CORBA를 이용한 XML기반의 오더관리시스템 개발

- Development of An XML-based Order Management System Using CORBA -

김영균*
Kim Young Gyun
이홍희*
Lee Hong Hee

Abstract

An Order Management System related to press die fabrication in automobile industry are developed in this research. This system is constructed by CORBA and XML to avoid disorder from data integration. Because CORBA can integrate diverse applications within distributed heterogeneous environments and XML can be made use of data exchange format, The developed system is available on almost any hardware platforms, operation system environments, and programming languages.

1. 서론

산업 및 경쟁의 글로벌화에 따라 기업을 둘러싼 시장과 기술이 급변하고 있다. 이러한 생산 환경에서 생존하고 성장하기 위하여 제조기업들은 그들의 제조 조직을 다수의 공장, 공급업체와 협력업체 등의 분산생산시스템으로 대응해 나가고 있다. 이 때, 각 조직들간의 긴밀한 협력이 필요하나, 업무처리의 시간적·공간적인 제약성과 서로 이질적인 생산기술 및 시스템으로 인하여 생산 정보의 통합과 교류는 현실적으로 어려운 상황이다. 최근 네트워크 컴퓨팅의 발달로 인터넷을 통해 전 세계의 사람들이 공동체를 형성할 수 있게 되었다. 그러므로, 글로벌 시장에 민첩하게 대응해 나가려는 생산시스템의 업무 흐름을 파악하여 이에 적합한 정보시스템으로 생산활동을 지원해야할 필요성이 절실하다. 따라서, 본 연구에서는 시간과 공간의 제약을 극복할 수 있는 인터넷이라는 매체를 활용하여 분산제조 환경에서 생산을 지원하는 정보시스템을 구축하고자 한다.

* 인하대학교 산업공학과

이 시스템은 자동차 프레스 금형 생산 프로세스의 외주 관리를 대상으로 하고 있다. 즉, 수주업체와 발주업체가 공유한 정보를 바탕으로 하여, 의사결정에 필요한 정보를 서로 입력하고 조회하면서 발주 대상 오더를 결정하고 수주업체를 선정하며 납기 준수를 위한 진도 관리를 수행하는 외주관리 업무를 지원한다. 본 연구의 목적은 이기종 컴퓨팅 환경과 다중 어플리케이션 환경에 인터넷과 객체지향을 도입하여 문서를 통합하는데 있다. 이를 위하여 이종의 분산된 환경 하에서 응용프로그램들을 통합할 수 있는 표준기술인 OMA(Object Managements Architecture)의 CORBA(Common Object Request Broker Architecture)를 전송 Layer로 하고, XML(eXtensible Markup Language)을 데이터 교환 포맷으로 사용하여 데이터의 통합에서 야기되는 혼란을 피하도록 오더 관리 시스템을 구축한다. 시스템 구성은 3-tier 클라이언트 서버 환경으로, Java Application과 Web Browser의 Client, Web Server와 Application Server의 Middle server, 그리고 Database Server이다. [1]

2. CERBA와 XML

CORBA(Common Object Request Broker Architecture)는 분산객체 개발환경의 표준을 구현하기 위해 OMG(Object Management Group)에서 제정한 아키텍처이다. 플랫폼 환경에 구애받지 않고, 작성한 프로그래밍 언어가 달라도 ORB를 통하여 네트워크 상에 존재하는 어떠한 객체끼리도 서로 통신할 수 있다. 이런 CORBA 환경에서는 클라이언트와 서버가 서로 주고받을 데이터의 인터페이스를 정의한 후, IDL(Interface Definition Language)을 작성하면 비지브로커(VisiBroker) 등의 미들웨어가 클라이언트 통신코드인 스텁(Stub)과 서버 통신코드인 스켈레톤(Skeleton)을 자동으로 생성해 준다.[2]

마크업(markp)이란 본문, 즉 원래의 내용에 뭔가 특별한, 또는 추가적인 정보를 표시하는 것이다. XML은 확장성 마크업 언어란 이름 그대로 HTML 같은 고정된 형식이 아닌 확장이 가능한 언어이다. HTML은 태그의 종류가 한정되어 있는 반면 XML은 문서의 내용에 관련된 태그를 사용자가 직접 정의할 수 있으며 그 태그를 다른 사람이 사용하도록 할 수 있다. XML은 본질적으로 다른 언어를 기술하기 위한 언어, 즉 메타언어이다.[3][4]

3. 자동차 프레스 금형의 오더 관리

오더 관리 시스템은 D자동차의 프레스 금형 제작과 관련된 업무를 대상으로 하고 있다. 그림 1은 오더 코드의 생성을 나타낸 것이다. 현재 생산되고 있는 자동차의 이름을 프로젝트명으로 한다. 자동차 금형에 들어가는 모든 부품들을 전체 항목으로 정한 후 프로젝트별로 항목을 정해준다. 하나의 판재가 필요로 하는 형상으로 프레스 되기 위하여 거치는 가공순서에 따라 공정을 정하고 이에 맞게 금형을 만들어준다. 오더

코드는 그림과 같이 프로젝트, 항목, 공정별로 부여한다.

오더 관리 시스템은 D자동차의 프레스 금형 제작과 관련된 업무를 대상으로 하고 있다. 그림 2는 오더 관리 업무의 흐름을 나타낸 것이다. 신규 프로젝트를 등록할 때에는 프로젝트의 코드와 이름을 먼저 설정하고 각 프로젝트에 해당하는 항목들을 선택하여 공정별로 오더코드와 생산일정 계획을 입력한다. 이 때, 각 오더를 제작하는데 필요한 M/H를 금형 제작 이력에 근거하여 산출한 후 capacity로 입력한다. 기존에 등록되어진 프로젝트의 생산 일정이나 capacity의 변동이 있을 경우 그 정보를 수정한다. 이렇게 수정된 정보는 이력관리를 위해 보관한다. 금형생산 공장의 실제 가능한 월 별 capacity 정보를 M/H 단위로 등록한다.

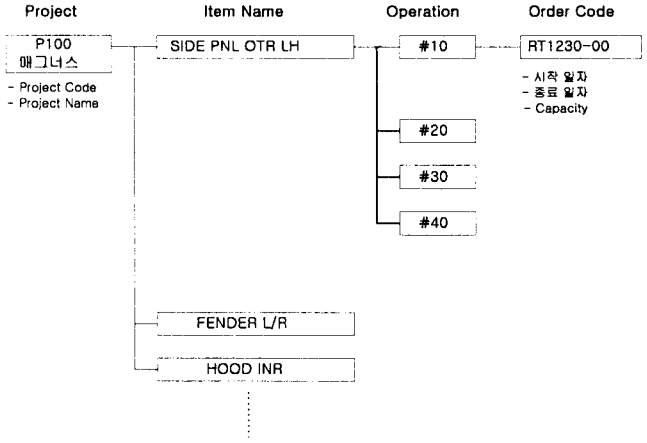

그림 1. 오더 코드 생성

금형 생산 공장의 상황에 변동이 생기면 capacity 정보를 변경한다. Project 등록과 금형 생산 공장의 capacity 등록을 통해 얻은 정보를 바탕으로 Load Table을 작성한다. 이 load를 금형 생산 공장의 capacity로 나눈 비율이 100%보다 큰 오더를 발주대상으로 고려하게 된다. 분산되어 있는 수주 가능업체들로부터 월 별로 수주업체의 공장에 관한 생산 능력과 생산일정에 따른 생산 계획량 정보를 받아서 잔여 capacity를 계산한다. 해당기간에 잔여 capacity가 오더r를 제작하는데 필요로 하는 capacity보다 큰 업체를 수주가능 업체로 하여 이들 중에 적절한 업체를 선정하여 발주한다. 진척상황을 파악해 가면서 발생하는 문제를 해결하여 수주업체가 납기를 준수하도록 한다.

4. 코바 프로그래밍

코바 자바 프로그램은 다음과 같은 순서로 개발한다: (1) IDL 인터페이스를 작성, (2) IDL 컴파일러를 이용해 스텁/스켈레톤, (3) 코드를 생성, (4) 구현객체를 작성, (5) 서버 main()함수를 작성, (6) 클라이언트 프로그램을 작성. 그림 3.은 코바 자바 개발 과정을 나타내고 있다.

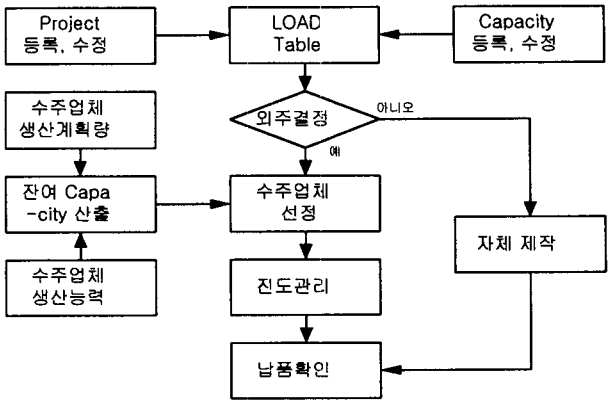

그림 2. 오더 관리 업무 흐름도

코바 서버 프로그램은 다음과 같은 순서로 작성한다: (1) 프로젝트 생성, (2) 코바 객체에 대한 인터페이스 정의, (3) IDL 파일 컴파일, (4) 구현객체 작성, (5) 코바 서버 프로그램 작성. 코바 어플리케이션 클라이언트 프로그램은 다음과 같은 순서로 작성한다: (1) 프로젝트에 Frame을 갖는 Application추가, (2) 클라이언트 화면 설계, (3) 각 컨트롤의 속성을 지정하고 함수를 추가, (4) 소스에 필요한 코드 추가(스텁/스켈레톤 패키지와 코바 패키지 임포트 등). 코바 애플릿 클라이언트 프로그램은 다음과 같은 순서로 작성한다: (1) 프로젝트에 Applet 추가, (2) 애플릿 화면 설계, (3) 각 컨트롤의 속성을 지정하고 함수를 추가, (5) 소스에 필요한 코드 추가(스텁/스켈레톤 패키지와 코바 패키지 임포트 등).

IDL은 OMG에서 규정하고 있는 인터페이스 정의 언어의 표준으로 객체지향 개념을 기초로 하고 있으며, 다중상속 및 동적 호출 메커니즘을 지원한다. 그리고 IDL은 특정 언어에 독립적인 인터페이스 정의 언어이다. 구현언어가 아닌 정의 언어이며, 구현 언어로의 매핑을 지원한다. Interface는 attribute와 operation이 포함되어 있다. Attribute에는 속성이 포함되는데 디폴트는 read-write이며, IDL 컴파일러는 자동으로

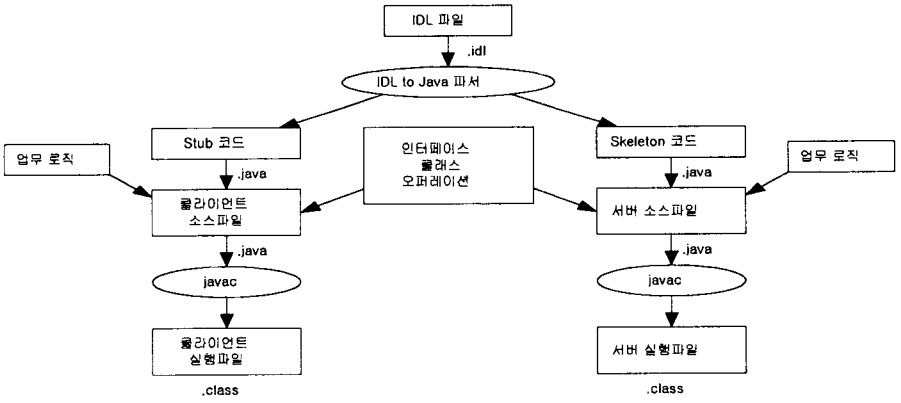

그림 3. 코바 자바 개발 과정

attribute에 접근할 수 있는 set(), get() 함수를 만들어 준다. Interface에 operation을 추가할 때는 method이름, method리턴형, parameter데이터형과 주고받는 방향(in, out, inout), 에러가 발생했을 때에 반환되는 예외 등을 지정한다.

5. 시스템 구성 및 구현

본 시스템은 Client, Middle Server, Database Server 즉, 3-tier로 구성된다. Client는 다시 Java Application과 Web Browser로 나뉘어지며 Middle Server는 Application Server와 Web Server로 나뉘어지고 마지막에 Database Server가 위치하게 된다. 수주자는 Web Browser로 수주업체의 공장 정보를 입력하고, 발주자는 자바 어플리케이션을 사용하여 각 업체의 상황을 파악하여 발주한다.

RDBMS와 XML based Application 사이에 연결을 담당하는 bridge 역할을 하는 무엇인가가 필요한데, 이것을 SQLX라 부른다. SQLX는 SQL문을 XML로, XML을 SQL문으로 변환시켜주는 역할을 한다. 그래서 단순한 SQL Query를 실행하기 위해, 구조적 XML file(DOM tree)로 바꾸어 주는 작업이 중간에 추가된다. 오더 관리 시스템의 데이터 흐름을 그림 4에서 보여주고 있다. Application에서 만들어진 데이터는 어플리케이션 측 XML Parser를 통하여 구조화되고 정형화된 데이터로 걸러진 후 CORBA ORB 객체에 전해지게 된다. SQLX Server에서는 이렇게 전해진 데이터를 다시 XML Parser를 통해 DB구조에 적합한 데이터로 변환한 후 SQLX를 사용하여 DB에 저장한다.

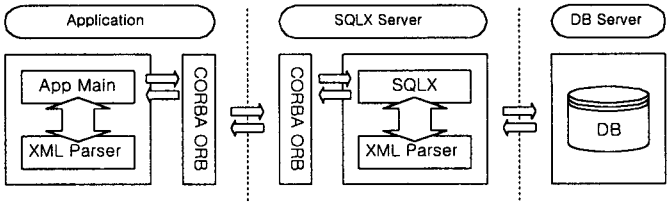

그림 4. 데이터 흐름도

오더와 업체 관련 물리적 데이터 모델은 그림 5과 같다.

프로젝트 등록을 위하여 프로젝트의 코드와 이름 그리고 프로젝트에 포함되는 항목들을 등록한다. 이 때, 공정별로 오더 코드를 부여하게 되며, 일정계획에 근거하여 시작 일, 종료일 그리고 capacity를 입력한다. 금형 생산 공장의 실제 가능한 capacity 정보를 월 별로 각각의 M/H 단위로 등록한다. 금형 생산 공장의 상황에 따라 capacity 정보를 수정할 수 있다. 테이블을 통해서 각 오더들의 월 단위 capacity를 합산하여, 해당 월의 실제 금형 공장의 capacity로 나눈 부하율(%)을 구한다. 부하율이 100%보다 클 경우에 우선적으로 발주를 고려하게 된다. 업체 선정을위해 잔여 capacity를 구한다. 잔여 capacity는 외주업체의 생산 가능한M/H로, 공장이 최대한 생산할 수 있는 생산능력에서 일정에 이미 잡혀 있는 생산계획량을 뺀 값이다.

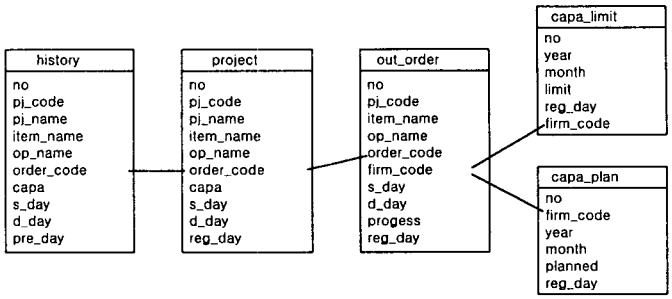

그림 5. E-R 다이어그램

그림 6에 발주결정화면이 있다. 이 때, 생산능력과 생산계획은 WEB을 통하여 수주를 원하는 업체로부터 입력을 받는다. 발주결과에 발주 대상 오더와 수주업체가 나란히 올라오고, 확인 버튼을 누르면 발주결정이 확정된다. 이력관리는 수정되기 전의 정보와 수정된 후의 정보를 검색한다. 오더코드, 프로젝트 코드와 변경일자의 3가지 검색 방법이 있다.

진도관리에서진척상황을 파악한다. 수주업체, 프로젝트 코드와 오더코드의 3가지 검색방법이 있다. 수주업체의 공장에 관한 생산 능력과 생산일정에 따른 생산 계획량을 입력할 수 있는 있다. 입력받은 값들을 가지고 잔여 capacity를 계산하여 발주결정에 이용하게 된다. 수주 받은 오더들에 대한 진척률을 입력한다. 그림 7에 진척률 입력화면이 있다.

6. 결론

인터넷을 이용하여 분산 환경에 있는 기업사이의 오더 관리 정보를 교환해야 하는 필요성을 충족시키기 위해서 특정 하드웨어 플랫폼이나
운영체제, 프로그래밍 언어에 구속되지 않도록, 문서통합기준인 XML과 분산처리가 가능하게 하는 CORBA를 연동시켜 오더관리 시스템을 구현하였다.데이터 처리의 유연성은 매우 증대되지만 시스템에 걸리는 부하가 가중되어 처리속도 등의 효율이 떨어지는 문제점이 있다. 기업 내부에서 수행하던 활동이 다른 경쟁기업 또는 공급업체, 그리고 고객과의 네트워크로 연결되는 하나의 네트워크형 조직을 통해 수행하는 방식으로 점차 바뀌고 있다.

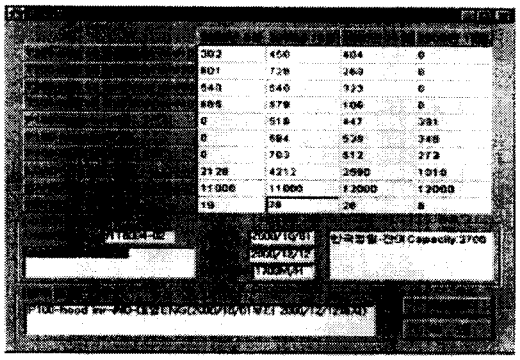

그림 6. 발주결정 화면

이에 따라 앞으로 XML과 CORBA의 기능이 더욱 강력해지고 구조가 간단해지면 이 두 가지 개념의 조합으로 다양한 기업용 어플리케이션이 개발될 것으로 예상된다. 연구대상이된 D자동차사에 이 연구로 개발된 시스템을 적용하여 오더관리, 발주관리에 시간과 노력을 크게 절감하는 효과를 보았다. 이 시스템은 다른 인터넷기반 관리시스템과 연계하여 효과적으로 작동하였다.

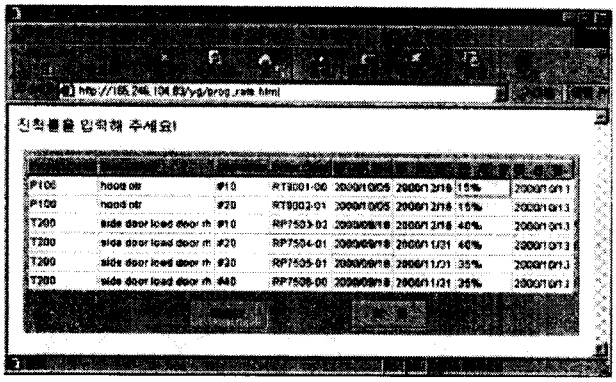

그림 7. 진척률 입력 화면

7. 참고문헌

[1] Amit Misra, Sztipanovits Janos, Karsai Gabor, Moore Michael, Ledeczi Akos, and Long Earl, "Model-Integrated Computing and Integration of Globally Distributed Manufacturing Enterprises: Issues and Challenges", *Procd of 1999 IEEE Conference on Computer-Based Systems*, IEEE Computer Society, 1999.

[2] Damiani E, di Vimercati SD, Paraboschi S, and Samarati P, "Design and implementation of an access control processor for XML documents", *The*

International Journal of Computer & Telecommunications Networking, Elsevier, 2000.

[3] Ken'ya NISHIKI, YOSHIDA Kenichi, and OOTA Masataka, "Integrated Management Architecture based on CORBA", *Proceedings of the 2000 IEEE/IFIP Network Operations and Management Symposium*, IEEE, 2000.

[4] Vinoski S., "Introduction to CORBA", *Proceedings of the 2000 International Conference on Software Engineering*, IEEE, 2000.

저 자 소 개

김영균 : 인하대학교 산업공학과에서 학사와 석사학위를 취득.
현재 대상정보시스템(주)에서 근무하고 있음.

이홍희 : 서울대학교 기계설계학과에서 학사와 석사학위를 취득.
미국 펜실베니아주립대학교 산업공학과에서 박사학위를 취득하였음.
현재 인하대학교 산업공학과 부교수 재직중.