분산 환경 하에서의 공동 작업을 위한 프레임웍 설계

박상문[0] 이태동 조성업 정창성
고려대학교 전자공학과 분산멀티미디어컴퓨팅 연구실
{sangmoon, lyadlove, easyup, csjeong}@snoopy.korea.ac.kr

A framework for CSCW in Distributed Environment

Sang-moon Park[0] Tae-dong Lee Sung-up Cho Chang-seong Jeong
Dept. of Electronics Engineering, Korea University

요 약

본 논문에서는 최근 다양한 분야에 응용되고 있는 Computer-supported Cooperative work (CSCW)를 위한 새로운 구조적 모델을 제안한다. 본 논문에서 제안하는 구조적 모델은 현재까지 만들어져온 CSCW 시스템 모델들의 특징과 구조를 분석하고서, 분산 환경을 기반 환경으로 해서 보다 객체 지향적으로 설계한 것으로서 완벽한 peer to peer 의 구조를 취한다. DOVE(Distributed Object-oriented Virtual Environment)라는 분산 환경 하에서 설계된 본 시스템은 D OVE가 제공하는 서비스의 하나인 객체 그룹 원격 호출(Object group method invocation)을 이용해서 객체들에 대해서 멀티캐스팅하는 방식을 사용했으며 이를 통해 cooperative work를 위한 최고의 성능을 제공할 수 있도록 설계되었다.

1. 서론

최근 들어서 컴퓨터의 성능이 향상되고 네트웍 속도가 개선되면서 과거 꿈으로만 여겨져 왔던 공동작업(Computer-supported Cooperative work : CSCW)이 보다 더 구체화되고 있다. 멀티유저가 실시간으로 멀티미디어 데이터를 주고 받을 수 있는 환경이 제공 된 것이다.
이에 따라서 최근 들어 보다 나은 공동작업을 위한 환경을 구축하기 위해서 다양한 플랫폼들이 개발되고 제안되어 왔다.[1,2,3] 그러나 다수의 시스템들이 비 객체지향적 설계[1], 성능 문제[2], heterogeneity를 지원하지 못하는 문제[3] 등으로 인해서 제약된 환경 하에서 쓰이고 있는 실정이다.
본 논문에서는 이러한 기존의 CSCW 시스템의 한계를 극복하고 heterogeneous 환경 하에서 성능 제약에 구애받지 않고 쓰일 수 있도록 객체 지향적으로 설계한 구조적 모델을 제시하고자 한다. 본 모델은 분산 환경의 하나인 DOVE에 기반해서 설계된 것으로서 이전까지의 시스템들의 제약된 기능을 모두 보완해 줄 수 있는 방향으로 설계되었다.
모든 모듈들은 분산 객체로서 만들어지게 되고, 분산 객체들간의 원격 호출(Remote method invocation)을 통해서 데이터를 교환하고, interaction을 취할 수 있다. 그리고 같은 타입의 분산 객체들을 하나의 그룹으로 묶어서 객체 그룹 원격 호출(Object group method invocation) 을 함으로써 기존에 객체들간의 유니캐스트 방식으로 멀티미디어 데이터를 전송해 왔던 시스템들에 비해서 효율적인 CSCW 시스템을 구축할 수 있다.
본 논문은 다음과 같은 차례로 쓰여졌다. 먼저 2장에서는 기존의 관련 연구들을 소개하고 문제점들을 지적한다. 3장에서는 본 시스템의 구조를 설명하고 4장에서는 본 시스템을 바탕으로 해서 진행되는 공동 작업의 시나리오를 설명한다. 그리고 마지막 5장에서는 결론을 짓고 향후 나아갈 방향을 모색한다.

2. 관련연구

CSCW 관련 연구는 최근 10여년간 꾸준히 있어 왔고, 각각은 나름대로의 장점을 가지고 있다. 먼저 collaborative visualization groupware 인 Shastra[1]의 경우 RPC(Remote Procedure Call)를 기반 환경으로 해서 개발되었다. 이 때문에 객체지향성격이 크게 약하다는 문제점을 가지고 있다. Web을 기반 환경으로 한 자바 애플릿 groupware 의 하나인 JETS(Java-Enabled Telecommunication System)[2] 의 경우 자바 기반이므로 기본적으로 heterogeneity 문제를 해결했고 객체지향적인 성격을 띄고 있지만, 인터프리터 언어인 자바의 기본적인 성능 문제를 드러

내고 있다. 그리고 Netmeeting이나 CUSeeMe[3]등 상용화 목적으로 개발된 시스템들은 대부분이 처음부터 Microsoft Windows 환경에 기반해서 설계되었기 때문에 분산 프로그래밍의 최대 이슈인 heterogeneity 문제를 떠안고 있는 형편이다.

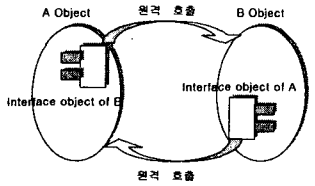

그림 1. 분산객체에 기반한 peer to peer 원격 호출 모델

따라서 이와 같은 각 시스템의 약점들을 모두 보완할 수 있는 방법으로 분산 프로그래밍 모델이 제안되었고 그 중에서 가장 널리 쓰이고 있는 것이 CORBA[4](Common Object Request Broker Architecture)이다. CORBA는 범세계적으로 인정 받고 있는 국제 표준으로써 heterogeneous 환경에서 객체 지향적인 설계를 가능하게 해주는 분산 프로그래밍을 위한 미들웨어로서 현재 널리 응용되고 있다.

3. 공동작업을 위한 프레임웍 설계

본 모델에서는 CORBA의 역할을 해줄 분산 프로그래밍 미들웨어로 본 연구실에서 개발한 DOVE(Distributed Object-oriented Virtual Environment)[5] 를 사용했다. DOVE는 CORBA와 마찬가지로 분산 객체간의 원격 호출로 정보를 주고 받는 분산 프로그래밍 미들웨어의 하나로서, CORBA의 장점인 heterogeneity, 객체지향성을 모두 충족시켜 주고 있고, 더 나아가 Object group의 개념을 가지고 있다. DOVE에서는 같은 타입의 객체들을 하나의 그룹으로 묶어서 객체 그룹 원격 호출을 멀티캐스팅 방식으로 수행할 수 있도록 지원해 주므로, DOVE의 Object group 개념을 이용하면 보다 효율적인 CSCW 시스템을 설계할 수가 있다.
분산 객체 모델에 근거해서 공동 작업 환경을 설계하기 위해서 다음과 같은 객체들을 정의했다. 각 객체들의 역할은 다음과 같다.

Collaboration Manager(CM) : 각 호스트마다 1개씩 존재하는 객체로서 모든 호스트에 있는 FrontEnd, SessionManager등의 정보를 갖고 있고, SessionManager 리스트 관리, FrontEnd 리스트 관리, SessionManager 객체의 생성, 종료 그리고 FrontEnd 객체의 등록, 삭제 등의 일을 수행한다. 즉 공동 작업 환경의 메니저 역할을 하는 객체이며 다음과 같은 원격 호출 메소드들을 제공해 준다.

Register - FrontEnd가 공동 작업 환경에 참여하고자 할 때 호출
Unregister - FrontEnd가 공동 작업 환경에서 떠나고자 할 때 호출
Createsession - FrontEnd의 요청에 따라서 SessionManager 객체 생성
Terminatesession - FrontEnd의 요청에 따라서 SessionManager 객체 생성
Getsessionlist - 요청하는 FrontEnd 에게 SessionManager 리스트 보내줌
Getfrontendlist - 요청하는 FrontEnd 에게 FrontEnd 리스트 보내줌

Session Manager(SM) : 하나의 세션마다 1개씩 존재하는 객체로서 세션에 조인한 FrontEnd의 정보를 관리하고, 각 FrontEnd 의 억세스 레벨을 결정하며, floor 턴(사용자가 공유 데이터를 사용하는 순서)을 결정하는 역할을 한다. 즉 하나의 세션 내에서 메니저 역할을 하는 객체이며 다음과 같은 원격 호출 메소드들을 제공해준다.

Joinsession - FrontEnd가 세션에 조인하고자 할 때 호출
Leavesession - FrontEnd가 세션에서 떠나고자 할 때 호출
Requestfloor - FrontEnd가 floor를 요청할 때 호출
Releasefloor - FrontEnd가 floor를 반환할 때 호출

FrontEnd(FE) : 한 명의 사용자마다 1개씩 생성되는 객체로서 실제 사용자와의 인터액션을 하는 사용자 인터페이스 객체의 역할을 한다. 응용 프로그램을 만들고 하는 사용자는 공동작업에 필요한 기본적인 메소드만 구현된 BaseFrontEnd 클래스를 상속받아서, 자신에게 필요한 객체를 구현해서 사용할 수 있다. FrontEnd에 구현된 GUI를 통해서 사용자로부터 직접 GUI 이벤트를 받아서, 이 이벤트의 콜백함수(callback function)에 원격 호출을 하는 모듈을 넣는 방식으로 구현할 수 있다.
FrontEnd는 다음과 같이 서로 다양한 타입의 데이터를 주고 받을 수 있는 원격 호출 메소드들을 제공해 준다.

SendMessage - FrontEnd 끼리 특별한 메시지를 주고 받을 때 호출 (공동작업 control message)
SendText - FrontEnd끼리 텍스트 데이터를 주고 받을 때 호출
SendFile - FrontEnd끼리 파일을 주고 받을 때 호출
SendImage - FrontEnd 끼리 이미지 데이터를 주고 받을 때 호출

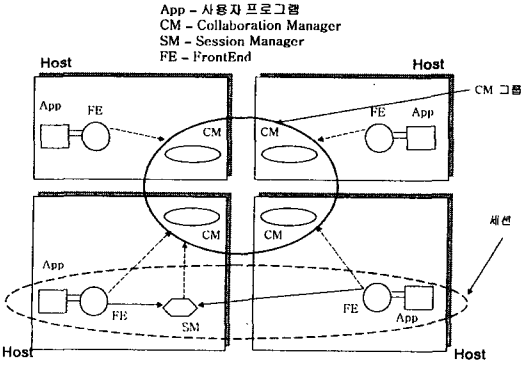

그림 2. CM, SM, FE 객체로 구성된 공동작업 환경

그림 2는 CM, SM, FE 의 객체들로 구성된 공동 작업 환경을 보여준다. 그림에서 호스트마다 1개씩 존재하게 되는 CM은 그룹으로 묶여질 수 있게 되고, 따라서 DOVE에서 지원해주는 객체 그룹 원격 호출기능을 이용해서 단 한 번의그룹 메소드 호출만으로 그룹 내의 모든 CM 객체들에게 정보를 보내줄 수 있게 된다. 그림 3은 실제 세션이 시작됐을 때의 상황을 보여준다. 하나의 세션에 조인한 각 FE들은 하나의 그룹으

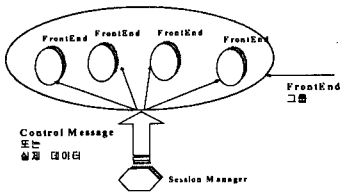

그림 3. 실제 세션 내에서의 정보의 이동

로 묶여질 수 있게 되고, 역시 한 차례의 그룹 메소드 호출만으로 그룹내의 모든 FE 객체에게 정보를 보내줄 수 있게 된다.
하나의 세션 내에서 주고 받는 데이터 타입은 무척이나 다양하다. 기본적으로 보내지는 데이터 타입은 크게 공동 작업 환경 관리를 위해서 필요한 콘트롤 메시지(Control message)와 공동 작업 환경에서 사용자들끼리 주고 받게 되는 실제 데이터의 두 가지로 나눌 수 있다. 사이즈가 작은 콘트롤 메시지나 텍스트 데이터를 보내는 경우라면 문제가 안 되겠지만, 최근의 멀티미디어 공동 작업 환경을 생각해볼 때 상당수의 데이터는 큰 사이즈의 그래픽, 오디오, 비디오 등의 데이터가 된다. 이 데이터들을 DOVE에서 제공해주는 객체 그룹 원격 호출을 이용해서 단 한 번에 다수의 사용자에게 멀티캐스트 방식으로 전송해 줄 수 있으므로 크게 성능을 향상시킬 수가 있게 된다.

4. 공동 작업 수행의 실제적인 시나리오

사용자는 우선 자신의 FE 객체를 하나 생성시키면서 공동 작업을 시작하게 된다. 우선 로컬 CM 객체를 생성시키고 로컬 CM 객체를 CM 그룹에 등록시킨다. 로컬 CM 객체를 생성시킴으로써 사용자 측의 호스트는 공동 작업을 할 준비를 끝내게 된다.
이제 하나의 FE 객체에서 로컬 CM 객체로 원격 호출을 해서 현재 진행중인 세션의 정보, 사용자의 정보를 알아 올 수 있다. 그러면 사용자는 세션리스트를 보고서 자신이 조인하고 싶은 세션을 선택하거나 또는 로컬 CM 객체에 요청해서 자신이 하나의 세션을 별도로 생성할 수 있다.
이렇게 해서 세션에 들어가게 되면 이 때부터는 SM 객체의 관리 하에서 다수의 FE들이 공동 작업을 수행하게 된다. 사용자가 발생시킨 모든 원격 호출은 먼저 SM 객체로부터 허가를 받아야 한다. SM은 그 사용자가 그 원격 호출을 발생시킬 수 있는지를 체크한다. 만약 SM 객체에서 사용자의 원격 호출을 허락하면, 사용자가 보낸 메시지 또는 데이타는 다른 모든 FE들에게 객체 그룹 원격 호출 방식(Object group remote invocation)으로 보내지고, 다른 사용자 측에서는 보내진 메시지 또는 데이터를 받게 된다.
그림 4는 본 모델에 기초해서 구현된 CSCW Visualization 응용 프로그램을 한 사용자 측의 화면에서 본 그림이다. 다수의 사용자들에게 그래픽 데이터를 보내줘야 하는 경우, 객체 그룹 원격 호출을 사용해서 멀티캐스트 하는 방식을 적용해서 성능을 최대화했다.

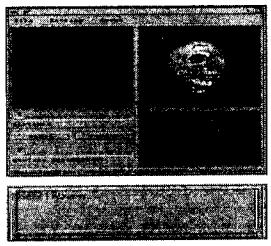

그림 4. CSCW Visualization 응용 프로그램

5. 결론

본 논문에서는 분산 환경에서 공동 작업을 위한 응용 프로그램이 갖춰야 할 모델을 제시했다. 먼저 앞에서 기존의 여러 CSCW 시스템 모델들을 소개했고 그것들의 한계점을 지적했다. 그리고 본 모델이 기존 모델들에 비해서 가질 수 있는 장점을 소개했고, 모델의 구조적인 아키텍쳐와 Collaboration Manager, Session Manager, FrontEnd 분산 객체의 역할, 그리고 본 모델을 적용했을 때의 공동 작업 시나리오를 소개했다.
본 모델의 장점을 다시 한번 부연하자면, 본 모델은 먼저 모든 모듈들이 분산 객체로 구현되어 있어서 완벽한 객체 지향성을 띈다. 또한 분산 미들웨어를 기반으로 설계되었으므로 platform independence의 장점을 지니게 되고, 마지막으로 하부 분산 미들웨어가 Object group 서비스를 제공해 준다면, 객체간의 멀티캐스팅이 절실히 필요한 CSCW 시스템을 제작하는데 있어서 본 논문에서 제시한 모델은 최적이 될 수 있다.
아직 부족한 점은 현재까지 본 모델에 기초해서 작성된 응용 프로그램이 많지가 않다는 점이다. 앞으로 본 모델에 기초해서 각종 CSCW 응용 프로그램을 제작하면서 문제점을 고쳐나가는 일이 필요하다.

6. 참고문헌

[1] Vinod Anupam, Chandraji L. Bajaj," Collaborative Multimedia Scientific Design in Shastra ", Proc . Of the ACM International Conference on Multimedia, August.1993
[2] S. Shirmohammadi and N.D. Geroganas, "JETS: a Java -Enalbed Telecommunication System", Proc. IEEE Int ernational Conference on Multimedia Computing and Systems, pp. 541 -547, June 1997
[3] *CuSeeMe World.* http://www.cuseeme.com/
[4] Object Group Management Inc., The Common Object Request Broker: Architecture and Specification, OMG Document Revision 2.3, December 1998.
[5] H.D.Kim, " DOVE: Distributed Ob ject-oriented Virtual computing Environment ", Thesis for the degree of Doctor, December 1999