프레즌스 및 openAPI를 활용한 URC서비스 플랫폼

URC Service Platform based on presence and openAPI

배 정 일[1], 김 동 훈[1], 이 현 주[1], 연 승 호[1]

Bae Jeong-Il[1], Kim Dong-Hoon[1], Lee Hyun-Joo[1], Yeon Seung-Ho[1]

Abstract Combining robot and network gives us many advantages like lightweight hardware specification of robot, a various robot service, simple upgrade of robot, easy management and so on. Among these advantages, Presence service and openAPI are most important. Presence is simple but powerful service. It makes user to know the status information of robot and enables user to control robot from a remote place. OpenAPI which is also a feature of WEB2.0 enables 3rd parties to make a various mashup service easily and rapidly. Finally presence and openAPI can help URC service to be ubiquitous and successful.

Keywords : URC, Service Platform, Presence, openAPI

1. 서 론

지능형 로봇은 성장 잠재력이 매우 큰 분야로서 미국, 일본 등 로봇 선진국에서는 일찌감치 다양한 형태의 연구 및 산업화를 추진해 왔으며, 국내의 경우에는 최근 정부의 주도 및 지원 하에 우리나라 최고 강점중의 하나인 IT기술을 로봇과 결합시킨 네트워크 로봇(URC : Ubiquitous Robotic Companion)의 형태로 추진해 나가고 있다. URC는 로봇플랫폼에 통신기능을 장착하고 원격지의 서버(서비스 플랫폼)와 네트워크를 통해 연결되어 사용자가 원하는 다양한 서비스를 제공할 수 있도록 해준다.

이렇게 로봇과 통신 기능을 결합함으로써 로봇 플랫폼의 경량화, 제공 서비스의 다양성, 기능 업그레이드 등의 융통성, 관리의 용이성 등 얻을 수 있는 장점은 매우 많다. 그러나 이러한 기능적인 측면 이외에 무엇보다 가장 큰 장점은 첫째 언제 어디서라도 로봇의 상태(Presence)를 확인하고 이를 바탕으로 로봇을 제어하는 등 유비쿼터스 로봇 서비스를 제공 할 수 있다는 것이며, 둘째 네트워크를 통해 개방된 형태(openAPI)로 특정 로봇 제조업체, 서비스 제공업체, 컨텐츠 제공업체(CP, Contents Provider)에 얽매이지 않고 다양한 형태의 서비스 재창출이 가능하여 이를 통해 로봇 서비스를 더욱 더 풍부하게 키워 나갈 수 있다는 것이다.

본 고에서는 로봇이 통신 기능을 통해 얻을 수 있는 가장 핵심적인 두 가지 요소인 프레즌스와 openAPI를 소개하고 이들을 URC 서비스에 적용한 사례 및 향후 전망에 관하여 기술하고자 한다.

2장에서는 URC 서비스 플랫폼의 기능과 구조에 대해서 소개하고, 3장과 4장에서는 이들 중 로봇 서비스의 유용성과 확장성을 극대화 시킬 수 있는 프레즌스 및 openAPI 기능과 그 구현 사례에 대해서 각각 기술하며, 마지막으로 결론에서는 이러한 측면에서 향후 기대되는 사항을 정리하였다.

2. URC 서비스 플랫폼 개요

이 장에서는 네트워크를 통해 로봇과 유기적으로 연결되어 다양한 서비스를 제공하는 URC 서비스 플랫폼의 주요 기능 및 구조에 관하여 기술한다.

2.1 개요

URC 서비스 플랫폼은 로봇의 두뇌와 같은 역할을 수행하게 된다. 즉 로봇에 전달된 사용자의 입력이나 로봇이 센싱한 정보들을 네트워크를 통해

※ 본 연구는 정토통신부 및 정보통신연구진흥원의 IT 신성장 동력 핵심 기술 개발 사업의 일환으로 수행하였음[2006-S-027-02, URC 응용을 위한 가용성 및 신뢰성 고도화 기술].

[1] KT 인프라연구소

획득하고 이를 바탕으로 적정한 동작 및 서비스 등을 로봇에 전달하는 것이다. 특히 음성 인식, 합성, 동영상 변환 등 다양한 서비스를 처리하기 위해서는 적지 않은 처리 능력이 요구되는 데 이러한 복잡한 기능을 URC 서비스 플랫폼에서 처리해 줌으로써 로봇의 경량화, 저가화가 가능해지고 이는 로봇의 보급 확대와도 직결되는 것이다.

2.2 주요 기능 및 구조

URC 서비스 플랫폼이 로봇에 서비스를 제공하기 위해서 요구되는 기능은 많지만 간략히 표현하면 다음과 같다.

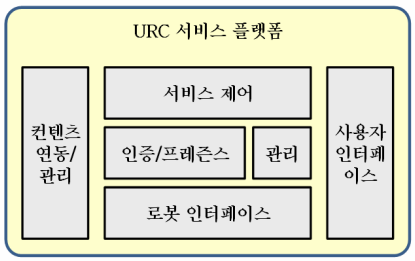

그림 1. URC 서비스 플랫폼 기능 구조

로봇 인터페이스 기능은 말 그래도 네트워크를 통해서 로봇과 통신하는 부분으로서 로봇으로부터의 다양한 정보 및 서비스 요구를 수신하고 해석한 뒤 그것에 대한 적정한 처리 후 로봇으로 응답을 송신하는 부분이고, 인증/프레즌스 기능은 로봇 인터페이스 부분으로부터 수신된 정보가 적정한 권한을 가진 로봇으로부터 송신 되었는지 여부를 확인하고 수신된 정보로부터 해당 로봇의 상태(프레즌스) 정보를 추출 및 관리하는 기능을 수행하는 부분이고, 컨텐츠 연동/관리 기능은 외부 CP와 openAPI를 통해 연동하여 컨텐츠의 정보 또는 컨텐츠 그 자체를 획득하고 관리하며, 서비스 제어 기능은 로봇 인터페이스와 컨텐츠 연동/관리 기능의 중간에서 로봇에 컨텐츠를 활용한 서비스를 제공하는 기능을 수행한다. 이외 로봇의 관리를 담당하는 기능과 웹 등을 통해 사용자와의 인터페이스를 제공하는 기능 등을 URC 서비스 플랫폼은 보유하게 된다.

3. URC 프레즌스 서비스

이미 프레즌스라는 개념은 누구나 사용하는 인스턴트 메신저에서부터 VoIP에 이르기까지 다양한 분야에 적용되고 있으며 간단한 개념이기는 하지만 그 활용도는 매우 높다. 이 장에서는 프레즌스의 개념 및 이를 활용한 URC에서의 적용 및 응용 방안에 대하여 기술한다.

3.1 프레즌스의 개념 및 서비스 모델

프레즌스(Presence)란 해당 엔터티의 상태를 나타내는 것으로 현재 통신이 가능한 상태인지, 접속 가능한 주소는 어떤 것인지, 어떤 서비스의 사용이 가능한지, 현재 무엇을 하고 있으며, 기분 상태는 어떠한지 등에 대한 다양한 정보(Presence Tuple)를 의미한다. 프레즌스 서비스(Presence Service)는 네트워크 상에 떨어져 있는 특정 사용자의 프레즌스 정보를 제공하고/받는 서비스를 말하며 일반적으로 다음과 같은 구조를 지닌다.

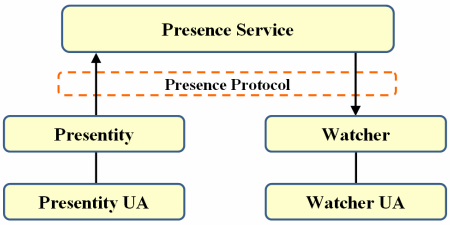

그림 2. 프레즌스 서비스 모델

프레즌스 서비스 모델[1]은 프레즌스 정보의 원천인 프레젠티티(Presentity, Presence Entity)와 프레즌스 정보를 요구하고 수신하는 와처(Watcher), 그리고 프레젠티티가 제공하는 프레즌스 정보를 저장하며 와처에게 프레즌스 정보를 전달하는 프레즌스 서비스(presence service)의 세가지 요소로 구성된다. 프리젠티티와 와처는 유저에이전트(UA)를 통해 사용자 및 응용프로그램 등과 교신하게 된다.

프레즌스 정보를 송수신하는 프로토콜은 다양한 형태로 구현이 가능하나 일반적으로 프리젠티티가 자신의 프레즌스 정보[2]를 알라는 publish[3], 와처가 프레즌스 정보를 요청하는 subscribe, 와처에게 프레즌스 정보를 전달하는 notify 등으로 구성된다. 이외에도 프레즌스 정보를 서버에 저장하고 관리하기 위하여 별도의 추가적인 프로토콜을 사용하기도 한다.

3.2 URC 서비스 플랫폼에서 프레즌스 기능의 구현

프레즌스를 URC 서비스에 적용하기 위하여 먼저 다루어져야 하는 프레즌스 정보로서 접속주소, 접속상태, 로봇 유형, 기분(Mood) 상태 등을 정의

하였으며, 프레즌스 정보의 송수신을 위한 프로토콜은 구현 대상 로봇이 Embedded OS를 사용하는 비교적 낮은 성능임을 고려하여 SIP 등 표준화된 프로토콜이 아닌 UDP를 기반으로 한 binary format의 light-weight한 형태로 publish, subscribe, notify 등의 기능을 보유한 프로토콜을 디자인하여 적용하였다.

로봇으로부터 프레즌스 정보를 수집, 관리하기 위하여 URC 서비스 플랫폼에는 다음과 같은 구조로 프레즌스 기능을 구현하였다.

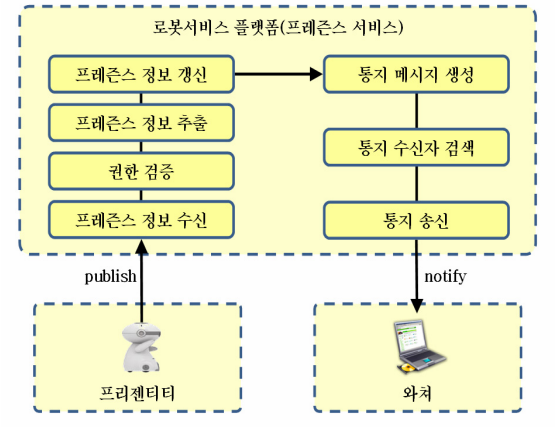

그림 3. URC 서비스 플랫폼의 프레즌스 서비스 구현

프리젠티티(로봇)가 보낸 프레즌스 정보 메시지를 수신하게 되면 이것이 적정한 프레즌티티로부터 송신되었는지의 검증을 거친 후 프레즌스 정보를 추출한다. 이를 내부의 데이터베이스 등에 반영한 후 해당 프레즌스 정보를 요구하고 있는 와처(로봇 또는 사용자)가 있는지를 찾아서 통지(notify) 메시지를 생성하고 통지 메시지의 수신자 주소를 검색한 뒤 그곳으로 송신하게 된다. 이로서 로봇의 프레즌스는 서비스 플랫폼에 저장되고, 이를 원하는 개체들에게도 전송이 될 수 있게 되는 것이다.

4. OpenAPI의 제공

3rd Party 및 사용자들의 참여를 어떻게 유도하느냐가 성공적인 서비스의 지름길이 되고 있는 상황에서 openAPI를 제공하는 것은 이제 어느 포털에서나 당연스러운 일로 받아들여지고 있으며[4], 또한 URC 산업에 있어서도 openAPI는 서비스의 활성화에 있어 중요한 비중을 차지한다고 할 수 있다.

4.1 OpenAPI의 개념 및 특징

openAPI는 전통적인 API(Application Programming Interface)의 개념을 웹으로 확장한 것이다. 특정한 기능이나 서비스를 제공하는 서버 또는 사이트에 접속해서 필요한 데이터를 요청하고 받아오는 행위를 기존 컴퓨터에서 함수를 호출하고 결과를 받는 것과 동일하게 생각하는 것이다. 예를 들어, URC 서비스 플랫폼이 3rd party 및 사용자들을 위해 로봇의 현재 접속 상태 정보를 얻을 수 있는 openAPI를 만들었다면, 다른 사이트에서는 별다른 노력 없이 URC 서비스 플랫폼에서 로봇의 접속 상태 정보를 얻어와 고객들에게 보여줄 수 있다. “open”의 의미는 인터넷 상의 API라는 뜻이기도 하지만, 독점적인 정보를 외부로 열어주겠다는 뜻도 포함되어 있는 것이며 이를 통해 보다 풍성하고 새로운 서비스를 창출할 수 있는 것이다.

그리고 SOAP[5], REST 등의 openAPI 방식은 JAVA, .NET 등 플랫폼에도 독립적이고 방화벽에도 제한을 받지 않으므로 개발 및 운용관리도 용이하다.

4.2 OpenAPI 구현 방법

일반적으로 openAPI를 구현하는 방식에는 보통 웹서비스로 일컫는 SOAP(Simple Object Access Protocol)과 REST(Representational State Transfer)가 있다. SOAP은 기존의 RMI, CORBA, DCOM 등의 단점을 극복하고자 설계된 것으로 HTTP등의 전송프로토콜을 사용하여 XML로 정의되어 있는 메소드(또는 함수)를 호출하는 형태이다. SOAP은 메소드의 명칭, 데이터 타입 등의 상세 정보를 기술하는 WSDL(WebService Description Language) [6]을 가지고 손쉽게 구현할 수 있는데, 이 SOAP과 WSDL은 W3C에서 권장하는 방식이기도 하다. 하지만 보다 간편하고 개발의 용이성을 지향하는 근래의 경향 속에서 URL을 통해 API를 구동하고 결과를 받는 REST 방식이 인기를 얻어가고 있다. URC 서비스 플랫폼에서는 openAPI의 원래 취지에 부합하여 개방성과 그 활용을 최대화하기 위해서는 특정 방식으로만 제공하기보다는 SOAP, REST 등 여러 가지 방식으로 제공하는 것이 필요하다.

이렇게 구현된 openAPI는 URC 서비스 플랫폼의 웹페이지 등을 통해 목록, 사용법 등을 제공해야 할 것이다. 물론 SOAP의 경우 이러한 openAPI들의 검색 등을 위한 리파지토리(repository)로서 UDDI (Universal Description, Discovery & Integration)가 표준으로 제정되어 있지만 그 사용의 경우가 드물고 많은 경우 단순히 웹을 통해 제공하고 있다.

4.3 URC에서의 주요 OpenAPI

URC의 경우 전형적인 openAPI인 인증/권한, 상태정보, 그리고 URC 서비스 플랫폼이 가진 기능을 open 하는 것 이외에도 CP 들이 제작한 서비스들을 수용할 수 있도록 하는 보다 융통성 있는 형태가 되어야 한다.

URC 서비스 플랫폼에서는 openAPI를 다음과 같이 분류하고 이를 구현하였다.

표 1. URC 서비스 플랫폼의 openAPI 분류

분류	내용
인증, 권한 획득	사용자가 openAPI를 사용하고 등록할 수 있는 권한을 부여하고 확인하는 기능 getCpAuthRequest, getUserAuthRequest
URC 로봇 상태 (프레즌스)	URC 로봇의 접속 상태, 접속 주소 등을 알 수 있는 기능 getRequestDevTypeInfo, getRequestUserDevInfo, getRequestDevPresence, getRequestDevHWCapability, getRequestDevSWCapability, getRequestDevMediaCapability
URC 로봇 제어	CP들이 URC 로봇을 제어할 수 있도록 하는 기능 getDevControlListRequest, getDevSetControlRequest
URC 서비스 등록	CP들이 신규 서비스 개발시 이를 URC 서비스 플랫폼에 등록하는 기능 getContentByRegister, getContentByUpdate, getContentByDelete, getContentByInfo, getContentByList
URC 서비스 사용	등록된 URC 서비스들을 CP들이 사용할 수 있도록 하는 기능 getContentDownLoad, getContentByUpLoad

4.4 OpenAPI를 통한 URC 서비스 제공의 예

아래 그림은 프레즌스와 openAPI를 결합하여 서비스를 제공하는 시나리오로서 사용자가 포털 사이트의 블로그 등에서 자신 또는 친구의 로봇 상태를 확인하고 이를 제어하는 과정을 도식한 것이다. 여기서 CP는 포털 사이트가 될 것이며 이들은 웹 페이지 등의 사용자 인터페이스를 통해 로봇의 프레즌스 정보 요청을 받아 이를 openAPI를 통해 URC 서비스 플랫폼으로 전달하고 그에 대한 응답을 수신한 후 사용자에 전달한다. 이를 확인한 사용자가 로봇에 대한 제어를 입력하면 CP는 위와 동일한 형태로 다시 openAPI로 URC 서비스 플랫폼에 사용자의 제어 요청을 전송하고 그 응답을 수신한 후에 사용자에게 표시한다. 물론 여기서는 사전에 CP가 openAPI 사용을 위한 인증 및 권한을 획득한 것을 가정한 것이다.

이러한 openAPI를 사용하여 새로운 서비스를 창출하는 mashup 형태의 경우는 구글[7], 아마존[8] 등 여러 곳에서 성공적인 사례들이 나타나고 있으며 향후 URC 서비스의 개방화, 다양화에 있에서 도 openAPI는 매우 중요한 역할을 수행할 것이다.

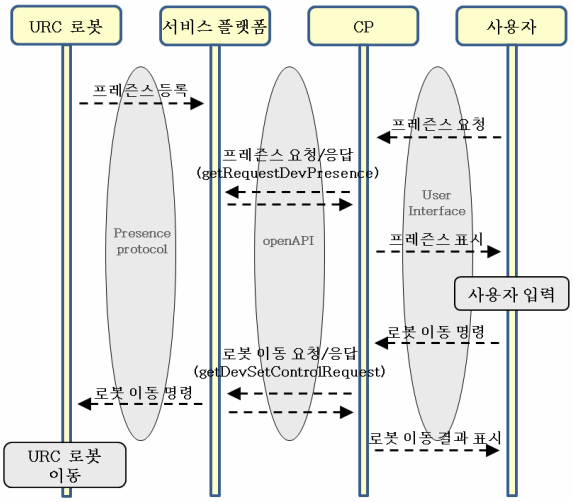

그림 4. URC 로봇에서 openAPI를 통한 서비스 흐름

5. 결 론

지금까지 본고에서는 프레즌스와 openAPI에 대해서 살펴보고 이들의 URC 서비스 플랫폼에 구현 사례를 살펴보았다.

프레즌스 서비스는 네트워크를 통해 로봇의 상태, 주소 및 그 상태 등을 표준화된 틀로 관리하고 이러한 프레즌스 정보를 상호 교환으로써 로봇과 로봇, 로봇과 사람 사이의 상호 소통과 이를 통한 다양한 서비스를 제공할 수 있도록 해주며, openAPI는 CP 및 사용자의 참여와 공유를 이끌어 내고 이를 통해 보다 풍부한 서비스가 창출되도록 해주며 아울러 이를 통해 URC 서비스가 상업적으로도 성공을 거둘 수 있는 요소가 될 것이다.

특히 호처리와 인스턴트 메시징을 위한 IETF 및 3GPP의 프레즌스 서비스, 프로토콜, 상태 들에 관련된 표준들과 Parlay/OSA[9]의 사용자 mobility, 프레즌스 관련 openAPI 등과 같이 URC 분야에 있어서도 로봇의 프레즌스 및 관련 openAPI의 표준화가 앞으로 이루어져야 할 것으로 판단된다.

향후 URC 서비스 플랫폼에서는 이러한 프레즌스 및 openAPI 등을 통해 능동적이고 지능적인 서비스를 제공하고 제어하는 기능들의 개발이 필요할 것이며, 이들을 활용한 킬러 어플리케이션들의 발굴 또한 요구된다.

참 고 문 헌

[1] M. Day, J. Rosenberg and H. Sugano, “A Model for

Presence and Instant Messaging", IETF RFC 2778, February, 2000.
[2] J. Rosenberg, "A Presence Event Package for the Session Initiation Protocol (SIP)", IETF RFC 3856, August, 2004.
[3] "3GPP-Presence service using the IP Multimedia (IM) Core Network (CN) subsystem; Stage 3(R6)"
[4] http://openapi.naver.com
[5] Nilo Mitra and Yves Lafon, "SOAP ver1.2", W3C, April, 2007
[6] David Booth and Canyang Kevin Liu, "Web Services Description Language(WSDL) ver 2.0", W3C, June 2007
[7] http://code.google.com
[8] http://www.amazon.com
[9] Parlay X Web Services Specification, Version 3.0 (www.parlay.com)
[10] 나재욱, 최진영, 조현덕, 김용훈, 이진구, 박종태, "확장성 있는 프레즌스 기능을 가진 무선 인스턴트 메시징 시스템 설계 및 구현", 한국정보과학회 논문지 제33권 제3호, pp257~268, 06, 2006.
[11] 정의헌, 김화성, "유무선 통합서비스를 위한 openAPI", 한국통신학회지 제20권 제11호, 11, 2003.

배 정 일

1998 중앙대학교 제어계측공학과(공학사)
1998~1999 삼성SDS
1999~현재 KT 인프라연구소 선임연구원

관심분야 : URC 서비스 및 플랫폼, openAPI, WEB2.0, 인터넷 정보 단말 서비스

김 동 훈

1994 부산대학교 컴퓨터공학과(공학사)
1996 부산대학교 컴퓨터공학과(공학석사)
1996~현재 KT 인프라연구소 책임연구원

관심분야 : WEB2.0, openAPI, URC 서비스, 인터넷 정보 단말 서비스

이 현 주

2001 덕성여자대학교 전산학과(공학사)
2006 서울대학교 전기컴퓨터공학부(공학석사)
2001~2004 삼성전자 SW센타 연구원
2007~현재 KT 인프라연구소 전임연구원

관심분야 : URC 서비스 및 플랫폼, openAPI

연 승 호

1985 충북대학교 컴퓨터공학과(공학사)
1988 충북대학교 컴퓨터공학과(공학석사)
2000 충북대학교 컴퓨터공학과(공학박사)
1990~현재 KT 인프라연구소 수석연구원

관심분야 : URC 서비스, 인터넷 정보 단말