OPC 기반의 OLE 컨트롤들을 위한 컨테이너의 연구

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

A Study on the Container for OPC-based OLE Controls

Sungkyu Park° Minsuck Sim Daesung Yoo Jonghwan Kim Myeongjae Yi
school of computer & information technologies, University of Ulsan, Korea

요 약

현재 수많은 OPC(OLE Process Control)관련 제품들이 나와 있다. 특히, 클라이언트 제품군 가운데 가장 많이 눈에 띄는 것은 OPC 서버와 통신하여 사용자에게 장비의 정보를 전달하거나 장비를 제어 할 수 있는 OLE(Object Linking and Embedding) 컨트롤(ActiveX 컨트롤)들이다. 하지만, 이러한 OLE 컨트롤을 시스템에 적용하기 위해서는 이를 지원하는 개발 언어와 컴파일러에 대한 프로그래밍적 지식이 필요하다.

이에 본 논문에서는 프로그래밍에 대한 전문적 지식이 없이도 HMI(Human Machine Interface) / SCADA(Supervisory Control and Data Acquisition)와 같은 시스템을 구축할 수 있도록 OLE 컨트롤 컨테이너를 제공함으로써 사용자가 쉽게 디자인할 수 있는 방법을 제시한다. 본 논문에서 제안하고자 하는 OLE 컨트롤 컨테이너는 OLE/COM 기술을 기반으로 하는 OLE 컨트롤을 지원하고 웹으로의 빠른 확장을 위해 설계되었다.

1. 서 론

소프트웨어 산업은 하드웨어 산업에 비해 생산성이 매우 낮다. 이러한 문제를 해결하기 위해 소프트웨어의 조립 생산에 대한 많은 시도가 있었고 1980년대 이후 객체 기술 및 컴포넌트 기술의 등장으로 소프트웨어에서도 조립 생산이 가능하게 되었다. 이와 더불어, 산업현장에서 사용되는 소프트웨어 또한 많은 변화가 있었는데, HMI(Human Machine Interface) / SCADA(Supervisory Control and Data Acquisition)[1] 소프트웨어에서도 이러한 조립식 개념이 도입되고 있다. 이제는 전문적인 프로그래머가 아니더라도 간단한 교육으로 짧은 기간에 산업 자동화 시스템을 구축할 수 있게 되었다. 하지만, 개발사마다 소프트웨어의 구성에 있어 다른 표준을 지원함으로써 시스템 통합 측면에서 불안 요소를 가지고 있다. 이에 본 논문에서는 마이크로소프트사의 OLE(Object Linking and Embedding) / COM(Component Object Model)[2][3][4]을 지원하는 컴포넌트를 통해 통합화를 시도하고자 한다. 이러한 시도는 OPC(OLE Process Control)[6] 및 필드 기기들에 대한 다양한 OLE 컨트롤을 지원함으로써 점점 더 많이 보급되고 있다. 특히, 최근 공장 자동화의 커다란 흐름의 하나인 OPC는 OLE/COM을 보다 효율적으로 지원함으로써 차후 웹의 지원과 다양한 벤더들의 통합을 가능하게 할 수 있었다. OPC 제품군은 크게 클라이언트 제품군, 서버 제품군, 개발 툴과 서비스등 세 가지로 나뉘어 진다.

OPC 서버와 OPC 클라이언트는 COM 인터페이스를 통해 정보를 요청하거나 받을 수 있다. 그리고 OPC 클라이언트 또한 OLE 컨트롤로 만들어질 경우 보다 쉽게 사용자 어플리케이션과 통합될 수 있다. 하지만, 어플리케이션과의 통합 과정에서 마이크로소프트사의 비주얼 베이직이나 비주얼 C++과 같은 컴파일러를 사용할 경우 전문적인 프로그래밍 지식이 필요하고 수정 사항이 발생할 때마다 새로 해야하는 컴파일 과정은 유지보수 비용을 증가시킬 수 있다.

따라서, 본 논문에서는 OPC 클라이언트용 OLE 컨트롤들을 사용자가 런-타임 시에 컨테이너로 통합 가능하게 함으로써 시스템 운영자의 프로그래밍 언어나 컴파일러에 대한 전문적인 지식 없이 프로젝트를 가능하게 할 수 있는 방법을 제시한다.

본 논문은 2장에서 기술적 배경이 되는 HMI/SCADA, OPC, OLE/COM에 대해 알아보고 3장에서 OLE 컨트롤과 컨트롤 컨테이너에 대해 살펴본다. 4장에서는 OPC 클라이언트 컨트롤과 컨트롤 컨테이너의 통합에 대해 설명하고 5장에서 웹으로의 확장, 마지막으로 6장에서는 결론을 맺고 향후 연구 계획에 대해 논한다.

2. 기술적 배경

최근 HMI/SCADA 시스템에서는 OPC 인터페이스의 지원을 통하여 단일화된 모니터링 및 제어 방법을 시도하고 있는 추세이다. OPC는 산업 자동화에서 각 회사별로 제공하던 드라이버를 표준화된 OLE/COM 기술을 도입하여 해결하려 한다. 이번 장에서는 이러한 기술적 배경에 대해 알아본다.

2.1 HMI(Human Machine Interface) / SCADA(Supervisory Control and Data Acquisition)

HMI란 운영자와 기계사이의 인터페이스로 시스템 운용자의 조작 의도를 기기에 전달하고, 기기의 운전 데이터를 운용자가 인식할 수 있게 하는 소프트웨어 및 하드웨어를 통칭해서 부르는 용어이다.[1] HMI 범주에는 소프트웨어와 Panel PC, HMI 기기등이 속한다. 산업 현장이 자동화되고, 통신 기술이 도입되면서 대형 시스템의 조작은 통신을 이용하여 원격에 위치한 중앙 제어실 등에서 HMI Software를 이용하여 조작하는 추세에 있다. SCADA의 경우는 EMS(Energy management system), DMS(Distribution management system), SYSCON(System control of communication network)[1]등에서 유용하게 사용되고 있다.

2.2 OPC(OLE Process Control)

본 연구는 한국과학재단 지정 울산대학교 네트워크 기반 자동화 연구센터의 지원에 의해 이루어졌습니다.

OPC의 개발은 WinSEM(Windows for Science, Engineering , and Manufacturing)으로 잘 알려진 마이크로소프트 인더스트리 포커스 그룹으로부터 시작되었다. OPC 표준은 프로세스 데이터의 클라이언트 어플리케이션들과 서버들 사이의 인터페이스 방식을 규정한 것이다. OPC 이전에는 특정 장비를 위해 특정한 소프트웨어만을 사용했어야 했다. 하지만, OPC의 경우 어떠한 OPC 호환 클라이언트 어플리케이션이든지 OPC호환 서버를 적용한 컨트롤 장비에 쉽게 인터페이스 할 수 있다.

OPC의 사양은 OLE/COM 기술을 사용하기 때문에 COM의 다양한 장점들을 그대로 가져올 수 있다. COM의 자세한 내용은 다음 절에서 언급한다.

2.3 OLE(Object Linking and Embedding) / COM(Component Object Model)

COM은 컴포넌트 소프트웨어의 개발을 가능하게 하는데, 이로 인해 어플리케이션 개발자, 벤더들, 앤드-유저, 그리고 관련 회사들에게 중요한 영향을 미친다. COM의 대표적인 장점으로는 다음과 같은 것들이 있다.

-언어 독립성(Language Independence). COM은 바이너리 표준으로서 어떠한 언어나 컴파일러에 대해서도 호환성을 가지고 있다.

-상호 운영성(Interoperability). 서로 다른 소프트웨어 벤더들로부터 만들어진 컴포넌트들 역시 상호 운영될 수 있다.

-위치 투명성(Location Transparency). COM 과 DCOM은 같은 프로세스 안에서, 또는 다른 프로세스 안에서, 심지어는 다른 컴퓨터에서도 클라이언트 입장에서는 똑같은 방식으로 통신할 수 있다.

-강력한 버전관리(Robust Versioning). COM은 서버 컴포넌트의 업그레이드에 대해 유연하게 대처할 수 있다. 새로운 컴포넌트가 시스템에 설치되더라도 클라이언트에서 발생하게 되는 오류를 막을 방법을 제시하고 있다.

-보안성(Security). COM 과 DCOM은 몇 가지의 클라이언트들의 인증에 관한 보안형태를 제공한다.

이러한 장점들로 인해 필드기기들의 통합과 실시간 모니터링 및 제어를 위해 OLE/COM 기술이 도입되고 있다.[8][9][10][11] HMI 툴에서도 이러한 필드기기들과 인터페이스된 OLE 컨트롤들의 지원은 필수 조건이 되어 가고 있다. 또한, 웹을 통한 모니터링과 제어를 가능하게 함으로써 보다 사용자 편의성을 증대시키고 있다.

3. OLE 컨트롤과 OLE 컨트롤 컨테이너

이 장에서는 컴포넌트 기반의 솔루션을 개발하기 위해 신뢰성 있고 상호운영이 가능하고, 최종적으로 컴포넌트들의 사용을 위해 OLE 컨트롤과 컨트롤 컨테이너를 설계한다. 여기서 언급되는 컨트롤 컨테이너는 일반 OLE 컨테이너와는 구분된다. 컨트롤 컨테이너는 OLE 컨테이너에 몇 가지 추가적인 기능을 포함하고 있다. OLE 컨테이너로 구현된 응용 프로그램은 자체 문서에 포함 항목이나 연결 항목을 통합할 수 있는 응용 프로그램이다. 컨테이너 응용 프로그램에서 관리되는 문서는 응용 프로그램 자체에서 작성된 데이터뿐만 아니라 OLE 복합 문서 구성 요소를 저장하고 표시할 수 있다. 또한 컨테이너 응용 프로그램에서 새 항목을 삽입하거나 기존 항목을 편집할 수 있는 기능이 있다.[5]

3.1 OLE 컨트롤

OLE 컨트롤은 OLE 컨트롤의 특징을 가지기 위해 IOleObject 등의 인터페이스들을 지원해야 한다. 또한 대부분의 컨트롤들은 속성들을 가지고 있어야 한다. 하지만, 이러한 속성들은 노출되는 것을 요구받지 않고 있다. 이러한 문제를 해결하기 위해 속성 페이지를 제공함으로써 이러한 속성 값들을 바꿀 수 있는 방법을 제공한다.

다음 [그림 1]에서는 OLE 컨트롤들이 지원해야 할 인터페이스들을 도식화하여 보여준다.

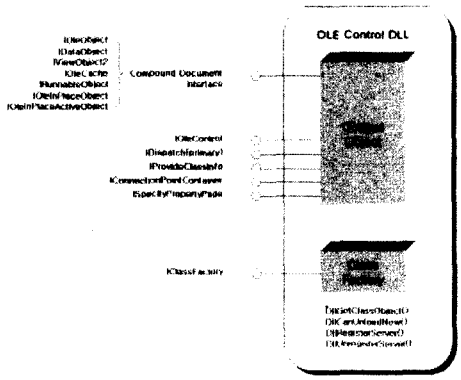

그림 1 OLE Control DLL

3.2 OLE 컨트롤 컨테이너

OLE 컨트롤 컨테이너는 OLE 컨테이너와 구분되기 위해 다음의 것들을 추가로 제공해 주어야 한다.[5]

1. 인-프로세스 서버로부터 객체들의 포함(Embedded objects from in-process servers)
2. 인-플래이스 활성화(In-Place activation)
3. 인사이드-아웃 활성화(Inside-out activation)
4. 시각 활성화(OLEMISC_ACTIVATEWHENVISIBLE)

다음 [그림 2]에서는 컨테이너에서 OLE 컨트롤들을 위해 구현해야할 인터페이스들과 OLE 컨테이너 프레임을 보여준다. 이 OLE 컨트롤 컨테이너를 구현하기 위해서는 OLE 컨트롤들을 보이게 하거나 활성화시킬 수 있도록 IOleClientSite등의 인터페이스를 지원해주어야 한다.

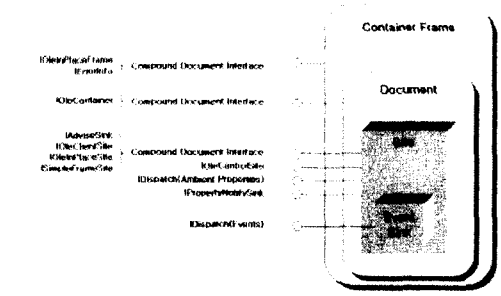

그림 2 OLE Container Frame

이밖에 OLE 컨트롤 컨테이너는 추가적인 메서드들과 상태 비트들도 제공해야 한다.

4. OPC 클라이언트 컨트롤과 컨트롤 컨테이너의 통합

OPC 클라이언트 제품군 가운데 가장 눈에 많이 띄는 것은 다른 개발자들을 위한 OLE 컨트롤(ActiveX 컨트롤)이다. 이러한 컨트롤을 지원하고 프로그래밍적 지식이 없이도 시스템 구축이 가능하도록 하기 위해 본 논문에서 제시하는 OLE 컨테이너는 필수이다.

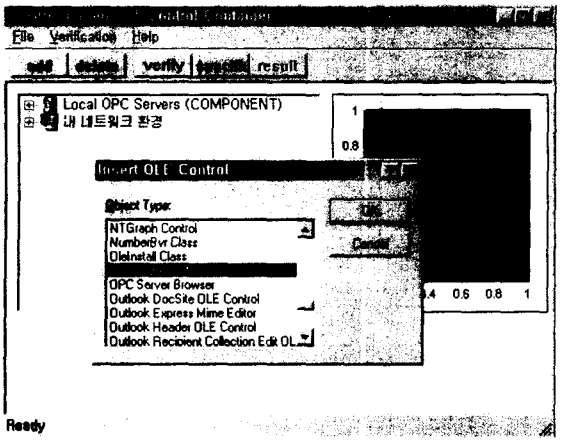

그림 3 OLE 컨트롤들을 추가하는 과정

대부분의 OPC 클라이언트 컨트롤들은 센서에서 수집한 데이터를 효과적으로 사용자에게 보여주거나 값을 변경하는데 사용된다. 따라서, 특정 OPC 서버와 접속하고 특정 아이템(또는 태그)과 연결작업이 필수적이다.

OPC 클라이언트 컨트롤의 경우 속성 페이지에서 기본적으로 OPC 서버를 설정할 수 있다. 다음에 해주어야 할 작업은 그 서버에 포함된 특정 아이템과의 연결하는 것이다. 태그라고도 불리우는 이 아이템에서 실제 값들을 받아오게 되는데, 대부분의 컨트롤의 경우 특정 값에서의 이벤트를 설정할 수 있는 기능들이 있다.

본 논문에서 연구하는 컨테이너의 경우 OPC 기반의 OLE 컨트롤들을 보다 잘 지원하기 위해 서버와의 접속을 쉽게 해줄 수 있는 OPC Browser 기능을 추가하였다. 이밖에 등록된 컨트롤들은 툴박스로 제공되어 사용자의 마우스 드래그&드랍으로 시스템을 디자인 할 수 있도록 도와준다.

5. 웹으로의 확장

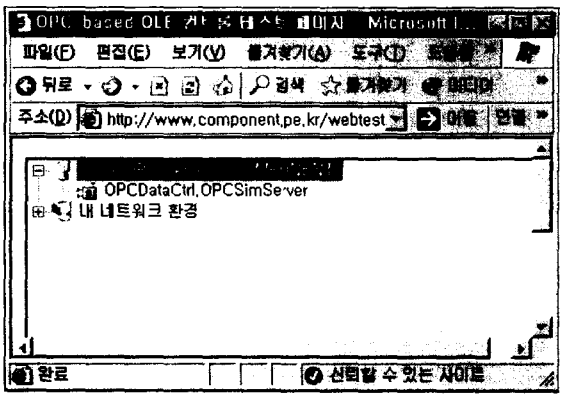

그림 4 Web에서의 실행 모습

최근 웹이 활성화되면서부터 이러한 HMI/SCADA 시스템들은 웹을 지원하지 않고서는 살아 남기 힘든 상황이 되어 버렸다. 따라서, 본 논문에서 제시하고 있는 컨테이너 역시 웹을 지원하기 위해 자동 배포파일을 만들어 주고 html 문서를 만들어주는 기능을 포함한다. [그림 4]에서는 OLE 컨트롤을 웹을 통해 본 모습을 보여주고 있다.

6. 결론 및 향후 연구과제

OPC가 도입됨으로써 장비 벤더는 클라이언트 소프트웨어에 영향을 주지 않고 여러가지 기능들을 수정 및 변경할 수 있게 되었다. 클라이언트 벤더들은 각종 장비들에 대한 인터페이스 드라이버의 라이브러리를 유지보수하기 위해 기울였던 지금까지의 노력 대신에 제품 자체의 개발에 더 많은 자원을 투입할 수 있게 되었다. 이것을 가능하게 해준 것이 바로 OLE/COM의 컴포넌트 기술이다.

본 논문에서 이러한 OLE/COM 기술을 바탕으로 OPC 기반의 OLE 컨트롤들을 위한 컨테이너를 제시하면서 기본 기능들을 테스트하였다. 하지만, 필드에서 직접 사용하기에는 부족한 기능이 많은 것도 사실이다. 차후 연구에서는 OPC를 지원하는 다양한 클라이언트용 OLE 컨트롤들을 추가해야할 뿐만 아니라, 프로젝트 관리 기능이 필요하다. 이 밖에, 단순 모니터링 기반의 컨테이너 역할이 아닌 제어 흐름을 쉽게 사용자가 디자인할 수 있도록 기능을 제공해야 한다. 마지막으로 최근의 HMI/SCADA 소프트웨어의 추세는 웹을 지원하는 것이다. OLE의 표준 인터페이스를 제공함으로써 웹으로의 확장이 용이한 장점을 가지고 있으므로 자동으로 웹에서 배포를 위한 html 문서를 만들어주는 기능 역시 필수요건이다.

[참고 문헌]

[1] Ghosh, S.K. "Changing role of SCADA in manufacturing plant" 1996 IEEE , Volume: 3 , 6-10 Oct. 1996 Pages:1565 - 1566 vol.3

[2] D. Box. *Essential COM*. Addison Wesley, 1998

[3] K. Brockschmidt. *Inside OLE*. Microsoft Press, 1995.

[4] Microsoft. *http://www.microsoft.com/com/*.

[5] Microsoft. MSDN *http://msdn.microsoft.com/*

[6] OPC Fundation. *OLE for Process Control Standard. http://www.opcfoundation.org/*

[7] OPC Council. Japan. "ABC's OPC Applications". Chapter 1. Nikkan Kogyo Shimbun. 2001.

[8] T.-W. Kuo and A. K. Mok. Load adjustment in adaptive real-time systems. In *IEEE Real-Time Systems Symposium*, December 1991.

[9] Microsoft. Dcom technical overview. In *Micorsoft Professional Developers Conference*, 1997.

[10] D. Schmidt, R. Bector, D. Levine, S. Mungee, and G. Parulkar. An orb endsystem architecture for statically scheduled real-time applications. In *IEEE Workshop on Middleware for Distributed Real-Time Systems and Services*, pages 52 60, December 1997.

[11] A. Wolfe. Windows turned on to real-time tasks. *Electronic Engineering Times*, February 1998.