OMA DM 기반의 무선 단말기 소프트웨어 배포 관리 시스템 ThinkSync DM-SoftMan 개발

주 홍 택[†] · 박 기 현[††] · 방 대 욱[†††]

요 약

무선이동통신 단말기의 기능 및 서비스가 복잡해지면서 단말기의 소프트웨어도 복잡해져서 관리의 대상이 되었다. 통신, 방송이 융합되고 텔레매틱스나 홈네트워크가 활성화되면 단말기 소프트웨어는 더욱 복잡해지고 관리의 필요성은 증가할 것으로 예상된다. 단말기 관리에 관한 세계적인 표준으로서 많이 적용되고 앞으로 더욱 확산될 것으로 예상되는 기술이 바로 OMA DM이다. 본 논문에서는 OMA DM 표준을 기반으로 무선단말기 관리 응용으로서 무선이동통신 단말기의 소프트웨어를 관리하는 ThinkSync DM SoftMan의 개발 결과를 제시한다. 본 논문에 앞서 실시한 ThinkSync DM SoftMan의 설계결과를 요약하고 설계에 따라서 구현한 결과와 시험을 실시한 결과를 제시한다.

키워드 : OMA DM, 소프트웨어 배포 관리, ThinkSync DM-SoftMan

Software Release Management System : ThinkSync DM-SoftMan for Wireless Device based on OMA DM

HongTaek Ju[†] · KeeHyun Park[††] · Dae Wook Bang[†††]

ABSTRACT

There has been a continued increase in the complexity of software equipped with wireless mobile devices, due to the introduction of new device functionality and services via network connection. The increasement expected to be accelerated by convergence of telecommunication and broadcasting, and proliferation of telematics and home networking services using wireless mobile devices. The higher the complexity of mobile device software, the higher the necessity of management for the software. As for the global standard of mobile device management technology, OMA DM has been widely adopted by device manufacture and expected to be accelerated its adoption. In this paper, we present a development result of mobile device software release management system ThinkSync DM SoftMan. The implementation details of ThinkSync DM SoftMan are provided in implementation architecture and its working scenario based on the design of ThinkSync DM SoftMan that is summarized in this paper as our previous work. The conformance and performance test of the system are presented.

Key Words : OMA DM, Software Release Management, ThinkSync DM-SoftMan

1. 서 론

무선이동통신 단말기는 단말기 자체뿐만 아니라 네트워크 연결을 통해서 다양한 기능과 서비스를 제공할 수 있게 발전하였다. 이러한 발전은 많은 수의 다양한 소프트웨어가 단말기에 탑재됨으로써 가능하게 되었다. 단말기에 탑재되는 소프트웨어의 수가 증가할수록 단말기의 안정적이고 편리한 사용을 위하여 단말기 소프트웨어 관리는 필수적이다. 더욱이 방송, 통신이 융합되고 텔레매틱스, 홈네트워크와 같은 다양한 응용 분야에 단말기가 적용될 예정이므로 무선통신단말기의 기능과 서비스는 더욱 고도화되고 이에 따라서 단말기 소프트웨어가 더욱 복잡해 질 것이며 단말기 소프트웨어에 대한 관리의 필요성은 더욱 증가될 것이다. 단말기 소프트웨어 관리는 사용자가 아니라 중앙의 단말기 관리 시스템이 단말기 소프트웨어의 고장을 복구하고 단말기의 사용 환경 변화에 적응시키거나 성능 및 기능 개선 그리고 새로운 서비스 개시하기 위하여 새로운 소프트웨어를 단말기에 설치, 실행 시킬 수 있다. 또한 불필요한 소프트웨어를 제거하거나 정상동작 여부를 감시할 수도 있다[1].

현재 단말기 소프트웨어 관리는 사용자들 직적 관리를 수행하는 초보적인 방법에 주로 의존하고 있다[8, 9]. 즉 사용자가 직접 소프트웨어를 설치하고 새로운 버전으로 업데이

※ 이 논문은 산업자원부 지방기술혁신사업(RT104-03-02) 연구비를 지원받았음.
† 정 회 원 : 계명대학교 컴퓨터공학과 조교수
†† 종신회원 : 계명대학교 컴퓨터공학부 교수
††† 정 회 원 : 계명대학교 컴퓨터공학과 교수
논문접수 : 2006년 4월 13일, 심사완료 : 2006년 8월 4일

트하며 불필요한 소프트웨어를 제거한다. 이 방법보다 개선된 방법은 단말기에 탑재되는 각 소프트웨어 별로 별도의 관리 방법을 내재하여 관리하는 방법이다[2,4,6,7]. 그러나 이 방법도 역시 설치나 오류가 발견된 경우에 대해서는 수작업에 의존할 수밖에 없다. 더욱이 각 프로그램 별로 소프트웨어 관리 방법을 내재해야 하므로 전체 프로그램의 크기가 증가한다. 그리고 사용상의 일관성 부족으로 사용이 불편하며 소프트웨어간의 의존성을 고려할 수 없다. 따라서 소프트웨어 회사마다 관리 서버를 운용해야 하므로 관리비용이 증가된다.

자동화된 소프트웨어 관리 방법을 사용하면 다음과 같은 장점이 있다. 첫째 중앙의 소프트웨어 관리 시스템이 소프트웨어를 관리하기 때문에 사용자의 편리성이 증대된다[2,3,7,8,9]. 둘째 소프트웨어 업데이트가 필요한 시점에 즉각적으로 소프트웨어를 갱신하여 관리 주기를 단축할 수 있다. 여기서 관리 주기란, 소프트웨어를 한번 업데이트하고 난 후 다음 번 업데이트까지의 기간을 의미한다. 관리 주기가 단축되면 안정적인 소프트웨어 관리를 수행 할 수 있다.

이와 같이 안정적인 소프트웨어의 사용과 관리를 위해서는 중앙에서 소프트웨어를 자동으로 관리해 주는 소프트웨어 관리 시스템이 필요하다. 소프트웨어 관리는 단말기 관리의 한 응용 분야이다. 즉 단말기가 안정적이고 만족할만한 성능으로 사용하기 위해서 중앙에서 관리하는 단말기 관리의 한 분야이다. 현재 단말기 관리를 위한 표준으로 OMA (Open Mobile Alliance)에서 제시한 DM (Device Management)가 있다. 간단히 OMA DM으로 불리는 이 표준은 현재 단말기 관리에 적용되기 시작했으며 거의 유일한 산업계 표준으로 받아들여지고 있다. 본 논문에서는 단말기 관리 표준인 OMA DM을 기반으로 단말기 소프트웨어를 관리하기 위한 단말기 소프트웨어 관리 시스템, ThinkSync DM-SoftMan 개발 결과를 제시한다. 우리는 이전 논문에서 ThinkSync DM을 개발한 결과를 제시하였다[7]. 이를 기반으로 단말기의 소프트웨어를 관리하는 단말기 소프트웨어 관리 시스템으로 확장하였다.

ThinkSync DM-SoftMan은 매니저와 에이전트로 구성된다. 매니저는 중앙에서 운영자의 지시를 받아서 단말기에 소프트웨어를 설치하고 삭제하는 등 단말기 소프트웨어 관리를 주도적으로 실행하는 프로그램이다. 에이전트는 단말기에 설치되는 프로그램으로서 매니저의 관리 요청을 받아서 실행해주는 프로그램이다. 관리 요청은 소프트웨어 설치, 삭제 등을 포함한다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구 부분으로서, OMA DM에 대한 소개, 기존의 OMA DM 기반의 단말기 관리 시스템 개발 및 기존의 소프트웨어 배포 관리 연구를 소개하였고, 이 기술들을 ThinkSync DM-SoftMan에 활용 가능한 방법에 대해 분석하였다. 3장에서는 이 논문에 앞서 실시된 연구결과로 ThinkSync DM-SoftMan 설계를 요약 제시한다. 4장에서는 ThinkSync DM-SoftMan 구현 및 검증 결과를 제시하고 5장에서는 시험결과를 제시한다. 마지막으로 6장은 본 논문의 결론과 향후 연구 방향에 대해서 논의한다.

2. 관련 연구

이 장에서 기존의 소프트웨어 관리 방법을 요약하고 ThinkSync DM-SoftMan은 OMA DM을 응용하여 개발되었으므로, OMA DM 규격을 소개한다. 그리고 OMA DM 규격을 바탕으로 개발된 시스템과 상용제품을 소개한다.

2.1 무선단말기 소프트웨어 관리 방법들

WinCE 운영체제가 탑재된 PDA의 응용 소프트웨어 설치는 기존의 Windows 환경의 소프트웨어 설치 방법과 동일한 형태로 수행된다[19]. 이 방법은 데스크 탑 PC와 동일한 설치 방법을 제공함으로써 PDA 만을 위한 사용자의 부가작업을 줄였다는 점이다. 하지만, 소프트웨어의 계속적인 업데이트 작업의 자동화는 지원하지 않고 있으며, 또한 소프트웨어 설치를 위한 웹 검색 및 다운로딩 작업 역시 사용자가 직접 수행하여야 한다. 결국 윈도의 소프트웨어 관리 방법을 PC를 통하여 이루어지므로 기술적으로 동일하다고 볼 수 있다.

Palm 운영체제가 탑재된 PDA에서는 PC와의 연결을 손쉽게 할 수 있는 인터페이스를 제공한다[22]. 하지만, 소프트웨어의 지속적인 업데이트 작업의 자동화는 지원하지 않고 있다. 이로 인해 소프트웨어의 업데이트 작업이 필요할 경우 사용자가 직접 수행한다. 소프트웨어 결함이 생길 경우 자동으로 업데이트 작업이 이루어지지 않기 때문에 단말기를 사용하지 못하거나 서비스를 사용할 수 없다. Palm에서는 API를 공개하고 있다. 공개된 Palm API를 통해 각 소프트웨어 관리 절차들을 도출하는데 참고가 되었다.

2.2 소프트웨어 패키징 방법들

소프트웨어를 관리하기 위해서는 설치, 삭제, 업데이트를 위한 소프트웨어 패키지가 필요하고, 여기서 소개되는 두 가지 방법은 가장 많이 사용되는 소프트웨어 패키지 방법이다. 본 논문에서는 소프트웨어 패키징 방법으로 RPM을 사용한다.

RPM은 리눅스에 응용프로그램을 추가 및 삭제하는 과정이 무척 번거롭다는 이용자들의 불만을 고려해 도입된 방식이다[15]. RPM은 설치된 패키지와 파일이 저장되어 있는 데이터베이스(Database)를 유지하기 때문에 설치된 소프트웨어 현황에 대한 강력한 질의와 검증을 실행할 수 있다. 그리고 패키지 설치 시 의존성 추적으로 설치를 위한 요구조건의 검증을 할 수 있다. 또한 소프트웨어 제거 시 제거될 소프트웨어의 사용 현황 파악도 가능하여 다른 소프트웨어에 영향을 주는 소프트웨어의 제거를 방지할 수 있다.

마이크로소프트사에서 만든 MSI는 윈도 설치 관리자이면서 윈도에서 프로그램을 설치할 때 지켜야 할 표준이다[16]. MSI에는 파일명, 레지스트리 키, 바로 가기 아이콘 및 애플

리케이션 설치 요구에 관한 정보가 들어있다. MSI 표준을 준수하는 배포 파일은 설치, 관리 및 삭제가 용이하다. 본 논문에서는 RPM을 사용하고 있으나 향후 WinCE로 확장하기 위해서는 MSI도 ThinkSync DM-SoftMan의 패키지 구성 방법으로 활용해야 할 것이다. 그러나 현재까지는 본 논문에서는 이를 연구 범위에 포함시키지는 못하였다.

2.3 OMA DM 표준

단말기 관리 프로토콜은 관리 매니저와 관리 에이전트들 간에 전달되는 관리 명령에 대한 규약이다[10,13]. 관리 명령은 메시지로 표현되며 메시지 형식과 절차에 대해 기술하고 있다. 단말기 관리 프로토콜은 두 가지의 기능으로 나눌 수 있다. 첫 번째, 관리 대상이 되는 관리 객체를 조작하기 위해서 사용되는 기능이다. 매니저는 관리 객체의 정보를 읽고, 새로운 내용을 추가, 업데이트 혹은 삭제하기 위해 에이전트에게 관리 명령을 지시한다. 관리 명령을 받은 에이전트는 매니저가 전달한 관리 명령을 관리 객체를 조작함으로서 관리를 수행한다. 두 번째는 단말기 사용자와의 접속을 가능하게 하는 기능이다. 관리 행위에 대한 정보를 사용자에게 제공하거나 관리 행위에 대한 확인을 사용자로부터 요청 받는다.

단말기 관리 객체 표준은 단말기의 구성 요소이면서 관리의 대상이 되는 요소들을 추상화하여 관리 객체로 표현하는 방법을 제시하고 있다[11,12]. 관리 객체는 트리 구조로 되어 있으며, 트리의 단말 노드가 하나의 관리 대상을 대표한다. 단말기 관리 서비스를 제공하는 매니저는 단말기 관리 세션을 통해 에이전트 단말기 내의 관리 객체들에 대한 트리 구조를 탐색하거나, 새로운 객체를 추가 혹은 업데이트할 수 있다. 이렇게 관리 객체의 트리를 조작하여 관리 대상을 원하는 상태로 만들 수 있다. 본 논문에서는 관리 대상이 되는 소프트웨어 정보를 단말기 관리 객체 표준을 바탕으로 관리 객체를 정의하였다. 그리고 관리 트리를 통해 소프트웨어를 관리하기 위한 방식을 ThinkSync DM-SoftMan에 활용하기 위해 제시하였다.

2.4 OMA DM 개발 사례

Sync4j 단말기 관리 시스템은 OMA DM 소스가 공개된 공개 프로젝트이다[3]. 단말기 관리는 중앙에서 Sync4j DM 매니저에 의해 수행 된다[8, 9]. Sync4j DM은 OMA DM 1.1을 바탕으로 개발되었다. Sync4j DM은 단말기 관리에 관한 기본 통신 기능만을 제공하고 있으며 이를 실제 단말기 관리에 활용하기에는 아직 초보 단계에 있다.

INRIA 연구소의 MADYNES팀에서 개발한 단말기 관리 에이전트 시스템은 소스가 공개된 공개 프로젝트이며 단말기 관리 에이전트 시스템의 명칭은 MADMAX이다. 현재는 단말기관리 에이전트의 개발 결과와 확장 방법을 제시하고 있다[2]. MADMAX도 Sync4j와 마찬가지로 OMA DM의 기본 기능만을 제공하는 초보 단계에 있다. 본 논문에서 제시하는 소프트웨어 관리 방법을 Sync4j나 MADMAX에 적용하여 소프트웨어 관리 시스템으로 확장할 수 있다.

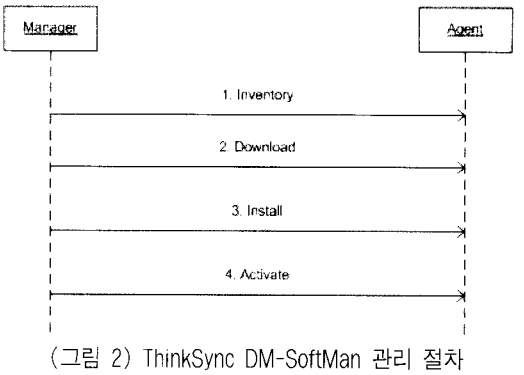
3. ThinkSync DM-SoftMan 설계

ThinkSync DM-SoftMan은 중앙에서 여러 개의 단말기 소프트웨어를 관리하는 시스템이다. 중앙에서 실행되는 프로그램이 관리 매니저이고 단말기에 설치된 프로그램이 관리 에이전트이다. 운영자는 매니저 프로그램을 이용해서 단말기에 새로운 소프트웨어를 설치하거나 기존에 배포된 소프트웨어를 업데이트 할 수 있다. 매니저는 이와 같은 관리를 위해서 소프트웨어 배포 관리에 필요한 관리 명령을 생성하여, 에이전트에게 관리 명령을 전달한다. 또한 관리명령을 이용하여 원격 제어가 가능하기 때문에 소프트웨어의 오류가 발생하여 단말기로부터 관리 요청을 받으면 사용자들을 대신하여 소프트웨어 오류를 복구한다.

(그림 1) ThinkSync DM-SoftMan 전체 구조

3.1 소프트웨어 관리 절차

이 절은 ThinkSync DM-SoftMan에 소프트웨어 배포 관리를 위해서 매니저와 에이전트 간에 이루어지는 관리 행위 절차에 대해 설명한다. 아래의 (그림 2)는 ThinkSync DM-SoftMan의 관리 절차의 전체 세션(Session) 과정을 나타낸다. 각 세션이 시작될 때마다 새로운 네트워크 연결하여 인증 절차를 거친다. 즉, 논리적으로 물리적으로 세션은 완전한 통신 단위마다, 그리고 각 세션 안에서 여러 개의 관리 메시지를 교환한다.



(그림 2) ThinkSync DM-SoftMan 관리 절차

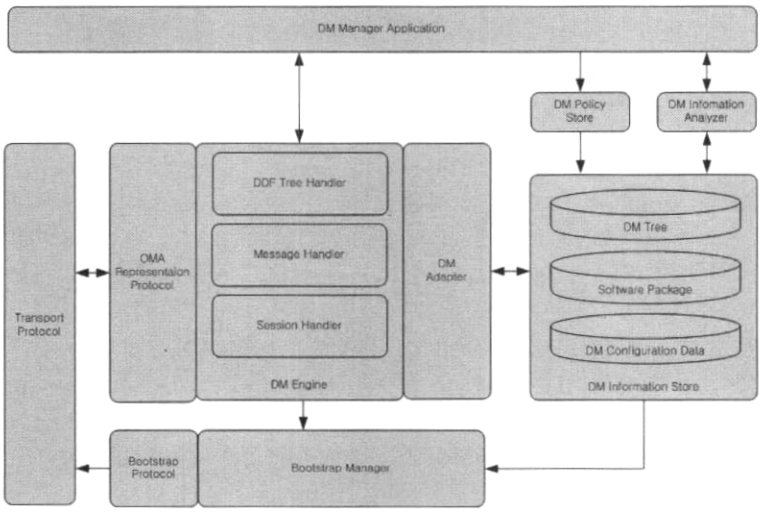
ThinkSync DM-SoftMan의 세션은 기존의 소프트웨어 배포 관리 절차에 따른 행위를 분석하여 Inventory, Download, Install, Activate 세션으로 나누었다. Inventory 세션에서는 다양한 형태의 소프트웨어들을 설치하기 전에 기존에 설치된 패키지들의 정보를 의존성 검사와 버전 확인을 위한 작업을 수행한다. Download 세션에서는 매니저 측에서 의존성 문제를 해결되고 가장 최신의 소프트웨어를 설치 혹은 업데이트하기 위해 에이전트에게 소프트웨어 패키지를 전달하는 과정이다. 이 과정은 매니저가 에이전트에게 소프트웨어를 전달만 하고 설치하지는 않는다. Install 세션에서는 매니저로부터 전달받은 소프트웨어를 시스템에 설치하는 과정이다. 그러나 이 과정에서는 단지 소프트웨어의 설치만 수행할 뿐 실제 시스템 상에서 서비스를 받을 수 있도록 소프트웨어를 활성화 시키진 않는다. Activate 세션에서는 에이전트에 의해 설치된 소프트웨어를 단말기에 활성화 시키는 과정이다. 이 과정을 통해서 설치된 소프트웨어가 실행된다. 그리고 Download, Install, Activate 세션에서의 관리 작업이 완료되면 에이전트는 매니저에게 완료 결과를 돌려준다.

3.2 ThinkSync DM-SoftMan 구조 설계

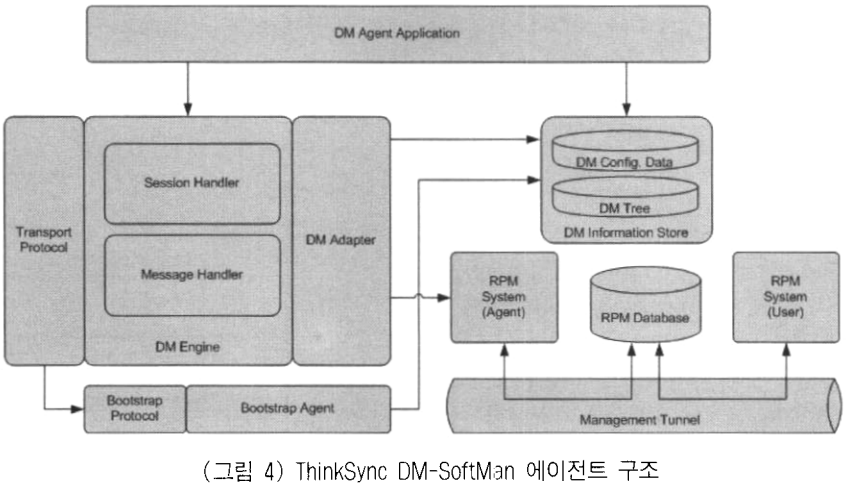
이 절은 소프트웨어 배포 관리를 위한 ThinkSync DM-SoftMan 매니저 모듈의 기능과 상호 연관성 및 매니저 구조 설계에 대해서 설명한다.

(그림 3)은 ThinkSync DM-SoftMan 매니저의 구조를 보여주고 있으며 크게 5개의 구성요소로 이루어져 있다. 운영자와 접속하기 위해 필요한 것이 DM Manager Application이다. OMA DM 프로토콜을 구현한 결과가 DM Engine이며 이는 관리 객체를 처리하는 DDF Tree Handler, 하나의 관리 명령을 처리하는 Message Handler와 관리 세션을 위한 Session Handler로 구성되어 있다. 단말기 초기 설정을 위해서 필요한 Bootstrap Manager는 Bootstrap Protocol을 통해 수행하며, 매니저와 에이전트 사이의 메시지 전송을 위한 Transport Protocol로 구성된다. 그리고 DM Information Store는 관리 객체를 저장하는 DM Tree, 설치되는 소프트웨어를 조작하는 Software Package와 매니저의 설정 값을 저장하는 DM Configuration Data로 구성된다.

ThinkSync DM-SoftMan 에이전트는 6개의 큰 구성요소로 이루어져 있으며 (그림 4)에 나타나 있다. 관리 명령을



(그림 3) ThinkSync DM-SoftMan 매니저 구조



(그림 4) ThinkSync DM-SoftMan 에이전트 구조

실행하기 위해 필요한 DM Agent Application, 단말기 초기 설정을 위해서 필요한 Bootstrap Agent, 매니저와 에이전트 사이의 메시지 전송을 위한 Transport Protocol 구성된다. 그리고 OMA DM을 구현한 결과인 DM Engine은 Session Handler와 Message Handler로 구성된다. 단말기의 관리 트리 정보, 구성 정보 및 소프트웨어의 저장을 위한 DM Information Store와 RPM 명령을 처리하기 위한 RPM System으로 구성된다. (그림 4)는 ThinkSync DM-SoftMan 에이전트의 구조를 나타낸다.

3.3 소프트웨어 관리 객체

(그림 5)는 소프트웨어를 관리하기 위해 필요한 관리 객체들을 객체 구조로 나타낸 DDF 스키마 구조이다. 제시한 객체 구조는 OMA DM에서 소프트웨어 관리를 위해서 제시된 객체를 그대로 사용하였으며 일부 필요에 따라서 확장하였다[16]. Inventory 엘리먼트는 소프트웨어 설치 혹은 업데이트 시 설치 된 소프트웨어의 정보를 확인하기 위한 기본 정보로서 Delivered와 Deployed인 하위 엘리먼트가 있다. Delivered는 매니저로부터 에이전트에게 전달된 소프트웨어에 대한 정보만을 저장하는 노드이다. Deployed는 전달된 패키지를 시스템에서 활성화 시킨 후의 정보를 저장한다.

전달된 패키지가 활성화 되었다면 Delivered에서 저장한 패키지의 정보를 삭제한 후 Deployed에 추가되어야만 한다. Download 엘리먼트는 패키지를 다운로드하기 위해 필요한 정보를 저장하며, URI에 의해서 식별되어야 한다. Operation 엘리먼트는 전달되었거나 현재 실행중인 소프트웨어에 대한 현재의 상태를 변경하기 위해 사용한다. 이와 같은, (그림 5)의 DDF 스키마를 이용하여 ThinkSync DM-SoftMan은 소프트웨어 관리를 수행할 수 있다.

4. ThinkSync DM-SoftMan 구현

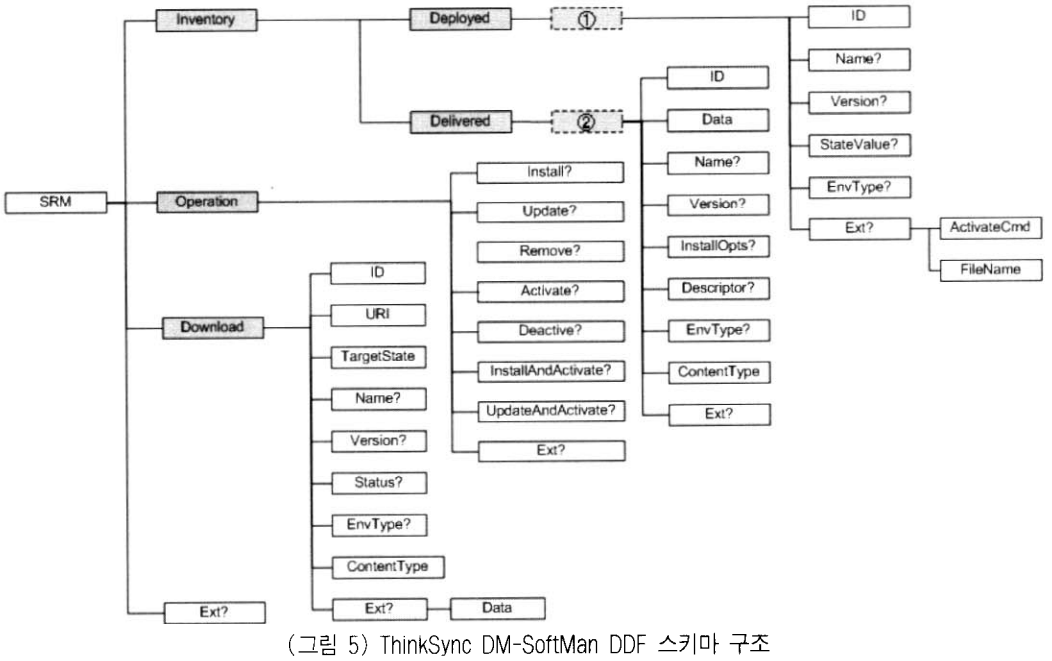
4.1 매니저 구현 구조

이 절은 소프트웨어 배포 관리를 위한 ThinkSync DM-SoftMan 매니저의 설계를 바탕으로 적용된 각 모듈의 기능과 상호 연관성 및 매니저 구현 구조에 대해서 설명한다. (그림 6)은 리눅스 기반에서 개발한 ThinkSync DM-SoftMan 매니저의 구현 구조를 나타낸다.

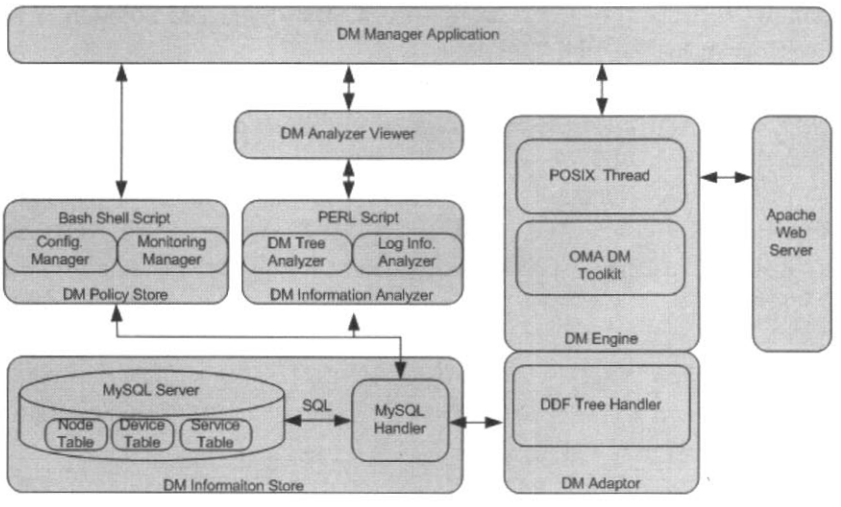
ThinkSync DM-SoftMan은 HTTP 프로토콜을 사용하는 아파치 웹 서버를 사용하였다. Apache Web Server는 현재 가장 많이 사용되고 있는 웹 서버이고 필요한 기능을 모두 제공하고 있으므로 ThinkSync DM-SoftMan에 적합한 웹 서버이다.

관리 정보에 대한 메시지 구성과 처리를 결정하는 DM Engine은 성능적인 면을 고려하여 C++로 구현하였다. 또한 DM Engine은 POSIX Thread와 OMA DM Toolkit으로 구성된다. DM Adaptor는 트리로 표현한 관리 대상의 관리 정보를 처리하시 위해 C 언어를 이용하여 구현하였으며, DDF Tree Handler를 포함하고 있다. DDF Tree Handler는 C++로 구현되어 있다. DDF Tree Handler를 통한 소프트웨어 패키지 관리 정보는 XML 트리형태로 구성하며, DM Engine으로부터 넘겨받은 관리 정보를 MySQL 핸들러에 전달한다.

DM Information Analyzer, DM Policy Stores는 MySQL Handler에게 데이터베이스와의 연산을 요구한다. MySQL Handler는 MySQL에서 제공하는 API(Application Programming Interface)를 사용하여 구현하였다. MySQL Server는 노드의 정보를 저장하는 Node 테이블(Table), 단말기의 구성 정보를 저장하는 Device 테이블, 그리고 RPM



(그림 5) ThinkSync DM-SoftMan DDF 스키마 구조



(그림 6) ThinkSync DM-SoftMan 구현 구조

(그림 7) 매니저 데이터베이스 스키마

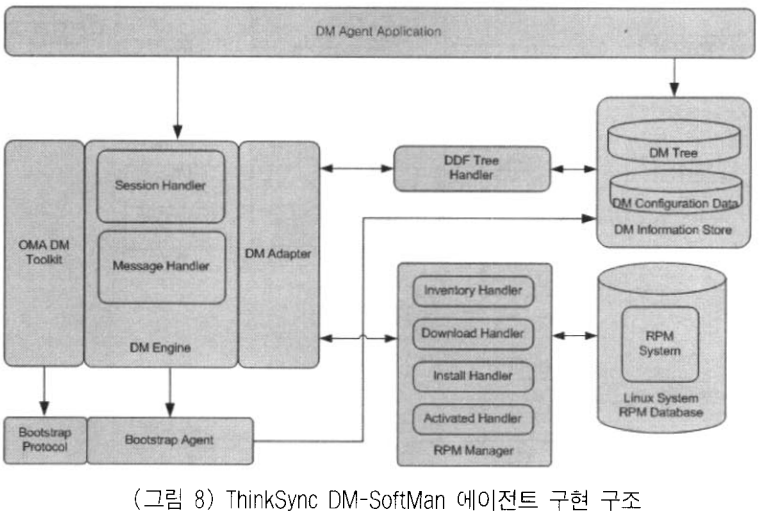
및 다양한 서비스의 확장을 위해 필요한 정보를 저장하는 Service 테이블로 구성되어 있다. (그림 7)은 매니저 데이터베이스 스키마를 나타낸 그림이다.

NodeInfo 테이블과 Service 테이블은 Device 테이블의 DevId를 참조하여 각 단말기 별로 서비스와 단말기 구성 정보를 관리한다. 이와 같은 구성을 바탕으로 하여, 운영자는 리눅스 기반의 텍스트 환경에서 작동하는 DM Manager Application을 통해 매니저와의 관리를 수행한다.

4.2 에이전트 구현 구조

이 절은 소프트웨어 배포 관리를 위한 ThinkSync DM-SoftMan 에이전트의 설계를 바탕으로 적용된 각 모듈의 기능과 상호 연관성 및 에이전트 구현 구조에 대해서 설명한다. (그림 8)은 리눅스 기반에서 개발한 ThinkSync DM-SoftMan 에이전트의 구현 구조를 나타낸다.

본 연구에서 ThinkSync DM-SoftMan 에이전트는 OMA DM Toolkit에서 제공하는 HTTP 프로토콜을 사용하였다. 관리 정보에 대한 메시지 구성과 처리를 결정하는 DM Engine은 C++로 구현되어 있다. 또한 DM Engine의 Message Handler와 Session Handler는 OMA DM Toolkit을 사용하여 OMA DM 메시지를 생성하고 세션연결과 유지를 위한 작업을 수행한다. 에이전트 측에서도 OMA DM



(그림 8) ThinkSync DM-SoftMan 에이전트 구현 구조

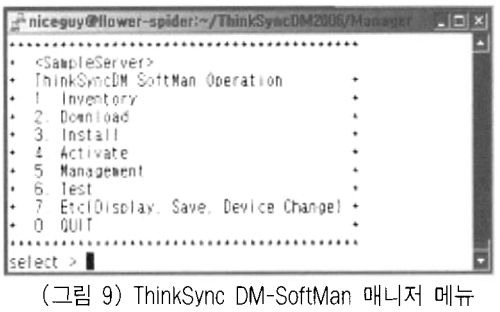
Toolkit을 그대로 사용하여 구현되었다. 이와 같이 OMA DM Toolkit에서 제공하는 함수들을 이용하여 DM Engine은 관리 정보에 대한 메시지 구성과 처리를 결정한다. DM Adaptor는 C++를 이용하여 구현하였으며, 트리로 표현한 관리 대상의 관리 정보를 처리하는 DDF Tree Handler와 RPM 패키지들을 처리하기 위한 RPM Manager와 연동을 통해 소프트웨어 및 단말기 관리에 관한 정보를 저장한다. DDF Tree Handler는 C++로 구현되어 있다. 잦은 트리의 변경 작업, 즉, 삽입, 삭제, 업데이트 및 실행 작업으로 인해 다양한 확장성을 요구한다. 그리고 DDF Tree Handler를 통한 소프트웨어 패키지 관리 정보는 XML 트리형태로 구성되며, DM Information Store에 저장된다.

RPM 패키지를 처리하기 위한 RPM Manager는 C++로 구현되어 있다. RPM Manager는 에이전트에게 새로운 소프트웨어 관리 기능이 추가 될 경우 추가 변경작업 없이 새로운 서비스를 적용하는 기능을 하도록 구성되어 있다. 이와 같은 구성을 바탕으로 하여, 사용자는 리눅스 기반의 텍스트 환경에서 작동하는 DM Agent Application을 통해 소프트웨어 관리를 요청한다.

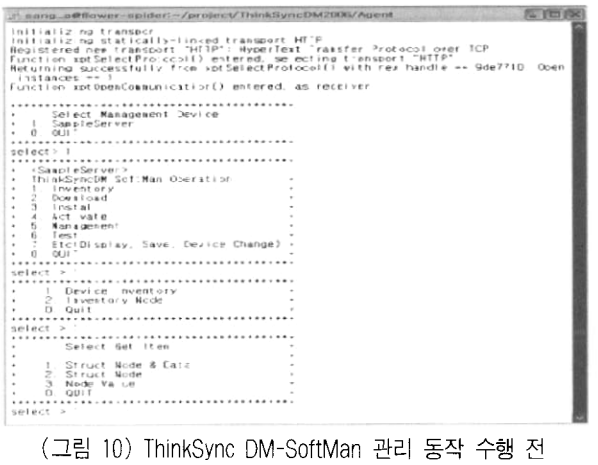
4.3 구현 결과

이 절은 ThinkSync DM-SoftMan을 이용하여 실제 소프트웨어의 설치 시 실행되는 과정에 대한 결과를 제시한다. (그림 9)는 ThinkSync DM-SoftMan의 매니저가 제공하는 운영자 메뉴이다.

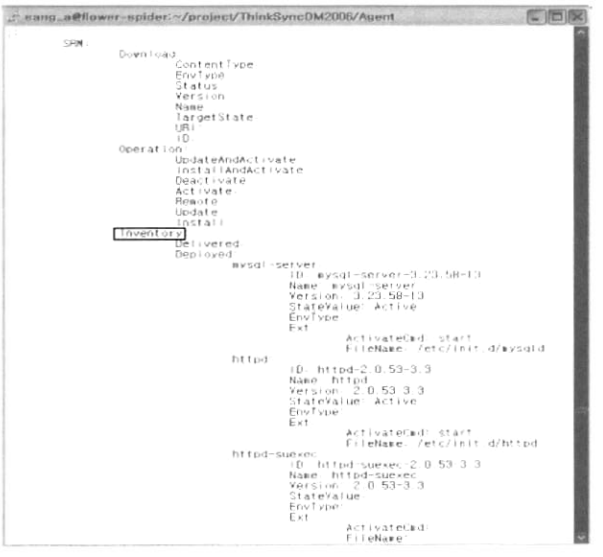
(그림 10)은 ThinkSync DM-SoftMan의 관리 동작 수행



(그림 9) ThinkSync DM-SoftMan 매니저 메뉴



(그림 10) ThinkSync DM-SoftMan 관리 동작 수행 전



(그림 11) Inventory 세션 후 소프트웨어 관리 트리

전의 그림이다. 매니저는 관리할 단말기 선택 후 소프트웨어 관리 세션을 결정한다. 결정된 관리 세션은 Inventory이다. Inventory 세션의 메뉴는 매니저가 에이전트에게 에이전트 관리 트리에 대한 전체 정보를 가져오기 위해 Device Inventory와 다운로드 되거나 설치 혹은 실행 중인 소프트웨어의 정보를 가져오는 Inventory Node로 구성된다. (그림 11)의 Inventory 노드의 하위 노드에 대한 관리 정보를 가져오기 위해 Inventory node를 선택한다. 그리고 하위 노드의 소프트웨어 정보를 가져오기 위한 방법은 3가지의 메뉴로 구성된다. Struct Node & Data는 하위 노드에 대한 노드의 속성 및 관리 정보를 모두 가져온다. Struct Node는 하위 노드의 노드 이름만 가져온다. 그리고 Node Value는 특정 노드에 대한 관리 정보만을 가져온다. Inventory 노드의 하위 노드에 존재하는 모든 정보를 가져오기 위해 Struct Node & Data를 선택한다.

Inventory 세션을 위한 설정을 완료한 후 에이전트와 소프트웨어 관리 동작을 수행한 후 그 결과를 매니저는 관리 트리에 적용하여 나타내고 있다. (그림 11)은 Inventory 세션의 결과를 매니저 소프트웨어 관리 트리에 적용한 결과를 나타낸다.

Inventory 노드는 Delivered와 Deployed로 구성된다. 소프트웨어가 전달되고 난 후 설치 혹은 삭제가 되지 않은 경우에는 Delivered에 나타나고, Delivered의 하위노드가 없을 경우, 에이전트에게 전달되었으나 설치되지 않은 소프트웨어가 없음을 나타낸다. 소프트웨어가 실행되거나 설치 된 경우 Deployed의 하위 노드에 나타나게 된다. Deployed 노드에는 mysql-server, httpd, 그리고 httpd-suexec등 소프트웨어의 정보가 나타나고 있다. 소프트웨어의 정보는 ID, Name, Version, StateValue, EnvType, Ext/ActivateCmd, 그리고 Filename로 구성된다. (그림 12)는 Inventory 세션 수행 후 Inventory 노드의 하위 노드 정보 중에서 3개의 패키지에 대한 설치 정보만 보이고 있다.

5. 구현 결과 평가

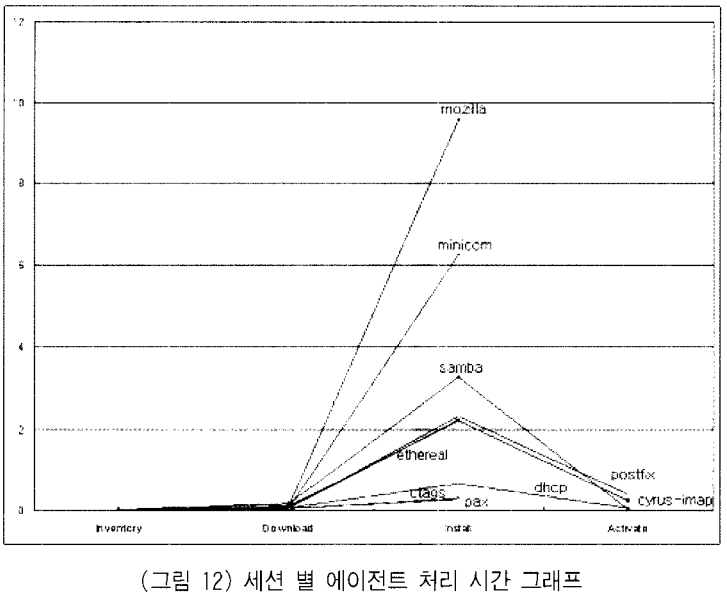
시스템 전체적인 측면에서 구현결과에 대하여 평가한다. 첫째 본 논문에서는 OMA DM 1.1을 사용하여 매니저와 에이전트를 개발하였다. 구현된 매니저와 에이전트가 OMA DM 1.1에 적합하게 동작하는지를 확인하기 위해서 SCTS (SyncML Conformance Test Suite) 툴킷과 적합성 시험을 수행하였다. SCTS와 에이전트 적합성 검증 항목은 해당하는 20개의 그룹 및 총 44개의 항목이 있다. 이 중 필수 지원 항목의 경우 13개의 그룹 및 33의 항목이 있고, 선택 지원 항목의 경우 7개의 그룹 및 11개의 항목이 있다. 본 논문에서는 필수지원 항목과 선택지원 항목 모두 적합성 검증을 수행하였고 수행 결과 모두 합격결과를 보였다.

둘째 매니저와 에이전트 간에 주고받는 메시지는 전송되는 패킷의 크기를 최대한 줄이고 효율적인 전송을 위해서 WBXML을 사용하여 구현되었다. 그리고 전체 관리 동작을 세션으로 분리하여 각 세션에서 통지 기능을 활용하였다. 통지 기능은 에이전트에서 특정작업을 완료했음을 알리는 것이다. 이렇게 함으로써 매니저가 폴링 하는 메시지를 삭제하였다. 또한 패키지를 ftp로 전달함으로서 네트워크를 효율적으로 사용하였다. OMA DM 프로토콜로 패키지를 전달할 수도 있으나 OMA DM 프로토콜은 ftp보다 오버헤드가 크다.

셋째 사용자와의 확인 작업을 통해 다음 관리 동작을 결정하는 User Interaction Alert 기능을 구현하였다. 예를 들어 기존 단말기에 설치 된 소프트웨어를 업데이트할 경우 새로운 소프트웨어는 기존의 설치 된 소프트웨어와 호환성의 문제가 발생할 수 있다. 이 경우 중앙 운영자는 사용자에게 업데이트에 관련된 사항들을 전송하고 사용자는 업데이트 여부를 결정하게 된다. 사용자에 의해 결정된 결과를 바탕으로 운영자는 다음 소프트웨어 관리를 결정한다.

다음은 매니저 측면에서 구현 결과에 대해 평가한다. 첫째 매니저를 사용해서 소프트웨어를 직접 설치해 본 결과 필요한 모든 기능을 모두 제공하였다. 기능의 검증은 SCTS와의 적합성 검증을 통해 입증하였다. 둘째 매니저는 POSIX에서 표준으로 제안한 쓰레드를 사용하여 구현하였다. 에이전트에 소프트웨어 관리를 수행할 경우 새로운 쓰레드를 생성하여 각각의 단말기를 동시에 관리한다. 쓰레드의 장점은 다음과 같다. 첫째 응용 프로그램이 긴 작업을 수행하더라고 프로그램의 수행이 계속되는 것을 허용함으로서 사용자에 대한 응답성을 증가시킨다. 예를 들면, 다중 쓰레드 매니저는 한 에이전트의 관리 세션을 수행하고 있다. 수행 중 다른 에이전트의 관리 요청이 있으면 다른 에이전트의 관리 요청 또한 동시에 수행한다. 둘째 프로세스의 자원들과 메모리를 공유한다. 예를 들어 다수의 에이전트에서 관리 요청을 할 경우 에이전트는 서버의 정보가 저장된 메모리에서 서버 정보를 공유하며 사용할 수 있다. 단말기에 설치된 다른 프로그램에 영향을 주지 않고 자원을 효율적으로 사용할 수 있다.

마지막으로 에이전트 측면에서 구현결과를 평가한다. 첫째 소프트웨어 관리 방법으로 RPM을 사용하였다. (그림 11)는 에이전트 측에서 RPM과의 연동을 통해 리눅스 시스템에 설치 된 패키지들을 에이전트 관리 트리에 적용한 결과를 나타낸다. RPM은 소프트웨어의 설치, 삭제, 검증 및 업데이트 기능을 제공한다. ThinkSync DM-SoftMan에서는 RPM에서 제공하는 기능뿐만 아니라 스케줄까지도 추가하여 소프트웨어 관리를 위해 지원한다. 둘째 설치 시 소프트웨어의 의존성을 점검할 수 있어야 한다. Inventory 세션을 통해 의존성 문제를 해결할 수 있다. Inventory 세션에서는 에이전트가 소프트웨어 설치 시 발생하는 의존성 문제에 관해 매니저에게 보고하고, 매니저는 이 결과를 중앙 운영자에게 보여 줌으로서 의존성 문제를 해결하게 되어 있다. 셋째 단말기의 한정된 자원으로 인해 에이전트의 크기는 작아야 한다. <표 1>은 에이전트 프로그램의 각 모듈 별 크기와 전체 프로그램의 크기 및 실행프로그램 크기를 나타낸다.



(그림 12) 세션 별 에이전트 처리 시간 그래프

〈표 1〉 에이전트 프로그램 크기, 모듈 개수

구 성 요 소	코드크기	모듈개수
세션 핸들러	173 K	5 개
메시지 핸들러	109.1 K	6 개
DDF 트리 핸들러	63.1 K	4 개
DS 어댑터	13.16 K	6 개
사용자 Application	104.2 K	14 개
전체 코드크기	462.56 K	

국내 게임소프트웨어 회사에서 제공하는 휴대폰용 게임소프트웨어 중에서 대중적으로 이용되는 소프트웨어의 크기를 보면 최소 323K부터 최대 8951K로서 평균 2089.3K이다. 따라서 ThinkSync DM-SoftMan의 462.56K는 충분히 작은 크기임을 알 수 있다.

(그림 12)는 소프트웨어 배포를 위한 절차에 따른 실행시간을 측정한 결과이다. 소프트웨어 패키지 다운로딩은 웹서버를 통해서 수행한다. 소프트웨어 패키지 samba의 경우 다운로딩에 소요되는 시간은 0.165143초이다. 뿐만 아니라 다른 소프트웨어의 다운로딩 시간 또한 매우 적게 소요됨을 알 수 있다. 그러나 ThinkSync DM-SoftMan 관리 절차에 소요되는 시간은 에이전트 시스템 사양 혹은 네트워크 환경 등에 따라 차이가 발생한다.

6. 결 론

본 논문에서는 OMA DM 기반의 무선통신 단말기 소프트웨어 배포 관리 시스템 ThinkSync DM-SoftMan 개발 결과를 제시하였다. 그리고 매니저와 에이전트 설계 및 구현 구조를 제시하였다. 또한 소프트웨어 배포 관리에 필요한 각 소프트웨어 배포 관리 조작 모듈 구조를 제시하였다. 그리고 소프트웨어 배포 관리 조작 모듈 부분의 처리 흐름을 중점적으로 기술하였다. 구현된 ThinkSync DM-SoftMan 에이전트는 현재 리눅스 기반 RPM과 연동으로 단말기에 동작하고 있다.

향후 과제로는 다양한 RPM 뿐만이 아니라 다른 소프트웨어 패키지 관리 모듈과의 연동을 통한 검증이 필요하다. 또한, 소프트웨어 배포 관리 모듈의 코드를 세밀히 점검하여 불필요한 코드의 제거, 메모리 사용에 대한 최적화 및 실행 코드를 최소화함으로써, PC에 비해 상대적으로 처리 환경이 열악한 이동무선통신 단말기에 적용하여 좀 더 빠른 응답시간과 효율적인 자원 사용이 가능하도록 개선할 필요가 있다. 더 나아가 현재 개발된 소프트웨어 배포 관리자로 인한 축적된 기술을 바탕으로 OMA DM 기반의 무선 이동통신 단말기 구성 관리자, 펌웨어 업그레이드 관리자를 개발할 계획이다.

참 고 문 헌

[1] Uwe Hansmann, Riku Mettala, Apratim Purakayastha, Peter Thompson, "OMA Synchoronizing and Managing Your Mobile Data," pp. 21-34, PRENTICE HALL PTR, New Jersey, 2003.

[2] R.State, O.Festor, B.Zores, "An Extensible Agent Toolkit for Device Management," Network Operations and Management Synposium, 2004. NOMS 2004. IEEE/IFIP Volumn 1, 19-23 April 2004 Page(s):845 - 858 Vol.1

[3] P.Oommen, "A framework for integrated management of mobile-stations over the air." Proc of 7th. IEEE/IFIP Symposium on Integrated

[4] Management IM2001. 2001. Editors N.Anerousis and G.Pavlou ndr'e van der Hoek, Alexander L.Wolf, "Software Release Management for Component-Based Software," In Software - Practice and Experience, 33:2003, pages 77-98.

[5] 나승원, 오세민, "개인 휴대단말에서 응용 프로그램 동기화를 위한 자동설치 시스템의 설계 및 구현", 정보처리학회논문지A 제10-A권 제6호, 2003.12

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

[7] 장대진, 주홍택, 박기현, "SyncML DM 기반의 이동무선통신 단말기 관리 시스템 설계", KNOM Review 제 6권 2호, pp. 7?12, 2003

[8] Sync4j, "Sync4j Device Management Architecture," http://sync4j.funambol.com/main.jsp?main=documentation, April 2004

[9] Sync4j, "Sync4j.org," http://sync4j.sourceforge.net/web/theproject.html, 2005.10.04 검색

[10] Open Mobile Alliance, "OMA Device Management Protocol v1.1," http://www.openmobilealliance.org/release_program/dm_v112.html, 2004.01

[11] Open Mobile Alliance, "OMA Device Management Tree and Description Session v1.1," http://www.openmobilealliance.org/release_program/dm_v112.html, 2004.01

[12] Open Mobile Alliance, "OMA Device Management Standard Objects," http://www.openmobilealliance.org/release_program/dm_v112.html, 2004.01

[13] Open Mobile Alliance, "OMA Device Management Representation Protocol v1.2, " http://www.openmobilealliance.org/release_program/dm_v112.html, 2004.01

[14] Microsoft, "Windows Mobile Software for Pocket PCs and Smartphones: Microsoft Windows Mobile," http://www.microsoft.com/mobile/downloads/default.asp, 2005.10.04 검색

[15] Rpm, "www.rpm.org homepage - RPM Package Manager," http://www.rpm.org, 2005

[16] Microsoft, "다른 공급업체의 Microsoft Installer 패키지(MSI)를 만드는 방법," http://support.microsoft.com/default.aspx?scid=kb;ko;257718, 2005.10.04 검색

[17] Palm, "Welcome to Palm, Inc., formerly palmOne - Select a Destination," http://www.palm.com, 2005.10.07 검색

주 홍 택

e-mail : juht@kmu.ac.kr
1989년 한국과학기술원 전산학과 (공학사)
1991년 포항공과대학교 대학원 컴퓨터공학과(공학석사)
2002년 포항공과대학교 대학원 컴퓨터공학과(공학박사)
1991년~1997년 대우통신 종합연구소 선임연구원
2002년~현재 계명대학교 컴퓨터공학과 조교수
관심분야 : 네트워크 관리, 차세대무선이동통신, 이동통신단말기 관리, 네트워크 모니터링 등

박 기 현

e-mail : khp@kmu.ac.kr
1979년 경북대학교 전자공학과 학사
1981년 한국과학기술원 전자계산학과 석사
1990년 미국 밴더빌트 대학교 컴퓨터공학과 박사
1981년~현재 계명대학교 컴퓨터공학부 교수
관심분야 : 병렬/분산 운영체제, 모바일통신 소프트웨어, 성능분석 등

방 대 욱

e-mail : dubang@kmu.ac.kr
1980년 경북대학교 수학교육학과(학사)
1982년 한국과학기술원 전산학과 (이학석사)
1995년 서울대학교 대학원 컴퓨터공학과 (공학박사)
1982년~1986년 금오공과대학교 전자공학과 전임강사
1986년~현재 계명대학교 컴퓨터공학과 교수
관심분야 : 분산시스템소프트웨어, 이동에이전트, 상황인식, 의료정보 등