유비쿼터스 환경에서 온톨로지를 이용한 지능형 로봇의 소프트웨어 프레임워크

박제현O 홍광희 최중민

한양대학교 컴퓨터공학과

{jhparkO, khhong, jmchoi}@cse.hanyang.ac.kr

An Intelligent Software Framework using Ontologies for Robots in the Ubiquitous Environment

Jeahyun ParkO Kwanghee Hong Joongmin Choi

Department of Computer Science & Engineering, Hanyang University

요 약

유비쿼터스 환경에 관한 연구가 계속되면서 생활 지원형 로봇에 대한 관심도 함께 높아지고 있다. 공장이나 산업체에서 주로 사용되었던 로봇이 가전제품으로 일반 가정으로 도입되면서 기존의 로봇과 달리 새롭게 고려되어야 할 점들이 있다. 이 논문에서는 로봇의 지능에 해당하는 소프트웨어와 물리적 활동을 담당하는 하드웨어를 쉽게 분리하고, 변경할 수 있는 로봇의 소프트웨어 구조를 제안한다. 이 구조를 이용하여 로봇 사용자는 유연하게 하드웨어를 변경할 수 있으며, 자신의 목적에 맞는 지능 소프트웨어를 탑재시킬 수 있다. 계층적으로 구성된 소프트웨어 구조는 이후 유지관리에도 큰 이점을 제공하며 사용자에 많은 선택의 폭을 제공함으로써 로봇이 대중화되는데 기여할 수 있을 것이다.

1. 서 론

유비쿼터스 시대가 다가오면서 시장성 있는 미래 산업 중의 하나로서 가정용 로봇이 주목 받고 있다. 새로운 컴퓨팅 패러다임으로서의 유비쿼터스 환경은 집이나 회사, 판매 업소와 같이 평범한 일상 공간을 중심으로 형성되고 있으며, 각종 정보 기기들도 이런 환경에 맞게 변화하고 있다. 지금까지 산업 현장이나 과학 연구소에서 특별한 목적을 위해서 주로 이용되었던 로봇도 일상생활과 밀접한 관련이 있는 공간으로 도입하려는 연구가 진행되면서 자연스럽게 로봇의 머리에 해당하는 지능형 소프트웨어에 대한 관심도 높아지고 있다.[1]

로봇은 주어진 환경을 인지하고 현재 자신이 처한 상황을 이해하고 적절한 판단을 통해 인간에게 이로운 행동을 수행해야 한다. 이러한 과정을 지금까지의 로봇처럼 일괄적인 프로세스로 표현하는 것은 유연성이나 확장성에 있어서 새로운 환경에 적합하지 않다. 새로운 요구가 나타날 때마다 새 로봇을 필요로 한다면 여러 면에서 효율적이기 못하기 때문이다.

로봇이 가지는 하드웨어는 점차 발전하고 다양해질 것이다. 가정이나 사무실의 사용자들은 자신이 소유하는 로봇의 하드웨어를 자유롭게 변경할 수 있으며, 그에 따라 로봇이 할 수 있는 일의 범위도 변화할 것이다. 경우에 따라서 아주 작은 로봇이 많은 지능 정보를 다루어야 하는 일도 있을 수 있다. 로봇을 이용하여 위와 같은 업무를 수행하기 위해서 반드시 로봇이 가지는 하드웨어와 로봇의 지능은 분리되어야 한다. 이 논문에서는 로봇의 지능과 하드웨어를 분리하기 위해서 프레임워크 형태의 시스템 구조를 제안한다. 로봇은 하드웨어와 미들웨어, 그리고 지능형 소프트웨어로 구분되며 특별히 지능형 소프트웨어는 모듈 형태로 다른 구성요소와 분리된다. 또한 미들웨어와 소프트웨어는 자신의 데이터를 표현하기 위해 온톨로지를 사용하며, 이를 통해 상호 독립적(platform independent)인 구조를 지향한다.

특별히 이 논문에서는 기존 연구와 달리, 로봇이 가지는 하드웨어나 소프트웨어적 기능이 스스로 제어하거나

* 본 논문은 정통부 및 정보통신연구진흥원의 정보통신선도기반기술개발사업의 연구 결과로 수행되었습니다.

변경하기 어렵다는 가정을 한다.[12] 로봇의 변경 사항은 모두 사용자의 선택이며 로봇은 주어진 조건과 기능에 적응하여 사용자의 요구를 수행하여야 한다. 만일 로봇이 가진 기능이 사용자의 요구를 수용할 수 없는 경우에는 아무 행동을 취하지 않는다. 이 논문에서는 이러한 조건 하에 유연성과 확장성을 지닌 로봇의 소프트웨어 구조를 제안한다.

2. 관련 연구

2.1 소프트웨어 프레임워크

소프트웨어 공학에서 사용되는 프레임워크는 역의존 이론(Dependency Inversion Principle)에 따라 계층 구조에서 상위 계층이 하위 계층의 구현에 직접 의존하지 않고, 상위 계층이 추상적인 하위 계층의 인터페이스를 이용하도록 설계된 구조이다.[2]

이러한 프레임워크는 지속적으로 상위 계층을 재사용할 수 있으며, 상위 계층에 수정을 가하지 않고 하위 계층을 개선해나갈 수 있는 장점을 가지고 있다.

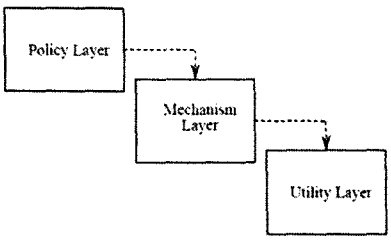

그림 3. 단순 의존 관계[2]

그림 1.에서 Policy Layer는 Mechanism Layer가 변경되었을 때 자신도 변경되어야 한다. 마찬가지로 Utility Layer가 변경된 경우 두 상위 계층은 모두 수정되어야 한다. 이런 경우에, 그림 2.와 같이 상위 계층이 하위 계층의 구체적인 구현에 직접 의존하지 않고 추상화된 인터페이스에 의존한다면 실제로 하위 계층의 구현에 관계없이 상위 계층을 사용할 수 있게 된다. 또한 Reflection과 같은 동적 바인딩 기법을 사용한다면 실행 시간에 구현을 변경할 수 있는 유연한 시스템을 구성할 수도 있게 된다.[11]

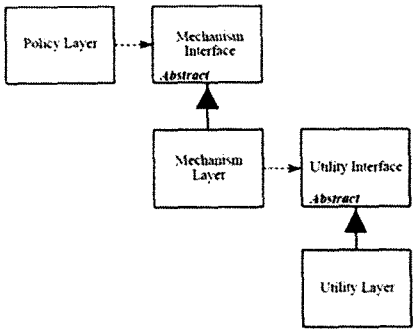

그림 4. 프레임워크 의존 관계[2]

2.2 온톨로지

전문가 시스템과 같이 소프트웨어가 받아들이는 정보가 지식의 형태로 표현되는 경우, 다시 말해서 여러 시스템에서 공유할 수 있는 형태로 정보를 표현해야 하는 경우, 시스템 특성과 무관하게 정보의 내용을 표시(self-describable)할 수 있어야 한다. 이 논문에서 위와 같은 정보를 표현하기 위하여 온톨로지를 사용하였다. 넓은 의미의 온톨로지는 우리가 접하는 모든 정보에 대한 정형적 기술을 뜻한다. 온톨로지에 포함된 개념들은 다시 세부적 개념으로 분할되거나 다른 개념과의 관계(relationships)를 통해 재정의 될 수 있으며 이러한 표현을 위한 제반 구조(schema)까지 포함한다.[3, 4, 5]

표 1. 온톨로지

온톨로지	설명
SHOE	메릴랜드 대학에서 개발 HTML 파일에 add-on하는 방식
OIL	OntoKnowledge 프로젝트의 일환 기술 논리 도입
RDF	W3C 표준 자원 표기법 Triple 구조 기술 논리 채택
DAML	DARPA 개발 지식 표현에 필요한 axiom 포함
OWL	W3C 표준 온톨로지 언어 시맨틱 웹 표준 언어

3. 시스템

전체 시스템은 다음과 같은 구조를 갖는다.

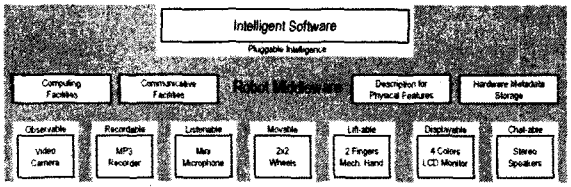

그림 5. 시스템 전체 구조

이 시스템은 크게 세 가지 계층 구조를 갖는다. 최하위 계층은 로봇과 연결된 하드웨어 구성 요소들을 나타내며, 상위의 지능형 소프트웨어는 로봇이 처리해야 할 작업에 대한 결정과 계획을 담당한다. 중간의 미들웨어는 일괄 프레임워크로서 주어진 작업과 하드웨어의 작동 사이의 조율을 담당한다.

3.1 하드웨어 계층

로봇은 물리적인 하드웨어를 가진다. 각 하드웨어가 가지는 목적이나 기능, 성능은 하드웨어 자체에 의존적이다. 이러한 요소들은 로봇이 할 수 있는 작업을 규정할 뿐만 아니라 로봇이 가지는 한계도 결정한다. 서로 다른 하드웨어가 상황에 따라 변경 되는 환경에서 지능형 소프트웨어가 적응적으로 하드웨어를 이용할 수 있는 방법은 표준화된 방법을 통해 하드웨어의 정보를 습득하고 이를 이용하여 적절한 작업을 수행하는 방법이다.

이 시스템에서는 하드웨어가 로봇과 연결되면 미들웨어로 하드웨어가 가진 기능 정보를 온톨로지 형태로 전달하도록 한다. 이렇게 전달된 기능 정보는 미들웨어 내부의 메타데이터 저장소(Hardware Metadata Storage)에 저장되며, 로봇은 간단한 질의 처리 과정을 이용하여 이 메타데이터에 접근할 수 있다. 기능적으로 미들웨어는 읽거나 쓰는 작업 이외의 복잡한 기능은 수행하지 않으며 메타데이터는 지능형 소프트웨어에서 직접적으로 사용된다.

3.2 로봇 미들웨어 계층

사용자가 자신에게 필요한 서비스를 위해, 로봇의 하드웨어를 변경시킨 경우, 로봇은 자신과 연결된 하드웨어에 적응하기 위해서, 먼저 하드웨어가 가진 기능의 성능과 한계에 대한 명세를 전달받는다. 이러한 명세는 온톨로지 형태로 기술되며, 로봇에 탑재된 지능이 수행해야 할 작업의 가능성을 판단하고 작업 계획을 수립하는데 사용된다. 일반적으로 로봇과 연결될 수 있는 하드웨어와 그 기능(Functionality)에 대한 개념적 체계는 다음과 같이 나타낼 수 있다.

표 2. 하드웨어 기능 분류

기능 (클래스)	하드웨어 (인스턴스)
Observable	비디오 카메라, 웹 캠 등
Listenable	마이크 등
Movable	바퀴, 캐터필러, 보행족 등
Lift-able	유압식 집게, 기중기 등
Displayable	액정 모니터, 프로젝터 등
Chat-able	스피커, 핸드폰 등

위와 같은 기능적 분류 체계는 다시 온톨로지를 이용하여 그림 4.와 같이 나타낼 수 있다. 각 기능은 자신의 한계를 설명하기 위해 속성(property)을 이용하며 실제로 연결되는 하드웨어는 속성에 해당하는 값을 미들웨어로 전달하여 메타데이터 저장소에 보관한다.

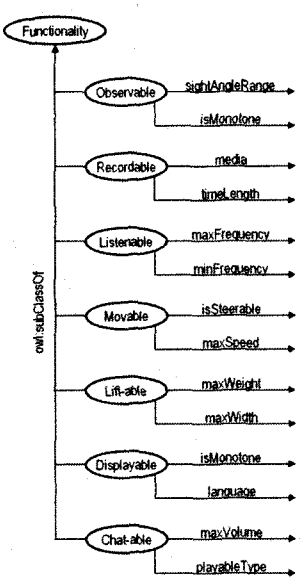

그림 6. 하드웨어 기능 온톨로지

3.3 지능형 소프트웨어 계층

로봇 프레임워크의 상단에 위치하는 지능형 소프트웨어는 다음과 같은 두 가지 주요 기능을 수행한다.

- *사용자의 요구가 실행 가능한 것인지 판단*
- *사용자의 요구가 실현되기 위해 필요한 계획 수립*

3.3.1 요구 판단

로봇은 하드웨어의 제약 때문에 사용자의 모든 요구를 수행할 수 없다. (예: 자신보다 좁은 문을 통과하라) 로봇의 지능은 주어진 요구가 현재 로봇이 처한 하드웨어 정보와 환경 정보를 바탕으로 수행 가능한 것인지 판단해야 한다. 만일 판단 결과가 긍정적일 경우 적절한 계획을 수립하여 사용자에게 가능한 서비스를 제공하게 되고, 그렇지 않을 경우 사용자에게 원하는 요구가 허용되지 않았음을 알리고 그에 따른 절차를 수행한다.

3.3.2 계획 수립

주어진 조건에서 사용자의 요구가 수행 가능하다고 판단된 경우에는 미들웨어의 메타데이터에 저장된 하위 하드웨어의 기능 정보를 이용하여 현재 상태에서 사용자 요구가 수락된 상태까지 전이할 수 있는 계획을 수립하여야 한다. 각각의 상태와 상태의 전이는 하드웨어가 가진 기능 정보의 조합으로 이루어지며, 로봇은 자신의 연산 능력 범위 내에서 가장 효과적으로 최종 목적 상태까지 도달할 수 있는 계획을 수립한다. 계획이 사용자가 원하는 최종 상태까지 도달 가능하게끔 수립된 이후 계획을 미들웨어에게 전달하고, 미들웨어는 그 계획을 순서대로 하드웨어에게 전달하여 동작을 실행시킴으로써 사용자가 원하는 동작을 수행하게끔 한다.

3.3.3 원격 지능

로봇이 수행해야 할 작업에 대한 지능적 연산을 수행할 수 있을 만큼의 CPU 연산 능력이나 메모리 기억 용량을 확보하지 못하는 경우에 로봇은 원격 지능을 이용할 수 있어야 한다. 원격 지능이란 지능적 과정을 처리할 수 있는 기능이 물리적으로 로봇과 떨어진 위치에 존재하는 경우를 이야기한다. 원격 지능은 이 지능을 이용하려는 로봇의 소프트웨어와 독립적으로 서비스를 제공할 수 있어야 한다. DIG 인터페이스는 원격 지능의 한 예이다. [6]

DIG는 HTTP 메시지에 XML 기반의 DIG 메시지를 함께 전송함으로써 추론 엔진과 같은 지능적 소프트웨어가 새로운 사실을 받아들이고, 이를 토대로 사용자가 원하는 추론을 하여 그 결과를 다시 보낼 수 있게끔 해준다. 이 때 원격 지능이 현재 추론을 요구하는 시스템이 가진 지식(KB)을 가지고 있지 않기 때문에 이 메시지에 지식 정보를 함께 전송하게 된다.

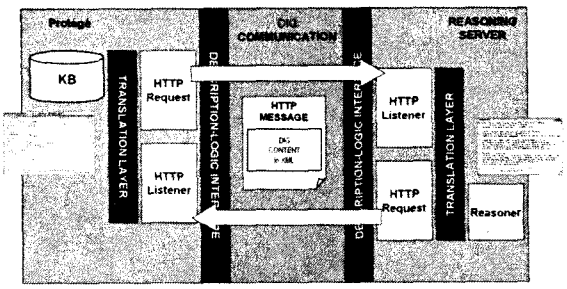

그림 7. DIG 개념도

DIG와 유사하게, 로봇 미들웨어가 가진 Communicative Facilities를 이용하여 CORBA나 Web Services와 같은 서비스를 통해서 지능 서비스를 이용할 수 있다. 향후 OWL-S나 WSML과 같이 온톨로지를 이용하는 서비스까지 이용할 수 있다면 로봇은 더욱 지능적인 작업이 가능해질 것이다.[7, 8]

4. 절차

4-1. 하드웨어 인식

하드웨어가 물리적으로 연결되면 하드웨어는 우선 자신이 정상적으로 작동할 수 있는지 진단한다. 만일 정상적으로 동작할 수 있다고 판단될 경우, 하드웨어 기능 명세를 온톨로지 형태로 전달하고, 동시에 자신을 구동할 수 있는 인터페이스의 구현체를 미들웨어에 연결한다. 이 정보들은 로봇 미들웨어의 Hardware Meta-data Storage에 저장된다. 하드웨어 변경으로 인해 로봇의 외형적 조건이나 규격 등이 변경된 경우 Physical Information에 그 정보를 함께 기록한다.

4-2. 요구 판단

사용자의 요구 사항은 하드웨어 중 Observable 하드웨어나 Listenable 등 입력이 가능한 형태의 하드웨어를 통하여 전달된다. 이 정보는 바로 로봇 지능으로 전달되며 지능은 이 요구가 현재 로봇이 수행할 수 있는지 판단한다. 1차적으로 로봇 미들웨어의 Physical Information과 온톨로지에 저장된 하드웨어 기능 명세를 이용하여 가능성을 판단한다. 예를 들어, Movable 하드웨어의 maxSpeed보다 빠른 속도를 요구하는 작업은 수행할 수 없기 때문에 적절한 방식으로 사용자에게 그 사실을 알리고 대처한다. 하지만 경로 탐색과 같이 명백히 결정할 수 없는 문제는 계획 수립 단계에서 다단계 탐색을 수행한 뒤 결정하게 된다.

4-3. 계획 수립

현재 위치에서 다른 위치로 물건의 이동을 명령한 경우, 물건의 이동은 **1) 물건의 위치까지 이동** (Movable 하드웨어) **2) 물건을 수취** (Lift-able 하드웨어) **3) 경로 탐색** (지능) **4) 로봇의 이동** (Movable 하드웨어) **5) 물건 수하** (Lift-able 하드웨어) 의 과정으로 이루어진다. 이 과정은 지능 소프트웨어의 계획 수립(AI planning)에 의해 결정된다. 다시 말하면, 로봇이 취할 수 있는 행동으로 인한 상태 공간에서의 변화에 대한 검색으로 이루어진다. 상태 공간에서 검색 결과로 상태 변화 연쇄가 사용자가 원하는 결과까지 연결된다면 단계에 따라 이 행동이 적용되며, 그렇지 않은 경우 사용자의 요구가 받아들여질 수 없다고 판단하고 적절한 대처를 한다.[9, 10]

5. 결론

이 논문에서는 유비쿼터스 환경에서 사용될 로봇이 효과적으로 이용할 수 있는 소프트웨어 프레임워크를 제시하였다. 이 프레임워크를 통해 로봇은 다음과 같은 이점을 갖게 된다.

- 유연한 하드웨어 적응성
- 다양한 지능형 소프트웨어 확보
- 간단한 유지보수 기반

이런 분리 관계는 비단 기술적인 이점뿐만 아니라 실제로 로봇 관련 제품을 이용하는 사용자들에게 선택의 폭을 제공함으로써 경제적인 상승효과를 불러올 수 있다. 또한, 여러 형태의 로봇을 빠르고 간단하게 조합함으로써 다목적 로봇을 실현할 수 있다. 탑재된 지능 소프트웨어가 어떤 소프트웨어인가에 따라 로봇 하드웨어에 연결된 같은 팔(Lift-able)이나 바퀴(Movable)가 서로 다른 목적으로 사용될 수 있는 점은 로봇의 실효성을 증대시키는 데 이점으로 작용할 것이다.

참고 문헌

[1] 국민로봇사업단 http://www.urcdc.or.kr

[2] Robert C. Martin, *Agile Software Development, Principles, Patterns, and Practices*, Prentice Hall, 2002

[3] Jeff Heflin, James Hendler, Sean Luke, *SHOE: A Knowledge Representation Language for Internet Applications*, Technical Report CS-TR-4078 (UMIACS TR-99-71), 1999

[4] Resource Description Framework (RDF), http://www.w3.org/RDF

[5] Web Ontology Language (OWL), http://www.w3.org/2004/OWL

[6] Sean Bechhofer, *The DIG Description Logic Interface: DIG/1.1*, Proceedings of DL2003 Workshop, 2003

[7] OWL-S: Semantic Markup for Web Services, http://www.w3.org/Submission/OWL-S

[8] WSML - Web Service Modeling Language, http://www.wsmo.org/wsml

[9] The UCPOP Planner, http://www.cs.washington.edu/ai/ucpop.html

[10] D. Nau, Y. Cao, A. Lotem, and H. Munoz-Avila, *SHOP: Simple Hierarchical Ordered Planner*, IJCAI-99, pp.968-973, 1999

[11] The Java Reflection API, http://java.sun.com/docs/books/tutorial/reflect

[12] 박수용, 장형수, 김동선, 고인영, 박연출, 이관우, 서비스 로봇을 위한 Self-Managed 소프트웨어 프레임워크 개발, 한국정보과학회지, 제24권, 제3호, pp.35~42, 2006