정보가전 제어 멀티에이전트시스템

김일연* · 송준현* · 김일곤**

Information Appliance Control MultiAgentsystem

Il-Yun Kim* · Joon-Hyun Song* · Il-kon kim**

요 약

가정 내에서의 초고속망 이용이 활성화 되면서 가정 내에서의 활동은 매우 다양하고 복잡하며 그에 따른 서비스도 여러 가지 형태가 존재한다.

외부 인터넷망과 연동되는 홈 네트워크의 개념이 생기기 전부터 홈 오토메이션, 홈시큐리티 등의 서비스를 제공하기 위한 고립적인 형태의 망과 서비스 제공 시스템이 있었다.

월드 와이드 웹 서비스를 기본으로 한 인터넷의 폭발적인 활성화와 고속 인터넷망의 확산에 따라 가정 내에서 제공되는 서비스를 인터넷과 연계시키려는 시도가 보편화되었고 가정 내에서 사용되는 독립적인 기기들을 인터넷에 연결하여, 외부의 정보를 이용해서 활용하거나, 가전내의 기기들을 외부에서 액세스할 수 있게 되었다. 이렇게 하여 보다 풍부한 서비스를 제공하고, 사용상의 시간적인 제약을 완화할 수 있다.

예들 들어 인터넷상에서 조리법을 전송받아 전자레인지를 작동할 수 있으며, 외부에서 잘 못 켜둔 가스밸브를 잠글 수도 있다.

가정 내에서 wrhd되던 이러한 서비스들은 서로 다른 하드웨어와 통신방법을 사용하고 있어, 이러한 서비스를 통합하여 운영, 관리할 수 있도록 하는 홈 서버의 개념이 대두되었다. 외부 인터넷 망과 연계된 홈 오토메이션, 홈 시큐리티등을 제공할 수 있는 기반이 마련되었고, 가정 내에서 독립적으로 제공되던 서비스들을 단일 홈 서버를 통해서 관리하려는 시도가 진행되었다.

그러나 홈 서버는 개발하는 업체의 보유기술에 따라 중심기능은 약간씩 차이를 보이고 있다.

홈 서버 응용 영역은 전 문 지식을 소유한 인력과의 연계가 필요하고, 다양한 서비스 영역 행위가 아주 복잡하기 때문에 이들을 연결해 주는 자치 시스템을 필요로 한다.

또한, 홈 서버 서비스 영역에서 정보는 각 영역 전문가가 가진 지식의 전달을 필요로 하기 때문에, 지식을 주고받는 멀티 에이전트의 시스템의 활용 영역으로 적당하다.

멀티 에이전트 시스템은 분산된 환경에서 에이전트 간에 에이전트 간에 에이전트 통신 언어를 사용해서 대화를 하기도 하고, 상호 협력하는 에이전트들로 구성된 시스템을 뜻하며 홈 서버처럼 동적이고 고도의 자치성을 요구하는 영역에 적당하다.

멀티 에이전트 플랫폼으로는 FIPA(Foundation for Intelligent Physical Agents)가 제시한 에이전트 표준 플랫폼이 1997년부터 2000년에 이르기까지, 계속적으로 변화, 발전하고 있다.

본 연구는 FIPA에서 제시하는 플랫폼을 기반으로 홈 서버에 정보가전을 제어하는 에이전트를 두고 외부에서 다른 에이전트가 홈 서버에 위치하는 에이전트와의 통신을 통하기 정보가전을 지능적으로 제어하도록 하였다.

정보가전 에이전트는 가정 내 가전제품을 외부에서 제어하기 위한 에이전트이다. 단순한 관리가 아닌 에이전트로 하여금 지능적으로 가전제품 관리를 하게 한다. 정보가전 에이전트는 홈 서버에서 작동하는 에이전트와 PDA에서 작동하는 에이전트로 구성된다. 정보가전 에이전트는 전력량, 수도 사용량 제어와 가전제품 제어 기능과 보안 관련 서비스를 제공한다.

두 에이전트는 FIPA에서 정의된 규격에 맞게 만들어지기 때문에 FIPA 명세서를 따르는 다른 에이전트와 자유로운 통신이 가능하다.

Keyword : 정보가전, MultiAgent, HomeServer Agent, Mobile Agent

---

* 경북대학교 컴퓨터과학과 지능정보 연구실 석사과정
** 경북대학교 컴퓨터과학과 교수

1. 서 론

정보 가전 생활이 서서히 현실로 다가오고 있다. 가정 내의 모든 가전제품들이 외부에서 마음대로 조절하고 사용가능한 홈 네트워크 서비스 시대가 눈앞에 펼쳐지고 있다.

업계간 표준 전쟁도 만만치 않다. Sony, Philipse, Panasonic등의 가전업체를 중심으로 한 하비(HAVi)진영과 Microsoft가 지휘하는 UPnP진영과 Sun Micro System사가 중심이 되는 Jini진영이 대립하고 있다.

이번에 우리가 제작한 정보가전 에이전트는 FIPA-OS를 바탕으로 하여 OSGi(Open Services Gateway Initiative)를 이용함으로써 TV, PC, VTR, 냉장고, 기타(전등, 가스밸브, 현관문) 등 가정에서 쓰는 모든 전자제품을 어떤 표준에 구애받지 않고 외부에서 단순한 제어가 아닌 좀더 지능적으로 제어하기 위해 만들어진 소프트웨어다.

정보가전 에이전트는 에이전트플랫폼인 FIPA-OS (Foundation for Intelligent Information Agent - Open Source)와 Micro FIPA-OS를 기반으로 만들어졌다.

FIPA(Foundation for Intelligent Information Agent)는 에이전트를 만들기 위한 소프트웨어 표준을 제시한다. FIPA-OS 플랫폼은 FIPA Spec에 맞게 자바로 구현된 에이전트 플랫폼이며, Micro FIPA-OS는 모바일기기에서 작동하는 에이전트를 위한 플랫폼이다. 정보가전 에이전트는 크게 두 부분으로 나뉘는데, 집안의 가전제품 조작을 담당하는 홈 서버에서 작동하는 HomeServer 에이전트와 PDA상에서 작동하는 Mobile 에이전트이다. Home Server 에이전트는 FIPA-OS를 바탕으로 만들어졌고, Mobile 에이전트는 Micro FIPA-OS를 이용하여 만들어졌다.

2. 정보가전과 멀티에이전트 시스템 플랫폼, FIPA-OS

2.1 정보가전

TV를 비롯한 냉장고, 전자레인지등의 가전기기가 디지털화 되고 유 무선으로 네트워크화 되어 상호 데이터통신이 가능한 인터넷 단말기를 말한다.

거실에서 사용돼는 디지털TV, 주방에 사용되는 디지털 냉장고, 디지털 전자레인지, 침실과 욕실에서 사용되는 디지털 의료기기, 디지털 세탁기 등이 대표적인 정보가전이라 할 수 있다.(인터넷 정보가전, 12월)

정보 가전은 하나의 통신망으로 묶는 홈 네트워크가 형성되어 상호기기 간의 통신은 물론 이를 통한 정보의 공유 및 엔터테인먼트 향유, 그리고 에너지 절약 기능과 홈 오토메이션 기능 등을 제공할 수 있는 시스템과 소프트웨어가 지원되어야 한다.

홈 네트워크는 홈 네트워크 프로토콜로 HomePNA(Home Phoneline Networking Allia nce), USB(Universal Serial Bus), IEEE1394, HomeRF(Home Radio Frequency), Bluetooth등이 사용되고 있고 이들을 관리하고 인터페이스하기 위한 미들웨어로 HAVi(Home Audio Video Inter operability), VHN(VESA), JINI, CEBus, UPnP(Universal Plug and Play)등이 제안되었다.

또한 이러한 홈 네트워크 프로토콜과 디바이스 인터페이스 표준들을 적용한 다양한 연구들이 진행되고 있다.(박성호 and 강순주, 2001)

본 논문에서는 홈 제어 네트워크 프로토콜로서는 PLC(PowerLine Communications)를 기반으로 하는 LonWorks을 이용하며 미들웨어로는 OSGi를 이용한다. LonWorks기술은 빌딩 자동화를 비롯하여 공자, 교통관제 등의 분야에서 널리 사용되고 있는 제어네트워크솔루션이다.

현재 많은 업체들이 다양한 LonWorks제품들을 생산하고 있으며 제조업체가 달라도 하나의 시스템을 구성할 수 있는 개방형 멀티-벤더 구조를 지닌다. LonWorks 프로토콜은 분산 제어 네트워크를 구축할 수 있으며, 다양한 유선 통신 매체뿐만 아니라 무선(RF)과 적외선(Infrared)등의 통신매체도 물리계층 통신 미디어의 특성에 무관한 시스템 개발이 가능하며 네트워크 관리 체계가 매우 간단해진다. 이 프로토콜은 응용 분야의 특성상 실시간성 보장을 위하여 각 데이터 패킷에 우선순위 정보를 포함하고 있으며 동기화된 엄격한 타이밍 처리 기능을 가지고 있다. OSI 프로토콜 계층 가운데 애플리케이션 계층을 제외한 모든 계층이 표준화된 하드웨어(Neuron Chip)로 구현되어 편리한 개발환경을 제공한다. 또한 LonWorks 네트워크는 각 노드의 정보를 네트워크 변수를 이용해 공유하며, 정의된 네트워크 변수간의 바인딩(Binding)작업을 통해 구성, 설치가 이루어진다.

LonWorks 표준화 단체인 LonMark에서 이 네트워크 변수의 표준인 SNVT(Standard Network Variable Types)를 제공하며, 현재 홈 네트워크기술을 개방하고 있는 많은 업체(Cisco, Sun Microsystems, Microsoft)들이 LonWorks기술을 채택하고 있다. (전호인 and 송동일, 2001)

OSGi는 1999년 3월 1일에 15개 회사가 미들웨어와 응용 프로그램간의 API를 정의하는 것을 목적으로 시작한 단체이다. 현재 미들웨어에 대해서는 Jini와 UPnP, 혹은 HAVi등을 많이 이야기하고 있지만 각각의 목표와 내용이 다르기 때문에 응용 프로그램을 구현하는 입장에서 어떤 특정한 미들웨어를 선택할 수도 없는 입장이다.

OSGi는 Java를 기반으로 외부 망과 인터페이스나 응용 프로그램과의 인터페이스에 대한 API들이 정의되어 있지만 앞으로는 다른 미들웨어나 service들에 대한 API도 정의할 예정이다.

OSGi는 특정 기능을 수행하는 자바 인터페이스와 실제 구현 객체인 Service, Service를 제공하기 위한 기능적 배포단위인 Bundle, Bundle들의 라이프사이클을 관리하는 Bundle 실행 환경인 프레임워크로 구성되어 있다.

서비스는 미리 정의된 서비스 인터페이스를 통해 접근이 가능한 컴포넌트다. 하나의 애플리케이션은 여러 개의 서비스의 협동작업을 통해 구성되고, 런타임 시에 필요한 서비스를 요청할 수도 있다.

프레임워크는 각 서비스와 그 서비스에 해당하는 실제구현에 대한 매핑을 가지고 있고, 간단한 쿼리 메커니즘을 통해 서비스의 실제구현을 찾을 수 있다. 또한 프레임워크는 각 서비스간의 상호 의존관계를 관리한다.

번들은 여러 서비스의 구현을 하나의 패키지로 묶은 JAR 파일의 형태로 존재한다. JAR파일에는 하나 이상의 서비스의 구현객체와 리소스 파일 및 매니페스트 파일을 포함하고 있다.

번들 콘텍스트는 프레임워크 내부의 번들의 실행환경이며, 번들의 설치, 실행, 정지 및 삭제 등의 번들의 라이프 사이클을 관리한다.

2.2 멀티 에이전트 시스템

그림 6은 두 개의 에이전트 플랫폼(Agent Platform)간의 교류를 보여준다.

각 에이전트 플랫폼은 하나 이상의 에이전트를 가질 수 있고, 이 에이전트들의 관리를 위한 시스템인 Management System과 다른 에이전트에게 자신의 능력을 알릴 수 있도록 하는 Directory Facilitator를 가지고 있다. 각 에이전트는 메시지 전달 시스템(Message Transport System)을 통해 메시지를 주고받을 수 있는데 이를 통해서 자신이 속한 플랫폼이 아닌 다른 플랫폼 상에 있는 다른 에이전트와도 메시지를 교환하여 통신할 수도 있다.

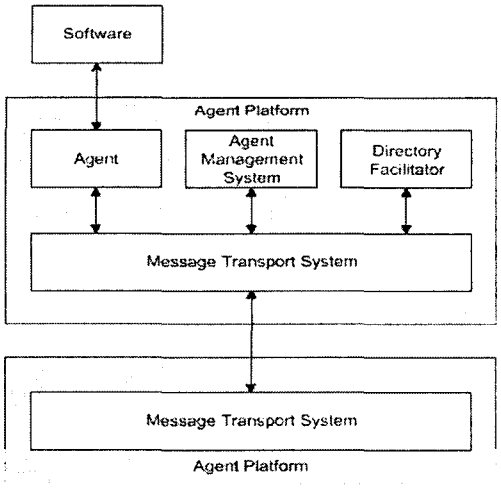


그림 6 에이전트 관리 참조 모델

즉 에이전트 플랫폼은 에이전트가 만들어지고 동작하며 다른 에이전트와 상호 교류할 수 있는 물리적 기반을 제공하는 것이다. 이것은 에이전트의 생성, 등록, 해지, 특정 위치로의 이동, 의사소통 등의 논리적인 행위를 모두 포함할 수 있게 구성된다.

2.3 FIPA OS

FIPA-OS는 FIPA 규격을 준수하는 에이전트를 빠르게 개발할 수 있도록 도와주는 컴포넌트 기반의 툴킷이다. FIPA 규격을 준수하고 있으며, 다른 에이전트 플랫폼과 상호 운용성이 높은 플랫폼을 제공하고 있으며, 다양한 API를 제공함으로써, 프로그래머가 쉽게 FIPA 규격을 준수하는 에이전트를 만들 수 있게 도와준다.

FIPA-OS는 기본적으로 상위 레벨의 Communication과 Conversation을 관리하는 함수와 task와 subtasks의 생성을 제공하고, 기본 플랫폼에서는 FIPA에서 정의한 AMS와 DF등의 기능을 제공하고 있으며, 하위 레벨의 transport system은 RMI, IIOP, http 등을 지원한다.

FIPA-OS는 스택을 기초로 한 layer로 디자인되어 있고 각각의 에이전트는 스택 안에 위치하여 컴포넌트로 구성되어 있다. FIPA-OS의 추상적인 모델은 그림 1와 같다.

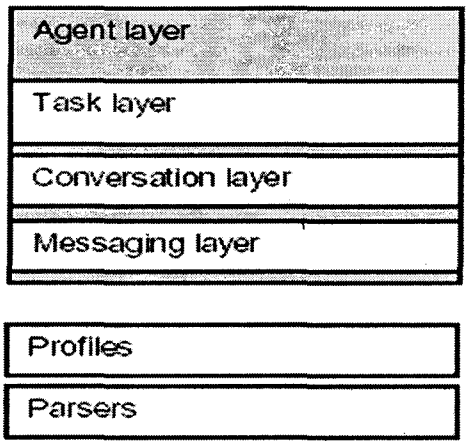


그림 1 FIPA-OS의 추상적 구조

FIPA-OS는 task와 conversation을 관리하는 컴포넌트와 메시지 서비스, 그리고 메시지 전송 부분으로 구성된다.

에이전트 플랫폼은 인터 플랫폼 커뮤니케이션을 다룰 수 있는 exception을 가지고 있는 에이전트로서 구현되어 있다.

FIPA-OS 에이전트 플랫폼은 에이전트 중심적으로 하나의 에이전트는 자신의 transport 스택과 함께Profile 정보를 기초로 하여 MTS에서 수행되어진다.

FIPA-OS 에이전트 플랫폼은 에이전트 중심적으로 하나의 에이전트는 자신의 transport 스택과 함께 profile 정보를 기초로 하여 MTS에서 수행되어진다.

그림 2은 FIPA-OS의 클래스와 인터페이스의 관계에 대해서 간략하게 보여주고 있다. 회색으로 표시되는 박스는 인터페이스를 나타내고, 흰 박스로 표시되는 부분은 인스턴스화 될 수 있는 클래스를 나타낸다.

각각의 메시지의 전송과 수신은 Conversation 매니저에 의해서 조정되고, 각각의 Conversation Manager는 역시 Task Manager에 의해서 조정된다. Task Manager는 Conversation 뿐만 아니라 작은 단위의 Task에 대해서도 관리할 수 있다.[5]

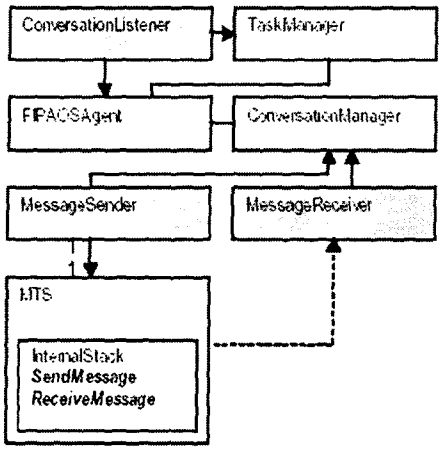


그림 2 FIPA OS
각 컴포넌트 사이의 관계

2.4 Micro FIPA-OS

Micro FIPA-OS는 FIPA-OS를 축소한 것으로 FIPA-OS를 간단하게 구현한 것이다. 기본적으로 Micro FIPA-OS는 임베디드 환경에서 최적화되어 있다.

Sun사의 Personal JAVA VM이나 JDK1.1.8/1.2.X /1.3.X, EpocVM 등에서 작동될 수 있으며, Comp aq의 iPAQ H3600 시리즈와 Casio사의 Cassiopeia E115, Psion Series 5mx, Intel X86 구조 등에서 사용될 수 있다.

Micro FIPA-OS는 RMI나 IIOP를 지원하지 않고 있으며, 기본적으로 내부 http transport를 사용하거나 연동을 위해서 FIPA에서 정의하고 있는 http를 사용한다.

Micro FIPA-OS 에이전트는 FIPA-OS 에이전트와 마찬가지로 task와 conversation을 사용하고 있으며, FIPA-OS 에이전트를 개발하는 방법으로 프로그래밍 한다.

Micro FIPA-OS는 transport와 resouce pool을 공유하고 가능한 로컬 메시징 방법으로, 작은 디바이스 위에 여러 개의 에이전트를 지원한다. 그러나 task, conversation, 로컬 메시징은 많은 오버헤드를 가져옴으로 에이전트 개발자는 로컬 메시징의 사용을 피하고 잘 통합된 에이전트를 만들어야 한다.

Micro FIPA-OS는 하나의 FIPA-OS 플랫폼의 종속된 형태로 동작할 수도 있으며, 또는 하나의 독립된 플랫폼의 형태로서 다른 플랫폼과 같이 상호 운용될 수도 있다.

그림 3는 Micro FIPA-OS가 어떻게 사용되어 질 수 있는지를 보여준다.

위의 그림은 Micro FIPA-OS가 이미 존재하고 있는 에이전트 플랫폼에 종속되어 동작하는 형태로 Micro FIPA-OS 상에 존재하는 에이전트들은 이를 포함하고 있는 다른 에이전트 플랫폼의 AMS와 DF를 사용하고, 이 두 플랫폼 간의 메시지 교환은 내부 transport를 이용한다.

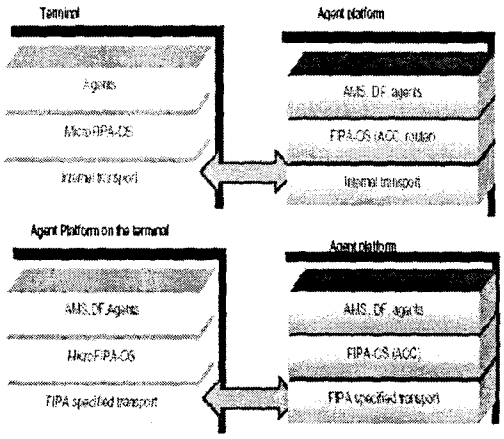


그림 3 Micro FiPA-OS의 배치

그림 3의 아래쪽은 Micro FIPA-OS가 하나의 독립적인 플랫폼의 형태로 동작하는 것으로 일반적인 에이전트 플랫폼과 같이 자신의 AMS와 DF를 이용하여 각 에이전트들을 관리하거나 에이전트들에게 서비스를 제공하고 다른 에이전트와의 메시지 교환에는 http와 같이 FIPA에서 정의한 방법을 이용한다.

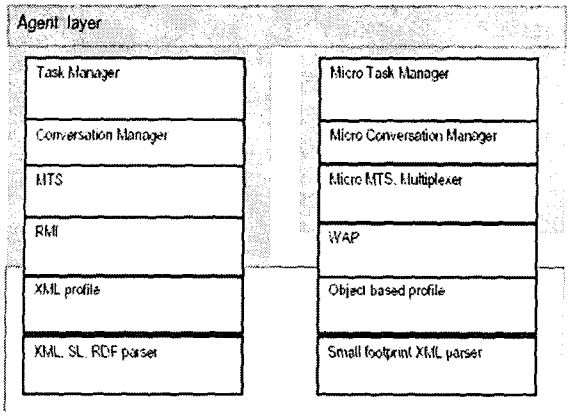


그림 4 Micro FIPA-OS와
FIPA-OS의 구조 비교

3. 정보 가전 제어 멀티 에이전트 시스템 설계

3.1 구현 시나리오

아래의 정보가전이 정보가전 제어 멀티에이전트시스템에서 구체적으로 어떻게 작동되고 대해서 살펴보도록 하자.(단 아래의 서비스는 사용자가 원하는 서비스라고 가정한다.)

사용자는 모바일 에이전트에 있는 Configration Menu를 이용하여 자신이 사용하고 있는 정보가전을 선택하고 각각의 정보가전에 대한 서비스에 대해서 본인이 원하는 것을 선택하는 것이 필요하다.

사용자가 원하는 모바일 에이전트를 구축하였다면

홈 서버 에이전트에게 집안의 정보가전에 대한 상태를 요청한다.
요청받은 홈 서버 에이전트는 집안의 정보가전을 체크한 후 모바일 에이전트에게 상태에 대한 정보를 전달하게 된다.
또한 홈 서버 에이전트는 모바일 에이전트로 하여금 정보가전의 제어에 대한 메시지가 도착하면 이 메시지가 어떤 정보 가전에 대한 메시지이며 어떠한 서비스를 원하는 것인지를 내부적으로 판단하여 그 해당하는 정보가전을 제어한 후, 그 결과에 대해서 모바일 에이전트에게 전달한다.
만약 사용자가 직접 제어하는 경우는 원하는 서비스 충족되지 않았을 경우 바로 다시 충족되도록 서비스 요청하면 되지만 사용자의 자동설정에 의해 모바일 에이전트가 알아서 홈 서버 에이전트에게 어떤 서비스를 요구하고 그 서비스 요청의 결과에 대해서 모바일 에이전트는 판단하여 사용자의 원하는 서비스가 아니라면 다시 홈 서버 에이전트에게 서비스를 요구하게 된다.

3.2 시스템 구성

정보가전 에이전트는 크게 두 부분으로 나뉜다. 집안의 홈 서버에서 작동하는 홈 서버 에이전트와 PDA 상에서 작동하는 모바일 에이전트로 나뉜다. (그림 7 참조)

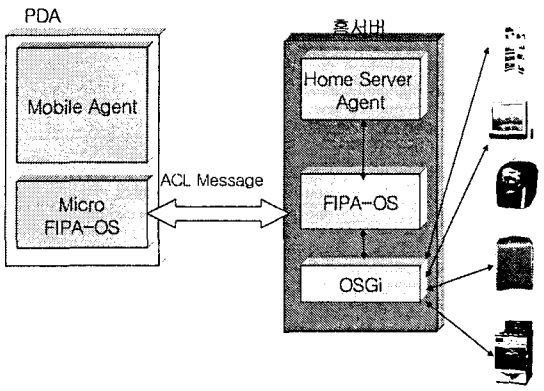


그림 7 시스템 구성도

두 에이전트간의 통신은 FIPA-OS에서 제공해주는 통신 채널을 이용한다.
우선 채널을 이용하기 위해선 메시지를 만들어야 하는데, FIPA-OS 에서는 ACL(Agent Communication Language), 즉 메시지 포맷을 제공해주고 있다.
집안의 홈 서버 에이전트는 정보가전 및 기타 제품을 직접 제어하는 에이전트로 OSGi를 이용하여 디바이스 영역을 관리하게 된다.
홈 서버 에이전트는 집안에 있는 정보가전의 상태에 대한 정보와 누적된 정보를 보유하고 있으며, 이에 대해 외부 에이전트 즉 모바일 에이전트에게서 오는 모든 요청에 대해서 처리하게 된다.
정보가전에 대한 단순처리는 물론이며, 모바일 에이전트의 특별한 요청에 대해서도 처리하며, 그 상태에 대해서 자기 자신은 물론, 모바일 에이전트에게 모든 정보를 알려준다. 예를 들어 TV를 켜라는 단순한 메시지뿐만 아니라 냉장고를 어느 계절엔 어떤 강도로 동작하라든지, 에어컨의 동작에 대해서 최대한 전기를 아끼면서 작동할 수 있도록 요청할 수 있으며 이에 대해서 홈 서버 에이전트는 처리하고 관리하게 된다.
그림 8은 FIPA에서 정의한 ACL 메시지의 형태로 이러한 단순한 메시지부터 복잡한 메시지까지 두 에이전트간에 통신할 수 있게끔 하고 있다.

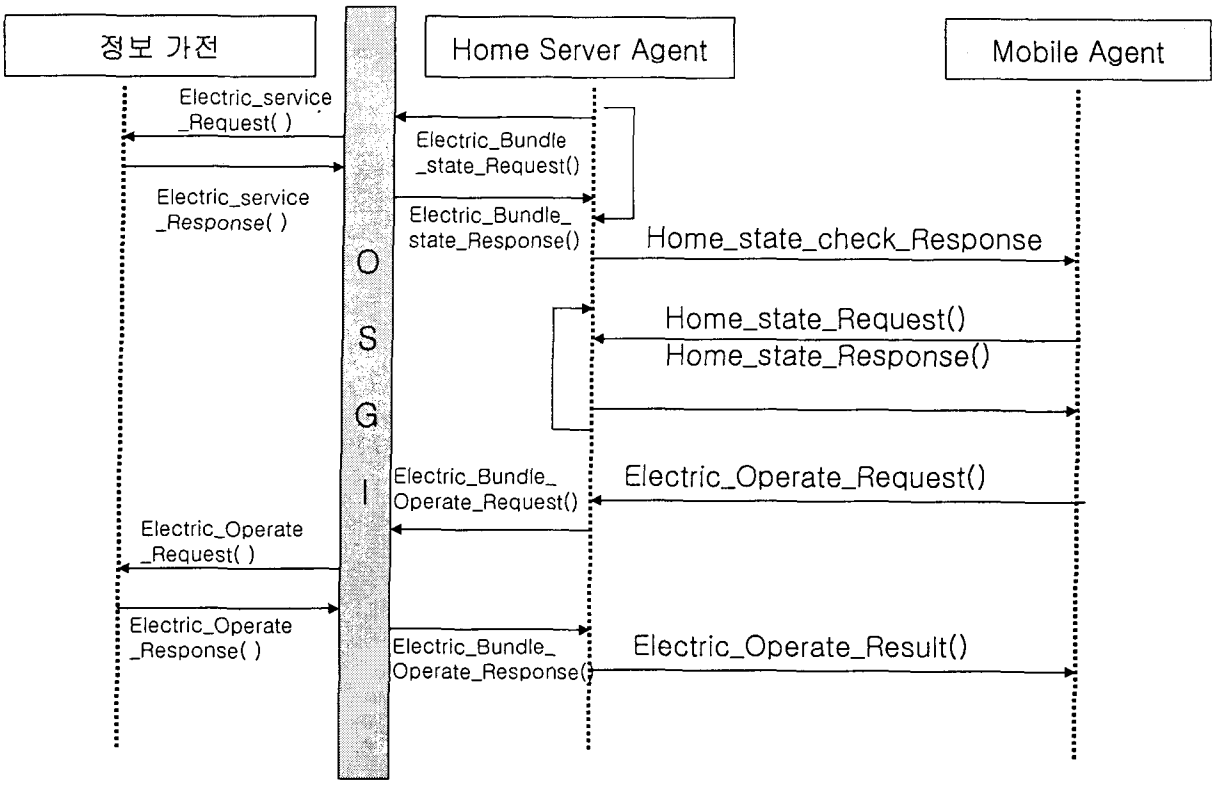
```
(request
:sender
(agent-identifier :name dummy@foo.com
:addresses (sequence iiop://foo.com/acc))
:receiver (set (agent-identifier :name ams@foo.com
:addresses (sequence iiop://foo.com/acc)))
:language FIPA-SL0
:protocol FIPA-Request
:ontology FIPA-Agent-Management
:content (action (agent-identifier
:name ams@foo.com
:addresses (sequence iiop://foo.com/acc))
(register (ams-agent-description
:name (agent-identifier
:name dummy@foo.com
:addresses
(sequence iiop://foo.com/acc))
:state active))))
```

그림 8 ACL Message

또한 FIPA-OS는 기본적으로 ACL 메시지를 처리하는 API를 제공함으로서 그림과 같이 작성하지 않고 ACL 메시지의 쉽고 안전하게 에이전트 간 통신을 할 수 있다.

3.3 프로그램 흐름도

프로그램의 흐름을 표현한 그림 9는 크게 두 부분으로 나누어 살펴 볼 수 있다.
하나는 두 에이전트간의 메시지의 흐름이고, 또 하나는 가정의 정보가전의 제어를 담당하고 홈 서버 에이전트와 정보가전 사이에 OSGi를 통한 메시지 전달 과정을 살펴 볼 수 있다.
모바일 에이전트는 홈 서버에 에이전트에게 일정한 간격으로 집안의 정보 가전에 상태에 대한 메세지를 보내며 홈 서버 에이전트는 OSGi를 통하여 집안의 정보가전에 대한 상태를 체크한 후 모바일 에이전트에게 응답한다.
또한 홈 서버 에이전트는 일정 간격으로 OSGi를 통하여 집안의 정보 가전에 대한 상태를 체크하는 메시지를 보내고, 만약 변화가 있을 경우 변화된 상태를 모바일 에이전트에게 전달한다.
그리고 모바일 에이전트는 홈 서버 에이전트로 사용자의 정보가전에 대한 제어를 요청하고 홈 서버 에이전트는 OSGi를 통하여 정보 가전을 제어 후 그 결과를 모바일 에이전트에게 통보한다.



4. 정보 가전 제어 멀티 에이전트

4.1 HomeServer 에이전트

홈 서버 에이전트는 집안의 정보 가전를 직접 관리하는 핵심 에이전트이다.

일정한 간격으로 정보가전의 대한 상태를 체크하며, 변화된 사항에 대해 모바일 에이전트에게 통보하기도 하고, 일정 간격으로 모바일 에이전트가 집안에 상태에 요청이 오면 이에 대해 응답함으로서 사용자가 집안의 상태를 알 수 있도록 한다.

또한 사용자의 복잡한 요구에 대해서 정보가전을 관리하고, 그 상태를 모바일 에이전트에게 전달하게 된다.

홈 서버 에이전트 구성도를 보면(그림10 참조) 크게 사용자 GUI 부분과 실제 구동되는 프로그램 부분으로 나누어진다.

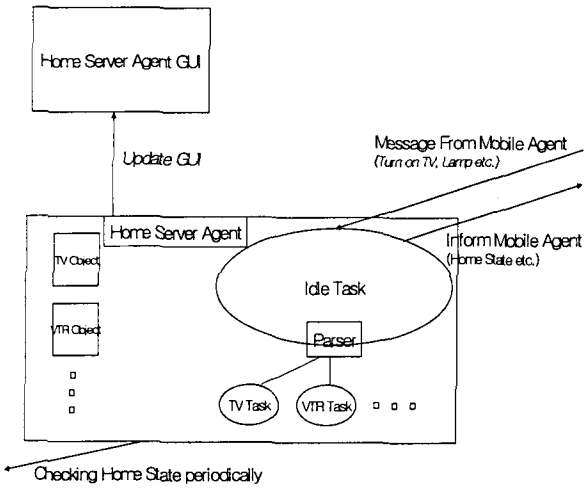


그림 10. 홈 서버 에이전트 구성도

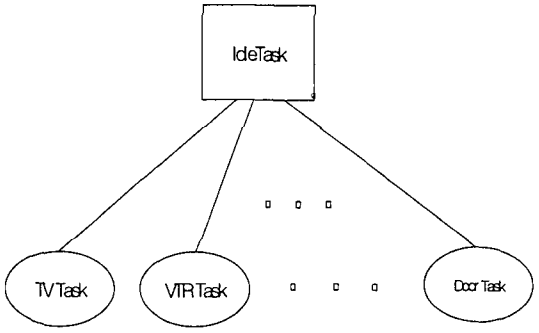


그림 11. Task 구성

각각의 정보가전마다 Task란 단위로 나누어져 있고, Idle 태스크가 각각의 태스크를 전체적으로 관리하며 모바일 에이전트로부터 전달되는 메시지에 대해서 분석하여 해당되는 태스크가 정보가전의 관리를 담당하도록 한다.(그림 11 참조)

홈 서버 에이전트는 모바일 에이전트가 서비스 받

고자 하는 정보가전 상태를 체크하고 있고, 모바일 에이전트의 요청 시에 상태에 대한 정보를 보낸다. 또한 모바일 에이전트의 요청에 따라 정보가전을 직접 제어 후 그 상태를 모바일 에이전트에게 통보한다. 그리고 주기적으로 체크하면서 집 안의 정보가전의 상태가 변화하면 변화된 정보를 모바일 에이전트에게 보낸다.

홈 서버 에이전트 FIPA-OS플랫폼 위에서 동작하는 에이전트로서, OSGi를 통하여 정보가전을 제어하며 플랫폼이 Java로 구현되었기에 Java로 제작하였다. 다른 에이전트와는 ACL 메세지를 통하여 통신한다.

4.2. Mobile 에이전트

모바일 에이전트는 원격지에서 PDA상에 작동하는 에이전트로 사용자의 정보가전에 대한요구를 홈 서버 에이전트에게로 전달하며, 집안의 정보가전에 상태를 홈서버 에이전트를 통해서 전달받아 사용자에게 보여준다.

모바일 에이전트는 전기량, 수도량 제어, TV, VTR 동작 제어, 냉장고제어, 램프 제어 및 가스 조절기, 출입문 제어등 여러 가지 정보가전에 대해서 Setting을 통하여 사용자가 원하는 정보가전을 선택하게 하고, 구체적인 서비스에 대해서도 사용자가 선택하게 함으로서 최적의 모바일 에이전트로 구축하게 된다.

한마디로 사용자가 간단한 설정을 통하여 모바일 에이전트 내부에 또 하나의 에이전트를 만드는 것이다.

이것으로 사용자는 불필요한 서비스를 제거하고, 시스템의 부하를 줄일 수 있으며, 나만의 에이전트를 갖게 되는 것이다.

모바일 에이전트내의 프로그램에서 사용자의 설정에 의해 선택된 태스크는 해당하는 메시지를 홈 서버 에이전트로 전달하게 된다.

모바일 에이전트는 홈 서버 에이전트에게 자신이 설정한 정보가전에 대한 상태에 대해 요청하게 된다. 이를 감지한 홈 서버 에이전트는 모바일 에이전트에게 집안의 정보 가전에 대한 상태를 전달하게 된다. 또한 모바일 에이전트는 정보가전에 대한 제어를 요청하고 그 요청에 대한 결과를 홈 서버 에이전트에게서 전달 받는다.

5. 결 론 및 향후 과제

본 논문은 정보가전을 관리하기 위하여 멀티 에이전트 시스템을 사용하였고, 그 설계 및 구현 방법에 대해 기술하였다.

집안의 가전 제품을 외부에서 제어하는 소프트웨어는 이미 많이 나와있다. 하지만 지능적이고 사용자 요구에 맞는 소프트웨어는 부족하다. 이 논문에서는 FIPA-OS 플랫폼 상에서의 멀티에이전트 시스템을 갖는 장점을 이용하여 사용자의 복잡한 요구에 대해 원할히 처리할 수 있도록 하였으며, 사용자가 원하는 에이전트를 직접 구축할 수 있도록 하여 좀 더 사용자 중심의 서비스를 할 수 있도록 하였다.

정보가전 제어 멀티 에이전트시스템의 장점은 FIPA에서 제시한 규격을 따르기 때문에 FIPA 규격에 맞춰서 개발된 다른 에이전트와의 통신도 자유롭다는 것이며, 사용자의 요구에 맞는 에이전트를 만들 수 있다는 것이다. 하지만 복잡한 사용자의 요구에 대해서 홈 서버 에이전트가 모든 제어를 담당 할 수 있도록 하여 홈 서버의 기능을 극대화하는 것이 앞으로의 과제라 할 수 있을 것이다.

참고문헌

인터넷 정보가전, 한국전산원웹진-인포진 http://www.nca.or.kr/main/ncadata/newsletter/2000/12/opinion.html, 2000년 12월

전호인, 송동일, 정보가전을 위한 IEEE1394 기술의 적용방안, 정보처리 제8권 제 1호, 2001

박성호, 강순주 홈 네트워크에서 네트워크와 데이터 네트워크의 상호연동을 지원하는 미들웨어 구조, 정보과학회지 제19권 제 4호 2001

The Foundation for Intelligent Physical Agents, FIPA Specification 2000, http://www.fipa.org

Emorphia, FIPA-OS(Open Source) Tutorial, http://fipa-os.sourceforge.net

Emorphia, Micro FIPA-OS User Guide, http://fipa-os.sourceforge.net

Open Services Gateway Initiative http://www.osgi.org