SyncML 자료 동기화 클라이언트를 위한 세션 핸들러 모듈의 설계 및 구현

하 병 훈[†] · 박 기 현[††] · 주 홍 택[†††] · 우 종 정[††††]

요 약

SyncML은 OMA(Open Mobile Alliance)에 의해 제안된 개방적 표준 자료 동기화 프로토콜이다. 본 논문에서는 SyncML 자료 동기화 클라이언트 개발을 위해서 필요한 세션 핸들러(Session Handler) 모듈과 사용자 환경 설정 프로그램을 설계하고 구현하였다. 세션 핸들러는 통신 세션을 제어하고 교환되는 메시지의 헤더 부분을 생성하고 적합성을 판단하는 모듈로서, SyncML 자료 동기화 시스템에서 주요한 역할을 담당한다. 본 논문에서 구현한 세션 핸들러 모듈과 사용자 환경 설정 프로그램의 정상적인 동작여부를 검증하기 위해 리눅스를 운영체제로 하는 자우루스 PDA에 포팅한 후, 무선인터넷 국제표준화 포럼인 OMA의 인증을 받은 Synthesis 서버와 연동하여 자료 동기화 작업을 수행하였다.

키워드 : 자료 동기화, 동기화 마크업 언어, 세션 핸들러, 이동 컴퓨팅, OMA

Design and Implementation of a Session Handler Module for SyncML Data Synchronization Clients

Byoung-Hoon Ha[†] · KeeHyun Park[††] · Ju HongTaek[†††] · Jongjung Woo[††††]

ABSTRACT

SyncML is an open standard data synchronization protocol proposed by OMA(Open Mobile Alliance). In this paper, a Session Handler module, one of major modules for developing SyncML data synchronization clients, and a client User Setup program are designed and implemented. The Session Handler Module controls communication sessions, generates header parts of messages exchanged, and determines the legitimacy of incoming messages. In order to justify normal operations of the Session Handler module and the client User Setup program implemented in this paper, they are ported to a Zaurus PDA, which runs on LINUX operating system. In addition, data synchronization operations are performed between the PDA and a Synthesis sever, whose SyncML data synchronization operation is certificated by OMA, Wireless Internet International Standard Forum.

Key Words : Data Synchronization, SyncML, Session Handler, Mobile Computing, OMA

1. 서 론

노트북이나 휴대폰, PDA 등과 같은 이동무선통신 단말기를 통해 인터넷 서비스를 받고자 하는 요구가 늘어나면서 이를 지원하기 위한 다양한 단말기와 접속 기술이 개발되고 있다. 또한, 이러한 무선 인터넷 서비스에 대한 사용자들의 이용 영역이 일정관리, 전자메일 교환과 같은 개인적인 영역에서 기업의 업무 효율 증진을 위한 비즈니스 영역으로 확대되고 있으며, 한 개인이 다수의 단말기를 보유하고 있는 사용자의 수도 늘어나고 있다[1, 2].

하지만, 이동무선통신 환경의 네트워크 서비스 범위의 한계 때문에 각각의 단말기들은 통합관리 서버와 항상 연결 상태를 유지할 수 없으며, 이로 인해 연결 혹은 연결해제 작업 동안에도 자료가 변경될 수 있다. 또한, 한 개인이 다수의 단말기를 사용함으로써 동일한 자료를 여러 단말기에 분산되어 저장할 수 있다. 이러한 연결 상태 변화 기간 동안에 갱신된 자료 혹은 동일한 자료에 대해 분산 저장된 자료에 대해서 자료의 일치성을 보장하기 위하여 자료 동기화(Data Synchronization) 과정이 필요하다[1].

현재 각 단말기 회사에서 제공되는 다양한 자료 동기화 방식이 존재하지만, 각각의 제조회사 장치들 간의 상호 운용성을 보장할 수 없으므로, 단말기 사용자나 단말기 제조회사, 응용 소프트웨어 개발자, 그리고 서비스 제공자 모두에게 많

※ 본 연구는 산업자원부의 지역혁신 인력양성사업의 연구결과로 수행되었음.
† 준 회 원 : 계명대학교 일반대학원 컴퓨터공학과
†† 종신회원 : 계명대학교 정보통신학부 교수
††† 정 회 원 : 계명대학교 정보통신학부 교수
†††† 종신회원 : 성신여자대학교 컴퓨터정보학부 교수
논문접수 : 2005년 6월 14일, 심사완료 : 2005년 9월 9일

은 문제점들을 가지고 있다. 이러한 문제점을 해결하기 위해서 2000년 2월 OMA(Open Mobile Alliance)에서 SyncML을 제안하여 자료 동기화 방법에 대한 공개적인 표준화를 주도하고 있다[1, 4, 5].

본 논문에서는 SyncML 자료 동기화 클라이언트 개발을 위해서 통신 세션을 제어하고 교환되는 자료 동기화 메시지의 헤더 부분을 생성하고, 생성된 메시지의 적합성을 판단하는 역할을 담당하는 세션 핸들러(Session Handler) 모듈을 설계 및 구현하였다. 또한, Qt 라이브러리[3]를 이용하여 세션 핸들러가 포함된 전체 SyncML 동기화 클라이언트를 위한 GUI 기반의 사용자 환경 설정 프로그램을 구현하였다. 본 논문에서 구현한 세션 핸들러 모듈과 사용자 환경 설정 프로그램의 정상적인 동작여부를 검증하기 위해 리눅스를 운영체제로 하는 자우루스 PDA에 포팅한 후, 무선인터넷 국제표준화 포럼인 OMA의 인증을 받은 Synthesis 서버와의 연동하여 자료 동기화 작업을 수행하였다.

본 논문의 구성은 다음과 같다. 2절에서는 관련 연구 부분으로서, SyncML에 대한 소개 및 규격, 기존의 자료 동기화 클라이언트 개발 및 연구 사례에 대해서 분석하고, 3절에서는 SynML 자료 동기화 클라이언트를 위한 세션 핸들러 모듈과 사용자 설정 프로그램의 설계에 대해서 설명한다. 4절에서는 구현된 세션 핸들러 모듈이 포함된 자료 동기화 클라이언트를 자우루스(Zaurus) PDA에 포팅하고, 이를 Qt 라이브러리로 구현된 사용자 설정 화면을 통해 자료 동기화 작업이 수행되는 실험 결과를 제시한다. 마지막으로, 5절에서는 결론을 제시하고 향후 연구 방향에 대해서 논의한다.

2. 관련 연구

2.1 HotSync 프로토콜

HotSync는 Palm사에서 Palm OS에서 동작하도록 만든 자료 동기화 방법이며, Palm OS를 탑재한 PDA 등에서 주로 사용된다. HotSync 매니저는 데스크탑 컴퓨터와 디바이스 사이의 데이터를 유지하거나, 데스크탑에서 디바이스의 데이터를 백업 받거나, 디바이스에서 데스크탑으로 데이터를 다운받는 동안 Conduit라 불리는 코드모듈의 기본 단위를 호출한다.

HotSync가 동작하는 동안 동기화 과정을 제어하는 HotSync 매니저, 디바이스 어플리케이션과 데tm크탑 데이터 간의 실질적인 통신을 조작하는 모듈인 Conduit, 데이터의 손실이나 중복 혹은 잘못된 데이터가 발생되면 데스크탑 어플리케이션에 알려주는 Notifier DLL, 디바이스에서 사용되는 Handheld 어플리케이션, 데스크탑에서 사용되는 데스크탑 어플리케이션 그리고 디바이스의 데이터베이스에 직접 접근해 쓰거나 읽을 수 있는 Sync 매니저 API와 같은 요소들을 전부 혹은 부분적으로 포함하게 된다[6, 7].

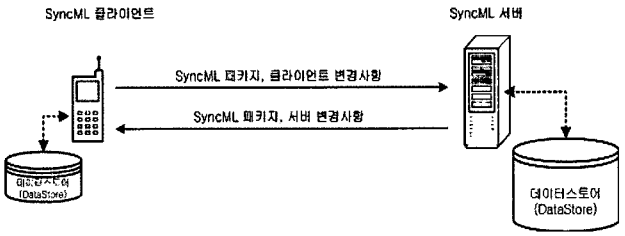
2.2 ActiveSync 프로토콜

ActiveSync는 Windows CE 기반에서 동작하는 PDA와 데스크탑 PC간의 데이터 동기화를 위한 방법이다. 또한, ActiveSync는 Windows CE 기반 단말기를 위해 장치 정보(Deivce Info.)의 저장과 백업, 프로그램의 인스톨과 제거와 같은 특징들을 제공하고, 데스크탑 PC와 Windows CE 기반 단말기 사이의 상호작용을 위해 데이터 동기화, 데스크탑 컴퓨터와 단말기 사이의 파일 변경, 데이터베이스 테이블 상호교환, 데스크탑 원격접속을 제공한다.

ActiveSync는 서버에 해당하는 서비스 매니저와 클라이언트에 해당하는 서비스 제공자로 구성되어 있다. 서비스 매니저는 ActiveSync의 동기화 엔진으로 데스크탑 PC와 Windows CE 기반의 단말기 모두에 존재한다. 모든 종류의 데이터에 적용될 수 있는 여러 동기화 명령들을 처리한다. 서비스 제공자는 서비스 매니저에 의해 변경된 사항들을 바탕으로 동기화할 데이터를 결정한다[8].

2.3 SyncML

SyncML은 각 제조회사의 단말기가 가지고 있는 서로 다른 플랫폼, 통신 프로토콜, 자료 형태, 응용 서비스에 이용될 수 있는 자료 동기화 방식과 장치 관리에 대한 개방형 표준 인터페이스 개발을 목적으로 한다. (그림 1)은 SyncML을 기반으로 하는 클라이언트와 서버간의 자료 동기화를 나타내고 있다[1, 2].



(그림 1) SyncML을 기반으로 하는 클라이언트-서버 구조

SyncML 클라이언트가 먼저 서버에게 갱신된 자료를 포함된 메시지를 전송하면, SyncML 서버는 클라이언트가 요청한 동기화 타입에 의해 서버 측 자료와 동기화 작업을 수행한 후, 다시 클라이언트에게 작업 결과 및 서버 자신의 변경된 자료도 전송한다. 이러한 몇 번의 메시지 교환을 수행함으로써, 자료 동기화가 이루어지며 클라이언트와 서버간의 해당 자료에 대하여 일치성을 보장하게 된다.

2.4 SyncML 규격

SyncML 자료 동기화 규격은 XML 기반의 자료 표현(Data Representation) 프로토콜, SyncML 동기화(Synchronization) 프로토콜 그리고 전송 바인딩(Transport Bindings) 프로토콜로 구성되어 있다[2, 9, 10].

2.4.1 SyncML 데이터 표현 프로토콜

SyncML 자료 표현 프로토콜은 자료 동기화를 위해 교환되는 SyncML 메시지의 논리적인 구조와 형태를 XML 형식으로 정의하고 있다. 각각의 필드가 어떠한 정보를 담고 있으며 해당 정보가 어떤 의미를 내포하는 것인지에 대한 약속을

정의하고 있다[10, 11].

2.4.2 SyncML 동기화 프로토콜

SyncML 동기화 프로토콜은 SyncML 클라이언트와 서버 간에 이루어지는 자료 표현 프로토콜 규격에 의해 생성된 자료의 추가, 삭제, 갱신과 같은 동기화 명령과 그 밖의 상태 정보에 대한 메시지가 교환되는 방법에 대해서 정의하고 있다[10, 12].

2.4.3 전송 바인딩 프로토콜

SyncML이 메시지를 전송하기 위해서 사용하는 전송 바인딩 프로토콜은 HTTP, WSP, OBEX 등과 같은 프로토콜인데, SyncML 규격에서는 이러한 전송 바인딩 프로토콜에 대해서 새로운 프로토콜을 정의하는 것이 아니라, 기존의 전송 프로토콜과 바인딩 규칙만을 정의한다. 따라서 자료 표현 프로토콜과 동기화 프로토콜이 전송 프로토콜과 독립적으로 구성되어 있으므로 향후 다른 전송 프로토콜과의 바인딩이 가능하다[10, 13].

2.5 사례 연구

2.5.1 Sync4j 동기화 시스템

SyncML 기반의 Sync4j 동기화 서버는 자바를 이용하여 개발되었고, SyncML 기반의 Sync4j 클라이언트는 자바와 C++를 이용하여 개발되었다. Pocket PC 기반에서 클라이언트가 동작하며, Microsoft Exchange 서버와 연동하여 주소록이나 이벤트와 같은 PIMS 정보를 동기화 할 수 있다[14].

2.5.2 Synthesis 동기화 시스템

Synthesis AG사에서 개발한 Synthesis 서버는 PHP 기반으로 MS-SQL ODBC를 연동하여 웹을 통하여 PIMS 데이터 관리가 가능하며, Synthesis 클라이언트는 PIMS 정보를 vCard, vCalendar 형식의 XML 파일로 변환하며, Synthesis에 내장된 파서는 각각의 동기화 정보에 대해서 각 필드별로 비교하여 동기화를 수행한다[15].

2.5.3 SyncLE

국내의 Neosteps라는 회사에서 개발한 SyncML 기반의 동기화 시스템이다. SyncML 국제 적합성 시험과 상호 운용성 시험에 합격하였으며, SyncML 스펙 1.1.1을 준수하고 있다. 셀룰러 폰이나 PDA와 같은 이동무선통신 단말기뿐만 아니라 MS-Windows가 설치된 PC 플랫폼 환경에서도 동작하는 동기화 솔루션이다[16].

3. 설 계

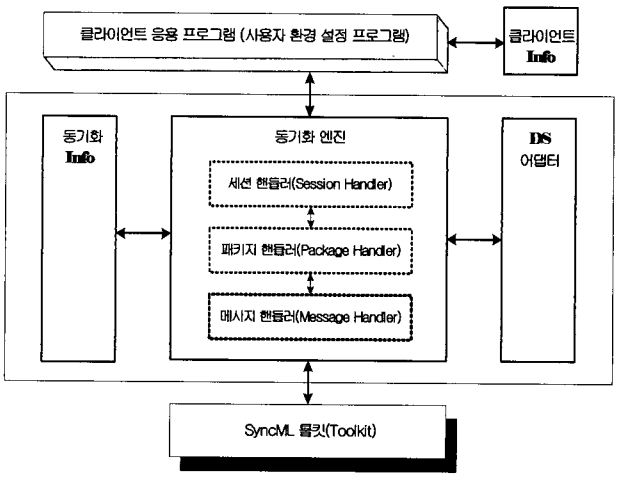
SyncML은 자료 동기화 방식과 장치 관리에 대한 개방형 표준 인터페이스 개발을 목적으로 OMA에서 제안되었으며, 3개의 기본 규격으로 구성되어 있다. 첫째, 자료 동기화를 위해 교환되는 SyncML 메시지의 논리적인 구조와 형태를 XML 형식으로 정의하고 XML 기반의 데이터 표현 프로토콜과 둘째, 동기화 명령과 상태 정보에 대한 메시지가 교환되는 방법에 대해서 정의하기 위한 동기화 프로토콜, 마지막으로 실제 동기화 메시지를 전송하기 위한 전송 바인딩 프로토콜로 구성되어 있다.

본 논문에서 설계한 세션 핸들러 모듈은 통신 세션을 제어하고 교환되는 자료 동기화 메시지의 헤더 부분을 생성 및 메시지의 적합성을 판단하는 역할을 담당하며, 전체 SyncML 동기화 클라이언트의 엔진 구조는 세션(Session), 패키지(Package), 메시지(Message)와 같은 자료 동기화 단위에 따라 세션 핸들러, 패키지 핸들러, 메시지 핸들러로 구성되어 있다.

3.1 SyncML 기반의 동기화 클라이언트의 구조

SyncML 클라이언트는 동기화 엔진, DS(DataStore) 어댑터, 클라이언트 정보, 동기화 정보, SyncML 툴킷[17] 그리고 클라이언트 응용 프로그램 부분으로 구성되어 있다. 클라이언트 동기화 엔진 부분은 동기화 할 자료를 포함하여 서버에게 전송하여야 할 동기화 메시지를 생성하고, 서버로부터 수신된 메시지를 처리 및 반영한다. 또한 동기화 엔진은 동기화 단위에 따라 분류된 세 개의 모듈로 구성되어 있다. (그림 2)는 동기화 클라이언트의 전체 구조를 나타내고 있다.

클라이언트 응용 프로그램은 PIMS(Personal Information Magagement System) 데이터의 추가, 수정, 삭제 등의 처리와 사용자 환경 설정 프로그램을 통해서 클라이언트 정보를 처리한다. 클라이언트 정보는 동기화하기 위한 목적지의 주소(Target URL), 클라이언트의 주소(Source URL), 각 콘텐츠별 데이터 스토어(Data Store)의 경로, 인증을 위한 사용자의 아이디(ID)와 패스워드(Password) 등을 XML 문서 형태로 가지고 있다.



(그림 2) 동기화 클라이언트의 구조

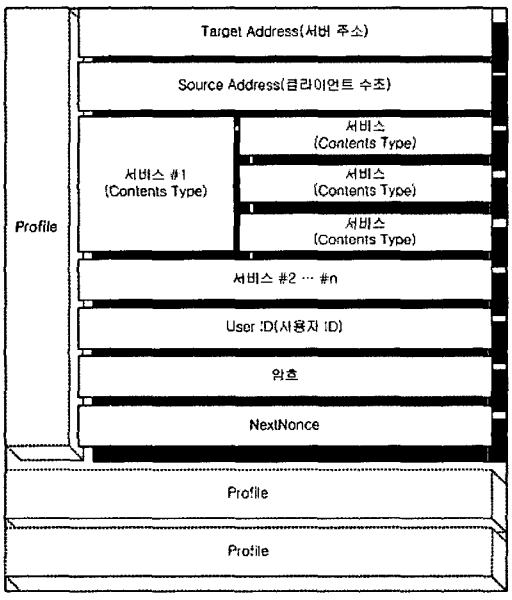
동기화 엔진은 세션 핸들러, 패키지 핸들러, 메시지 핸들러의 세 개의 모듈로 구성되며, 세션 핸들러(Session Handler)는

SyncML 툴킷에서 연결된 전송 프로토콜(Transport Binding : HTTP, WAP, OBEX) 종류에 따라 선택된 해당 프로토콜의 세션을 초기화하며, 동기화하기 위해 송신하는 메시지의 헤더 부분을 생성하거나, 수신된 메시지의 헤더의 적합성을 검사한다. 패키지 핸들러(Package Handler)는 SyncML 문서의 <SyncBody> 엘리먼트(element) 부분의 논리적 구성을 담당한다. 논리적 구성을 위해서 메시지 핸들러에게 서버로부터 받은 메시지의 종류를 확인하고, DS 어댑터에게 이전 동기화 이후의 변경사항을 확인한다. 메시지 핸들러(Message Handler)는 패키지 핸들러가 명령어를 생성하려 할 때 호출되는 함수들의 집합이며, 모든 명령어들에 대해서 적절한 처리를 해주는 콜백(Call Back) 함수 모듈을 포함하고 있다. SyncML 툴킷은 전송 프로토콜간의 실제 메시지의 송신과 수신을 담당하면서, 동기화하기 위해 생성된 SyncML 메시지를 인코딩하고, 서버로부터 수신된 메시지를 디코딩한다.

3.2 GUI 기반의 사용자 환경 설정 프로그램

Qt 라이브러리를 이용한 GUI 기반의 사용자 환경 설정 프로그램은 클라이언트 정보에 대하여 각 사용자 정보의 추가, 수정 및 삭제가 가능하다. (그림 3)은 클라이언트 정보의 구조를 보여주고 있다.

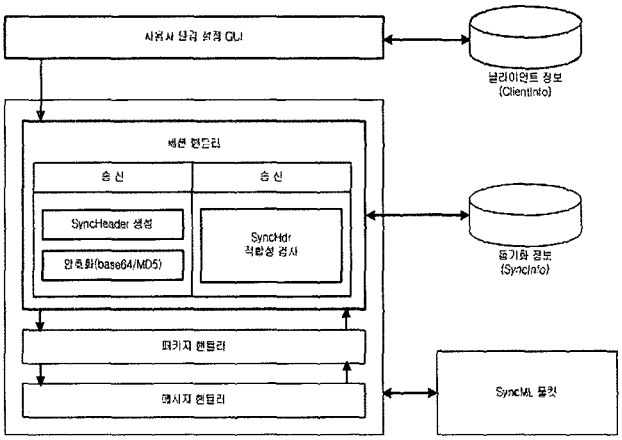
클라이언트 정보는 다수의 각 사용자에 대한 프로필(profile) 정보와 동기화 대상이 되는 서비스를 가지고 있다. 타켓 주소는 동기화 대상이 되는 서버의 주소이며, 소스 주소는 동기화를 요청하는 클라이언트의 주소이다. 서비스는 동기화 대상이 되는 각각의 컨텐츠 타입들을 의미하고, 서비스 내에는 동기화를 시도하는 컨텐츠의 동기화 모드를 설정하고, 서버 측과 클라이언트 측의 동기화 경로를 가진다. 사용자 인증을 위해 사용자 ID와 패스워드, NextNonce에 대한 값이 저장되어 있으며, Base64와 MD5 방법을 이용하여 인증 절차를 수행한다.



(그림 3) 클라이언트 정보의 구조

3.3 세션 핸들러 모듈

세션 핸들러는 SyncML 메시지를 주고받는 통신 세션을 제어하는 기능과 SyncML 메시지 중에서 <SyncHdr> 엘리먼트에 관련된 처리를 하는 두 기능을 담당하고 있다. 첫째, 통신 세션을 제어하는 기능은 SyncML 툴킷에서 제공하는 통신 인터페이스를 사용하는데, 사용자 환경 설정 프로그램에서 클라이언트 정보를 넘겨받아 해당 단말기에서 사용 가능한 프로토콜을 선택하고, 이 후 선택된 프로토콜에 적합한 환경으로 초기화하며 메시지를 주고받는 과정에서 일어날 수 있는 오류 검사를 수행한다. 둘째, <SyncHdr> 엘리먼트에 관련된 기능은 클라이언트에서 동기화를 요청할 때는 패키지 핸들러와 메시지 핸들러로부터 수신된 동기화 내용에 대한 <SyncHdr> 엘리먼트를 생성하고 클라이언트가 동기화 대상이 되어 수신된 SyncML 메시지의 <SyncHdr> 엘리먼트의 적합성을 검사를 수행한다. (그림 4)는 세션 핸들러 모듈의 구조를 나타낸다.



(그림 4) 세션 핸들러 모듈의 구조

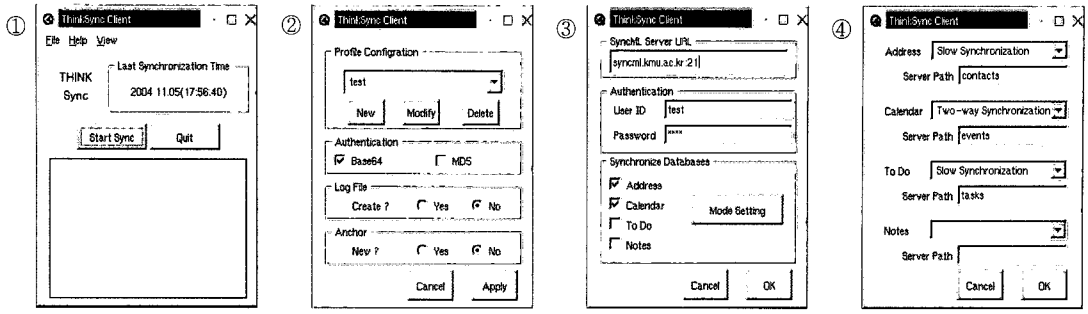
4. 구현 및 검증

본 연구에서 구현된 사용자 환경 설정 프로그램과 세션 핸들러 모듈은 리눅스(커널 버전 2.6.0) 운영체제 기반으로 SyncML 툴킷(Reference Toolkit) 4.3과 Qt Library(버전 3.1.1)를 이용하였다. 세션 핸들러를 포함하고 있는 동기화 엔진은 C 언어로 작성하였으나, 사용자 환경 설정 프로그램에서 이용한 Qt Library가 C++기반 언어이기 때문에 전체적인 컴파일은 g++ 3.2.2를 사용하였다.

4.1 사용자 환경 설정 프로그램의 구현

사용자 환경 설정 프로그램은 동기화의 실행 명령, 프로그램의 종료, 마지막으로 동기화된 시각 및 동기화 결과를 나타내는 화면으로 구성된 동기화 시작 화면이 있다. (그림 5)는 Qt 라이브러리 기반의 에뮬레이터에서 실행된 각각의 사용자 환경 설정 화면을 보여주고 있다.

①번 그림은 동기화 클라이언트가 대상 서버에게 동기화를 요청하기 위한 초기화면을 보여주고 있다. ②번 그림은 동기



(그림 5) Qt 라이브러리 기반의 사용자 환경 설정 프로그램

화 요청 이전에 사용자 정보에 대한 변경 및 이전 사용자에 대한 설정을 변경하기 위해 프로필을 선택하고, 인증 방식, 동기화 로그파일 저장 여부 및 새로운 앵커 정보 값을 적용할 것인지에 대해서 선택할 수 있다. ③번 그림은 동기화 대상이 되는 서버의 주소와 인증을 위한 사용자 ID, 암호 및 동기화 대상이 되는 컨텐츠 타입에 대해서 선택할 수 있다. ④번 그림은 동기화 모드와 서버측의 동기화 경로를 설정할 수 있다.

클라이언트 정보 파일은 XML 파일로서 저장되는데, 클라이언트 정보 파일은 사용자 환경 설정 프로그램이 실행될 때 XML 파서를 이용하여 분석한 후, 최종적으로 동기화가 시작될 때, 리스트로 구성되어 세션 핸들러 모듈로 넘겨준다.

4.2 세션 핸들러 모듈의 구현

구현된 동기화 클라이언트는 동기화 세션에 관여하는 세션 핸들러와 생성할 메시지를 판단하는 패키지 핸들러, 실제 메시지를 생성하는 메시지 핸들러 및 다양한 형태의 서비스 자료들과 동기화 엔진 사이의 인터페이스 역할을 담당하는 데이터스토어 어댑터로 구성되어 있다.

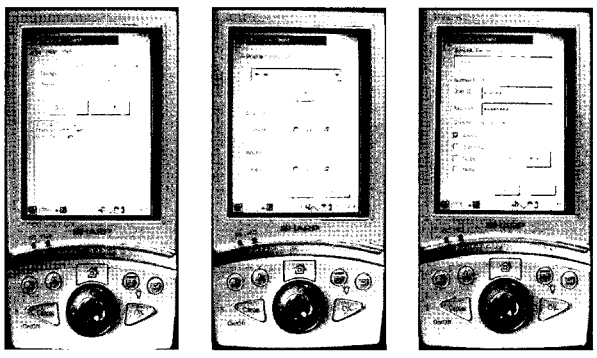
사용자 인터페이스를 통하여 사용자가 동기화 명령을 요청하면 동기화 엔진이 작동하게 된다. 세션 핸들러는 해당 프로토콜에 대한 통신 세션을 툴킷에서 제공하는 인터페이스를 사용하여 동기화 세션을 열고, 동기화 세션에 필요한 정보들을 초기화하게 된다. 서버와 동기화 할 준비가 완료되면 패키지 핸들러에서 생성할 메시지를 결정 및 판단하고, 메시지 핸들러를 통하여 SyncML 메시지 생성을 완성한다. SyncML 메시지가 생성되는 과정에서 서버와 동기화 할 자료가 있다면 DS 어댑터를 통하여 메시지 내에 포함되어야 한다. 동기화 세션 과정 중에서 서버로부터 SyncML 메시지를 받게 되면 세션 핸들러를 거쳐서 메시지 핸들러에서 정의된 콜백 함수들이 호출된다. 각 함수들은 SyncML 메시지들이 원하는 동작들을 수행되도록 한다.

서버로부터 받은 SyncML 메시지 내에 저장되거나 변경될 자료에 대한 정보가 있다면 데이터스토어 어댑터를 통하여 자료 변경을 수행한다. 동기화 처리 대상의 자료는 단순한 파일 시스템으로 제어를 하며, 현재 구현되어 있는 파일 시스템에는 PIMS 정보가 저장되어 있다. 동기화 프로그램이 동기화를 진행하는 동안에 PIMS 정보에 접근하면 동기화 오류가 발생하므로, 이를 방지하기 위하여 동기화 터널을 통하여 파일에 접근하게 하였다.

4.3 검증

구현된 세션 핸들러 모듈을 포함한 SyncML 자료 동기화 클라이언트에 대한 정상적인 동기화 수행 여부를 검증하기 위해서 OMA 상호 운용성 시험에 합격한 Synthesis 자료 동기화 서버와 연동하여 시험하였다. Synthesis DS 서버에는 기본적으로 PIMS(Personal Information Management System) 정보가 저장되어 있으므로, 동기화 검증에 사용된 동기화 대상 자료로서 PIMS 정보를 사용하였다. 동기화 클라이언트에서 생성된 PIMS 정보는 HTTP 프로토콜을 통해서 SyncML 메시지 형태로 Synthesis 서버로 전송되고, 전송된 PIMS 정보는 Syntheis 엔진으로 전달된다. Synthesis 서버는 도착된 PIMS 정보를 데이터스토어 객체로 변환하여 처리한 후, 자신의 변경사항 혹은 처리결과에 대한 응답을 SyncML 메시지로 작성하여 동기화 클라이언트로 전송한다. 동기화 클라이언트에서 동기화 수행 가능한 서비스의 형태는 주소록(Contacts), 일정관리(Events), 작업관리(Tasks), 메모(Notes) 등과 같은 PIMS 자료이다. 동기화 클라이언트는 PC 기반의 터미널 환경에서 개발한 후, 자우루스(Zaurus) PDA에 포팅하였다. (그림 6)은 자우루스 PDA에 포팅된 동기화 클라이언트 화면을 보여주고 있다.

구현된 세션 핸들러 모듈의 정상적인 동작 여부는 송신한 <SyncHdr> 엘리먼트에 대한 응답 메시지에서 확인할 수 있다. 또한, 정상적인 <SyncHdr> 엘리먼트를 송신하였다면 서버 측으로부터 수신한 SyncML 메시지의 <Status> 엘리먼트에 212 코드값이 저장되어 수신된다. (그림 7)은 동기화 클라이언트에서 생성된 <SyncHdr> 엘리먼트와 수신된 <Status> 엘리먼트에 대한 XML 정보를 보여주고 있다.



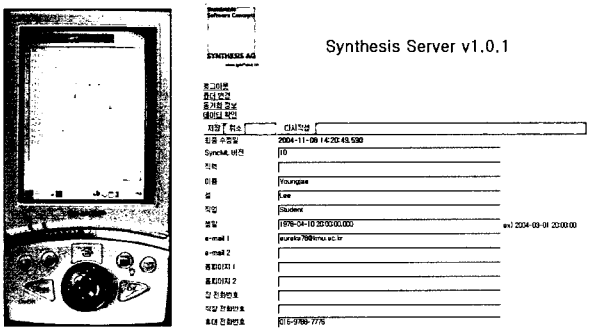
(그림 6) 자우루스 PDA에 포팅된 동기화 클라이언트

```
<?xml version="1.0" encoding="UTF-8" ?>
- <SyncML xmlns="SYNCML:SYNCML1.1">
  - <SyncHdr>
      <VerDTD>1.1</VerDTD>
      <VerProto>SyncML/1.1</VerProto>
      <SessionID>2</SessionID>
      <MsgID>1</MsgID>
    - <Target>
        <LocURI>syncml.kmu.ac.kr:21</LocURI>
      </Target>
    - <Source>
        <LocURI>IMEI:CEE111111111111</LocURI>
        <LocName>test</LocName>
      </Source>
    - <Cred>
      - <Meta>
          <Format xmlns="syncml:metinf">b64</Format>
          <Type xmlns="syncml:metinf">syncml:auth-basic</Type>
        </Meta>
        <Data>dGVzdDp0ZXN0</Data>
      </Cred>
    </SyncHdr>
```

```
- <Status>
    <CmdID>1</CmdID>
    <MsgRef>1</MsgRef>
    <CmdRef>0</CmdRef>
    <Cmd>SyncHdr</Cmd>
    <TargetRef>syncml.kmu.ac.kr:21</TargetRef>
    <SourceRef>IMEI:CEE111111111111</SourceRef>
  - <Chal>
    - <Meta>
        <Format xmlns="syncml:metinf">b64</Format>
        <Type xmlns="syncml:metinf">syncml:auth-basic</Type>
      </Meta>
    </Chal>
    <Data>212</Data>
  </Status>
```

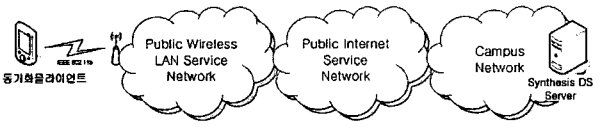
(그림 7) 생성된 〈SyncHdr〉 엘리먼트 및 수신된 〈Status〉 엘리먼트

자우루스 PDA에 포팅된 클라이언트가 세션 핸들러 모듈을 이용하여 정상적인 동기화가 이루어지는지 확인하기 위해 vCard 형식의 자료를 Synthesis DS 서버와 동기화하였다. (그림 8)은 vCard 형식의 자료가 동기화 클라이언트와 Synthesis DS 서버에서 정상적으로 동기화되었음을 보여주고 있다.



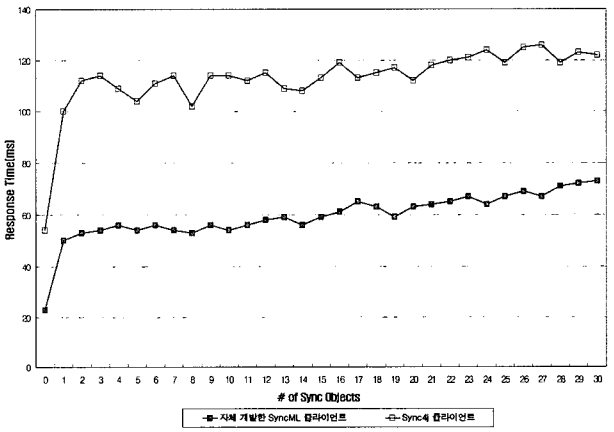
(그림 8) 동기화 클라이언트와 Synthesis 서버간의 PIMS 자료 동기화

(그림 9)는 성능 시험을 위한 네트워크 구성도를 보여주고 있다. 네트워크 성능 시험을 위하여 사용한 동기화 객체는 PIMS 정보 중에 연락처 정보에 대한 표준인 vCard 2.0을 사용하였다. 성능 측정을 위하여 사용한 vCard 정보는 일반적으로 사용되는 정보만을 구성하여 사용하였으며, 응답시간을 측정하기 전에 ping 도구를 이용하여 네트워크 지연시간을 측정하였다.



(그림 9) 성능시험 네트워크 구성

(그림 10)은 세션 핸들러 모듈이 포함된 전체 SyncML 기반의 동기화 클라이언트를 이용하여 동기화를 요구하고, 응답을 수신하는 시간을 측정한 결과를 보여주고 있다. 본 논문에서 자체 개발한 SyncML 동기화 클라이언트의 응답 시간에 대한 성능 측정을 위해서 공개 프로젝트 클라이언트인 Sync4j와 성능 비교를 하였다. 또한, 성능 평가를 위한 동기화 대상 서버는 동일하게 Synthesis AG사의 Synthesis DS 서버를 사용하였다.



(그림 10) 자체 개발한 SyncML 클라이언트와 Sync4j 클라이언트 간의 응답시간 비교

SyncML 규격에 의하면 데이터 동기화는 인증과정과 데이터 전달과정으로 이루어져 있으며, 패키지 #1에서 패키지 #6까지 교환함으로써, 정상적인 데이터 동기화 과정을 수행하게 된다. 또한, 데이터 동기화 과정에서 패키지 #1과 패키지 #2는 인증절차 및 동기화 대상 장치에 대한 정보를 교환함으로써 실제 동기화해야 할 데이터는 포함되어 있지 않다. 따라서, 패키지 #1과 #2가 포함된 인증과정은 실제 데이터 오브젝트에 대한 응답시간과는 관련이 없으므로, 본 실험에서 측정한 응답시간은 인증과정을 포함시키지 않고 데이터 전달만 측정하였다. 동기화 객체의 수가 0일 때 응답시간은 서버에서 동기화 객체가 없다고 요청 메시지를 보내고 이에 대한 응답시간을 측정한 것이다. 또한, 30개를 초월한 동기화 객체의 경우 다른 SyncML 패키지로 전달되어 의미가 없기 때문에 동기화 객체수를 30개까지만 측정했다. 본 논문에서 자체 개발한 SyncML 기반의 동기화 클라이언트는 동기화 객체가

증가하면서 응답시간이 비교적 증가하지만, Sync4j 클라이언트의 응답시간에 비하여 평균 48%의 성능이 상대적으로 향상됨을 알 수 있다. 이는 제안한 SyncML 기반의 동기화 클라이언트는 동기화 대상이 되는 객체의 처리 단위에 따른 구조적인 설계와 C 언어를 기반으로 구현하였으며, 또한 자료 동기화를 위한 별도의 API가 존재하여 자료 처리 속도를 개선했기 때문이다. 반면에 기존의 Sync4j 클라이언트 경우 성능보다는 기능 위주의 개발 및 JVM 기반의 Java 언어로 개발되었으며 자료 동기화를 위한 별도의 API가 없어 동기화 엔진 자체에서 복잡한 코드를 포함하여 모든 자료를 개별적으로 처리해주어야 하기 때문이다.

5. 결 론

본 논문에서 구현한 클라이언트의 세션 핸들러 모듈은 개방형 표준 자료 동기화 기술인 SyncML을 기반으로 구현되었으며, 사용자 환경 설정 프로그램은 이동무선통신 환경에 맞도록 Qt 라이브러리를 기반으로 설계하고 구현하였다. 구현한 모듈은 클라이언트 장치의 특성을 고려한 효율적 구현을 위하여 하나의 프로세스 형태로 된 경량의 구조를 가지도록 하였으며, 응답처리 속도 향상을 위하여 동기화 단위별로 조작 모듈을 구분하였다. 또한, Synthesis DS 서버를 이용하여 Sync4j와 응답시간에 대한 성능비교를 한 결과, Sync4j 클라이언트의 응답시간에 비하여 자체 개발한 SyncML 클라이언트가 평균 48%의 향상됨을 알 수 있다.

향후 과제로는 동기화 대상이 되는 자료가 대부분 개인 정보 자료이므로, 동기화 과정에서의 송수신되는 SyncML 메시지에 대한 보안을 강화할 필요가 있다. 그리고, 동기화 클라이언트는 이동무선통신 기기의 특성상 제한적인 처리 능력을 가지고 있으므로 클라이언트 장치의 특성을 최대한 고려해서 최적화된 메모리 사용과 불필요한 코드의 최소화 작업이 요구된다. 향후 구현된 모듈이 다양한 플랫폼에서 동작할 수 있는 환경과 여러 통신 프로토콜로 전송이 가능하도록 개선하는 연구가 필요하겠다.

참 고 문 헌

[1] Uwe Hansmann, Riku Mettala, Apratim Purakayastha, Peter Thompson, SyncML Synchoronizing and Managing Your Mobile Data, pp.21~PRENTICE HALL PTR, New Jersey, 2003.

[2] SyncML Initiative, Building an Industry-Wide Mobile Data Synchronization Protocol, SyncML WhitePaper, 2000.

[3] Mattbias Kalle Dalbeimer, Programming with Qt, O'REILLY, 2000.

[4] S. Agarwal, D. Starobinski, A. Trachtenberg, "On the Scalability of Data Synchronization Protocols for PDAs and Mobile Devices," Network IEEE, Vol.16, Issue 4, pp.22~28, 2002.

[5] DaeJin Jang, Hong Taek Ju, KeeHyun Park, B.H.Ha, M.C.Lee, Sung-Chae Bae, "Design of ThinkSync DM based on SyncML Device Management," The 3rd APIS, pp.569~574, 2004.

[6] Lonno R. Foster, Palm OS Programming, WILEY

[7] Neil Rhodes, Julie McKeehan, Palm OS Programming, O'REILLY

[8] Douglas Boling, Programming MiCROSOFT WINDOWS CE .NET, Microsoft Press

[9] Byung-Yun Lee, Tae-Wan Kim, Dae-Woong Kim, Hoon Choi, "Data Synchronization Protocol in Mobile Computing Environment Using SyncML," The 5th IEEE International Conference, pp.133~137, July, 2002.

[10] Ligang Ren, Junde Song, "Data Synchronization in the Mobile Internet," The 7th IEEE International Conference, pp.95~98, September, 2002.

[11] SyncML Initiative, SyncML Representation Protocol v1.1, SyncML Forum, 2002.

[12] SyncML Initiative, SyncML Synchronization Protocol v1.1, SyncML Forum, 2002.

[13] SyncML Initiative, SyncML HTTP Binding v1.1, Sync ML Forum, 2002.

[14] Sync4j, http://sync4j.sourceforge.net/web/theproject.html

[15] Synthesis AG, http://www.synthesis.ch/, Zürich Switzerland, 2003.

[16] SyncLE, http://neosteps.com/

[17] SyncML Toolkit, http://sourceforge.net/projects/syncml-ctoolkit/

하 병 훈

e-mail : hooni76@kmu.ac.kr

1995년~1999년 계명대학교 정보통신학부 컴퓨터공학과(학사)

2002년~2005년 계명대학교 일반대학원 컴퓨터공학과(석사)

관심분야 : 모바일소프트웨어, 무선통신 프로토콜, 무선 인터넷

박 기 현

e-mail : khp@kmu.ac.kr

1975년~1979년 경북대학교 전자공학과(학사)

1979년~1981년 한국과학기술원 전자계산학과(석사)

1986년~1990년 미국 Vanderbilt 대학교 전자계산학과(박사)

1981년~현재 계명대학교 정보통신학부 교수

관심분야 : 병렬처리시스템, 모바일 소프트웨어, 임베디드 소프트웨어, 운영체제

주 홍 택

e-mail : juht@kmu.ac.kr

1986년~1989년 한국과학기술원 전산학(학사)

1989년~1991년 포항공과대학교 컴퓨터공학(석사)

1997년~2002년 포항공과대학교 컴퓨터공학 박사

1991년~1997년 대우통신, 종합연구소, 선임연구원

2000년~현재 계명대학교 정보통신학부 교수

관심분야 : 모바일단말기관리시스템, Network Management, 임베디드 시스템

우 종 정

e-mail : jwoo@sungshin.ac.kr

1976년~1982년 경북대학교 전자공학과 학사

1982년~1988년 산업연구원 책임연구원

1988년~1993년 텍사스주립대학(오스틴) 전기컴퓨터공학과 석사 및 박사

1998년~1999년 텍사스주립대학(오스틴) 전기컴퓨터공학과 객원교수

1993년~현재 성신여자대학교 컴퓨터정보학부 교수

관심분야 : 컴퓨터구조, 병렬처리, 임베디드 시스템, 모바일컴퓨팅, 원격교육