RUPI 클라이언트 통합 소프트웨어 플랫폼에 대한 연구동향 및 현황

정승욱 | 이승익 | 김성훈
한국전자통신연구원

요 약

RUPI(Robot Unified Platform Initiative, 로봇 통합 소프트웨어 플랫폼) 기술은 기존 로봇의 한계를 극복하고 보다 다양하고 복잡한 서비스를 쉽고 편리하게 개발하고 제공하기 위해 네트워크 로봇에 필요한 각종 소프트웨어의 표준 규격을 제정하고 이를 참조 구현하는 것을 목표로 한다. 본고에서는 로봇을 통해 환경을 인지하거나 로봇을 제어하고자 할 때 필요로 하는 로봇 내부 소프트웨어에 해당하는 RUPI 클라이언트 로봇 소프트웨어 플랫폼에 관하여 소개하고자 한다. RUPI 클라이언트 로봇 소프트웨어 플랫폼은 로봇 소프트웨어의 재사용성 및 상호호환성을 위해 컴포넌트 기반의 로봇 개발 프레임워크를 제공한다.

I. 서 론

2000년대 이후 로봇 패러다임은 산업용 로봇에서 지능형 서비스 로봇 분야로 빠르게 변화하고 있다. 산업용 로봇이 일반인의 접근이 통제된 고정된 환경에서 미리 순서가 정해진 일을 반복 수행하는 것과는 달리 지능형 서비스 로봇은 일반 사용자와 접촉하며 변화하는 환경에 맞게 적절한 일을 선택적으로 수행할 수 있어야 한다. 이러한 지능형 서비스 로봇을 산업화하기 위해서는 로봇이 사용자와 사용자의 명령을 인식하고 주행 혹은 보행 기능을 가지고 사용자와 상호작용할 수 있는 최소한의 지능을 보유해야 하며, 값싸게 제작되어 널리 보급될 수 있어야 한다. 과거의 로봇 시스템은 단독형(stand-alone) 구조로 새로운 기능을 확장하거나 부가할 수 있는 확장성이 부족했으며, 또한 로봇 소프트웨어의 단위 기능들이 표준화되지 않고 상호 종속되어 새로운 로봇 시스템을 개발하는 작업을 처음부터 다시 반복하는 비효율적인 방법을 따르고 있어 로봇 모듈의 이식성과 호환성 그리고 재사용성이 떨어졌다[1].

최근에는 인공지능, 네트워크 등의 IT 기술을 바탕으로 네트워크와 연계하여 인간과 서로 상호작용하면서 가사, 교육, 오락, 공공 서비스 등 다양한 서비스를 제공하는 네트워크 로봇에 대한 연구가 활발하게 이루어 지고 있다. 정부에서도 2004년도부터 국민 소득 2만 달러 시대를 여는 핵심 기술 및 신 성장 동력의 하나로 로봇을 선정하고 정부, 학계, 연구소, 산업계가 공동으로 URC(Ubiquitous Robotic Companion)라는 개념으로 네트워크 기반의 지능형 서비스 로봇 사업을 추진하고 있다. URC는 "언제 어디서나, 나와 함께 하며, 나에게 필요한 서비스를 제공하는 로봇"을 의미하며, 세계 최고 수준의 유무선 인프라를 활용하여 로봇 기술 고도화 및 경쟁력 강화를 목표로 하고 있다. URC 사업에서는 네트워크 기능을 활용하여 로봇의 인식기술과 HRI(Human Robot Interaction) 요소 기술의 성능을 개선하고 이를 통해 지능형 로봇의 서비스 질과 지능을 높일 수 있을 것으로 기대하고 있다[2].

네트워크와 연계된 지능형 서비스 로봇은 변화하는 주변 환경과 네트워크를 통해 획득한 정보를 기반으로 사용자에게 다양하고 유익한 서비스를 제공할 수 있으며, 사용자가 어디에 있든지 네트워크를 통해 로봇을 제어할 수 있는 속

성을 가지고 있다. 네트워크 로봇이 이러한 기능을 수행하기 위해서는 로봇이 주변 환경과 쉽게 연동 가능해야 하며, 네트워크 상에 존재하는 다양한 정보를 획득해야 하고 또한 음성인식/음성합성, 사용자 인식 등의 인간 친화적인 기능을 제공해야 한다.

그러나 이러한 복잡한 기능을 수행하는 로봇을 하나의 로봇 업체가 모두 개발한다는 것은 쉽지 않으며, 결국 이미 개발된 제품을 조합하여 사용자가 요구하는 다양한 기능을 만족하는 로봇을 개발해야 한다.

RUPI 기술은 기존 로봇의 한계를 극복하고 보다 다양하고 복잡한 서비스를 쉽고 편리하게 개발하고 제공하기 위해 네트워크 로봇에 필요한 각종 소프트웨어의 표준 규격을 제정하고 이를 참조 구현하는 것을 목표로 한다.

RUPI는 로봇 소프트웨어 컴포넌트의 재사용성 및 상호호환성, 다양한 정보 기기와의 상호 운용성, 이종 통신망과의 상호 접속성을 갖는 지능형 로봇의 소프트웨어 규격 및 구현 모델이다. RUPI 규격은 로봇 소프트웨어가 따라야 하는 최소한의 요구 사항을 의미론적으로 기술한 것과 컴포넌트 및 인터페이스 등에 대한 기술 내용을 의미하며, RUPI 구현 모델은 RUPI 규격을 만족하는 소프트웨어 모듈로서 바로 적용할 수 있는 구현물을 의미한다. RUPI 규격은 크게 로봇 내부 소프트웨어와 관련된 RUPI 클라이언트 로봇 규격, RUPI 클라이언트 로봇을 관리하고 서비스를 제공하는 RUPI 로봇 서버 규격, 로봇과 서버간 통신 프로토콜 규격, 개발도구 규격 그리고 음성인식/음성합성, 얼굴인식 등의 로봇 응용 컴포넌트 규격으로 이루어져 있다.

본고에서는 로봇을 통해 환경을 인지하거나 로봇을 제어하고자 할 때 필요로 하는 로봇 내부 소프트웨어에 해당하는 RUPI 클라이언트 로봇 소프트웨어 플랫폼에 관하여 소개하고자 한다.

본고의 RUPI 클라이언트 로봇 소프트웨어 플랫폼은 로봇 소프트웨어의 재사용성 및 상호호환성을 위해 컴포넌트 기반의 로봇 개발 프레임워크를 제공한다.

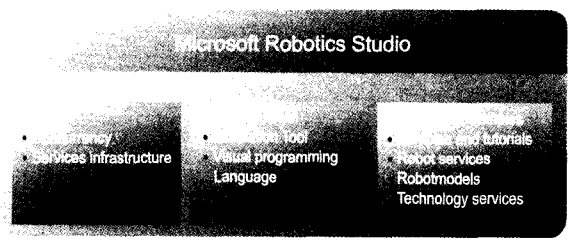
본고의 II장에서는 로봇 소프트웨어 플랫폼에 관한 기존 연구 동향 및 제품 현황을 소개한다. III장에서는 RUPI 클라이언트 로봇 소프트웨어 플랫폼에 대해 설명하고, IV장에서 결론을 맺는다.

II. 기존 연구 동향

지능형 서비스 로봇에 대한 연구 개발 및 시장이 활성화됨에 따라 로봇 소프트웨어 개발을 지원하기 위한 로봇 소프트웨어 플랫폼에 대한 연구도 활발해지고 있다. 본 장에서는 로봇 소프트웨어 플랫폼과 관련된 연구 및 제품에 대해 기술한다.

미국 마이크로소프트사의 MSRS(Microsoft Robotics Studio)[3]는 로봇 소프트웨어 전문 개발자뿐 아니라 초보자도 쉽게 로봇 소프트웨어를 제작할 수 있도록 .NET 기반의 시각적 로봇 응용 개발 환경 및 실행 환경을 지원하고 있다.

(그림 1)은 MSRS의 특징을 나타낸 그림이며, 내용은 아래와 같다.



(그림 1) MSRS의 특징

- o 서비스 기반의 런타임 아키텍처: MSRS상에서 개발되는 기능들은 서비스로 노출되며, 로봇 응용은 기본적으로 서비스의 조합으로 이루어진다. 로봇 응용 개발 시 서비스의 인터페이스와 해당 서비스가 제공하는 기능에만 초점을 맞추기 때문에 컴포넌트 개념과 유사하며 소프트웨어의 이식성과 재사용을 제공한다.
- o REST[4] 스타일의 분산 응용 패턴 제공: MSRS상에서 개발된 로봇 응용은 여러 컴퓨터에 분산되어 실행 가능하며, 이들 간의 통신은 REST(REpresentational State Transfer)를 통해 이루어진다.
- o 동시처리 및 제어 기술: 멀티 쓰레딩 환경에서 로봇 프로그래밍을 작성할 때 발생하는 동시성의 문제를 해결하

고 단순한 코딩 작업을 가능하게 한다.

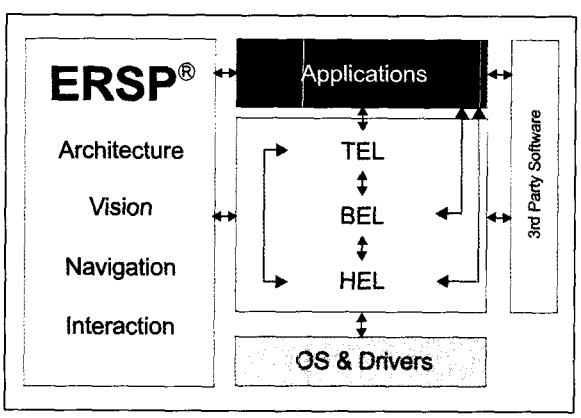
- o 로봇 시뮬레이터(simulator) 제공: Agea사의 PhysX엔진을 기반으로 한 3D 동역학 로봇 시뮬레이션(simulation) 환경을 제공한다.
- o 시각적 로봇 응용 개발 언어 지원: 로봇 프로그램 초보자들도 쉽게 개발이 가능한 비쥬얼 개발 환경을 지원한다.

2. ERSP

미국 Evolution Robotics사의 ERSP(Evolution Robotics Software Platform)[5]는 로봇 응용 개발을 위한 종합적 개발 플랫폼으로서, 개발자들이 로봇 응용을 신속하고 용이하게 구축할 수 있게 하는 기반 구조, 핵심 컴포넌트 및 도구를 제공한다. ERSP는 장치 추상화 계층(Hardware Abstraction Layer; HAL), 행위 실행 계층(Behavior Execution Layer; BEL), 태스크 실행 계층(Task Execution Layer; TEL) 등 3계층으로 이루어진 실행 환경 ERSA, 물체 인식 등을 위한 ViPR, 자율 주행을 위한 vSLAM, 위치 인식을 위한 NorthStar 및 로봇 행위 개발 도구 등으로 구성되어 있으며, 표준 플랫폼이라기 보다는 ER사의 비전, 자율주행 등의 컴포넌트를 실행하기 위한 로봇 소프트웨어 플랫폼이다.

ERSP는 다양한 로봇 응용을 개발하기 위한 소프트웨어 개발 툴킷(toolkit)으로 ERSP의 유연한 소프트웨어 구조는 다양한 드라이버와 행위(behavior), 태스크(task)와 같은 소프트웨어 모듈을 포함하고 있어서 쉽고 빠르게 로봇 응용을 개발할 수 있도록 한다.

(그림 2)는 ERSP의 구조를 나타낸 것이다.



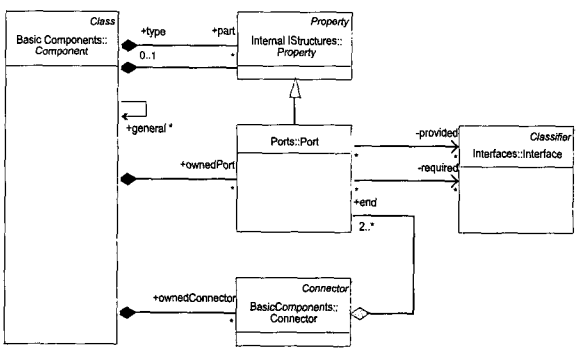
(그림 2) ERSP 구조

3. RTC

RTC(Robot Technology Component)[6]는 일본의 AIST 및 미국의 RTI 주도로 OMG 의 Robotics DTF에서 2006년 9월 표준으로 채택된 로봇 소프트웨어 컴포넌트 규격으로 실행 시 멘 틱 스 (Execution Semantics), 인 트 로 스 펙 션 (Introspection) 및 경량(Lightweight) RTC 등을 특징으로 한다.

RTC는 재사용이 가능한 로봇 소프트웨어 컴포넌트에 대한 규격으로 로봇 컴포넌트에 대한 포괄적인 내용을 정의하고 있다. 경량 RTC는 컴포넌트, 컴포넌트의 내부 구조를 나타내는 프로퍼티(Property), 컴포넌트 간의 통신을 위한 포트(Port) 등을 정의하고 있다. 실행 시멘틱스는 로봇 응용에서 자주 사용되는 프로그래밍 패턴(Finite State Machine, Stimulus Response Processing 등)을 제공하여 개발자가 쉽게 로봇 응용을 개발할 수 있도록 한다. 인트로스펙션은 컴포넌트, 포트 등의 상태를 실행 시간에 검사할 수 있는 모니터링 인터페이스를 제공한다.

(그림 3)은 경량 RTC의 구조를 나타낸 것이다.

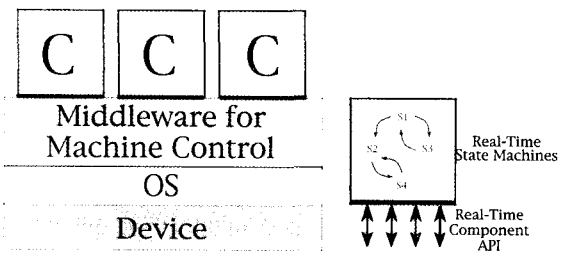


(그림 3) 경량 RTC 구조

4. OROCOS

OROCOS(Open Robot Control Software)[7]는 2000년 12월에 로봇 제어 소프트웨어 개발을 위한 오픈 소스 프로젝트로 시작되어, EU(유럽연합)의 EURON(European Robotics Network)과 결합하여 벨기에의 K.U.Leuven, 프랑스의 LAAS Toulouse, 그리고 스웨덴의 KTH Stockholm을 주된 파트너로 2001년 시작되어 2002년부터 결과를 내놓고 있다.

(그림 4)는 OROCOS의 구조를 나타낸 것이다.



(그림 4) OROCOS 구조

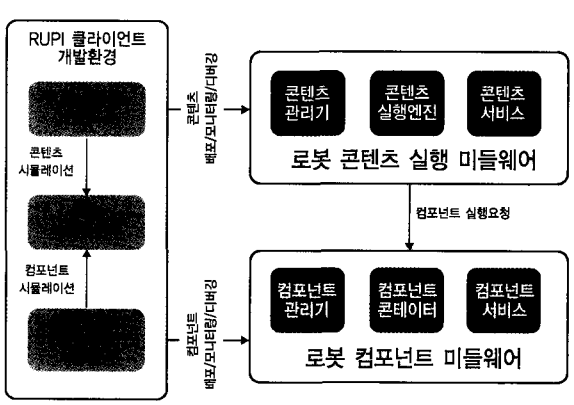
대부분의 실시간 제어 어플리케이션은 특정 운영 체제 위에서 개발되어 왔으며, 피드백 제어(feedback control)를 위한 체계적인 구조를 제공하지 못하였다. 물론 Simulink[8]와 같은 사용자 위주의 제어 시스템 디자인 툴킷이 존재하지만, 대규모 분산 환경 및 다양한 하드웨어 플랫폼에 적용되기에는 한계가 있었으며 이벤트 처리, 동기/비동기 태스크 프로그래밍에 대한 지원이 매우 부족하였다. OROCOS는 이러한 문제를 해결하기 위해 실시간 제어 응용과 이를 지원하는 기반 기술을 명확히 구분하였으며, 다양한 실시간 제어 응용을 개발하기 위해 실시간 툴킷(RealTime Toolkit, RTT)을 제공하고 있다. OROCOS는 실시간 컴포넌트가 가져야 할 컴포넌트 모델 및 인터페이스를 정의하고 있으며, 손쉽게 실시간 로봇 응용을 개발할 수 있도록 C++ 클래스로 이루어진 RTT와 스크립트 언어를 제공한다.

III. RUPI 클라이언트 소프트웨어 플랫폼

RUPI 클라이언트 소프트웨어 플랫폼은 로봇 내부 소프트웨어의 재사용성 및 상호 호환성을 보장하기 위해 컴포넌트 모델과 표준 인터페이스를 정의하고 있으며, 이를 기반으로 로봇 응용을 개발하도록 한다.

지능형 서비스 로봇 시스템이 복잡해지고 다양한 서비스를 제공하면서 컴포넌트 기반 로봇 응용 개발 및 컴포넌트 실행 미들웨어에 대한 연구가 활발하게 이루어지고 있다[9, 10]. 일반적으로 네트워크 기반 지능형 서비스 로봇은 로봇의 센서(Sensor) 혹은 엑추에이터(Actuator)를 접근하기 위한 하드웨어 디바이스 접근 컴포넌트, 음성 인식, 음성 합성, 얼굴 인식, 자율 주행 등의 로봇 구동 및 서비스에 필요한 알고리즘을 구현한 로봇 응용 컴포넌트 등의 다양한 컴포넌트를 이용한다. 이러한 컴포넌트들은 지능형 서비스 로봇 제조회사나 개발자에게 공통적으로 필요한 것들이며, RUPI 로봇 소프트웨어 플랫폼은 이러한 컴포넌트를 보다 손쉽게 개발하고 유지보수 할 수 있도록 표준화된 형태의 구조와 플랫폼을 제공한다. RUPI 규격에 맞게 개발된 로봇 컴포넌트들은 향후 다른 로봇을 개발할 때 재사용이 가능할 뿐만 아니라 개발 시간을 단축시킬 수 있으며, RUPI 규격에 맞게 개발된 다른 업체(third-party)의 로봇 컴포넌트도 용이하게 사용할 수 있는 장점이 있다.

네트워크 기반 지능형 서비스 로봇은 로봇 응용 컴포넌트와 네트워크를 통해 수집한 다양한 정보를 기반으로 사용자에게 일정 서비스, 교육 서비스, 오락 서비스, 음악 서비스, 보안 서비스, 청소 서비스 등의 다양한 서비스를 제공한다. 이러한 서비스를 기술하는 방법은 크게 두 가지로 요약될 수 있다. 하나는 C++나 Java와 같은 범용 프로그래밍 언어를 사용하는 방법[11, 12]으로 실행이 상대적으로 빠르고 개발자에게 익숙한 언어라는 장점이 있으나, 한번 컴파일 되어 로봇에 탑재되면 다시 컴파일 하기 전에는 변경이 불가능하다는 단점이 있다. 반면 스크립트 형태의 언어를 사용하는 접근방식[13, 14]은 실행이 느린 단점이 있지만 개발이 쉽고 로봇에 탑재된 후라도 수정이 가능하며 유지 보수 및 수정이 쉬운 장점이 있다. RUPI에서는 프로그램 언어로 개발된 로봇 컴포넌트를 조합하여 로봇 응용을 개발하는 방법과 스크립트 언어를 이용하여 개발하는 방법을 함께 지원하며,



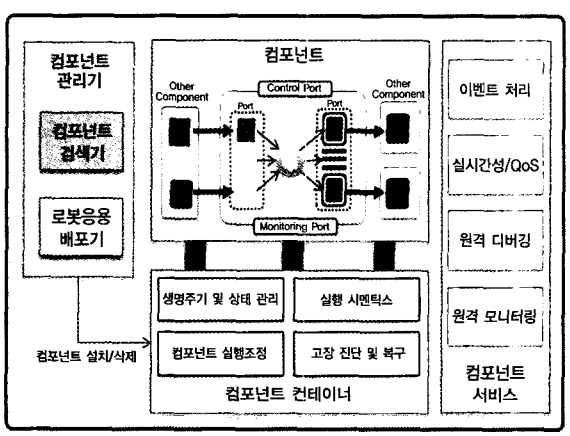
(그림 5) RUPI 클라이언트 S/W 플랫폼 구조

스크립트 형태로 작성된 로봇 서비스를 로봇 콘텐츠라고 칭한다.

RUPI 클라이언트 소프트웨어 플랫폼은 크게 로봇 컴포넌트 미들웨어, 로봇 콘텐츠 미들웨어, RUPI 클라이언트 개발 환경으로 이루어져 있으며, (그림 5)는 RUPI 클라이언트 소프트웨어 플랫폼의 개략적인 구조를 나타낸 것이다.

1. 로봇 컴포넌트 미들웨어

(그림 6)은 로봇 컴포넌트 미들웨어의 기본 구조를 나타낸 것이다.



(그림 6) 로봇 컴포넌트 미들웨어 기본 구조

로봇 소프트웨어 컴포넌트는 로봇에 존재하는 다양한 센서나 액추에이터와 같은 하드웨어를 제어하기 위한 장치 컴포넌트와 이러한 장치 컴포넌트를 이용하여 음성인식, 음성합성, 얼굴인식, 자율 주행과 같은 알고리즘을 구현하기 위한 로봇 응용 컴포넌트로 구분된다. RUPI 컴포넌트 규격에서는 두 가지의 개발 관점을 지원하기 위해 장치 컴포넌트 개발을 위한 장치 컴포넌트 인터페이스와 응용 개발을 위한 응용 컴포넌트 인터페이스, 그리고 컴포넌트간 통신을 위한 포트(Port) 인터페이스를 명시하고 있다[15, 16].

로봇 컴포넌트 미들웨어의 주요 기능을 기술하면 다음과 같다.

o 컴포넌트 관리 기능
- 컴포넌트 배포기: 컴포넌트를 조합하여 개발한 로봇 응용을 배포하는 기능
- 컴포넌트 검색기: 필요한 컴포넌트를 검색하는 기능

o 컴포넌트 컨테이너
- 컴포넌트 생명주기 관리 및 상태 관리: 배포된 컴포넌트의 실행 시작/중지 등의 생명주기 관리 및 실행 상태 관리 기능
- 실행 시멘틱스: 실시간/비실시간 등 컴포넌트 실행 특성에 따라 자원을 할당하고 해당 컴포넌트를 실행하는 기능
- 컴포넌트 실행 조정: 다수의 컴포넌트가 동시에 실행되었을 경우 컴포넌트의 우선순위, 실행 마감시간 등을 고려하여 컴포넌트의 실행 순서를 조정하는 기능
- 고장 진단 및 복구: 컴포넌트의 고장을 감지하고 이를 동일한 기능의 다른 컴포넌트로 대치/복구하는 기능

o 컴포넌트 서비스
- 이벤트 처리: 로봇에서 비주기적으로 발생하는 다양한 이벤트를 처리하기 위한 발행자/구독자(publisher/subscriber) 기반의 이벤트 처리 기능
- 실시간성/QoS: 로봇에서 발생하는 다양한 실시간성 및 QoS(Quality of Service) 요구사항을 처리하기 위한 기능
- 원격 모니터링: 원격에서 컴포넌트의 실행 상태 및 내부 정보를 모니터링하기 위한 기능
- 원격 디버깅: 개발된 로봇 응용을 원격에서 디버깅하는 기능

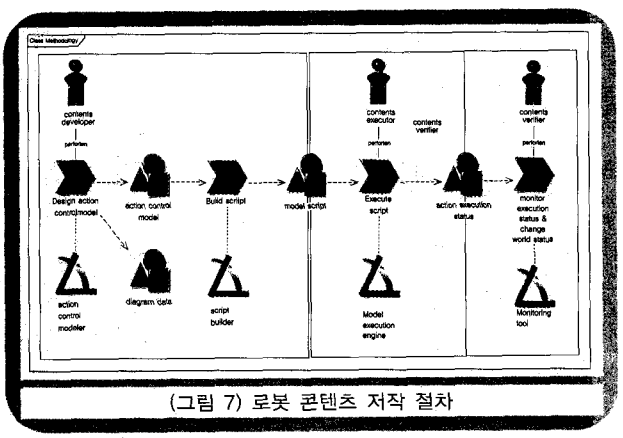
로봇 컴포넌트 미들웨어는 RUPI 컴포넌트 규격을 준수하여 개발한 로봇 소프트웨어 컴포넌트의 생명주기(시작/중지) 및 실행 상태를 관리한다. 또한 다양한 개발자에 의해 개발된 컴포넌트를 실행하고 상호 통신할 수 있는 기본 기능을 제공하며 이벤트 서비스, 실시간성/QoS, 고장감지 및 복구 등 로봇 응용 개발에 필요한 부가 기능을 제공한다.

2. 로봇 콘텐츠 실행 미들웨어

로봇 콘텐츠는 크게 두 가지 종류로 나눌 수 있다. 하나는 로봇 동작을 주로 나타내는 로봇 동작 콘텐츠이고, 다른 하나는 플래시나 멀티미디어 위주의 멀티미디어 콘텐츠이다. 로봇 동작 콘텐츠는 로봇이 동작하는 환경과의 상호작용 속에서 로봇이 어떠한 동작을 취할지를 결정하는 콘텐츠를 의

미한다. 따라서, 로봇 동작 콘텐츠는 기본적으로 외부 환경에서 발생한 다양한 이벤트를 기반으로 하는 이벤트 주도형 콘텐츠라고 할 수 있다. 반면에, 로봇 멀티미디어 콘텐츠는 멀티미디어 콘텐츠의 플레이와 더불어 간단한 형태의 로봇 동작을 병행하는 것으로 기본적으로 멀티미디어 주도(시작, 중지, 끝, 또는 멀티미디어 플레이 중의 특정 시간 프레임을 트리거링 신호로 하는)형 로봇 콘텐츠이다.

(그림 7)은 로봇 콘텐츠를 저작하는 절차를 나타낸 것이다 [17, 18].

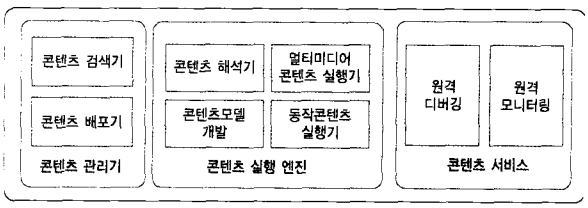


(그림 7) 로봇 콘텐츠 저작 절차

로봇 콘텐츠 개발절차에 참여하는 역할로는 콘텐츠 개발자(contents developer)와 콘텐츠 실행자(contents executor, 주로 April 2008_실행자가 됨), 콘텐츠 검증자(contents verifier)로 구분된다. 콘텐츠 개발자는 동작 콘텐츠 저작 도구를 사용하여 콘텐츠를 개발하고 이를 스크립트 저작 도구를 이용하여 스크립트 형태의 파일로 변환을 한다. 콘텐츠 실행자는 콘텐츠 실행 엔진을 이용하여 변환된 스크립트를 해석하고 실행한다. 콘텐츠 검증자는 모니터링 도구를 이용하여 실행되는 콘텐츠의 실행 상태 및 로봇의 상태를 모니터링 하고, 경우에 따라서는 콘텐츠 및 로봇의 상태 변경을 수행하여 콘텐츠를 검증한다.

(그림 8)은 로봇 콘텐츠 실행 미들웨어의 기본 구성 요소를 나타낸 것이며, 주요 기능을 기술하면 다음과 같다.

o 콘텐츠 관리기
- 콘텐츠 배포기: 콘텐츠를 미들웨어에 배포하는 기능
- 콘텐츠 검색기: 필요한 콘텐츠를 검색하는 기능



(그림 8) 로봇 콘텐츠 실행 미들웨어 구성요소

o 콘텐츠 실행 엔진
- 콘텐츠 모델: 로봇에 적합한 콘텐츠 모델 및 규격 개발
- 콘텐츠 해석기: 스크립트 언어로 작성된 콘텐츠를 해석하는 기능
- 멀티미디어 콘텐츠 실행기: 멀티미디어 콘텐츠를 실행하는 엔진
- 동작 콘텐츠 실행기: 동작 콘텐츠를 실행하는 엔진

o 콘텐츠 서비스
- 원격 모니터링: 원격에서 콘텐츠의 실행 상태를 모니터링하는 기능
- 원격 디버깅: 개발된 로봇 콘텐츠를 원격에서 디버깅하는 기능

로봇 콘텐츠 실행 미들웨어는 사용자가 저작한 동작 콘텐츠와 멀티미디어 콘텐츠를 실행하고 모니터링하기 위한 기본 기능을 제공한다.

3. RUPI클라이언트 개발 환경

RUPI 클라이언트 개발 환경은 로봇 컴포넌트 및 로봇 콘텐츠를 개발하기 위한 개발 도구와 이들을 디버깅하고 시뮬레이션하기 위한 로봇 시뮬레이터로 구성된다. RUPI 클라이언트 개발 환경의 구성 요소를 살펴보면 다음과 같다.

o 로봇 콘텐츠 개발 도구: 로봇 동작 콘텐츠 및 멀티미디어 콘텐츠를 저작하고 이를 로봇에 배포하여 모니터링 및 디버깅하기 위한 도구

o 로봇 S/W 컴포넌트 개발 환경: 단위 로봇 소프트웨어 컴포넌트 및 단위 컴포넌트를 조합한 로봇 응용을 개발하기 위한 도구이며, 개발 된 로봇 응용을 로봇에 배포하여 모니터링 및 디버깅하기 위한 도구

o 로봇 시뮬레이터: 동역학 기반 3D 로봇 시뮬레이터

RUPI 클라이언트 개발 환경은 로봇 콘텐츠, 로봇 컴포넌트 저작뿐 만 아니라 가상 로봇 과 가상 환경을 모델링하여 개발된 로봇 콘텐츠 및 로봇 컴포넌트가 제대로 동작하는지 시뮬레이션 할 수 있는 로봇 시뮬레이터를 포함하고 있다.

IV. 결 론

본고에서는 로봇 소프트웨어의 재사용성 및 상호호환성을 위해 컴포넌트 기반의 로봇 개발 프레임워크인 RUPI 클라이언트 로봇 소프트웨어 플랫폼에 대해 기술하였다. RUPI 클라이언트 로봇 소프트웨어 플랫폼은 다양한 로봇 응용 및 콘텐츠를 용이하게 저작하고 서비스하기 위한 표준 규격 및 참조 구현을 제공한다. RUPI클라이언트 소프트웨어 플랫폼을 통해 로봇 개발자는 개발하고자 하는 로봇 서비스에만 집중하고 실시간성/QoS, 컴포넌트 간 통신, 실행 시멘틱스, 동기화 처리 등의 기능은 미들웨어에서 처리하게 하여, 로봇 응용의 재사용성, 상호호환성, 개발 생산성을 크게 향상시킬 수 있다.

참 고 문 헌

[1] 이승익, 장철수, 정승욱, 김중배, "로봇소프트웨어 아키텍처 연구동향과 현황," 정보통신동향분석, 제20권, 제2호, pp. 1-13, 2005

[2] Ha Y. G., Sohn J. C., Cho Y. J., Yoon H. S., "Towards ubiquitous robotic companion: Design and implementation of ubiquitous robotic service framework," ETRI Journal, vol. 27, no. 6, pp. 666-676, Dec. 2005.

[3] MSRS, http://www.microsoft.com/korea/robotics/

[4] Fielding R. T., "Architectural Styles and the Design of Network-based Software Architectures," Ph.D. Thesis, UC Irvine, Information and Computer Science, 2000.

[5] ERSP, http://www.evolution.com/

[6] "The Robotic Technology Component Specification," OMG Adopted Specification, ptc/06-11-07, Nov. 2006

[7] OROCOS, http://www.orocos.org/

[8] Simulink, http://www.mathworks.com/

[9] Blum S. A., "From a corba-based software framework to a component based system architecture for controlling a mobile robot," Lecture notes in computer science, vol. 2626, pp. 333-344, 2003.

[10] Cabrera-Gamez J., Dominguez-Brito A., Hernandez-Sosa D., "Coolbot: A component-oriented programming framework for robotics," Lecture notes in computer science, vol. 2238, pp. 282-304, 2000.

[11] Sukhatme G. S., Matarik M. J., "Robots: intelligence, versatility, adaptivity," Communications of the ACM, vol. 45, issue 3, pp. 30-32, 2002.

[12] Gat E., "Reliable goal-directed reactive control for real-world autonomous mobile robots," Ph.D. Thesis, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, 1991.

[13] Georgeff M. P., Lansky A. L., "Reactive reasoning and planning," Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI-87), pp. 677-682, 1987.

[14] Simmons R., Apfelbaum D., "A task description language for robot control," Proceedings of Conference on Intelligent Robotics and Systems, Oct. 1998.

[15] "리치 클라이언트 로봇의 로봇 소프트웨어 컴포넌트 규격서," RUPI 2.0 규격서, RUPI 2.0.1-R11.3, 2007. (http://www.rupi.or.kr)

[16] "리치 클라이언트 로봇의 로봇 소프트웨어 컴포넌트 관리 규격서," RUPI 2.0 규격서, RUPI 2.0.1-R31.3, 2007. (http://www.rupi.or.kr)

[17] "리치 클라이언트 로봇을 위한 로봇 콘텐츠 규격서," RUPI 2.0 규 격 서 , RUPI 2.0.4-R11.1, 2007. (http://www.rupi.or.kr)

[18] "리치 클라이언트 로봇 S/W의 통합개발환경 규격서," RUPI 2.0 규 격 서 , RUPI 2.0.5-R11.2, 2007. (http://www.rupi.or.kr)

정 승 욱

1996년 전남대학교 학사
1998년 광주과학기술원 석사
1998년~현재 한국전자통신연구 선임연구원
관심분야:: 로봇 소프트웨어 아키텍처 분야, 컴포넌트 기반 로봇 응용 개발 분야, 지능형 서비스 로봇, 미들웨어 분야

이 승 익

1995년 연세대학교 학사
1997년 연세대학교 석사
2002년 연세대학교 박사
2002년 일본 ATR 연구소 연구원
2004년~현재 한국전자통신연구원 선임연구원
관심분야:: 로봇 소프트웨어 아키텍처 분야, 컴포넌트 기반 로봇 응용 개발 분야, 지능형 시스템, 복잡성 과학

김 성 훈

1995년 광운대학교 학사
1997년 광운대학교 석사
2007년 한양대학교 박사과정
1996년~현재 한국전자통신연구원 선임연구원
관심분야:: 로봇 소프트웨어 아키텍처 분야, 컴포넌트 기반 로봇 응용 개발 분야, 지능형 서비스 로봇, 미들웨어 분야