모바일 개인정보관리시스템(PIMS)의 설계 및 구현

최 이권, 이 상범
단국대학교 전자계산학과

Design and Development of Mobile PIMS

LeeKwon Choi, SangBum Lee
Dept. of Computer Science, Dankook Univ.

요 약

본 논문에서는 무선 인터넷 주요 기술로 관심을 받고 있는 분산데이터 동기화 표준인 SyncML을 이용한 개인정보관리 시스템의(PIMS) 설계 및 구현에 대한 소개하고자 한다. 객체 지향 설계 기법을 사용하여 분석,설계하여 최적화 된 시스템을 도출하고 그것을 통해 무선인터넷상에서 경량화 된 동기화 최적의 시스템을 설계와 구현하는데도 목적이 있다. 특히 무선인터넷의 여러 제약 사항 특히 데이터의 관리에 있어서의 문제점을 동기화 기술을 통해 많은 데이터를 효과적으로 관리함으로써 대용량의 데이터 베이스와 디바이스에 상관없이 데이터 일관성을 유지하여 지연성 및 복수성에 신뢰성을 유지 할 수 있도록 시스템을 개발하였다.

1.서론

현재 제3세대 이동통신인 IMT2000의 등장과 더불어 무선 인터넷이 음성중심의 서비스에서 데이터통신 서비스로 전환되고 있다. 모바일 인터넷은 통신 기술의 발달과 단말기 기술에 힘입어 2004년도에는 전 세계적으로 7억 5020만의 사용자로 이는 유선 인터넷 가입자 수를 능가하게 된다고 예측하고 있다. [1][2]

핸드폰이나 PDA와 같은 모바일 단말기는 이동성은 좋지만 단말기 저장 능력의 한계로 인해 많은 정보를 저장할 수도 없을 뿐만 아니라 경우에 따라서는 여러 곳에 분산되어 있는 정보를 이용할 필요성이 많이 생긴다. 따라서 분산된 데이터간의 데이터 변경에 따라 정보의 내용을 일치시킬 수 있는 데이터 동기화 (Data Synchronization)가 절대적으로 필요하게 된다.

현재 급속하게 성장하고 있는 응용기술 및 장비와 함께 표준 동기화 프로토콜기술이 제공되어 진다면 개인이 소유하고 다양한 정보기기 및 네트워크에 존재하는 수많은 종류의 정보를 디바이스에 상관없이 접속하여 최신의 정보를 주기적으로 업데이트시킬 수가 있다. 이렇게 된다면 사용자는 PC, PDA, PCS, 휴대폰, 웹패드, 스마트폰 등 다양한 차세대 이동통신 단말기에 대한 중복투자를 방지할 수 있을 뿐만 아니라 업무에 많은 효율을 높일 수가 있게 된다. 예를 들어 웹상의 주소록을 갱신하면 실시간으로 휴대폰, PDA 등의 단말기안에 보관되어 있는 주소록도 자동으로 갱신되기 때문에 소비자들은 어떠한 기기

에서도 동일한 데이터를 관리할 수 있는 장점이 있다.

본 논문에서는 서로 다른 모바일 디바이스를 통해 개인정보을 관리할 때 데이터동기화 프로토콜인 SyncML 기술을 적용하여 시제품을 설계 및 구현하였다. 본 연구에서는 SyncML기술 적용하여 개발시 예상되는 문제점, 개발도구, 개발 환경 등을 미리 셋팅하고 다른 언어로 converting이 용이하게 세부적인 process들을 모듈화 하여 UML로 설계하였으며, 개인이 사용할 수 있는 PIMS(Personnel Information Management System)을 구현하였다.

제 2장에서는 본 연구의 기반기술인 데이터동기화 기술인 SyncML에 대하여 기술하였으며, 제 3장에서는 모바일 PIMS 솔루션에 대하여 설명하였다. 제 4장에서는 결론과 향후 연구 방향에 대하여 기술하였다.

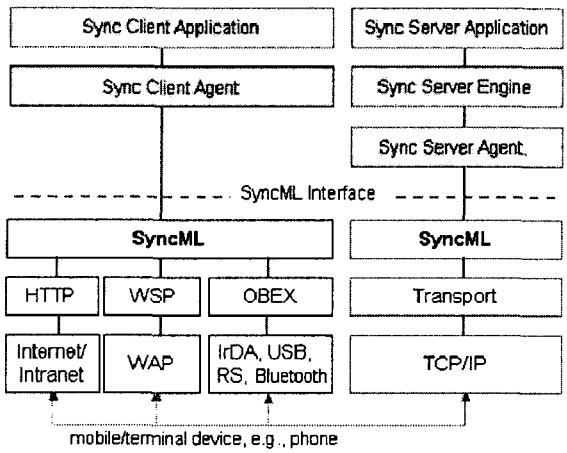
2. 데이터 동기화 표준(SyncML)

데이터 동기화(Data Synchronization)란 여러 이동단말과 서버에 존재하는 데이터의 내용을 일치시키는 것을 말한다. SyncML이란 서로 다른 디바이스 및 어플리케이션 간에데이터 동기화를 제공해 주기 위해 제정된 표준 언어이다 . SyncML은 유, 무선간에 데이터 동기화를 제공하며,데이터 전송을 위해 HTTP, WSP, OBEX 등 다양한 전송 프로토콜을 이용한다. 그리고 일정관리, 주소록등 PIMS의 여러기능 이외의 다양한 서비스를 이용할 수 있도록 확장 가능하다. 또한 명령어를 포함한 데이터 동기화를 위한 모든 정보를 XML DTD 타입으로 정의하고, 일반 텍스트 형태인 XML 뿐만 아니라 바이너리 형태인 WBXML 형식의 메시지를 제공함으로써 쉽고 빠르게 디바이스, 서버간에 데이터 동기화를 제공한다.[4][5][6][7][9]

2.1 SyncML 아키텍쳐

SyncML은 Server와 Client의 구분을 정의하고 있다. 논리 구분상 동기화 요청을 하는 쪽을 Client라 칭하고 반대로 동기화 요청을 받는 쪽을 Server부른다. Server 쪽이 되는 App A는 Sync Engine을 통해 동기화 요청을 처리하고 Sync Server Agent를 통해 동기화 요청이 있는지를 지속적으로 감시한다. Agent는 내부적으로 SyncML Interface를 통해 XML 형식의 메시지를 만들어 Transport로 전송한다.[4][5]

Client가 되는 App B는 Sync Client Agent를 통해 동기화 요청을 시작한다. Agent는 Server와 마찬가지로 SyncML Interface를 통하여 XML 형식의 만들어 Transport로 전송한다. SyncML의 이러한 아키텍쳐를 기능단위로 분리하여 좀더 확장이 쉽고 기능 추가가 용이한 아키텍쳐 구조로 변경하였다.[6][7]



[그림1] syncML 구조

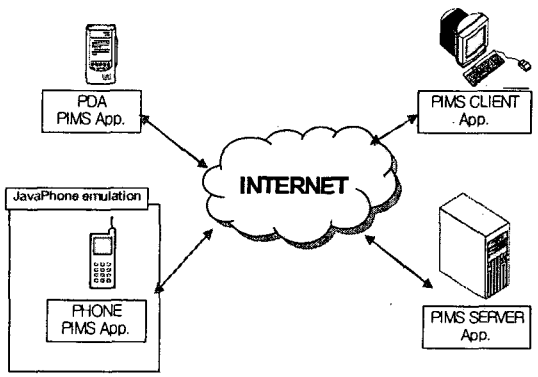
동기화 시스템은 논리적 모듈단위인 서브시스템(Sub-system)으로 구분하면 세가지의 모듈로 나눌수 있다. Application layer 와 Transport Layer 그리고 SyncML Framework이다. 각각의 서브시스템의 내부에 그 기능에 따라 구분된다. 본 어플리케이션은 이러한 객체와 모듈의 구조적인 결합으로 완성된다. 이러한 구조는 결과적으로 각각의 Layer를 다른 Layer 와 구분시켜 다른 Layer의 확장 및 변환을 자기 이외의 Layer로 확장 시키지 않는다. 결과적으로 기능 및 성능의 확장이 자유로운 구조가 된다. 또한 이러한 서브시스템 내부의 모듈들도 인터페이스를 적극 활용하여 기능 모듈 또는 객체의 이식이 자유

롭도록 설계하였다.

3. 시스템 설계 및 구현

3.1 시스템 개요

PIMS 시스템은 개인 컴퓨터와 무선단말기간 또는 회사업무컴퓨터와 본 개인 컴퓨터에서의 일정 및 메일 주소록을 관리하는 시스템을 말한다. 본 논문에서 제시하고 있는 데이터 동기화 시스템의 구성도는 [그림2] 와 같으며 3-tier 구조로 구현하였다. 한 단말기의 데이터 동기화를 통해서 다른 단말기에 있는 모든 PIMS Application를 동기화시킬 수 있다. 시스템구성은 SyncML의 응용 모델인 SDA(SyncML Demo Application)에 기초하여 설계하였으며 phone 클라이언트는 디바이스 특성상 자바기반으로 구성하였으며 테스트는 자바 에뮬레이터를 사용하여 실시하였다. 클라이언트인 무선 단말기인 PDA, phone 또는 일반 PC에서 개인 주소록의 데이터를 update, delete, add할 경우 SyncML에 의해 서버와 연계하여 데이터의 동기화가 가능하다



[그림 2] PIMS 구성도

3.2 PIMS 시스템의 구성 및 구현

본 시스템의 개발을 위해 구축한 시스템에 대한 내용은 아래와 같다.

본 연구의 어플리케이션 개발을 위하여 PDA와 Phone 에뮬레이트를 사용하여 서버는 운영체제가 윈도우 2000인환경에서 개발하였다. 개발환경으로는 서버는 Java 서블릿, jni, c, c++을 사용하였으며 클라이언트는 J2ME toolkit를 사용하였다.[8][9]

SyncML은 SUN에서 제공하는 MIDP를 기반으로 구현했다. 파서는 XML 의 SAX중에서 일부분만 채택했고 현재 비연결 된 상태에서 이루어지며 http를 이용해서 서버에 connecting 해서 데이터 Sync가 이루어진다. PDA sync는 WinCE 기반의 PDA에서 기존 유선망을 이용하여 서버와 동기화를 한다.

3.2 테스트 시나리오

테스팅에서는 무선단에서 자바폰과 서버와의 데이터 동기화를 구현하였다. 직접 폰에 syncML 미들웨어단을 올려 테스팅하기에는 폰에 제약사항이 너무 많기 때문에 자바폰 애뮬레이터를 사용하여 로컬 환경에서 테스팅 했다. 시스템구성은 SyncML의 응용 모델인 SDA(SyncML Demo Application)에 기초하여 설계되었으며 클라이언트는 디바이스 특성상 자바기반으로 구성했고 테스트는 sun사에 제공하는 J2ME 기반의 자바 에뮬레이트에서 수행했다.

o 프로그램의 구조 및 설치

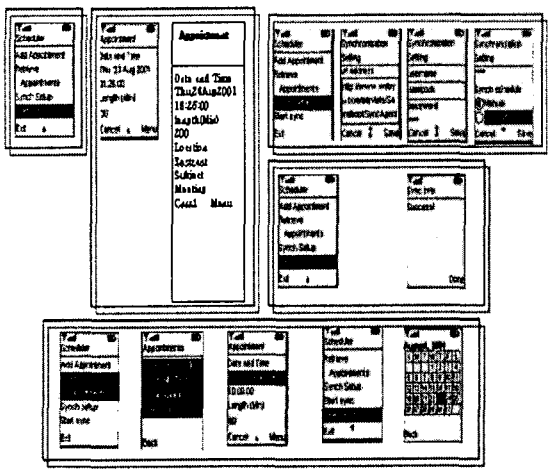
설치는 먼저 J2ME toolkit 설치하고 syncml.class, NativetrasferN.class 설치한다. 설치를 위해 제공되는 syncml.class, NativetrasferN.class을 resin의 doc₩WEB_INF₩classes 에 위치 시킨다.

SyncAdapterN.dll, TookitN.dll, syncEngineN.dll, SMLNotesN.dll 설치하고 설치를 위해 제공되는 SyncAdapterN.dll, TookitN.dll, syncEngineN.dll, SMLNotesN.dll 을 시스템 path에 등록된 system32 디렉토리에 위치시키고 시스템의 clsspath가 지정된 애뮬레이터를 실행하고 종료한다.[5][6][10]

o 폰 에뮬레이터 실행

동기화에서 사용한 스펙은 폰에서 제공하는

vCalendar version 1.0을 사용하였고 메모리의 문제점 때문에 가장 기본적인 명령어만 구현했다. 아래 [그림3]은 폰 동기화 화면 구성이다.



[그림 3] 폰 동기화 애뮬레이터 캡쳐화면

4. 결론과 향후 연구 방향

다양한 회사와 표준와 단체들이 플랫폼과 디바이스간 데이터 동기화의 문제점을 보완하기 위해 SyncML이라는 표준이 만들어지게 되었고,이를 이용한 SyncML을 이용한 SyncML 서버를 구현하였다. 본 논문에서는 SyncML에 대한 개략적인 소개와, 구현서버의 프레임, 구현한 서버 중 Sync Engine에 대하여 기술하였다. 본 논문에서 제시한 동기화 엔진은 데이터 동기화의 표준이라는 SyncML 자체의 장점 뿐만 아니라 서비스에 종속적인 부분을 Sync Engine이라는 프레임으로 분리를 함으로써 서비스의 확장성에 빠르게 대처할 수 있으며, 데이터간 충돌 발생 등에 관한 정책 결정사항에 대하여 쉽게 수정을 할 수 있다.

동기화는 현재 급속하게 성장하고 있는 응용기술 및 장비와 함께 표준 동기화 프로토콜 기술이 제공되어진다면 개인이 소유하고 다양한 정보기기 및 네트워크에 존재하는 수많은 종류의 정보를 장비에 상관없이 접속하여 최신의 정보를 주기적으로 업데이트 시켜주기 때문에 개인 PC, PDA, PCS, 휴대폰 , 웹패드 , 스마트폰 , 등 다양한 차세대 이동통신 기술에 대한 막대한 규모의 중복투자를 방지할 수 있다. 예를 들어 웹상의 주소록을 갱신하면 실시간으로 휴대폰 , PDA 등의 단말기안에 보관되어 있는 주소록도 자동으로 갱신되기 때문에 소비자들은 어떠한 기기에서도 동일한 데이터를 관리할 수 있는 장점을 가지고 있기 때문에 지금 무선 인프라에서 가장 필요한 기반 기술이라고 할수 있겠다.

본 논문이 제시한 서버 및 Sync Engine 프레임은 사용자에게는 어느 디바이스, 어느 서버든 쉽게 사용할수 있고, 개발자에게는 쉽고 빠른 서비스의 확장 및 정책에대한 반영을 할 수 있다는 장점을 가지고 있다. 향후 연구 과제로는 서비스의 추가등에 대해 코드를 직접수정하지 않고 자동으로 추가할 수 있는 엔진 라이브러리를 만들어 빠르고 쉽게 엔진을 확장하게 하는 방법을 고려할 수 있다.

[참고 문헌]

[1] ARC Group(1999. 5), Wireless internet application, technology & strategies.

[2] Forrester Research(2000. 5), The Mobile internet gold rush is on.

[3] Durlacher(1999. 11), Mobile commerce report.

[4] syncML.org, http://www.syncML.org

[5] SyncML Device information DTD, SyncML

[6] SyncML Meta information DTD, SyncML

[7] SyncML Synchronization Protocol, SyncML

[8] Extensible Markup Langage (XML) 1.0, W3C

[9] SyncML White paper, SyncML

[10] SyncML protocol, SyncML