한국 CAD/CAM 학회 논문집
제5권 제 1호 2000년 3월 pp. 42-49

학술논문

실시간 원격 협력 설계 시스템 - CoDes

양상욱*, 최 영**

CoDes: A Real-Time Collaborative Design System

Yang, S.* and Choi, Y.**

ABSTRACT

This paper presents a solid modeling system that enables real time co-operative modeling and discussion on WWW between geographically separated designers. The modeling system is implemented using CORBA architecture and integrated in WWW with Java technology. The server objects use a commercial solid modeling kernel to provide modeling features and access database including product model data. The client is implemented using Java2 platform so that enables end-users access the system with web-browsers.

Key words : CORBA, Java, ObjectWeb, Web Collaboration, Solid Modeling.

1. 서 론

제품의 생산 과정에 있어서 개발 및 생산 기간의 단축과 품질의 향상은 기업의 경쟁력과 직결된다. 또한 공학 분야의 대부분의 제품은 하나의 기업이나 부서에서 개발 및 생산의 전 과정이 수행되는 것이 아니므로 기업, 부서 및 작업자간 원활한 의사와 정보의 교환은 기업경쟁력 강화를 위한 필수적인 것이라고 할 수 있다.

현재 지리적으로 분산된 부서, 기업 또는 작업자간 데이터 교환을 위해서 가장 널리 쓰이고 있는 것이 인터넷이다. 인터넷은 정보의 공유에 있어서 문제되는 공간적인 제약을 극복할 수 있는 가장 저렴하면서도 효과적인 도구이며, WWW을 통해서 각각의 작업자들에게 플랫폼에 독립적으로 일관적인 사용자 인터페이스를 제공할 수 있다. 인터넷은 이러한 이점과 함께 자바와 같은 인터랙션 가능한 기술에 힘입어 정보 공유의 가장 효과적인 도구 중 하나로 인식되고 있으며, 생산 과정에서 이를 이용하는 연구도 활발히 진행되어왔다[1].

웹을 이용한 엔지니어링은 초기에 제품 데이터의 형상을 정지된 이미지나 VRML모델로 변환하여 웹에 게시하는 단계부터 시작하였지만 현재에 이르러서는 설계 단계에서부터 3D 모델을 공유하거나, 모델의 검토 작업을 애니메이션으로 공유한다든지 프로젝트의 관리 등을 가능하게 하는 다양한 상업용 소프트웨어들이 발표되는 등 보다 복잡 다양한 분야에 응용되는 단계에 있다[2].

또 한편으로는 급변하는 컴퓨팅 환경에서 정보의 공유 및 공동 작업 뿐 아니라, 하드웨어 및 소프트웨어의 유지 보수에 드는 비용을 줄이기 위하여 소프트웨어를 공유하는 시도가 진행중이다. 작업에 필요한 소프트웨어를 작업자의 시스템에 설치하는 것이 아니라 필요한 클라이언트 소프트웨어를 실시간으로 다운로드 받아 사용한 시간 만큼의 비용을 지불하게 하는 이 방식은 고가의 CAD 소프트웨어를 구입하기 힘든 중소 기업에 특히 매력적이며 소프트웨어의 유지, 보수를 위한 별도의 비용과 노력을 필요로 하지 않는다.

본 논문에서는 앞서 소개한 기술들을 이용하여 웹상에서 지리적으로 분산되어 있는 개발자간 협력 설계가 가능한 솔리드 모델링 시스템의 구현에 대하여 논의한다.

구현된 시스템은 ObjectWeb[3] 기술을 이용하여 구현되었으며, 솔리드 모델링 커널을 포함하여 모델링 기능과 사용자간 의사 교환을 중재하는 서버 객체들

*중앙대학교 기계설계학과
**종신회원, 중앙대학교 기계공학부

과 실시간으로 다운로드 되어 사용자 인터페이스를 제공하게 되는 클라이언트로 구성되어 있으며, CORBA(Common Object Request Broker Architecture) 를 통하여 상호 작용하게 된다.

본 연구에서 개발된 이 협동 설계 시스템을 CoDes (COllaborative DEsign System)라 명명하였다.

2. 분산 협력 설계 기술

분산된 작업자간의 협력 모델링은 이미 존재하는 제품 모델이나 새로 만들어진 모델에 대한 문제점 발견 단계와 이를 해결하는 단계로 나눌 수 있다. 문제점의 발견 단계에서 여러 작업자간 의견을 개진하고 이를 조율하는 일이 필요한데 이를 위해서 뷰(view) 데이터와 마크업(mark-up) 데이터를 공유할 수 있어야 하고, 의견 교환을 하기위한 채팅이나 화상회의 기능이 필요하다. 이러한 과정을 통해 발견된 문제를 해결하는 단계에서는 모델을 수정할 수 있는 기능이 필요한데, 이러한 기능은 블렌딩, 테이핑 등 솔리드 모델러가 제공해야 하는 기능이 되며, 모델 데이터에 대한 BOM(Bill Of Material)정보 등도 함께 다룰 수 있어야 한다.

이 절에서는 이러한 기능들을 제공하는 시스템을 구현하기 위한 기본적 아이디어를 소개한다.

2.1 뷰 데이터의 실시간 공유

3D 뷰를 제공하는 응용프로그램은 데이터를 화면에 나타내기 위하여 내부적으로 몇 가지 좌표변환을 수행하여 모델을 화면에 디스플레이하게 된다[4]. 특히 모델의 회전, 이동, 확대/축소 같은 동적인 뷰를 제공하려 한다면 사용자의 마우스나 키보드의 입력이 있을 때마다 변환 행렬을 재구성하고 이를 이용하여 렌더링(rendering)하게 된다. 본 연구에서 사용된 뷰 공유에 대한 기본적인 아이디어는 사용자 입력에 의해 변환행렬이 갱신되어야 할 때 이 변환행렬을 CORBA의 IDL(Interface Definition Language)을 통하여 다른 사용자와 공유하는 것이다. 즉, 한 사용자가 뷰를 변경하였다면 이 사용자의 프로그램에서는 새로운 뷰를 위한 변환행렬을 서버에 넘겨주고 서버는 새로운 변환행렬을 다른 모든 사용자들에게 전달하여 뷰를 갱신하게 함으로써 모든 사용자가 동일한 뷰를 공유할 수 있게 된다. 본 연구에서는 사용자 측의 렌더링을 위해서 Java 3D를 사용하였는데, Fig. 1에 Java3D에서의 사용자에 의한 뷰 변환 메시지와 전체적인 scene graph의 관계가 나타나 있다[5].

그림에서 굵은 테두리로 표시된 변환 그룹(Transform Group; TG)이 관심의 대상이 되는 모델의 변환행렬을 포함하게 되는데 Java3D는 마우스 조작에 의한 회전, 이동, 확대/축소를 변환 그룹에 연결하는 API와 변환 그룹의 변화가 있을 때 이를 가로채게 할 수 있는 API를 제공하므로 마우스에 의한 변환행렬의 변경을 쉽게 다룰 수 있다. 랜더러를 직접 구성하거나 OpenGL을 사용하는 등의 프로그래머가 뷰의 변환행렬 작업을 다루어야 하는 경우는 여러 사용자가 같은 뷰를 보려 한다고 할지라도 각자 화면의 해상도와 종횡비는 다를 수가 있으므로 프로젝션 변환행렬을 제외한 모델뷰 변환행렬만이 공유의 대상이 된다.

이렇게 변환행렬만을 공유하는 방법은 4×4 행렬의 각 셀을 단정도 실수형으로 표현하면 뷰 변경이 한 번 일어날 때 64 바이트의 데이터만 이동하게 되므로 네트워크에 주는 부하가 매우 적어 빠른 반응속도를 기대할 수 있다.

2.2 솔리드 모델러의 분산화

Fig. 2는 하나의 컴퓨터에서 독립적으로 실행되는 일반적인 솔리드 모델링 어플리케이션의 구조를 간략하게 나타내고 있다. 일반적인 솔리드 모델링 어플리케이션은 대화형의 사용자 인터페이스를 제공하게 되는데 사용자로부터 입력 받은 모델링 오퍼레이

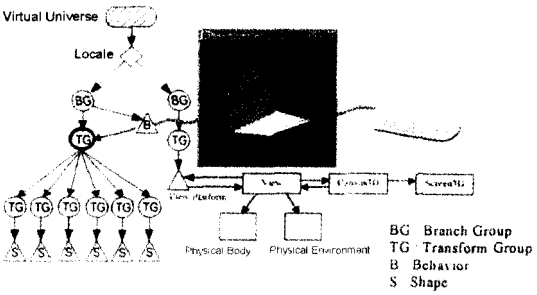

Fig. 1. An example of scene graph of Java3D.

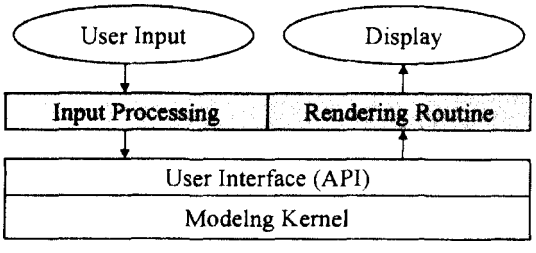

Fig. 2. A standalone solid modeling application.

션을 하부 구조의 커널이 받아들일 수 있는 오퍼레이션으로 변환하여 전달하고, 그 결과에 대한 형상 데이터를 커널로부터 얻어 화면에 나타내게 된다.

이러한 모델링 어플리케이션을 분산 환경에 적용할 수 있도록 한다면 우선적으로 고려되어야 할 사항은 공유될 정보와 그렇지 않을 정보의 구분과, 어느 선에서 분산화가 이루어졌을 때 네트웍의 트래픽(network-traffic)을 최소화 할 것인지에 대한 결정이다.

모델 데이터와 모델링 오퍼레이션에 관한 정보는 모델링 커널(kernel) 내에서 관리되므로 커널은 서버 측에 위치하여야 한다. 또한 모델에 대한 정보를 탐색한다든지 하는 등의 모델에 직접 수정을 가하지 않는 오퍼레이션은 공유될 필요성이 적으므로 사용자의 입력 처리부분은 클라이언트에 위치하는 것이 바람직하다. 모델의 디스플레이를 위한 부분은 모든 사용자에게 제공되어야 하므로 당연히 클라이언트는 모델의 디스플레이 기능을 포함하여야 하지만, 이 경우는 모델의 렌더링을 위한 과정의 여러 단계 중 어느 단계를 기준으로 서버 쪽과 클라이언트 쪽으로 나눌 것 인지가 고려의 대상이 된다. 예를 들자면 렌더링된 이미지를 비트맵으로 공유할 것인지, 삼각화된 데이터를 공유할 것인지 등의 문제가 될 수 있는데, 정지된 뷰를 여러 사용자의 클라이언트에 디스플레이 하는 경우 서버측에서 특정 뷰에서 바라보는 모델 형상의 렌더링된 이미지를 GIF나 JPG같은 포맷으로 클라이언트에 전송해 주는 것이 효과적인 방법이 될 수 있지만, 실시간으로 변하는 뷰를 공유하려고 한다면 매번 서버는 렌더링을 위한 여러 과정을 수행하고 이미지를 전송하여야 하므로 실시간 작업을 위한 속도를 기대하기 힘들다. 각각의 클라이언트가 서버로부터 삼각화 된 데이터를 받아 가지고 있는 경우에는 앞 절에서 설명한 바와 같이 최소의 데이터로 동적인 뷰 공유를 구현할 수 있으며 이때의 렌더링 속도는 거의 클라이언트 시스템의 성능에만 영향을 받게 된다. 그러나 이 경우에도 모델의 수정이 일어날 때마다 클라이언트는 서버로부터 데이터를 받아와야 하는데 이 때 각각의 클라이언트의 네트웍 접속 요건과 시스템의 성능을 고려하여 삼각화의 디테일을 조절함으로써 삼각형 데이터 생성과 전송 속도의 향상을 꾀할 수 있다.

또한 고려되어야 하는 중요한 사항은 여러 명의 사용자가 동시에 협력 설계를 수행하는 경우, 각 사용자가 작업중인 모델에 대해 적용하려는 오퍼레이션이 서로 상충하거나 우선 순위가 모호하게 되는

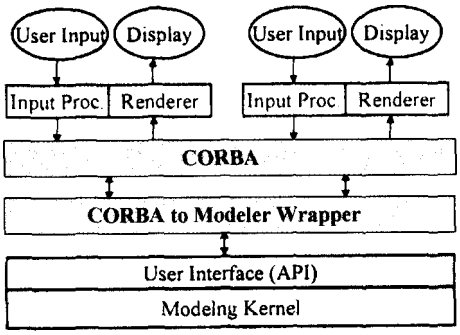

Fig. 3. A solid modeling system on distributed environment

경쟁 조건(race conditon)이 나타날 수도 있다는 것이다. 따라서 본 연구에서는 작업중인 모델은 서버 측의 모델링 커널만이 직접 다루고, 사용자들은 서버가 제공하는 서비스를 통해 이를 사용함으로써 이러한 문제점을 줄이고자 하였다. 현재 커널에서 작업중인 데이터에 대해 누군가가 조작을 시도하였다면 모델링 커널은 오류나 예외를 발생시키고, 서버는 그러한 조작을 시도한 사용자에게 그 내용을 전달하고 해당 오퍼레이션만 취소함으로써 현재 조작중인 형상이나 위상 데이터를 잠그는(locking) 기능을 구현할 수 있다. 물론 하나의 파트를 동시에 다루는 경우라도 한 사람이 조작하고 있는 형상이나 위상데이터의 영향을 받지 않는 형상과 위상데이터는 다른 사람에 의해 동시에 다루어질 수 있어야 한다.

Fig. 3은 이러한 조건들을 고려하여 모델링 시스템을 분산환경에 적용한 예를 나타낸다.

3. 협력 설계 시스템 구현

3.1 구현 시스템의 구조

Fig. 4는 서버와 클라이언트를 구성하는 요소들을 보여주고 있다.

서버 객체 구현부는 C++로 구현되어 있는데, 서버 객체는 클라이언트를 관리하고 솔리드 모델링 커널을 둘러 싸 CORBA에 대한 wrapper의 역할을 하는 세션 객체와 모델링 과정에서 사용되는 여러 객체들, 파라솔리드 커널로 이루어져 있으며 설계 데이터와 부가적인 정보를 표현하기 위한 주석(annotation)을 저장하기 위해서 데이터베이스를 사용한다.

클라이언트는 Java3D와 swing 패키지를 이용한 사용자 인터페이스 부분과, 서버의 질의에 응답하기 위한 콜백 객체, ORB(Object Request Broker) 커넥

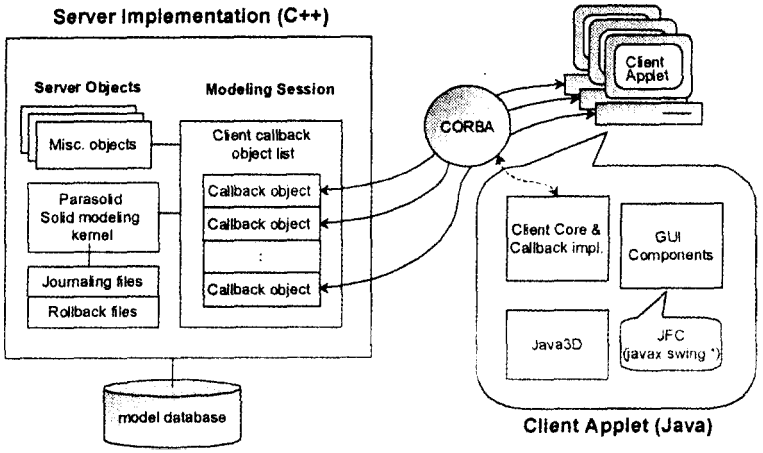

Fig. 4. Modeling system for distributed collaborative design.

션을 담당하는 코어 부분 등으로 이루어져 있다.

Fig. 5는 구현된 시스템을 위한 IDL의 소스 코드 중 exception, 타입, 인터페이스 만을 나타낸 것이며 각각의 상세한 속성은 생략되어 있다.

CoDes 시스템에서 실제의 모델링은 모두 서버 쪽에서 이루어지며 위상 정보 또한 서버쪽에 있다. 클라이언트 측에서는 3D 형상을 사용자에게 보여주고 이러한 형상의 페이스(face)나 에지(edge) 단위로 사용자의 선택을 받기 위한 최소한의 데이터, 즉 삼각형 집합과 페이스 또는 에지의 인덱스만을 필요로 하며 좀 더 세부적인 정보가 필요한 경우에만 서버 측에 질의하여 부가적 데이터를 얻어낸다.

클라이언트는 ORB를 통하여 세션객체를 참조함으로써 서버 객체에 특정 데이터나 질의 또는 오퍼레이션을 요구할 수 있으며 서버가 제공하는 대부분의 서비스는 세션을 통하여 제공된다. 또한 서버측에서 클라이언트에 오퍼레이션을 요구할 경우 클라이언트가 그러한 요구에 대한 응답을 할 수 있어야 하는데, 이것을 가능하게 하는 것이 콜백(call-back) 메커니즘이다. CDCallback은 콜백 객체를 정의하기 위한 인터페이스이며, 콜백 객체는 일반적인 CORBA 객체들과는 달리 그 구현이 클라이언트 쪽에서 이루어진다. 한 사용자가 데이터에 변경을 가하는 오퍼레이션을 한다거나, 컨퍼런스를 위하여 뷰를 변경하는 경우, 마크업이나 대화를 위한 메시지를 발생 시키는 경우 등은 협력 설계에 참여한 다른 사용자에게 그 내용이 전달되어야 하는데, 이때 서버는 자신이 참조한 콜백 객체들에게 이러한 내용을 전달함으로써 모든 사용자가 실시간으로 동일한 상태를 유지하게 된다.

```
module caucad {
    module codes {
        //exceptions
        exception PKException;
        exception CDException;

        // interfaces definition for types
        typedef double CDTVector3d[3];
        typedef double CDTInterval[2];
        typedef float CDTMatrix4f[16];
        interface CDTAxis2Place;
        interface CDTFacet3f;
        interface CDTPolylineData;

        // interface for model information
        interface PartInformation;

        // server objects
        interface CDCallback;
        interface CDServer;
        interface CDSession

    }; // end of module codes
}; // end of module caucad
```

Fig. 5. Interface definition for modeling system.

예를 들어서, 세션의 참여자가 하나의 블록을 생성하였다면 클라이언트는 블록 생성을 위한 세션의 서비스를 호출하고 세션은 모델링 커널을 이용하여 블록을 생성한 후 이를 가시화하기 위한 faceted triangle 데이터를 생성하고, 뷰 모델이 업데이트 되었다는 정보를 클라이언트에 통보한다. 업데이트 메시지를 통보 받은 클라이언트는 세션에 가장 최근의 삼각형 데이터 및 와이어프레임 데이터를 요구함으로써 사용자의 화면에 모델링중인 데이터를 보여줄 수 있다. 이 때 하나의 클라이언트로부터 모델링 오

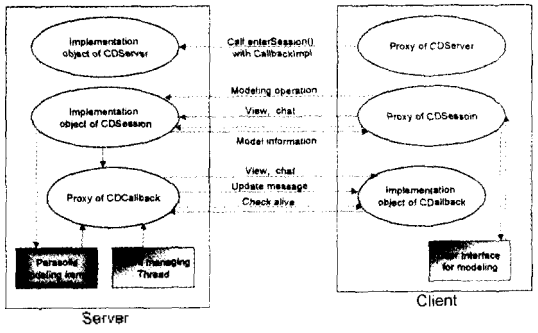

Fig. 6. Operations between session and callbacks.

퍼레이션을 받아들이더라도 모든 클라이언트에 업데이트 정보를 전달함으로써 세션에 참여한 모든 클라이언트가 실시간으로 설계중인 정보를 공유할 수 있게 한다. Fig. 6은 이러한 콜백 객체와 세션 오퍼레이션 및 데이터의 상호 작용을 나타낸 그림이다. 세션은 세션에 참여한 클라이언트의 콜백 객체를 참조하고 있으며 클라이언트는 자신이 참여한 세션 객체를 참조하고 있다. 그림에 나타난 proxy 객체들은 실제의 구현은 네트워크 건너편에 있는 객체이지만 로컬 플랫폼에 있는 것처럼 사용되며 실제 객체의 호출(invocation)은 ORB에 의해 사용자에 투명하게 일어난다[6].

서버 객체는 클라이언트의 로그인 및 로그아웃과 콜백 객체들을 관리하는 기능을 가지고 있고, 모델링에 필요한 모든 기능은 세션 객체가 제공하고 있다. 세션 객체의 메쏘드는 모델러를 제어하기 위한 메쏘드와 뷰공유, 채팅등의 컨퍼런스 관련 메쏘드, 그리고 데이터 베이스로부터 파트들의 정보를 얻거나 저장하는 메쏘드들로 이루어져 있다. 솔리드 모델링 커널과 직접 연결되는 메쏘드들은 모델링 오퍼레이션의 성공 여부를 클라이언트에 전달하여야 하는 것은 물론이지만 잘못된 모델링 오퍼레이션이 실행되려고 한 경우 그 내용을 리포트 할 수 있어야 하므로 이를 위한 예외(Exception) 객체가 정의되고 커널에서 에러 발생시 클라이언트에 전달된다. 또한 컨퍼런스 관련 메쏘드 들은 동시에 여러 사용자에 의해서 발생 될 수도 있으므로 클라이언트-서버간의 고착(deadlock)을 피하기 위하여 단방향으로 정의되었다. 삼각화된 데이터와 같이 데이터의 이동에 많은 시간이 걸릴 수 있는 기능은 모델이 업데이트 되었을 때 서버는 이러한 업데이트 되었다는 메시지만을 단방향 메쏘드를 통해 클라이언트에 전달하고 각각의 클라이언트가 별도의 쓰레드를 통하여 데이터를 전달 받음으로써 고착상태를 방지할 수 있다.

3.2 구현 환경 및 도구

CORBA 를 이용한 소프트웨어의 개발에 있어서 가장 먼저 이루어지는 것은 IDL로 객체간의 상호 작용을 정의하는 것이다[7]. 네트워크상에서 작동하는 어플리케이션이 아닌, 독립적으로 실행되는 어플리케이션을 개발한다 할지라도 각각의 모듈간 인터페이스를 정의하는 것이 일반적이다. 모듈정의가 명확하게 이루어 지면 이를 IDL로 기술하기가 용이해 진다. 본 시스템의 IDL은 솔리드 모델링 커널의 API wrapper와 사용자의 로그인/로그아웃을 관리하기 위한 인터페이스를 중심으로 클라이언트와 서버가 상호 작용하게 된다.

작성된 IDL은 서버 스켈리턴(skeleton)과 클라이언트 스텁(stub)으로 컴파일되는데 이들은 각각의 구현을 위한 언어로 표현된다. 본 시스템에서 C++ 서버 스켈리턴을 생성하기 위해서 Iona Technology사의 Orbix 3.0.1의 IDL 컴파일러를 사용하고, 자바 클라이언트 스텁을 생성하기 위해서 같은 회사의 OrbixWeb 2.0.1의 IDL 컴파일러를 사용하였다.

서버 구현부는 솔리드 모델링 커널로 Unigraphics Solution사의 Parasolid v11.0.114를 사용하여, Orbix 3.01의 ORB 라이브러리와 함께 MicroSoft Visual C++ 6.0 상에서 컴파일 되었다.

클라이언트는 Sun Microsystem사의 JDK 1.2.2와 Java3D 1.1.1을 사용하여 Java platform 2로 작성되었다. 개발된 클라이언트 애플릿은 OrbixWeb의 ORB 클래스들과 함께 웹상에 게시된다.

4. 협력 설계의 수행 예

4.1 협력 설계 시나리오

협력 설계를 위하여 구현된 시스템을 이용한 작업 시나리오는 일반적인 ObjectWeb어플리케이션이 작동하는 방식으로 실행된다. 사용자가 웹브라우저로 클라이언트 애플릿을 포함하는 서버에 접속하여 웹 문서를 브라우징 하면 클라이언트가 다운로드 되어 실행되고, 서버에 연결할 수 있다. 최초로 서버에 연결한 클라이언트의 요구에 의해 ORB는 서버 프로그램을 실행시켜서 세션 객체 등의 구현 객체들을 활성화 하고, 세션에 참여하게 된다. 세션이 시작되면 모델러가 초기화 되며, 이 때부터 시스템이 제공하는 모델링 기능이 사용 가능하게 된다. 첫 사용자가 세션에 참여한 이후 다른 사용자가 세션에 참여

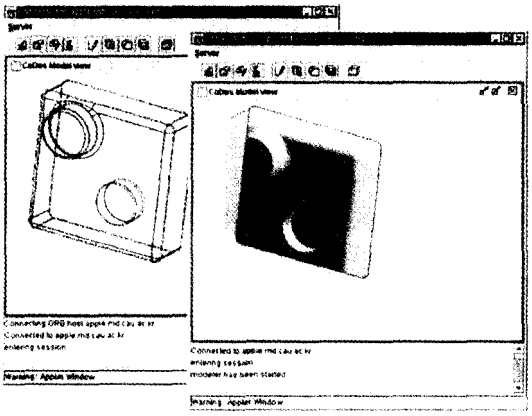

Fig. 7. An example of CoDes client.

하려고 하면, 세션의 개설자는 그 사용자를 참여시킬 것인지를 결정할 수 있고, 개설자의 허가 하에 다른 사용자가 참여할 수 있다.

Fig. 7은 실행 중인 클라이언트 애플릿의 예이며 렌더링된 파트는 구현된 시스템 기능 중 face sweep과 edge blending 등을 이용하여 모델링되었다. 실제의 데이터는 서버의 데이터베이스나 모델링 커널에 있으며 한 사용자가 변경한 파트는 클라이언트 콜백 메커니즘에 의해 즉시 모든 사용자의 클라이언트 뷰에 반영된다. 오른쪽과 왼쪽 창의 타이틀 바에 나타나 있는 정보와 같이 각 창은 다른 사용자가 로그온하여 작업하는 것이며, 그림에는 같이 나타나 있지만 각각의 창은 분산된 다른 시스템에서 실행 되는 별도의 애플릿 인스탄스이다. 한 명은 셰이딩된 모델로, 다른 한 명은 와이어 프레임으로 보고 있지만 모델의 회전된 각도와 위치를 보면, 현재 뷰가 공유되고 있음을 알 수 있다.

4.2 협력 모델링 수행 과정

이 절에서는 Fig. 8과 같은 간단한 셀 모델에 대하여 두 명의 사용자가 협업하여 모델링 작업을 하

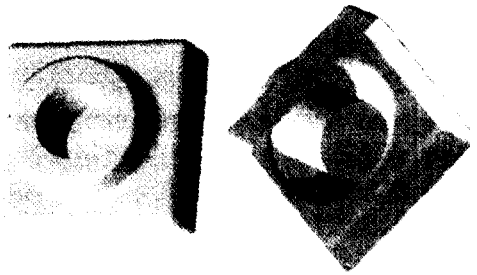

Fig. 8. A test part for cooperative modeling example.

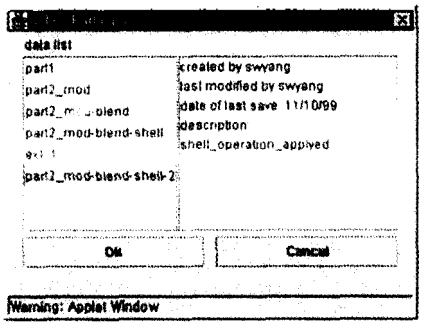

Fig. 9. Dialog box for reading part data from data repository.

는 예를 통하여 구현된 모델링 시스템의 기능과 동작 방식에 대하여 설명한다.

이 예는 사용자 A와 사용자 B가 동시에 모델링 시스템에 접속해 있고, 데이터 저장소로부터 하나의 단순 블록을 읽어 들인 후 Fig. 8에 나타난 모델을 생성하는 과정을 설명한다. 설명의 편의상 번갈아 작업 하는 것으로 예를 들며, 실제로 설명하는 작업들은 두 사용자 중 어느 누군가가 혼자 하였을 때에도 결과는 마찬가지이며 모델링의 전 과정은 모든 사용자에게 공유된다.

Fig 9는 서버측에 저장된 모델들을 모델링 시스템 안으로 읽어들이기 위한 대화 상자이며 왼쪽에는 현재 서버에 저장된 모델의 이름이 리스트로 나타나 있고 오른쪽 창에는 리스트에서 선택된 모델에 대한 정보가 나타나있다. 한 사용자가 여기서 모델을 선택하게 되면 선택된 모델은 모델러 안으로 읽혀 들어오고 모든 사용자의 뷰에 선택된 모델이 나타나게 된다.

Fig. 10에는 단순 블록으로부터 Fig. 8에 나타난 모델을 생성하는 과정이 나타나 있는데 사용자 이중 1, 3, 4, 6번의 화면은 사용자 A 측에서, 2, 5번은 사용자 B측에서 캡쳐한 것이다.

화면 1에서 하나의 블록이 있고, 블록의 전면에 하나의 원이 imprint 된 것이 보인다. 이 상태에서 사용자 A가 이 원을 extrude 하기 위하여 대화상자에 필요한 데이터를 입력하고 있다. 화면 1에서 나타난 오퍼레이션의 결과로 화면 2에서는 블록의 가운데 원이 돌출되어 있는 것을 볼 수 있으며 사용자 B가 원기둥의 윗 면에 작은 원을 imprint하고 이를 안쪽으로 extrude 하기 위하여 데이터를 입력하는 것을 볼 수 있다. 화면 3은 이 결과가 사용자 A의 클라이언트에 나타난 것이다.

화면 4에는 사용자 A가 와이어프레임 뷰로 화면

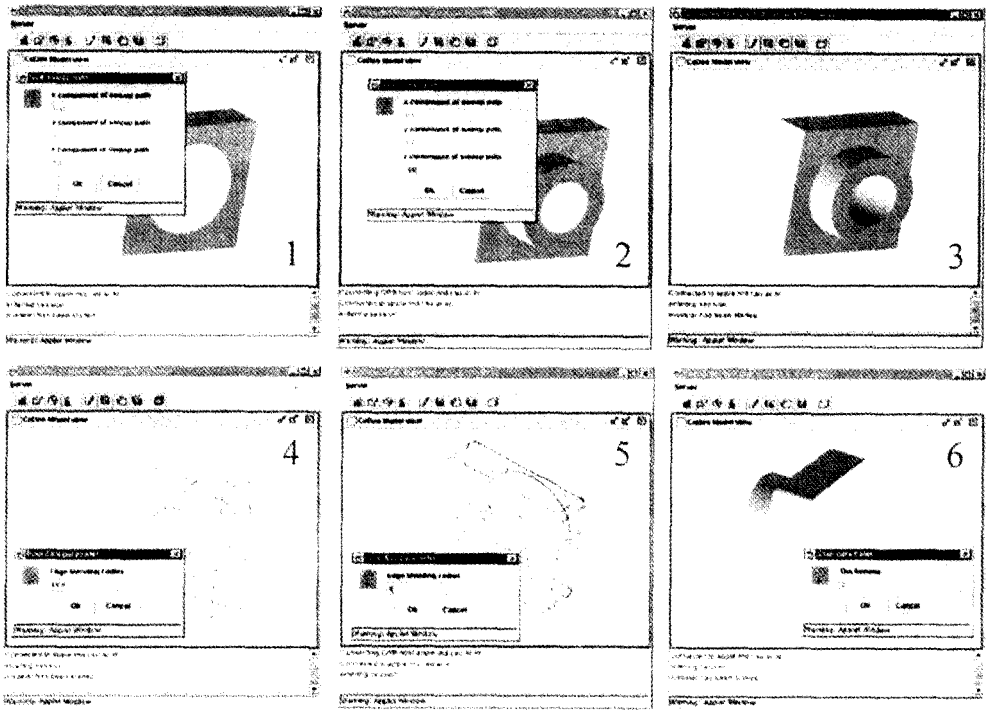

Fig. 10. An example of real-time collaborative design.

모드를 바꾸고 블록의 바깥쪽 에지들을 선택하여 라운딩 지름을 주기 위하여 대화상자에 입력하는 장면이 나타나 있다. 화면 5는 사용자 B가 화면 모드를 와이어프레임 뷰로 바꾸고 원기둥의 상단 부분에 라운딩 지름을 주려고 하는 장면이다. 이때의 모델은 와이어 프레임으로 나타나 있으나 화면 4의 결과로 블록의 바깥쪽이 라운딩 되어있음을 볼 수 있다. 화면 6에서는 사용자 A가 제거될 면(뒷면)을 선택하고 두께를 주어 Shell을 만들고 있다. 이의 결과로 Figure 8의 모델이 생성된다.

이상과 같은 방법으로 하나의 제품 모델에 대한 협력 모델링 과정을 수행 해 보았다. 설명에는 언급하지 않았지만 이 예의 각 단계 중 언제든지 모델을 저장할 수 있다. 모델의 저장은 Parasolid의 transmit 기능에 의해 이루어지며 부가정인 정보는 별도로 데이터베이스화 되어 서버측에 저장된다. 위의 예는 LAN으로 연결된 두 대의 WindowsNT PC에서 수행되었으며, 네트워크 부하를 고려하지 않고 있어 각 오퍼레이션의 수행 속도는 독립 실행형의 모델링 시스템과 거의 차이가 없다. 이러한 사실에 근거하여 원격 사이트간 모델링 작업을 수행 할 때 오직 네트워크의 속도가 사용자와 시스템간의 반응속도를 결정하게 되는 요소로 작용하게 될 것임을 추측할 수 있다.

현재까지 구현된 모델링 기능은 위에서 예시한 기능들이 거의 전부라고 할 수 있으나, 그 외의 모델링 기능들은 구현된 시스템의 프레임워크의 변경 없이 쉽게 추가될 수 있으며 그럼으로써 실용적인 솔리드 모델링 시스템으로 확장 될 수 있다.

5. 결 론

본 논문에서는 급변하는 현재 및 미래의 생산환경과 시장의 요구에 빠르게 부응하고 기업 경쟁력 향상을 꾀하기 위한 방법으로써 인터넷/인트라넷에서 적용 가능한 여러 기술들과 제품의 생산 및 설계정보 공유를 위한 기반 기술에 대한 연구와 그 결과로 제품 개발 초기단계에서부터 적용할 수 있는 원격 협력 설계 시스템의 프로토타입 구현에 대해서 논의하였다. 구현된 시스템에서는 웹브라우저만 있으면 플랫폼에 관계없이 클라이언트 어플리케이션이 일관성 있게 실행되게 하였다. 자바기술을 도입함으로써 서론에서 언급한 바와 같이, 사용자의 입장에서는 고가의 솔리드 모델러를 구입, 설치하지 않아도 웹브라우저 만으로 구현 시스템을 사용 할 수 있게 하였고, CORBA 기술의 적용으로 분산된 여러 사용자

간의 정보 공유를 실시간으로 가능하게 하였다.

현재 구현된 모델링 기능은 sketch, extrusion, edge blending과 shell 등으로 실용적인 제품 모델을 설계하기에는 부족하지만 이미 구현된 프레임워크 내에서 기능 확장을 통해 실용적인 모델링 시스템으로 전환 할 수 있을 것으로 판단된다. CoDes는 독립적인 모델러의 기능을 제공하는 서버 시스템과 클라이언트로 구성되어 있으며 자체적으로 설계한 인터페이스에 의해 상호간의 인터랙션을 수행하지만 추후, 현재 제정중인 STEP part 26, SDAI-IDL의 지원에 대한 연구를 통해 표준화된 객체로 재구성 할 수 있어야 하며, OMG에서 제정하는 CORBA Manufacturing specification[8]을 적용함으로써 CORBA의 수직적 기능(vertical facility)으로 사용된다면 해석, 최적화 설계 등의 기능을 지원하는 다른 표준 CORBA 객체와 함께 작동할 수 있을 것이다. 또한 본 연구에서는 고려되지 않았지만 실제의 설계 프로세스에 이러한 시스템이 사용되게 된다면 모델이나 작업 단위에 대한 보안에 대해서도 고려하여야 할 것이다.

본 연구를 통하여 개발한 기술은 인터넷/인트라넷을 이용한 PDM 구축에 효과적으로 활용될 수 있을 것으로 기대 되며, 원격 회의, 동시공학 등의 분야 다각도로 활용될 수 있을 것으로 기대된다.

감사의 글

이 논문은 1999학년도 중앙대학교 학술연구비 지원에 의해 수행되었습니다. 이에 관계자 여러분께 감사드립니다.

참고문헌

1. Potter, C., "Web-Enabled Engineering: step-by-step", *Computer Graphics World*, pp. 64-69, Nov., 1997.
2. Kempfer, L., "Tools for Web Collaboration", *Computer-Aided Engineering*, 1999, April.
3. Orfali, R., Harkey, D., and Edwards, J., "CORBA, Java, and the Object Web", *Byte magazine*, http://www.byte.com/ art/9710/sec6/ art3.htm, Oct., 1997.
4. Anand, V., B. *Computer Graphics and Geometric Modeling for Engineers*, John Wiley & Sons Inc., 1993.
5. Bouvier, D.J. and Dankwardt, K., P., *Getting Started with the Java 3D™* API, K Computing, Feb. 1999.
6. Iona Technologies, *OrbixWeb™ Programming Guide*, Iona Technologies, 1996.
7. Siegel, J., *CORBA' fundamentals and programming*, John Wiley & Sons, Inc., 1996.
8. OMG, CORBAmanufacturing: *Manufacturing Domain Specifications*, Object Management Group Inc., Oct, 1999.

양 상 욱

1998년 중앙대학교 기계설계학과 학사
2000년 중앙대학교 기계설계학과 석사
관심분야: network-enabled CAD, solid modeling

최 영

1979년 서울대학교 기계설계학과 학사
1981년 한국과학원 생산공학과 석사
1989년 Carnegie Mellon University 기계공학과 박사
1981년~1984년 대우중공업 중앙연구소 연구원
1989년~1990년 Engineering Design Research Center 연구원
1992년~현재 중앙대학교 기계설계학과 교수
관심분야: 네트워크 CAD, Non-manifold modeling, 솔리드모델링, 곡선 및 모델링