로봇 소프트웨어 모델링을 위한 관련기술 분석 및 UML 과의 호환성에 대한 연구

이규만*, 맹지찬**, 유민수**
한양대학교 지능형로봇학과*, 한양대학교 정보통신학부**
e-mail : {kmlee, jcmeang, msryu}@rtcc.hanyang.ac.kr

A Study for Robot Software Modeling Methods and UML compatibility

Kyu-Maan Lee*, Ji-Chan Maeng** and Minsoo Ryu**
Dept. of Intelligent Robot Engineering, Hanyang University*
College of Information and Communications , Hanyang University**

요 약

최근 서비스 로봇 산업에 대한 관심이 증가하면서 소프트웨어의 중요성이 크게 증가하고 있다. 이에 따라 본 논문에서는 서비스 로봇에 탑재되는 소프트웨어를 위한 다양한 모델링 방법을 분석하고, 아울러 순수 소프트웨어 분야의 대표적인 모델링 방법인 UML(Umified Modeling Language)과의 호환성 여부를 분석하였다. 그 결과 기존의 다양한 로봇 소프트웨어 모델링 방법이 UML 로 충분히 표현할 수 있음을 확인하였고, 이를 통해 차후 UML 기반의 로봇 소프트웨어 모델링 방법을 표준적인 방법으로 발전시키는 것이 가능할 것으로 판단된다.

1. 서론

전통적으로 로봇 산업이라고 하면 산업용이나 원격조정용 로봇을 의미하였으나 최근 뉴스 등 언론매체를 통해서도 어렵지 않게 접할 수 있을 만큼 서비스 로봇 산업에 대한 기대와 투자가 증가하고 있다.

개발하는 입장에서 보면 현재까지는 기업마다의 각자 로봇에 특화된 서비스 개발에 한정되어, 서비스의 질과 다양성을 확보하는데 다소 어려움이 있으며 연구의 중복투자가 발생하고 있다. 이에 로봇 서비스 소프트웨어간 호환성과 이를 진행하기 위한 소프트웨어 공학의 중요성이 점차 커지고 있다.

이에 각 기업의 각 로봇이나 특정기업의 소프트웨어 모델에 종속되는 서비스 개발보다는, 범용적으로 적용할 수 있는 서비스의 호환성 확보가 필요하고, 그로 인해 서비스 개발 환경에 대한 기술을 기업 차원에서 공유한다면 결국 서비스의 질과 양 두 면을 만족시킬 뿐만 아니라 로봇산업의 세분화, 고도화 및 시장 규모의 확대를 촉진시킬 수 있을 것이다.

이러한 관점에서 여러 가지 표준화 대상을 고려해 볼 수 있겠지만 본 논문에서는 로봇 소프트웨어 저작도구들을 분석하고 소프트웨어 업계의 실질적인 표준인 UML(Umified Modeling Language)을 이용하여 로봇 소프트웨어 모델링의 표현해 봄으로써 로봇 소프트웨어 모델링의 표준적인 방법으로 적용할 수 있는지 알아보고자 한다.

본 논문은 다음과 같은 내용을 포함한다. 첫째, 대표적인 로봇 소프트웨어 저작 도구인 ERSP, MSRS, CMU 의 BAT, NEC 사(社)의 RoboStudio 등의 저작방법에 대하여 조사 및 분석하였다. 둘째, UML 을 이용한 로봇 소프트웨어 모델링의 적합성을 분석한다. 이를 위해 기존의 로봇 소프트웨어 저작 도구들의 모델링 예제들을 UML 모델로 변환하여 표현하는 것이 가능한지를 검증하였다.

2. 관련기술 조사 및 분석

본 절에서는 기존의 로봇 소프트웨어 개발에 관련된 저작도구를 조사하여 저작도구 별 특성과 관련기술의 UML 로 표현 시 필요한 기술적 요소를 파악하였다.

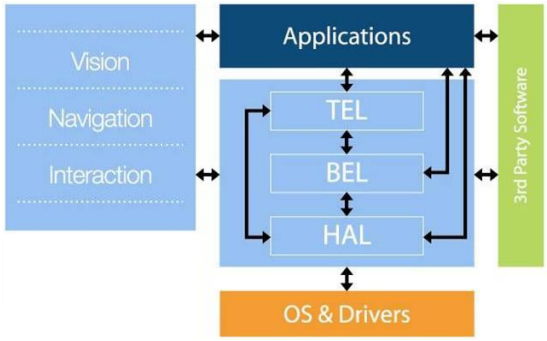
각 저작도구 내용 중 로봇 소프트웨어 모델링 방법을 위주로 정리하였다.

2.1. Evolution Robotics 사(社)의 ERSP 3.1

총 4 개의 C++ 라이브러리(Architecture, Interaction, Navigation, Vision)로 구성되어있다.

이 중 Architecture 를 담당하고 ERSA™로 명명된 라이브러리를 살펴보면 3 개의 Layer 와 이에 속한 소프트웨어 모듈로 이루어져 있는데 각 Layer 와 소프트웨어 모듈 간 인터페이스가 명확하게 정의되어 있고 각 Layer 는 그 목적에 따라 하드웨어 관리, 행위 생성, 그리고 각 애플리케이션에 대한 목적 지향적 업무로 구별된다. 이들은 Hardware Abstraction Layer

(HAL), Behavior Execution Layer (BEL), Task Execution Layer (TEL)이다.



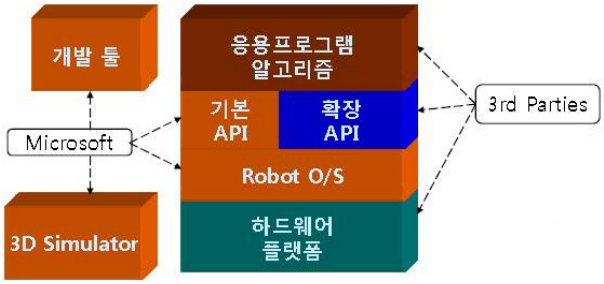
(그림 1) ERSP 의 구조

BEL 의 기본적인 구성요소는 여러 가지 입력 값에 의해 결정되는 Behavior 이다.

Behavior 는 일련의 입력값에서 일련의 출력물로 연결하는 연산의 단위로 정의되고 블록으로 표현되는데, 이는 센서값을 읽어오고 액츄에이터를 구동하는 것에서부터 수학적 연산과 알고리즘을 실행하는 것에 이르기까지 광범위한 기능을 의미한다. Behavior 는 여러 가지 형(type)의 입력과 출력을 갖을 수 있고 서로 같은 형의 입력과 출력을 연결하여 Behavior network 을 구성한다. 전형적인 Behavior network 의 경우, 데이터는 센서로부터 논리 Behavior 로 공급된다. 그 뒤로 연결되어 있는 Behavior 로 반복적으로 전달된다. 최종적으로 액츄에이터나 드라이브 메커니즘을 나타내는 Behavior 로 출력을 보낸다(그림.3 참조).

다른 라이브러리의 기능을 구현하는 Behavior 을 상비하고 있고 이 Behavior 들을 쉽게 연결할 수 있는 Behavior Composer 라는 GUI 툴을 제공한다.

2.2. Microsoft 사(社)의 Robotics Studio(MSRS) 1.5



(그림 2) MSRS 의 구조

개발툴 및 환경뿐 만이 아니라 개발된 DLL 모듈의 실행 런타임 어플리케이션(실행 플랫폼)도 함께 제공한다. 로봇을 위한 프로그래밍을 진행하는데 있어서 동시성의 문제를 해결하고 단순한 코딩 작업으로도 이러한 기능을 구현할 수 있도록 동시처리 및 제어 기술(CCR)을 제공한다. 서비스 기반의 런타임 아키텍처와 분산 애플리케이션 패턴은 MSRS 내에서 분산화된 소프트웨어 서비스 (Decentralized Software Services, DSS)로 불린다. MSRS 에서 제작된 실행 플랫폼은 DSS 와 CCR 을 기반으로 한다.

개발 툴로는 초급자용으로 Visual Programming Language(VPL)을 제공하고, 전문가용으로 C#, VB.Net, C++.Net, Python 을 권장한다. 개발툴을 통해 개발된 결과는 DLL 파일 형태로 생성된다.

VPL data flow 는 다른 activity 블록에 연결될 수 있는 입력과 출력을 가진 블록으로 표현된 activity 들의 연결된 순서로 구성된다.

Activity 는 사전에 빌드된 서비스, 데이터 흐름 제어, 함수 또는 다른 코드 모듈을 표현한다. 각 activity 는 연결 핀을 통하여 연결되는데 좌측의 연결 핀은 입력 메시지를, 우측의 연결핀은 출력 메시지를 위한 연결 점을 표시하고 이들을 연결하여 data flow 를 나타낸다.

또한 activity 는 다른 activity 들의 조합을 포함할 수 있다. 이것은 activity 를 조합한 애플리케이션의 재사용을 가능하게 한다.

2.3. Interbots 사(社)의 Behavior Authoring Tool(BAT)

Carnegie Mellon University 의 Entertainment Technology Center 에서 시작된 Interbots 프로젝트는 상대방과의 상호작용 경험을 축적하여 활용하는 대화형 로봇 기술 프로젝트로 로봇 하드웨어 및 소프트웨어, 그리고 콘텐츠 제작 도구까지를 모두 아우르는 프로젝트이다.

BAT 는 로봇의 behavior 를 state 간의 연결을 통해 나타내며, 이러한 behavior 와 시스템 입력값에 따른 state 전이 규칙들이 모두 데이터베이스로 저장되어 있다. Superstate 와 substate 로 구성된 behavioral model 과 이를 기반으로 동작하는 Character State Control System (CSCS)이라는 런타임 의사결정 시스템을 통해 행동을 결정한다. behavioral model 은 일반적인 유한상태기계 (finite state machine)처럼 각 substate 로 구성된 superstate 들간의 네트워크로 이루어진다.

2.4. NEC 사(社)의 Robostudio

로봇의 behavior 와 action 은 프로그래밍 언어로 기술된 scenario 에 의해 정의되며 로봇은 scenario 를 해석하여 action 을 만든다. 복잡한 action 은 script 로 기술하거나 시나리오 개발 자동화 도구를 사용하여 시나리오를 작성한다. 자동화 도구는 블록을 조합해서 쉽게 프로그래밍이 가능하다. 로봇이 외부 자극에 대응하는 action 과 dialogue 들을 디자인 할 수 있는 Scenario Editor 와 목, 다리, LED 등의 복합 동작을 디자인하기 위해 사용하는 Action Editor 를 제공한다.

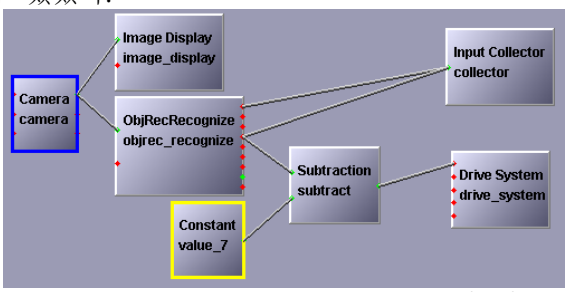
3. 기존의 로봇 SW 모델과 UML 모델의 비교 검증

본 절에서는 UML 기반의 모델링 방법을 제안하기에 앞서 UML 이 기존의 로봇 콘텐츠 모델을 충분히 표현할 수 있는지를 분석하고 그 결과를 기술하였다.

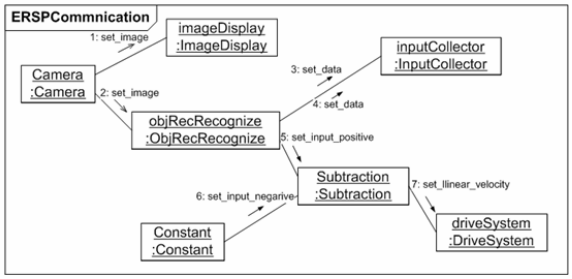
앞절에서 소개한 로봇 소프트웨어 저작도구들의 모델 예제들을 UML 모델로 변환하여 표현하는 것이 가능한 지를 검토하였다. 여러 다이어그램으로 표현 가능한 경우 지면 관계상 대표적인 것만 제시하였다.

3.1. ERSP 3.1

Class Diagram, Communication Diagram, Component Diagram 의 3 가지 다이어그램을 사용하여 표현할 수 있었다.



(그림 3) ERSP Behavior Network 의 예

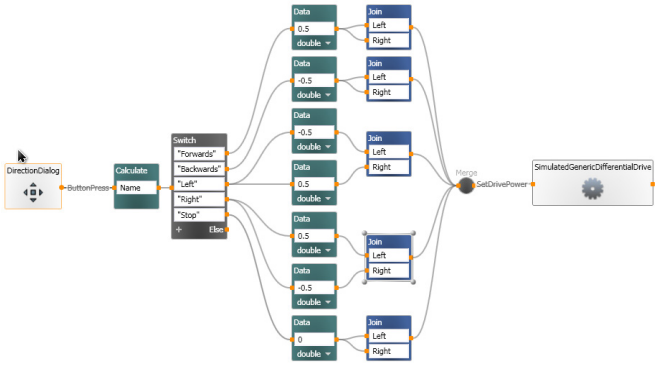


(그림 4) Behavior Network 예제변환 - Communication Diagram

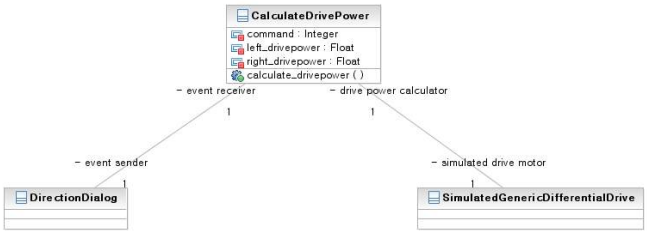
Class Diagram 에서는 각 클래스 간의 정적인 관계만을 표현하므로 서로간의 호출 관계나 메시지 전달과 같은 동적인 관계를 표현하지 못한다. 따라서, 객체간의 상호 데이터 전달 관계를 표현하기 위해 Communication Diagram 을 사용할 수 있다.

3.2. MSRS 1.5

Class Diagram, Communication Diagram, Activity Diagram 으로 나타내는 것이 가능하였다.



(그림 5) VPL diagram 의 예

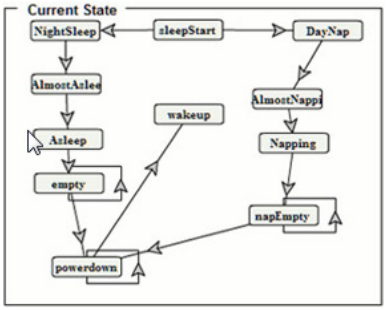


(그림 6) MSRS 예제변환-Class Diagram

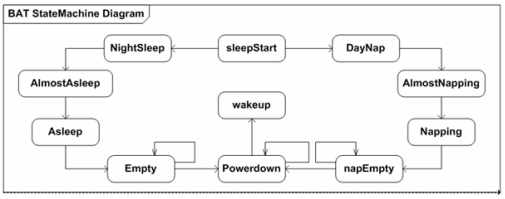
Class Diagram,으로 연산부분을 CalculateDrivePower 라는 클래스로 표현하고, 내부에서 다루는 좌우바퀴에 대한 데이터를 저장하는 변수를 속성으로 지정하여 표현하였다.

3.3. Behavior Authoring Tool

계층적인 상태 다이어그램으로 표현된 로봇의 행위를 작성하는 도구로서 UML 의 State Diagram 을 그대로 사용하는 것이 가능하다.



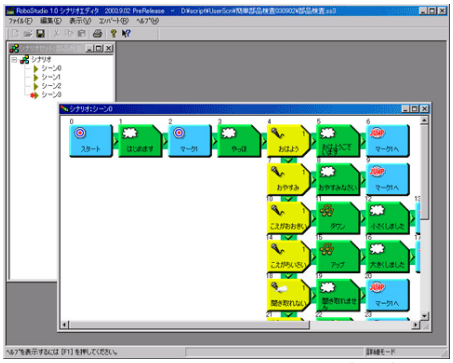
(그림 7) CSCS 의 예



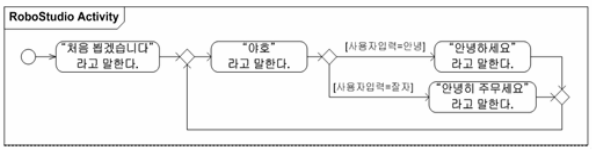
(그림 8) CSCS 예제변환-State Diagram

3.4. Robostudio

Scenario Editor 의 예제는 Activity Diagram 으로 표현할 수 있다.

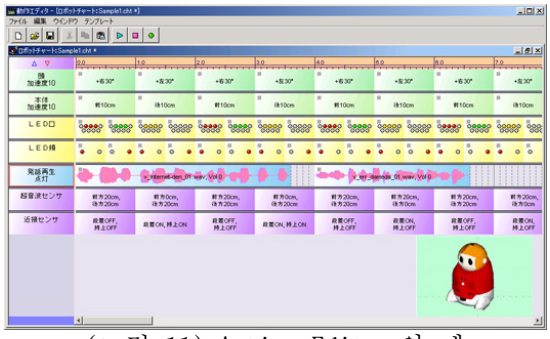


(그림 9) Scenario Edior 의 예

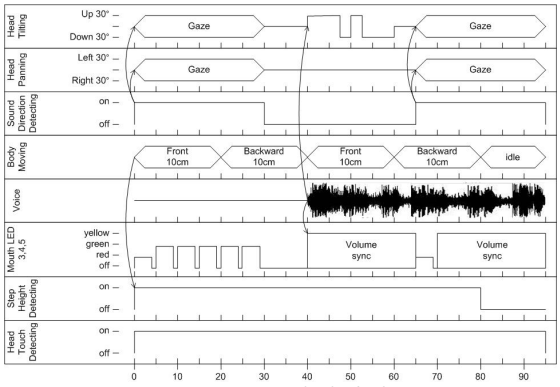


(그림 10) Scenario Editor 예제변환 - Activity Diagram

Action Editor 의 예제는 각 부분을 객체로 표현하고 이들간의 관계를 Timer 객체를 추가하여 정의하고, 시간을 기준으로 하는 동기화 표현은 Timing Diagram 을 통해 표현하는 것이 가능하다.



(그림 11) Action Editor 의 예



(그림 12) Action Editor 예제변환-Timing Diagram

3.5. RUPI2.0

국내의 로봇 플랫폼 표준인 RUPI 2.0 의 경우 Rich Client 와 Thin Client 로 나누어지는데, 각각 Rich Client 는 Activity Diagram 으로, Thin Client 는 Class Diagram, Communication Diagram, Timeline Diagram 으로 변환가능하였다.

4. 결론

UML 2.0 의 경우 총 13 가지의 다이어그램을 정의하고 있지만, 각 다이어그램의 의미가 독립적이지 않고 상당히 중첩적이다. 예를 들면, Communication Diagram 과 Sequence Diagram 은 시각적인 형태가 다를 뿐 다이어그램이 표현 하는 의미는 거의 동일한 경우이다. 본 보고서에서는 아래와 같은 6 가지의 UML 다이어그램을 로봇 콘텐츠 모델링에 필요한 다이어그램으로 선정하였다.

- Class Diagram
- Communication Diagram
- Component Diagram
- Activity Diagram
- State Diagram
- Timing Diagram

추후 현재 국내적으로 진행중인 RUPI 2.0 에도 표준화가 적용이 되어 통일성, 호환성등에 기여할 수 있을 것으로 기대해 본다.

참고문헌

[1] 이승익 외 3 人, 로봇 소프트웨어 아키텍처의 연구동향과 현황, 전자통신동향분석, 2005 년 4 월.

[2] 이순요, 휴머노이드 로봇 기술정보분석, 한국과학기술정보연구원, 2005 년 11 월.

[3] Michael Somby, "A review of robotics software platforms," Available at: http://www.linuxdevices.com/articles/AT5739475111.html,Aug.2007.

[4] Evolution Robotics, ERSP 3.0 User's manual, Available at: http://www.evolution.com

[5] Munich, M.E., Ostrowski, J. Pirjanian, P., "ERSP: a software platform and architecture for the service robotics industry," IROS 2005, IEEE/RSJ International Conference on 2-6 Aug. 2005, Pages: 460 – 467

[6] R.A. Brooks, "A Robust Layered Control System for a Mobile Robot," IEEE Journal on Robotics and Automation, Vol.2, No.1, 1986.

[7] E. Gat. "On three-layer architectures," In D. Kortenkamp et al., editor, *AI and Mobile Robots*. AAAI Press, 1998.

[8] Micorsoft, Microsoft Robotics Studio 1.5 Documentation, Available at: http://msdn2.microsoft.com/ko-kr/robotics

[9] 김영준 , MS 로보틱스 스튜디오, Available at: http://cafe.naver.com/msrskorea

[10] NEC Personal Robot Research Center, Available at: http://www.incx.nec.co.jp/robot

[11] CMU ETC, Interbots Project home, Available at: http://www.etc.cmu.edu/projects/ibi/

[12] Interbots, http://www.interbots.com/

[13] Sabrina Haskell, Andrew Hosmer, Eugenia Leu, "An extensible platform for interactive, entertaining social experiences with an animatronic character, 2005 ACM SIGCHI International Conference on Advances in computer entertainent technology," ACM International Conference Proceeding Series; Vol. 265, Pages: 141 – 148

[14] OMG, UML Profile Specifications, Available at: http://www.omg.org/technology/documents/profile_catalog.htm

[15] OMG, UML® Resource Page, Available at: http://www.uml.org

[16] 한국지능로봇산업협회, Robot Unified Platform Initiative (RUPI) 2.0, Dec. 2007, Available at: http://www.rupi.or.kr

[17] M Kim, S Kim, S Park, M.T. Choi, M Kim, H Gomaa, UML-based service robot software development: a case study, Proc. of 28th Int. Conf. on Software engineering2006, Shanghai, China, May, 2006, pp. 534-543.