a gateway is needed to convert different forms of data into a unified form of data and vice versa. Some problems have been found in the experiments with the previously implemented gateway. First, a bottleneck state might arise if a number of mobile devices request data synchronization simultaneously. Second, the gateway itself would become a serious vulnerable point to an attack during the process of converting received data. The proposed gateway can solve this bottleneck state and security issue since it converts data and processes synchronization request in a mobile device. In this paper, the proposed gateway is constructed and the results of some experiments with the gateway are presented. According to the results, the proposed gateway operates properly and it is relatively insensitive to the bottleneck state.

키워드

자료동기화, 임베디드, 게이트웨이, 윈도우모바일, OMA

Key word

Data synchronization, Embedded, Gateway, Window Mobile, OMA

* 계명대학교 컴퓨터공학과
** 성신여자대학교 IT학부 (교신저자)

접수일자 : 2009. 12. 03
심사완료일자 : 2009. 12. 31

I. 서 론

최근 모바일 단말기는 기본적인 음성통화 기능뿐만 아니라, 연락처, 일정 관리 등 다양한 용도를 위한 응용 프로그램과 데이터를 활용하고 있다. 특히, 모바일 단말기는 물리적으로 휴대하기 간편한 크기 및 무게뿐만 아니라, 데스크톱 수준의 사용자 조작 화면을 지원함으로써, 개인적인 용도를 비롯하여 비즈니스 영역까지 그 활용 분야를 확장하고 있다. 이로 인해 데이터의 최신성을 보장하기 위한 자료동기화 기술이 모바일 영역에서 필수적인 기술이 되었고, 모바일 단말기가 더욱 다양한 기능과 데이터를 제공함에 따라 자료동기화 기술의 중요성이 커져가고 있다. 이에, 각 단말기 제조회사들은 자사 단말기 데이터의 최신성을 유지하기 위해 독자적인 자료동기화 프로토콜을 개발하여 사용하고 있다[1-3]. 하지만, 주요 단말기 제조 회사에서 제공하는 독자적인 자료 동기화 방식은 같은 방식의 프로토콜을 사용하는 단말기 간의 상호 호환성을 보장하지만 기기종간의 상호 호환성은 보장하지 않는다. 이러한 문제를 해결하기 위한 방안은 다음과 같다.

- 자료동기화에 관한 표준 프로토콜을 개발: 모든 단말기 제조 회사에게 표준 프로토콜의 사용을 강제함으로써 상호 호환성 문제를 해결할 수 있다. 하지만 단말기 제조 회사들은 독자적인 자료동기화 프로토콜을 사용하여 시장 점유율을 높이려고 하기 때문에 어렵다.
- 모든 자료동기화 프로토콜을 수용할 수 있는 자료동기화 서버 개발: 서로 다른 형식의 동기화 요청이 오더라도 해석하고 처리할 수 있다면, 상호호환성 문제를 해결할 수 있다. 그러나 모든 자료동기화 프로토콜의 구조와 형식을 알고 있어야 한다. 하지만 각 단말기 제조 회사들은 자사의 자료동기화 프로토콜의 내부 구조를 공개하지 않고 있으므로 어렵다.
- 단말기 내에서 사용하는 데이터를 직접 추출하여 통일된 형식으로 변환하는 방식: 이를 위해서는 별도의 응용프로그램 또는 게이트웨이가 필요하며, 데이터를 추출하고 변환하는 과정에서 동기화 시간이 증가하고 병목현상이 발생할 수 있다[4].

제조회사에 의한 문제점으로 세 번째 방식인 자료 동기화 게이트웨이 방식에 대한 연구가 최근 활발하게 진행되고 있으며 [4, 5], 대부분 데스크톱을 이용한 고정식 자료동기화 게이트웨이 방식이다. 이는 하나의 게이트웨이 시스템이 각 단말기의 동기화 대상 데이터를 통일된 방식으로 변환한 후 관리 서버에 송신하고, 관리 서버로부터 수신한 데이터를 각 단말기가 사용하는 형식의 데이터로 변환하여 전달하는 방식이다. 그러나 이 방식은 몇 가지 문제점을 가지고 있다. 첫째, 여러 대의 단말기들로부터 수신한 동기화 요청에 의해 동시에 처리할 데이터의 양이 증가함으로써, 게이트웨이에 병목 현상이 발생할 수 있다. 둘째, 게이트웨이 내부에서 각 단말기 또는 동기화 서버로부터 수신한 데이터를 변환하기 위해서는 암호화된 데이터를 복호화하는 과정이 필요하다. 만약 이 과정에서 보안 공격을 당한다면 민감한 개인 자료들이 외부에 노출될 수 있다.

이러한 단점을 보완하기 위해 본 논문에서는 임베디드 자료 동기화 게이트웨이를 제안한다. 제안된 게이트웨이는 기존 게이트웨이와 달리 각 사용자의 모바일 단말기 내에 탑재되고, 단말기 내에서 자료를 추출하여 변환한다. 임베디드 자료동기화 게이트웨이의 성능을 위한 시뮬레이션에 의하면 기존의 고정식 자료동기화 게이트웨이에 비해 병목현상을 완화할 수 있다[6]. 본 논문에서는 임베디드 자료동기화 게이트웨이 개발을 위한 연구의 일환으로서 윈도우 모바일 기반의 PDA에서 운영되는 임베디드 자료동기화 게이트웨이를 구축하고 결과를 제시한다.

제안된 게이트웨이는 데이터스토어 어댑터와 동기화 엔진으로 구성된다. 데이터스토어 어댑터는 단말기 내에서 사용되는 데이터를 추출하고 저장한다. 동기화 엔진은 추출된 데이터를 동기화 메시지로 변환하거나 서버로부터 수신한 메시지를 해석한다. 개발된 임베디드 자료동기화 게이트웨이의 정상적인 동기화 수행 여부를 검증하기 위해서 자료동기화 서버와 연동하여 실험을 하였고, 동기화 대상 데이터의 증가에 따른 동기화 소요시간을 측정하였다. 실험 결과, 자료 동기화가 정상적으로 수행됨을 검증할 수 있었고, 데이터가 증가함에 따라 동기화 소요시간이 완만하게 증가함을 알 수 있었으며, 심각한 병목현상이 초래되지 않았다.

본 논문의 구성은 다음과 같다. 2장에서는 다양한 자료 동기화 방식에 대한 분석과 자료동기화 게이트웨이 방식에 대해 고찰한다. 3장은 제안된 임베디드 자료동기화 게이트웨이의 구조, 동작 메커니즘 및 구현 결과를 서술하며, 4장에서는 개발된 시스템의 기능 및 성능에 대한 검증 결과를 제시한다, 마지막으로 5장에서는 결론 및 향후 결과에 대해서 제시한다.

II. 관련연구

2.1 자료동기화 프로토콜

ActiveSync는 윈도우 모바일 또는 윈도우 CE 기반의 모바일 단말기를 위한 자료동기화 프로토콜이며 ActiveSync 제공자와 ActiveSync 서비스 매니저로 구성되어 있다. ActiveSync는 투웨이 싱크(Two-way Sync)와 슬로우 싱크(Slow Sync)에 대한 동기화 모드를 지원하고 있다[2]. ActiveSync의 내부 구조는 공개되지 않고 서로 다른 동기화 프로토콜 간의 상호 호환성도 보장하지 않는다.

HotSync는 Palm OS를 사용하는 단말기를 위한 자료 동기화 프로토콜이다. HotSync 프로토콜은 SyncML 프로토콜의 투웨이 싱크와 유사한 슬로우 싱크와 패스트 싱크 두 가지 자료 동기화 모드를 지원한다. 패스트 싱크의 경우 기존에 자료 동기화 작업을 수행한 PC에서만 수행된다. HotSync에서는 패스트 싱크를 사용할 수 없는 경우 슬로우 싱크가 수행된다[3]. ActiveSync와 마찬가지로 프로토콜 내부 구조를 공개하지 않고 서로 다른 동기화 프로토콜 간의 상호 호환성도 보장하지 않는다.

OMA DS(Open Mobile Alliance Data Synchronization)는 XML 형식의 메시지를 통하여 동기화를 수행하는 국제 표준안이다. OMA DS는 프로토콜 내부 구조를 공개하고 있다[7]. 하지만 OMA DS 또한 모든 단말기 제조업체들이 이 방식을 수용하고 따를 때 상호호환성을 보장할 수 있을 뿐, ActiveSync 또는 HotSync와 같은 독자적인 프로토콜을 사용하는 단말기와는 자료동기화를 수행할 수 없다. 본 논문에서 제안하는 임베디드 자료동기화 게이트웨이는 OMA DS를 기준으로 하여 데이터를 변환한다.

2.2 고정식 자료동기화 게이트웨이

고정식 자료동기화 게이트웨이 시스템은 그림 1과 같이 외부의 데스크톱 서버에 설치되어 여러 단말기들의 자료동기화 요청을 처리한다[4-5].

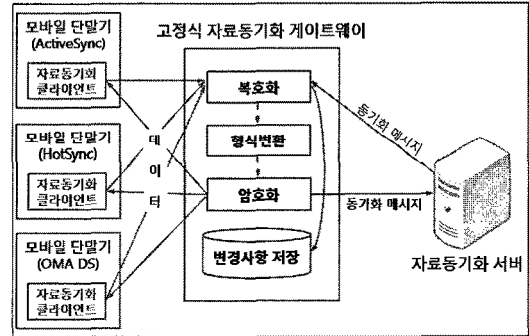


그림 1. 고정식 자료동기화 게이트웨이 구조
Fig. 1 The Architecture of the Stationary DS Gateway

고정식 자료동기화 게이트웨이의 경우 각 모바일 단말기는 자료동기화 클라이언트를 탑재하고 있다. 자료동기화 클라이언트는 단말기 응용프로그램이 사용하는 데이터를 추출 또는 저장한다. 고정식 자료동기화 게이트웨이 시스템의 자료동기화 과정은 다음과 같이 6단계로 처리된다.

- 1) 자료동기화 클라이언트로 또는 서버로부터 데이터 또는 동기화 메시지를 수신
- 2) 데이터 또는 메시지를 복호화
- 3) 복호화한 자료를 변환
- 4) 변환된 자료를 암호화
- 5) 자료동기화 클라이언트 또는 서버로 전송
- 6) 변경사항이 발생한 경우 변경사항을 저장

이와 같이 고정식 게이트웨이 시스템에서는 여러 단말기로부터 수신한 자료를 변환하여야 하고, 이를 위해 자료를 복호화 및 암호화하는 작업이 추가적으로 필요하다. 이로 인해, 동시에 여러 대의 단말기로부터 자료동기화 요청을 받을 경우, 게이트웨이 시스템에 병목현상이 발생할 수 있다. 또한 만약 단계 2)와 3) 사이 보안 공격이 발생한다면, 모든 사용자들의 개인 자료가 외부에 유출될 위험이 존재한다.

Ⅲ. 임베디드 자료 동기화 게이트웨이

3.1 시스템 구조

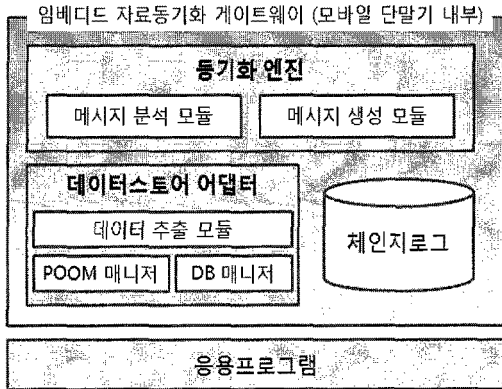


그림 2. 임베디드 자료동기화 게이트웨이 구조
Fig. 2 The Architecture of the Embedded DS Gateway

임베디드 자료동기화 게이트웨이는 그림 2와 같이 크게 데이터스토어 어댑터, 체인지로그, 동기화 엔진으로 구성된다. 데이터스토어 어댑터[8]는 POOM(Pocket Outlook Object Model) 매니저, DB 매니저, 데이터 추출 모듈로 구성된다. POOM 매니저는 응용프로그램에서 사용하는 데이터를 추출·저장 및 수정하며, DB 매니저는 체인지로그 DB에 저장된 해쉬값을 추출·저장 또는 수정한다. 데이터 추출 모듈은 POOM 매니저와 DB 매니저로부터 전달받은 값을 비교하여 실제 동기화 대상이 될 데이터를 추출한다.

동기화 엔진은 메시지 생성 모듈과 분석 모듈로 구성된다. 메시지 생성 모듈은 데이터스토어 어댑터로부터 전달받은 데이터와 동기화 명령을 OMA DS 프로토콜의 메시지로 변환하여 자료동기화 서버로 전송한다. 메시지 분석 모듈은 자료동기화 서버로부터 수신한 동기화 메시지를 분석하여 동기화 명령과 데이터를 추출한 후 데이터스토어 어댑터에게 전달한다.

3.2 동작 메커니즘

임베디드 자료동기화 게이트웨이의 동작 메커니즘은 그림 3과 같으며, 동기화가 시작되면 POOM 매니

저는 POOM 라이브러리[9-10]를 사용하여 응용프로그램이 사용하는 데이터를 추출하거나 저장한다. POOM은 윈도우 모바일 또는 윈도우 CE 기반의 단말기 응용프로그램의 PIMS(Personal Information Management System) 정보를 나타내는 클래스들의 집합이다. POOM 매니저가 POOM 라이브러리를 사용하여 데이터를 추출, 저장, 삭제, 수정하는 과정은 다음과 같이 3단계로 구성된다.

- 1) OutlookSession 객체를 생성하여 POOM 세션 시작
- 2) ItemID 객체를 사용하여 응용프로그램이 사용하는 데이터의 고유한 ID인 LUID 추출
- 3) 데이터 추가·삭제·수정: 데이터 추가 시 데이터를 매개변수로 하는 Add() 메소드 호출하고 새롭게 할당된 LUID를 반환. 데이터 삭제 시 LUID를 매개변수로 하는 Remove() 메소드 호출. 데이터 수정 시 LUID와 데이터를 매개변수로 하는 Update() 메소드 호출

DB 매니저는 POOM 매니저가 추출한 데이터의 LUID를 참고하여 쿼리문을 사용하여 체인지로그에 저장된 해쉬값을 추출한다. 체인지로그는 윈도우 모바일에서 제공하는 CEDB를 사용하며 응용프로그램이 사용하는 데이터와 쌍을 이루는 LUID와 해쉬값을 가지고 있다. CEDB는 윈도우 모바일 기반의 단말기에서 사용하는 데이터베이스이다[11]. 데이터베이스 매니저가 CEDB 쿼리를 사용하여 해쉬값을 추출, 저장, 삭제, 수정하는 과정은 다음과 같다.

- 1) SqlConnection 객체를 생성하여 데이터베이스에 접속
- 2) POOM 매니저가 추출한 데이터의 LUID와 동일한 레코드를 선택하기 위해 ExecuteReader() 메소드를 사용하여 SELECT 쿼리문을 실행
- 3) ExecuteNonQuery() 메소드를 사용하여 INSERT, DELETE, UPDATE 쿼리문 실행

데이터 추출 모듈은 POOM 매니저와 데이터베이스 매니저로부터 전달받은 한 쌍의 데이터를 비교하여 동기화 명령(변경사항)과 LUID, 데이터를 메시지 생성 모듈에 전달한다.

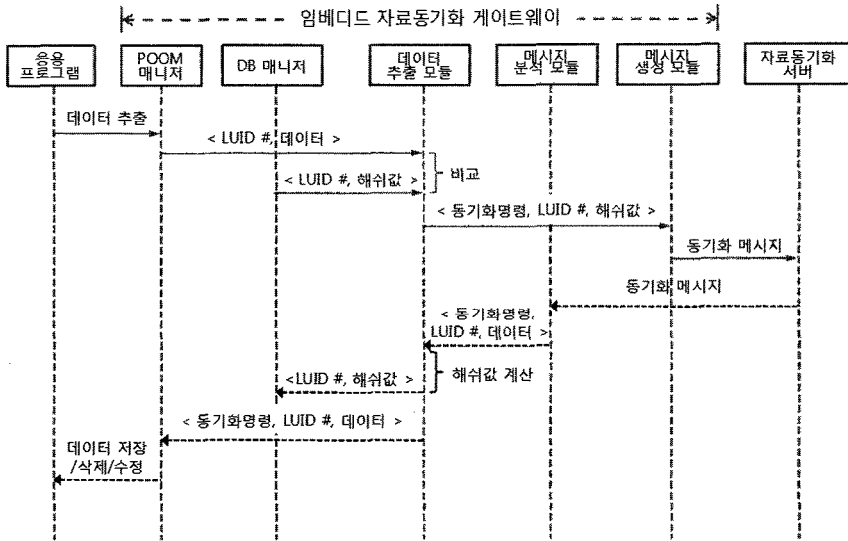


그림 3. 임베디드 자료동기화 게이트웨이 처리 흐름도
Fig. 3 Sequence Diagram of the Embedded DS Gateway

그림 4는 데이터 추출 모듈에서 전달하는 동기화데이터의 구조를 보여준다. 특정 데이터의 동기화 명령(변경사항)을 결정하는 조건은 다음과 같다.

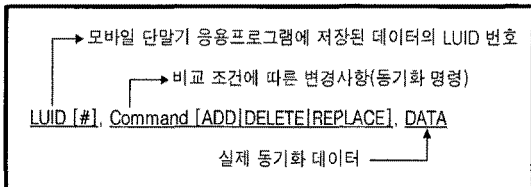


그림 4. 동기화 데이터 구조
Fig. 4 The Structure of DS DATA

표 1. 동기화 명령
Table 1 Synchronization Commands

변경 상태	조건
추가	응용프로그램의 데이터 LUID가 데이터베이스 내에 존재하지 않을 경우
삭제	데이터베이스의 데이터 LUID가 응용프로그램 내에 존재하지 않을 경우
수정	응용프로그램과 데이터베이스에 동일한 LUID가 존재하지만 해쉬값이 다를 경우

메시지 분석 모듈로부터 동기화 대상 데이터와 동기화 명령을 수신하면 추가(ADD)/삭제(DELETE)/수정(REPLACE)과 같은 동기화 명령과 LUID, 데이터를 POOM 매니저에게 전달한다. 또한 데이터의 해쉬값을 계산하여 LUID와 함께 데이터베이스 매니저에게 전달한다.

메시지 생성 모듈은 데이터 추출 모듈로부터 전달받은 동기화 데이터를 OMA DS에서 규정한 XML 형식의 동기화 문서로 변환한다. 동기화 문서는 헤더 부분과 바디 부분으로 구성되며, 헤더 부분은 동기화 메시지에 대한 정보와 인증 정보를 포함하고 있다. 바디 부분은 실제 동기화 대상이 되는 데이터와 명령을 포함하고 있다.

메시지 분석 모듈은 동기화 서버로부터 전달받은 동기화 메시지로부터 동기화 명령과 데이터를 추출하여 데이터 추출 모듈에게 전달한다. 데이터 추출 모듈은 동기화 명령에 따라 POOM 매니저와 DB 매니저에게 데이터를 추가/삭제/수정할 것을 요청한다.

3.3 구현 결과

제안된 임베디드 자료 동기화 게이트웨이는 윈도우 모바일 기반의 모바일 단말기 내에 탑재될 수 있도록

개발되었다. 개발된 게이트웨이 시스템은 윈도우 모바일에서 사용하는 ActiveSync 동기화 프로토콜 데이터를 OMA DS 동기화 메시지로 변환하거나 그 반대의 기능을 한다. 이 게이트웨이 시스템은 C# 을 사용하여 구현하였고, 윈도우 모바일 기반의 에뮬레이터에서 실행되었다. 또한 단말기 내의 자료 추출을 위해 POOM이라고 불리는 COM 기반의 라이브러리를 사용하였다[9]. 임베디드 자료동기화 게이트웨이의 실행 코드 크기는 265KB로 단말기 내에 탑재되기 적당한 크기이다.

그림 5는 윈도우 모바일 5.0 기반의 에뮬레이터에 탑재된 게이트웨이의 조작 화면을 나타낸다. Sync 버튼을 클릭하면 자료동기화 서버와의 동기화 세션이 시작된다.

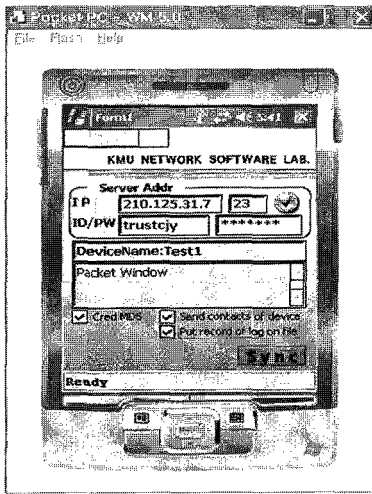


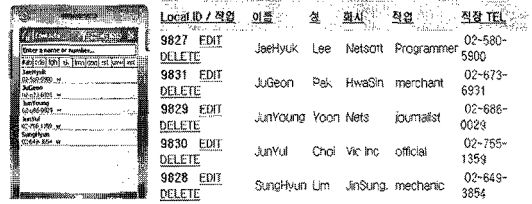
그림 5. 임베디드 자료동기화 게이트웨이 조작 화면
Fig. 5 Operation Screens of the Embedded DS Gateway

IV. 성능 분석

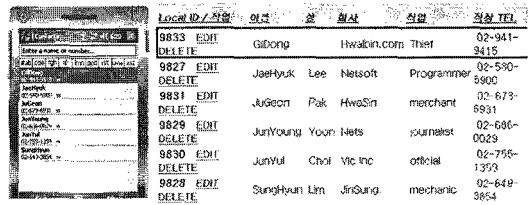
제한한 임베디드 자료동기화 게이트웨이의 기능 및 성능에 대한 적합성을 검증하기 위해 OMA DS 기반의 Synthesis 서버[12]와 연동하여 자료 동기화를 수행하였고 자료 동기화 시간을 측정하였다.

4.1 기능 검증

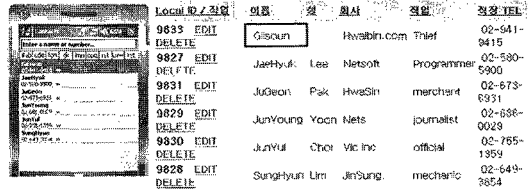
그림 6은 연동시험 결과를 나타낸다. 그림 6에서 왼쪽 그림은 임베디드 자료동기화 게이트웨이가 탑재된 에뮬레이터의 화면이고, 오른쪽 그림은 Synthesis 자료동기화 서버의 화면을 나타낸다.



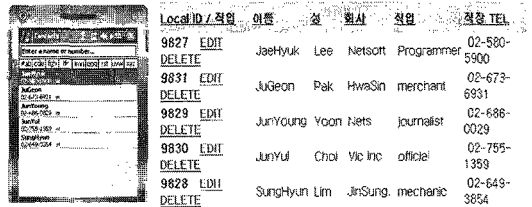
(a)



(b)



(c)



(d)

그림 6 임베디드 자료동기화 게이트웨이 동기화 결과 (a) 초기 동기화 결과 (b) 데이터 추가 후 동기화 결과 (c) 데이터 수정 후 동기화 결과 (d) 데이터 삭제 후 동기화 결과

Fig. 6 Synchronization Result of the Embedded DS Gateway (a) in the beginning (b) after data insertion (c) after data modification (d) after data deletion

그림 6(a)에서 에뮬레이터에 5명의 연락처를 저장하고 동기화를 수행한 결과, 서버에 5명의 연락처가 추가되었음을 알 수 있다. 그림 6(b)에서 에뮬레이터에 GilDong 이라는 사람의 연락처를 추가하고 동기화를 수행한 결과, 서버에 GilDong의 연락처가 추가되었음을 알 수 있다. 그림 6(c)에서는 에뮬레이터에 저장된 연락처 중 GilDong이라는 사람의 이름을 GilSoun으로 수정한 후, 동기화를 수행한 결과 서버에 GilDong이라는 이름이 GilSoun으로 수정되었음을 알 수 있다. 그림 6(d)에서 GilSoun를 삭제하고 동기화를 수행한 결과 서버에 GilSoun이 삭제되었음을 알 수 있다. 위 실험을 통해 본 논문에서 개발한 임베디드 자료동기화 게이트웨이의 정상적인 동작 여부를 검증할 수 있다. 또한 OMA DS 기반의 Synthesis 서버와 정상적으로 연동됨을 통해 OMA DS 표준도 만족한다는 것을 검증할 수 있다.

4.2 성능 평가

제안한 임베디드 자료동기화 게이트웨이의 성능을 평가하기 위해 Synthesis 서버와 동기화 수행 중 소요되는 시간을 측정하였다. 그림 7은 동기화에 따른 소요 시간을 그래프로 나타내고 있다. 동기화 대상 데이터가 증가함에 따라, 동기화에 소요되는 시간을 측정하기 위해 데이터를 1개에서 40개까지 5개씩 증가시키며 소요 시간을 측정하였다. 동기화 데이터의 개수가 증가함에 따라 소요시간의 증가율은 평균 53%로 완만하게 증가함을 알 수 있으므로, 자료동기화에 따른 심각한 병목 현상이 나타나지 않음을 알 수 있다.

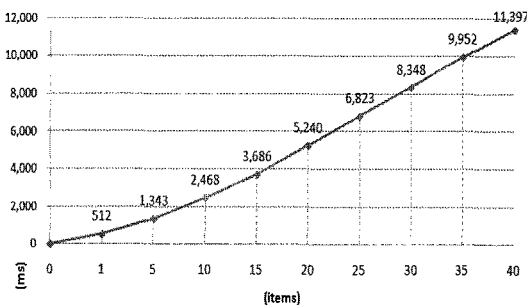


그림 7. 동기화 소요 시간
Fig. 7 Synchronization Time

V. 결론

본 논문에서는 서로 다른 자료동기화 프로토콜 간 상호 호환성을 보장하기 위한 방안으로서 임베디드 자료동기화 게이트웨이를 제안하고, 구축 결과를 제시하였다. 제안된 게이트웨이 시스템은 기존에 개발한 게이트웨이가 외부 데스크톱 서버에 설치되는데 반해, 각 사용자의 모바일 단말기 내에 탑재되고, 단말기 내에서 자료를 추출하여 변환하는 시스템이다. 임베디드 자료동기화 게이트웨이는 C#을 사용하여 개발되었으며, 윈도우 모바일 5.0 기반의 에뮬레이터에서 실행되었다.

임베디드 자료동기화 게이트웨이의 기능 및 성능적 검증을 위하여 OMA DS 기반의 Synthesis 서버와 연동 테스트를 하였으며, 동기화 시 소요되는 시간을 측정하였다. 검증 결과 본 게이트웨이가 정상적으로 동작함을 알 수 있었으며, 동기화 데이터의 개수가 늘어남에 따라 소요시간이 완만하게 증가함을 알 수 있었고, 자료동기화에 따른 심각한 병목 현상이 나타나지 않음을 알 수 있었다.

향후 연구 과제로서는, 다양한 자료 동기화 방식을 지원할 수 있는 확장된 임베디드 자료동기화 게이트웨이에 대한 추가적인 연구가 필요하다. 또한, 게이트웨이의 각 모듈에 대한 세부적인 코드 최적화를 통하여, 성능을 향상시킬 수 있는 방안에 대한 연구가 요구된다.

참고문헌

- [1] Uwe Hansmann, Riku Mettala, Apratim Purakayastha, Peter Thompson, SyncML Synchronizing and Managing Your Mobile Data, New Jersey : PRENTICE HALL PTR, 2003.
- [2] ActiveSync, Microsoft Mobile, <http://www.microsoft.com/windowsmobile/default.msp>
- [3] Palm OS, Palm, <http://www.palm.com>
- [4] 장대진, 박기현, 주홍택, 우종정, “ActiveSync 자료 변환을 위한 SyncML 자료 동기화 게이트웨이”, 한국인터넷정보학회 논문지, 제 7권, 제 3호, pp. 61-69, 2006.
- [5] 장대진, 박기현, “다양한 이동통신 단말기들 간의

상호 운용을 위한 SyncML 기반의 통합 자료 동기화 게이트웨이 시스템”, 한국정보과학회 논문지: 컴퓨팅의 실제 및 레터, 제 14권, 제 2호, pp. 117-129, 2008.

- [6] 박기현, 우종정, “임베디드 모바일 자료동기화 게이트웨이 성능 분석”, 한국정보기술학회 논문지, 제 7 권, 제 3호, pp. 1-7, 2009.
- [7] OMA Forum, OMA Data Synchronization Protocol Specification, www.openmobilealliance.org
- [8] 박주건, 박기현, 이근진, “임베디드 자료동기화 게이트웨이를 위한 데이터스토어 어댑터”, 한국정보과학회 2009년 가을 학술발표대회 논문지, 제36권, 제2호(D), pp. 152-155, 2009.
- [9] POOM API, <http://msdn.microsoft.com>
- [10] Boling, D., PROGRAMMING MICROSOFT WINDOWS CE .NET : Microsoft Press, 2004
- [11] CEDB API, <http://msdn.microsoft.com>
- [12] Synthesis AG - Synthesis AG, <http://www.synthesis.ch>



우 종 정 (Jongjung Woo)

1982년 경북대학교 전자계산학
학사

1990년 텍사스 주립대학(Austin)
석사

1993년 : 텍사스 주립대학(Austin) 박사

1982년 ~ 1988년 : KIET 책임연구원

1993년 ~ 현재 : 성신여자대학교 교수

※관심분야: 임베디드 소프트웨어, 모바일 컴퓨팅,
컴퓨터구조, 병렬처리, e-learning

저자소개

박 주 건 (Ju-Geon Park)



2006년 계명대학교 컴퓨터공학과
학사

2008년 계명대학교 컴퓨터공학과
석사

2009년~현재 : 계명대학교 컴퓨터 공학과 박사과정
※관심분야: 임베디드 소프트웨어, 단말기 관리, 자료
동기화

박 기 현 (Kee-Hyun Park)



1979년 경북대학교 전자계산학
학사

1981년 한국과학기술원
전자계산학 석사

1990년 : 미국 Vanderbilt 대학교 전자계산학 박사
1981년 ~ 현재 : 계명대학교 컴퓨터공학과 교수
※관심분야 : 병렬처리, 모바일 소프트웨어, 임베디드
소프트웨어, 운영체제