韓國精密工學會 '97年度 春季學術大會論文集 pp.395~399

STEP을 이용한 분산 CAD/CAE 환경

권기억, 박명진 (중앙대학교 대학원 기계설계학과)
최영, 조성욱 (중앙대학교 기계설계학과)

Distributed CAD/CAE Environment Using STEP

Ki-Eak Kwon, Myung-Jin Park, Young Choi, Seong Wook Cho
(Dept. of Mech. Design and Prod. Eng., Chung-Ang Univ.)

Abstract

An international standard for the product model data, STEP, and a standard for the distributed object technology, CORBA, will play a very important role in the future manufacturing environment. These two technologies provide background for the sharing of product data and the integration of applications on the network. This paper describes a prototype CAD/CAE environment that is integrated on the network by STEP and CORBA. Several application servers and client software were developed to verify the proposed concept.

Key word : STEP, CORBA, CAD, CAE.

1. 서 론

현재의 CAD/CAE 시스템들은 대부분 독립적으로 운용되며 각 시스템마다 고유의 데이터 포맷을 갖는다. 만일 공동의 프로젝트를 수행하는 협력 업체들이 각기 다른 CAD/CAE 시스템들을 보유하고 있다면 프로젝트 진행에 어려움이 생기게 된다. 이렇게 기업마다 상이한 CAD/CAE 시스템들을 분산 객체 기술의 표준인 CORBA를 이용하여 네트워크로 연결 한다면, 원격지에 있는 시스템 간의 협동 작업을 통하여 서로의 자원을 공유하거나, 필요한 정보를 주고 받는 등 일련의 상호작용이 가능해 진다. 또한 CAD/CAE 정보를 제품 데이터 교환의 표준인 STEP으로 통일 한다면 시스템간의 데이터 변환이 필요 없이 제품의 생명주기의 모든 측면을 다룰 수 있다. 그러나 아직까지 CORBA와 STEP을 지원하는 CAD/CAE 시스템이 발표되지 않았으므로 이와 관계된 연구의 필요성이 시급하다.

본 연구의 목적은 설계·제조 환경에서 사용할 수 있는 분산 환경 — STEP을 기반으로 하는 제품 데이터 관리 모듈, CAD 모듈, CAE 응용 모듈 들이 CORBA를 통해 네트워크 상에서 연결되어 동시에 상호작용(interoperation)하는 네트워크 환경 — 을 구현하여 기업내의 설계·제조 파트를 구축하는 것이다. 이러한 분산 환경은 CAD/CAE 부문에만 한정되도록 설계되어 있지 않기 때문에 기업내의 전 영역으로 확장시킨다면 가상 기업(virtual enterprise) — 네트워크를 통하여 정보를 공유하며, 생산활동에 참여하는 기업들의 모임 — 을 구현하는 데에 매우 중요한 핵심기술이 될 것이다. 그림 1은 STEP과 CORBA를 이용한 가상 기업의 구조를 나타낸 것이다. [1, 2]

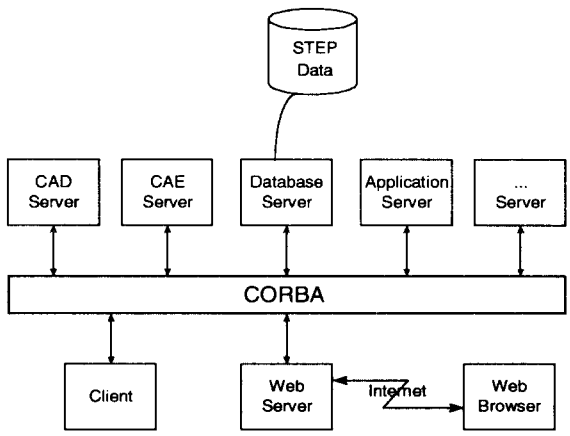

그림 1 가상기업의 구조

2. 관련 표준과 기술

2.1. STEP

제품 데이터 교환의 새로운 표준으로 추진되고 있는 ISO 10303 STEP (Standard for the Exchange of Product Model Data)은 모든 산업에서 제품 생명 주기의 모든 측면을 다루는 중립적인 메커니즘을 제공하는 것을 목표로 하고 있다. STEP은 생산정보와 설계정보를 모두 포함하고 있다는 점에서 다른 표준들과 근본적인 차이를 갖는다. STEP 표준의 근간이 되는 것은 EXPRESS 정보 모델링 언어이다. EXPRESS는 객체지향적으로 설계되었기 때문에 기존의 STEP에 새로운 기술들을 첨가하는데 매우 적합하다. [3, 4]

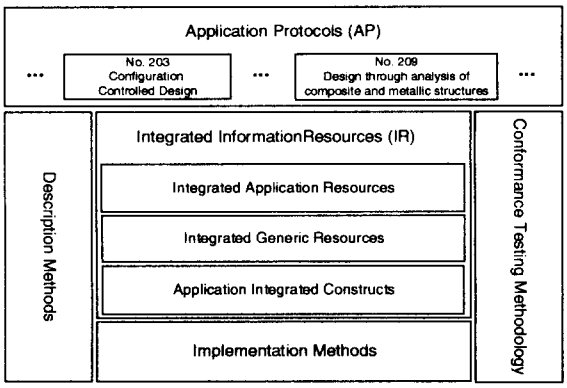

그림 2 STEP의 구조

그림 2는 STEP의 전체적인 구조를 나타낸 것인데 이중 응용 프로토콜(application protocol, AP)은 각 산업 응용 분야의 정보 요구(information requirement)를 명시하여, 신뢰성 있는 정보 교환을 위한 메카니즘을 정의하는 것이다. 본 연구에서 개발된 **DICCES**(DIstributed CAD/CAE Environment using STEP)는 STEP AP중에서 CAD와 관련된 AP203 "Configuration controlled design"과 CAE와 관련된 AP209 (Design through analysis of composite and metallic structures) 를 사용하였다. [5, 6]

2.2. CORBA

분산 컴퓨팅 시스템의 개발을 위한 객체 지향 표준을 제정하기 위해 700여 이상의 컴퓨터 관련 단체가 모여 OMG(Object Management Group)를 결성하였고 OMG에서 이종의 분산된 환경 하에서 응용 프로그램들이 서로 통합될 수 있는 표준 기술인 OMA(Object Management Architecture)를 제안하였다.

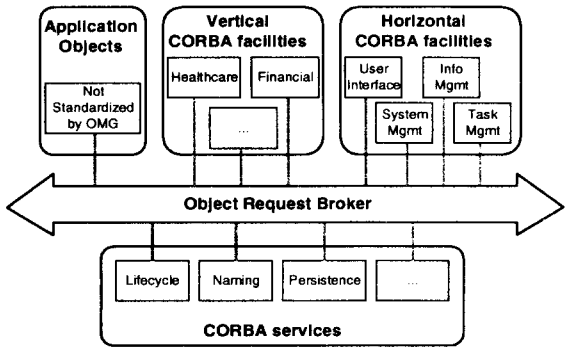

그림 3 OMG's OMA

CORBA(Common Object Request Broker Architecture)는 OMA에서 이종의 네트워크 시스템에서 객체간의 통신을 가능하게 해주는 아키텍처를 문서화하여 발표한 것이다. 1991년에 발표된 CORBA 1.1은 IDL과 구현된 ORB에서 클라이언트/서버간의 상호작용을 가능하게 하는 API를 정의 하였다. 그 이후 1994년 12월에 채택된 CORBA 2.0은 서로 다른 벤더들에 의해 생산된 ORB간의 상호작용을 가능하게 하였다.

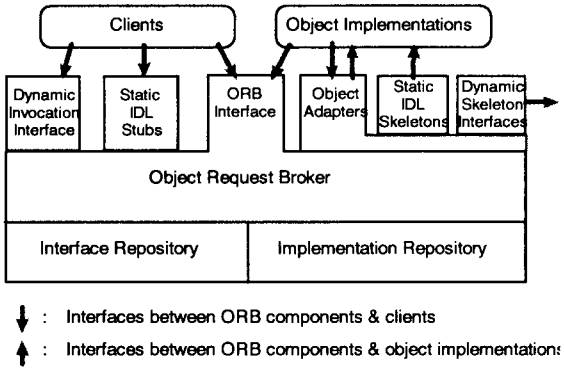

그림 4 ORB의 구조

CORBA의 핵심 기술은 ORB(Object Request Broker)이다. ORB는 객체간에 클라이언트/서버 환경을 구축해주는 미들웨어이다. 즉 ORB는 클라이언트의 요청 메시지를 적당한 객체들로 보내준 다음 그 객체들로부터 나온 결과 메시지를 클라이언트로 보내주는 역할을 한다. 또한 ORB는 다중 오브젝트 시스템(multiple object systems)을 서로 연결할 수 있다. ORB를 사용하여 클라이언트는 로컬 머신이나 네트워크 건너편에 있는 서버 객체의 method를 호출할 수 있다. 클라이언트는 객체들이 어느 네트워크에

있는지, 그들과 어떻게 통신 할 것인지, 그들이 어떠한 프로그래밍 언어로 구현되었는지, 그들의 OS가 무엇인지, 그들이 어떻게 저장되었는지, 그리고 그들이 어떻게 실행 되는지 등에 대하여 사전 지식이 없어도 객체들의 서비스를 이용할 수도 있다. 그림 4는 ORB의 구조를 보여주고 있다.

CORBA에서 각각의 객체는 인터페이스를 필요로 한다. 이러한 객체의 인터페이스는 각 객체가 제공하는 서비스의 타입을 의미하며 IDL(Interface Definition Language)로 기술된다. IDL은 basic data type들(short, long, float, double, 그리고 Boolean)과 constructed type들(struct, union)과 template type들(sequence와 string)을 제공한다. 그리고 각각의 인터페이스 내에는 여러 개의 오퍼레이션들이 있을 수 있다. 바로 이 오퍼레이션이 객체가 제공하는 서비스를 의미하는 것이다. [7, 8, 9]

3. 프로토타입 분산 CAD/CAE 환경

3.1. 분산 객체 시스템 구축

현재의 네트워크 환경은 여러 벤더(vendor)들에 의해서 생산된 이기종의 시스템들로 이루어져 있다. 이러한 네트워크 시스템에서 정보를 공유하는 분산 시스템을 구축하기 위해서는 어플리케이션(application)의 통합(integration)과 분산 프로세싱(processing)이 동시에 고려되어야 한다. 즉 응용프로그램에 네트워크 기능을 추가시켜야 하고 네트워크 시스템간의 상호작용(Interoperability)을 고려해야 한다. 그리고 제작된 응용프로그램은 앞서 밝힌 바와 같이 하드웨어 종류, 운영 체제, 네트워크 프로토콜과 어플리케이션 포맷에 무관하여야 한다.

3.2. 개발 환경

DICCES의 개발 환경은 표 1에 나타나 있다.

표 1 DICCES의 개발 환경

H/W	Pentium PC
OS	MS Windows NT 4.0
STEP tool	ST-Developer 1.5
CORBA tool	IONA ORBIX 2.02
C++ compiler	MS Visual C++ 4.2
Network protocol	TCP/IP
Graphic library	OpenGL Library

3.3. DICCES의 구조

DICCES은 DB 서버와 응용 프로그램 서버를 사용자들이 사용하는 클라이언트 부분과 완전히 분리하고 이들 사이에 연결 기능을 제공하는 미들웨어, 즉 ORB를 위치시킴으로써 좀더 유연하고 확장 가능한 시스템을 구현하였다. DICCES은 STEP을 기반으로 한 DB 서버를 기점으로 하여 여러 종류의 응용 서버들(CAD 서버, CAE 서버 등)을 네트워크로 연결하여 데이터와 응용 시스템들을 공유하게 된다. 기본적으로 클라이언트는 모든 서버와 연결하여 작업할 수 있으며, 각각의 응용 서버들은 DB에 데이터를 요청할 수 있다. DICCES의 구조는 그림 5에 표현되어 있다.

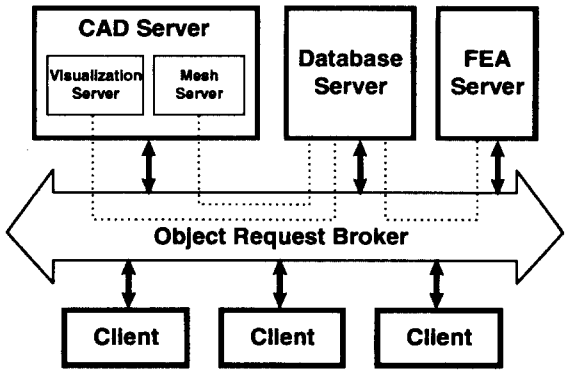

그림 5 DICCES의 구조

3.4. 클라이언트의 작업 시나리오

클라이언트가 서버에 연결하여 작업을 하는 순서는 다음과 같다.

① DB 서버에 연결하면 DB 서버가 작업 대상 STEP 물리 파일(AP203, AP209, ...) 리스트를 클라이언트에 보내준다. 이 리스트에서 작업을 해야하는 STEP 물리 파일을 선택한다.

② 응용 서버에 연결하여 선택한 STEP 물리 파일의 이름을 넘겨주면, 응용 서버는 DB 서버에 연결하여 DB 서버의 메모리에 선택된 STEP 물리 파일을 인스턴스한다.

③ 클라이언트가 응용 서버에 작업(visualization, mesh, analysis, ...)을 요구하면 응용 서버는 DB 서버에서 작업에 필요한 엔티티의 인스턴스 리스트를 요청하여 클라이언트가 요구한 작업을 수행한다.

④ 응용 서버의 계산이 끝나면, 클라이언트는 응용 서버에서 결과를 받을 수 있다.

⑤ 모든 작업이 성공적으로 수행되어 응용 서버에 저장 명령을 내리게 되면 응용 서버는 변경된 내용을 DB 서버에 넘겨준다. DB 서버에서는 작업 대상인 STEP 물리 파일에 변경된 내용을 갱신시켜서 저

장하고, 파일 리스트를 업데이트시킨다.

여기서 주목할 점은 서버와 클라이언트간의 연결이 직접 이루어지지 않고 ORB를 통하여 이루어 진다는 것이다. 즉, 클라이언트는 서버의 위치나 종류에 상관없이 ORB에 서비스를 요청하기만 하면 된다.

3.5. 개발 과정

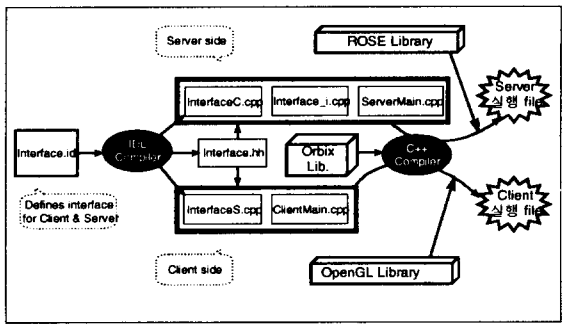

그림 6 DICCES 개발과정

프로그램의 개발 과정은 다음과 같다.

① 클라이언트와 서버간의 인터페이스를 IDL로 작성한 뒤 컴파일하면 헤더 파일, 서버 와 클라이언트 쪽 C++ 소스(IDL2C++) 파일이 생성된다.

② 클라이언트 IDL2C++ 파일에 GUI를 위한 라이브러리를 추가하여 클라이언트 프로그램을 완성한다.

③ 각각의 서버를 위한 IDL2C++ 구현파일을 작성한다.

④ AP 203 과 AP 209 EXPRESS 파일을 컴파일하여 C++ 소스(EXPRESS2C++) 파일을 생성한다.

⑤ 서버쪽의 IDL2C++ 파일, IDL2C++ 구현파일, EXPRESS2C++ 파일에 ROSE 라이브러리, Orbix 라이브러리를 추가하여 DB 서버를 완성한다.

⑥ 응용 서버의 기능을 구현하기 위한 코드에 서버쪽의 IDL2C++ 파일, IDL2C++ 구현파일, Orbix 라이브러리를 추가하여 각각의 응용 서버를 완성한다.

⑦ 완성된 응용 서버를 Orbix daemon에 등록시킨다.

전체적인 프로그램의 개발과정은 그림 6에 설명되어 있다. [10, 11]

3.6. 각 서버의 구현

본 연구에서 구현된 프로토타입 시스템은 DB 서버, 가시화(visualization) 서버, 메쉬(mesh) 서버, 해석(analysis) 서버 등으로 구성되어 있다.

① 가시화 서버

가시화 서버는 DB 서버로 부터 모델링 데이터를 얻은 후 이 데이터를 클라이언트에서 가시화하기 위한 데이터로 변환시킨 후 클라이언트에 전송하는 기능을 한다. 즉 STEP 물리 파일의 데이터가 클라이언트에서 가시화 될 수 있도록 IDL로 정의한 인터페이스에 맞게 데이터를 변환시키는 서버이다.

② 메쉬 서버

메쉬 서버는 DB 서버로부터 유한 요소로 분할하기위한 모델링 데이터를 얻은 후, 클라이언트로부터 입력된 메쉬 사이즈 정보를 통하여 요소를 생성한다. 본 논문에 사용된 요소의 생성 기법은 몇몇 상용 프로그램에서 주로 사용되고 있는 사상법(mapped element approach)을 사용하여 임의의 네 변으로 이루어진 해석 영역을 매개 변수 영역로 사상함으로써, 사변형의 유한 요소를 생성하는 방법을 사용하였다. [12]

③ 해석 서버

유한 요소 해석을 위한 해석 서버는 DB 서버로부터 분할된 유한 요소 데이터를 얻고 클라이언트로부터 입력된 경계 조건(하중, 변위)에 대한 정보를 사용하여 해를 구한다. 계산된 결과는 클라이언트의 요구에 의하여 DB 서버에 저장되거나 클라이언트에 다시 전송되어 보여진다. [12, 13]

3.7. 클라이언트의 구현

클라이언트는 각 응용 서버와 연결하여 사용자의 입력 정보를 연결된 응용 서버에 전달하는 역활과 응용 서버로부터의 메시지나 결과를 사용자에게 보여주는 역할을 한다. 이를 위해 보다 편리하고 명확한 사용자 인터페이스를 제공하여 이를 통해 입력된 정보를 응용 서버에서 사용되는 데이터로 변환시키는 기능과 연결된 서버로부터 전송된 데이터를 해석하여 디스플레이 해주는 기능을 갖는다. [14]

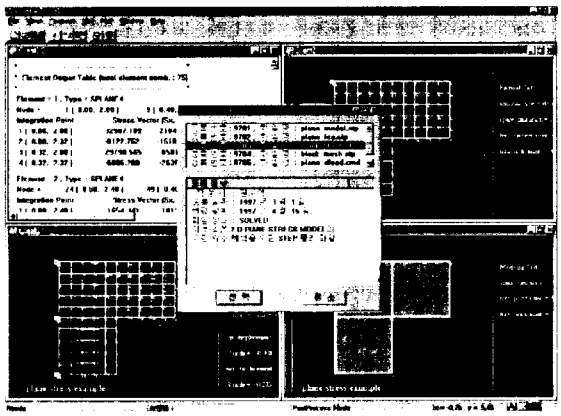

그림 7 DICCES 의 클라이언트

그림 7은 DICCES 환경에서 클라이언트가 구현된 예이며 중간에 위치한 다이얼로그 박스 윈도우는 DB 서버에 연결하여 사용자가 원하는 STEP 물리 파일을 선택할 수 있도록 한 것이다. 그림 7의 우측 하단에 위치한 윈도우로부터 반시계 방향으로 각각 가시화 서버에서 전송된 모델링 데이터를 보여주는 윈도우, 메쉬 서버에서 전송된 메쉬된 모델링 데이터를 보여주는 윈도우, 해석 서버에서 전송된 해석 결과를 텍스트로 보여주는 윈도우, 해석 서버에서 전송된 해석 결과를 그래픽으로 보여주는 윈도우를 보여 주고 있다.

4. 결론 및 향후 연구 방향

본 논문에서는 네트워크를 기반으로 하는 제품정보의 공유 기술과 분산객체환경 기술을 연구하고 DB 서버, 가시화 서버, 메쉬 서버, 해석 서버를 클라이언트를 통하여 이용하는 DICCES을 구현하였다. 여기서 중요한 점은 S/W나 H/W 환경의 통일에 의한, 혹은 기업별 자체 표준 DB의 구축에 의한 기업별 정보 공유의 개념이 아니라는 것이다. 생산 데이터의 저장 및 운용 방식에 있어서 국제 표준을 사용하고 모든 응용 프로그램은 이를 이용하는 것이다. 또한 DICCES에서는 새로운 STEP을 기반으로 한 응용 서버를 개발하여 기능을 확장시키는 것이 가능하다. STEP에 등록되어 있는 모든 AP에 대한 응용 서버를 개발하여 DICCES에 추가하는 것이 가능하므로 CAD/CAE 분야 이외의 산업 부문에도 이용이 가능하다.

이 시스템을 구성하는 각각의 요소는 모두 표준을 사용하게 되는데, 이는 기업활동에서 적어도 생산정보 공유의 기술적인 측면에서는 국가간의 경계도 허물 수 있는 가능성을 보여주는 것이다. 따라서 본 연구의 결과로 나오게 되는 각 표준에 대한 응용 기술 및 개별 기술의 통합에 의한 프로토타입 시스템의 구현은 미래의 생산정보 인프라 구축에 대한 인식의 확산에 기여함은 물론 앞으로의 이 분야에 대한 기술적인 토대를 제공하게 될 것으로 기대된다.

향후 연구 방향은 본 논문에서 제시한 DICCES의 응용 서버의 기능을 강화하고 적용 범위를 확장시키는 것이다. STEP 물리 파일에 근거한 DB 서버를 오라클과 같은 관계형 DB나 객체지향 DB로 교체하여 생산, 설계 정보를 기반으로 하는 DB 서버를 구축하는 연구도 필수적이다. 또한 자바(Java)를 기반으로 하는 클라이언트 소프트웨어 구현에 의해 웹 브라우저를(Web browser)를 이용하는 분산 환경을 구현하는 방안도 연구되어야 한다.

5. 참고 문헌

1. M. Hardwick et al., "Sharing Manufacturing Information in Virtual Enterprises", Communications of the ACM, Vol 39, pp. 46-54, 1996
2. M. Marache et al., "A CORBA-based infrastructure managing STEP distributed models for virtual reality applications", Proceedings of European Conference on Product Data Technology Days 1997, pp. 119-128, 1997
3. STEP 연구회, 제품 모델 정보 교환을 위한 국제 표준 (ISO 10303) 스텝, 성안당, 1996
4. ISO 10303-11 - Part 11: Description Method : EXPRESS language reference manual, ISO, 1994
5. ISO 10303-203 - Part 203: Application Protocol: Configuration controlled design, ISO, 1994
6. ISO 10303-209 - Part 209: Application Protocol: Composite and metallic structural analysis and related design, ISO, 1996
7. The Common Object Request Broker: Architecture and Specification 2.0, OMG, 1995
8. Jon Siegel, CORBA fundamentals and programming, John Wiley & Sons, Inc., 1996
9. 박재현, "분산객체 기술을 꿈꾸며, CORBA", 마이크로 소프트웨어, 1996년 3월호 ~ 1996년 6월호
10. STEP Programmer's toolkit - ROSE library reference manual, STEP Tools Inc., 1996
11. Orbix 2 programming & reference guide, IONA Ltd. 1995
12. 이원양, CAE 시스템을 위한 격자 생성기 및 열해석 모듈의 개발, 석사 학위 논문, 중앙대학교, 1997
13. K. J. Bathe, Finite element procedures, Prentice Hall, 1996
14. 이현찬 등, 컴퓨터 그래픽스 및 형상 모델링, 시그마 프레스, 1996