컴퓨터활용교육/훈련을 위한 CSCW 모형 개발

심 부성
시스템공학연구소 시스템응용연구부

The Development of CSCW Model for Distance Education/Training

B.S.Shim
Systems Engineering Research Institute

요 약

Network를 통하여 분산되어 있는 멀티미디어 PC 들의 사용자 간에 개별/공동학습 및 저작을 지원하기 위한 CSCW 모형에 대하여 논한다. 이 응용 모형은 사용자나 저작자들이 어떻게 협력할 수 있는지, 다사용자 간의 상호작용을 지원할 수 있는 모드에는 어떤 것이 있는지를 기술한다. 또한 통신과 협력기능을 갖는 하부구조를 기반으로 멀티미디어 인터페이스를 통한 공유작업공간을 구축하고, 협력하여 복잡한 문제를 해결하는 환경 설계에 대하여 논한다.

1. 개요

정보통신 기술의 발달로 통신망을 통한 정보의 교환기능이 제공되고, 협동작업의 필요성이 대두됨에 따라 CSCW(Computer Supported Cooperative Work)라는 새로운 응용분야 연구가 활발히 진행되고 있다. 이와 아울러 거리상의 이유와 평생학습의 필요성 등으로 원격학습이 성장일로를 걷고 있는 주요한 교육분야이다.

본 논문에서는 멀티미디어 PC 를 소유하고 있는 원격지의 저작자나 학습자가 통신망에 접속되어 있고, 개별/공동학습 및 저작을 지원받기 위해서 요구되는 CSCW 모형에 대하여 논한다. 지역적인 관점에서는 투명하나, 시간적인 관점에서는 동기적 모드에서 수행되는 공동작업을 전제로 멀티캐스트 방식으로 참석자 간에 전송하는 경우에 국한하였다.

클라이언트와 서버간 메시지의 전송을 담당하는 수송계층의 프로토콜은 TCP/IP 의 연결지향 방식인 TCP 와 비연결형 방식인 UDP 를 혼용하며, 네트워크 상의 프로세스 간 통신을 구현하기 위하여 WinSocket Library 를 이용한다고 전제하였다. 원격 개별/공동학습 및 공동저작을 위한 CSCW 구조를 계층별로 분류하여 설명하였으며, 공동작업을 위한 실시간 환경을 예를 들어 기술하였고, 공동작업 모드에서 공동으로 하이퍼미디어를

이용할 수 있는 시나리오를 예시함으로써 CSCW 모형을 개발할때 세부적으로 어떤 사항을 고려할 지 연구하였다. 본 논문에서는 저작물이 학습환경에서 프리젠테이션되어야 할 경우 필요한 멀티미디어 동기화 모델과 OODB 기술개발에 관한 내용은 다루지 않았다.

2. CSCW 모형

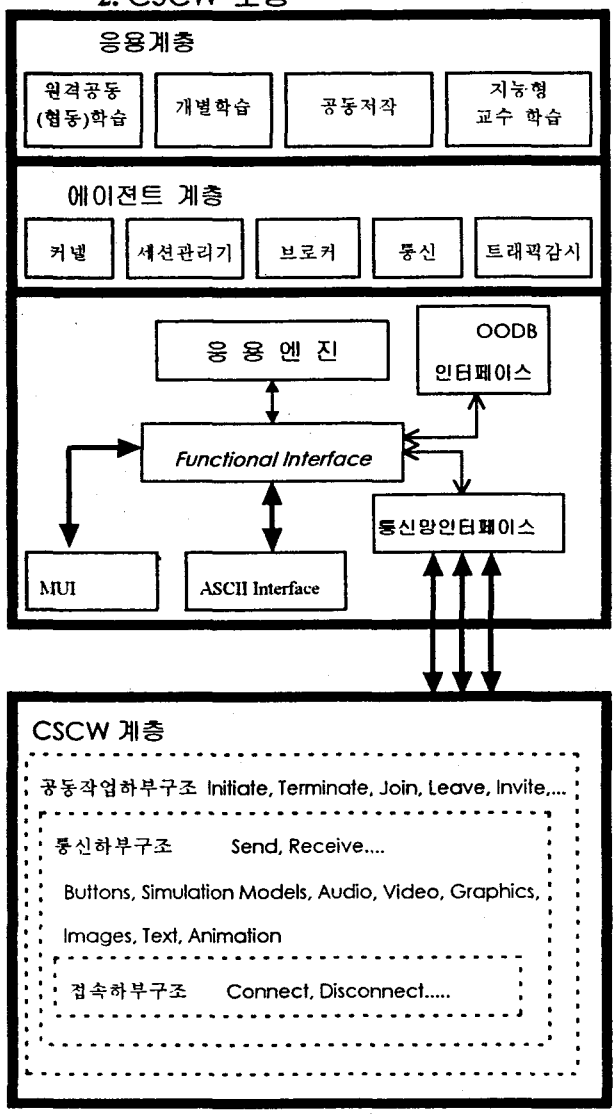

그림 1. 공동작업 모형

CBE/CBT 를 위한 CSCW 모형은 CSCW 계층, 응용엔진, 에이전트 계층, 응용계층으로 분류할 수 있다.

2.1 CSCW 계층

통신 하부구조는 다른 종류의 통신망 Events 에 대한 콜백함수를 지정할 수 있으며, 적당한 용법지시서를 지정하는 것에 의해 다른 종류의 서비스가 다른 인터페이스에서 제공될수 있다.

2.2 응용엔진

응용에 맞는 핵심부를 일컫는다.

2.3 에이전트 계층

가. 커넬

커넬은 메시지를 route 하는 관리도구로서 실시간 환경을 유지해 주고, 상호협조 커넬 프로세스들의 그룹을 구성해 준다. 분산되어 있는 도구들의 인스턴스에 대한 모든 정보를 유지하면서 공동작업 환경에 관련된 각종 활동들을 추적한다. 크게 3 가지의 **Facility** 를 제공하며, **Directory facility** 는 진행되는 공동작업 세션과 그 회원들에 대한 정보를 포함하며, **Location facility** 는 도구들이 실행되고 있는 곳에 대한 접촉 정보를 제공하고, **Routing facility** 는 도구 인스턴스들 간에 데이타와 제어 정보의 전송을 가능하게 한다. 커넬은 다음의 환경 유지 요청을 지원한다.

•Register : 실시간 공동작업환경에 들어가기를 요청하는 경우로써, 커넬프로세스는 상호간 통신하며 디렉토리, 위치, 라우팅 정보를 교환한다.

•Unregister : 공동저작 환경을 떠나기를 요청하는 것이며, 비정상적으로 종료된 도구들은 자동으로 등록 해제된다.

•Terminate : 다른 도구의 종료를 일으

키도록 요청하는 경우임

•Message : 다른 도구로 데이타나 제어 메시지를 보낼것을 요청

•Session Start : 새로운 공동작업 세션을 시작할 것을 요청

•Join : 진행되는 공동작업 세션에 참여할것을 요청

•Invite : 진행되는 세션에 참여하기 위해 도구를 초청할것을 요청

나. 세션관리기

세션들은 동기적인 다사용자 공동작업이나 회의 환경의 인스턴스들로써 세션관리기는 세션을 유지하고 접속의 상세와 세션관리, 상호 작용제어 및 액세스 규제를 담당하는 관리도구의 역할을 담당한다. 공동작업 그룹에 속해 있는 회원들을 추적하고, 공동작업에서 공유된 객체들의 저장소 기능과 동기적인 다사용자 회의 시나리오에서 정보교환에 필요한 멀티캐스트 기능을 지원한다. 다사용자 상호작용의 결과로써 발생되는 대립(충돌)을 해결하는 제약관리 서브시스템을 포함하며, 상호운영의 일관성 유지 기능이 필요하다. 동기적인 여러 파티의 상호작용의 제어하는 규제적 서브시스템이 요구되며, 순서 인계에 입각한 발언권 제어기능이 필요하다.

모든 세션관리기는 다음의 세션제어 요청을 서비스하기 위한 기능이 필요하다.

•Format : 세션포맷(formal 혹은 informal)을 설정할 것을 요청

•Capabilities : 액세스 규제 능력을 설정하기를 요청

•Interaction Mode : 세션에 대한 상호작용 모드(regulated mode 혹은 free mode)의 설정을 요청

다음의 3가지 서비스는 regulated mode의 경우에 해당되는 경우이며, free mode의 경우에는 세션 참가자들에게 할당된 능력에 의해 상호작용이 규제된다.

•Request Floor : 세션에 대한 발언권을 얻기를 요청

•Release Floor : 세션에 대한 발언권을 해제 요청

•Assign Floor : 세션에 대한 발언권 할당을 요청하는 경우이며, leader는 발언권 제어가 가능하다.

•Invite : 진행되는 세션으로 도구를 초대할 것을 요청

•Join : Request to join an ongoing session.

- 도구들이 진행되는 세션에 동참을 요청

•Remove : 리더는 세션으로부터 도구를 제거할 것을 요청

•Leave : 도구가 회원이 되고 있는 세션을 떠날것을 세션관리기에게 요청

•End : 리더가 공동작업 세션을 종료할 것을 요청하는 메시지

하나의 도구가 세션 start 메시지를 지역 커넬로 보낼때 공동작업 세션이 시작된다.

시작하는 도구가 세션리더가 됨.

◦ 한 도구는 포맷메시지를 사용하여 세션 포맷을 설정한다.

참가자가 초대된 경우는 Formal, 어떤 도구가 동적으로 회의에 참가한 경우 Informal.

◦ 리더는 Capabilities 메시지를 사용하여 세션중에 있는 공동작업 활동에 대한 다른 참가자들의 능력을 할당한다.

◦ 세션에 대한 상호작용 모드는 Interaction mode 메시지를 사용하여 지정된다.

다른 공동작업 과제들에 대한 어플리케이션 지정의 세션 관리기들은 애플리케이션에 독립적인 접속, 통신 및 공동작업 제어기능을 제공하는 세션관리기로부터 생성된다.

다. 브로커

브로커는 분산 및 타스크 중매 역할을 담당한다. 여러 다른 서비스들에 대한 서버 도구들의 많은 인스턴스들을 생성하며, 도구들은 여러 서비스 요청을 하나의 브로커에게 보낸다. 브로커는 복수의 요청을 서비스하기 위해 접속된 서버들의 집합을 사용하며 그 결 과를 클라이언트 도구들로 보낸다.

•Start : 서버도구를 시작할 것을 요청 ; 서버 인스턴스들의 생성을 제어

•Stop : 서버도구를 종료할 것을 요청

•Connect : 네트워크를 통해 기존의 서버도구로 접속 요청 ; 타스크에 사용되어야 할 서버 인스턴스들을 선택

•Disconnect : 서버접속 종료 요청

•Service : 계산타스크의 서비스를 요청 ; 적절한 서버로 요청을 forward 함. 보다 복잡한 시나리오에서 도구들은 커다란 계산타스크들을 애플리케이션 지정의 브로커들로 보내고 여기서 과제들을 독립적인 서브타스크들로 나눈다. 이 서브타스크들은 서버도구들의 접속된 풀을 사용하여 서비스 된다. 결과들은 함께 넣어져 요구하는 도구들로 전송된다. 애플리케이션 지정의 브로커들은 기본적인 브로커로부터 생성된다. (새로운 요청들을 이해하도록 메시지집합을 확장함으로써.)

라. 프론트

프론트는 프로토콜 메시지를 이해하고 커넬, 세션관리기, 브로커 및 표준서비스와 인터페이스하기 위한 요청을 하는 '공동작업을 알아채리는 최소의 도구'인 기본적 프론트 엔드를 지정하는 것에 의해 생성된다.
다른 도구들 간의 공동작업환경 지원을 위한 상호작용관계를 담당한다.

마. 툴킷

동시의 공동저작, 교육/훈련, 프리젠테이션 및 게임 등과 분산된 문제해결을 담당한다.

바. 서비스

통신 및 매체편집도구로써 다른 매체 종류를 이해하는 인터페이스 에이전트로 동작한다. 동기적인 n-방향 텍스트의 통신도구, 간단한 2 차원 그림의 생성과 표시, 비디오와 그래픽 애니메이션의 편집과 재생기능, 오디오 정보의 녹음과 재생기능 및 이미지화일의 뷰기능, 문제유형별 지원을 위한 상호작용 등 멀티미디어 정보(text, image, audio, video, simulation model, hotword, graphics, animation)의 사용과 전송을 가능하게 하며, 상호작용의 품질과 효과적인 상호협조를 지원한다.

3. 공동작업의 실시간 환경

- 실시간 시스템의 합성
- 프론트뿐만 아니라 커넬, 세션관리기, 브로커들의 인스턴스들의 집합을 구성
- 도구들은 동적인 공동작업환경을

지원하기 위해 상호작동 한다.

3.1 통신과 세션 개시

프론트들은 멀티미디어 통신판넬을 셋업하기 위해 시스템의 메시지 기능과 접속하여 커넬의 Directory와 Location facility를 사용한다. 통신판넬은 기본으로 텍스트 대화는 허용한다. 공동작업환경은 호스트들 상의 오디오, 비디오와 그래픽 기능의 유용성을 추적한다. 이는 도구 사용자들로 하여금 보다 높은 수준의 기능이 사용될 수 있다면 통신판넬을 적절히 구성할수 있게 한다. 제어판넬은 대화를 수행하기 위해 서비스 도구들을 구동시킨다.

(예) 공동작업 환경의 통신 facilities는 세션개시 절차에서 매우 유용하다.

① 한 세션은 도구 사용자가 Session Start 요청을 보내는 것에 의해 시작된다.

② 커넬은 세션관리기를 시작한다. 세션생성자가 세션리더가 되며 자동으로 회의에 참가하게 된다.

③ 세션리더는 시스템에 공동작업 환경의 다른 활성화된 사용자들을 발견하기 위해 물어보고, Invite 요청을 함으로써 그들이 세션에 동참하도록 초청한다.

④ 초청받은 자들은 세션참가에 대해 그들의 커넬을 통해 프롬프트를 받으며, 통신판넬은 각각의 초청된 사이트에 생성된다.

⑤ 세션리더는 초청된 그룹으로 방송할 수도 있고, 각 개인과 그의 통신판넬을 통해 통신할 수도 있다. 초청은 초청된 자가 참가할 때 가지게 될 능력을 지정한다.

⑥ 통신은 초청의 수락이나 거절로 급격한 집중을 허용한다. 초청에 수락하는 자는 회의에 참가하게 된다. (초청받은 자들은 여러 회의에 동시에 참가자가 될 수 있다.)

⑦ 도구의 사용자들은 관심있는 세션을 찾기 위해 공동작업 환경에 질의해 볼 수 있으며, 허락을 얻기 위해 Join 메시지를 사용한다.

⑧ 요청은 커넬과 세션관리기를 통해 세션리더에게 routed 된다.

비공식적인 세션에 대해서는 어떤 도구도 "개방된" 세션에 참여하는게 허용된다.

공식적인 "폐쇄된" 세션에 대해서는 요청자와 세션리더간에 협상할 수 있게 함.

3.2 공동작업의 상호작용

가. Regulated Mode

발언권 요청 메시지를 사용하여 공동작업 내용을 바톤전달하는 방식으로써, 마스터-슬레이브 상호작용, 모든 상호작용은 바톤을 갖고 있는 마스터 사이트에서 발생하고 적절한 정보는 모든 슬레이브 사이트로 중계된다. 애플리케이션 개발자들은 낮은 계산과제들의 입력과 고도의 계산결과(출력)를 분배하기 위해서 세션관리기의 방송기능을 활용한다. 바톤 전달방식은 Non-Preemptive 이거나, Preemptive 일 수 있다. 리더는 발언권 할당 메시지를 사용하여 차례를 정해줄 수 있다.

나. Free Interaction Mode

활동들이 리더에 의해 세션참가자들에 할당된 능력을 통해 규제된다. 동시적인 다수의 참가자 상호작용을 허용하는 공동작업 설정에 있어서 세션 참가자들의 능력을 동적으로 제어하는데 사용된다.

• Access : 사용자가 공유된 내용과 공동작업의 상호작용을 보게할지 말지를 제어("Read" 허용)

• Browse : 사용자가 독립적인 local view를 제어할지 말지를 규제("Execute"허용)

• Modify : 사용자가 공유된 상태를 수정할수 있게 할지 말지를 규제("Write" 허용)

• Copy : 사용자가 공유된 객체를 복사할수 있을지 제어

• Grant : 사용자가 세션규제 동작을 수행할수 있을지 없을지를 제어, 사용자들을 세션에 참여하도록 초청할수 있게 할지를 규정

액세스 규제 정보는 세션관리기가 유지한다. 사이트능력은 리더나 Grant할 자격을 갖고 있는 사용자에 의해 규제된다. 오브젝트 허락은 공동작업 객체들의 소유자에 의해 규제된다. 참가자들과 객체들에 대한 다른 capability와 permission settings은 실행시에 다양한 상호작용 modes를 생성한다.

4. 하이퍼미디어 공동 저작시스템의 이용사례

공동저작시스템에서는 공유된 코스웨어 상의 다른 사람들의 활동을 알아차리는 기능의 지원이 중요하다.

4.1 확인과 록킹 지원

OODB에서의 기본적인 이벤트 확인 구조는 공유된 하이퍼텍스트에 있어 다음처럼 보다 높은 수준의 이벤트(higher-level events)를 추적할 수 있게 해준다.

- 전체 하이퍼텍스트를 생성, 삭제 및 수정하는 일
- 하이퍼텍스트 내의 컴포넌트(기본, 링크 혹은 복합객체)를 생성, 삭제, 수정하는 일
- 컴포넌트 내의 앵커와 속성들을 생성, 삭제, 수정하는 일
- 하이퍼텍스트와 컴포넌트 상의 록변화(Lock changes)

다른 런타임 프로세스에 의해 활성화된 트랜잭션의 시작과 빠져나가는 것(abort)뿐만 아니라 트랜잭션의 시작에 대한 확인을 신청하는 것도 가능하다. 공유된 하이퍼텍스트의 상태와 내용 모두에 있어서의 변화를 사용자가 알아차리게 해 준다. 사용자들은 객체의 수정에 대한 정보를 얻기 위해서 하이퍼텍스트와 컴포넌트 상의 속성뿐만 아니라 확인기록(Notification Log)을 검사할 수 있다. 컴포넌트 수준의 확인 신청은 컴포넌트 브라우져나 특수 편집기에 의해 이루어진다.

컴포넌트 상의 확인 신청에 대한 사용자 인터페이스는 브라우져 윈도우에 표시된다. Notification Menu의 Component subscribe..., Composite subscribe..., Component unsubscribe..., Composite unsubscribe..., Show log, Remove symbol 등의 메뉴항목 중에서 Component subscribe...를 선택하면 대화상자가 나오며, 컴포넌트 수준의 사례화에 대한 확인 신청을 위한 대화상자 내용은 다음과 같다.

1) 신청대상(Subscription for) : 아이콘 이름 입력

2) 이벤트 종류 : Get with Write Lock, Get with Read Lock, Lock Change, Update, Create 등에 대해 on, off 선택 가능.

3) 사용자 : 전체를 대상으로 할 것인지, 일부 사용자에 제한 시킬 것인지 list box 에서 선택할 수 있음.

4) 요구되는 반응의 종류 : Show In Console, Immediate update, Show marks on objects, Play sound 등에 대해 on, off 선택 가능

5) 컴포넌트 레벨 : 객체, 클래스이름 등에 대해 on, off로 선택 가능.

6) 기타 : 취소, 확인 버튼이 있음.

컴포넌트를 다시 가져오는 사용자 인터페이스도 브라우져 윈도우에 표시되며, 벨 표시가 된 컴포넌트는 수정 확인을 받은 것을 나타낸다. .변화를 검사하기 위해서 사용자는 컴포넌트 상에서 'Refetch component'를 작동시킨다. 참고로 화일 메뉴 중에는 Save component, Save All, Refetch component, Refetch All, Get Info, Refetch component, Recompute, Quit window 등의 메뉴항목이 있다.

4.2 공동의 하이퍼미디어 사용을 위한 시나리오

공동 작업모드에서 공동으로 하이퍼미디어를 이용할 수 있는 시나리오를 예시하면 다음과 같다.

시나리오 1: 직접 수정. 갑과 을은 모두 하이퍼텍스트 H1 에 대한 세션을 시작한다. 갑은 컴포넌트 C1 에 대한 write lock 을 얻는다. 을은 read access 로 C1 을 열고, 다른 사람들에 의해 C1 이 변화될 때 직접 수정으로 신청한다. 지금 C1 에 대한 사례화는 수정 이벤트 확인이 나타날 때마다 OODB 내에 저장된 C1 의 가장 최근 버젼을 다시 가져오는 것(reGetting)에 의해 그 자체를 자동으로 수정한다. 갑은 변경을 하고 그것들을 데이타베이스에 알림으로써 직접 수정이 을의 스크린에 나타나도록 한다.

시나리오 2: 이벤트 로깅(Logging). 몇 명의 사용자(갑, 을, 병)가 하이퍼텍스트 H1 에 대한 세션을 함께 시작했다. 을은 C1 을 read access 로 열었으며, C1 에 대한 변화를 로깅하도록 신청하였다. 갑은 C1 을 write lock 으로 열었다-을은 콘솔에서 "갑이 '95 년 11 월 30 일 목요일 11 시 28 분 08 초에 write lock 으로 C1 을 열었다."라는 메시지를 통해 확인할 수 있다. 갑은 수정하여 C1 을 저장한다.-을은 콘솔에서 "갑이 '95 년 11 월 30 일 목요일 12 시 00 분 11 초에 C1 을 수정하였다."는 메시지를 통해 이를 확인할 수 있다. 갑은 write lock 을 해제한다.-을은 "갑이 '95 년 11 월 30 일 목요일 12 시 01 분 00 초에 C1 에 대한 write lock 을 해제하였다." 는 것도 확인할 수 있다. 다음에 병은 write lock 으로 C1 을 연다.-을은 "병이 '95 년 11 월 30 일 목요일 13 시 10 분 08 초에 write lock 으로 C1 을 열었다."는 것을 확인할 수 있다.

시나리오 3: 하이퍼텍스트에 관한 확인을 알아차림. 갑과 을은 모두 하이퍼텍스트 H1 에 대한 세션을 시작하면서 H1 을 누가 사용하는지에 대한 확인을 신청하였다. 그 결과는 H1 에 대해 'Get' operation 을 수행해온 사용자들의 목록을 콘솔에서 보여주는 것이다. 지금 병이 H1 에 대한 세션을 시작하자 병의 Id.가 갑과 을의 콘솔에 나타난다.

시나리오 4: 컴포넌트와 복합컴포넌트에 관한 확인을 알아차림. 몇명의 사용자(갑, 을 ,병)가 을에

게 책임이 있는 같은 "case"에 대해 작업하고 있다. 그들은 해당 하이퍼텍스트 H1에 대한 세션을 시작했다. 을은 이 "case"의 현재 활성화된 문서에 해당하는 C1, C2, C3 및 C4 컴포넌트를 포함하는 복합컴포넌트 CS1을 만든다. 을은 CS1에 포함된 컴포넌트들에 발생되는 변화에 대한 확인을 신청하기 위해 'composite subscribe . . .' 메뉴명령어를 사용한다. 그 결과로 다른 사용자들이 C1, C2, C3 및 C4에서 수정, 특변화 등을 실행할 때 을이 이를 확인할 수 있다는 것이다.

시나리오 5: 특별한 유형의 객체들을 생성/삭제하는데 대한 확인. 몇 명의 사용자들(갑, 을, 병)은 하이퍼텍스트 H1에 대한 세션을 시작한다. 을은 H1에 있는 텍스트 컴포넌트 생성의 로깅을 신청한다. 갑은 H1의 새로운 텍스트 컴포넌트 C1을 생성하고, 그 내용을 편집하고 저장한다.- 을은 콘솔에서 "갑이 '95년 11월 30일 목요일 11시 28분 08초에 텍스트 컴포넌트 C1을 생성했다."는 메시지를 통해 이를 확인할 수 있다.

시나리오 6: 록 교환(Lock exchange). 갑, 을 및 병은 하이퍼텍스트 H1에 대한 세션을 시작한다. 갑은 컴포넌트 C1에 write lock을 얻고 있다. 을과 병은 read access로 C1을 연다. 병은 C1의 변화에 대한 로깅을 신청한다. 어떤 시점에 을은 "Change lock . . ."이라는 메뉴명령어를 사용할 수 있다. 이는 을에게 갑이 C1에 대하여 write lock을 가지고 있다고 알려준다. 그러면 을은 갑에게 전화를 하여 갑이 가까이 그의 수정부분을 저장하고 C1에 대한 write lock을 해제해 줄 것을 요청하게 된다. 갑은 그것에 동의하고, 수정부분을 저장하여 C1에 대한 write lock을 단지 read access로 바꾸게 된다. 을은 곧바로 write lock을 얻게 된다. 이 교환 중에 병은 "갑이 수정부분을 저장했고, C1에 대한 write lock을 해제하였으며, 을이 C1에 대한 write lock을 획득하였다. 갑은 C1에 대한 모든 변화에 대한 로깅을 신청하였으며, 을은 몇가지 수정을 하고 그것들에 관하여 갑과 병에게 모두 확인 메시지를 보내도록 OODB에 요청하였다."는 확인 메시지를 받았다.

시나리오 7: Simultaneous linking. 갑과 을은 각각 하이퍼텍스트 H1에 대한 세션을 시작하여 read access로 텍스트컴포넌트 C1을 열고, C1에 대한 직접 수정을 의뢰한다. 갑은 C1에 있는 텍스트 영역으로부터 컴포넌트 C2에 있는 텍스트 영역으로 public link를 생성한다. 갑은 그 변경사항을 의뢰하여 C1의 앵커 목록 상의 write lock에 short upgrade를 만들고, 새로운 링크마커로 C1에 대한 뷰를 바로 수정하게 된다. 을은 C1으로부터 컴포넌트 C3로 다른 public link를 만들어 변경을 의뢰하게 되고, C1에 대한 갑의 뷰는 마찬가지로 직접 수정된다.

4.3 록과 확인을 다루는 방법

록킹과 확인 구조는 OODB에서 일반적 수준으로 개발되어 왔다. 따라서 우리의 일반적인 스토리지 클래스, 하이퍼텍스트 및 컴포넌트는 더 확장될 필요가 없다. 그러나 실시간 클래스, 세션 및 사례화는 OODB 서버(Server)로부터 받은 이벤트 확인을 취급하고, 전파하기 위한 operation과 함께 확장되어 왔다. 이벤트 확인은 런타임 프로세스에 의해 해석되고 다시 그들의 클라이언트들(편집기들)에게 확인사실을 전파한다. 그 확인들은 전형적으로 응용들과/혹은 하이퍼미디어 복합객체 브라우져에 디스플레이된다. 응용들은 또한 사용자가 관심있는 이벤트 확인들에 위탁할 수 있는 사용자 인터페이스를 공급할 수 있다. 혹은 어떤 종류의 이벤트 확인을 자동으로 말기고 조정할 수 있다.

세션, 사례화 및 링크마커 개념은 우리의 설계에서 모델 기능에 해당하는 오퍼레이션을 갖는 클래스로 나타난다. 사용된 프로그래밍 언어는 클래스의 block-structured nesting을 지원하며, 세션 관리기(SessionMgr) 클래스에서의 실시간 클래스를 encapsulate하기위해 사용된다. 런타임 프로세스는 클래스로부터 사례화된 객체를 구성한다. 세션 객체는 항상 액세스되어야할 해당 하이퍼텍스트 객체에 대한 OODB와의 트랜잭션을 시작한다.

록을 다루는데 있어 상대적으로 전체 하이퍼텍스트나 단일 컴포넌트에 대한 록을 변화시키게 해 주는 세션과 사례화 클래스 상의 changeLock 오퍼레이션을 정의해 왔다. 그러나 changeLock 오퍼레이션은 쉽게 임의의 수준의 크기를 갖는 하이퍼텍스트 객체 상에서 change locks할 수 있는 가능성을 제공하도록 추가될 수 있다. 예를 들면 링크마커가 개별적인 앵커들 상의 록을 바꾸는 기능을 지원하도록 확장되었다. 확인을 다루는 것도 마찬가지로 설계된다. 실시간 클래스들은 subscribe하고 unSubscribe하는 오퍼레이션으로 확장된다.

ReGet 오퍼레이션은 스토리지 객체(예를 들면, OODB 서버로부터 하이퍼텍스트나 컴포넌트)의 새로운 버젼을 검색하도록 하기 위해 도입된다. 예를 들면 사례화 수준에서 해당 컴포넌트의 최신 버젼을 검색하는 것이 가능하다. 전형적으로 ReGet을 해야 할 필요성이 특정 사례화에 대한 컴포넌트가 바뀌었다는 확인을 런타임 프로세스가 받을 때 발생한다. ReGet은 또한 이벤트 확인에 대한 반응으로써 런타임 객체에 의해 자동으로 불리워질 수 있다.

결과적으로 런타임 프로세스가 확인을 받을때 적합한 반응을 성취하기 위해 reaction classes가 도입된다. 반응객체는 "private parts"에 보여진 클래스들 중 하나로부터 사례화된다. 의뢰(subscription)를 실행할때 확인 상에 실행되어야 할 반응 객체가 등록된다. 각각의 런타임 클래스는 그 수준에서 관련성 있는 각 이벤트 종류에 대한 반응 클래스를 포함한다. 예를 들면, 그 사례화 수준에서 수정반응(update reaction : UpdateR)은 해당 컴포넌트 상의 수정 이벤트를 다루기 위해 도입된다. 런타임 클래스들의 네스팅(nesting)을 반영하는 상속 계층구조로 조직된다. 주된 런타임 클래스들 각각은 그 클래스에 대한 반응들의 유사성을 수집한 추상적인 반응 슈퍼클래스를 갖는다. Nested class에 대한 반응은 항상 enclosing class의 추상적인 반응 클래스의 특수화이다. 이것은 상속 계층구조가 반응으로 하여금 모든 중간 레벨에서 다루어지도록 허용하므로 잇점을 갖는다. 예를 들면 하나의 컴포넌트가 그 사례화가 닫혀진 후에 확인을 받으면 enclosing session은 확인을 취급하게 될 것이고, 결국 세션관리기가 이를 취급하게 될 것이다.

5. 결론

네트워크를 통하여 원격교육/훈련을 실시하는 것은 비디오 컨퍼런스 처럼 유명 강사가 그대로 존재하는 스튜디오 방식과 출퇴한 코스웨어와 효율적인 학습 관리 시스템을 통해 스스로 학습하는 코스웨어 방식이 있다. 물론 향후에는 이 두가지 방식이 합쳐져 보다 효율적인 시스템으로 발전하겠지만 본 논문에서는 후자의 경우를 개별학습/공동학습, 공동저작의 응용시스템을 중심으로 CSCW 모형과 그 실시간 환경에 대하여 고찰하였으며, 공동작업 환경에서 공동으로 하이퍼미디어를 이용할 수 있는 몇가지 시나리오를 예시함으로써 정보통신 시스템 구축상 애플리케이션 층에 속해 있는 원격교육이기는 하나 학습방법이나 내용 및 범위에 따라 CSCW 계층이나 에이젼트 계층등의 상세설계가 대단히 복잡할 수 있음을 쉽게 알 수 있다. 초고속망의 시대로 접어들면서 그 열기는 뜨거운데 초고속망에서 원격교육시스템을 어떻게 구축하여야 할지는 다른 어떤 정보통신 Infra Structure 구축에 못지않게 중요한 분야인데도 불구하고 투자가 미흡한 것이 현실이다.

<참고문헌>

[1]Ellis,C.,Gibbs,S.,Rein,G.,(1991),"Groupware: Some Issues and Experiences", Comm. of the ACM, Vol.34 No.1, Jan 1991, pp.38-58.

[2] Hill,R., (1992), "Languages for Construction of Multi-User, Multi-media Synchronous (MUMMS) Applications", In Brad Meyers (ed.) Languages for Developing User Interfaces, Jones and Bartlett, 1992.

[3] Rangan, V., Vin, H., (1992), "System Support for Computer Mediated Multimedia Collaborations", Proc. ACM Conference on CSCW '92, 203-209.

[4] Bulterman,D.C.A., van Rossum, G. and van Liere, R.A "structure for transportable, dynamic multimedia documents" Proc. of the Summer 1991 USENIX Conf.

[5] 심 부성, 오 영배, "암스테르담 하이퍼미디어 동기화 모델을 이용한 공동저작 시스템", 1994 한국정보처리학회 학회지.

[6] A.W.Bates,"Applications of New Information Technologies in Distance Education for Adult Learning and Higher Education,1995 국제원격교육 워크숍 자료집,3-32.

[7] 정회경, 최경호, 이수연,"그룹웨어를 위한 그룹통신 플랫폼의 설계 및 구현", 1995.9. 한국정보과학회 논문지.