# 홈 네트웍 게이트웨이 셋탑 박스(HNGS)의 소프트웨어 구조 설계

임효상, 문재철, 강순주

경북대학교 전자전기공학부

Tel: 053-950-6604, E-mail : hslim@palgong.kyungpook.ac.kr

# Design of A Software Architecture for Home Network Gateway Set-Top-Box

Hyo-Sang Lim, Jae-Chul Moon, Soon-Ju Kang

Kyungpook National University

Tel: 053-950-6604, E-mail : hslim@palgong.kyungpook.ac.kr

**Abstract** : In this paper, we propose an software architecture for home network gateway set-top-box that supports the connectivity between various consumer devices and the Internet simultaneously. To improve the scalability of the software, the proposed architecture uses the abstracted protocol driving structure, and to enhance the user-friendliness, the unified device access and management user interface is implemented using web technology. A prototype for the proposed architecture is implemented for evaluating the usability under the home network test bed.

## 1. 서론

가전기기의 디지털화는 고성능 저가의 디지털 홈 네트웍 구축을 촉진시키고 있으며, 인터넷과 같은 컴퓨터 네트웍과의 연동 및 홈 오토메이션 시스템 구축을 용이하게 하고 있다. 이미 고성능 저가의 홈 네트웍 표준으로써 IEEE 1394[1], CEBus[2], X10[3]등이 사용되고 있으며, 이 표준에 맞춘 정보 가전 기기들의 개발이 활발히 이루어 지고 있다. 또한 이들 홈 네트웍 프로토콜과 인터넷과의 연동에 관한 표준으로 IP over IEEE1394[4]가 제정되어 있고, 이런 표준화와는 별도로 새로운 개념 즉 인터넷과 홈 네트웍 및 가전기기와의 접속을 시험적으로 구현한 시스템들이 개발되어 있는 단계이다. 개발된 시스템으로는 TV 와 웹과의 연동을 위한 인터넷 TV 시스템[5]이 대표적이며, 웹 폰[6][7]과 같은 통신기기를 이용한 인터넷 접속 시스템들도 개발되었다. 이외에 홈 오토메이션을 위한 게이트웨이 시스템[8], Java 기반의 브라우저를 이용한 홈 오토메이션 사용자 인터페이스 시스템[9]등이 개발되어 있다. 그러나 위의 시도들은 하나의 홈 네트웍 표준 프로토콜을 이용하여 개발되어 있어서 사용될 수 있는 가전기기가 한정적이고, 인터넷과의 접속을 지원하는 가전기기 경우에도 단순히 웹 브라우징 기능만을 제공하여, 인터넷과 홈 네트웍과의 연동을 지원하지 못한다. 본 논문에서는 이런 문제점들을 해결하기 위해서 CEBus 및 IEEE1394 와 같은 홈 네트웍 프로토콜을 동시에 지원하면서, 사용자 인터페이스로는 웹 브라우저를 이용할 수 있는 홈 네트웍 구조를 제안하고 이에 따른 소프트웨어 구조를 제안한다. 본 논문은 2 장에서 위의 문제점을 해결하기 위한 홈 네트웍의 구조를 설명하고, 3 장에서는 제안한 구조를 지원하는 셋탑박스의 소프트웨어 구조를 설명한다. 4 장에서는 프로토타입으로 개발된 시스템을 이야기하고, 5 장에서 결론을 맺겠다.

## 2. 제안하는 홈 네트웍의 구조.

IEEE1394[1] 및 CEBus[2]와 같은 홈 네트웍 프로토콜은 모든 가전 기기에 적합하도록 설계된 것이 아니다. 즉, CEBus 의 경우 가전기기의 원격 제어가 주 목적으로 멀티미디어 디바이스인 DVD, 디지털 캠코더등에 사용할 수 없고, IEEE1394 의 경우 멀티미디어 가전기기를 위해 개발되어 전구와 같은 간단한 가전기기용으로 비용이 너무 비싸다. 따라서 본 논문에서는 <그림 1>에서 보는 것과 같이 멀티미디어 기기들은 IEEE1394 에, 이외의 가전기기들은 CEBus 에 접속되고, 외부의 네트웍과의 연결은 케이블 모뎀을 이용한 인터넷 고속 접속을 하는 구성을 제안한다. 제안된 구조의 경우 기기들이 각각의 특성에 맞는 프로토콜을 이용하므로 이들간의 연동과 사용자에게 일관된 네트웍 인터페이스를 제공하는 기기가 필요하다. 본 논문에서는 이런 기능을 하는 셋탑박스를 홈 네트웍 게이트웨이 셋탑박스 약어로 HNGS(Home Network Gateway Set-top-box)라 하겠다.

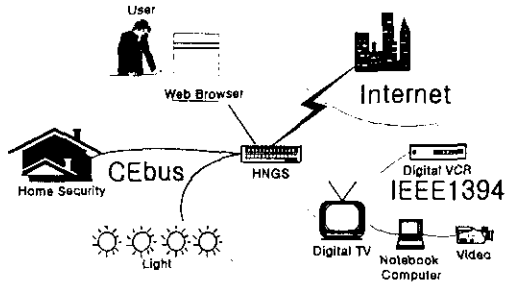


그림 1. 홈 네트웍 구성도

HNGS 는 각각의 네트웍에 접속된 각종 기기들이 셋탑박스를 통해서 인터넷과 연동이 가능하여 특히 웹을 이용한 홈 네트웍 기기 제어가 가능하다. 홈 네트웍 또는 가전

기기의 웹과의 연동은 사용자가 웹 브라우저만을 이용하여 네트웍 기기를 제어할 수 있어서 개발자쪽에서는 사용자 인터페이스를 위한 개발 시간이 단축되고, 사용자 입장에서는 홈 네트웍 제어를 위한 새로운 인터페이스의 학습이 필요 없는 장점을 가지고 있다. 또한 가전기기 유지보수 차원에서도 월드 와이드 웹을 이용하여 원격에서 소비자들의 홈 네트웍 기기들을 통합 유지 보수할 수 도 있다.

## 3. HNGS 소프트웨어 구조

셋탑 박스는 다양한 형태의 네트웍을 연동시키고 다양한 디바이스를 관리한다. 따라서 제안한 소프트웨어 구조의 프로토콜 구동 부분은 추상 클래스로 정의하여 새로운 프로토콜의 추가를 용이하게 하였고, 각 디바이스들은 일관된 방식으로 관리하여 각 홈 네트웍 프로토콜에 따라 다르게 관리되는 디바이스를 쉽게 접근사용할 수 있게 하였다. <그림 2>에서 보는 바와 같이 제안한 HNGS 구조는 디바이스 관리, 프로토콜 변환 및 데이터 전달 그리고 사용자 인터페이스 관리로 구성되어 있다. 우선 디바이스 관리는 네트웍에 접속된 디바이스의 관한 정보를 저장하고 이 정보를 필요한 모듈에게 전달한다. 각각 상세 기능을 살펴보면 디바이스 관리경우 디바이스 트리 관리, 디바이스 정보 전달 그리고 디바이스 찾기이다. 디바이스 트리의 관리는 각 프로토콜에 따른 어드레싱 구조를 추상화하여 셋탑 박스에서 일관된 어드레싱 인터페이스를 제공하는 기능을 담당하고, 디바이스 정보 전달은 사용자나 디바이스의 요구에 맞추어 디바이스 트리 정보등을 전달하는 기능을 담당한다. 그리고 디바이스 찾기는 각종 디바이스가 네트웍에 접속 또는 단절되는 것을 알아내고 사용자가 요구한 디바이스가 디바이스 트리에 없을 경우 네트웍에서 찾는 기능을 담당한다. 둘째, 프로토콜 변환 및 데이터 전달부에서는 셋탑 박스로 전달된 데이터를 목적 디바이스가 연결된 네트웍에 따라 데이터를 변환하는 기능과, 네트웍간의 데이터 전달을 담당한다. 또한 웹 브라우저에서 동작하는 사용자 인터페이스와 셋탑 박스간의 연결 설정을 위한 기능이 있다. 마지막으로 사용자 인터페이스부는 웹을 통한 사용자 인터페이스 전달 과 사용자 인터페이스 컴포넌트 ( CMAgent : Control & Monitor Agent) 및 웹 페이지를 관리한다. 위의 기능들을 셋탑박스의 기본 기능으로 설정하고, 이외의 기능들은 추상 클래스을 이용하여 기능을 추가 할 수 있게 하였다. 따라서, 각종 디바이스 인터페이스나 프로토콜 스택 변경되어도 전체 소프트웨어 구조 변경없이 새로운 홈 네트웍 구조를 수용할 수 있도록 하였다.

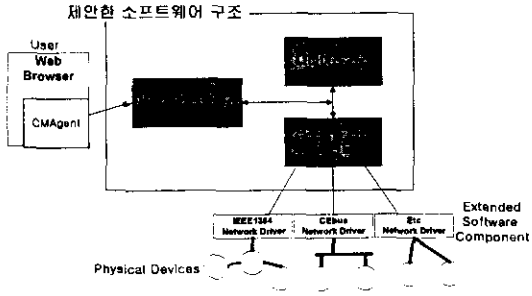


그림 2. 전체 소프트웨어 구조

## 3.1 디바이스 관리부의 설계

디바이스 관리부의 기능은 네트웍에 등록되어 있는 디바이스의 일관된 어드레싱 구조를 제공하고, 각 디바이스에 관련된 정보를 저장하는데 있다. 다음 <그림 3>은 이를 위해서 본 논문에서 설계한 디바이스 관리부의 개괄적인 클래스 다이어그램을 UML(Unified Modeling Language)[10]을 이용하여 표현한 것이다.

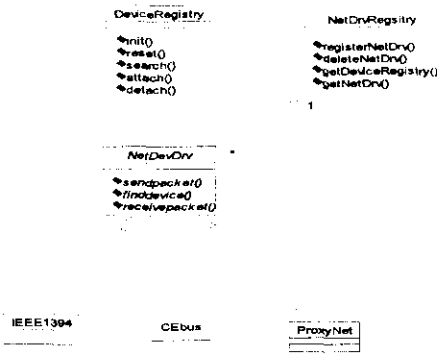


그림 3. 디바이스 관리부의 UML 표현

전체 클래스는 <그림 3>에서와 같이 크게 세 부분(DeviceRegistry, NetDrvRegistry, NetDevDrv)으로 나누어 진다. 각 NetDevDrv 는 추상 클래스로 팩킷 전달과 디바이스 찾기 서비스를 제공한다. 팩킷 전달은 사용자 인터페이스나 다른 디바이스에서 팻킷을 전달할 때 사용되고, 디바이스 찾기는 네트웍에 접속된 디바이스를 찾거나, 디바이스가 접속 또는 단절 되었을 때 이벤트를 발생할 때 사용된다. 위 두 메소드는 가상함수로 정의되어 있어서 NetDevDrv 를 상속 받은 하위 클래스 즉 네트웍 디바이스 드라이버(<그림 3>에서 CEBus 나 IEEE1394)에서 구체적인 동작을 정의한다. 본 논문에서 이와 같이 NetDevDrv 를 추상 클래스로 정의함으로써 셋탑박스의 전체 소프트웨어 구조 변경 없이 새로운 프로토콜을 수용할 수 있게 된다. <그림 3>에서 IEEE1394, CEBus 및 ProxyNet 클래스등이 NetDevDrv 추상 클래스를 상속 받아 새로운 프로토콜을 수용한 예이다. 여기서 ProxyNet 클래스는 디바이스가 네트웍에 연결되어 있지 않고 직접 셋탑 박스에 연결되어 있거나, 네트웍에 연결된 디바이스가 계산 능력이 부족할 경우(예 형광등) 셋탑 박스에서 직접 제어하는 프록시 객체(proxy object)를 관리하는 클래스이다. 따라서 아주 간단한 형태의 홈 디바이스도 셋탑박스를 이용하여 제어할 수 있다. <그림 3>에서 NetDevDrv 객체들을 관리하는 클래스가 NetDevDrvRegistry 이다. DeviceRegistry 는 디바이스 테이블을 이용하여 일관된 어드레싱을 지원하고, 디바이스 찾기 서비스를 제공한다. 이때 새롭게 등록되는 디바이스를 테이블에 저장하기 위해서 search(), attach(), detach() 메소드를 지원한다. 여기서 search() 메소드는 사용자 인터페이스나 소프트웨어 모듈에서 필요한 디바이스가 네트웍에 접속되어 있는지를 알아보거나, 네트웍에서 플러그 인(plug-in) 기능을 지원하지 않을 경우 사용자가 직접 접속된 디바이스를 설정할 때 사용된다. attach(), detach() 메소드는 플러그 인 기능을 지원할 경우에 새로운 디바이스가 접속되었을 때, NetDevDrv 가 DeviceRegistry 로 플러그 인 이벤트를 전달할 때 사용된다.

## 3.2 프로토콜 변환 및 데이터 전달부의 설계

제안한 홈 네트웍 구성에서 각 디바이스는 여러 개의

CMAgent 로부터 연결이 가능하다. 그러나 홈 네트웍 프로토콜이 연결기반(connection oriented)의 통신을 지원하지 않는 경우가 있으며, 또한 디바이스의 연산능력이 여러 개의 연결을 관리하지 못하는 경우가 있다. 이 경우를 위해서 셋탑박스는 CMAgent 와 소켓을 이용하여 연결기반으로 통신하고, 디바이스와는 데이터 그램기반으로 통신을 한다. 따라서 제안한 소프트웨어의 데이터 프로토콜 변환 및 데이터 전달부는 CMAgent 와의 소켓 연결 관리와 홈 네트웍 디바이스와 여러 CMAgent 간의 통신을 다중화하는 기능을 가진다. 또한 CMAgent 로부터 전달된 데이터를 각 홈 네트웍 프로토콜에 맞게 변환 전달한다.

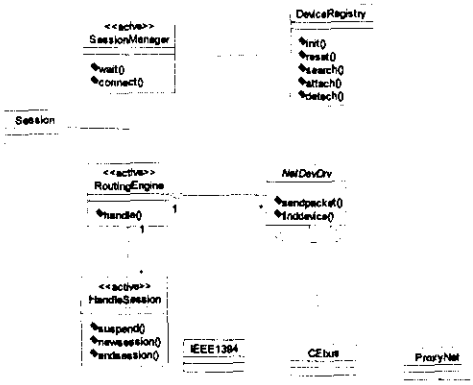


그림 4. 데이터 변환 및 전달부의 UML 표현

<그림 4>는 본 논문에서 프로토콜 변환 및 데이터 변환부를 설계한 것이다. <그림 4>에서 SessionManager 는 CMAgent 소켓과 홈 네트웍 디바이스와의 연결(세션)작업을 한다. 그리고 RoutingEngine 은,세션을 통해 전달되는 데이터를 각 디바이스로 다중화하여 전달한다.

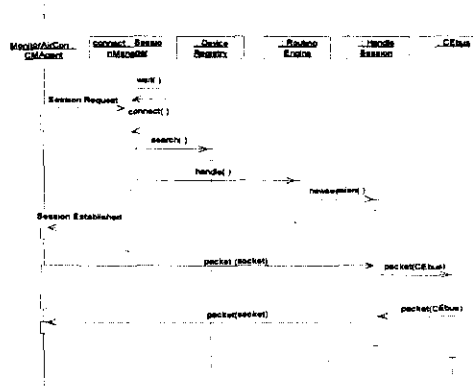


그림 5. 세션 연결과 데이터 전달과정

<그림 5>은 위의 클래스의 동작을 나타내는 UML 의 시퀀스 다이어그램(sequence diagram)이다. 먼저 SessionManager 는 CMAgent 로부터 세션요구를 기다린다(wait()). CMAgent 가 소켓 접속 후 디바이스와의 세션을 요구하면 SessionManager 는 DeviceRegistry 에게 디바이스가 연결된 네트웍에 관한 정보( 네트웍 디바이스 드라이버 객체 레퍼런스 및 네트웍 주소)를 받아 세션을 확립한다(connect()). 일단 세션이 설정되면 SessionManager 는 세션 정보를 RoutingEngine 에게 전달한다(handle()). RoutingEngine 에서는 각 세션 마다 쓰레드(<그림 4>에서 HandleSession 클래스)를 생성시키고, 세션정보를 쓰레드에게 전달한다. 세션 정보로는 CMAgent 소켓 레퍼런스, 디바이스의 네트웍 어드레스 등이 포함된다. 세션이 설정되고 쓰레드가 동작하기 시작하면, CMAgent 에서 전달되는 데이터는 네트웍 디바이스 드라이버(그림 4 에서 CEbus, IEEE1394 등)로 전달되고, 디바이스에서 전달된 데이터는 소켓을 통해서 CMAgent 로 전달된다.

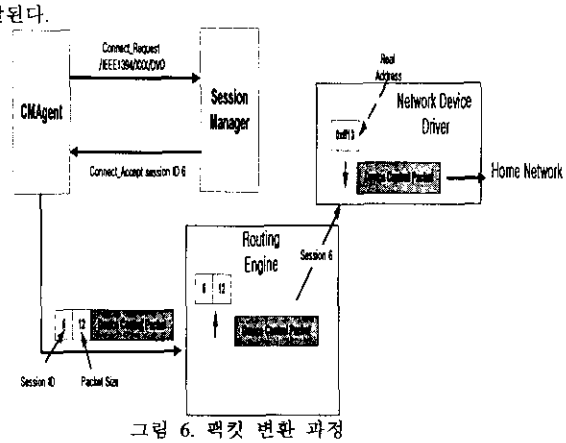


그림 6. 팩킷 변환 과정

설계한 시스템에서는 CMAgent 가 데이터를 디바이스가 제공하는 제어 프로토콜 포맷에 따라 전달하여, 셋탑 박스에서 각 제어 프로토콜에 따른 팩킷 변환 작업을 없앴다. 그러나 CMAgent 와 셋탑박스간의 세션 연결과 데이터 전달을 위한 프로토콜은 필요하다. 이 목적을 위해 셋탑박스에서 설계한 프로토콜은 세션 연결을 위한 핸드 쉐이크(handshake)만을 정의하고, 전달하는 팩킷에는 최소한의 오버헤드를 첨가하였다. 세션 연결을 위해서 <그림 6>에서와 같이 세션 요구 팩킷을 정하였으며, 이 팩킷에는 DeviceRegister 내의 디바이스 주소(<그림 6>에서 /IEEE1394/XXX/DVD)을 가지고, 셋탑 박스에서 세션이 확립되면 세션 ID(<그림 6>에서 session ID 6)를 되돌린다. 만약 디바이스가 없으면 session ID 0 를 전달하고 이어서 에러 코드를 전달가 전달된다. 일단 세션이 연결되면 CMAgent 에서 전달하는 팩킷은 <그림 6>에서 보듯이 팩킷의 네트웍 주소는 비워두고 앞에 세션 ID 와 사이즈가 첨가된다. 그러면 전달 받은 팩킷은 셋탑박스에서 세션 정보에 따라 네트웍 주소가 첨가된다. 이 과정은 NetDevDrv 를 상속 받은 네트웍 디바이스 드라이버에서 정의된다. 네트웍 디바이스 드라이버는 이 과정과 함께 세션 다중화 작업도 한다. 다중화 방식은 각 프로토콜 마다 다르게 정의된다. 예를 들어, IEEE1394 의 경우 어드레스를 할당하는 방식이 노드 ID 즉 디바이스 ID 와 디바이스내의 주소로 구분되어 있다. 따라서 다중화를 위해서 각 세션은 셋탑박스의 하나의 주소를 할당 받도록 한다. 이렇게 하면 각 세션으로 전달되는 팩킷을 다중화 시킬 수 있다. X10 의 경우 디바이스(receiver)간의 통신 기능이 없이 단지 제어기(transceiver)와의 통신만 이 가능하다. 따라서 이런 경우 디바이스에서 보낸 팻킷에는 목적 주소가 없으므로 네트웍 디바이스 드라이버는 팻킷을 디바이스에 접속된 모든 세션들로 전달한다. 따라서 X10 와 같은 프로토콜의 경우 CMAgent 는 전달된 팩킷이 자신이 요구한 팩킷인지 확인하는 작업이 필요하다.

## 3.3 사용자 인터페이스부의 설계

제안된 시스템에서 사용자는 Java applet 으로 작성된 사용자 인터페이스 에이전트인 CMAgent 를 셋탑박스로부터 제공 받아 웹을 통해 디바이스를 제어한다. 사용자 인터페이스가 전달되는 과정을 설명하면 먼저 사용자는 셋탑박스 홈 페이지에 연결한다. 연결이 되면 웹 서버의 Runtime Web Page Generator 가 DeviceRegistry 에서 홈 네트웍에 연결 중인 디바이스 정보를 받아서 웹 페이지로 변환하여 사용자에게 전달한다. 웹 페이지를 전달 받은 사용자는 원하는 디바이스에 클릭하면 웹 서버는 사용자가 요구한 디바이스를 찾고 디바이스 인터페이스 에이전트를 웹 브라우저로 전달한다. 전달 받은 에이전트는 앞에서 설명한 것과 같이 자신이 관리할 디바이스와 소켓을 이용하여 세션을 연결하고, 이 세션을 통해서 디바이스를 제어한다. 이때 에이전트가 어떻게 디바이스 제어하는 가에 대해서는 정하지 않는다. 즉, 각각의 디바이스 제어 프로토콜등을 정의하지 않고 셋탑박스는 단지 팻킷 전달과 CMAgent 만 관리함으로써 각각의 디바이스 제작회사는 각자의 고유한 사용자 인터페이스와 기능을 첨가할 수 있다. 다음은 위의 과정을 표현한 것이다.

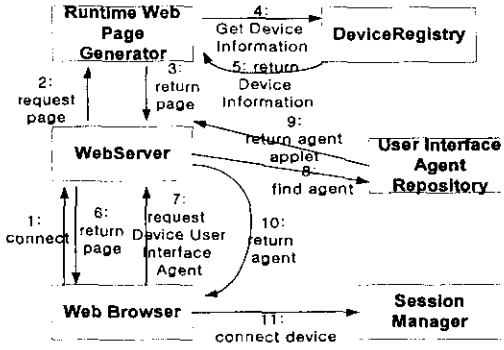


그림 7. CMAgent 전달과정

설계한 시스템에서는 CMAgent 를 사용자에게 전달하는 것은 <그림 7>과 같이 정의하였으나, 셋탑 박스에 CMAgent 를 등록하는 방식에 대해서는 정의하지 않았다. 다만, 네트웍 디바이스 드라이버 객체나 확장 모듈이 CMAgentRegistry 의 기능을 이용하여 추가할 수 있도록 하였다.

## 4. 구현

제안한 소프트웨어를 적용하기 위해서는 셋탑 박스 하드웨어의 구현이 필요하다. 그러나, 아직 하드웨어의 설계와 구현이 되어 있지 않아 현재의 소프트웨어를 windows NT 환경에서 프로토타입으로 구현하고, 시뮬레이션 디바이스 이용하여 실험해보았다. 프로토타입은 언어로 C++를 사용하였고, 사용자 인터페이스를 위해서 최소한의 기능을 가지는 HTTP 서버를 구현하였다. <그림 8>은 구현된 프로토타입의 CMAgent 가 웹 브라우저에서 실행되는 모습을 잡은 것이다. <그림 8>의 CMAgent 는 선풍기 시뮬레이션 디바이스와 연결되고 선풍기 속도를 제어하고, 현재의 선풍기 속도를 측정할 수 있다. 향후 셋탑박스 하드웨어가 완성되면 WindowsCE 환경하에서 내장형 실시간 시스템으로 재구현 될 것이다.

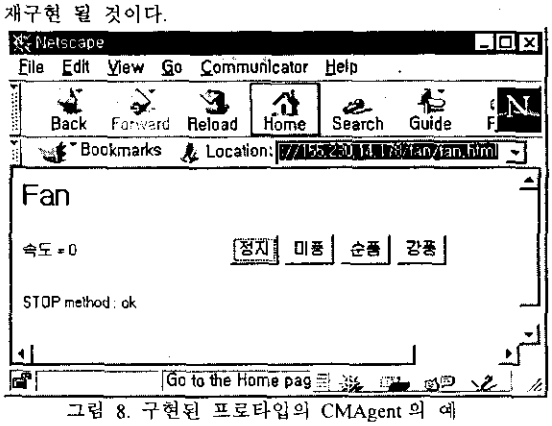


그림 8. 구현된 프로타입의 CMAgent 의 예

## 5. 결론

본 논문에서는 HNGS 즉 홈 네트웍 게이트웨이 셋탑 박스의 소프트웨어 구조에 대해서 논하였다. HNGS 를 이용함으로써 각기 다른 목적으로 개발된 홈 네트웍 프로토콜을 하나의 사용자 인터페이스로 통합 시킬 수 있었고, 셋탑 박스의 기능을 특정 프로토콜에 한정시키지 않도록 설계하여 확장성 좋은 소프트웨어를 설계할 수 있었다. 특히 이 확장성 덕분에 셋탑 박스 하드웨어에 비의존적으로 프로토타입을 구현할 수 있었다.아직 제안한 시스템은 셋탑 박스에 적용된 것이 아니어서, 실제 셋탑박스 시스템에 적용하고, 각 프로토콜을 연동에 따르는 효율성 문제와 각 프로토콜 디바이스 드라이버 구조에 대해서도 연구를 더 해야 한다. 그러나 사용자 인터페이스와 디바이스 관리에 관련된 기능들을 설계 실험해 봄으로써 제안한 소프트웨어의 유용성을 입증할 수 있었다.

## 참고 문헌

[1] IEEE P1394a Draft Standard for a High Performance Serial Bus, P1394a Draft 1.00, Sep. 1997
[2] EIA Home Automation System(CEBus) Standard IS-60, "Introduction to the CEBus Standard," EIA, Part 1. June 1992
[3] http://www.x10.org
[4] P. Johansson, "INTERNET PROTOCOL(IP) over IEEE1394-1995," Revision 3, Proposal for IP1394WG, Aug. 1997.
[5] http://www.webtv.com
[6] http://www.alcatel.com
[7] http://www.nortel.com
[8] J. Desbonnet and P. M. Corcoran, "System Architecture and Implementation of a Cebus/Internet Gateway," IEEE Trans. Consumer Electronics, vol. 43, no. 4, Nov. 1997.
[9] P. M. Corcoran and J. Desbonnet, "Browser-Style Interface to a Home Automation Network," IEEE Trans. Consumer Electronics, vol. 43, no. 4, Nov. 1997.
[10] OMG, "UML Notation Guide," OMG, Aug. 1997.