장비 제어 로직들의 분산 배치 및 통합에 관한 연구

심민석[0], 유대승, 박성규, 김종환, 이명재
울산대학교 컴퓨터정보통신공학부
{sms, yds, icoddy, bearknight, ymj}@mail.ulsan.ac.kr

A Study for Distributing Deployment and Integration about Instrument's Control Logic

Minsuck Sim[0] Daesung Yoo Sunghue Park Jonghwan Kim Myeongjae Yi
School of Computer Engineering & Information Technology, University of Ulsan

요 약

본 논문은 OPC 기반 장비의 제어 소프트웨어 생성 시스템(iMaker)의 중앙 집중식 제어 모델의 취약점(제어 컴퓨터의 오류시 모든 제어 중지)을 해결하고 성능 향상을 위하여 제어 소프트웨어를 구성하는 제어 로직들의 분산 배치 및 통합 방법을 사용하여 소기 목적을 달성하고자 한다. 우리가 제안하는 분산 배치 통합 모델은 제어 소프트웨어 실행 시에 발생할 수 있는 오류(시스템 에러, 네트웍 오류)를 극복하는 구조로 디자인하였기 때문에 PC 기반의 제어 시스템에 대하여 안전성을 보장하는 방법을 제공한다.

1. 서론

장비 제어 시스템은 개인용 컴퓨터(PC) 관련 기술의 급격한 발달에 따른 풍부한 양질의 서비스와 값 싼 비용 문제 때문에 PLC와 같은 산업 전용 제어 기술에서 PC 기반의 장비 제어 기술으로 빠르게 변하고 있다. 이러한 흐름에 제어 시스템들은 PC 기반의 장비 제어 기술을 쉽고 편리하게 수렴할 수 있는 개방구조(open architecture)로 진화를 시도하고 있다. 특히 XML (Extensible Markup Language)과 OPC(OLE for Process Control) 기술을 사용하여 복합장비를 쉽게 제어하고 레거시 시스템들과의 통합을 하는 방향으로 흘러가고 있다.[1][2]

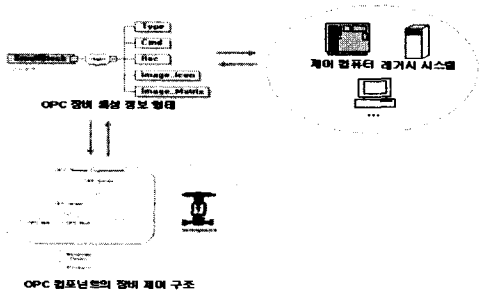

그림 1. OPC와 XML기술을 사용한 장비 제어

PC 기반 장비 제어 기술은 PC기반의 검증된 고급 기술들을 사용함으로써 제어 소프트웨어 개발 및 유지 보수 비용을 기존 PLC 기반에 비해 상대적으로 값 싸고 양질의 서비스를 제공한다. 특히 위의 [그림 1]과 같이 XML과 OPC 기술은 복합 장비 제어 시스템 또는 레거시 시스템들과의 통합과 유지보수에 필수요소로 자리잡아 가고 있다.[3]

그러나 많은 이점을 가지는 PC 기반의 제어 기술이 현장에 적용하고 사용함에 있어서 PC의 안정성 문제가 종종 제기 되고 있는 상황이다. 특히 화학 공단이나 제철 공단등과 같이 안정성을 최우선하는 영역에 대해서는 더욱더 민감하다. 따라서 본 논문에서는 제어 시스템을 구성하는 제어 로직들의 분산 배치 및 통합을 통하여 PC 기반 제어 시스템에 안정성을 보증하는 방법을 부여한다.

2. 관련연구

인터넷 및 네트워크 환경과 흩어져 있는 컴퓨터를 클러스팅하여 보다 나은 성능을 발휘하려는 여러 연구가 국내외에서 이루어져 왔다.

전통적으로 사용되는 Parallel Virtual Machine(PVM)과 Message Passing Interface(MPI)는 워격지 주소의 호스트에 존재하는 프로세스간에 서로 메시지를 전달하는 기본적인 근간을 마련 하였다.[4] JavaParty[5]는 원격객체를 구현할 때 발생하는 코딩 오버헤드를 줄이기 위해 임의의 호스트에서 생성된 객체를 여러 호스트에 자동으로 분산

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

시키는 원격 객체에 대한 메커니즘을 제시 하였다. ATLAS[6]는 멀티쓰레드로 작성된 프로그램 기술을 네트워크 기반으로 확장하여 병렬수행 할 수 있는 설계 기법 사용하여 웹 컴퓨팅의 자원을 활용방법을 제시하였으며 또한 JVM인터프리터를 수정한 자바 병렬 클래스 라이브러리 (Java Parallel Class Library)를 구축통해서 웹 컴퓨팅의 자원을 활용할 수 있는 연구도 있었다.

본 논문에서 제안하는 분산 개념을 적용한 장비제어 관한 적용 사례 및 연구는 빈약하며 특히 분산 개념의 응용 사례는 찾아보기 힘들다.

3. 분산 배치 및 통합 모델

본 논문에서 제시하는 분산 배치 모델은 [그림 2]와 같이 제어 소프트웨어 생성 도구(iMaker-Instrument's Control Logic Maker), 제어 로직으로 구성된 제어 소프트웨어(XBML), 확장 블록 저장소(XBML Repository), 그리고 장비 제어 서비스(ICB-Instrument's Control Bridge)로 디자인 하였다.

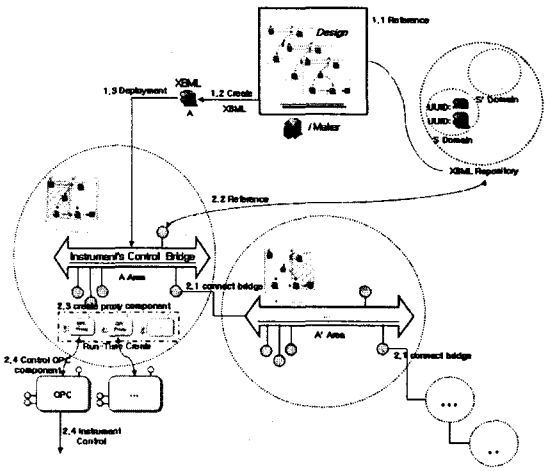

그림 2. 분산 배치 모델

분산 배치 모델은 제어 소프트웨어 생성 도구(iMaker)를 사용하여 OPC 컴포넌트(장비와 결합되어 제어를 담당하는 COM 컴포넌트)의 인터페이스로부터 장비 제어 구조(Data Access Structure) 정보를 추출하고, XML 기술을 사용하여 블록(본 논문에서는 확장 블록이라 명함)으로 모델링한다. 또한 재사용 가능한 여러 블록들을 스케줄링하여 제어 소프트웨어를 생성한다. 이러한 과정을 통하여 생성된 제어소프트웨어는 ICB에 배치된다. 장비 제어 서비스(ICB) 중 Bridge Naming 서비스에 등록 되어 있는 다른 ICB에 복제되어 저장된다. ICB 서비스는 디자인 시 제어 소프트웨어를 구성하는 제어 로직들의 배치 정책(Deployment Policy)에 의해 확장 블록 저장소(XBML Repository)로부터 장비에 결합된 OPC컴포넌트와 결합 가능한 OPC 프락시(Proxy)를 재구성하고 OPC 컴포넌트와 결합을 한다. 이러한 과정을 통하여 ICB 서비스들은 확장 블록의 제어 로직을 읽어서 각자에 위임된 제어 루틴을 실행한다. 또한 타이밍 및 동기화 서비스를 사용하여 분산 환경의 여러 서비스가 하나의 서비스처럼 인식되고, 특정 서비스 오류(시스템 및 네트웍 오류) 발생시 오류 서비스에 위임된 제어 로직의 상태 정보(Transaction Service)에 따라서 다른 서비스에 위임되어 자동 오류 극복(fault torance)할 수 있다.

3.1 iMaker 시스템

iMaker 시스템은 확장 블록(eXtension Block)을 생성 및 스케줄링하고 배치 정책을 지원하는 도구이다. 시스템의 구조는 장비의 특성 정보를 추출하여 블록을 생성 및 동작 시키는 부분(Visual Instrument eXtension Maker), 확장 블록을 보관하고 관리 하는 부분(eXtension Block Repository) 그리고 제어 소프트웨어를 구성하는 제어 로직들의 배치 정책을 지원하는 부분으로 구성한다.

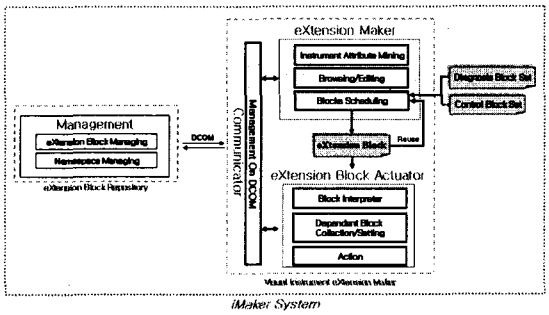

그림 3. iMaker 시스템

확장블록을 생성하는 부분(eXtension Block Maker)은 장비를 컨트롤하는 OPC 컴포넌트에서 장비의 특성과 행위를 추출하고(Instrument Attribute mining), 편집하여 XBML 형태로 포장한다(Browsing/Editing). 이렇게 생성한 확장블록은 재사용 가능하며 DCOM 기반의 전송 부분(Communicator)를 사용하여 원격에 존재하는 확장 블록 저장소에 저장하여 중앙 집중적으로 관리 할 수 있다. 또한 진단 블록(Diagnosis Block)과 제어 블록(Control Block)을 조합하여 사용자 정의의 확장블록을 생성 및 배치 정책을 설정 할 수 있다. 하나 이상의 장비 제어 로직을 포함 하고 있는 확장 블록은 실행자(eXtension Block Actuator)를 통하여 확장 블록을 구성하는 하위의 확장 블록을 해석하고(Block Interpreter), 하위 블록들의 정보를 전송자(Communicator)를 이용하여 읽고 OPC 서버의 프락시 객체를 생성하여 로컬 및 원격에 존재하는 OPC 서버와 DCOM을 통하여 연결한다(Dependant Block Collection/Setting). 마지막으로 장비의 제어 로직을 프락시 객체들을 사용하여 제어한다(Action). 대행자(Communicator)는 XBML 파일을 분석하여 OPC 컴포넌트가 원격에 존재하면 OPC 컴포넌트의 프락시 코드를 생성하고 원격에 존재하는 OPC 컴포넌트와의 연결을 시켜서 원격에 존재하더라도 로컬에 존재하는 것처럼 위치 투명성을 제공한다.

확장 블록 저장소(eXtension Block Repository)는 확장 블록을 관리하고(eXtension Block Managing), 블록의 중복성을 피하기 위하여 네임스페이스 개념을 이용하여 처리하였다.(Namespace Managing)

3.2 XBML(eXtension Block Markup Language)

확장 블록(eXtension Block)은 장비 제어의 기본 단위이며 여러 블록들을 스케줄링 할 수 있는 중첩된 구조와 재사용 하기 편리하게 설계 되었다. 확장 블록은 기본 타입과 확장 타입의 형태로 분류할 수 있다.

기본 타입은 장비가 가질 수 있는 하나의 요소(제어 정보, 예를 들면 OPC-Server-Name.Group.Item의 형식)의 정보와 선택한 요소가 가질 수 있는 값의 집합을 중심으

로 여러 정보(Block Information, View, Basic OPC Component)가 결합된 형태이다. 이렇게 구성한 제어 정보와 값들로 인하여 장비는 행위를 하게 된다.

확장 타입은 다양한 블록들-기능 블록(Function Block), 컨트롤 블록(Control Block), 진단 블록(Diagnosis Block)-을 사용하여또 다른 하나의 확장 블록으로 재구성된 블록이다.[3] 재구성 정보는 장비제어 로직(Instrument Control Logic)으로 표현되며 이 정보를 중심으로 여러 정보(Block Information, View, Basic OPC Component, Reference Block)가 결합된 형태를 가진다.

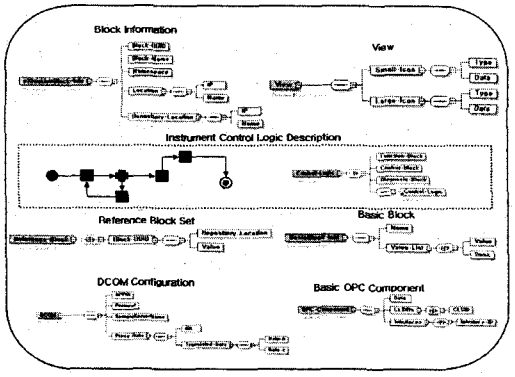

그림 4. 확장 블록구조(eXtension Block Structure)

확장 블록 디자인 시 특별한 뷰(View)와 기본 OPC 컴포넌트(Basic OPC Component) 부분에 기입 정보는 기존에 많이 사용하는 형식(파일의 위치 정보)이 아니라 파일의 내용(Image Data) 자체를 태그 사이에 기입하는 형태로 디자인 하였다. 물론 이러한 시도가 많은 단점을 가지고 있지만 인터넷을 사용하여 방화벽을 자유롭게 통과하여 어디서든지 이동할 수 있고, 확장 블록을 지원하는 툴(iMaker)에서 이미지나 컴포넌트를 재구성하여 사용할 수 있는 이점을 가지고 있어서 본 논문에서는 이러한 방법을 채택하였다.

3.3 확장 블록 저장소(XBML Repository)

확장 블록의 정보는 각기 분산되어 있는 저장소(eXtension Block's Repository)에 저장되도록 설계 하였다. 저장소는 XML 형태의 저장 카테고리와 이를 관리 하는 확장 블록 저장 컴포넌트(eXtension Block Storage Component)로 구성한다. 아래의 그림 [그림 2]는 확장 블록(eXtension Block)이 자신을 구성하는 다른 확장 블록들의 정보(다른 저장소에 저장되어 있는 블록 정보)를 가져오는 과정을 기술한 것이다.

확장 블록(eXtension Block)은 ICB에 의해서 해석되며, 확장 블록을 구성하는 다른 확장 블록이 있는 지 없는 지 알아본다. 만약 있으면 구성되는 블록의 식별자(UUID) 값과 저장되어 있는 저장소(Repository)의 위치 정보를 취득한다(0). 현재 참조하는 저장소를 관리 하는 컴포넌트(eXtension Block Storage Component)를 이용하여 참조하는 블록 정보를 취득한다(1, 1.1.x). 현재 참조하지 않는 저장소에 있는 확장 블록은 컴포넌트의 위치 투명성을 사용하여 원격에 있는 확장 블록을 관리하는 컴포넌트에 접속하고(1.2) 결과를 OPC 레퍼 컴포넌트가 참조할 수 있는 곳으로 읽어 와서(1.2.x) 확장 블록을 구성하는 모든 구성 요소를 갖추게 된다.(2)

이런 과정을 통하여 OPC 래퍼 컴포넌트는 블록의 정보들을 분석하여 OPC 컴포넌트를 재구성하고 레지스트리에 정보를 추가한다. 이때 재구성한 OPC 컴포넌트는 확장 블록에 명시된 컴퓨터의 IP(Computer's IP)에 존재하는 OPC 컴포넌트와 연결할 수 있는 OPC 컴포넌트의 프락시 역활을 한다. (COM은 COM 컴포넌트에 접근할 수 있는 프락시 역할을 하는 부분을 포함) 이러한 과정을 거치면서 하나의 블록에 의존적인 여러 블록에 연결되어 있는 컴포넌트들과의 연동을 할 수 있다.

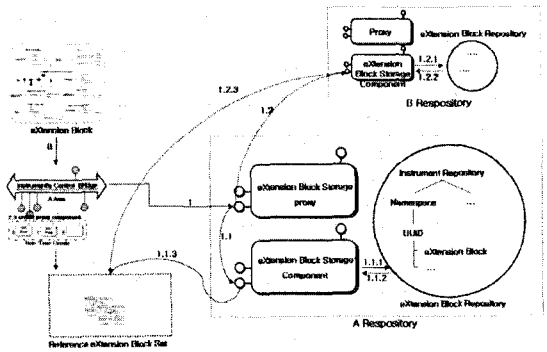

그림 5. 확장 블록 저장소

3.4 Instrument's Control Broker

ICB는 분산 환경에 존재하는 ICB들과 상호 협력 관계(Coordination)를 유지하고, ICB와 제어 로직 사이의 관계를 서비스와 요청의 관계로 디자인하였다. 이로 인하여 제어 로직의 실패(Fault)에 대해서 롤백과 같은 다양한 서비스를 제공할 수 있는 구조가 된다. 이러한 서비스 개념으로 인하여 우리는 ICB가 여러 ICB와 협력할 수 있는 것들을 공통 서비스(Common Service)라고 정의하며, 확장 블록 저장소(eXtension Block Repository)에서 확장 블록(XBML)을 검색하여 특정 장비에 맞는 프록시를 생성하

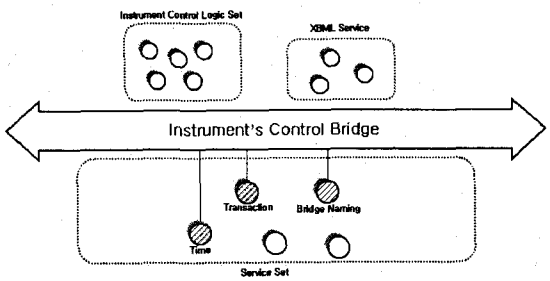

그림 6. Instrument's Control Bridge

는 부분을 확장 블록 서비스(XBML Service)하였다. 이러한 과정에서 CORBA의 ORB(Object Request Broker)[x]의 아키텍처를 참조하였으며, ICB의 실제 구현은 C#을 사용한 윈도우 서비스 형식으로 디자인하였다.

- 장비 제어 로직(Instrument's Control Logic)

장비 제어 로직은 iMaker 시스템에서 생성한 확장 블록을 의미 하며, 여러 개의 제어 로직이 모여서 하나의 제어 소프트웨어가 된다. 특히 각각의 제어 로직은 디자인시 개별적인 배치 정책을 가진다.

- 공용 서비스(Common Service)

ICB에서 제공하는 공용 서비스는 시간(Time), 트랜젝션

(Transaction), 동기화(synchronization), 핑거(Pinger) 서비스 등이 있다.

* 시간(Time) 서비스: 여러개의 ICB가 모였을때 시스템마다 다른 클럭을 가지고 동기화 하기 때문에 시스템 의존적인 동기화로 인하여 전체 연동에 문제가 발생할 수 있다. 이에 서로 연결된 ICB 들간에 주 ICB에 맞는 시스템의 클럭을 사용하여 동기화하는 서비스이다.
* 트랜젝션(Transaction) 서비스: 여러개의 ICB가 연동되어 있는 경우 제어 로직들이 작업을 하는 상황에서 예기치 않은 오류로 인하여 제어 로직이 실패 했을 경우 어떻게 처리할지를 설정하는 서비스이다. 트랜젝션의 종류는 3가지로 분류하였다. 지원하지 않음(Not Support), 지원함(Supported), 요구(Required)로 구성한다. 트랜젝션 규칙은 제어 로직을 포함하는 제어 로직의 룰에 따르는 특징을 가지고 있다. "지원하지 않음"은 오류 발생시 아무런 영향없이 넘어가는 경우이며, "지원함"은 구성하고 있는 로직이 트랜잭션에 속해 포함 관계일 때 최 상위에 포함된 정책에 의해 다른 ICB에 의해서 재 할당되어 동작하게 된다. "요구"는 항상 트랜잭션을 가지며 오류 발생시 다른 ICB에 의해서 재 할당되어 동작하게 된다.
* 동기화 (Synchronization) 서비스: 타임 서비스를 사용하여 동기화를 제공하는 서비스이다.
* 핑거(Pinger) 서비스: 분산되어 설치된 ICB의 CPU와 네트웍 트래픽의 상태 정보를 알려주는 서비스이다.

- 확장 블록 서비스(XBML Service)

확장 블록 서비스는 확장 블록 저장소(XBML Repository)에 저장되어 있는 확장 블록 정보를 찾고, 장비에 연결되어 있는 OPC 객체와 결합할 수 있는 OPC 프록시(Proxy)를 생성하는 결합 서비스, 사용 후 해제(Release)시키는 서비스 등으로 구성되어 있다.

4. 배치 정책 (Deployment Policy)

분산 환경에서 특정 작업의 협력 과정 중 발생하는 마스터 컴퓨터(Master Computer)의 개념을 ICB에서는 어떻게 처리하는지 알아본다. 또한 제어 소프트웨어를 구성하는 제어 로직들이 ICB 서비스들 사이에 초기 배치 정책과 특정 ICB 오류 발생시 특저 ICB가 수행해야 될 제어 로직의 처리 문제에 대해서 논한다.

- 일반 배치 정책

토큰 개념을 사용하여 토큰을 가진 ICB가 마스터(Master) ICB가 되다. 토큰은 공용서비스 중 핑거(Pinger) 서비스를 사용하여 ICB가 있는 CPU의 트래픽을 조사하며, 트래픽이 적은 순위를 기준으로 ICB는 공통 순위 리스트를 가지며 첫 번째 순위의 ICB가 토큰을 부여 받는다. 각각의 ICB는 공통 순위 리스트를 각자 가진며 공통 순위 리스트를 가지는 과정은 현재 상위 순위 리스트에 있는 ICB는 하위 리스트 중 CPU의 트래픽을 변경할 수 있는 규칙에 의해서 만들어진다. 이렇게 부여 받은 마스터 ICB는 공용서비스(Common Service)를 사용하여 다른 ICB와의 접속하고 여러 서비스와 동기화 시킨다.

- 오류 발생 시 정책

제어 로직의 오류 발생시 제어 로직의 트랜잭션의 특성에 따라 제어 로직을 스킵(skip) 또는 롤백하여 처리 할 수 있다. ICB 오류시는 마스터가 아닌 ICB의 오류와 마스터 ICB의 오류로 분류할 수 있다.

마스터가 아닌 ICB의 오류 발생의 경우 마스트가 오류 발생 ICB의 제어 로직을 2번째 순위(마스터를 제외한 가장 적은 CPU의 부하를 가진 ICB의 ICB에게 처리 하도록 유도하여 해결한다.

마스터 ICB 오류 발생의 경우 서로 다른 ICB들이 핑거 서비스를 사용하여 공용 순위 리스트를 작성 하므로 가장 부하가 적은 ICB에게 토큰이 이동하고 마스터가 수행하던 제어 로직은 새로 작성된 2번째 ICB에게 이양된다. 물론 이양된 ICB는 제어와 연동된 OPC 프락시가 설정되고 이전의 작업을 트랜젝션 속성을 기준으로 해결한다.

5. 결론 및 향후 연구 과제

우리는 iMaker 시스템이 제공하는 중앙 집중적인 제어 방식에 안정성을 제공하기 위하여 분산 모델로 변경을 시도 하였다. 특히 시스템 오류시 자동 오류 극복(fault torance)할 수 있는 분산 배치 및 통합 모델로 인하여 PC 기반의 장비 제어 기술에 신뢰성(Reliability)을 부여 하였다.

향후 연구 과제는 공용 서비스(Common Service)와 확장 블록 서비스(XBML Service) 부분에 대한 세부적인 연구와 함수 포인터를 사용하여 사용자 정의 서비tm를 포함할 수 있는 구조로의 연구가 수행되어야 한다.

참고문헌

[1] P.K. Wright, "Principles of open-architecture manufacturing" , Journal of Manufacturing System, vol. 14, no.3, pp. 187-202, 1995

[2] P.K. Wright, "Principles of open-architecture manufacturing" , Journal of Manufacturing System, vol. 14, no.3, pp. 187-202, 1995

[3] Minsuck Shim, Sungkyu Park, Dae-Sung Yoo, Jong-Hwan Kim, Myeongjae Yi, " A Study on the Flexible and Efficient Instrument Control Software Generation ", Proceedings of the 7th Korea-Russia International Symposium, KORUS 2003, vol2. pp, 447-452

[4] 최장원, 박찬열, 박학수, 이필우, 황일선, "인터넷 기반 분산컴퓨팅 테스트베드 구축" , 정보처리학회 2002년 추계학술대회, VOL.09 NO.02, 2002.10

[5] Michael Philippsen and Matthies Zenger, "JavaParty-Transparent Remote Objects in Java", In the ACM Workshop on Java for Science and Engineering Compution, 1997

[6] J.E. Baldeschwieler, R.D. Blumofe, and E,A,Brewer, "ATLAS: An Infrastructure for Global Computing", In Proc. of the 7th ACM SIGOPS European Workshop