유비쿼터스 환경에서의 FIPA 기반 에이전트 플랫폼을 이용한 서비스 디스커버리[1)]

김형준*, 이규민*, 최기현*, 신동렬*
*성균관대학교 정보통신공학부
e-mail:{mairi, kmlee, gyunee, drshin}@ece.skku.ac.kr

Service Discovery Using FIPA-Compliant Agent Platform in Ubiquitous Environments

Hyung-Jun Kim*, Kyu Min Lee*, Kee-Hyun Choi* and Dong-Ryeol Shin*
*School of Information and Communication Engineering, SungKyunKwan University

요 약

에이전트 시스템(agent system) 관련 기술은 이질적이고 분산된 환경에서 상호 운용성(interoperability)을 확보하기 위한 방법이 될 수 있다. 이런 에이전트 시스템의 국제 표준화를 진행해 나가고 있는 단체인 FIPA(Foundation for Intelligent Physical Agents)는 1996년 에 형성되었고, 이질적인 시스템간의 상호 운용성을 최대화하기 위해서 국제적으로 인정된 명세서를 정기적으로 발표하고 있다. 하지만 최근까지 FIPA 표준안은 모바일 ad-hoc네트워크와 같은 환경을 고려하지 못하고 있다. 이러한 환경에서 에이전트는 이질적인 네트워크들에서 제공하는 서비스들을 이용할 수 없다. 이러한 문제점에 초점을 맞추어 본 논문에서는 FIPA 표준을 참고하여 만든 FIPA-OS(Foundation for Intelligent Physical Agents Open Source) 를 수정한 에이전트 플랫폼을 제공한다. 우리는 이러한 에이전트 플랫폼을 이용하여 이질적인 환경에서 제공하는 서비스들에 대한 상호운용성과 에이전트 플랫폼 안에 DM(Discovery Middleware)을 추가하여 확장성을 보장한다. DM은 에이전트가 ad-hoc 네트워크 안에서 동작하는 서비스 디스커버리 기법들을 사용하여 서비스들을 찾고 생성할 수 있게 도와준다. 우리는 다양한 서비스 디스커버리 기법 중 UPnP (Universal Plug and Play)와 LSD (Lightweight Service Discovery Protocol)를 이용하여 DM을 구현하였다. UPnP는 UPnP 포럼에서 개발된 서비스 디스커버리를 위한 프로토콜의 집합이며 LSD는 우리가 만든 모바일 ad-hoc 네트워크에서 동작하는 새로운 서비스 디스커버리 프로토콜로 캐쉬 관리를 강조하여 개발하였다. 우리가 제안하는 DM을 이용하여 수정된 에이전트 플랫폼에서 UPnP와 LSD의 장비에서 제공되는 서비스들을 상호간 이용 할 수 있다.

1. 서론

몇 년 전부터 서비스 디스커버리 분야에 대한 많은 연구들이 이루어지고 있고, 여러 논문들이 출간되고 있다. 비록 이 논문들이 서비스 디스커버리에 대한 효율적이고 빠른 탐색 기법들을 제안하고 있지만, 그 기법들은 이기종의 디스커버리 프로토콜 사이에서의 상호 호환성을 제공하지는 못한다. 본 논문에서 우리는 크게 확장된 또는 확장 가능한 네트워크에서의 서비스 디스커버리에 초점을 맞춘다. 유비쿼터스 환경에서는 이기종의 서비스일지라도 사용자들이 쉽게 이용할 수 있는 인터페이스 및 기법들이 필요하다. 현재의 서비스 디스커버리 프로토콜들은 단지 자신의 영역 혹은 자신 과 같은 프로토콜일

1) 본 논문은 한국 CUCN의 지원으로 연구되었음.

때만 서비스를 발견하고, 사용할 수 있게 한다. 이러한 이유 때문에 우리는 확장성과 이기종의 서비스들을 사용할 수 있게 하는 FIPA 표준에 기반으로 한 새로운 에이전트 플랫폼을 제안한다.

FIPA는 이기종 환경에서의 에이전트들 간의 호환성을 제공하기 위해 세계 표준을 제시하고 있는 비영리 단체이다. 우리는 FIPA 표준안에 기반으로 만들어진 FIPA-OS 에이전트 플랫폼을 수정하여 보다 향상되고, 발전된 에이전트 플랫폼을 개발하였다. 기존의 에이전트 플랫폼과의 두드러진 차이점은 에이전트 플랫폼에 DM (Discovery Middleware) 모듈을 추가함으로써, ad-hoc 환경에서 제공하는 서비스들을 발견하고, 사용자들이 이용할 수 있게 한다. 현재 다양한 서비스 디스커버리 프로토콜들이 존재하고 있는데, 이중에서 DM의 구현부분으로 2가지를 선택하였다. 그 첫 번째로 최근 많이 사용되고 있는 UPnP (Universal Plug and Play)를, 두 번째로는 우리가 개발한 LSD이다. LSD는 ad-hoc 환경에 산재되어 있는 서비스들을 이용 할 수 있게 하기 위해 고안한 경량의 서비스 디스커버리 엔진이다.

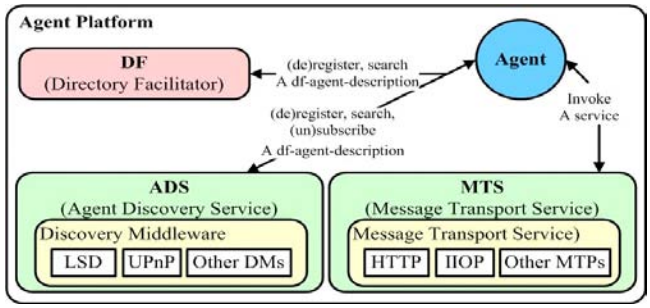
본 논문에서는 다음과 같이 내용이 전개 되어 진다. 섹션2에서는 LSD 아키텍처를 포함한 DM 기술들을 제공하고, 섹션 3에서는 FIPA 에이전트 서비스 디스커버리 아키텍처와 시나리오를 보여준다. 섹션 4에서는 실제 개발한 구현과정을 나타내어 주고, 섹션 5에서는 결론을 정리한다.

2. 관련 연구

2.1 FIPA의 에이전트 플랫폼

FIPA 에이전트 플랫폼 (AP)은 에이전트들로 구성되어 있다. 노드들이 자주 이동하는 ad-hoc 환경에서의 디스커버리 기능을 제공하기 위해서 에이전트 플랫폼은 에이전트 디스커버리 서비스 (ADS)모듈을 제공한다. ADS는 FIPA 아키텍처 내에 위치하고 있는 AP 컴포넌트로서, MTS와 같은 레벨에 위치하고 있다. 그것은 에이전트들에게 DF와 비슷한 인터페이스를 제공한다. 그림 1은 ADS가 FIPA 아키텍처에서 어디에 위치하고 있는지를 보여준다.

ADS는df-agent-descriptions (DADs)의 등록과 탐색을 지원하는 DF같은 인터페이스를 제공한다. 하지만, DF와는 대조적으로 ADS는 DAD 탐색 템플릿과 일치되는 DADs를 예약 탐색하는 (subscribe) 기능을 제공한다. 예약 탐색이 된 이후로, 탐색 기준을 만족하는 각각의 새롭게 등록된 DAD는 예약 탐색 해제 (unsubscription)가 될 때까지 에이전트에게 통보될 것이다. 이 기능들은 이용가능한 모든 DM들에 적용될 수 있다. DM은 에이전트 플랫폼에 존재하는 에이전트들이 ad-hoc 네트워크에 산재되어 있는 서비스들을 발견하고 이용할 수 있도록 돕는다. 앞에서 언급한 ADS의 기능을 가능하게 하기 위해, 각각의 DM들이 필요하다.



(그림1) FIPA 에이전트 플랫폼 구조

2.2 DM에 적합한 서비스 디스커버리 기술들

DM으로서 사용될 수 있는 많은 서비스 디스커버리 기술들이 있다. 본 논문에서 우리는 그것들 중에 대표적인 2가지 기술들을 요약해 보겠다.

UPnP는 Universal Plug and Play 포럼에 의해 개발 된 서비스디스커버리를 위한 프로토콜이다. 그것은 장비와 control point로 구성되고, 장비의 형태와 기능들을 설명하기 위해 XML을 사용한다. 장비의 advertisement 메시지들은 control point에게 XML 파일의 위치를 나타내는 URL을 제공한다. 이 XML파일을 통하여, control point는 장비의 정보를 알 수 있다.

LSD는 모바일 ad-hoc 네트워크를 위해 우리가 설계한 새로운 디스커버리 프로토콜이다. 서로 무선 네트워크로 연결되고, 작고 이동성을 가진 장치들로 구성된 ad-hoc 네트워크는 기존의 인프라스트럭처 기반의 환경과는 차이점을 가지고 있다. 우리는 ad-hoc 네트워크에 있는 서비스들을 발견하고, 이용하기 위해서 LSD를 설계하고 구현하게 되었다. 다음 2.3에서 보다 자세히 LSD 아키텍처에 대해 알아보겠다.

2.3 DM#1: UPnP, DM#2: LSD

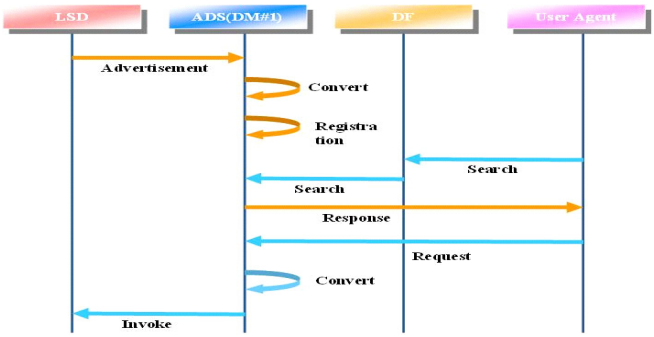
사용자 또는 에이전트는 ad-hoc과 P2P 환경 안에서 존재하는 서비스 디스커버리 엔진들을 에이전트 플랫폼안의 모듈로써 추가되는 DM을 사용한다. 다양한 서비스 디스커버리 기술들이 에이전트 플랫폼에 DM으로 추가될 수 있다. 본 논문에서 우리는 이

동성(mobility)을 가지고 ad-hoc 환경에 적합한 기술들 중에 두 개를 선택하였다. 첫 번째 DM은 UPnP로 다른 하나는 LSD로 선택하여 개발하였다. 이미 UpnP는 많은 분야에서 활용되고 있고 우리가 자체 제작한 LSD는 캐쉬와 delta 메시지를 사용하여 ad-hoc 환경에서 효율적으로 동작하는 서비스 디스커버리 엔진이기 때문이다.

3. FIPA 에이전트 서비스 디스커버리

3.1 시나리오

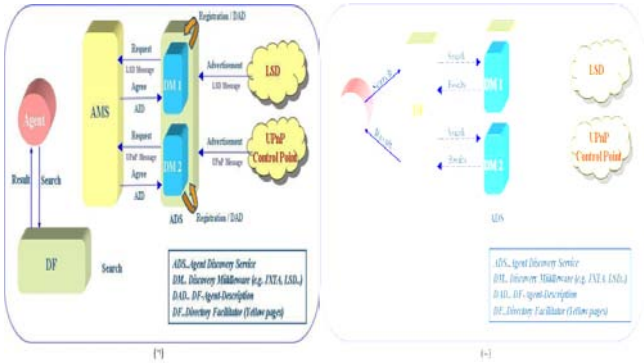
우리의 목표는 에이전트 플랫폼을 사용하여 LSD와 UPnP 사이의 상호운용을 실현시키는 것이다. 이러한 상호운용을 실현하기 위해서는 LSD와 UPnP를 DAD로 변환 시키는 DM 모듈이 필요하다. 그림 2는 LSD, DM, 사용자 에이전트 사이의 내부 동작 과정을 보여준다.



(그림2) LSD, DM#1, 사용자 에이전트 간의 상호동작

먼저 LSD의 장비가 광고 메시지를 멀티캐스트 한 것이 에이전트 플랫폼안의 DM#1에게 전달된다. 그리고 나서 DM#1은 변환 모듈을 이용하여 수신된 메시지를 DAD로 변환시키고 그것을 저장한다. 이제 사용자가 특정한 서비스를 찾고 사용하기를 원하게 되면 사용자 에이전트는 먼저 DF로 가서 해당 서비스를 찾게 되고 DF에 존재하지 않으면 ADS에게 질의하게 된다. 원하는 서비스를 DM#1에서 찾게 되면 사용자 에이전트는 해당 서비스를 이용하고 싶다는 요청 메시지를 보내게 되고 DM#1은 이를 다시 LSD내부에서 동작하는 메시지 형식으로 변환한 후 LSD의 해당 장비에게 전달한다. 비슷한 방식이 UPnP의 경우에도 적용된다.

LSD의 경우와 다른 것은 UPnP는 중앙의 관리 모듈인 UPnP Control Point를 가지고 있다는 점이다. 모든 UPnP 장비들은 자신의 정보를 UPnP Control Point에 등록하게 되고 각각의 장비의 존재를 UPnP Control Point가 ADS에게 알려주면 이를 변환하여 DM#2에 저장하게 된다. 그 후의 동작은 LSD의 경우와 비슷하다. 마지막으로 우리의 에이전트 플랫폼 안에서의 서비스 등록과 디스커버리 시나리오를 그림 3 (ㄱ), (ㄴ)에서 나타내고 있다.



(그림3) 서비스 등록(ㄱ), 서비스 디스커버리(ㄴ)

그림 3 (ㄱ)에서 보듯이 서비스 등록 절차는 변환 모듈이 필요하고 이를 위해 DM은 AMS에게 들어온 메시지를 변환하겠다는 메시지를 보내고 AMS로부터 허가 메시지를 수신하는 추가적인 동작이 필요하다. 수신된 메시지에는 에이전트 ID값이 포함되어 있으며 이를 이용하여 수신된 메시지는 DAD 형태로 저장되게 된다. 서비스 등록절차에 비해 서비스 디스커버리 절차는 간단하다. 이 과정에서는 사용자 에이전트가 DAD를 사용하여 서비스를 검색한다는 것이 중요한 점이다.

3.2 DM 변환 알고리즘 - LSD와 UPnP

그림 4는 LSD와 UPnP에 관한 변환 알고리즘을 pseudo 코드로 요약한 것이다.

```
receive_msg(msg, aid) {
   Message_format buf = new Message_format();
   While(read msg until End_of_msg) {
      Ontology = getElementsByTagName
      ('DeviceType');
      Service_name = getElementsByTagName
      ('friendlyName');
      Service_type = getElementsByTagName
      ('serviceType'); }
   Buf.append(ontology);
   Buf.append(Service_name);
   Buf.append(Service_type);
   Buf.append(aid);
   Send buffer to dm@localap;}
   Message_format {
      // predefine a DAD's form }
```

(그림4) 변환 알고리즘

만약 DM이 LSD의 장비나 UPnP Control Point로부터 메시지를 수신하면 DM은 에이전트 플랫폼 내부에서 사용되는 DAD로 변환하여야 한다. 해당 모듈에는 인자 값으로 처음 수신된 메시지와 함께 AMS로부터 받은 에이전트 ID (AID)를 받게 되고 ADS는 이들을 사용하여 변환 모듈을 수행하게 된다. 먼저 수신된 메시지를 통하여 ontology, service_name, service_type 요소들을 추출해내고 그 다음에 이들과 인자 값으로 받은 AID를 미리 정의해둔 DAD 형태의 버퍼에 저장함으로써 에이전트 플랫폼 내부에서 동작하는 DAD를 완성하게 된다. LSD와 UPnP의 광고 디스크립션이 매우 비슷하기 때문에 그림 4의 pseudo 코드로 두 경우를 모두 설명할 수 있다.

4. 구현

우리는 동적이고 적응적인 에이전트 플랫폼을 위하여 FIPA-OS를 수정하였고 LSD와 UPnP를 DM으로써 플랫폼 내부에 추가하였다. 그림 5(ㄱ)은 PDA에서 동작하고 있는 우리의 LSD 엔진을, 그림 8(ㄴ)는 테스트에 사용하기 위해 제작된 UPnP Control_Point와 몇 가지 UPnP 장비의 구현 화면을 보여준다.

(그림5) LSD 엔진(ㄱ), UPnP Control Point와 장비(ㄴ)

LSD는 Personal Java 1.2와 J2ME CLDC/MIDP를 이용하여 구현하였다. 이 구현을 위하여 우리는 LG IBM Thinkpad와 WinCE based's from Compaq의 iPAQ기반의 WinCE를 사용하였다. 아직 LSD의 내부 컴포넌트들은 미 구현된 부분이 존재하며 우리는 계속하여 해당 부분의 구현을 해나갈 것이다. 또한 우리는 UPnP v1.1 표준에 맞게 UPnP Control Point와 몇 가지 장비들을 어플리케이션으로 구현하였고 현재 UPnP 포럼 웹사이트인 www.upnp.org에서 UPnP 1.0표준이 이용 가능하다.

5. 결론

우리는 유비쿼터스 환경에서 동적이고 적응적인 에이전트 플랫폼을 위하여 FIPA-OS 플랫폼을 수정하였다. 그리고 나서 ad-hoc 환경에서 동작하는 다양한 서비스 디스커버리 기술들 중에 LSD와 UPnP를 선택하여 DM으로써 추가하였다. 이러한 과정을 통하여 우리는 UPnP의 장비로부터 제공되는 서비스와 ad-hoc 환경에서 효과적으로 동작하도록 자체 제작한 LSD에서 제공되는 서비스들을 에이전트 플랫폼에서 이용할 수 있게 되었다. DM을 추가하였다는 것은 이질적인 환경에서 제공되는 서비스들을 이용할 수 있게 되어 상호 운용성을 보장한다는 점과 언제든지 새로운 DM을 추가함으로써 확장성을 제공할 뿐만 아니라 시스템 점유 용량을 최적화 할 수 있다. 이러한 DM은 개발자의 입장에서는 다른 환경을 고려할 때 에이전트 플랫폼 안에 하나의 DM을 추가하기만 하면 되므로 개발이 용이하다는 것과 사용자 입장에서 볼 때 서비스가 어디에 있든지 편하게 이들을 이용할 수 있다는 점을 장점으로 들 수 있다. 앞으로 우리는 DM에 보다 많은 서비스 디스커버리 프로토콜을 적용함으로써 연구분야를 추가/확장시켜 나갈 것이다.

참고문헌

[1] Sumi Helal, “Standards for service discovery and delivery, Pervasive Computing, IEEE, Volume” 1,3,Pages:95 - 100 , July-Sept. 2002

[2] FIPA, “Agent Discovery Service Specification” Oct. 2003

[3] Universal Plug and Play Specification, v1.0, http://www.upnp.org.

[4] Jae-Wan Park, Byung-in Lim, Kee-Hyun Choi and Dong Ryeol Shin “Design and Implementation of Light-weight Service Discovery System for Ad-Hoc Environments”ICACT 2005, Feb. 2005