CMIML 생성을 위한 VI Wizard의 설계 및 구현

유대승° 심민석 박성규 김종환 이명재
울산대학교 컴퓨터정보통신공학부
{ooseyds°, sms, icoddy bearknight, ymj}@mail.ulsan.ac.kr

The Design and Implementation of VI Wizard for Generating CMIML

Daesung Yoo° Minsuck Sim Sunghue Park Jonghwan Kim Myeongjae Yi
School of Computer Engineering & Information Technology, University of Ulsan

요 약

과거 우리는 장비에 대한 제어 및 모니터링 소프트웨어의 효율적인 생성을 위한 프레임웍[1][2]을 제안하였다. 제안한 프레임웍은 세가지의 XML 문서(IID, MAP, CMIML), VI Wizard, Generator로 구성되었다. 본 논문에서는 제안한 프레임웍의 핵심이라고 할 수 있는 VI Wizard에 대한 설계와 구현을 보인다. VI Wizard는 장비의 인터페이스 정보를 기술하고 있는 IID(Instrument Interface Description) 문서를 이용해서 GUI 기반으로 VI(Virtual Instrument)를 구성하고, 새롭게 구성된 VI를 이용하여 장비의 제어정보, 사용자 인터페이스 정보, 모니터링 정보, 통신 정보, 스케줄 정보 등을 기술하는 CMIML(Control & Monitoring Instrument Markup Language)문서를 생성한다.

1. 서 론

산업의 발전과 그에 따른 작업 환경의 변화는 점점 생산 현장에서 사용되는 장비들을 자동화시켜갔고 이런 자동화 장비들은 필드버스, Ethernet과 같은 산업용 네트워크를 통해서 서로 연결이 되어 크고 복잡한 설비를 이루게 된다. 이렇게 장비들을 직접 제어하기보다는 PC 기반으로 제어하게 됨에 따라 이런 자동화 장비들의 제어 및 모니터링을 위한 소프트웨어의 중요성이 점차 증대되고 있다.

그러나 다양한 네트워크를 통해서 연결된 많은 자동화 장비들의 제어와 모니터링을 위한 소프트웨어의 개발에는 몇 가지 문제가 존재한다. 자동화 장비의 제조사 별로 제공하는 드라이버 API가 다르기 때문에 복잡한 설비를 이루고 있는 많은 장비들 각각에 대해서 소프트웨어를 개발해야 하고, 만약 장비의 드라이버API가 변경된다면 소프트웨어를 다시 개발해야 한다는 어려움이 있다. 그리고 장비들을 연결하는 네트워크와 장비가 지원하는 플랫폼이 다양하기 때문에 소프트웨어가 다양한 네트워크와 플랫폼을 지원하도록 개발되어야 하는 문제가 있다. 이런 문제들로 인해서 자동화 장비들의 제어와 모니터링을 위한 소프트웨어의 개발과 유지보수에 많은 비용이 들어가고 있다.

이에 우리는 이런 문제들을 해결하기 위해서 효율적인 제어 및 모니터링 소프트웨어의 개발을 지원하는 프레임웍을 제안하였다[1][2]. 제안한 프로임웍은 세가지의 XML 문서(IID, MAP, CMIML)와 두가지의 도구들(VI Wizard, Generator)로 구성되었다. VI Wizard는 장비에 대한 인터페이스 정보를 기술한IID를 이용하여 GUI기반으로 VI를 구성하고, 새롭게 구성된 VI를 CMIML로 변환하는 도구이다. VI Wizard를 통하여 생성된 CMIML은 Code Generator에 의해서 자동으로 소스코드로 변환되어 운용될 것이다.

본 논문에서는 우리가 제안한 프레임웍의 핵심이라고 할 수 있는 VI Wizard에 대한 설계와 구현을 소개한다.

본 논문의 구성은 다음과 같다. 2장에서는 관련연구와 연구동향에 대해서 살펴보고, 3장에서는 VI Wizard에 대하여 세부적으로 설명한다. 4장에서는 CMIML 생성과정에 대하여 설명하고, 5장에서는 결론 및 향후 연구 과제에 대해서 살펴보도록 한다.

2. 관련연구

기존의 산업 현장에서 사용되던 장비나 설비는 내부에 제어를 위한 프로세서를 탑재하고 있었고 사용자가 직접 조작을 하여 장비나 설비를 가동시켰다. 이러한 기존의 장비나 설비는 상당히 고가였고, 유지보수에도 상당한 비용을 소모하게 되었다. 따라서 산업체들은 비용의 절감을 위해 장비나 설비의 자동화에 관심을 갖게 되었고, 자동화 장비의 구현이나 장비들을 연결하는 프로토콜의 구현에 대한 연구가 진행되어 왔다. 90년대 초 PC기반의 자동제어 시스템[3]에 대한 연구가 시작되었고, 90년대 후반, 2000년에 이르러 본격적으로 PC기반의 제어 시스템의 개발에 대한 연구[4][5][6]가 이루어지기 시작했다. 그러나 산업 현장에서 사용되어지는 장비들의 자동화에 대한 관심이 커지고 많은 장비들이 자동화되어져 감에도 불구하고 국내에서는 이런 자동화 장비들의 제어와 모니터링을 위한 소프트웨어의 개발에 대한 연구가 거의 전무한 상황이다.

국내에 비해서 국외에서는 이미 많은 연구가 진행되어 오고 있고, 이미 상용화 되어있는 관련 소프트웨어들이 출시되어 있

본 연구는 한국과학재단 지정 울산대학교 네트워크 기반 자동화연구센터의 지원에 의해 이루어졌습니다.

다. 최근에는 NASA에서 천체 관측에 사용되어지는 장비들의 제어를 위한 새로운 Markup언어를 제안하는 등의 장비나 설비의 인터페이스 또는 산업용 네트워크의 프로토콜인 필드버스를 XML로 표현하는 방법들이 제안되고 있다.

상용화되어 있는 소프트웨어 중 대표적인 것에는 NI(National Instrument)사의 LabVIEW[7]와 Rockwell소프트의 RsVIEW[8]가 있다. 이 두 소프트웨어는 기본적으로 자동화 장비의 제어와 모니터링을 위한 소프트웨어를 생성하는 기능을 수행하며, 개발자가 직접 소프트웨어의 코드를 작성하는 것이 아니라 GUI 기반의 VI(Virtual Instrument) 편집 환경을 제공함으로써 쉽고 빠른 개발과 유지보수를 지원하고 있다.

그러나 이런 소프트웨어들은 장비 제조업체에서 제공하는 API를 사용하여야 하고 장비의 API가 변경되면 변경된 API를 다시 제공받아야 하는 문제가 있다. 그래서 장비의 API를 XML 문서로 기술함으로써 이런 문제들을 해결하고자 하는 노력이 있었다. 대표적으로 NASA의 IML(Instrument Markup Language)[9]과 AIML(Astronomical Instrument Markup Language)[10]이 있다. 천체 관측에 사용하는 장비들의 인터페이스를 XML 문서로 기술하고, 이 정보와 API의 정보를 연결시킴으로써 API의 변경이 제어 소프트웨어에 끼치는 영향을 최소화시키고 쉽고 빠른 개발을 지원하도록 한다.

근래에는 산업용 네트워크의 프로토콜 중 가장 각광을 받고 있는 필드버스에 대한 연구가 가장 활발하게 이루어지고 있다. 대표적인 것으로 필드버스의 표준 프로토콜 중 하나인 CAN(Controller Area Network)에 대한 XML 어플리케이션인 CoML(CANopen Markup Language)[11]이 있고, 이 CoML을 이용해서 구현한 CANINSIGHT 시스템[12]이 있다. 그러나 필드버스는 독립적으로 구성된 서로 다른 시스템들 사이에 유연성이나 호환성이 많이 결여되어 있다. 그래서 이런 문제들을 해결하기 위한 연구들이 진행되고 있으며, 대표적으로 Tag-based Trees를 이용해서 서로 다른 필드버스들을 상호 연동시키는 방법에 대한 연구[13]와 필드버스 내의 장비와 시스템 사이의 통신 인터페이스의 표준화에 대한 연구[14]가 진행 중에 있다.

위와 같은 국내외 연구에서 보듯이 자동화 장비의 드라이버 API에 의존적이지 않으면서, GUI 환경에서 쉽고 빠르게 제어 소프트웨어를 개발할 수 있도록 하는 환경이 필요하다. 따라서 이런 환경을 제공하기 위해서 제안했던 프레임웍의 VI Wizard를 구현한다.

3. VI Wizard

VI Wizard는 장비에 대한 인터페이스 정보를 기술한 IID파일을 이용해서 GUI 기반의 VI 편집 환경을 제공하고, 새롭게 구성된 VI를 CMIML로 변환하는 도구이다. 생성된 CMIML은 Code Generator에 의해 제어 소프트웨어로 생성되어 운용되거나 다른 VI 구성을 위해 재사용될 수 있다.

3.1 VI Wizard System Architecture

그림 1의 (a)는 [1][2]에서 제안한 전체 프레임웍이며 (b)는 VI Wizard의 세부적인 시스템 구조이다.

그림 1의 (b)와 같이 VI Wizard는 3개의 모듈, 3개의 매니저와 2개의 데이터저장소로 구성된다.

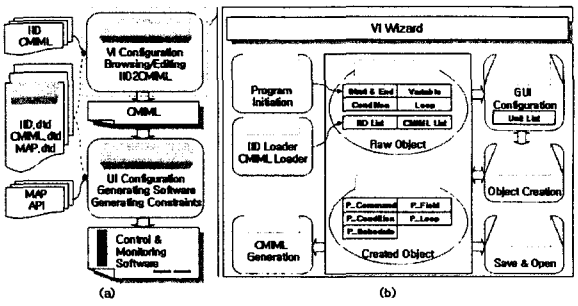

그림 1 VI Wizard의 시스템 구조

3개의 모듈은 다음과 같다.

• Initiator Module

Vi Wizard의 실행에 필요한 초기화 작업을 수행하는 모듈로써 처음 시작 시 실행되며, 주요 기능으로는 Object Pool에 Start, End, Variable, Condition, Loop 요소들을 등록하고 그림 2의 (a)의 Control 그룹에 각 요소들을 추가한다.

• Loader Module

IID Loader와 CMIML Loader로 구성된다. IID Loader는 IID파일을 읽어서 Field 요소와 Method 요소들을 추출하여 Object Pool의 IID_List에 등록하고 그림 2의 (a)에 IID의 이름으로 새로운 그룹을 추가하고 새롭게 추가된 그룹에 읽어 들인 Method 요소들을 추가한다. CMIML Loader는 CMIML파일을 읽어서 Schedule 요소들을 추출하여 Object Pool의 CMIML_List에 등록하고 그림 2의 (a)의 User Custom 그룹에 Schedule 요소들을 추가한다.

• Generator Module

Object Pool과 Instant Pool의 데이터를 추출하여 CMIML을 생성한다.

3개의 매니저는 다음과 같다.

• Editor Manager

GUI 편집 환경에서 VI에 대한 구성을 책임지는 관리자로 Object Manager를 통해서 Instant Pool에 새로운 객체를 생성하거나 삭제하는 기능을 수행한다.

• Object Manager

Editor Manager의 요청에 의해 Object Pool의 객체를 실제 GUI 객체에 대한 인스턴스를 생성하여 Instant Pool에 추가하거나 삭제하는 기능을 수행한다.

• Project Manager

프로젝트 기능을 책임지는 관리자로 프로젝트 단위로 작업을 관리하며 작업환경을 저장하고 가져오는 기능을 수행한다.

2개의 데이터저장소는 다음과 같다.

• Object Pool

Initiator와 Loader 모듈이 추가한 객체들을 저장하는 객체 저장소로 Editor Manager가 VI를 구성시 필요한 정보를 제공한다.

• Instant Pool

GUI 편집 환경에서 VI를 구성하고 있는 인스턴스화된 GUI 객체들의 정보를 가지는 저장소로 Object Pool의 객체가 Object Manager에 의해 인스턴스화 되면 Instant Pool에 저장되게 된다.

3.2 VI Wizard System Implementation

VI Wizard는 Visual Basic 6.0 환경에서 XML 파서로 MSXML 3.0을 사용하여 구현하였다.

그림 2는 VI Wizard의 실행 화면이다. 그림 2의 (a)는 Initiator와 Loader에 의해 Object Pool에 등록된 객체들을 그룹별로 보여준다. (b)는 선택된 객체들의 속성정보를 보여준다. (c), (d), (e)는 Editor Manger에 의해 GUI 기반으로 편집된 VI를 나타낸다. (d)는 조건문, (e)는 반복문이 수행하는 내부적인 동작을 나타낸다. (f)는 CMIML Generator에 의해 새롭게 구성된 VI를 CMIML로 변환된 문서다.

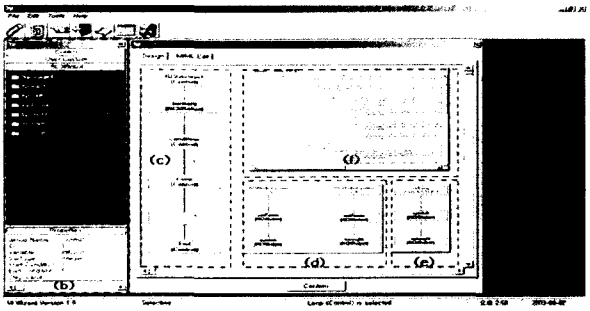

그림 2 VI Wizard의 실행 화면

4. CMIML 생성 과정

그림3에서는 제안했던 프레임웍을 사용하여 제어 및 모니터링 소프트웨어를 생성하는 과정을 보이고 있다. 본 논문에서 구현한 VI Wizard가 수행하는 부분은 점선으로 된 사각형으로 표시되어 있다. 이미 작성되어 있는 IID 문서 또는 CMIML 문서를 입력으로 받아들인다. CMIML 문서의 경우는 장비나 설비의 스케줄 정보만 읽어 들인다. 이렇게 읽어 들인 정보들을 이용해서 GUI 환경에서 새로운 VI를 구성하고 이 VI를 이용해서 CMIML 문서를 자동으로 생성해 낸다. 생성된 CMIML은 Code Generator에 의해서 소스코드로 변환되게 될 것이다.

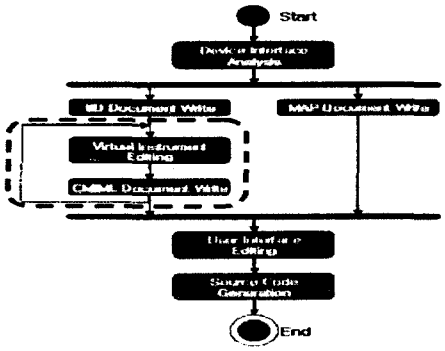

그림 3 제어 및 모니터링 소프트웨어 생성과정

5. 결론 및 향후 연구과제

본 논문에서는 장비의 인터페이스 정보를 기술하고 있는 IID 문서를 이용해서 GUI 기반으로 VI를 구성하고, 새롭게 구성된 VI를 이용하여 장비의 제어정보, 사용자 인터페이스 정보, 모니터링 정보, 통신 정보, 스케줄 정보 등을 기술하는 CMIML 문서를 생성하는 VI Wizard에 대한 설계와 구현을 보였다.

VI Wizard는 Code Generator와 함께 전문적인 지식(장비제어, 제어 소프트웨어, XML)없이도 CMIML문서를 이용해서 자동으로 소프트웨어를 생성함으로써 원격으로 제어되는 생산현장의 여러 자동화 장비들에 적용될 수 있는 제어 및 모니터링 소프트웨어를 기존의 방법에 비해서 적은 비용으로 쉽고 빠르게 개발할 수 있을 것이다.

향후에는 다양한 플랫폼에 적용 가능한 범용적인 Code Generator에 대하여 연구하고 개발하여 우리가 제안한 프레임웍을 완성하고 자동 생성된 소프트웨어의 안정성을 검증할 수 있는 가상 시뮬레이션 등을 통하여 제안한 프레임웍에 대한 완성도를 높이겠다.

본 연구는 한국과학재단 지정 울산대학교 네트워크 기반 자동화연구센터의 지원에 의해 이루어졌습니다.

[참고문헌]

[1] 유대승, 심민석, 박성규, 김종환, 이명재, "제어 및 모니터링 소프트웨어의 효율적인 개발을 위한 프레임워크 설계", 정보과학회춘계학술발표대회, 2003

[2] Dae-sung yoo, Min-suck sim, Sung-ghue park, Jong-hwan kim, *Myeong-Jae yi*, 'A *Framework for automatic generation of instrument* control and monitoring software', Proceedings of the 7th Korea-Russia International Symposium, KORUS 2003, vol. pp, 428-432

[3] 구영재, 이준서, 이인범, 장근수, 'PC를 이용한 자동제어시스템 개발', 한국자동제어학술회의논문집(KACC), 1991

[4] 변승현, 마복렬, '대용량 플랜트 제어를 위한 PC 기반 I/O 인터페이스 시스템 구축에 관한 연구', 한국자동제어학술회의논문집(KACC), 1999

[5] 김정구, 최경현, 홍금식, 'PC에 기반을 둔 개방형 로봇제어시스템:PC-ORC A PC-Based Open Robot Control System:PCORC', 제어.자동화.시스템공학논문지, 2000

[6] 박남준, 김홍석, 박종구, 'PC기반의 생상시스템을 위한 운용소프트웨어 구조', 제어.자동화.시스템공학 논문지, 2001

[7] National Instrument "www.ni.com/"

[8] Rockwell Automation "www.rockwell.com/"

[9] NASA "pioneer.gsfc.nasa.gov/public/iml/"

[10] NASA "pioneer.gsfc.nasa.gov/public/aiml/"

[11] Dieter Buhler, "The CANOpenn Markup Language Representing Fieldbus Data with XML", Industril Electronics Society, 2000. IECON 2000. 26th Annual Conference of the IEEE, Volume: 4, 22-28 Oct. 2000 Page(s): 2449-2454 vol.4

[12] Dieter Buhler, Wolfgang Kuchlin, "Remote Fieldbus System Management with Java and XML", Industrial Electronics, 2000. ISIE 2000. Proceedings of the 2000 IEEE International Symposium on, Volume: 1, 4-8 Dec. 2000 Page(s): 1-6 vol.1

[13] Steffen Deter, 'Fieldbus Device Description using Tag-based *Trees*', *Africon Conference in Africa*, 2002. IEEE AFRICON. 6th, Volume: 1, 2-4 Oct. 2002 page(s): 263-268 vol.1

[14] FDT Joint Interest Group, 'http://www.fdt-jig.org/index2.html'