협력시스템에서 3D 스튜디오 맥스 플러그인 설계 및 개발

(Design and Implementation of 3D Studio Max Plug-In in Collaborative Systems)

권 태 숙 † 이 승 룡 ††
(Taisook Kwon) (Sungyoung Lee)

요 약 협력 시스템은 3D 애니메이션, 컴퓨터 게임, 산업디자인 제작과 같은 애플리케이션을 원격지에서 하나의 가상공간을 통해 공동작업을 수행할 수 있는 환경을 제공한다. 본 논문에서는 단일(stand-alone) 시스템에서 작동되는 Kinetic사의 3차원 비주얼 모델링 도구인 3D 스튜디오 맥스를 분산 시스템에서 작동 할 수 있게 확장시켜 다수의 사용자가 공동으로 3D 모델링 작업을 수행할 수 있도록 환경을 제공한 개발 경험을 기술한다. 본 논문에서는 3D 스튜디오 맥스 플러그인 SDK(Software Development Kit)를 사용하여 분산 협력 시스템에서의 3D 객체 공유 플러그인 개발에 대하여 다루었는데, 이는 모델링 데이타의 공유를 위해 3D 객체 정보를 추출하여 공유 메모리에 쓰는 기능과, 공유메모리로부터 3D 객체정보를 읽어와서 3D 객체를 생성하는 기능을 제공한다. 그리고, 협력 시스템 클라이언트와 3D 스튜디오 맥스간에 데이타 교환을 위한 방법으로 공유 메모리를 사용함으로써 가변적인 크기를 갖는 3D 객체의 저장이 용이하도록 하였다. 또한, 3D 스튜디오 맥스에서 제공되는 객체를 구성하는 데이타를 분석한 후 클라이언트가 요구하는 최소한의 데이타만을 추출함으로써 전송되는 불필요한 공유 데이타의 양을 줄일 수 있어 네트워크의 부하를 감소시켰다. 플러그인의 개발로 인하여 단일 컴퓨터 시스템에서 수행되는 3D 스튜디오 맥스 작업을 분산 환경에서 작업할 수 있게 확장시킴으로써, 3D 모델링 작업 시 공간과 시간의 제약을 최소화하여 경제적 이윤 창출 효과가 클 것으로 기대된다.

Abstract Collaborative systems allow users, who may be far removed from each other geographically, to do collaborative work such as 3D animation, computer game, and industrial design in a single virtual space. This paper describes our experience to develop a collaborative system framework that aims at expanding the some functions of a stand-alone visual modeling tool, called 3D Studio Max, into those of the distributed collaborative working environments. The paper mainly deals with design and implementation of a 3D shared-object Plug-In with respect to the 3D Studio Max Plug-In Software Development Kit in the distributed collaborative system developed by the authors. There are two major functions of the proposed scheme; one is to write 3D object-information to the shared memory after extracting it from the 3D Studio Max, the other is to create 3D objects after retrieving them from the shared memory. Also, the proposed scheme provides a simple way of storing 3D objects that have variable size, by means of shared memory which located in between the collaborative system clients and 3D studio Max. One of the remarkable virtures of the Plug-In is to reduce a considerable amount of shared object data which in consequence can mitigate the network overhead. This can be achieved by the fact that the system is able to extract a minimum amount of 3D objects that are required to transmit. Also, using the proposed scheme, users can facilitate 3D Studio Max into distributed collaborative working environments. This, in consequence gives many benefits such as saving time as well as eliminating space constraints in the course of 3D modeling when we are under industrial design process.

† 비 회 원 : 경희대학교 전자계산공학과
suki@oslab.kyunghee.ac.kr
†† 종신회원 : 경희대학교 전자계산공학과 교수
sylee@oslab.kyunghee.ac.kr

논문접수 : 2000년 6월 23일
심사완료 : 2001년 6월 1일

1. 서 론

협력 시스템은 원격지에서 많은 사용자들에게 단일 가상공간에서 공동작업 환경을 제공할 수 있는 시스템으로, 인터넷, 멀티미디어, 가상현실 그리고 고성능 마이크로 프로세서 기술의 급속한 발전에 힘입어 구현이 가능하게 되었다. 이러한 협력 시스템은 멀티미디어, 산업 디자인, 3D 애니메이션, 만화 및 영화, 제품설계, 의료, 건축, 자동차, 선박, 항공 등 전 산업분야의 제품 생산과정에 응용되어 경제적으로 많은 비용절감 효과를 가져다 줄 수 있을 뿐만 아니라, 원격 회의, 원격교육, 전자상거래에도 응용될 수 있다. 그러나 협력시스템은 공유 객체의 일관성, 빠른 반응시간 및 영속성 유지, 동시성 제어, 자원 관리, 보안, 시스템 통합 등 아직도 해결해야 할 많은 기술적 난제를 가지고 있다[1].

본 논문에서는 분산환경에서 산업 제품의 3D 디자인 협력 공동작업 응용에 적용할 수 있는 3D 스튜디오 맥스 플러그인 개발에 대한 경험을 기술한다. 개발된 공유 플러그인을 적용한 협력 시스템은 저자들이 지난 2년간 개발한 시스템으로, 이는 플랫폼에 독립적이고, 확장과 이식이 용이하며, 작업의 부하를 효과적으로 분산시키고 객체관리를 용이하게 하기 위하여 분산형과 중앙집중형을 절충한 혼합형 서버구조를 지니고 있다[2]. 서버 시스템은 시스템 확장이 용이하도록 기능에 따라 사용자 관리(User Manager Server: UMS), 세션관리(Session Manager Server: SMS), 정보관리(Information Server: IS) 서버로 분류하였다. UMS는 사용자 인증 및 상태 정보 제공, 사용자 검색, 실시간 메시지 전송을 지원한다. SMS는 데이타 베이스 관리, 세션 관리, 3D 스튜디오 맥스의 3D 객체 관리, 화이트보드의 2D 객체 관리를 담당한다. IS는 서버 인증, UMS들 사이의 정보 교환과 메시지 전송 등을 지원한다. 이와 같이 기능별로 구분된 서버 구조는 부하를 쉽게 분산시킬 수 있을 뿐 만 아니라, 필요시 IS에 UMS를 첨부하거나 삭제할 수 있어 융통성과 확장성을 지원한다. 클라이언트 시스템의 경우 향후 협력시스템에 응용 프로그램의 개발과 추가가 용이하도록 클라이언트의 세션과 협력 응용(collaborative application)간에 인터페이스를 고려하여 설계되었다.

본 논문에서 다루는 주제는 단일 컴퓨터 시스템에서 작동되는 비주얼 3차원 모델링 도구인 3D 스튜디오 맥스를 지리적으로 떨어진 분산 환경에서 작동 할 수 있게 확장시켜 다수의 사용자가 공동으로 3D 모델링 작업을 수행할 수 있는 환경 제공을 목표로 한다. 이를 위하여 3D 스튜디오 맥스 플러그인 SDK를 사용하여 3D 객체 공유 플러그인을 개발하였다. 플러그인은 3D 스튜디오 맥스의 3D 객체 공유를 위해 공유객체를 전송하고자 하는 측에서는 공유객체에 대한 정보를 추출하는 기능을 제공하며, 수신측에서는 네트워크를 통해 전달받은 정보를 이용하여 전송측과 같은 3D 객체를 생성하는 기능을 제공한다. 전송측과 수신측에서는 여러 가지 형태의 3D 객체 관리를 위해 3D 객체 분석기능을 포함하는 전송 및 수신 관리자를 두어 공유 객체의 종류에 따라 분산처리 하도록 하고, 네트워크를 통한 데이타의 송·수신은 네트워크 인터페이스를 통해 처리하도록 하여 서버와 클라이언트간의 공유 객체 관리가 용이하도록 하였다. 그리고, 네트워크 인터페이스와 공유 플러그인은 3D 객체를 구성하는 가변적인 정보를 빠르게 저장할 수 있도록 공유 메모리를 통해 데이타를 공유하도록 한다. 이 같은 공유 플러그인의 개발로 인하여 단일 컴퓨터 시스템에서 수행되는 3D 스튜디오 맥스 작업을 분산 환경에서 작업할 수 있도록 확장시킴으로써, 3D 모델링 작업 시 공간과 시간의 제약을 최소화할 수 있다.

본 논문의 구성은 다음과 같다. 제 2 절에서 기존의 플러그인에 대한 관련 연구와 협력 시스템을 소개하고, 제 3 절에서는 본 논문에서 제안한 협력 시스템의 구조를 기술하며, 제 4 절에서는 협력 시스템의 클라이언트에서의 공유 플러그인 설계에 관해 기술하며, 제 5 절에서 실제 구현된 시스템의 구현 사례를 소개하고, 제 6 절에서 개발된 플러그인의 적용결과를 기술한 후, 마지막으로 제 7 절에서 결론을 내린다.

2. 관련 연구

많은 3D 모델링 도구들 중 3D 스튜디오 맥스는 개방형 구조를 기반으로 객체지향형 프로그래밍 환경을 제공한다. 3D 스튜디오 맥스는 자신이 제공하는 기능을 변경하거나 추가하기 위해 플러그인 SDK를 지원한다. 그리고, 3D 스튜디오 맥스는 시간이 많이 소요되는 렌더링 작업을 네트워크에 접속된 컴퓨터에게 분배할 수 있다[3]. 이런 개방적인 정책 때문에 Effect 모델링 관련 플러그인들, 사람이나 생물의 발자국의 패스를 지정할 수 있는 footpath와 같은 3D 스튜디오 맥스에서의 모델링 작업지원 플러그인[4] 등이 개발되고 있어 3D 스튜디오 맥스의 모델링 기능은 점점 확장 되어가고 있다.

분산 협력시스템 프레임워크와 관련된 국내외 연구는 다음과 같다. Habanero[5]는 자바로 개발된 클라이언트/서버 환경의 협력 시스템 프레임워크로 자바 객체를 인터넷을 통해 공유할 수 있는 환경을 제공하고, 협력

응용 개발을 위한 API를 제공한다. Habanero는 자바 객체 공유라는 특징 때문에 인터넷을 통해 플랫폼 독립적인 협력 시스템을 구축할 수 있는 장점을 가지나, 기존의 개인 PC상에서 작동되는 응용 소프트웨어와 연동이 미약한 단점을 가지고 있다.

GroupKit[6]은 캘거리 대학에서 개발한 실시간 응용 개발을 위한 협력 시스템 툴킷이다. 사용자는 Registrar에 접속하여 사용자 정보를 검색하고 Conference 정보를 획득하여 협력 작업에 참여한다. Registrar는 분산된 시스템의 정보를 저장하고 접속된 각 시스템에서 생성된 Conference의 목록 정보와 사용자를 구분할 수 있는 고유 계정을 제공하고 사용자 정보를 저장한다. 또한, 이를 접속한 시스템에게 제공하여 현재 협력 작업이 가능한 사용자 및 개설된 Conference의 정보를 제공한다. 협력 작업은 각 시스템 사이에 피어투피어(Peer-to-Peer) 방식으로 수행되어 각 시스템에게 협력 작업에서 요구되는 기능을 부담하여야 하는 반면, 중앙 서버에 각 시스템의 관리 정보만을 제공하여, 타 시스템의 상태를 빠르게 분석할 수 있으며, 협력 작업 시 발생되는 부하가 각 시스템에게 분산된다.

Egret[7]은 하와이 대학에서 개발된 협력 시스템 프레임워크로 그룹의 공동 작업을 지원하며, 동적이며 확장 가능한 프레임워크 구축을 목표로 개발되었다. 다중 클라이언트, 다중 서버, 다중 에이전트 구조로 되어 있으며, 클라이언트/서버 환경의 데이타베이스 구조를 사용한다. 서버의 부하를 줄이기 위해 서버의 데이타 처리부를 간단히 하고, 클라이언트에게 부하를 분산하였다. 공유 데이타는 하이퍼텍스트 기법을 사용하여 서로 연결되어 있으며, 이를 효과적 처리하기 위해 Gtable을 정의하였다.

CW-MAN(The Cooperative-Work Management System with Hybrid Architecture for Efficient Multimedia Collaboration)[8]은 효율적인 멀티미디어 공동 저작을 위한 혼합형 구조의 공동 저작 관리 시스템을 제안하였다. 이는 공동 저작 관리 정보를 중앙 관리 시스템에 집중시키고, 공동저작에 사용되는 공유 객체들은 각 저작자 시스템에게 분산시키는 혼합형 구조를 갖는 시스템 모델이지만 아직 구현은 되지 않았다.

[9]에서는 WWW상에서 플랫폼에 독립적으로 공동 작업을 지원하는 시스템을 소개하였다. 이는 웹 브라우져를 통해 협력 작업을 수행함으로써 협력 작업에 필요한 부가적인 소프트웨어의 구입 및 설치 과정이 필요없는 장점이 있으며, 웹을 통해 언제라도 새로운 버전의 소프트웨어를 통한 협력 작업이 가능하다. 하지만, 웹 브라우져에서 제공하는 기능에 의해 협력 작업이 제약받을 수 있으며 병행성 제어 등은 고려되지 않았다

CSpray[10]는 초고속 정보 통신망을 기반으로 하여 원거리에 있는 사용자들이 데이타를 공유하고 서로간에 시각 정보를 생성하고 이를 분석할 수 있는 도구이다. 그러나 이는 3D 스튜디오 맥스에 관련된 다양한 3D 객체 공유를 지원하지는 않는다.

본 연구에서는 산업 디자인이라는 특정 협력 작업에서 3D 스튜디오 맥스를 통한 실질적인 모델링 작업을 공유할 수 있도록 하는 공동 모델링 작업을 위한 공유 플러그인의 개발 경험을 기술한다.

3. 협력시스템 구조

이 장에서는 3D 맥스 플러그인이 작동되는 협력 시스템의 서버와 클라이언트의 구조에 대해 간단히 설명한다.

3.1 협력시스템 서버

[그림 1]에서 보는 것처럼 개발된 협력 시스템의 서버

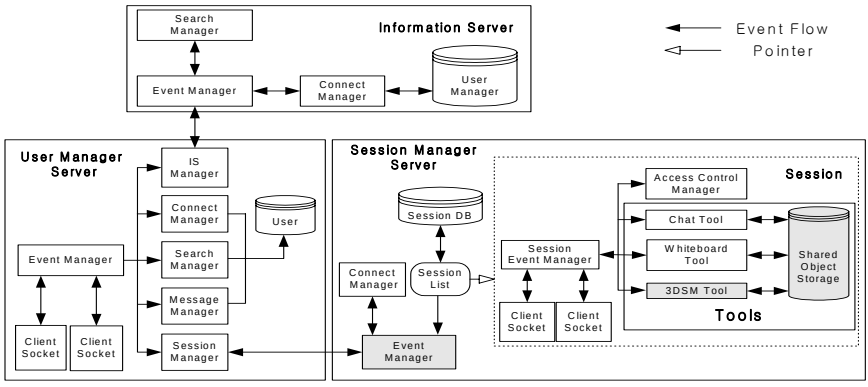

그림 1 협력 시스템 서버 구조

는 산업 디자인 협력 작업을 지원하며, 확장이 용이하도록 사용자 관리 기능과 세션 관리 기능으로 분리하였다. UMS는 사용자의 접속과 타 사용자 접속 및 메시지 전달 등의 협력 작업 이전의 서비스를 지원하며, 세션 생성 요청을 받아 세션 관리 서버에게 서비스를 요청하여 그 결과를 참여자에게 알린다. 세션 관리 서버인 SMS는 현재 디자인 프로젝트 스케줄을 관리하며, 협력 작업이 수행되는 세션의 관리와 협력 작업 시 발생되는 공유 데이타의 관리를 위한 이벤트 처리 서비스를 제공한다. SMS에서는 공유 객체의 일관성을 유지하기 위해서 낙관적 로킹 알고리즘이 사용된다[11][12]. 그리고 관계형 데이타 베이스와 연동되며, 우선순위 큐 스케줄링 기법을 사용하여 이벤트를 처리한다. IS는 사용자 서버간 이벤트 교환 및 공유 데이타의 교환을 가능하게 하여 협력 환경에 따라 시스템의 확장이 용이하도록 하였다.

3.2 협력시스템 클라이언트

클라이언트 시스템은 서버로부터 디자인 프로세스에 의한 스케줄 정보를 획득하여 현재 프로젝트의 진행상황을 인지하고 다수의 참여자와 디자인 협력 작업을 지원한다. 이를 위해 클라이언트 시스템은 [그림 2]에서 음영부분에 나타난 것과 같이 협력 작업 준비와 협력 작업 모듈로 나누어져 있다.

협력 작업 준비 단계에서는 UMS에 접속하여 타 사용자들의 접속유무와 협력 작업 진행 상황들을 사용자에게 알려준다. 이는 통신 관리자를 통해 입력된 패킷의 이벤트를 객체로 만들어 SMS와 UMS에서 처리하여 사용자 인터페이스를 통해 제공함으로써 구현된다.

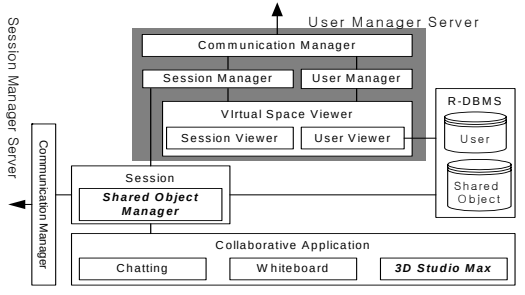

그림 2 협력 시스템 클라이언트

UMS는 타 사용자의 접속 유무에 대해 정보를 제공하며, 사용자 검색, 사용자에게 메시지 보내기 등의 서비스를 제공한다. SMS는 현재 협력 작업 상황을 제공하며 세션 생성, 참여 서비스 요청 방법을 제공한다. 가상공간 뷰어 (Virtual Space Viewer)는 사용자 정보 및 세션에 대한 정보를 출력하는 사용자 인터페이스 모듈이다.

협력 작업 단계는 세션 생성 또는 참가를 통해 SMS에 접속한 뒤 타 사용자와 데이타를 공유하여 작업을 수행하는 것을 의미한다. 세션의 참여는 UMS로부터 접속 인증을 받은 후 세션 접속에 필요한 정보를 수신하여 SMS에 접속한다. 세션 모듈은 세션의 참여시 필요한 정보를 SMS로부터 전송 받아 세션에 접속하고, 협력응용 모듈은 협력 작업 시 사용되는 응용 프로그램에 대한 인터페이스를 제공한다. 현재 구현된 응용 프로그램은 채팅, 화이트보드와 본 논문에서 다루는 공유 플러그인을 포함한 3D 스튜디오 맥스가 있다.

클라이언트의 이벤트 처리와 통신 모듈은 서버와는 달리 이벤트를 정의하는 클래스와 이벤트를 처리하는 프로토콜 클래스로 나누어진다. 프로토콜 클래스는 이벤트를 처리할 수 있는 코드를 담고 있으며, 3D 스튜디오 맥스의 이벤트를 정의하는 클래스와 프로토콜 클래스를 포함한다. 3D 객체 정보는 채팅과 화이트보드의 정보와 같이 협력시스템의 공유객체 관리자에 의해 관리된다.

4. 공유 플러그인 설계

이 장에서는 본 논문에서 제안하는 공유 플러그인 설계에 관하여 기술하려고 하는데, 우선, 3D 스튜디오 맥스 시스템과 공유 플러그인의 전체구조에 대해 설명하고, 공유 플러그인의 초기화, 공유 플러그인의 3D 객체 정보 추출 모듈과 3D 객체 생성 모듈에 대하여 설명한다.

4.1 3D 스튜디오 맥스 시스템과 공유 플러그인 전체 구조

[그림 3]은 공유 플러그인의 전체 구조로 크게 3D 객체 생성모듈, 3D 객체정보 추출모듈, 그리고 공유 메모리에 데이타를 기록하는 송신자(Sender)와 공유 메모리로 3D 객체 정보를 읽어 오는 수신자(Receiver)로 구성되어 있다. 그림에서 빗금부분의 인터페이스는 SDK에서 제공되는 클래스로서 플러그인 내부에서 3D 스튜디오 맥스 시스템에 접근할 수 있도록 하며 플러그인은 이를 이용하여 뷰포트상의 객체에 접근하여 3D 객체 정보를 추출·생성할 수 있다.

이때, 3D 스튜디오 맥스 시스템에 접근하기 위해 사용되는 클래스들은 [그림 4]와 같으며, 여기서 Front EndController는 새로운 프론트 엔드(Front end) 플러그인을 생성할 수 있는 SDK에서 제공되는 기본 클래스이다[13]. 이는 3D 스튜디오 맥스의 전반적인 사용자 인터페이스를 제어할 수 있으며 기본적인 폼을 사용한

애플리케이션의 생성을 가능하게 한다. 이를 통해 구현 가능한 사용자 인터페이스에는 타이틀 바, 드롭다운 메뉴, 툴바, 커맨드 패널, 타인 컨트롤, Right-click 뷰포트 레이블 메뉴, Right-click 객체 메뉴 등이 있다.

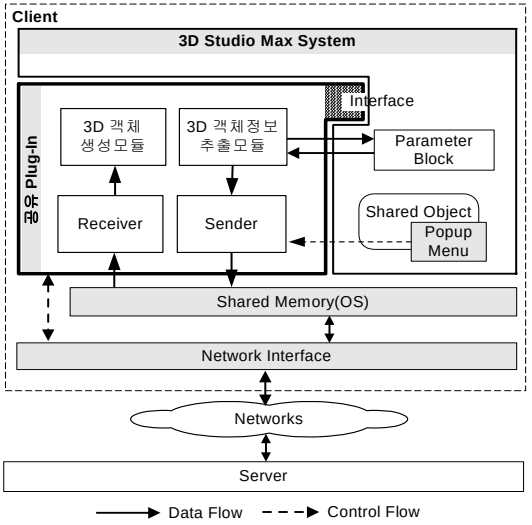

그림 3 공유 플러그인 전체 구조도

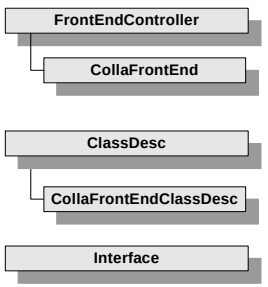

그림 4 플러그인 메인 클래스

CollaFrontEnd는 협력 작업 환경에 적합한 3D 스튜디오 맥스의 사용자 인터페이스를 구성하기 위한 기본 클래스이다. 3D 스튜디오 맥스 구동 시 플러그인의 초기화 작업을 수행하며 상위 클래스인 FrontEnd Controller 함수들을 이용한 사용자 인터페이스 이벤트를 인터럽트한다. 그리고 주로 사용자의 모델링 데이타를 전송하기 위하여 이벤트 발생기 역할의 Right-click 객체 메뉴를 재 설정한다. 메뉴의 초기화 작업 및 메뉴 선택 이벤트 발생에 대한 처리 루틴도 포함한다.

인터페이스는 3D 스튜디오 맥스 플러그인 시스템의 접근을 위한 인터페이스 클래스로써 직접적으로 뷰포트에 제어권을 가지며, FrontEndController는 인터페이스의 참조를 가짐으로써 간접적으로 뷰포트상의 오브젝트 제어권을 가진다. ClassDesc는 플러그인이 필수적으로 가지는 클래스로 DLL에서 플러그인 클래스에 대한 정보를 시스템에 제공한다. 이 클래스 메소드를 통하여 종류별, 기능별 플러그인 클래스 디스크립터(class descriptor)를 생성한다. CollaFrontEndClassDesc는 개발 플러그인 정보를 상세히 기술하는 클래스로써 클래스 이름과 다수의 플러그인과 구별할 수 있는 유일한 클래스 아이디, 커맨드 패널의 카테고리 이름 등의 함수들을 제공한다.

한편, 전체적인 공유 플러그인의 작동 과정은 다음과 같다. 3D 객체를 공유하고자 할 경우, 3D 객체를 선택한 다음 추가된 사용자 인터페이스 팝업 메뉴를 이용하여 이벤트를 발생시켜 공유 플러그인을 작동시킨다. 3D 객체정보 추출모듈은 파라미터 블록[그림 5]과 플러그인 초기화 시 3D 스튜디오 맥스 시스템으로부터 선택된 3D 객체 정보를 넘겨받은 인터페이스를 이용하여 파라미터 블록 이외의 기타 객체 정보들을 추출하여 공유 플러그인의 송신자에게 전달한다. 송신자는 전달받은 3D 객체 정보를 공유 메모리에 기록하고 네트워크 인터페이스에게 공유 메모리에 기록한 3D 객체 정보를 서버에 전송하도록 요청한다. 그리고 서버에서는 전달받은 이벤트를 분석하고 세션의 다른 참여자들에게 객체 정보를 전달한다. 수신측에서는 3D 객체의 정보가 서버로부터 도착하면, 네트워크 인터페이스가 공유 메모리에 3D 객체정보를 기록하고 수신자가 공유 메모리의 정보를 읽은 다음 3D 객체 생성모듈에 전달하여 3D 스튜디오 맥스의 3D 객체를 생성하도록 한다.

3D 스튜디오 맥스 시스템에서 3D 객체를 구성하는 주요 데이타는 파라미터 블록에 저장된다[13]. [그림 5]는 파라미터 블록의 구조를 나타내는 것으로 내부적으로 가상 배열 메카니즘을 유지하며 인덱스를 이용하여 파라미터 블록에 입출력할 수 있다. 하나의 파라미터 배열은 하나의 3D 객체와 대응되며 각 파라미터 블록은 블록 인덱스와 블록에 저장되는 데이타 형태를 나타내는 데이타 타입 그리고, 실제 데이타 값으로 구성된다. 3D 객체 정보는 파라미터 블록으로부터 추출할 수 있으며 추출된 정보를 공유 메모리를 통해 네트워크 인터페이스에 전달한다. 공유 플러그인과 네트워크 인터페이스는 [그림 6]에서 정의한 데이타 포맷 순서대로 입출력하며 데이타 포맷은 전송되는 객체 정보에 따라 5가지로 구성된다.

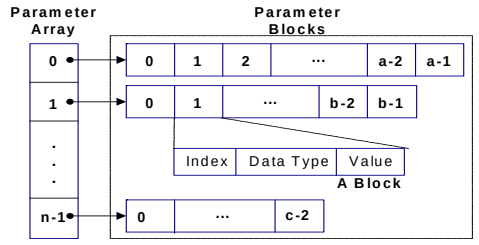

그림 5 파라미터 블록 구조

첫 번째 Mem&Handle은 공유 메모리 초기화와 윈도우 핸들 교환을 위한 데이타 포맷이다. 교환된 핸들은 네트워크 인터페이스와 공유 플러그인에 제어 메시지를 통해 공유 메모리를 읽거나 쓰도록 요청하는데 사용된다.

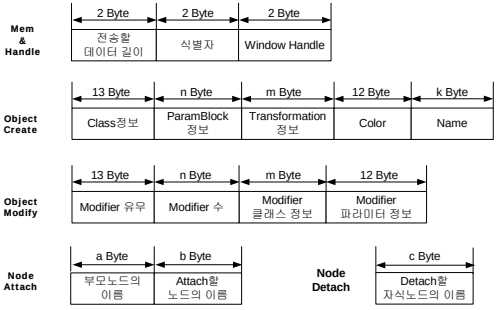

그림 6 플러그인과 공유 메모리간의 단계별 데이타 형식 정의

두 번째 객체 생성 전달자(Object Create Transfer)는 3D 객체 정보를 전송할 때 사용되는 데이타 포맷으로 3D 객체 정보를 구분하는 ClassID 정보, 파라미터 블록 정보, 이동, 회전, 확대 및 축소를 위한 정보와 3D 객체의 색, 이름으로 정의된다. 세 번째 객체수정(Object Modify)은 수정자 객체 정보를 전송할 때 사용되는 데이타 포맷으로 노드에 수정자가 적용되었는지의 유무, 적용되었다면 적용된 수정자의 개수, 적용된 수정자의 종류를 구분할 수 있는 ClassID 정보와 수정자의 속성 정보를 가지는 파라미터 블록으로 정의된다. 네 번째와 다섯 번째는 노드간의 관계를 정의하는 데이타로 객체의 계층 관계를 만들어 준다. 한 노드를 부모 노드로 하고 다른 노드를 자식 노드로 만들 경우, 네 번째 포맷의 부모노드의 이름과 자식 노드의 이름을 사용하며, 그 관계를 제거할 때는 다섯 번째 포맷의 자식 노드의 이름을 사용한다. 이와 같은 순서로 3D 객체 추출 모듈과 3D 객체 생성 모듈간에 데이타 전송 및 수신의 동기화를 이룰 수 있다.

공유 플러그인과 네트워크 인터페이스는 공유 메모리를 통해 정보를 교환한다. 3D 스튜디오 맥스의 공유 객체를 공유 메모리를 통해 통신 모듈과 송·수신하기 위해 [그림 7]과 같이 전송단과 수신단으로 정의하였다. 전송단과 수신단의 구조는 매우 유사하며 주로 전송단에서는 공유 데이타의 전송 순서를 정립하여 공유 메모리에 기록하며 수신단에서는 전송된 데이타를 공유 메모리를 통하여 순서대로 읽어오는 역할을 담당한다.

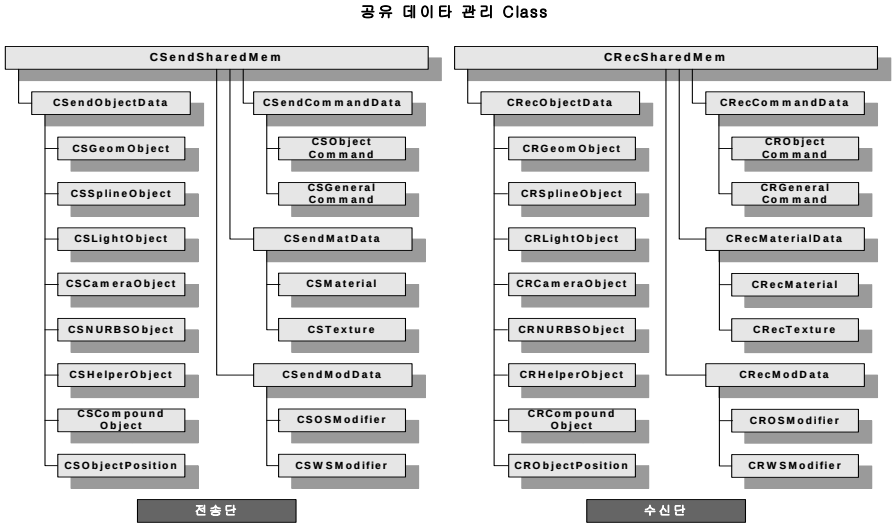

그림 7 공유 객체 송·수신을 위한 공유 메모리 관리 클래스

CSendSharedMem/CRecSharedMem는 공유 모델링 객체 데이타 관리 부분의 최상위 클래스이며 하위 데이타 분석 클래스의 공통 처리 모듈을 포함한다. 또한 객체의 파라미터 블록에 대한 접근 제어권을 가지며 공유 메모리의 실질적 관리를 담당한다. CSendObjectData/CRecObjectData는 3D 스튜디오 맥스의 오브젝트(Primitive Object, Extended Primitive Object, Shape Object, Compound Object, NURBS Object, Helper Object, Carema Object, Light Object. etc)에 대한 공통 분석 모듈을 포함한다. CSendModData/CRecModData는 Modifier(Object Space Modifier, World Space Modifier)에 대한 공통 분석 모듈을 포함한다. CSendMatData/CRecMaterialData는 Material과 Texture 대한 공통 분석 모듈을 포함한다. CSend CommandData/CRecCommandData는 3D스튜디오 맥스의 내부/외부 명령어들(File Loading, Viewport 제어, 초기화, 랜더링 등)에 대한 공통 분석 모듈을 포함한다.

본 논문에서 제안한 공유 플러그인은 3D 객체 공유를 위해 3D 스튜디오 맥스로부터 3D 객체 정보를 추출하고, 공유 메모리로부터 3D 객체 정보를 읽어와서 3D 스튜디오 맥스의 3D 객체를 생성하는 기능을 제공한다.

4.2 공유 플러그인 로딩과 초기화

3D 스튜디오 맥스가 실행되면 3D 스튜디오 맥스 시스템은 공유 플러그인을 로딩하고 초기화한다. 공유 플러그인은 초기화 시 3D 스튜디오 맥스 시스템에 접근할 수 있도록 인터페이스 포인터를 3D 스튜디오 맥스 시스템으로부터 넘겨받고, 공유 메모리를 통해 네트워크 인터페이스와 3D 스튜디오 맥스의 윈도우 핸들을 교환한다. [그림 8]은 공유 플러그인 초기화 알고리즘이다. 우선, 공유 메모리의 주소를 네트워크 인터페이스로부터 넘겨받고 3D 스튜디오 맥스의 윈도우 핸들을 얻어 [그림 6]에서 정의한 데이타 포맷(Mem&Handle)대로 공유 메모리에 기록한다. 그런 다음, 네트워크 인터페이스의 윈도우 핸들을 통해 네트워크 인터페이스에 메시지를 보내 3D 스튜디오 맥스의 윈도우 핸들을 공유 메모리로부터 읽도록 요청한다.

```
procedure Initialize(Interface *ip, HWND hClientWnd)
begin
    Create shared memory map
    Get 3ds_max_window_handle
    Write to shared memory(data length,
    identifier, 3ds_max_window_handle)
    SendMessage(network_interface, Request_Of_Reading)
end;
```

그림 8 공유 플러그인 초기화 알고리즘

이렇게 공유 플러그인은 윈도우 핸들을 통해 정의한 메시지를 보냄으로써 네트워크 인터페이스와 공유 플러그인이 서로 통신할 수 있도록 하고, 공유 메모리를 사용함으로써 가변적인 공유객체 정보를 빠르게 교환한다.

4.3 3D 객체 정보 추출 모듈

[그림 9]는 3D 스튜디오 맥스의 객체 및 명령을 전송하는 전송측의 모듈 구성도이다. 3D 객체 정보 추출 모듈은 크게 객체(Object), 재료(Material), 수정자(Modifier), 명령(Command)의 분석 모듈로 되어 있으며, 각각의 모듈은 3D 스튜디오 맥스에서 제공하는 여러 가지 형태의 객체에 따라 세부 분석 모듈을 포함한다. 예를 들어 객체 분석 모듈은 3D 스튜디오 맥스의 라이트(Light) 객체에 대한 라이트 객체 분석자 등을 포함한다. 이 분석 모듈들은 상위단에 지원객체의 공통 속성을 추출하는 공통 분석 모듈을 두어 일반화하였다. [그림 9]의 전송자는 각 분석 모듈이 추출한 정보를 받아 공유 메모리에 쓰는 작업을 하며, 네트워크 인터페이스에 제어 메시지를 보내 공유 메모리의 정보를 서버로 전송하도록 요청한다. 네트워크 인터페이스는 공유 메모리의 정보를 읽어 서버에 전송한다.

3D 객체는 파라미터 블록에서 추출한 정보와 색상, 이름 등의 기타 정보로 이루어져 있으며, 이 정보를 이용하여 전송한 측 클라이언트에서 생성된 객체와 동일한 3D 객체를 구성할 수 있다. 전송측 모듈의 각 추출 모듈들은 파라미터 블록에 접근이 허용되며, 파라미터 블록에서 추출된 데이타와 인터페이스를 이용해 추출한 3D 객체 정보들을 공유 메모리에 기록하고, 네트워크 인터페이스에게 객체 정보를 가져가도록 전송 제어 신호를 보낸다.

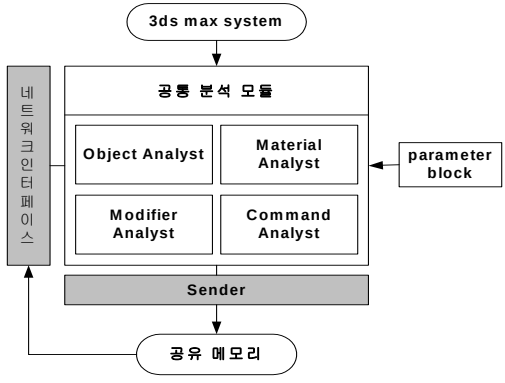

그림 9 전송측 모듈 구성도

[그림 10]은 3D 객체정보 추출모듈에서 3D 객체정보를 추출하여 공유 메모리에 기록하는 알고리즘으로 [그림 4]의 인터페이스를 이용하여 노드에 접근한 뒤 3D 객체의 색상, 이름 등과 같은 기타 정보를 추출하고, 이 노드로부터 객체에 접근하여 객체의 파라미터 블록에서 정보를 추출한다. class_info는 3D 스튜디오 맥스 SDK에서 정의한 3D 객체의 분류 방법으로 3D 객체 정보를 받아 최초의 객체를 생성하는데 사용한다.

```
procedure GetObjectInfo( )
begin
        Get selected object pointer and node pointer
        Get class_info from object
        Get parameter_block_info from object
        Get other_data from node
        Sender(class_info, parameter_block_info, other_data)
end;
```

그림 10 3D 객체정보 추출 알고리즘

4.4 3D 객체 생성 모듈

[그림 11]은 네트워크 인터페이스로부터 객체 정보를 입력받아 공유객체를 생성하도록 하는 수신측의 모듈 구성도이다. 네트워크 인터페이스는 네트워크로부터 받은 정보를 공유 메모리에 기록한 후, 3D 스튜디오 맥스에 객체 정보 수신 메시지를 보내어 공유 플러그인이 작동되도록 한다.

전송측 모듈을 통해 전송된 수신측 모듈의 전체적인 구조는 3D 객체 정보 추출 모듈과 대칭구조를 이룬다. 네트워크 인터페이스는 서버의 요청에 따라 3D 객체 정보를 받아 공유 메모리에 기록하고 3D 스튜디오 맥스 윈도우에 제어 메시지를 보내면 공유 플러그인이 작동하여 수신자(Receiver)는 공유 메모리로부터 정보를 가져온다. 공통 분석 모듈에서는 수신자로부터 받은 정보 중

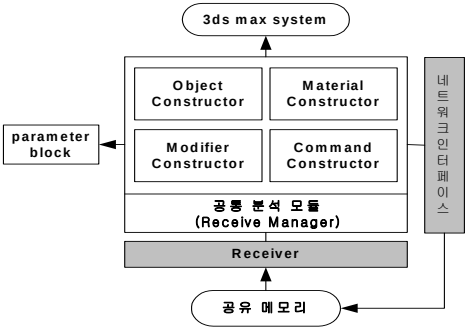

그림 11 수신측 모듈 구성도

class_info를 이용하여 객체(Object), 수정자(Modifier), 매트리얼(Material), 명령(Command)의 하위 생성 모듈 중 수신한 객체 정보에 해당하는 생성 모듈을 선택하여 3D 객체를 생성하도록 한다.

[그림 12]는 3D 객체 생성 모듈 알고리즘으로 전송받은 데이타를 이용하여 3D 객체를 생성하는 과정으로, 먼저 class_info를 이용하여 최초의 객체를 생성하여 파라미터 블록을 생성한뒤, 인터페이스를 이용하여 노드를 생성하고 3D 스튜디오 맥스의 뷰포트를 새로 그린다.

```
procedure ReceiveObject( )
begin
      Receiver(class_info, parameter_block_info, other_data)
      Create object using class_info
      Create parameter_block(parameter_bolcok_info)
      Create Node(created_object)
      Set other data values(other_data)
      Redraw viewport
end;
```

그림 12 3D 객체 생성 모듈 알고리즘

이 밖에 분산환경에서 여러 참여자가 공유 데이타를 수정할 때에는 3D 객체에 대한 제어권 문제가 발생한다. 본 논문에서 다루는 공유 플러그인에서는 실시간으로 공유 객체를 선택했을 때 접근 제어권을 주어 타 클라이언트의 3D 스튜디오 맥스의 해당 객체를 비 활성화시켜 수정하지 못하도록 함으로써 해결하였다.

5. 공유 플러그인 구현

이 장에서는 제안한 공유 플러그인을 산업 디자인 협력 시스템에서 구현한 결과를 기술한다. 개발된 산업 디자인 협력 시스템 서버는 이식성이 뛰어난 자바 언어를 이용하여 윈도우 NT에서 개발되었고, 협력 시스템에서 산업 디자인 프로세스를 지원하는 응용인 채팅, 화이트보드는 Visual C++ 6.0을 이용하여 개발되었다. 그리고 네트워크 환경은 TCP/IP 기반이며 공유 플러그인은 3D 스튜디오 맥스에서 제공하는 SDK를 사용하여 Visual C++ 6.0을 이용해 개발하였다[표 1].

표 1 서버/클라이언트 프로그램 동작환경

구분 / 항목	서 버	클라이언트
운영체제	Windows NT	
네트워크	TCP/IP	
개발언어	JDK 1.2	Visual C++ 6.0

[그림 13]은 제안한 공유 플러그인의 구현 시나리오를 순서도로 나타낸 것이다.

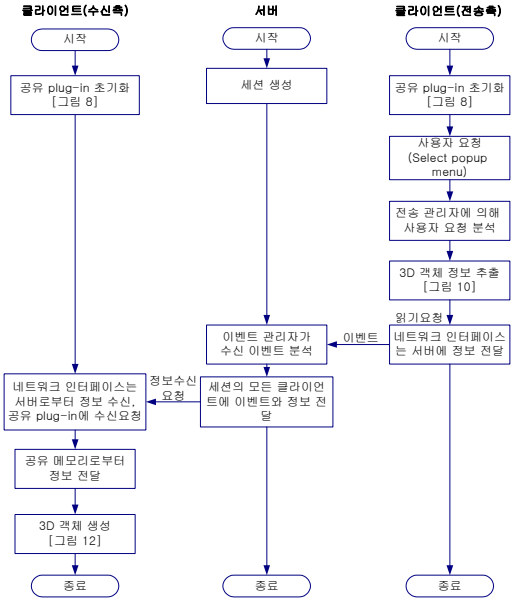

그림 13 제안한 공유 플러그인의 전체 시나리오

클라이언트들은 3D 스튜디오 맥스가 실행됨과 동시에 [그림 8]에서 나타난 바와 같이 공유메모리를 초기화하고 공유 플러그인과 네트워크 인터페이스는 서로의 윈도우 핸들을 공유한다. Right-click 객체메뉴 선택을 통해 사용자 요청이 발생하면, 공유 플러그인의 3D 객체 정보 추출 모듈은 요청을 분석하여 공유객체에 적합한 분석모듈을 이용해 3D 객체 정보를 추출한다. 그런 다음 공유 메모리에 추출한 정보를 기록하고 초기화 시 교환한 윈도우 핸들을 이용해 네트워크 인터페이스에게 공유 메모리의 정보를 읽도록 요청한다. 네트워크 인터페이스는 공유 메모리로부터 정보를 읽고, 정의된 이벤트 구분자와 공유객체 정보를 서버에 전달한다. 서버는 이벤트 구분자를 이용해 3D 스튜디오 맥스의 이벤트인지 아닌지를 판단한 다음 세션에 참여한 다른 클라이언트들에 정보를 전달한다. 수신측의 네트워크 인터페이스는 서버로부터 정보를 받아 공유 메모리에 기록하고, 공유 플러그인에 읽기 요청하면 공유 플러그인의 3D 객체 생성 모듈은 공유 메모리로부터 읽은 정보를 이용하여 3D 객체를 생성한다.

[그림 14]는 제안한 협력시스템에서 3D 스튜디오 맥스를 사용하여 객체를 공유하면서 마우스를 디자인하는

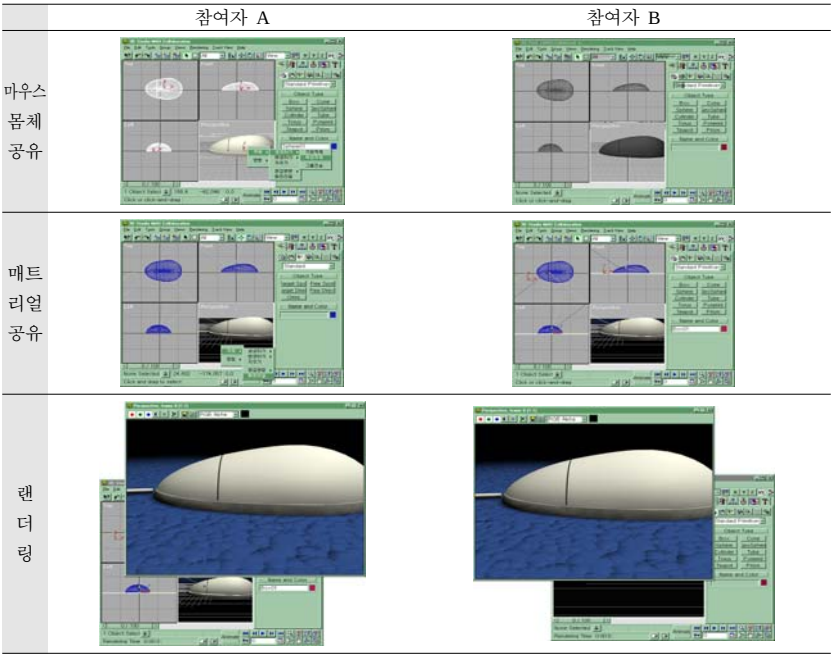

그림 14 공유 플러그인을 이용한 모델링 작업 예

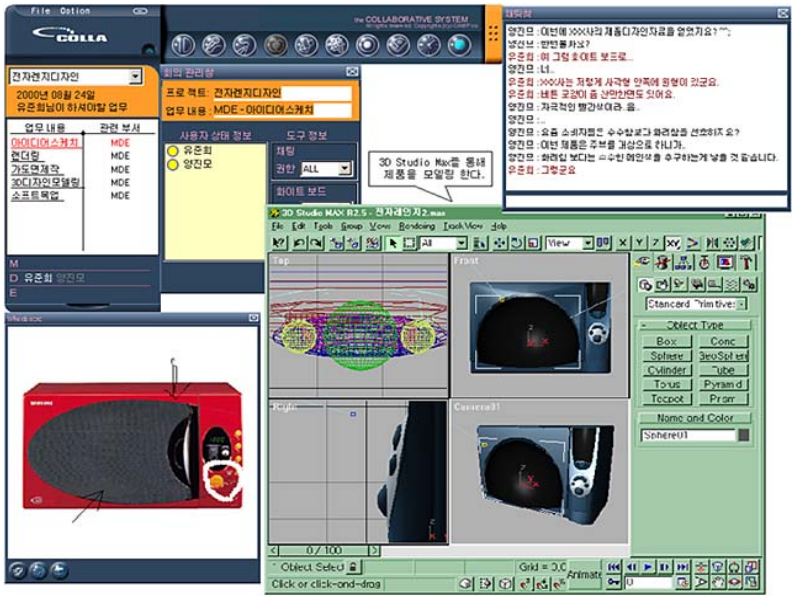

그림 15 클라이언트 구현 화면

협력 작업 예를 보여준다. 참여자 A는 객체를 생성하여 전송하는 전송측이며, 참여자 B는 이를 전송받은 클라이언트의 수신측이다. Right-click 객체메뉴를 통하여 전송 이벤트를 발생시키며, 메뉴는 작업의 편의성과 인터페이스를 생각하여 간결하게 구성하였고 내부에 분석 루틴을 가지고 있어 객체의 분류 처리가 용이하다. 또한 실시간으로 공유 객체를 선택했을 때 접근 제어권을 주어 타 클라이언트의 해당 오브젝트를 비 활성화하여 수정할 수 없도록 함으로써 충돌문제를 해결하였다.

[그림 15]는 채팅, 화이트보드와 3D 스튜디오 맥스를 협력 도구로 사용하는 디자인 협력 작업의 화면을 보여준다. 서버 시스템은 서버에 접속한 다른 서버나 클라이언트 목록을 표시해주며, 이벤트 모니터의 이벤트 큐의 상태를 출력하여 서버 상태를 모니터링 한다. 클라이언트 소프트웨어는 산업 디자인 프로세스의 공동의 협력 작업을 위한 사용자 인터페이스를 제공한다. [그림 15]의 좌측 상단에 출력된 산업 디자인 프로세스 중 현재 진행중인 전자레인지 디자인 협력 작업에 참여하여 공유 플러그인을 포함하는 3D 스튜디오 맥스를 사용하여 3D 객체를 공유하도록 한다. 참여자들은 3D 객체의 속성(이동, 재질·색·크기) 변경과 객체의 생성, 삭제 등을 통해 의사를 교환하며 작업중 생성되는 모든 객체들은 참여자들간에 공유된다. 화이트 보드는 랜더링된 기존의 전자레인지 이미지의 공유 방법을 제공한다. 또한, 채팅을 통해 객체 변경으로 전달할 수 없는 내용을 참여자들에게 제공한다.

6. 구현결과 및 분석

이 장에서는 공유 플러그인을 포함하는 서버의 성능 평가를 통해 서버의 특성을 분석한다.

성능을 평가하기 위해 사용된 시스템은 Pentium II 300Mhz와 224MB RAM 시스템에서 Windows NT 4.0 OS이며, 100Mbps 지역 랜 환경에서 각각 단일 UMS와 SMS에 접속자 수와 이벤트의 발생빈도에 따른 이벤트의 응답 시간을 검사하였다.

[그림 16]은 접속자의 수에 따른 UMS의 반응 시간을 나타낸 것으로, 접속자는 10초당 1개의 이벤트를 발

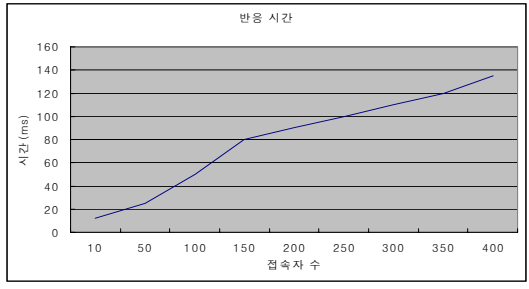

그림 16 UMS의 접속자에 따른 반응 시간

생시킨다. [그림 17]은 SMS의 사용자 수에 따른 반응 시간으로 10명이 한 세션에 참여하고, 30초와 60초마다 각 사용자가 이벤트를 발생시킨다. 이때, 반응 시간은 참여자의 수보다는 발생되는 이벤트의 빈도에 따라 급격하게 증가하는 것을 알 수 있으며 이는 발생된 메시지를 세션에 참여한 참여자에게 전송하고, 데이타 베이스에 저장하는 시간이 증가함에 따라 급격히 증가함을 보여준다.

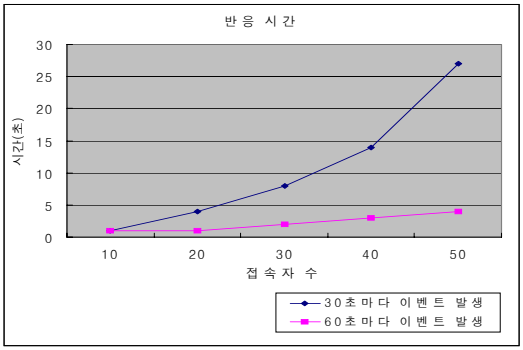

그림 17 SMS의 접속자수에 따른 반응 시간

공유 플러그인을 포함하는 서버의 구현결과에 대해 다음과 같이 정리할 수 있다. 첫째, 공유 데이타를 처리하는 SMS와 처리하지 않는 UMS의 경우, SMS가 현저히 느린 것을 알 수 있다. 이것은 공유 데이타의 처리 작업 시 협력 작업 참여자에게 공유 데이타를 네트워크를 통하여 제공하여야 하는 산업 디자인 협력 작업의 경우 공유 데이타의 크기가 클수록 많은 처리시간이 요구됨을 알 수 있다.

공유되는 3D 객체는 일반 협력 시스템에서의 공유 데이타에 비해 큰 용량이기 때문에 SMS의 속도저하에 큰 요인이 된다. 따라서 전송되는 공유 데이타의 크기를 줄일 수 있는 방법이 요구되는데, 본 연구에서 다루는 구현된 시스템에서는 3D 스튜디오 맥스의 모델링 데이타 각각을 분석하여 전송되는 필요 데이타를 최소화함으로써 다소 3D 객체의 크기를 줄일 수 있었다.

7. 결 론

본 논문에서는 단일 환경에서 작동되는 Kinetic사의 비주얼 3차원 모델링 도구인 3D 스튜디오 맥스를 지리적으로 떨어진 분산 환경에서 작동 할 수 있게 확장시켜 다수의 사용자가 공동으로 3D 모델링 작업을 수행할 수 있는 환경 구축 사례를 기술하였다.

개발된 플러그인은 3D 스튜디오 맥스로부터 3D 객체 정보를 추출하는 기능과 3D 객체 정보를 이용하여 3D 스튜디오 맥스의 3D 객체를 생성하는 기능 그리고, 3D 객체 정보의 전달과 수신을 위한 임시공간으로써 공유 메모리 맵을 이용하는 공유 메모리 입출력 기능이 있다. 3D 스튜디오 맥스 시스템의 접근방법으로 SDK에서 제공하는 인터페이스를 사용하였으며 공유 메모리를 사용하여 가변적인 형태의 3D 객체를 저장할 수 있도록 하였다. 3D 스튜디오 맥스에서 제공되는 객체를 구성하는 데이타를 분석한 후 클라이언트가 요구하는 최소한의 데이타만을 추출함으로써 전송될 공유 데이타의 크기를 줄일 수 있어 네트워크의 부하를 감소시킬 수 있었다. 또한 클라이언트들 간의 공유 객체의 동시성 제어 문제는 실시간으로 공유 객체를 선택한 클라이언트에게만 접근 제어권을 가지게 하여 해결하였다.

기존의 플러그인들이 3D 스튜디오 맥스 본래의 기능인 모델링 기능을 향상시키는데 목적하였다면, 본 논문에서 개발한 공유 플러그인은 단일 컴퓨터 시스템에서 수행되는 3D 스튜디오 맥스 작업을 분산 환경에서 작업할 수 있게 확장시킴으로써 산업 디자인 협력 시스템에서의 3D 모델링 데이타 공유 기능을 지원할 수 있도록 하는데 목적이 있다. 그러나, 현재 개발된 공유 플러그인은 인터페이스가 불편하며, 클라이언트들 간의 공유 데이타 정보의 불일치가 발생할 수 있다. 향후 공유 데이타의 실시간 공유와 불일치성의 문제의 연구가 필요하다.

참 고 문 헌

[1] Tom Rodden, "*A Survey of CSCW Systems,*" Interacting with computers, vol. 3 no. 3, pp. 319~352, 1991.

[2] 양진모, 이승룡, 전태웅, "확장성을 고려한 산업디자인 협력 시스템 설계 및 구현", 정보과학회 논문지, 제 6 권, 제 5호, pp. 513-527, 2000. 1.

[3] Autodexk Support, http://www3.autodesk.com/adsk/support/techdoc

[4] Effectware, http://www.effectware.com/download/help/help_dlu.htm

[5] Annie Chabert, Ed Grossman Larry Jackson, Stephen Pietrovicz, "*NCSA Habanero: Synchronous Collaborative Framework and Environment,*" 1998.

[6] Roseman, M. and Greenberg, S. "*Building Real Time Groupware with GroupKit, A Groupware Toolkit,*" ACM Transaction On Computer Human Interaction 3(1), 1996. 3.

[7] Philip M. Johnson, "*Experiences with EGRET: An*

exploratory group work environment. Collaborative Computing," 1994. 1.
[8] 이광행, 전재우, 오삼권, "CW-MAN:효율적인 멀티미디어 공동저작을 위한 혼합형 구조의 공동저작 관리 시스템", 한국 정보 처리학회 논문지, p.1253-1262, 1999.5.
[9] 정의현, 박용진, "WWW상에서의 공동작업 시스템의 설계 및 구현", 정보과학회논문지(C), 제3권, 제4호, pp.384-396, 1997. 8.
[10] Alex Pang, "Collaborative 3D Visualization with CSpray," IEEE Multimedia and IEEE Computer Graphics and Applications on 3D and Multimedia on the Information Superhighway, 1997. 3.
[11] M.T. Ozsu and P. Valduriez, *Principles of Distributed Database Systems*, Prentice-Hall, pp. 327~329, 1998.
[12] S. Bhola, G. Banavar, and M. Ahamad, "*Responsiveness and Consistency Tradeoffs in Interactive Groupware*," In Proceedings of 7th ACM Conference on Computer Supported Cooperative Work, 1998. 11.
[13] Christer Janson, Help for 3D Studio MAX SDK from within Visual C++, Kinetix, 1998.
[14] 김창한, 양재헌, "Collaborative Virtual Environment 에서의 동시성 제어와 기술", 정보과학회지, 제16권, 제7호, 1998. 7.
[15] 궁상환, 황승구, "Collaborative Computing 기술 및 응용", 정보과학회지, 제16권, 제7호, 1998. 7.
[16] J.H. Lee, A. Prakash, T. Jaeger, and G. Wu, "*Supporting multi-user, multi-applet workspaces in CBE*," Proceedings of the ACM Conference on Computer-Supported Cooperative Work(CSCW'96), pp. 344-353, 1996.
[17] G. Smith and T. Rodden, "*SOL: A Shared Object Toolkit for Cooperating Interfaces*," Technical Report CSEG/7/1995, Lancaster University, 1995.
[18] P. Dewan and R. Choudhary, "*A Flexible and High-Level Framework for Implementing Multi-User User Interfaces*," ACM Transactions on Information Systems, Vol. 10, No. 4, pp. 345-380, 1992. 10.
[19] A. Prakash and H.S. Shim, "*DistView: Support for Building Efficient Collaborative Applications using Replicated Objects*," Proceedings of CSCW '94, ACM Press, New York, pp. 153-164, 1994.
[20] 최종명, 김형진, 최재영, "CoDraw: 자율 객체를 이용한 웹에서의 유연성있는 CSCW 시스템 설계 및 구현", 정보과학회 논문지(C), 제5권, 제5호, pp. 574-582, 1999. 10.
[21] 조성빈, 김진석, 진성일, CSCW를 위한 분산 객체 공유시스템, 정보과학회 논문지(C), 제5권, 제3호, pp. 326-335, 1999. 6.
[22] Developer Consulting Group, http://support.ktx.com/~200, Kinetix

권 태 숙
2000년 한신대학교 정보통신학과 학사. 2000년 3월 ~ 현재 경희대 전자계산공학과 석사과정. 관심분야는 게임, Collaborative Computing, P2P

이 승 룡
1978년 고려대학교 재료공학과 학사. 1987년 12월 Illinois Institute of Technology 전산학 석사. 1991년 12월 Illinois Institute of Technology 전산학 박사. 1992년 ~ 1993년 Governors Srtate University 조교수. 1993년 ~ 현재 경희대학교 전자정보학부 (전자계산공학 전공) 부교수. 관심분야는 실시간 컴퓨팅, 멀티미디어 시스템