모델기반 시스템 설계 방법을 이용한 용접로봇의 상부아키텍쳐 정의에 관한 연구

A Study on Architecting Method of a Welding Robot Using Model-Based System Design Method

김 진 일*, 박 영 원
(Jin-Ill Kim and Young-Won Park)

Abstract : This paper describes the application of a model-based system design method critical to complex intelligent systems, PSARE, to a welding robot development to define its top level architecture. The PSARE model consists of requirement model which describes the core processes(function) of the system, enhanced requirement model which adds technology specific processes to requirement model and allocates them to architecture model, and architecture model which describes the structure and interfaces and flows of the modules of the system. This paper focuses on the detailed procedure and method rather than the detailed domain model of the welding robot. In this study, only the top level architecture of a welding robot was defined using the PSARE method. However, the method can be repeatedly applied to the lower level architecture of the robot until the process which the robot should perform can be clearly defined. The enhanced data flow diagram in this model separates technology independent processes and technology specific processes. This approach will provide a useful base not only for improvement of a class of welding robots but also for development of increasingly complex intelligent real-time systems.

Keywords : PSARE method, welding robot, architecting, systems engineering, requirement model, architecture model, control spec, extended data flow diagram

I. 서론

시스템의 상부아키텍쳐를 정의한다는 것은 시스템의 구성요소, 각 구성요소간의 관계, 각 구성요소가 수행해야 하는 기능 및 제약사항을 정의하는 것이다[1]. 이러한 시스템 상부아키텍쳐 정의의 목적은 고객, 사용자 및 기타 이해당사자의 모든 요구사항과 제약사항을 시스템의 규격으로 변환하여 기계, 전기, 소프트웨어 등 각 전문분야의 엔지니어가 세부 설계를 수행할 수 있도록 하는 것이다. 이렇게 상부 아키텍쳐를 먼저 엄격히 정의함으로서 모든 이해당사자의 요구사항이 설계에 반영되도록 하고, 각 전문분야의 엔지니어들에게 제시되는 설계 목표에 대한 논리적 근거를 제시할 수 있게 된다. 점점 복잡해지는 지능형 시스템들의 개발에 있어서 비싼 시행착오를 최소화하기 위해서는 이러한 단계적 시스템 개발의 접근 방법이 필수적이다.

본 연구의 대상인 용접로봇은 내장된 용접 프로그램을 해석하여 용접을 수행하고, 사용자가 용접작업을 직접 프로그래밍 할 수도 있는 용접로봇이다. 이러한 용접로봇의 개발에 있어 지금까지는 세부 컨트롤러의 설계나 소프트웨어 설계에 많은 노력을 들이고 있는 것이 현실이다. 그러나 상부아키텍쳐의 굳건한 논리적인 설계목표 없이 세부 설계를 진행하는 것은 항상 잠재적인 문제를 안고 설계를 진행하는 것이며, 이러한 잠재적인 문제는 향후 얼마만큼의 대가를 치러야 할지 모르는 일이다.

* 책임저자(Corresponding Author)
논문접수 : 2004. 6. 7., 채택확정 : 2004. 12. 1.
김진일, 박영원 : 아주대학교 시스템공학과
(jikim@se2u.com/ywpark@madang.ajou.ac.kr)

본 연구에서는 이러한 국내 현실을 고려하여 용접로봇의 상부아키텍쳐를 정의하는 절차와 세부적인 방법을 소개하고 이의 적용사례를 제시하였다. 이렇게 체계적으로 정의된 용접로봇의 아키텍쳐는 기술의 진보에 의한 기존 로봇의 개선이나 적용 현장에 따른 재설계에 있어 훌륭한 출발점으로 사용될 수 있다. 본 논문에는 주로 방법의 적용사례를 중심으로 기술하였으며 대상 로봇 제조회사의 설계지식 보호차원에서 시스템의 최상위 수준 아키텍쳐만을 제시하였으나, 실제에서는 약 제 3수준까지 세부 아키텍쳐를 정의하여야 세부 설계자들이 이해할 수 있는 수준의 설계규격을 제시할 수 있을 것이다. 하부 수준의 아키텍쳐 정의는 최상위 수준의 아키텍쳐 정의 방법을 그대로 적용하면 된다. 본 연구에서 사용한 아키텍쳐 정의 방법은 PSARE 방법이다. PSARE 방법은 국제시스템공학회(INCOSE : International Council on Systems Engineering)에서 제시하는 모델기반 시스템엔지니어링의 접근방법과 같은 맥락을 유지하고 있으며, 로봇과 같은 모든 실시간 시스템의 아키텍쳐 정의를 위해 개발된 방법이다.

II. PSARE 방법의 소개

PSARE (Process for System Architecture and Requirements Engineering)의 약어로서 Derek Hatley[2] 등이 개발한 실시간 시스템의 아키텍쳐 정의를 위한 방법이다. 이 방법은 Derek Hatley가 초기에 개발한 방법[3]을 수정한 것이다. Hatley가 제시한 방법은 전통적인 구조적인 방법에 유한상태기계(Finite State Machine)이론을 접합시켜 제어규격을 기술하고, 아키텍쳐를 표현할 수 있는 다이어그램을 통합시킨 것이다.

소프트웨어 시스템에서는 모든 언어를 UML로 통일시켜 사용하고 있으나, UML은 일반적인 시스템을 표현하기에는 다음과 같은 단점이 있다[6].

- 연속적인 거동의 표현
- 거동 및 구조의 계층적 표현
- 시스템, 하부시스템 및 컴퍼넌트의 표현 등

이러한 UML의 한계 때문에 본 연구에서는 PSARE 방법을 적용하였다.

PSARE 방법을 완벽하게 구현한 소프트웨어는 아직 없다. 그 이유는 최근 몇 년 동안 PSARE 방법론과 이의 사용에 대한 개선이 이루어 졌기 때문이다. 본 연구에서 사용한 TurboCASE/SysTM는 Hatley 등이 컨설팅에 참여하고 있는 회사에서 PSARE 방법론을 부분적으로 구현한 소프트웨어로서 PSARE 방법론을 구현한 유일한 소프트웨어로 볼 수 있다. 본 연구에서는 TurboCASE/SysTM를 이용하여 용접로봇의 상부아키텍처를 정의하였으므로 PSARE 방법에 대해서도 TurboCASE/SysTM에서 구현된 범위에 한하여 소개하고자 한다.

PSARE 방법은 크게 요구사항(Requirement)모델과 아키텍처(Architecture) 모델 그리고 이 두 모델을 연결시키기 위한 확장데이터흐름도(EDFD, Enhanced Data Flow Diagram)로 구성되어 있다. 요구사항과 아키텍처를 분리하는 것은 시스템엔지니어가 시스템의 필수적인 기능을 식별함에 있어서 그 기능의 구현에 대해 불필요하게 제약을 받는 것을 방지하기 위함이다. 이러한 방법은 개발문제 정의와 해결책 정의를 분리하여 시스템엔지니어가 시스템 요구사항을 만족시키기 위해서 가용한 기술을 가장 잘 이용할 수 있는 설계자에게 상세설계에서의 의사결정을 맡길 수 있도록 해 준다.

1. 요구사항 모델

요구사항 모델은 시스템이 수행해야 할 필수적인 기능, 즉 무엇을 수행해야 하는가를 표현한 것이다. 이러한 기능을 PSARE 방법에서는 프로세스라고 부른다.

PSARE 방법에서는 시스템의 기능을 정의하기 위하여 전통적인 구조적 분석방법과 유한상태기계 이론을 조합하였다. 이 요구사항 모델링에 사용되는 다이어그램은 정황도(Context Diagram), 데이터흐름도(DFD, Data Flow Diagram), 제어흐름도(CFD, Control Flow Diagram)들이 있다. 또한 데이터 흐름도상에 나타나는 각 프로세스를 기술하기 위한 프로세스 규격이 있으며, 제어규격을 나타내기 위해서는 상태전이도(STD, State Transition Diagram)가 사용되며, 의사결정표(Decision Table)는 프로세스 규격이나 제어규격 모두에 적절히 사용된다.

또한 프로세스 간에 전달되는 데이터 흐름, 제어흐름을 정의하기 위한 요구사항사전(Requirement Dictionary)이 있다. 각 다이어그램에 대한 설명은 모델링한 사례를 기술한 Ⅳ장에서 구축한 모델과 함께 설명하고자 한다. 각 다이어그램의 작성법 및 모델 구축 방법에 대한 상세 설명은 참고문헌 [2,3]에 나타나 있다.

2. 아키텍처 모델

아키텍처 모델은 시스템을 구성하는 구성요소인 모듈을 나타내고, 이 모듈들 간의 교환되는 데이터 및 물질, 에너지 등을 나타내며, 모듈간의 물리적인 연결 구조를 나타낸다. 이러한 아키텍처 모델에 사용되는 다이어그램은 아키텍처 정황도(ACD, Architecture Context Diagram), 아키텍처 연결정황도(AICD, Architecture Interconnect Context Diagram), 아키텍처 연결도(AID, Architecture Interconnect Diagram), 아키텍처 흐름도(AFD, Architecture Flow Diagram)들이 있다. 또한 아키텍처 간에 전달되는 흐름을 정의하기 위한 아키텍처 사전(Architecture Dictionary)과 아키텍처 모듈 규격, 아키텍처 연결규격이 있다.

3. 요구사항 모델과 아키텍처 모델의 연결

개발문제와 해결책 간의 일관성 있는 시스템 설계를 위해서는 요구사항 모델과 아키텍처 모델간의 추적성이 확보되어야 한다. 또한 각 아키텍처 모듈이 수행해야 하는 기능, 아키텍처간의 인터페이스의 정의가 요구사항 모델에 근간을 두고 있어야 한다. PSARE 방법에서는 이러한 요구사항 모델과 아키텍처 모델의 연결을 위해 확장데이터흐름도(Extended Data Flow Diagram)를 사용한다. 확장데이터흐름도는 데이터흐름도상에 있는 각 프로세스들을 아키텍처 모듈에 할당한 그림을 그릴 수 있도록 해 주는 다이어그램이다. 따라서 이 다이어그램을 통해서 문제영역인 요구사항과 이의 해법인 아키텍처 모듈을 동시에 볼 수 있다.

PSARE 방법의 특징과 장점을 요약하면 다음과 같다.

- 계층화된 모델을 이용하여 시스템요구사항을 시스템의 적절한 수준에서 기술할 수 있도록 해준다.
- 다이어그램을 통해서 시스템의 기능적, 물리적 인터페이스를 명확하게 보여준다.
- 요구사항 모델과 아키텍처 모델의 연결고리를 제공하여 기능적 인터페이스와 물리적 인터페이스의 일관성을 유지할 수 있도록 해준다.

Ⅲ. 상부아키텍처 정의 절차

시스템의 상부아키텍처를 정의하기 위한 절차를 모든 경우에 적용할 수 있도록 구축할 수는 없다. 이는 대상 시스템이 처음 개발하는 경우와 이미 개발한 시스템을 개선하는 경우에 달라질 수 있고, 기술의 수준, 인력 및 가용자원에 따라 프로세스가 달라질 수 있기 때문이다. 결국 적용할 절차는 정황에 따라 시스템엔지니어가 정의해야 한다. 그러나 시스템 엔지니어가 절차를 정의할 때 반드시 필요한 것은 기본 절차이다. 실제 수행할 절차는 이러한 기본 절차를 조정(tailoring)하여 정의되기 때문에 기본 절차는 매우 중요한 것이다. PSARE 방법의 핵심적인 기본 절차는 그림 1과 같다.

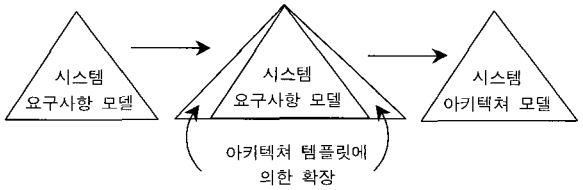


그림 1. 시스템 아키텍처 정의 기본 절차.

Fig. 1. Basic procedure for architecting a system.

그림1에서 제시한 기본 절차는 시스템의 한 계층구조 내에서 반복적으로 수행될 수 있다. 또한 경우에 따라서는 시스템 아키텍쳐 모델이 먼저 구축될 수도 있다. 이러한 기본 절차는 시스템이 계층적으로 분해 됨에 따라 각 계층구조 내에서 반복적으로 수행된다.

그림 1의 기본 절차를 바탕으로 본 연구에서 수행한 하향식 절차는 다음과 같다.

① 시스템 최상위 요구사항 모델 구축
② 확장데이터흐름도 작성
③ 시스템 최상위 아키텍쳐모델 구축 및 요구사항 할당
④ 요구사항 추적표 작성

위의 절차는 시스템의 최상위 모델을 완성하기 위한 절차이다. 위에서 정의된 각 아키텍쳐 모듈에 대해서 다음 단계를 수행하여 시스템의 제 2수준의 모델링을 수행한다.

⑤ 아키텍쳐 모듈에 할당된 확장데이터흐름도의 구성요소를 이용하여 새로운 데이터흐름도를 생성
⑥ 새로 생성된 데이터흐름도를 기반으로 데이터흐름도를 완성하고, 프로세스 규격, 제어흐름도, 제어규격을 작성
⑦ 해당 아키텍쳐 모듈에 대해서 ②에서 ③까지의 절차를 수행

Ⅳ. 프로세스 수행 결과

본 연구에서 모델링한 대상용접 로봇은 6축 다관절 아크용접 로봇이다. 로봇의 세부 규격은 제품 보안사항이므로 본 연구에서는 대상 로봇을 일반화시켜 모델링 하였다. Ⅲ장에서 제시한 과정에 대한 세부 과정 및 수행 결과는 아래와 같다. 본 연구에서 사용한 TurboCASE/SysTM은 한글사용이 제한되어 있으므로 모델링에 영어를 사용하였다.

1. 시스템 최상위 요구사항 모델 구축

시스템의 최상위 요구사항 모델을 구축하기 위한 세부 절차는 그림 2와 같다.

1.1 정황도 작성

정황도(Context Diagram)는 시스템과 인터페이스를 갖는 외부 시스템을 정의하여 시스템의 기능적인 경계를 정의하고, 외부시스템과 교환되는 흐름(데이터, 정보, 에너지)을 정의하기 위하여 사용되는 다이어그램이다.

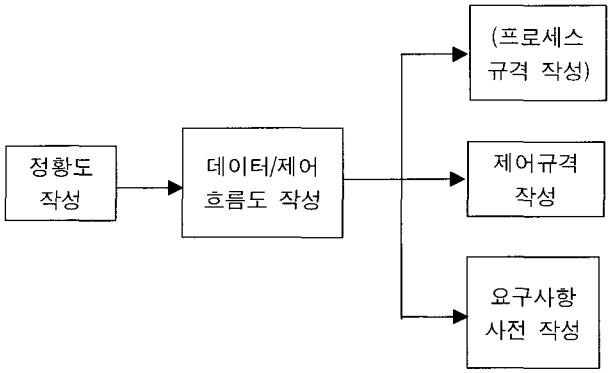
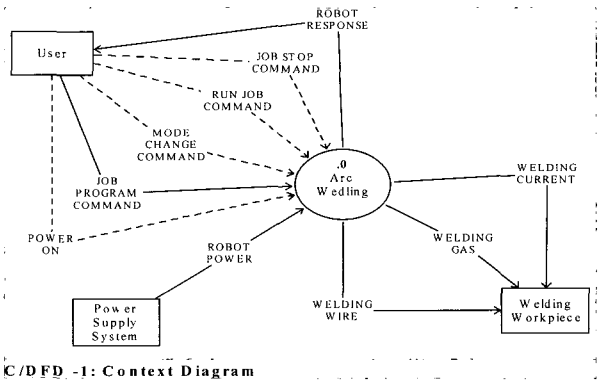


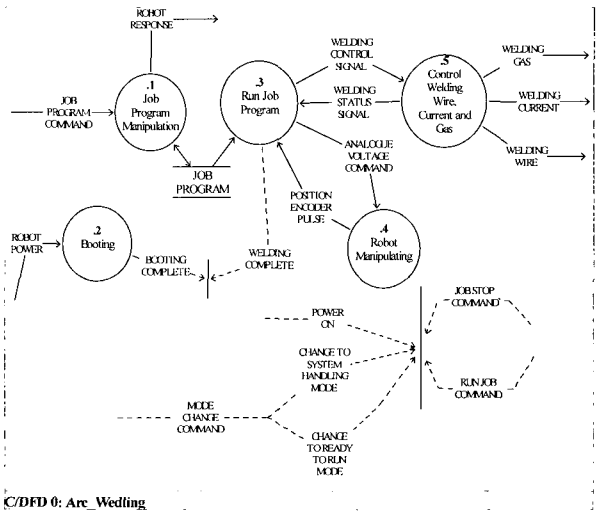
그림 2. 시스템 최상위 요구사항 모델 구축 과정.
Fig. 2. Procedure for system top level requirement modeling.

본 과제에서 작성한 용접로봇의 정황도는 그림 3과 같다. 그림 3에서는 아크용접을 수행하기 위해서 시스템이 외부로부터 받는 데이터, 제어 흐름 및 수행한 결과 외부에 제공하는 데이터 흐름을 보여주고 있다. 그림 3의 정황도는 아크용접로봇을 수행하기 위하여 외부에서 입력되는 사용자의 명령, 전원이 식별되어 있고, 아크용접 수행과 관련된 출력 흐름으로서 사용자에게 현황데이터를 제공하고, 용접부재에 전류, 용접봉 및 용접가스를 제공하는 기능적 인터페이스를 나타내고 있다.



○ : 시스템의 최상위 기능(프로세스)
□ : 외부 시스템
⟶ : 데이터 흐름
⋯⟶ : 제어 흐름

그림 3. 정황도.
Fig. 3. Context diagram.



○ : 외부의 입력을 출력으로 변환하기 위한 프로세스(기능)
| : 제어규격 막대(Control Spec Bar)
——— : 데이터 저장(Data Store)

그림 4. 시스템 수준의 데이터/제어 흐름도.
Fig. 4. System level D/CFD.

정황도나 뒤에 나오는 데이터 흐름도에서 사용되는 데이터의 흐름은 물질, 정보, 에너지 등의 흐름을 총칭하는 단어이다. 정황도에 사용된 영문명들은 별도의 설명이 필요치 않을 것으로 생각되어 생략한다.

1.2 데이터/제어 흐름도 작성

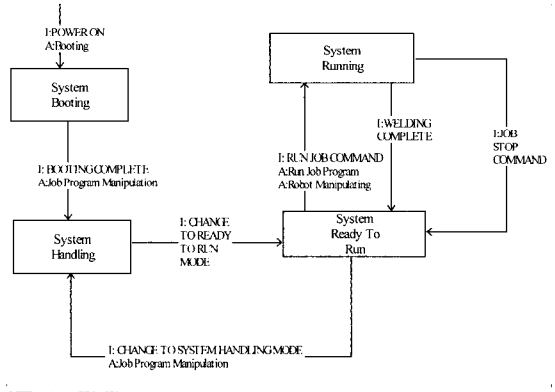
정황도에서 정의한 시스템의 최상위 프로세스는 다시 하부 프로세스로 분해 되며, 이때 데이터/제어 흐름도가 작성된다. 데이터흐름도와 제어흐름도는 독립적인 다이어그램으로 볼 수 있으나, 동일한 프로세스를 사용하므로 한 다이어그램에 나타낼 수 있다. 이때 최상위 기능에서 식별된 모든 입·출력 흐름은 데이터/제어 흐름도에서 모두 사용되어야 한다. 이것은 상·하부의 모델간의 일관성을 확보하기 위한 것으로 PSARE 방법에서는 균형맞추기(Balancing)라 한다. 이렇게 상·하부 모델간의 입·출력에 대한 균형을 맞추는 작업은 논리적 오류 없는 설계를 위해서 필수적인 작업이다. 이러한 균형맞추기는 데이터/제어흐름도 및 기타 모든 다이어그램의 검증에 적용되는데, TurboCASE/SysTM에서는 이러한 모델의 일관성 검증 기능을 제공하고 있다.

시스템 수준의 데이터/제어 흐름도는 그림 4와 같다.

그림 4는 정황도에 식별된 외부시스템과의 입·출력 흐름을 모두 사용하고 있음을 알 수 있다. 시스템 레벨의 데이터/제어 흐름도에서는 외부로부터 입력된 데이터흐름이 내부적으로 어떤 프로세스를 거쳐 원하는 출력으로 변환되는지를 나타낸다. 그림 4에서는 사용자의 명령에 따라 용접로봇이 용접 프로그램을 해석하여 각 관절을 움직이고, 해석된 신호에 따라 용접전류, 용접봉 및 가스를 제어하여 최종 용접을 수행하는 그림을 나타내고 있다. 이러한 다이어그램에 나타내는 엘리먼트의 수는 G.A. Miller에[4] 의하면 7±2개를 권장하고 있다. 이 범위내의 사실이나 객체 등을 동시에 다룰 때 인간의 능력이 최대가 된다는 것이 G.A Miller의 주장이다.

1.3 제어 규격 작성

그림 4에 나타난 각 프로세스는 데이터가 입력되면 즉시 그 프로세스가 수행되는 개념으로 작성되어 있다.



I : Input Signal(상태 전이를 일으키는 이벤트)

A : Activation

그림 5. 상태전이도.

Fig. 5. State transition diagram.

따라서 시스템이 수행할 기능과 입출력은 식별되지만 시스템의 거동 즉 제어의 측면은 나타나 있지 않다. 시스템으로 입력되는 모든 제어흐름은 제어규격막대로 입력되게 되어 있다. 시스템의 한 수준에서 제어규격은 단 하나만 존재한다. 따라서 데이터/제어 흐름도에 제어규격이 여러 개 나타나더라도 이들은 모두 동일한 것이다. 제어규격은 메모리가 없는 경우 간단한 의사결정표(decision table)로 나타낼 수 있으나 메모리가 있는 시스템의 경우에는 상태전이도(state transition diagram) 또는 상태전이표로 나타낼 수 있다. 본 연구에서는 상태전이도를 이용하여 시스템의 제어규격을 모델링 하였다.

그림 5는 시스템 수준의 데이터/제어흐름도에 대한 제어규격이다. 그림 5에서는 시스템의 상태에 따라 입력되는 제어신호에 대한 시스템의 거동과 활성화되는 프로세스를 보여주고 있다. 즉 초기 전원이 입력되게 되면 내부 운영프로그램을 부팅하는 상태로 이동하게 되며 이때 Booting 이라는 프로세스가 활성화된다. 부팅이 끝나면 시스템 핸들링 상태로 전이하게 되며 이때 용접 프로그램의 입·출력과 관련된 명령들을 받아들이는 프로세스가 활성화 되게 된다. 이 상태에서 사용자의 상태전환 명령이 입력되면 시스템이 용접준비상태로 전이되며 이 상태에서 용접수행 명령이 입력되면 용접프로그램 해석과, 로봇몸체를 움직이는 프로세스가 활성화 되어 용접이 수행되게 된다. 용접이 수행되는 도중에 용접중지 명령이 입력되거나 용접이 완료되게 되면 로봇은 다시 용접대기상태로 돌아오게 된다. 그림 4와 그림 5를 비교하면, 그림 4의 제어규격막대로 입력되는 모든 제어명령이 그림 5의 상태전이를 일으키는 이벤트로 작용하게 되며, 그림 4에 나타난 모든 프로세스는 그림 5의 상태 전이시 활성화되는 프로세스들과 일치한다.

따라서 시스템의 기능과, 제어의 측면이 일관성을 갖게 된다. 또한 시스템엔지니어는 상태전이도를 이용한 제어규격 작성을 통하여 사용자의 실수에 의한 명령이 부적절한 시스템의 상태에서 작동되지 않도록 제어로직을 구성할 수 있다.

Name	Definition	Type
AC POWER	*Power Supplied to Robot*	Data
JOB PROGRAM COMMAND	[LOAD_JOB_PROGRAM_COMMAND | EDIT_JOB_PROGRAM_COMMAND] *User commands related with job program*	Data
MODE CHANGE COMMAND	[CHANGE_TO_SYSTEM_HANDLING_MODE | CHANGE_TO_READY_TO_RUN_MODE]	Data/ Control

그림 6. 요구사항 사전.

Fig. 6. Requirement dictionary.

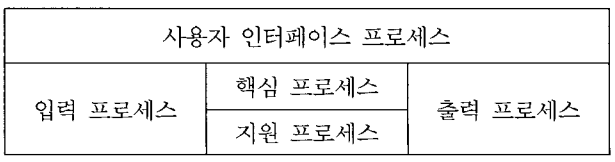


그림 7. 아키텍쳐 템플릿.

Fig. 7. Architecture template.

1.4 요구사항 사전 작성

요구사항 사전은 요구사항 모델에서 정의한 데이터/제어 흐름을 정의한 테이블이다. 요구사항사전에는 흐름의 이름과 정의, 형태를 기술한다. 흐름의 정의에는 흐름의 구성, 흐름에 대한 설명 등을 나타낸다.

그림 6에서 [ ] 내부는 흐름의 구성을 나타내며 *표는 흐름에 대한 설명을 나타낸다.

1.5 프로세스 규격 작성

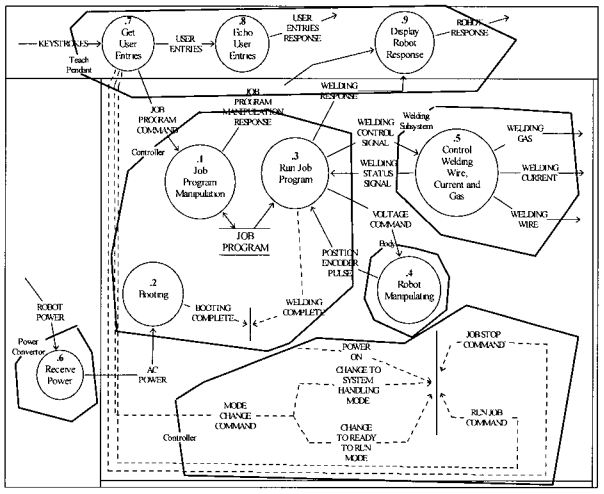
프로세스 규격은 입력을 어떻게 출력으로 변환시키는가를 기술하는 것이다. PSARE 모델에서는 프로세스를 하부로 분해해 가면서 명확히 규격을 기술할 수 있을 경우 이를 기본(Primitive)프로세스로 정의하고 프로세스 규격을 작성한다. 따라서 최상위 수준의 아키텍팅 과정에서도 기본프로세스에 해당하는 프로세스가 있다면 프로세스 규격을 작성해 주어야 한다. 그러나 최상위 수준에서 기본프로세스로 볼 수 없다면 그 프로세스는 하부수준의 프로세스로 분해 되어야 한다. 즉 모든 프로세스는 하부 프로세스로 분해 되거나 규격을 갖거나 둘 중의 한 가지를 만족시켜야 오류가 없게 된다. 그림 4에 나타난 데이터흐름도에 있는 프로세스는 아직 추상적인 프로세스로서 세부 프로세스로 분해 되어야 설계자들이 이해할 수 있는 수준의 프로세스 규격을 작성할 수 있다. 프로세스 규격의 작성법에 대해서는 본 논문의 IV.5 하부 모듈의 아키텍팅 부분에 상세히 기술하였다.

2. 확장데이터흐름도 작성

확장데이터흐름도는 두 가지 목적을 위해서 사용된다. 첫째는 시스템의 핵심 프로세스를 지원하기 위한 보조 프로세스의 식별이고, 둘째는 요구사항 모델에서 식별된 프로세스를 시스템의 물리적 아키텍처에 할당하는 것이다. 확장데이터흐름도에서는 첫째 목적을 달성하기 위해서 아키텍처 템플릿을 사용한다. 그림 7은 아키텍처 템플릿을 보여주고 있다. 그림 7의 아키텍처 템플릿에서 중앙에는 시스템의 핵심 프로세스를 나타내게 된다. 이 핵심 프로세스는 요구사항모델 구축을 통해서 작성된 데이터/제어 흐름도가 그대로 들어가게 된다.

중앙의 핵심 프로세스 주위에 있는 입·출력 및 사용자 인터페이스, 지원 프로세스 등은 핵심 프로세스를 수행하기 위한 보조적인 프로세스로 볼 수 있다. 이러한 보조적인 프로세스는 기술에 의존하게 된다. 즉 핵심 프로세스 수행을 위한 사용자의 명령을 지금은 교시조작기(Teach Pendant)를 이용하지만, 음성인식을 이용할 수도 있고, 좀 더 지능화된 로봇이라면 사람이 입력해야 할 내용을 스스로 탐색하여 입력할 수도 있을 것이다. PSARE 방법은 이처럼 기술에 의존하는 프로세스를 핵심프로세스와 분리함으로서 여러 분야의 전문가들이 팀을 이루어 설계할 때 각자 집중해야 할 분야를 쉽게 식별할 수 있도록 지원해고 있다. 또한 기술의 변화에 따라 변경해야 할 부분을 쉽게 식별할 수 있도록 해준다.

그림 8은 본 연구에서 작성한 확장데이터흐름도이다. 그림 8에서 프로세스를 둘러싸고 있는 다각형을 볼 수 있는데 이를 슈퍼버블(supper bubble)이라고 한다. PSARE 방법에서는 이러한 슈퍼버블을 이용하여 각 프로세스를 수행할 아키텍처 모듈을 나타내게 된다. 아키텍처 모듈은 확장데이터흐름도에서 생성할 수도 있고, 별도의 아키텍처 모델에서 작성할 수도 있다. 확장데이터흐름도의 중요성은 문제영역인 요구사항 모델과, 해법영역인 아키텍처 모델을 한 다이어그램에 나타낼 수 있다는 것에 또 다른 중요성이 있다.



EC/DFD Enhanced Arc Wedling

그림 8. 확장데이터 흐름도.

Fig. 8. Extended data flow diagram.

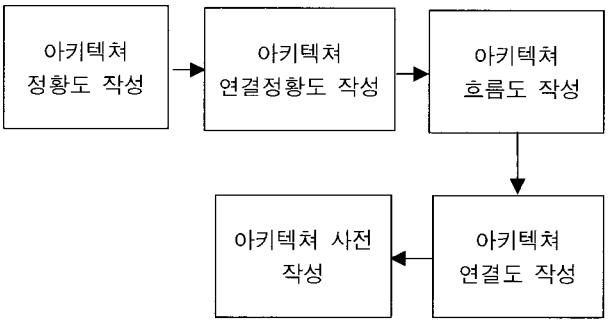
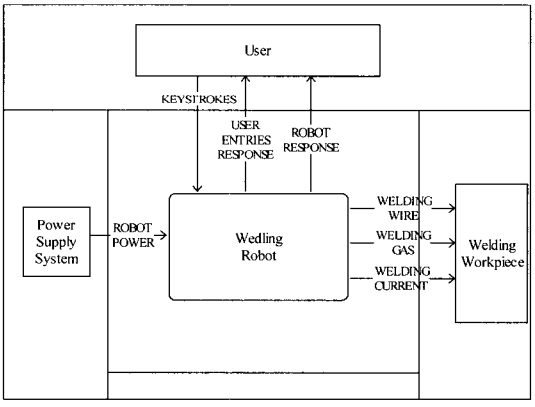


그림 9. 시스템 최상위수준 아키텍처 모델 구축 과정.

Fig. 9. Procedure for system top level architecture modeling.



ACD: Arc Wedling.ACD

그림 10. 아키텍처 정황도.

Fig. 10. Architecture context diagram.

확장데이터 흐름도 프로세스를 둘러싼 슈퍼버블을 만드는 것은 하나의 아키텍쳐 모듈을 생성하는 것과 동일한 효과를 갖는다. 그림 8에는 사용자 인터페이스를 담당하는 교시조작기와 외부 전력의 입력을 담당하는 전력변환기, 로봇 내부의 핵심 프로세스를 수행하는 제어기, 로봇본체, 용접기 등의 슈퍼버블이 각 구성품이 수행해야 할 프로세스 및 입·출력과 함께 나타나 있다.

TurboCASE/SysTM을 사용할 경우 시스템의 최상위 수준의 아키텍쳐는 반드시 확장데이터흐름도를 이용해서 만들어야 하지만, 하부 수준에서는 아키텍쳐 모델에서 모듈을 먼저 만들고 확장데이터흐름도에서 이를 참조하여 슈퍼버블을 만들 수도 있다. 그러나 실질적으로는 시스템의 최상위 수준이라 하더라도 아키텍쳐 모듈을 먼저 어느 정도 정해 두고 요구사항 모델과 확장데이터흐름도를 작성하게 된다고 PSARE 방법을 만든 Hatley가 주장하고 있다[3].

3. 시스템 아키텍쳐 모델 구축

아키텍쳐라는 말은 일반적으로 많이 사용되는 단어이지만 이에 대한 해석은 사용자에 따라 조금씩 다른 것 같다. PSARE 방법에서 정의한 아키텍쳐는 "시스템의 주요한 물리적 속성, 형태, 구조, 상호작용 및 목적"을 말한다.

본 연구에서 시스템 최상위 아키텍쳐 모델을 구축하기 위하여 수행한 절차는 그림 9와 같다.

그림 9의 각 단계에서 수행한 모델링의 내용을 상세히 나타내면 다음과 같다.

3.1 아키텍쳐 정황도 작성

아키텍쳐 정황도는 시스템이 외부시스템과 주고받는 흐름을 나타낸다. 요구사항 모델에서 작성한 정황도는 기능의 경계를 식별할 수 있도록 하는 반면, 아키텍쳐 정황도는 시스템과 외부시스템간의 정보의 경계를 식별할 수 있도록 해준다. 그림 10에서 보이는 시스템과 외부시스템간의 흐름의 교환은 요구사항 모델에서 나타내는 데이터 및 제어흐름과 일치해야 한다. 그러나 아키텍쳐 모델에서 나타나는 흐름은 데이터나 제어를 구분하지 않고 단지 그러한 흐름이 존재한다는 것만을 나타낸다는 점이 다르다. 이 흐름은 확장데이터흐름도에 나타나는 외부와의 데이터 및 제어흐름과 일관성이 있어야 한다.

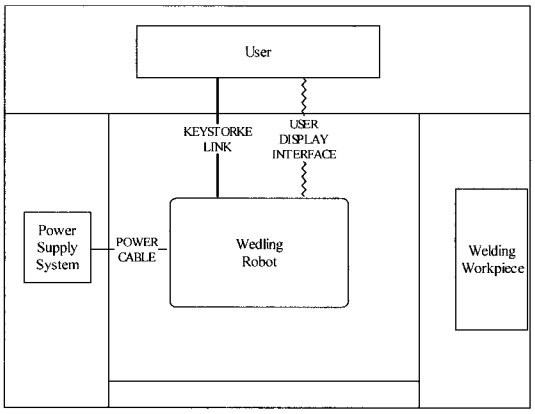
3.2 아키텍쳐 연결정황도 작성

아키텍쳐 연결정황도는 시스템이 외부시스템과 주고받는 흐름이 전달되는 물리적인 경로의 연결(Link)을 보여준다. 그림 11은 용접로봇의 아키텍쳐 연결정황도를 보여주고 있다. 아키텍쳐 연결정황도에 사용되는 물리적인 링크는 매우 다양하기 때문에 사용되는 기호의 의미를 모두 부여할 수는 없다. 그러나 PSARE 방법에서는 기본적으로 다음과 같은 기호의 의미를 권장하고 있으며, 본 연구에서 이 권장안을 사용하였다. 그림 11에서는 사용자가 키보드를 이용하여 입력할 수 있는 기계적인 연결과, 사용자의 입력 내용이나 로봇의 상태를 볼 수 있도록 하는 광학적 연결, 그리고 전기적인 연결을 볼 수 있다. 이러한 연결 역시 확장데이터흐름도상에 식별된 기술의존 프로세스 즉, 입·출력, 사용자 인터페이스 프로세스와 일관성이 있어야 한다.

3.3 아키텍쳐 흐름도 작성

아키텍쳐 흐름도는 시스템을 구성하는 아키텍쳐 모듈 간에 전달되는 흐름을 나타내는 다이어그램이다. 여기에 나타나는 흐름 역시 데이터흐름도나 제어흐름도에서 식별된 흐름과 일관성을 가져야한다. 그러나 아키텍쳐 흐름도에서는 데이터 흐름과 제어 흐름을 구별하지 않는다. 단지 그러한 흐름이 있다는 것만을 식별한다.

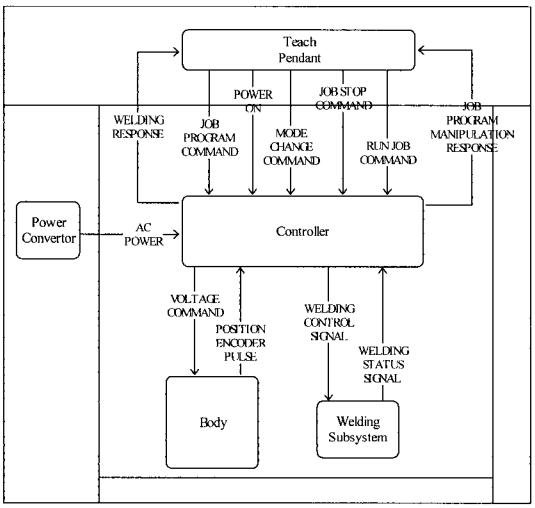
그림 12는 시스템수준의 아키텍쳐 흐름도를 보여주고 있다. 아키텍쳐흐름도에 나타난 아키텍쳐 모듈들은 확장데이터흐름도의 슈퍼버블로 나타내어진 아키텍쳐 모듈과 정확히 일치한다. 또한 시스템 수준의 데이터/제어 흐름도에 식별된 모든 흐름이 아키텍쳐 흐름에 나타나야 한다. 이렇게 하여 요구사항 모델과 아키텍쳐 모델간의 일관성이 확보되게 된다.



AICD: Arc Wedling.AICD

——— : 전기적 연결
∿∿∿ : 광학적 연결
▬▬▬ : 기계적 연결

그림 11. 아키텍쳐 연결 정황도.
Fig. 11. Architecture interconnect context diagram.



AFD: Wedling Robot.AFD

그림 12. 아키텍쳐 흐름도.
Fig. 12. Architecture flow diagram.

3.4 아키텍쳐 연결도 작성

그림 13은 본 연구에서 작성한 시스템 수준의 아키텍쳐 연결도를 보여주고 있다.

아키텍쳐 연결도는 아키텍쳐 연결정황도의 작성법과 동일하지만, 아키텍쳐 연결정황도에는 시스템과 외부시스템이 나타나는 반면, 아키텍쳐 흐름도는 시스템의 내부 구성요소인 아키텍쳐 모듈이 나타난다는 점이 다르다. 아키텍쳐 연결도를 통하여 시스템 내부의 인터페이스가 설계된다.

그림 13에는 사용자 인터페이스를 담당하는 교시조작기(Teach Pendant)와 제어기를 포함하여 각 아키텍쳐 모듈간의 물리적인 인터페이스를 보여주고 있다.

3.5 아키텍쳐 사전 작성

아키텍쳐 사전은 각 아키텍쳐 흐름의 정의와, 흐름의 방향, 흐름이 전달되는 물리적인 인터페이스(채널) 등을 나타내는 표이다. 그림 14는 본 연구에서 구축한 모델의 아키텍쳐 사전의 일부이다. 흐름의 정의 안에 내용 중 [ ] 표시 안에 있는 부분은 흐름의 구성을 나타내며, *표는 흐름에 대한 설명을 나타낸다.

4. 요구사항 추적표 작성

그림 15는 시스템 수준의 데이터흐름도와 아키텍쳐 흐름도의 요구사항 추적표를 보여주고 있다.

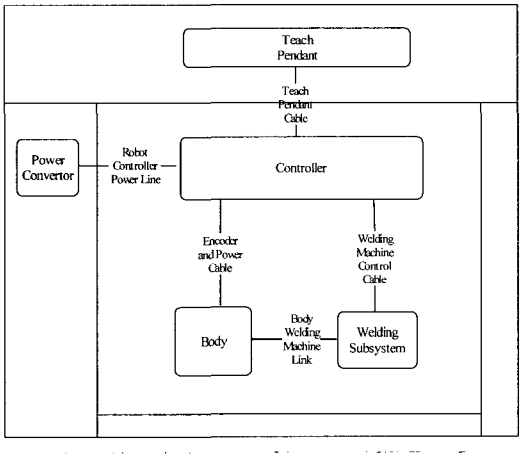


그림 13. 아키텍쳐 연결도.

Fig. 13. Architecture interconnect diagram.

Name	Definition	Type	Source	Destination	Channel
AC POWER	*Power Supplied to Robot*	Data	Power Convertor	Controller Robot Controller Power Line	
JOB PROGRAM COMMAND	[LOAD_JOB_PROGRAM_COMMAND | EDIT_JOB_PROGRAM_COMMAND] *User commands related with job program*	Data	Teach Pendant	Controller	Teach Pendant Cable
MODE CHANGE COMMAND	[CHANGE_TO_SYSTEM_HANDLING_MODE | CHANGE_TO_READY_TO_RUN_MODE]	Data/ Control	Teach Pendant	Controller	Teach Pendant Cable
POSITION	TBD	Data	Body	Controller	Encoder and

그림 14. 아키텍쳐 사전.

Fig. 14. Architecture dictionary.

요구사항 추적표는 요구사항 모델에서 식별된 모든 프로세스가 아키텍쳐 모델에서 정의한 모듈에 빠짐없이 할당되었는지를 나타내는 표이다. 이 요구사항 추적표를 통하여 할당의 적합성을 검사할 수 있다.

5. 하부 모듈의 아키텍팅

확장데이터흐름도를 이용하여 할당한 데이터/제어 흐름은 해당 아키텍쳐 모듈의 데이터/제어 흐름도가 된다.

그림 16은 제어기에 할당된 데이터/제어 흐름을 보여주고 있다. 이렇게 할당된 데이터/제어 흐름은 제어기의 세부 설계의 기초가 된다. 그림 16의 외부 입·출력을 유지하며 내부적인 세부 프로세스를 추가 시키고, 이를 확장데이터흐름도로 만들면, 제어기의 하부 모듈에 프로세스를 할당할 수 있게 된다. 여기에 제어기의 아키텍쳐 흐름도와 아키텍쳐 연결도를 작성하면 제어기에 대한 세부 아키텍팅이 완성된다. 이렇게 하부 수준으로 상세히 모델링을 수행하게 되면 프로세스의 입력과 출력이 명확해 지는 수준에 도달하게 된다. 이 수준에서 프로세스에 대한 규격을 작성하게 된다. 프로세스 규격은 해당 프로세스가 입력을 어떻게 출력으로 변환시키는가를 텍스트나 그림, 수식 등 적당한 방법을 사용하여 기술하면 된다. 따라서 모든 프로세스는 하부 프로세스로 분해 되거나 프로세스 규격을 가져야 한다.

Req. Component	Architecture Component				
**********	Teach Pendant	Power Convertor	Controller	Body	Welding Subsystem
Get_User_Entries	X				
Echo_User_Entries	X				
Display_Robot_Response	X				
Receive_Power		X			
Job_Program_Manipulation			X		
Run_Job_Program			X		
Booting			X		
Enhanced_Arc_Wedling.c			X		
JOB_PROGRAM			X		
Robot_Manipulating				X	
Control_Welding_Wire,_Current_and_GAs					X

그림 15.요구사항 추적표.

Fig. 15. Requirement traceability matrix.

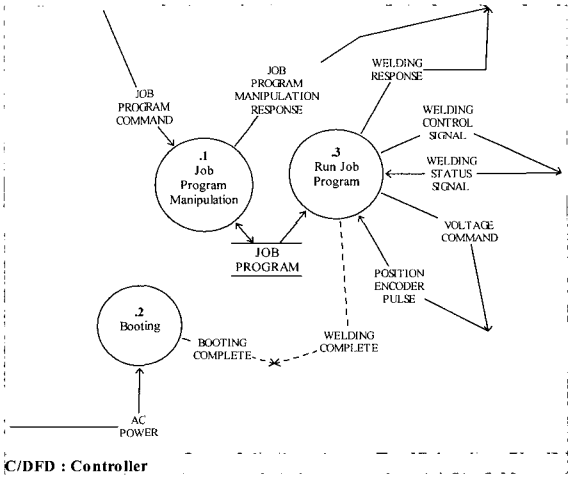


그림 16. 제어기의 데이터/제어 흐름도.

Fig. 16. D/CFD of controller.

결국 시스템의 최하위 수준이라는 것은 프로세스를 명확하고도 간단하게 기술할 수 있는 수준이라고 할 수 있다. 이 단계에서의 결과물 즉, 프로세스 규격과 인터페이스 규격, 아키텍처 모듈 규격이 결국은 최종적인 결과물이 된다. 아키텍처 모듈 규격에는 할당된 프로세스와 모듈에 대한 설명, 입·출력, 고려했던 대안 등이 기술되게 된다.

V. 결론

본 연구에서는 복잡한 실시간 시스템의 아키텍팅 및 설계 방법인 PSARE 방법을 이용하여 용접로봇의 상부아키텍처를 정의한 결과를 제시하였다. PSARE 방법을 이용하여 시스템 아키텍처를 설계할 경우 다음과 같은 장점이 있다.

- 시스템의 기능 요구사항과 물리적 아키텍처의 계층구조를 명확히 볼 수 있다.
- 확장데이터흐름도를 통하여 요구사항과 물리적 아키텍처를 동시에 볼 수 있다.
- 아키텍처 템플릿을 통하여 기술발전에 따른 인터페이스의 재설계시 시스템의 핵심 프로세스를 재사용할 수 있다.
- 문제 영역인 요구사항 모델과 해법 영역인 아키텍처 모델을 명확히 분리하여 다양한 해법에 대한 가능성을 고려할 수 있게 해준다.

PSARE 방법은 단지 용접로봇에만 적용될 수 있는 방법이 아니라 컴퓨터 또는 마이크로 프로세서, 임베디드 실시간 시스템의 아키텍팅과 시스템설계 규격 모델링에 매우 유용한 방법이다. 점차 지능화되어가는 로봇의 설계에 있어서 본 연구는 체계적인 아키텍팅 또는 시스템설계에 대한 구체적인 방법을 사례를 통해 보여준 것이 의미 있는 일이다.

본 사례를 통해서 모델기반 시스템설계 방법이 개발자들을 포함한 수명주기 상의 모든 이해당사자들 간의 의사소통을 개선하고, 설계 산출물의 모호성을 제거하며, 설계오류에 의한 품질문제를 예방하고, 보다 완전한 설계지식을 표현하고 수거 가능하게 한다는 교훈을 얻을 수 있었다. 지능형 로봇 시스템과 같은 복잡한 학제 복합형 시스템을 성공적으로 개발하기 위해서는 부품 또는 단품 개발과 같은 단순제품 개발 문화에서 탈피하여 선진 개발문화의 정착이 선결되어야 할 것이다.

참고문헌

[1] E. Rectin et. al., *The Art of Systems Architecting,* CRC Press, New York, 1997.

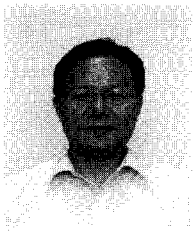
[2] D. Hatley, et. al., *Process for System Architecture and Requirements Engineering,* Dorset House Publishing, New York, 2000.

[3] D. Hatley, et. al., *Strategies for Real-Time Specification,* Dorset House Publishing, New York, 1988.

[4] Miller, A. Geroge, "The magical number seven, plus or minus two.", *The Psychological Review*, vol. 63, no. 2, 1596.

[5] 유일상 외, "견실한 시스템 아키텍처 개발 지침", *Proceeding of the KACC,* October 1999.

[6] S. A. Friedental et. al., "Extending UML™ from software to system", *INCOSE 2003 Symposium Proceedings.*



김 진 일

1985년 경희대 원자력공학과 졸업. 한국과학기술원 원자력공학과 석사(1995). 1999년~현재 아주대학교 시스템공학과 박사과정 재학중. 관심분야는 시스템 아키텍팅, 시스템 모델링 언어, Requirement Engineering.

박 영 원

1968년 한양대 전자공학과 졸업. Oklahoma State University 제어계측시스템공학 석사(1973). 동대학원 박사(1976). 1976~1996 맥도넬 더글러스 우주항공회사 선임수석연구원. 1996년~1998 고등기술연구원 연구위원. 1998~현재 아주대학교 시스템공학과 교수. 관심분야는 첨단시스템아키텍팅, DFSS 설계 및 해석기술.