

SyncML DM을 이용한 URC 로봇 클라이언트 관리

손은미, 황석찬, 광동규, 최재영
숭실대학교 컴퓨터학부

e-mail : {emson, schwang, coolman}@ss.ssu.ac.kr, choi@ssu.ac.kr

A Management System for URC Robot Client using SyncML DM

Eunmi Son, Seokchan Hwang, Dongkyu Kwak, Jaeyoung Choi
School of Computing, Soongsil University

요 약

다양한 이기종 로봇 클라이언트가 등장하면서 각 URC서버의 로봇 클라이언트 관리 및 프로파일링 방법 등은 생산 업체에 의존적이다. 이러한 URC 서버와 로봇 클라이언트간의 표준화된 통신 프로토콜만으로는 URC 서버들간의 호환성을 보장해줄 수 없다. 본 논문에서는, SyncML DM 기술을 사용하여 RUPI 참조 구현 모델인 URC 서버의 로봇 클라이언트 접속 및 프로파일링, 원격 진단 및 관리 모델을 제안한다.

1. 서론

로봇이 차세대 산업 경쟁력 확보를 위한 국가 과제들로 부상하면서 다양한 방법으로 서비스에 접근하는 지능형 서비스 로봇에 대한 장밋빛 전망이 대두되고 있다. 기존의 독립형 로봇에서 벗어나 시간과 장소에 제한 받지 않는 유비쿼터스 로봇을 중심으로 기술 개발하여 인간 생활의 새로운 문화를 창출하고 있다[1].

URC(Ubiquitous Robotic Companion)는 기존 로봇에 네트워크 및 정보 기술을 접목한 지능형 서비스 로봇의 새로운 개념으로서, 언제, 어디서나 나와 함께하며, 나에게 필요한 서비스를 제공하는 네트워크 기반 로봇이다[2]. 이와 같이 인간과 공존하는 생활 파트너인 URC 로봇은 가사 도우미 로봇, 개인비서 로봇, 동반자 로봇 등 다양한 형태로 출현하고 있다. URC 로봇은 로봇 핵심 기능을 외부 서버에 전달시켜 로봇 하드웨어 구성을 단순화하고 다양한 서비스를 실시간 제공하는 로봇으로, 그 안전성 및 효율성을 높이기 위하여 고도의 로봇 클라이언트의 관리 기술이 필요하다.

현재까지 인터넷 장비나 유선 통신 장비의 경우에는 SNMP(Simple Network Management Protocol)를 이용한 디바이스 관리가 주축이었고, 가정 내 홈 네트워크에 사용되는 장비들의 경우에는 TR069(short for Technical Report 069)를 이용한 디바이스 관리가 널리 사용되었다[3]. 그러나 상기 프로토콜들의 관리 기능은 제한적인데다, 정적인 네트워크 토폴로지로 구성되어 이동성 기능 제공에 한계가 있으므로 네트워크 기반 로봇 관리에는 적합하지 않다.

OMA(Open Mobile Alliance)는 디바이스 관리 규격인 DM(Device Management)에 데이터 동기화 기술 규격인 SyncML(Synchronization Markup Language)을 디바이스

관리 용도로 확장하여 SyncML DM이라는 새로운 표준을 제정하였다[4, 5]. 디바이스의 이동성, 원격 관리, 안정적이고 호환성 높은 기능들을 요구하는 네트워크 기반 로봇 클라이언트 관리 시스템에는 SyncML DM 기술이 적합하다. 국제 표준안인 SyncML DM 기술은 다양한 이기종의 URC 서버 및 클라이언트를 일관된 방법으로 관리할 수 있다.

RUPI(Robot Unified Platform Initiative)는 지능형 로봇에 대한 통합 소프트웨어 플랫폼 및 제반 표준 규격이다. 네트워크 로봇이 제공하는 복잡한 서비스를 쉽고 편리하게 개발하기 위한 각종 소프트웨어의 표준 규격을 제공하고 이를 참조 구현하는 것을 목표로 한다[6]. RUPI 규약 중, URC 로봇 클라이언트 관리 규약[7]은 디바이스 관리 표준인 OMA DM을 채택하여 URC 서버의 원격 로봇 클라이언트 진단 및 관리 기능을 제공한다. 현재 RUPI 2.0 규격이 공개되었으며 구현 모델은 차후 결정될 예정이다.

본 논문에서는, 국내 로봇 표준 플랫폼인 RUPI 규격을 참조하여 SyncML DM을 적용한 URC 로봇 클라이언트 관리 모델을 제안한다. 상기 시스템은 상이한 플랫폼간의 이식성을 향상시키고 응용 프로그램간의 의존성을 최소화 한다. 이동성 있는 네트워크 기반 지능형 로봇 관리의 중요성이 제기되고 있는 시점에서 상기 시스템을 개발함으로써, SyncML DM을 기반으로 한 로봇 클라이언트 관리 방식의 확산 및 국제 표준화에 기여할 수 있다.

본 논문의 구성은 다음과 같다. 2장에서는 SyncML과 RUPI의 특징 및 표준화 동향에 대해서 알아보고, 3장에서는 URC 로봇 클라이언트 관리 모델을 제시한다. 마지막으로 4장에서 결론을 맺고 앞으로의 연구 진행 방향을 언급한다.

"본 연구는 지식경제부 및 정보통신연구진흥원의 대학 IT 연구센터 지원사업의 연구결과로 수행되었음"
(IITA-2008-C1090-0803-0006)

2. 관련연구

2.1 MSRS

미국 마이크로소프트사에서 개발한 MSRS(Microsoft Robotics Studio)는 Windows 및 .NET 기반 환경에서 동작하는 로봇용 소프트웨어 플랫폼이다.

MSRS상에서 개발된 로봇은 Windows 또는 웹 기반의 인터페이스를 통하여 사용자와 직접적인 상호작용이 가능하다. HTML형식이나 JavaScript와 같은 웹 기술을 이용하며, 웹 브라우저를 통하여 원격지의 로봇을 모니터 및 컨트롤한다[8].

2.2 ERSP

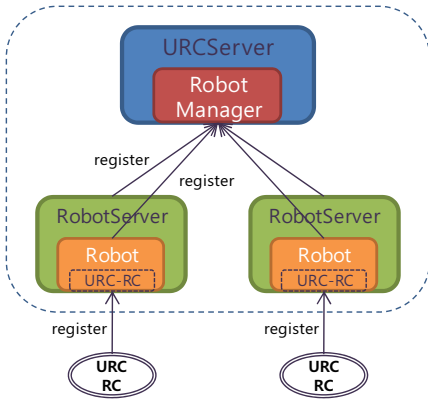
미국 Evolution Robotics사의 ERSP(Evolution Robotics Software Platform)는 모바일 로봇의 제반 소프트웨어 개발 플랫폼이다.

ERSP는 하드웨어 장치를 추상화하고 자원 및 장치에 대한 기술 명세(XML 형식)부분인 HAL (Hardware Abstraction Layer)계층과 행위를 중재하는 BEL(Behavior Execution Layer)계층, 그리고 태스크들의 실행을 관리하는 TEL(Task Execution Layer)계층으로 구성된다[9].

2.3 RUPI

RUPI는 로봇 소프트웨어 컴포넌트의 재사용성 및 상호호환성, 다양한 정보기기와의 상호운용성, 이중 통신망과의 상호 접속성을 갖는 지능형 로봇의 소프트웨어 규격 및 구현 모델이다[6]. 총 6개의 워킹 그룹(로봇 내부 S/W, URC 서버 S/W, 네트워크, 로봇 콘텐츠, 통합 개발 환경, 응용 컴포넌트)을 구성하여 RUPI 2.0 규격 28종을 작성하였고, 현재 RUPI 2.0 규격 3차 버전까지 공개되었다. RUPI 규격은 네트워크를 통하여 다양한 서비스를 제공하기 위한 RUPI 서버 소프트웨어, 네트워크 프로토콜, 로봇 콘텐츠를 함께 다룬다는 측면에서 국내외의 다른 기술과 차별화된다[10].

3. URC 로봇 클라이언트 관리



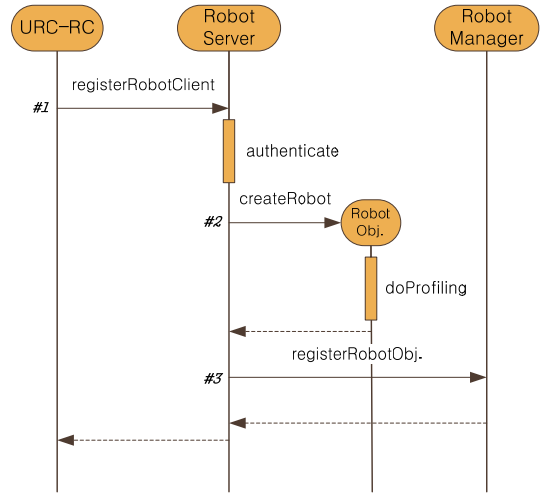
(그림 1) URC 로봇 클라이언트 관리 구조

URC 서버는 이기종의 URC 로봇 클라이언트들이 접속되기 때문에 여러 종류의 로봇들을 관리할 수 있어야 한다. 따라서 각각의 로봇들에게 어떤 기능이 있는지 또는 관리에 필요한 정보들은 어떤 방법으로 수집할 것인지에 대하여 미리 인지하여야 한다. URC 서버가 이기종의 다양한 로봇 클라이언트를 관리하는 체계는 (그림 1)과 같다.

URC 로봇 클라이언트 관리는 크게 RobotServer와 RobotManager 두 종류의 컴포넌트로 이루어진다. URC 서버는 RobotManager를 통하여 로봇 클라이언트들을 관리한다. URC 서버에는 반드시 하나의 RobotManager가 존재하며, 하나의 RobotManager에 대해서는 하나 이상의 RobotServer가 존재한다. RobotServer는 URC 로봇 클라이언트의 접속 및 프로파일링 작업을 수행한다. RobotServer에 접속된 URC 로봇 클라이언트는 Robot 객체 모듈로 추상화되고, RobotManager는 추상화된 Robot 객체들을 관리 한다.

3.1 URC 로봇 클라이언트 접속

URC 로봇 클라이언트 접속 관리란 URC 서버에 접속된 URC 로봇 클라이언트가 해당 생명 주기 동안 관리 되는 것을 의미한다. URC 로봇 클라이언트 접속 과정은 다음 (그림 2)과 같다.



(그림 2) URC 로봇 클라이언트 접속 과정

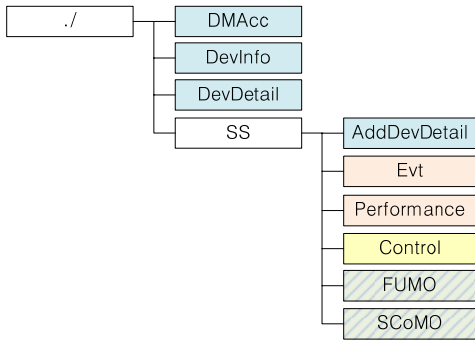
(그림 2)의 접속 과정에서, 패키지 #1은 URC 로봇 클라이언트가 RobotServer에 등록되는 과정이다. URC 로봇 클라이언트가 RobotServer에 등록되면서 자신의 디바이스 정보와 인증 정보가 함께 전달된다. RobotServer는 전달된 인증 정보를 참조하여 DM 서비스를 수행할 적절한 대상인지를 판단하고 전달된 디바이스 정보를 분석하여 하드웨어 및 소프트웨어 사양 정보를 얻는다.

패키지 #2는 RobotManager에서 관리될 수 있도록 로봇 객체를 생성하고 프로파일링하는 과정이다. 프로파일링이란 패키지 #1에서 전달된 디바이스 정보를 바탕으로 각 사양에 따른 로봇 추상화 과정을 의미한다. 예를 들어, URC 로봇 클라이언트가 카메라와 녹음기의 디바이스로 구성되어 있다고 할 때, URC 서버의 클라이언트 응용 서비스를 받을 수 있도록 카메라와 녹음기 디바이스 정보를 각각 얼굴 인식 및 음성 인식 기능으로 맵핑하여 프로파일을 확장 시키는 것이다. 따라서 URC 로봇 클라이언트는 다양한 응용 및 콘텐츠들을 수행할 수 있다.

패키지 #3은 패키지 #2에서 생성된 로봇 객체를 RobotManager에 등록하는 과정이다. 이후 로봇 객체는 RobotManager에 의해 관리되며, URC 로봇 클라이언트 응용 프로그램의 인터페이스 역할을 한다. 따라서 이기종의 다양한 URC 로봇 클라이언트들은 플랫폼에 관계없이 URC 로봇 클라이언트 응용 서비스를 제공받을 수 있다.

3.2 로봇 관리 객체

RobotServer의 또 다른 중요 역할은 URC 로봇 클라이언트를 원격 진단 및 모니터링 하는 것이다. 로봇을 원격에서 관리하기 위하여 필요 항목들을 정의해야 하는데 그 항목들을 관리 객체라 한다[11].



(그림 3) 상위 수준의 관리 객체 트리

<표 1> 상위 수준의 관리 객체 설명

분류항목	관리 노드	설명
디바이스	DMAcc	인증 정보 노드
	DevInfo	기본 정보 노드
	DevDetail	디바이스 상세 정보 노드
	AddDevDetail	디바이스 추가 상세 정보 노드
진단 및 모니터링	Evt	이벤트 관리 노드
	Performance	디바이스 성능 분석 노드
제어	Control	디바이스 제어를 위한 노드
소프트웨어	FUMO	디바이스 펌웨어 업데이트를 위한 노드
	SCoMO	디바이스 SW 컴포넌트 관리를 위한 노드

로봇 관리 객체는 디바이스, 진단 및 모니터링,

제어, 소프트웨어 배포의 크게 4항목으로 분류된다. (그림 3)은 4항목으로 분류된 관리 객체를 상위 수준의 트리로 나타내고, <표 1>은 (그림 3)에 대한 자세한 설명을 덧붙인다.

디바이스 관리 객체는 SyncML DM Standard Object[11] 스펙을 따른 표준 디바이스 관리 객체와 SyncML DM Tree and Description[12] 스펙을 참조한 추가적인 디바이스 관리 객체로 구분된다. 표준 디바이스 관리 객체는 필수 관리 객체인 DMAcc, DevInfo, DevDetail 가지고, 추가적인 디바이스 관리 객체는 AddDevDetail 관리 객체를 갖는다. AddDevDetail 관리 객체는 하드웨어, 소프트웨어, 네트워크, 유틸리티를 관리하는 객체들로 나누어진다.

진단 및 모니터링 항목은 디바이스의 정보를 수집하여 이를 근거로 서버에서 장애 방지 및 해결을 위한 항목이다. Evt, Performance 관리 객체가 여기에 해당된다.

소프트웨어 관리 항목은 OMA DM에서 표준 승인된 FUMO(Firmware Update Management Object)[14] 관리 객체와 표준화 진행중인 SCoMO(Software Component Management Object)[15] 관리 객체로 구분된다.

3.3 로봇 관리 프로토콜

RobotServer와 URC 로봇 클라이언트 간의 사용되는 관리 프로토콜은 SyncML Representation Protocol, Device Management Usage[13] 표준 스펙을 따른다. 상기 스펙에서 사용되는 주요 디바이스 명령 엘리먼트들은 <표 2>와 같다.

<표 2> 디바이스 명령 엘리먼트

엘리먼트	설명
Add	새로운 관리 대상의 추가
Alert	알림 메시지를 표현
Atomic	명시된 순서대로 실행되는 다중 명령어 포함. 하나의 Status로 응답
Copy	관리 대상의 복사
Delete	관리 대상의 삭제
Exec	관리 대상의 실행
Get	관리 대상의 정보를 얻음
Replace	관리 대상의 변경
Sequence	명시된 순서대로 실행되는 다중 명령어 포함

로봇 관리 프로토콜의 실행 단계는 RobotServer가 URC 로봇 클라이언트에게 관리를 위한 명령어의 수행을 요구하고, URC 로봇 클라이언트는 그 결과를 알려주는 패키지들로 구성되어 있다. 이러한 패키지들은 관리 목적에 따라 반복적으로 수행될 수 있다.

예를 들어, URC 로봇 클라이언트의 소프트웨어 백신을 최신 버전으로 업데이트 해야 할 경우, RobotServer는 현재 URC 로봇 클라이언트의 최신 백신 버전을 알아야 한다. 정보를 얻기 위하여 RobotServer는 다음과 같은 메시지를 URC 로봇 클라이언트로 송신한다.

RobotServer에서 보내는 명령 메시지:

```
<Get>
  <CmdID>2</CmdID>
  <Item>
    <Target>
      <LocURI>./Vaccine/Version</LocURI>
    </Target>
  </Item>
</Get>
```

RobotServer로부터 명령 메시지를 받은 URC 로봇 클라이언트는 명시된 URI(Uniform Resource Identifier)에 해당하는 백신 버전 1.1을 결과 메시지에 담아 반환한다.

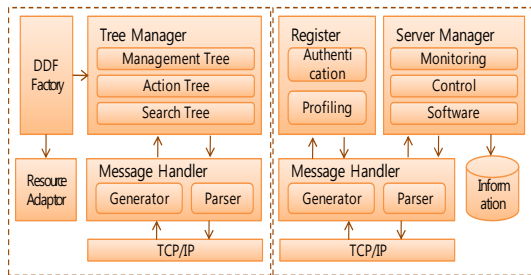
로봇 클라이언트에서 보내는 결과 메시지:

```
<Results>
  <CmdRef>2</CmdRef>
  <CmdID>5</CmdID>
  <Item>
    <Meta>
      <Format xmlns="syncml:metinf">chr</Format>
      <Type xmlns="syncml:metinf">text/plain</Type>
    </Meta>
    <Data>0001.1000</Data>
  </Item>
</Results>
```

만약, URC 로봇 클라이언트가 RobotServer의 명령 메시지를 정상적으로 수행하지 못했을 경우, 미리 정의된 상태 코드를 포함한 Status 메시지를 반환한다.

3.4 로봇 클라이언트 관리 시스템

(그림 4)는 본 논문에서 제안하는 URC 로봇 클라이언트 관리 시스템 구조이다. URC 로봇 클라이언트(왼쪽)와 RobotServer(오른쪽) 구조를 나타낸다.



(그림 4) 로봇 클라이언트 관리 시스템 구조

URC 로봇 클라이언트의 Message Handler는 송수신하는 SyncML DM 메시지의 파싱 및 생성을 담당한다. Tree Manager는 로봇을 구성하는 디바이스 정보를 트리 형식으로 추상화하여 가지고 있다. DDF Factory에 저장된 DDF(Device Description Framework)를 참조하여 노드의 추가, 삭제 등 관리가 이루어지며,

Resource Adaptor는 트리 노드와 실제 디바이스를 맵핑시키는 역할을 한다.

RobotServer의 Message Handler는 URC 로봇 클라이언트와 동일하다. Register는 접속하는 로봇 클라이언트의 인증을 담당하고, 로봇 객체를 생성하여 프로파일링 작업을 한다. Server Manager는 로봇 클라이언트의 모니터링, 제어, 진단 등을 담당하고, Information Factory는 로봇 클라이언트에 대한 정보와 명령 수행 결과를 저장하고 있다.

4. 결론 및 향후 연구 과제

본 논문에서는, 표준 로봇 플랫폼 RUPI를 참조하여 SyncML DM 기술을 적용한 URC 로봇 클라이언트를 관리 모델을 제안하였다.

URC 로봇 클라이언트를 로봇 객체라는 추상화 된 관리 객체로 다루므로 다양한 기기종의 로봇 클라이언트들을 동일한 체계로 관리할 수 있다. 또한 프로파일링을 통하여 제한된 URC 로봇 클라이언트의 기능을 다양한 범위로 확장 시킬 수 있다. 국제 표준인 SyncML DM을 수행할 수 있는 URC 로봇 클라이언트라면 일관된 방법으로 원격 관리 서비스를 제공 받을 수 있다.

향후 본 모델의 구현을 통하여 URC 로봇 클라이언트 관리에 한 획을 그으면서 국내 로봇 산업의 빠른 성장을 도모할 수 있을 것이라 기대해 본다.

참고문헌

- [1] 손주찬, 김재홍, 하여국, 장인수, 박천수, 이미경, "URC 지능형 서비스 기술 개발 동향", ETRI 주간기술동향 통권 1190호, 2005.4, pp30-44.
- [2] URC Korea, <http://www.urckorea.or.kr>
- [3] 이지은, 성승학, 정병덕, "OMA DM 기반의 휴대 인터넷 단말 관리 시스템", KNOM Review, Vol.10, No.2, Dec. 2007. pp1-11.
- [4] 장대진, 주홍택, 박기현, "SyncML DM 기반의 무선이동통신 단말기 관리 설계", KNOM Review, Vol. 6, No. 2, Feb. 2004, pp.7-12.
- [5] 박지은, 박준석, 한동원, "SyncML의 디바이스 관리 표준화 동향", 전자통신동향분석 제17권 제6호, 2002.12, pp59-66.
- [6] RUPI, <http://www.rupi.or.kr>
- [7] RUPI, "URC 로봇 클라이언트 관리", RUPI 2.0.2-R12.1
- [8] MSRS, <http://www.microsoft.com/korea/robotics>
- [9] ERSP, <http://www.evolution.com>
- [10] 정승욱, 이승익, 김성훈, "네트워크 로봇을 위한 로봇 소프트웨어 플랫폼에 대한 연구", 정보과학회지 제26권 제4호, 2008.4, pp38-48.
- [11] SyncML, SyncML Device Management Standardized Objects, www.syncml.org, 2002.
- [12] SyncML, SyncML Device Management Tree and Description, www.syncml.org, 2002.
- [13] SyncML, SyncML Representation Protocol, Device Management Usage, version 1.1, www.syncml.org, 2002.
- [14] OMA, Firmware Update Management Object, www.openmobilealliance.org, 2007.
- [15] OMA, Software Component Management Object, www.openmobilealliance.org, 2008.