韓國컴퓨터情報學會 論文誌
第16卷 第5號, 2011. 05.

2011-16-5-3-3

OMA DS 표준을 지원하는 자료동기화 서버 구축 및 적합성 검증

박 주 건*, 박 기 현*

Construction and Validation of a Data Synchronization Server supporting OMA DS Standards

Ju Geon Pak*, Kee Hyun Park*

요 약

본 논문에서는 모바일 통신 환경을 위한 자료동기화 서버를 구축하였으며, 구축된 서버의 적합성 및 성능 검증 결과를 제시한다. 본 논문에서 구축한 자료동기화 서버는 클라이언트(모바일 단말기)가 최신의 자료를 유지할 수 있게 도와주며, 클라이언트와 서버 간 자료의 일치성을 보장하도록 한다. 또한 다양한 자료동기화 타입을 제공하며, 자료의 변경 및 충돌을 탐지하고 충돌 발생 시 이를 해결하기 위한 정책을 제공한다. 게다가, OMA(Open Mobile Alliance) DS(Data Synchronization) 프로토콜을 준수함으로써, 상호 호환성을 보장하도록 설계 구현되었다. 뿐만 아니라, 기존의 자료동기화 서버들이 사용하는 레코드 단위 전송방식의 단점을 보완하기 위해 필드 단위 전송 방식도 지원하도록 구현되었다. 구축된 자료동기화 서버의 기능 및 성능 적합성을 검증하기 위해 OMA에서 제공하는 적합성 검증 도구인 SCTS(SyncML Conformance Test Suit)와의 동기화를 수행하였으며, 다양한 상황에서의 성능을 평가하였다. 검증 결과, SCTS 항목 중 대용량 오브젝트 기능을 제외한 모든 검증 항목을 만족함을 알 수 있었다. 대용량의 오브젝트를 분할하여 전송하는 기능은, 개인정보 동기화 목적의 본 연구에는 필요치 않아 구현하지 않았으며, 향후 구현할 예정이다. 서버의 성능을 평가하기 위한 동기화 소요시간 측정 결과, 본 논문의 자료동기화 서버는 동기화 자료 및 클라이언트 수의 증가에 따른 동기화 소요시간의 증가가 완만함을 확인할 수 있었으며, 확장성이 있다고 판단하였다. 또한, 제안된 서버와 동일한 프로토콜을 사용하는 SCTS 서버, Synthesis와 성능을 비교 한 결과, SCTS 서버에 비해 제안된 서버를 사용하였을 때가 동기화에 요구되는 시간이 더 짧음을 알 수 있었다. Synthesis 서버와 비교해보면, 제안된 서버가 더 많은 동기화 소요시간을 요구하지만, 동기화 할 자료의 수가 많고 동기화 대상 자료(레코드)의 일부 필드만 변경된 상황에서는 제안된 서버가 더 우수함을 알 수 있었다. 즉, 제안된 서버는 기존의 자료동기화 서버들과 비교해 동일하거나 추가된 기능을 제공하면서도, 동기화 자료의 수가 증가할수록 더 우수한 성능을 보여줌을 뜻한다. OMA DS를 준수하는 자료동기화 서버를 직접 구축해봄으로써, 향후 모바일 DS 기능을 개선하기 위한 다양한 연구의 기초 프레임이 될 것이라고 기대한다.

▸ Keyword : 모바일 자료동기화, 자료동기화 서버, OMA DS, SCTS

• 제1저자 : 박주건 • 교신저자 : 박기현
• 투고일 : 2010. 11. 18, 심사일 : 2010. 12. 24, 게재확정일 : 2011. 02. 13.
* 계명대학교 컴퓨터공학과(Dept. of Computer Engineering, Korea University)
※ 이 논문은 2010년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임 (No. 2010-0016454)

Abstract

In this paper, a DS (Data Synchronization) server for mobile communication environments is constructed and the suitability and the performance of its operations are validated. The DS server provides a way to update the newest data and keep data consistency for clients (mobile devices). In addition, the DS server constructed in this paper supports various synchronization types, and detects all changes and conflicts. In case of data conflicts, the DS server resolves the conflicts according to the several policies implemented in this work. The DS server conforms to the OMA(Open Mobile Alliance) DS standard protocol for interoperability with other mobile devices and servers. In addition to the transmission-by record scheme proposed by the OMA DS standard protocol, the DS server constructed in this paper also provides the transmission-by field scheme for the enhancement transmission performance between the server and clients. In order to validate its operations, data synchronization between the DS server and the SCTS (SyncML Conformance Test Suit), the suitability validation tool provided by the OMA, is performed. The validation results show that the DS server constructed in this paper satisfies all of the test cases except the Large Object function. The Large Object function will be implemented later because the function is not needed for the personal information synchronization process which this paper aims for. Also, synchronization times of the DS server are measured while increasing the number of data and clients. The results of the performance evaluations demonstrate that the DS server is scalable, in the sense that it has not suffered from any serious bottlenecks with respect to the number of data and clients. We expect that this work will provide a framework for various studies in the future for improving mobile DS operations.

▸ Keyword : Mobile Data Synchronization, Data Synchronization Server, OMA DS, SCTS

I. 서 론

자료동기화 시스템은 분산된 단말기 간 자료의 일치성을 보장하기 위한 시스템으로서, 최근 스마트폰, PDA 등과 같은 모바일 단말기의 보급으로 인해 구축된 모바일 통신 환경에서 중요성이 더욱 커지고 있다. 자료동기화 시스템은 클라이언트와 서버로 구성되는데, 동기화 클라이언트와 서버 간 교환되는 메시지의 형식 및 절차를 정의하기 위해 다수의 자료동기화 프로토콜들이 개발되었으며, 이 중 마이크로소프트사의 액티브싱크(ActiveSync)[1], 팜사의 핫싱크(HotSync)[2]. 그리고 현재 산업계 표준으로 받아들여지고 있는 OMA(Open Mobile Alliance) DS(Data Synchronization)[3] 등이 널리 사용되고 있다. 자료동기화 시스템에서 클라이언트는 동기화를 위해, 단말기에서 생성된 자료를 추출하여 서버에게 전송하며, 서버는 클라이언트로부터 수신한 자료와 서버 내 저장된 자료를 사전에 정의된 충돌해결 정책 (서버자료 우선, 클라이언트 자료 우선, 최신 자료 우선 등)에 따라 비교한 후, 클라이언트에게 동기화 명령을 전송한다.

본 연구팀은 이전 연구에서 모바일 단말기를 위한 자료동기화 클라이언트를 개발하였으며, 이를 외부에서 제작된 자료동기화 서버와 연결하여 다양한 연구를 시도하였다[4-6]. 해당 연구들에서는 상용 자료동기화 서버인 액티브싱크 서버, 핫싱크 서버 그리고 OMA DS 기반의 Synthesis 서버[7] 등이 사용되었다. 자료동기화 시스템에서 서버는 최적의 자료동기화 방식을 결정하고, 변경 사항을 탐지하며, 자료의 충돌을 탐지 및 해결하는 등 전체 자료동기화 시스템의 효율성, 신뢰성 및 성능을 결정하는 핵심 요소이므로 효율적인 자료동기화 서버 구축에 관한 연구가 반드시 필요하다. 하지만, 상용 서버들은 그 특성상 내부 소스코드를 공개하고 있지 않아 확장과 수정이 불가능하기 때문에, 동기화의 효율성을 높일 수 있는 여러 가지 방안 및 연구를 수행하기에는 많은 제약이 따른다. 또한, 해당 자료동기화 서버들은 모두 레코드 단위

전송 방식을 사용한다. 레코드 단위 전송 방식은 특정 자료가 가지는 다수의 필드(속성) 값 중 하나만 변경되어도 모든 필드 값들은 전송하는 방식이다. 따라서, 연락처, 일정과 같이 다수의 필드 값들을 가지는 자료의 동기화에서는 효율성이 떨어진다는 단점이 있다.

이에, 본 논문에서는 효율적인 자료동기화 방식에 관한 연구의 일환으로서 자체 자료동기화 서버를 설계 및 구축하였으며, 구축한 자료동기화 서버의 기능 및 성능 적합성을 검증하였다.

본 논문의 자료동기화 서버는 현재 개발된 다수의 자료동기화 프로토콜들을 분석하여 도출된 요구 사항과, 앞서 연구된 설계 결과를[8] 바탕으로 구축되었으며, 상호 호환성을 높이기 위해 OMA DS 프로토콜[3]도 수용하고 있다. 개발된 자료동기화 서버는 동기화 상황에 따라 적절한 자료동기화 방식을 결정하며, 변경 사항을 탐지하고 자료의 충돌을 탐지 및 해결하기 위한 다양한 정책을 제공한다. 뿐만 아니라 기존의 자료동기화 서버들이 사용하는 레코드 단위 전송 방식의 단점을 보완하기 위해 필드 단위 전송 방식도 지원하도록 구현하였다.

본 논문에서 제안된 동기화 서버는 OMA DS 프로토콜을 준수하므로, 해당 프로토콜과의 기능 적합성을 객관적으로 검증할 수 있어야 한다. 이에 SCTS (SyncML Conformance Test Suit) 클라이언트와[9]의 연동 테스트를 수행하였다. SCTS는 OMA에서 제공하는 적합성 검증 도구로서, SCTS와의 적합성 검증은 OMA 국제인증 획득을 위한 OMA TestFest[10]에 참가하기 위해 선행되어야 하는 필수 조건이다. 따라서 SCTS를 통해 구축한 자료동기화 서버의 기능 적합성을 객관적으로 검증할 수 있다.

테스트 결과, SCTS가 제시한 필수 평가 항목 중 대용량 오브젝트 (Large Object) 항목을 제외한 모든 항목을 만족함을 알 수 있었다. 따라서, 본 논문의 자료동기화 서버는 대용량 오브젝트 기능을 제외하고는, OMA에서 제시하는 기능 적합성을 만족하고 OMA DS 프로토콜을 준수함을 뜻한다. 대용량 오브젝트는 동기화 대상이 되는 단위 오브젝트가 클라이언트가 수용할 수 있는 용량을 초과할 경우, 복수 개의 메시지로 나눠서 전송하는 기능이다[11]. 본 논문의 자료동기화 서버는 우선적으로 개인정보관리 시스템 (Personal Information Management System)의 자료 (예: 연락처, 일정, 이메일 등)를 동기화 대상으로 하고 있고, 이러한 자료들은 용량이 크지 않기 때문에 현재 구현하지 않았지만, 향후 구현할 예정이다.

본 논문에서 제안된 자료동기화 서버의 성능 적합성을 검증하기 위해, 본 연구팀이 앞서 개발한 자료동기화 게이트웨이[6]와 연동하여 동기화 자료 및 클라이언트 수의 증가에 따른 동기화 소요시간과 각 동기화 패키지별 처리 시간을 측정하였다. 측정 결과, 제안된 자료동기화 서버는 동기화 자료 및 클라이언트 수의 증가에 따른 소요시간의 증가가 완만함을 검증할 수 있었다. 뿐만 아니라, 본 논문의 자료동기화 서버는 변경된 자료를 레코드 단위가 아닌 필드 단위로 동기화하는 방식을 아울러 제공하기 때문에 제안된 서버와 동일한 프로토콜을 사용하는 타 서버에 비해 성능이 우수함을 알 수 있었다.

본 논문의 구성은 다음과 같다. 2장은 관련 연구 부분으로 효율적인 자료동기화 서버의 설계 및 구축을 위한 요구사항을 도출하고, 도출된 요구사항을 바탕으로 현재 출시되어 사용되고 있는 자료동기화 서버들을 비교 분석한 결과를 제시한다. 또한, 본 논문에서 제안된 동기화 서버의 근간이 되는 OMA DS 프로토콜과의 기능 적합성을 검증하기 위한 도구인 SCTS에 대해 설명한다. 3장에서는 자료동기화 서버의 구축 결과를 제시하며, 4장은 자료동기화 서버의 기능 및 성능 적합성 검증 부분으로서, SCTS 적합성 검증 결과 및 성능 측정 결과를 제시한다. 마지막으로 5장에서는 본 논문의 결론과 향후 연구 계획에 대해 논한다.

II. 관련 연구

1. 자료동기화 서버

최근 모바일 분산처리 환경이 가속화됨에 따라, 각 모바일 단말기에서 처리된 데이터 또는 자료들에 대한 동기화가 점점 중요한 요소로 고려되고 있으며, 이에 다양한 자료동기화 서버를 포함한 시스템들이 제안되었다. 본 장에서는, 효율적인 자료동기화 서버의 설계 및 구축을 위한 요구사항을 도출하고, 도출된 요구사항을 바탕으로 현재 출시되어 사용되고 있는 자료동기화 서버들을 비교 분석한다. 현재까지 개발된 다양한 자료동기화 서버 중, 가장 널리 사용되고 있다고 판단되어지는 윈도우 모바일 기반의 액티브싱크 서버, Palm OS 기반의 핫싱크 서버, OMA DS 기반의 SCTS, Synthesis 서버, 그리고 CPISync 서버[12,13]를 비교 분석의 대상으로 삼았다.

- 자료동기화 프로토콜 : 자료동기화 프로토콜은 클라이언트와 서버 간 교환하는 동기화 메시지의 형식 및 절차를 정의한다. 현재까지 다양한 자료동기화 프로토콜들이 제안되었지만 널리 사용되고 있는 자료동기화 프로토콜로는 액티브싱

크[1], 핫싱크[2] 그리고 OMA DS[3]가 있다. 액티브싱크 프로토콜은 마이크로소프트사에서 개발한 동기화 프로토콜로 윈도우즈 CE 또는 윈도우즈 모바일 기반의 단말기에서 동작하며 핫싱크는 팜사에서 개발한 동기화 프로토콜로 팜OS 기반의 단말기에서 동작한다. OMA DS 프로토콜은 모바일 제조사 및 통신사들이 연합하여 제안한 표준안으로 현재 산업계 표준으로 받아들여지고 있으며 플랫폼 독립적이다. 액티브싱크 서버는 액티브싱크 프로토콜을 핫싱크 서버와 CPISync 서버는 핫싱크 프로토콜을 사용하며, SCTS와 Synthesis 서버는 OMA DS 프로토콜을 사용한다. 자료동기화 프로토콜은 자료동기화 서버의 상호호환성과 밀접한 관련이 있다. 자료동기화 서버는 통신 프로그램이기 때문에, 클라이언트와 서버 간 프로토콜이 일치하지 않는다면 동작하지 않기 때문이다. 따라서 공개된 표준안인 OMA DS 프로토콜을 사용하는 것이 바람직하다. OMA DS 프로토콜은 현재 산업계 표준으로 받아들여지고 있기 때문에, OMA DS 프로토콜을 사용함으로써 기존에 운용되고 있는 여러 자료동기화 시스템과의 상호 호환이 가능해진다.

- 변경 사항 탐지 : 자료동기화 시스템에서 서버는 서버 측 변경 사항을 클라이언트에게 전송하는데, 동기화 효율성을 높이기 위해서는 변경된 자료만을 정확히 탐지할 수 있어야 한다. 대표적인 변경사항 탐지 방법으로는, 상태 플래그 방식. 타임스탬프 방식이 있다. 상태 플래그 방식은 각 자료마다 플래그를 두어 마지막 동기화 이후 변경된 자료의 플래그를 특정 값으로 설정하여 구분하는 방식이다. 상태 플래그 방식을 사용하는 대표적인 자료동기화 서버로는 핫싱크 서버를 들 수 있다. 타임스탬프 방식은 각 자료마다 동기화 된 시각을 기하는 타임스탬프를 두어, 마지막 동기화 시각보다 늦은 시각을 가지는 자료를 선택하여 전송하는 방식이다. 액티브싱크 서버와 OMA DS를 준수하는 서버인 SCTS 서버, Synthesis 서버가 타임스탬프를 사용하는 대표적인 자료동기화 서버이다. 이외에도, 자료를 특성 다항식(characteristic polynomial) 형태로 저장하고 특성 다항식을 보간(interpolation) 및 인수분해(factorize)하여 동기화 하는 방식의 CPISync(Characteristic Polynomial Interpolation Sync) 서버[12,13]와 동기화 자료의 해싱 값을 주소로 하여 특정 배열의 값을 1로 설정하여 동기화하는 Bloom Filter[14]가 있다. CPISync와 Bloom Filter 방식은 각 자료에 대한 상태 정보를 저장할 필요가 없다는 장점이 있지만, 자료의 변경 여부만을 알 수 있을 뿐 변경된 시각을 알 수 없다는 단점이 있다. 즉, 뒤에서 설명하게 될 자료 충돌 해결 정책 중 최신 자료 우선 정책은 사용할 수 없다. 일정 및 연락처 정보와 같은 자료의 충돌을 해결하기 위해서는 자료가 갱신된 시점은 매우 중요한 요소이므로, 타임스탬프 방식이 적용되어야 한다.

- 동기화 타입 : 동기화를 위해 클라이언트와 서버 간 교환되는 자료의 선택 방식 및 교환 방식을 동기화 타입이라 하며, 자료 동기화 서버는 클라이언트와 서버의 상황을 고려하여 효율적인 동기화 타입을 결정할 수 있어야 한다. 동기화 타입은 상호 교환되는 동기화 대상 자료의 범위에 따라, 데이터베이스 내 존재하는 모든 자료를 추출하여 교환하는 전체 동기화(Slow Sync)와, 이전 동기화 이후 변경된 자료만을 추출하여 교환하는 부분 동기화(Fast Sync) 타입으로 구분된다. 또한 자료 교환 방식에 따라, 클라이언트 또는 서버 중 한 쪽의 자료만을 상대편으로 전송하는 단방향 동기화(One-Way Sync)와, 클라이언트와 서버 양쪽 모두 자신의 자료를 전송하는 양방향 동기화(Two-Way Sync)로 구분된다. 현재 사용되는 대부분의 자료동기화 서버들이 앞에서 설명한 동기화 타입을 지원한다. 이 외에도, 단방향 동기화와 비슷하지만 한쪽에서 송신된 자료를 무조건 수용하는 리플래시(Refresh) 타입, 그리고 서버가 먼저 동기화 개시를 요청하는 서버 주도형(Server Alerted) 타입이 있다.

- 동기화 자료 전송 방식 : 클라이언트와 서버는 동기화를 위해 자신의 데이터베이스로부터 변경 사항을 탐지하여 추출된 자료를 정해진 동기화 타입에 따라 전송하게 되는데, 이 때 추출된 자료는 레코드 또는 필드 단위로 전송된다. 연락처 동기화를 예로 들면, '홍길동' 이라는 레코드에서 주소 필드가 변경되었을 때, 레코드 단위 전송에서는 '홍길동' 레코드 전체(이름, 전화번호, 주소, 이메일 등)가 전송되는 반면, 필드 단위 전송에서는 변경된 주소 필드만을 전송하는 방식이다. 액티브싱크, 핫싱크, SCTS, Synthesis, CPISync 서버 모두 레코드 단위 전송 방식을 사용한다. 동기화를 위해 전송해야 할 자료가 많을 때, 레코드 단위 전송 방식은 필드 단위 전송 방식에 비해 오버헤드가 높다. 따라서, 변경된 필드만을 전송하는 필드 단위 전송 방식이 적용되어야 한다.

- 자료 충돌 탐지 및 해결 정책 : 클라이언트와 서버가 자신의 데이터베이스에서 변경된 자료를 상호 교환할 때, 동일한 자료에 대해 클라이언트와 서버 양측에서 서로 다른 값으로 변경되는 경우가 발생할 수 있다. 이를 자료 충돌(Data Conflict)라 하며, 충돌 발생 시 자료동기화 서버는 사전에 정의된 충돌 해결 정책에 따라 충돌을 해결할 수 있어야 한다. 자료 충돌 해결 정책으로는, 서버의 자료를 우선시하는 서버 우선 정책, 클라이언트의 자료를 우선시하는 클라이언트 우선 정책, 최근에 변경된 자료를 우선시 하는 최신 자료 우선 정책, 양측의 변경 자료를 중복하여 저장하는 중복화 정책 등이

있다. 각 정책들의 장단점은 동기화 상황에 따라 달라질 수 있으므로 효율적인 자료 동기화를 위해서는 4개의 충돌 해결 정책을 모두 지원할 수 있어야 한다. 액티브싱크 서버와 Synthesis 서버는 앞에서 설명한 모든 충돌 해결 정책을 제공한다. 반면, 핫싱크는 클라이언트 우선 정책과 서버 우선 정책만을 제공하며, SCTS 서버는 최신 자료 우선 정책만을 제공한다. CPISync는 자료 충돌 해결 정책에 대해 명시하고 있지 않다.

앞에서 설명한 바와 같이, 이상적인 자료동기화 시스템은 자료의 변경 사항을 정확히 탐지하여 최소한의 자료만을 효율적으로 상호 교환하고, 동기화 시 발생할 수 있는 모든 자료 충돌을 탐지 및 해결할 수 있어야 하는데, 이 역할을 자료동기화 서버가 수행한다. 따라서, 자료동기화 서버의 설계 및 구현 방식에 따라 전체 자료동기화 시스템의 성능이 좌우된다.

본 논문에서는, 앞서 제시한 자료동기화 서버의 요구사항을 만족하는 자료동기화 서버를 제안한다. 제안된 자료동기화 서버는 OMA DS 프로토콜을 준수하며 변경사항 탐지를 위해 상태플래그 방식과 타임스탬프 방식을 사용한다. 또한, 앞서 설명한 모든 동기화 타입과 자료 충돌 해결 정책을 지원하며, 동기화 자료 전송 단위로는 레코드 및 필드 단위 전송을 모두 지원한다. 본 논문에서 제안된 자료동기화 서버를 포함한 다양한 자료동기화 서버들의 특징을 요약하면 표 1과 같다.

표 1. 다양한 동기화 서버들의 특징
Table 1. Characteristics of various data synchronization servers

서버 / 항목	액티브 싱크	핫 싱크	CPI Sync	SCTS	Synthesis	제안된 서버
사용 프로토콜	액티브 싱크	핫싱크	핫싱크	OMA DS	OMA DS	OMA DS
변경 사항 탐지	타임 스탬프	상태 플래그	CPI	타임 스탬프	타임 스탬프	타임스탬프 상태플래그
동기화 타입	모두 지원	모두 지원	모두 지원	모두 지원	모두 지원	모두 지원
자료 충돌 및 해결	모두 지원	서버우선 클라이언트우선	지원 안함	최신자료 우선	모두 지원	모두 지원
동기화 자료 전송 단위	레코드 단위	레코드 단위	레코드 단위	레코드 단위	레코드 단위	레코드단위 필드단위

2. SCTS 적합성 검증 도구

SyncML 적합성 검증 도구인 SCTS는 OMA DS 프로토콜 기반의 클라이언트 및 서버의 기능 적합성을 객관적으로 검증하기 위한 프로그램이다[9]. 뿐만 아니라 ,실제 OMA DS 서버로도 활용 가능한 프로그램이다. SCTS에는 OMA에서 제시한 검증 항목이 사전에 정의되어 있으며, 해당 항목에 따라 기능 적합성을 검증할 수 있다. OMA DS 기반 자료동기화 서버의 기능 적합성 검증이 중요한 이유는, 구현된 자료동기화 서버가 기존에 운용되고 있는 타 OMA DS 기반의 클라이언트와 상호 연동될 수 있음을 객관적으로 증명하기 위한 수단이기 때문이다. SCTS는 적합성 검증 대상에 따라 클라이언트 및 서버의 역할을 수행할 수 있다. 즉, 구현한 자료동기화 서버를 검증하고자 할 때는 SCTS를 클라이언트로 활용하면 된다. 그림 1은 SCTS의 사용자 인터페이스를 나타낸다.

그림 1에서 계정 메뉴는 SCTS를 클라이언트 또는 서버로 정의하기 위한 메뉴이며, 테스트 항목 및 설명 메뉴는 사전에 정의된 테스트 항목 및 세부 항목과 각 항목에 대한 설명을 나타낸다. 교환 패키지 설명 메뉴는 명령어, 반환된 상태코드 등, 각 테스트 항목에서 교환된 패키지의 세부 항목을 설명한다.

클라이언트를 검증할 경우 SCTS는 14개의 검증 항목을 제공하며, 서버를 검증할 경우 총 17개의 검증 항목을 제공한다

.

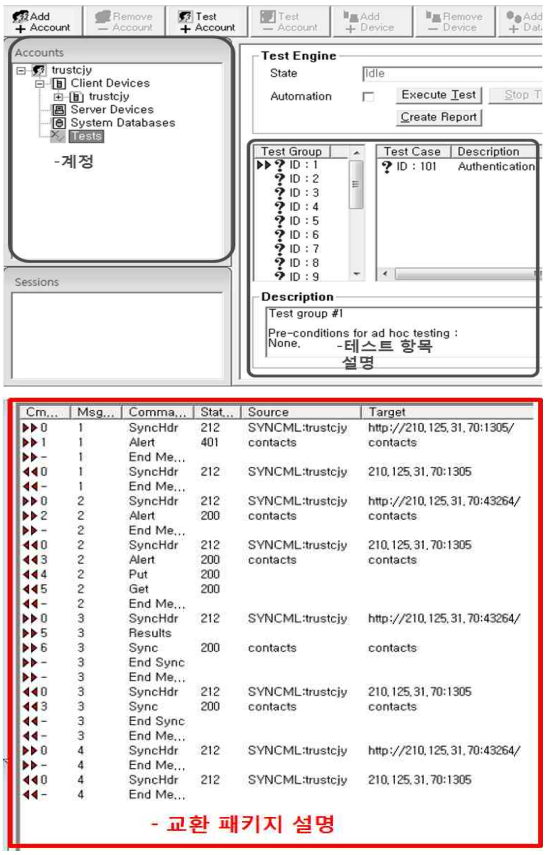


그림 1. SCTS 사용자 인터페이스
Fig. 1. SCTS User Interface

III. 자료동기화 서버

1. 자료동기화 서버 구조

본 연구에서 구축한 자료동기화 서버는 기능별로 프로토콜 매니저, 동기화 매니저, 그리고, 데이터베이스로 구성하였다[8]. 그림 2는 자료동기화 서버의 구조를 나타낸다.

- 프로토콜 매니저 : 동기화를 위한 전처리 과정을 수행하는 부분으로써, 통신 모듈, 세션 모듈, 인증 모듈로 구성하였다. 통신 모듈은 HTTP 전송 방식을 사용하여 클라이언트와 동기화 메시지를 교환한다. 세션 모듈은 클라이언트가 동기화를 요청할 때마다, 스레드를 생성하여 동기화를 위한 세션을 설정한다. 인증 모듈 동기화를 수행하기에 앞서 권한이 있는 클라이언트로부터 송신된 동기화 요청인지 체크한다. OMA DS에서 사용하는 BASIC과 MD5 해쉬 알고리즘을 모두 지원하도록 구현하였다.
- 동기화 매니저 : 동기화 엔진과 정책 모듈로 구성하였으며, 자료동기화 서버의 핵심 모듈이다. 동기화 엔진은 단말기로부터 수신한 동기화 메시지를 해석하여 동기화 명령 및 자료를 추출하는 메시지 해석 모듈과, 서버 측의 최신 자료 또는 단말기 측의 동기화 요청에 대한 응답 메시지를 생성하는 메시지 생성모듈로 구성하였다. 정책 모듈은 효율적인 자료 동기화를 위한 모듈로서, 변경 사항 탐지, 동기화 타입 결정, 충돌 탐지 및 충돌을 해결하는 역할을 한다. 본 논문의 자료동기화 서버는 변경 사항 및 충돌 탐지를 위해 각 자료의 상태플래그와 변경 시각 등의 정보를 사용하며, 이 정보들은 데이터베이스의 체인지 로그(Change Log)에 기록되도록 하였다. 동기화 타입 결정 모듈은 이전 동기화 상황, 클라이언트의 상태 등을 토대로, OMA DS에 명시된 전체 동기화, 부분 동기화, 단방향 동기화, 양방향 동기화, 리플래시 동기화, 서버 주도형 동기화 타입 중 한 가지 방식을 결정하도록 하였다. 또한 정책 모듈은 자료충돌을 해결하기 위해 서버 우선, 단말기 우선, 최신 자료 우선 및 중복화 정책 모두를 구현하였으며, 사전에 설정된 정책 중 하나를 선택하여 충돌을 해결하도록 하였다.
- 데이터베이스 : 동기화 대상 자료를 저장하고 있는 아이템 테이블, 자료의 변경 정보를 저장하고 있는 체인지 로그, 이전 동기화 정보를 나타내는 앵커(Anchor), 그리고 각 단말기의 인증 값이 저장된 인증 테이블로 구성하였다.

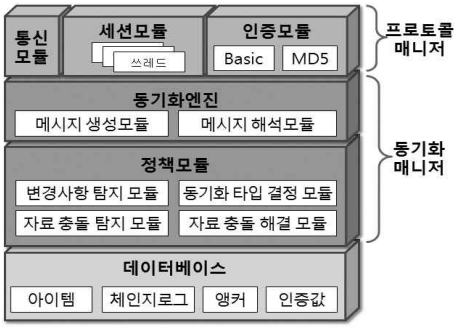


그림 2. 자료동기화 서버 구조
Fig. 2. The data synchronization server architecture

자료동기화 클라이언트와 자료동기화 서버는 자료동기화를 위해 일반적으로 6개의 패키지를 교환하도록 하였다. 클라이언트는 자신의 정보를 포함한 #1 패키지로 동기화를 요청한다. #1 패키지를 수신한 자료동기화 서버의 주된 역할은 단말기를 인증하고 동기화 타입을 결정하는 일이다. 그림 3은 자료동기화 서버가 클라이언트로부터 #1 패키지를 수신하고 #2 패키지를 생성하여 전송할 때까지의 처리 과정을 나타낸다.

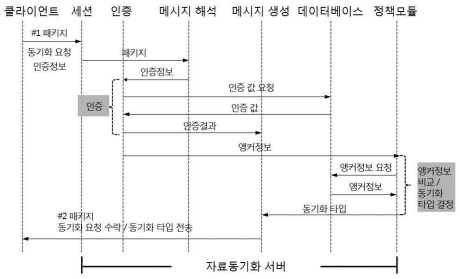


그림 3. 자료동기화 서버의 처리 과정 (#1 - #2 패키지)
Fig. 3. Process flow of the DS Server (#1 - #2 package)

동기화 요청이 수락되면, 클라이언트는 단말기 내 변경된 자료를 포함한 #3 패키지로 동기화를 수행하도록 하였다. #3 패키지를 수신한 자료동기화 서버는 #3 패키지로부터 동기화 명령과 자료를 추출한다. 또한, 체인지 로그를 참조하여 변경 사항을 탐지하고, 자료 충돌이 발생하지 않는지 탐지한다. 이러한 과정을 통해, 클라이언트 및 서버 양측의 자료 중 최신 자료를 선택하고 서버의 데이터베이스에 저장한다. 만약, 서버가 최신의 자료를 가지고 있을 경우에는, 최신 자료를 클라이언트에게 전달하기 위해 #4 패키지를 생성하여 전송한다. 그림 4는 자료동기화 서버가 클라이언트로부터 #3 패키지를 수신하고 #4 패키지를 생성하여 전송할 때까지의 처리 과정을 나타낸다.

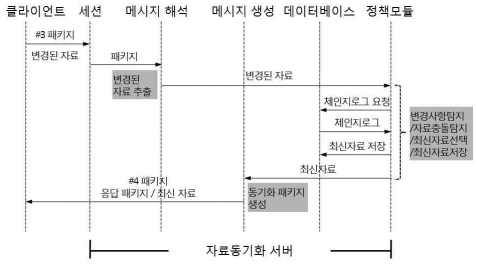


그림 4. 자료동기화 서버의 처리 과정 (#3 - #4 패키지)
Fig. 4. Process flow of the DS Server (#3 - #4 package)

#4 패키지가 최신 자료에 대한 추가 명령을 포함하고 있다면, 클라이언트는 해당 자료를 자신의 데이터베이스에 저장하고 저장 시 할당된 LUID (Local Unique ID)를 MAP 명령과 함께 #5 패키지로 전송하도록 하였다. 반면, GUID (Global Unique ID)를 사용하는 자료동기화 서버는 클라이언트로부터 추가된 자료에 대해서 자료의 ID를 전송할 필요가 없다[11]. #5 패키지를 수신한 자료동기화 서버는 해당 자료에 대한 GUID와 LUID를 매핑하고 결과를 #6 패키지로 전송한다. 그림 5는 자료동기화 서버가 #5 패키지를 수신하고 #6 패키지를 생성하여 전송할 때까지의 처리 과정을 나타낸다.

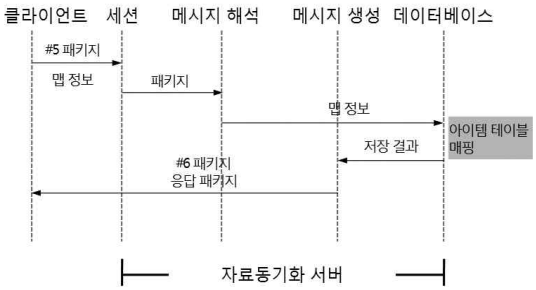


그림 5. 자료동기화 서버의 처리 과정 (#5 - #6 패키지)
Fig. 5. Process flow of the DS Server (#5 - #6 package)

2. 레코드 단위 및 필드단위 동기화 자료 전송

본 절에서는 기존의 자료동기화 서버들이 사용하는 레코드 단위 전송 방식과 본 논문에서 제안된 자료동기화 서버가 사용하는 필드단위 전송 방식에 대해 설명한다. 그림 6은 레코드 단위 전송과 필드 단위 전송의 차이점을 나타낸다.

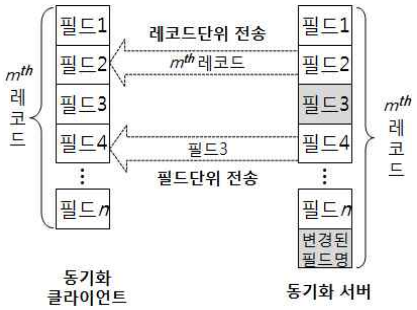


그림 6. 레코드 단위 전송과 필드 단위 전송의 차이
Fig. 6. Comparison between transmission-by record and transmission-by field

자료동기화를 위해 서버는 클라이언트에게 동기화 자료를 전송하는데, 기존의 자료동기화 서버들은 동기화 대상 자료를 레코드 단위로 전송한다. 그림 6에서와 같이, 레코드단위 전송에서는 서버의 m 번째 레코드의 필드 3만이 변경되었더라도 필드 1부터 필드 n까지의 모든 필드를 전송한다. 따라서, 동기화 자료 전송 시 네트워크상 오버헤드가 발생하게 된다. 뿐만 아니라, 클라이언트의 입장에서는 서버로부터 수신한 레코드가 포함한 모든 필드를 자신의 데이터베이스에 삽입해야 하기 때문에 비효율적이다. 반면, 필드 단위 전송 방식은 m 번째 레코드에서 변경된 필드 3만을 클라이언트에게 전송하므로, 네트워크상의 오버헤드가 줄어들며, 클라이언트의 입장에서도 서버로부터 수신한 필드 3만을 데이터베이스에 삽입하면 되기 때문에 효율적이다. 연락처 또는 일정 등과 같이 모바일 단말기에서 주로 사용되는 동기화 대상 자료들은 하나의 레코드가 이름, 전화번호, 주소 또는 시각, 장소 등 다수의 필드를 포함하고 있으며, 일반적으로 한 번에 변경되는 필드의 수가 적으므로 필드 단위 전송 방식이 레코드 단위 전송 방식에 비해 효율적이다.

본 논문의 자료동기화 서버는 이러한 기존의 레코드 단위 전송의 단점을 보완하기 위해 필드 단위 전송 방식을 추가로 설계 및 구현하였다. 이를 위해, 서버에 저장된 각 레코드들은 변경된 필드명을 명시하기 위한 필드인 'Modified' 필드를 추가적으로 가지게 된다. 그림 2의 변경사항 탐지 모듈은 변경 사항을 탐지하기 위해 데이터베이스 내 존재하는 모든 레코드의 상태 플래그를 체크한다. 만약 특정 레코드의 상태 플래그가 변경되었음을 뜻하는 'M'으로 설정되어 있다면, 해당 레코드의 'Modified' 필드에 명시된 필드만을 추출한다.

3. 자료동기화 서버 구현 결과

본 논문에서 제안된 자료동기화 서버는 C#으로 구현되었으며, 1기가바이트의 메인 메모리와 인텔 코어2 듀오 (2.66 GHz)의 프로세서를 사용하는 윈도우즈 서버 2003 기반의 데스크톱 컴퓨터에 설치되었다. 서버 환경을 구축하기 위해 아파치 서버를 사용하였으며, 데이터베이스는 MySQL 5.1 버전을 사용하였다. 그림 7은 자료동기화 서버의 실행화면을 나타낸다. 자료동기화 서버 실행화면에서 좌측은 충돌해결 정책, 포트번호를 설정하기 위한 메뉴와 서버의 상태 및 생성된 스레드의 정보를 나타낸다. 우측 창은 동기화 세션, 동기화 메시지 처리 및 생성 상황을 나타내는 정보 창이다.

그림 7. 자료동기화 서버 실행화면
Fig. 7. A screen shot of the DS Server

Ⅳ. 자료동기화 서버 적합성 검증

1. 기능 적합성

본 논문에서 구축한 자료동기화 서버의 기능 적합성을 검증하기 위해 OMA에서 제공하는 적합성 검증 도구인 SCTS[9]를 사용하였다. SCTS는 OMA 국제 인증을 획득하기 위해 거쳐야 하는 필수적인 검증 도구로서, SCTS에서 제시한 검증 항목을 모두 통과하여야 OMA 테스트페스트에 참가할 수 있게 되고, OMA 테스트페스트 [10]에서 OMA DS 클라이언트와의 연동성 테스트를 통과하면 OMA 국제 인증을 획득할 수 있다. 즉, 구축한 자료동기화 서버의 기능 적합성을 SCTS를 통해 객관적으로 1차적인 검증을 할 수 있다. SCTS는 서버가 OMA DS에서 명시한 명령어와 동기화 방식, 멀티플 아이템, 멀티플 메시지, 변경 자료의 수를 알려주는 NumberOfChanges와 대용량 오브젝트를 분할하여 전송하는 Large Object 기능의 지원 여부를 검증하기 위해서, 총 17개의 검증 항목을 제공한다. 표 1은 SCTS에서 제공하는 각 검증 항목 및 세부 항목에 대해 도표화한 것이며, 그림 8은 표 2의 항목에 따라 본 논문의 자료동기화 서버를 검증한 결과를 나타낸다.

표 2. SCTS 검증 항목
Table 2. SCTS Conformance Test Items

항목 번호	세부 항목	설 명
1	101	SCTS의 잘못된 인증 정보에 대한 서버의 처리
2	201	SCTS의 누락된 인증 정보에 대한 서버의 처리
	202	SCTS의 올바른 인증 정보에 대한 서버의 처리
	203	SCTS의 GET 명령에 대한 서버의 처리
	204	서버의 Alert 명령 지원 여부
	205	서버의 Sync 명령 지원 여부
3	301	서버의 Anchor 명령 지원 여부
	302	SCTS의 ADD 명령에 대한 서버의 처리
4	401	서버의 ADD 명령 지원 여부
	402	SCTS의 REPLACE 명령에 대한 서버의 처리(SCTS가 추가한 아이템 대상)
	403	서버의 REPLACE 명령 지원 여부 (SCTS가 추가한 아이템 대상)
	404	SCTS의 DELETE 명령에 대한 서버의 처리(SCTS가 추가한 아이템 대상)
	405	서버의 DELETE 명령 지원 여부 (SCTS가 추가한 아이템 대상)
5	501	SCTS의 REPLACE 명령에 대한 서버의 처리(서버가 추가한 아이템 대상)
	502	서버의 REPLACE 명령 지원 여부 (서버가 추가한 아이템 대상)
	503	SCTS의 DELETE 명령에 대한 서버의 처리(서버가 추가한 아이템 대상)
	504	서버의 DELETE 명령 지원 여부 (서버가 추가한 아이템 대상)
	505	존재하지 않는 아이템에 대한 SCTS의 REPLACE 명령을 ADD 명령처럼 처리하는지 여부
6	601	서버의 전체 동기화(Slow Sync) 모드 지원 여부
7	701	존재하지 않는 아이템에 대한 SCTS DELETE 명령 처리
8	801	초기화 과정과 동기화 과정을 하나의 패키지에 통합해 전송하는 SCTS의 요청 지원 여부
9	901	멀티플 아이템에 대한 ADD 명령 지원 여부
	902	멀티플 아이템에 대한 REPLACE 명령 지원 여부

	903	멀티플 아이템에 대한 DELETE 명령 지원 여부
	904	SCTS의 REPLACE 명령들에 대한 서버의 개별적인 STATUS 코드 전송 지원 여부
	905	SCTS의 DELETE 명령들에 대한 서버의 개별적인 STATUS 코드 전송 지원 여부
10	1001	서버의 멀티플 메시지 지원 여부
11	1101	서버의 NumberOfChanges 지원 여부
12	1201	SCTS가 NumberOfChanges를 지원하지 않을 때 서버의 처리
13	1301	서버의 Large Object 지원 여부
14	1401	SCTS가 Large Object를 지원하지 않을 때 서버의 처리
15	1501	서버의 MaxObjSize 지원 여부
16	1601	잘못된 크기의 Large Object에 대한 서버의 처리
	1602	전송이 미완료된 Large Object에 대한 서버의 처리
17	1701	이전 세션에서 전송이 미완료된 Large Object에 대한 서버의 처리

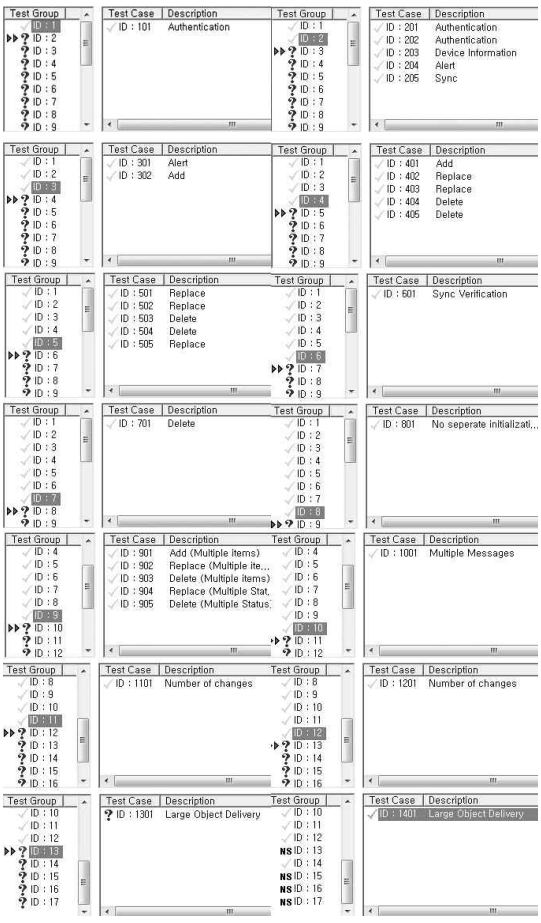


그림 8. 자료동기화 서버 기능 적합성 검증 결과
Fig. 8. Functional validation results of the DS Server

그림 8에서 항목 및 세부항목 아이디 앞에 표시된 체크 모양의 아이콘은 해당 항목의 검증을 성공적으로 수행하였음을 나타낸다. 본 논문에서 구축한 자료 동기화 서버는 총 17개의 검증 항목 중, 대용량 오브젝트 기능 (항목 14-17번)을 제외한 모든 항목을 만족함을 알 수 있다. 즉, 대용량 오브젝트 기능을 제외하고는, OMA에서 제시하는 기능 적합성을 만족하고 OMA DS 프로토콜을 준수함을 뜻한다. 대용량 오브젝트 기능의 경우, 대용량의 오브젝트를 복수개의 메시지로 나눠서 전송하는 기능이다[11]. 본 논문의 자료동기화 서버는 우선적으로 동기화 대상 자료 중 모바일 단말기 내에서 가장 많이 활용된다고 판단되는 개인정보관리 시스템 (Personal Information Management System)의 자료 (예: 연락처, 일정, 이메일 등)를 대상으로 하고 있고, 이러한 종류의 자료들은 단위 자료의 용량이 크지 않기 때문에, 현재 대용량 오브젝트 기능을 구현하지 않았으며, 향후 구현할 예정이다.

2. 성능 적합성

본 논문에서 구축한 자료동기화 서버는 클라이언트로부터 수신한 자료와 서버 내 저장된 자료를 비교하여 최신의 자료를 선택하고 클라이언트에게 전송한다. 따라서, 서버는 클라이언트의 동기화 요청을 신속하게 처리할 수 있어야 하며, 동기화 자료의 수가 증가하더라도 소요시간이 완만하게 증가하여야 한다. 구축한 자료동기화 서버의 성능 적합성을 검증하기 위해 본 연구팀이 앞서 개발한 자료동기화 게이트웨이[6]와 연동하여 연락처 동기화 수행 중 소요되는 시간을 측정하였다.

그림 9는 두 가지 상황 (클라이언트 측에 추가된 자료를 동기화할 때와 서버 측에 추가된 자료를 동기화)에서의 동기화에 따른 서버의 소요 시간을 그래프로 나타내고 있다. 자료의 수는 20개에서 200개까지 20개씩 증가시켰다. 그림 9의 그래프에 따르면, 각 상황에서 동기화 시, 서버의 소요시간은 자료 수의 증가에 따라 평균 25.5% 와 35.3% 증가하였다. 이를 통해, 동기화 자료 수의 증가에 따라 동기화에 소요되는 시간이 완만하게 증가함을 알 수 있다.

하지만, 동기화 시 클라이언트가 최신 자료를 가지고 있을 때 보다, 서버가 최신 자료를 가지고 있을 때의 동기화 시간이 평균 2.1배 빠른 것을 알 수 있다. 이는 자료동기화 서버의 입장에서 데이터베이스로부터 추가된 자료를 읽어올 때 보다 새로운 자료를 삽입(insert)할 때 더 많은 시간이 소요되기 때문이다. 이는 성능 향상을 위해 자료동기화 서버의 데이터베이스 삽입 동작이 개선되어야 함을 의미한다.

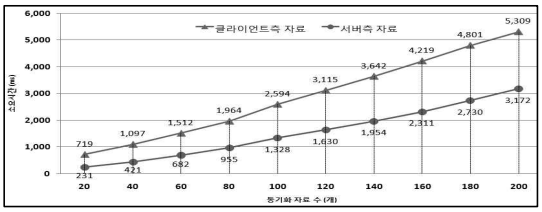


그림 9. 자료동기화 서버의 동기화 소요시간
(자료 수 20-200개)
Fig. 9. Synchronization times of the DS Server with respect to the number of data (items: 20-200)

다음으로 3.1절의 그림 3~5에서 설명한 자료동기화 서버의 처리 과정에 따른 소요시간을 패키지별로 측정하였다. 그림 10은 자료동기화 서버가 각 패키지를 처리 및 생성하는데 소요되는 시간을 나타낸 그래프이다. 단, 서버가 #1 패키지를 수신하여 해석한 후, #2 패키지를 생성하여 전송할 때까지의 과정은, 동기화 자료의 수와 관계없이 거의 일정하고 소요 시간 또한 평균 36ms 으로 아주 작아 그래프에서 생략하였다. 또한, 서버가 최신 자료를 가지고 있는 상황에서의 동기화 소요 시간을 측정하였다.

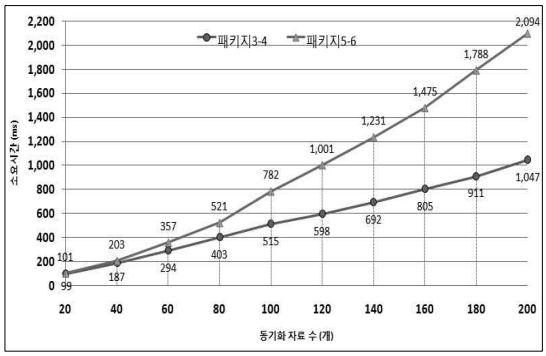


그림 10. 패키지별 자료동기화 서버의 동기화 소요시간
(자료 수 : 20-200개)
Fig. 10. Synchronization times of the DS Server for packages (items: 20-200)

그림 10의 그래프에 따르면, #3 패키지를 처리하고 #4 패키지를 생성하는데 소요되는 시간보다 #5 패키지를 처리하고 #6 패키지를 생성하는데 소요되는 시간이 평균 2.1배 높음을 알 수 있다. 이는 OMA DS의 아이디 매핑 방식에 기인한다. OMA DS에 따르면, 클라이언트와 서버는 각 자료에 대해 LUID와 GUID라는 서로 다른 값의 아이디를 할당하고 있으며, 서버는 클라이언트의 LUID와 자신의 GUID를 매핑하여야 하기 때문이다[11]. 즉, 자료의 수가 증가함에 따라, #5 패키지에서 수신된 클라이언트의 LUID를 서버의 데이터베이스에서 GUID와 매핑하는데 소요되는 시간이 증가한다. 따라서, OMA DS의 아이디 매핑 과정의 효율성을 증대시키기 위한 추가적인 연구가 필요할 것으로 판단된다.

자료동기화 시스템은 소수의 자료동기화 서버와 다수의 자료동기화 클라이언트로 구성된다. 따라서, 하나의 자료동기화 서버가 동시에 여러 클라이언트로부터 동기화 요청을 받을 수 있으며, 이 경우에도 병목 현상을 발생시키지 않고 원활히 처리할 수 있어야 한다. 이를 검증하기 위해, 다수의 모바일 단말기를 구매하여 소요시간을 측정하기에는 현실적으로 어려움이 따르므로, 앞서 개발한 자료동기화 게이트웨이를 윈도우 모바일 기반의 에뮬레이터에 탑재하여 1대에서 10대까지 증가시키되 동기화 자료 수는 200개로 고정하여 소요되는 시간을 측정하였다. 그림 11은 동기화에 따른 소요 시간을 그래프로 나타내고 있다.

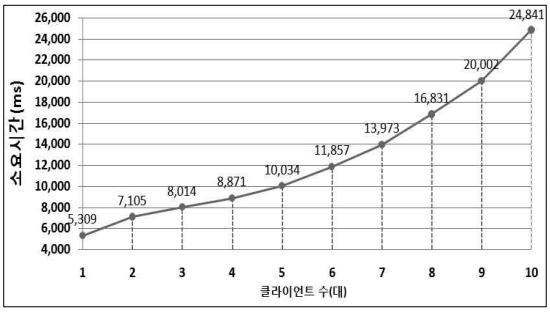


그림 11. 자료동기화 서버의 동기화 소요시간
(클라이언트: 1-10대, 자료 수: 200개)
Fig. 11. Synchronization times of DS Server with respect to the numbers of servers and clients (Clients: 1-10, items: 200)

그림 11의 그래프에 따르면, 1대의 클라이언트와 200개의 자료를 동기화하는데 서버에서 소요된 시간은 5,309ms이며, 이보다 10배 증가한 10대의 단말기와 200개의 자료를 동기화하는데 서버에서 소요된 시간은 24,841ms이었다. 클라이언트 수의 증가에 따른 평균 소요시간 증가율은 19%로서, 클라이언트 수의 증가에 따른 심각한 병목 현상이 나타나지 않음을 알 수 있다.

다음으로, 본 논문에서 제안된 필드 단위 전송 방식의 성능을 평가하기 위해, 필드 단위 전송 방식과 레코드 단위 전송 방식의 성능을 비교하였다. 서버는 필드 단위 전송을 위해 데이터베이스에 변경된 레코드가 존재하는지 1차적으로 체크한 후, 존재한다면 해당 레코드의 각 필드 중 어느 필드가 변경되었는지를 2차적으로 체크한다. 따라서, 레코드 단위 전송 방식에 비해 필드 변경을 체크하는 단계가 추가적으로 필요하

다. 하지만, 서버로 사용되는 컴퓨터들은 컴퓨팅 파워가 우수하기 때문에 실제 필드단위 자료 추출을 위해 소요되는 시간은 레코드 단위 자료 추출과 비교해 거의 동일하였다.

하지만, 서버가 자료를 전송하였을 때 서버에 비해 제한된 컴퓨팅 파워을 가지는 클라이언트가 이를 처리하기 위해 소요하는 시간은 두 방식에 따라 큰 차이를 보였다. 클라이언트는 필드단위로 전송받은 자료의 경우 해당 필드만을 데이터베이스에 업데이트하면 되지만, 레코드 단위로 전송받은 자료의 경우에는 해당 레코드를 구성하는 모든 필드를 데이터베이스에 업데이트해야 하기 때문이다. 따라서, 필드단위 전송 방식은 자료 추출을 위해 서버에서 소요하는 시간은 거의 동일한 반면, 자료가 클라이언트로 전송되었을 때 클라이언트의 처리에 소요되는 시간은 크게 줄일 수 있다는 장점이 있다. 즉, 필드 단위 전송 방식은 서버 자체의 성능 향상 보다는, 전체 동기화 시스템의 성능을 크게 향상시킬 수 있다. 이를 증명하기 위해, 다수의 필드를 가지는 연락처 자료를 자료동기화 게이트웨이와 동기화 하였는데, 그림 12는 필드 단위와 레코드 단위 전송 방식에서의 각각 소요된 전체 동기화 시간을 나타낸다.

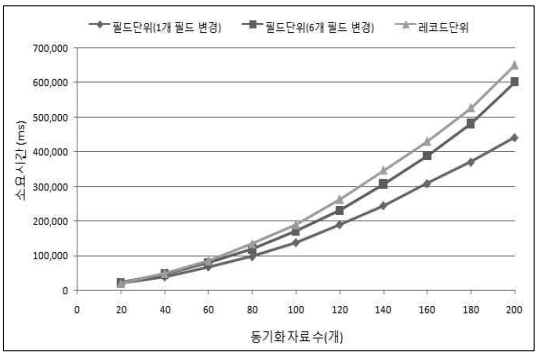


그림 12. 필드 단위 전송 방식과 레코드 단위 전송 방식의 성능 비교(자료 수:20-200개, 변경 필드 수:1개, 6개)
Fig. 12. Performance comparison between transmission-by records and transmission-by fields(items:20-200, modified fields:1-6)

모바일 단말기 내에서 사용되는 연락처 애플리케이션은 일반적으로 vCard 포맷을 따르며, 하나의 자료(레코드)는 총 13개의 필드를 가진다. 따라서, 레코드 단위 전송 방식에서 동기화 서버는 클라이언트에게 13개의 필드를 전송한다. 필드 단위 전송 방식에서는 1개의 필드만 수정되었을 때와 6개의 필드가 수정되었을 때를 가정하여 전체 동기화 소요시간을 측정하였다. 그림 12에 따르면, 동기화 자료 수가 20개이며 1개의 필드가 변경되었을 때는 21,475ms 소요되고, 6개의 필드가 변경되었을 때는 23,154ms 소요되며, 레코드 단위 전송 방식에서는 변경된 필드의 수와 관계없이 20,500ms 소요되었음을 알 수 있다. 즉, 동기화 자료 수가 20개 이하일 때는 레코드 단위 전송 방식이 필드 단위 전송 방식에 비해 더 우수함을 알 수 있다. 하지만, 동기화 자료 수 가 40개 이상일 때, 필드 단위 전송 방식에서 1개의 필드가 변경되었다고 가정했을 때는 40.096ms 소요되었으며, 6개의 필드가 변경되었다고 가정했을 때는 47.375ms 소요되는 반면, 레코드 단위 전송 방식에서는 49.576ms 소요되어 필드 단위 전송 방식이 레코드 단위 전송 방식에 비해 우수함을 알 수 있었다.

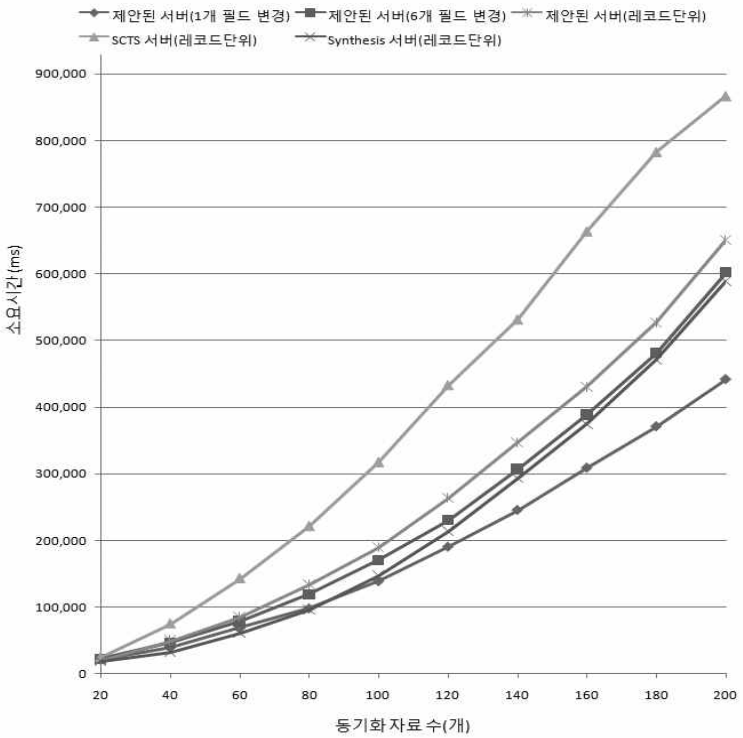
마지막으로, 제안된 동기화 서버와 기존의 타 자료동기화 서버들과의 성능을 비교하였다. 기존의 자료동기화 서버들 중, 액티브싱크, 핫싱크 및 CPI싱크는 사용하는 자료동기화 프로토콜이 제안된 서버와 다르며, 제공하는 기능 또한 다르기 때문에 성능 비교 대상에서 제외하였으며, 제안된 서버와 동일한 프로토콜을 사용하는 OMA DS 기반의 SCTS 서버, Synthesis 서버와 성능 비교를 수행하였다. SCTS 서버와 Synthesis 서버는 레코드 단위 전송 방식을 사용하며, 본 논문의 자료동기화 서버는 필드단위 전송 방식을 사용한다. 비교 결과는 그림 13의 그래프와 같다.

그림 13의 실험에서는, 자료동기화 서버가 동기화 자료를 20개에서 200개까지 20개씩 증가하며 클라이언트에게 전송할 때 소요되는 전체 자료동기화 시간을 측정하였다. 그림 13에 따르면, Synthesis 서버는 동기화 자료의 수가 80개 이하일 때, 본 논문에서 제안된 동기화 서버보다 성능이 우수함을 알 수 있다. 하지만, 동기화 자료의 수가 100개 이상이 되면, 필드 단위 전송 방식(1개의 필드가 변경되었다고 가정)을 사용한 본 논문의 서버가 더 우수함을 알 수 있으며, 동기화 자료의 수가 증가할수록, 그 차이는 더욱 커진다. 필드 6개가 변경되었다고 가정한 경우, Synthesis 서버는 제안된 서버보다 성능이 우수하지만, 동기화 자료의 수가 증가할수록 그 차이는 점차 좁혀진다. 이를 통해, 본 논문에서 제안한 자료동기화 서버는 SCTS 서버에 비해 성능이 대체로 우수하며, Synthesis 서버에 비해서는 동기화 소요시간이 더 요구되지만, 변경된 필드의 수가 적을수록, 그리고 동기화할 자료의 수가 많은 수록 유리함을 알 수 있다. 일반적으로, 연락처, 일정과 같은 애플리케이션에서 사용되는 자료들은 다수의 필드들을 가지고 있으며, 한 번에 변경되는 필드의 수가 많지 않기 때문에, 이러한 개인정보관리 시스템 (PIMS)의 자료를 동기화하는데 있어서는 본 논문에서 제안한 자료동기화 서버가 더 적합함을 알 수 있다.

본 논문에서는 자료동기화 서버의 구축 결과를 제시하며, 기능 적합성을 검증하기 위해 OMA에서 제공하는 적합성 검

증 도구인 SCTS와의 동기화를 수행하였다. 검증 결과, 대용량 오브젝트 기능을 제외한 모든 항목을 만족함을 알 수 있었다. 또한, 자료동기화 서버의 성능 적합성을 검증하기 위해 동기화 자료의 수와 클라이언트의수를 증가시키며 동기화에 소요되는 시간을 측정하였다. 측정 결과, 본 논문의 자료동기화 서버는 동기화 자료 및 클라이언트 수의 증가에 따른 소요시간의 증가가 완만함을 검증할 수 있었다. 또한, 측정 결과를 통해 자료동기화 서버의 데이터베이스 삽입 과정과 OMA DS의 아이디 매핑 과정의 효율성을 높이기 위한 추가적인 연구가 필요함을 알 수 있었다. 뿐만 아니라, 제안된 동기화 서버와 동일한 프로토콜을 사용하는 기존의 서버들과 성능 비교를 한 결과, 제안된 서버는 SCTS 서버보다 성능이 대체로 우수하며, Synthesis 서버에 비해 동기화 자료의 수가 증가할수록, 그리고 변경된 필드의 수가 적을수록 성능이 좋아짐을 알 수 있었다.

본 논문에서 구축한 자료동기화 서버는 자체로 제작하여 소스코드를 가지고 있기 때문에, 다양한 모바일 자료동기화 방식을 연구하는데 기본 프레임으로서 사용될 수 있을 것이다. 향후, 대용량 오브젝트 기능을 추가 구현할 예정이며, 다수의 클라이언트를 대상으로 한 성능 평가를 위해 시뮬레이션 모델을 구축하여 자료동기화 서버의 성능을 분석할 예정이다. 또한, 본 논문에서 도출된 개선사항을 토대로 전체 자료동기화 시스템의 성능을 향상시킬 수 있는 연구를 수행할 예정이다.



구 분	20	40	60	80	100	120	140	160	180	200
제안된 서버 필드단위(1개 필드 변경)	21,475	40,096	69,317	98,589	139,246	191,041	246,107	309,445	371,441	442,178
제안된 서버 필드단위(6개 필드 변경)	23,154	47,375	79,412	119,873	170,714	230,741	307,177	388,471	481,347	601,743
레코드단위	20,500	49,576	85,323	134,228	189,533	263,475	347,512	430,791	527,413	651,247
SCTS 서버	24,741	75,131	143,171	221,461	317,414	433,145	531,473	664,131	783,144	867,413
Synthesis 서버	18,761	32,411	61,213	95,412	147,122	214,122	294,131	374,164	472,144	589,141

그림 13 자료동기화 소요시간 비교(자료 수 : 20-200개, 변경 필드 수 : 1개, 6개)
Fig. 13. Comparison of data synchronization times between various DS servers (items:20-200, modified fields:1-6)

참고문헌

[1] ActiveSync, http://www.microsoft.com/windowsmobile

[2] HotSync, Introduction to Coduit Development http://www.accessdevnet.com/docs/conduits/win

[3] OMA DS, http://openmobilealliance.org

[4] djjang et al., "Data-Synchronization Gateway based on SyncML for an ActiveSync Data Transformation", Journal of Korean Society for Internet Information, Vol. 7, No. 3, pp. 61-69, Jun. 2006.

[5] djjang and khpark, "A Gateway System for Integrated Data Synchronization Based on SyncML for Interoperability Between Various Mobile Terminals", Communications of the Korean Institute of Information Scientists and Engineers, Vol. 14, No. 2, pp. 117-129, Apr. 2008.

[6] jgpak et al., "Construction of Embedded Data Synchronization Gateway", Journal of the Korea Institute of Maritime Information & Communication Sciences, Vol. 14, No. 2, pp. 335-342, Feb. 2010.

[7] Synthesis Server, Synthesis AG, http://www.synthesis.ch

[8] jgpak et al., "Design of a Data Synchronization Server for Mobile Communication Environments", The Journal of Korean Institute of Information Technology, Vol. 8, No. 1, pp. 17-26, Feb. 2010

[9] SCTS, SCTS Ver 1.1.2, http://sourceforge.net/projects/oma-scts

[10] OMA TestFest, http://www.openmobilealliance.org/TestFests

[11] OMA Data Synchronization Protocol Spec. Ver.1.2.2, http://www.openmobilealliance.org, 2009.

[12] Trachtenberg, A. et al., "Fast PDA Synchronization Using Characteristic Polynomial Interpolation", Proc. Annual Joint Conference of the IEEE Computer and Communications Societies, New York, pp. 1510-1519, Nov. 2002.

[13] Starobinski, D. et al., "Efficient PDA Synchronization", IEEE Transactions on Mobile Computing, Vol. 2, No. 1, pp. 40-51, Jan. 2003.

[14] Fan, L. et al., "Summary Cache: A Scalable Wide-Area Web Cache Sharing Protocol", Proc. ACM SIGCOMM '98 conference on Applications, technologies, architectures, and protocols for computer communication, pp. 281-293, Aug. 1998.

저 자 소 개

박 주 건

2006: 계명대학교 컴퓨터공학과 공학사.

2008: 계명대학교 컴퓨터공학과 공학석사.

현 재: 계명대학교 컴퓨터공학과 박사과정

관심분야: 유비쿼터스 컴퓨팅, 임베디드 소프트웨어

Email : corea@kmu.ac.kr

박 기 현

1979: 경북대학교 전자계산학과 공학사.

1981: 한국과학기술원 전자계산학과 공학석사.

2004: 미국 Vanderbilt 대학교 전자계산학과 공학박사

현 재: 계명대학교 컴퓨터공학과 교수

관심분야: 병렬처리, 모바일 소프트웨어, 임베디드 소프트웨어, 운영체제

Email : khp@kmu.ac.kr