멀티 플랫폼 기반 온라인 응용의 데이터 제어 기법

김진덕† · 진교홍††

요 약

기존의 고정 단말기(PC)에서만 주로 행해지던 각종 다중 사용자 접속 온라인 응용이 최근 PC와 PDA, 휴대폰 등이 공동 작업을 수행하는 멀티 플랫폼 기반 온라인 응용으로 전환되고 있다. 그러나 지금까지 멀티 플랫폼 기반 온라인 응용은 다중 사용자의 공동 데이터의 동시 접근 및 각 클라이언트의 처리 능력과 통신 속도의 비대칭을 고려하지 않아 컨텐츠가 매우 단순한 실정이다.

이 논문에서는 멀티 플랫폼 기반 동시성 제어 기법, 다양한 클라이언트간의 변경 전파 프로토콜, 모바일 클라이언트의 특성을 고려한 객체 관리 기술을 제안하였다. 그리고 멀티 플랫폼 기반 채팅 프로그램을 제작하여 제안한 기법들이 적절히 동작함을 보였다.

Data Control Methods of Online Application based on Multi-Platform

Jin-Deog Kim† · Kyo-Hong Jin††

ABSTRACT

Several multi-user online applications which are operated by the existing fixed terminals(PC) are being changed into online application based on multi-platform operated by the several PC, PDA and mobile phones to perform concurrent works recently. The contents of current online application based multi-platform are, however, very unsophisticated because the applications don't consider the simultaneous accesses of shared data by multi-user and also the asymmetry of computing power and network bandwidth among each client.

This paper proposed the methods of consistency control based on multi-platform, update propagation protocols among diverse clients, object management techniques which take the characteristics of mobile clients into account. We also implemented a chatting application based on multi-platform and it showed the proposed methods perform well.

1. 서 론

국내의 인터넷 사용자수가 2002년 KRNIC의 보고서에 따르면 2,500만명에 이르며, 휴대폰의 보급 또한 폭발적인 성장세를 보였으며, 최근 PDA 사용자 역시 꾸준히 증가하고 있는 추세이다. 이와 같은 각각의 기기들은 유선 또한 무선의 형태로 온라인 응용 프로그램을 운용할 수 있다. 아울러 기존의 PC에서 주로 행해지던 각종 프로그램이 이제 휴대폰이나 PDA와 같은 모바일

† 정 회 원: 동의대학교 컴퓨터공학과 전임강사
†† 정 회 원: 동의대학교 멀티미디어공학과 조교수
논문접수: 2002년 9월 8일, 심사완료: 2002년 10월 23일
* 본 논문은 2002년 부산테크노파크 연구지원(BTP-BDE-9)으로 수행되었음

/무선 단말기로도 확산될 것으로 예상된다[7].

따라서 앞으로 급속히 증가될 것으로 예상되는 모바일/무선 단말기용 온라인 응용을 위한 기반 구축 기술이 개발되어야 한다. 다수의 PC, PDA, 휴대폰을 통해 유무선으로 동시에 수행되는 멀티 플랫폼 기반 온라인 응용 프로그램이 원활히 서비스된다면 그 가치는 보다 증대될 것이다.

그러나 이러한 멀티 플랫폼(PC, PDA, 휴대폰) 기반 온라인 환경은 모바일/무선 단말기의 특성상 다음과 같은 이유로 기존의 온라인 응용 환경을 그대로 활용할 수가 없다. 우선, 각 클라이언트 기기의 사용 가능한 리소스의 차이가 심하고 프로세서의 처리 능력의 편차가 심하다. PC와 같은 고정 단말장치에 비해 휴대폰과 같은 모바일 휴대 장치는 극히 작은 용량의 메모리가 사용가능하며, 중앙 처리 장치의 처리 능력 또한 현격한 차이가 있다. 둘째, 통신 대역폭의 차이가 심하다. 모바일/무선 단말기는 광대역의 통신 대역폭을 사용할 수 없을 뿐더러 열악한 통신 환경으로 인해 새로운 통신 알고리즘이 필요하다.

이 논문에서는 멀티 플랫폼 환경에서 온라인 응용의 원활한 서비스 제공을 위한 인프라 구축 기술을 개발하는 것을 목적으로 한다. 구체적으로 위와 같이 특성을 갖는 멀티 플랫폼 기반 다중 사용자의 동시 접속 온라인 응용 프로그램은 새로운 동시성 제어(Concurrency Control) 기법과 변경 전파(Update Propagation) 기법 및 새로운 기법의 객체 관리 기술을 제안하고자 한다. 첫째, 일반적으로 데이터베이스 관리 시스템에서 동시성 제어를 위해 잠금(Lock) 기법을 사용한다. 그러나 멀티 플랫폼 환경은 통신 대역폭의 편차로 인해 공유 데이터를 서버에서 관리하는 것이 아니라 각 클라이언트에서 중복하여 관리하며 독자적으로 시나리오를 진행하다가 변경 전파를 하는 과정에서 일관성 제어 위반 사실이 발견되므로 새로운 동시성 제어 기법이 도입되어야 한다. 현재 멀티 플랫폼 환경에서 가장 많은 컨텐츠를 제공하는 게임 분야에서도 다중 사용자의 동시 수행에 따른 일관성 제어 문제를 다루는 경우는 거의 없다. 둘째, 응용 프로그램의 진행 정보에 대한 변경 발생시 접속되어 있는 다른 클라이언트에게 통보해야 하는 변경 전파 기법 또한 각 클라이언트마다 통신 기법이 상이하므로 새로이 연구되어야 한다. 셋째, 낮고 다소 불안정한 통신 대역폭을 가진 PDA와 휴대폰과 같은 클라이언트 내에서도 무리 없이 응용 프로그램이 수행되어야 하며, 소용량 저성능의 휴대 단말기에 적합한 객체 처리 기법[8] 또한 반드시 필요하다.

그리고 제안한 기법이 올바르게 동작함을 확인하고자 멀티 플랫폼 기반 채팅 프로그램을 제작하여 테스트한다. 채팅 프로그램은 클라이언트로 PC, PDA, Phone 버전이 각각 존재하며 개발 환경 또한 전혀 다르지만 서로 데이터를 공유하거나 송수신이 가능하다. 이 논문에서 채팅 프로그램을 통해 제안한 기법을 구현하였지만 멀티 플랫폼 기반 다중 접속자 응용인 게임, 온라인 학습, 모바일 커머스 등에서도 쉽게 활용할 수 있으리라 판단된다.

이 논문의 구성은 다음과 같다. 제 2장에서는 온라인 응용에서의 중복 데이터 동시성 제어, 변경 전파 기법 등을 연구한 기존의 연구들에 대해 살펴보고, 제 3장에서는 온라인 응용의 데이터 제어를 위한 전체 시스템 구조 및 이 논문에서 제안하는 멀티 플랫폼 기반 동시성 제어, 변경 전파, 객체 처리 기법 등에 대해 자세히 설명하다. 제 4장에서는 제안한 기법을 온라인 채팅 프로그램에 적용한 예를 들며, 제 5장에서는 결론 및 향후 연구 과제를 다룬다.

2. 관련연구

분산 환경에서 동시성 제어 문제는 전통적으로 동기식 제어[1]와 비동기 제어[3] 접근 방법이 있다. 동기식 제어 기법은 잠금과 2PC를 이용하여 모든 수정을 동기화 시켜서 직렬화(serializability)를 보장하는 방법이다. 이 기법은 공유 데이터의 동시 수정에 의한 불일치 문제(concurrency anomalies)가 발생하지 않는다는 장점이 있으나, 엄격한 제약으로 인하여 응답시간과 성능이 심각하게 저하되는 문제점이 있다. 비동기식 제어 기법은 '1 객체 소유자 : N 트랜잭션' 개념으로서 중복 제어를 한 사이트에서 담당하도록 서버를 두며, 수정은 이 서버에서만 허용한다. 이 후 적당한 시기에 서버의 책임으로

동시성 제어를 위해 수정 전파 통신을 수행하는 방식이다. 이 기법은 동기식 제어 기법에 비해 성능이 향상되었지만 일정 기간 동안 데이터베이스의 일관성이 위배된다는 단점이 있다. 전통적인 동시성 제어 기법에서는 서버에 도착하는 리소스의 요청 시간을 기준으로 하는 동기화가 이루어지는 데 반해, 이 논문에서는 멀티 플랫폼 상에서 수행되는 실시간 온라인 응용을 위해 각 클라이언트에서 리소스를 요청하는 시간을 기준으로 동시성제어를 수행한다. 예를 들어, 시나리오가 동일하게 진행되는 게임, 온라인 교육 응용 등에서는 서버에 도착한 리소스 요청 시간보다는 각 클라이언트 내의 리소스 요청 시간이 더 중요하게 여겨진다는 점을 고려한 것이다.

동시 작업에 의해 공동 작업 환경에서 동기화를 지원하는 방식으로서 병행 엔지니어링(Concurrent Engineering) 방식[2, 4]이 있다. 병행 엔지니어링에서는 사용자들이 동시에 두개 이상의 모듈을 작업하면서 상호 연관된 데이터를 송수신한다. 병행 엔지니어링 방식에서 모듈간 종속성이 존재하면 사용자간 변경 상충(Update Conflict)이 발생하여 작업이 다시 피드백되어 재작업하는 기법이다. 그러나 이 방법은 긴 트랜잭션에 대해 재작업을 수행할 경우 상당한 복구 비용을 요구한다.

공동 작업 환경에서 객체 들에 대해 사용자-제어 잠금에 의해 동기화 작업을 지원하는 방식으로 CSCW(Computer Supported Cooperative Work) 방식[5, 6]이 있다. 이 방식은 소프트웨어 시스템 개발을 위한 동기화된 공동 작업 환경을 제공하며 아울러 다중 사용자들의 동시 참여가 요구된다. 그러나 이 방식은 서로 다른 부분에 대해 사용자들간 동기화된 공동 작업 환경을 지원하지만 같은 부분에 대해서는 여러 사용자간에 발생하는 변경 상충은 해결하지 못한다. 따라서 변경 작업 종료 후에 undo/redo 연산을 해야 하며 긴 시간 작업을 철회해야 하는 문제가 있다. 그러나 이 논문이 다루는 온라인 응용의 실시간 처리를 위한 변경전파기법과는 차이가 있다.

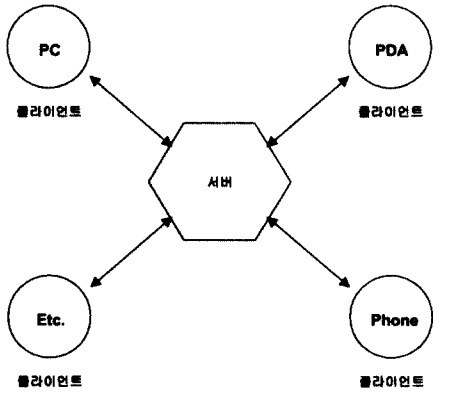
3. 멀티 플랫폼 환경의 데이터 제어

3.1. 멀티 플랫폼 응용의 시스템구조

여기서는 온라인 시스템 구조의 종류를 제시하고 각각의 특징을 살펴본다. 제시하는 구조들은 멀티 플랫폼 온라인 응용에서 항상 고정된 것이 아니라 상황에 따라 적합한 구조를 선택하여 사용하게 된다. 지금부터 온라인 응용이라 함은 멀티 플랫폼 환경의 온라인 응용을 의미한다.

(1) 클라이언트/서버 구조

이 구조에서 서버를 담당하는 컴퓨터는 서버의 역할만을 지속적으로 수행하고 모든 클라이언트는 응용 프로그램의 입력을 서버에 보내며 서버는 이러한 입력을 기반으로 모든 진행 결과를 전체 클라이언트에게 브로드캐스팅한다. 그림 1은 클라이언트/서버 구조를 표현한 것이다. 메시지의 수는 클라이언트의 수에 비례하기 때문에 비교적 많은 수의 클라이언트를 수용할 수 있다[12]. 예를 들어 온라인 게임의 경우 주로 사용자 로그인 및 전체 게임 정보의 변경 전파시에 주로 사용되는 시스템 구조이다. 그리고 3.3절에서 설명할 동시성 제어에도 클라이언트/서버 구조가 사용된다.



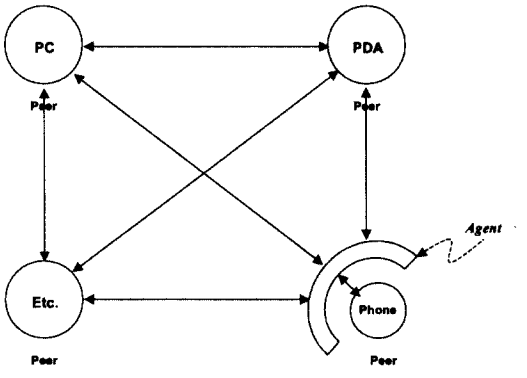
(그림 1) 클라이언트/서버 구조

(2) 피어-투-피어(Peer-to-Peer)구조

이 구조는 온라인 응용에서 각 클라이언트 책임 하에 모든 변경 정보 메시지를 해당 클라이언트에 보내며, 정보를 받은 클라이언트는 개별적으로 연산을 수행하여 진행한다. 따라서 네트워

크의 메시지 트래픽은 클라이언트의 수가 증가하면 기하급수적으로 증가하기 때문에 클라이언트의 수는 자연적으로 제한된다[12]. 그림 2에서 클라이언트의 수가 4개이므로 각 클라이언트는 최소 3개의 메시지 통신 소켓을 가져야 한다. 반면, 개별적인 메시지 전송으로 인해 부하가 각 클라이언트에 분산되며 서버의 부하를 줄일 수 있다는 장점이 있다. 예를 들어 실시간 전략 게임이나 액션 게임의 경우 각 클라이언트는 연결된 상대방의 클라이언트를 인식한 뒤 다른 서버의 도움 없이 서로 메시지를 주고 받는 방식이다.

그러나 이 구조는 각기 다른 플랫폼 클라이언트간의 통신이 요구되므로 메시지 전송방식의 적절한 변환 모듈이 요구된다. 특히 클라이언트/서버 구조와는 달리 하나의 클라이언트가 여러 개의 소켓을 가져야 하지만, 휴대폰은 오직 하나의 소켓만을 가질 수 있으므로 그림 2에 나타나듯 휴대폰은 에이전트(Agent)를 제안하고 도입하여 처리한다. 에이전트에 대해서는 3.2절에서 자세히 다룬다.

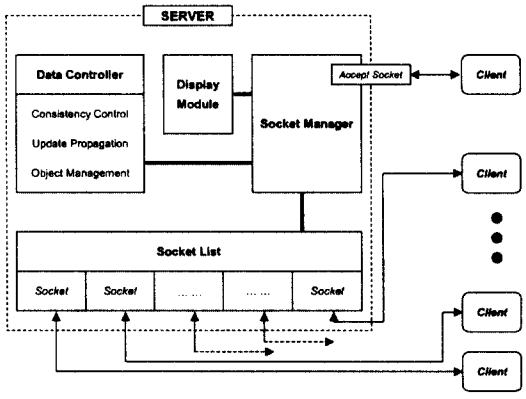


(그림 2) Peer-to-Peer 구조

3.2. 서버 및 클라이언트의 구조도

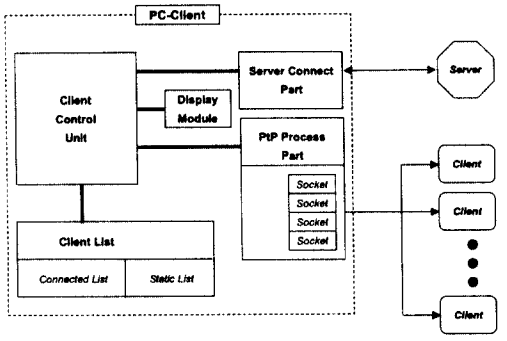
이 절에서는 멀티 플랫폼 온라인 응용을 위해 이 논문에서 설계한 서버, 각 클라이언트, 에이전트에 대해 설명하고자 한다.

그림 3은 온라인 응용을 위한 서버의 구조를 설계한 것이다. 서버는 클라이언트들이 모든 서비스의 요청을 위해 최초로 접속하는 모듈로서 접속 및 종료하는 클라이언트의 관리와 함께 각 클라이언트의 요청을 수행하는 역할을 한다. 서버는 크게 소켓 관리기와 디스플레이 모듈과 데이터 제어기 및 소켓 리스트로 나뉜다. 소켓 관리기는 클라이언트의 서버 연결과 각 클라이언트와의 메시지 송수신을 담당하는 부분으로서 진행결과를 디스플레이 모듈로 보내며 데이터 제어기의 결과를 토대로 메시지를 송수신할 대상과 내용을 결정하게 된다. 데이터 제어기는 다중 사용자 환경의 동시성 제어와 변경전파 및 객체 관리의 역할을 수행한다. 소켓 리스트는 현재 서버에 연결된 클라이언트의 각 소켓 정보를 포함하고 있다.



(그림 3) 멀티 플랫폼 환경의 서버 구조

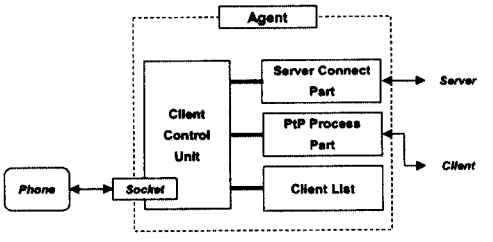
그림 4는 PC와 PDA용 클라이언트의 기본 구조로서 크게 제어부, 디스플레이 모듈, 서버 연결부, Peer-to-Peer 처리부, 클라이언트 리스트로 구성된다. 서버 연결부는 초기 로그인 과정, 동시성 제어(3.3절)등과 같은 클라이언트/서버 구조의 데이터 통신을 위한 모듈이며, 클라이언트 리스트는 변경 정보의 동기화(3.4절)를 위한 정보를 관리하는 모듈로서 연결된 전체 클라이언트와 해당 그룹에 속하는 클라이언트 리스트 정보를 유지한다. 제어부는 각 부분 모듈간의 데이터 제어를 담당한다.



(그림 4) PC 클라이언트 구조

휴대폰 클라이언트는 PC용 클라이언트와 크게 다르진 않지만 다음과 같은 차이점이 있다. PC용 클라이언트는 서버 접속 이후 다른 클라이언트와의 동작시 소켓 생성 및 모든 작업을 서버가 관여하지 않고 클라이언트에서 처리하지만 휴대폰 클라이언트는 서버에 접속하는 작업 이외의 클라이언트간에 이루어지는 소켓 생성 및 모든 작업은 에이전트를 거쳐 처리되므로 대부분의 작업을 에이전트가 수행하게 된다.

그림 5는 이 논문에서 제안한 에이전트의 구조도를 나타낸 것이다. 에이전트는 서버에 존재하는 것으로 하나의 휴대폰이 서버에 접속할 때마다 하나의 에이전트가 생성된다. 에이전트의 구조는 PC 클라이언트와 거의 흡사하지만 휴대폰과의 통신 소켓을 가지고 있는 것이 차이점이다. 휴대폰 클라이언트는 하나의 소켓을 이용하여 에이전트에게 스트림 명령을 내리면 에이전트는 명령을 해석하여 수행한 뒤 결과를 다시 휴대폰에게 넘겨준다. 이렇게 에이전트를 두는 이유는 휴대폰이 오직 하나의 소켓만 생성이 가능하여 Peer-to-Peer 통신을 할 수 없기 때문이며, 에이전트 구조를 제안하여 이 문제를 해결하였고, 저성능이라는 휴대폰의 단점을 극복할 수 있다.



(그림 5) 에이전트 구조

3.3. 동시성 제어

다중 접속자 온라인 응용 프로그램을 응답시간에 따라 분류하면 응용 프로그램 실행 시간의 지연이 있을지라도 반드시 각 클라이언트에서 유지되는 데이터에 대한 일관성이 유지되어야 하는 경우와 실시간 처리를 위해 순간적인 일관성 위배를 허용하는 경우도 있다. 전자는 전자 상거래, 은행 업무 등 정확성 요구하는 응용이며, 후자는 게임, 채팅 등 정확성보다는 실시간 처리를 요하는 응용이다.

이와 같이 다양한 유형의 다중 접속자 온라인 응용에서는 여러 개의 클라이언트가 동작하는 다중 사용자 환경에서 동일한 자원의 동시 접속 시 발생하는 문제를 해결하기 위한 동시성 제어 기법은 두 가지 방법이 존재한다. 우선, 항상 일관성을 유지해야 하는 경우에는 전통적인 잠금 기반 동시성 제어를 수행하는 반면, 실시간 실행을 위해 순간적인 일관성 위배를 허용하는 경우에는 타임 스탬프 기반 동시성 제어 기법[10]을 도입하여 각 클라이언트 연산의 병행성을 최대한 허용하면서 우선 권한이 없는 클라이언트의 연산을 철회하는 방법을 택해야 한다.

전통적으로 제시된 두 가지 동시성 제어 기법은 각기 장단점이 있지만, 멀티 플랫폼 온라인 응용에서는 적용하기 어렵다. 왜냐하면 전통적인 동시성 제어 기법을 적용한 경우 동일한 캐릭터를 두개 이상의 클라이언트에서 동시에 획득하고자 할 때 클라이언트에서의 리소스 요청 시간이 빠른 클라이언트가 실제 서버에서는 잠금 권한을 획득하지 못하는 경우가 종종 발생하기 때문이다. 이는 주로 프로세서의 처리 능력이 떨어지고, 네트워크 전송 속도 및 라우팅(Routing) 경로가 긴 휴대폰에서 주로 발생한다. 다시 말해 클라이언트마다 프로세서의 처리 능력 차이와 네트워크 속도 차이로 인해 전통적인 잠금 기법은 그 정확성을 보장할 수 없다는 것이다. 이 문제는 타임 스탬프 기법에서도 마찬가지로 적용된다. 단, 타임 스탬프 기법은 각 클라이언트 내의 대기 시간은 생략되지만, 공동 작업 결과의 동기화 처리를 해주어야 한다.

이 논문에서는 클라이언트의 처리 능력 차이 및 네트워크 속도에 관계없이 클라이언트에서 잠금을 요청한 시간을 기준으로 잠금을 획득하는 알고리즘을 제안한다. 알고리즘에서 최초 서버는 각 클라이언트가 접속할 때마다 클라이언트 시간과 서버 시간의 차이값을 저장해 놓는다. 그 뒤 하나의 객체 $object_i$에 대한 잠금 요청이 왔을 때 이후 적정 기간(threshold period)동안 $object_i$에 대한 잠금 요청이 있는 클라이언트를 살펴서 각 클라이언트가 잠금을 요청한 시간을 기준으로 보정한 뒤 제일 먼저 요청한 클라이언트에게 잠금 권한을 부여하는 것이다. 그림 6은 동시성 제어 알고리즘을 의사코드로 나타낸 것이다.

```
Algorithm Concurrency_Control_Multi_Platform
{
  Whenever Client(i) is connected
  {
    Calculate Diff(i)
      // see Fig. 7 for Synchronization C/S
  }
  ... ... ...
  When a Client(i) request Lock(objecti)
  {
    Calculate ReqestTime(i) = ClientTime(i)+Diff(i)
    During( threshold period )
    {
      Whenever Client(j) request Lock(objecti)
      Calculate ReqestTime(j) =
                ClientTime + Diff(i)
    }
    Select miminal RequestTime(i)
    Client(i) acquire Lock(objecti)
  }
}
```

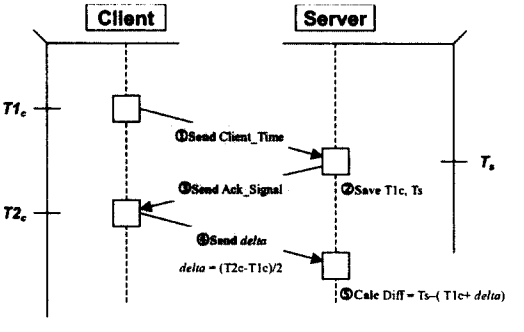
(그림 6) 동시성 제어 알고리즘

이 논문에서는 클라이언트가 접속되었을 때 서버와 클라이언트간의 Clock 동기화를 위해 그림 7과 같은 방법을 이용한다. 여기서 동기화라 함은 각 클라이언트의 시간을 서버의 시간과 같게 맞추는 것이 아니라(온라인 클라이언트의 특성상 각 지역마다 고유의 클라이언트 시간이 존재할 수도 있고, 한 응용을 위해 일일이 클라이언트의 시간을 조정할 수 없음), 서버에서 각 클라이언트의 시간과의 차이값을 보관한 뒤 추후 리소스 요청시 클라이언트의 시간을 서버에 맞게 보정하는 것을 의미한다.

그림 7에서 우선, 클라이언트의 지역 시간(Local Time)을 서버에 먼저 전송하면(①), 서버에서는 수신한 클라이언트 시각과 수신할 때의 서버의 시간을 저장한다(②). 이 때, 단순히 클라이언트의 시간과 서버의 시간 차이값만을 동기화한다면 클라이언트의 지역 시간이 네트워크를 통해 보내지는 시간을 감안하지 못하는 경우가 발생한다.

그러므로 이 논문에서는 서버가 다시 클라이언트에게 확인(Acknowledgement) 신호를 보내면(③), 클라이언트가 신호를 받은 시각($T2_c$)에서 최초 클라이언트 시간을 보낸 시각($T1_c$)을 뺀 값의 절반을 네트워크 전송시간(*delta*)으로 계산한다. 왜냐하면 네트워크의 연결지향(connection oriented) 상태에서는 데이터의 송수신 시간이 거의 같기 때문이다.

그 뒤 클라이언트는 계산된 네트워크 전송시간을 서버에 보내며(④), 서버는 차이값(Diff)을 구한다(⑤). 차이값은 클라이언트의 지역 시간과 서버 시간과의 차이값으로서 동기화할 때의 네트워크 전송시간까지 고려한 것이다. 이 차이값은 추후 각 클라이언트의 리소스 요청시간을 서버 시간을 기준으로 보정하기 위한 값으로 사용된다.



(그림 7) 클라이언트와 서버의 동기화

3.4. 변경 정보의 동기화

2장에서 언급한 공동 작업 결과의 동기화 문제를 다룬 기존의 연구는 이 논문의 온라인 실시간 환경에 적합하지 않다. 왜냐하면 인터넷 기반 학습, 게임 등과 같은 온라인 실시간 환경은 여러 클라이언트의 변경된 정보를 즉시 반영할 수 있는 변경 전파 기법이 필요하기 때문이다. 다중 접속자 온라인 응용 프로그램에서 각 클라이언트에 의해 변경된 데이터를 여러 PC 및 다른 모바일 클라이언트에게 제각기 다른 기법으로 반영하는 방법이 새롭게 설계되어야 한다.

이 논문에서는 온라인 변경 전파를 위해 3가지 부류로 구분하여 변경 정보의 즉시 전파를 수행한다. 첫째, 전체 클라이언트에게 브로드캐스팅, 둘째, 정적 관련성을 갖는 클라이언트에게 변경 전파, 셋째, 동적 관련성을 갖는 클라이언트에게 변경 전파 방법으로 구분한다. 각각의 전파 방법을 위해 이 논문에서는 다음과 같은 기법을 이용한다.

우선, 시나리오에 따른 종속성 도표(Dependency Table)를 작성하고, 이 종속성 도표를 근거로 하여 각 클라이언트에 의해 변경된 데이터를 여러 PC 및 모바일 클라이언트에게 변경 정보를 전파한다.

그림 8에 나타난 바와 같이 서버에서 관리하는 종속성 도표는 전체 3부분으로 나누어진다. '연결된 클라이언트' 섹션은 현재의 온라인 응용의 해당 개설 창구에 접속한 모든 클라이언트를 의미하는 것으로서 서버가 모든 클라이언트에게 브로드캐스팅하고자 할 때 사용된다. '정적 관련성' 섹션은 현재 연결된 클라이언트 중 연결시 결정되는 정적 그룹정보를 의미하는 것으로서 해당 그룹별로 선택적인 메시지를 전파하고자 할 때 사용된다. 그룹의 조합은 필요에 따라 여러 번 반복가능하다. 예를 들어 온라인 게임 응용에서 A, B 두 그룹으로 나누어 진행될 때 각 그룹마다 변경 전파 메시지가 달라야 할 경우에 사용된다. '동적 관련성'은 온라인 응용 도중에 순간 순간 발생하며 변경 전파 과정을 거친 후에 사라지는 정보이다. 예를 들어 동시성 제어에서 타임스탬프 기법을 사용했을 경우 동시에 같은 아이템을 요구한 클라이언트들은 모두 동적 관련성 정보로 생성이 되며, 아이템을 획득하지 못한 여러 개의 클라이언트는 서버로부터 메시지를 받아 복귀(Rollback)하게 된다. 그림 8에서 동적 관련성은 응용 프로그램이 진행되는 동안 관련성 정보가 생성 후 삭제 과정이 반복되며, 'Waiting'은 아이템에 대한 권한이 확정되지 않은 상태이며, 'Acquire'는 아이템을 획득한 상태이며, 'RollBack'은 아이템을 획득하지 못해 복귀해야 함을 의미한다. 그리고 각 클라이언트의 정보는 다음과 같다.

Info. of client C_i =

[Login_ID, IP_Address, Socket_Port_No]

지금까지 설명한 종속성 도표는 서버에서 클라이언트/서버 구조 형태로 변경 정보를 전파하기 위한 메타 데이터임에 반해 각 클라이언트 또한 Peer-To-Peer 구조 형태로 직접 다른 클라이언트와의 통신을 위해 '연결된 클라이언트'와 '정적 관련성' 정보를 유지하고 있다.

Dependency Dictionary

Connected Clients	Info. of client C1		
	Info. of client C2		
	… … …		
	Info. of client Cn		
Static Dependency	**Group A**	Info. of client C1	
		Info. of client C9	
		… … …	
	Group B	Info. of client C3	
		Info. of client C6	
		… … …	
	…	…	
Dynamic Dependency	**Item A**	Info. of client C5	Waiting
		Info. of client C3	Waiting
	Item B	Info. of client C2	Acquire
		Info. of client C8	RollBack
		Info. of client C7	RollBack
		↓ ↑	

(그림 8) 종속성 도표

3.5. 클라이언트의 객체 관리 기술

멀티 플랫폼 기반 온라인 응용에서 고정 단말기인 PC에 비해 모바일/무선 클라이언트인 PDA

나 휴대폰은 낮은 처리 능력 및 통신 대역폭을 가지고 있으므로 이러한 리소스의 차이를 극복하기 위한 방안이 필요하다. 이 논문에서는 이를 위해 클라이언트의 차별적인 객체 처리 기술을 도입한다.

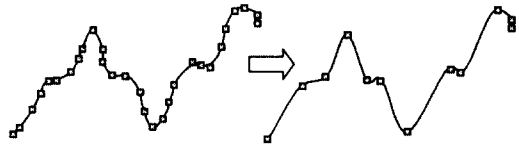
우선 데이터의 전송량을 줄이기 위해 응용 프로그램에 등장하는 각 객체의 종류를 분석하고 객체마다 적절한 우선 순위를 부여한다[12]. 정해진 우선 순위에 따라 TCP 패킷(높은 우선순위)을 사용할 것인지, UDP 패킷(낮은 우선순위)을 사용할 것인지, 보내지 않을 것인지를 결정하여 우선 순위 서비스를 제공한다. 또한 서버에서 객체에 대한 우선 순위에 따라 패킷의 전송 비율을 조절한다. 예를 들어 멀티 플랫폼 기반 온라인 게임에서 표 1과 같이 게임 진행에 중요한 캐릭터인 적 캐릭터와 탄환 등은 높은 우선 순위를 가지는 반면 장식적인 캐릭터와 배경 장식 등은 낮은 우선 순위를 가진다. 이 논문에서는 이와 같은 객체 우선 순위는 적은 메모리 용량과 낮은 대역폭을 가진 휴대폰에서도 적용될 수 있도록 패킷 전송 비율을 각 클라이언트에 맞게 탄력적으로 조정할 수 있도록 하였다.

<표 1> 객체 우선 순위

객체	우선순위
적	5
탄환	4
보조 캐릭터	3
장식 캐릭터	2
배경 장식	1

그리고 적은 메모리 용량을 가진 모바일/무선 클라이언트를 위해 데이터의 단순화(Simplication) 및 일반화(Generalization)[9, 11] 알고리즘을 이용해 객체의 표현량을 줄여야 하고, 낮은 처리 능력을 지닌 휴대 장치를 위한 클라이언트 모듈에서는 중요도에 따라 불필요한 모듈은 제거를 하고, 알고리즘 또한 보다 단순화함이 요구된다.

일반화는 그래픽이나 도형 자료를 기본 바탕으로 하는 온라인 응용에서는 객체의 형태를 크게 변화시키지 않는 범위에서 적절히 좌표수를 줄임으로써 저장 공간을 효율적으로 사용할 수 있는 방법이다. 그러나 무엇보다도 좌표의 삭감으로 인해 객체의 원형에 변화가 발생하지 않도록 주의하여야 한다. 좌표 삭감의 정도는 사용자가 필요로 하는 최종 결과물의 형태에 의해 결정된다. 삭감의 정도가 높아질수록 저장되는 좌표수가 적으며 객체의 형태를 단순화시키는 결과를 가져온다. 따라서 정확한 데이터 표현 및 분석을 요구하는 클라이언트에서는 데이터 삭감을 적게 해야 하며, 낮은 저장 용량을 가진 클라이언트에서는 데이터 삭감의 정도를 높이는 등, 사용자의 필요에 따라 표현의 정확도를 결정할 수 있다. 좌표 삭감은 또 세부적으로 두 가지 방법을 고려할 수 있다. 첫째, 일정 간격(Tolerance)을 부여하여 간격내의 점을 제거하는 방법이다. 둘째, 다중선의 각도를 조사하여 둔각을 이루는 중앙점을 제거하는 것이다. 그림 9는 두 번째 방법을 이용하여 좌표를 삭감한 것으로서 모양은 거의 유지를 하면서 좌표점의 개수는 거의 1/3으로 줄어들었다. 표현의 정확도를 조절하기 위해 첫 번째 방법은 간격의 값을 조절하고, 두 번째 방법은 둔각의 각도값을 조절하면 된다.



(그림 9) 객체 일반화

또한 PC 클라이언트의 다각형 정보를 휴대 장치에서는 점으로 표현하는 단순화 등을 고안하여 저장 공간의 한계를 극복할 수 있다.

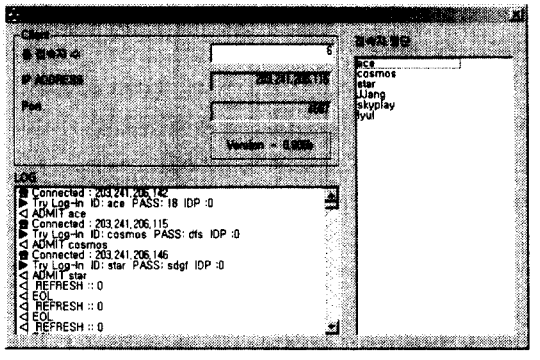
4. 프로그램의 구현 예

이 논문에서 제안한 시험 구조와 데이터 제어 기법을 도입하여 멀티 플랫폼 기반 온라인 채팅 프로그램을 구현해 보았다. 구현 환경은 다음과 같다.

- 서버 : Pentium 4, C++ Program
- PC 클라이언트 : Pentium 4, C++ Program
- PDA 클라이언트 : Compaq iPaq 3850, Embeded Visual Tool

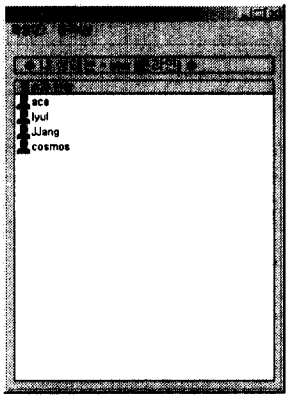
- Phone 클라이언트 : SK-VM Phone Emulator Ver 1.1, Java(J2ME)

그림 10은 Sever의 로그 화면을 나타낸 것이다. 로그 화면에서는 현재 접속자 수와 접속자 ID 및 기타 온라인 진행 정보를 나타내고 있다.



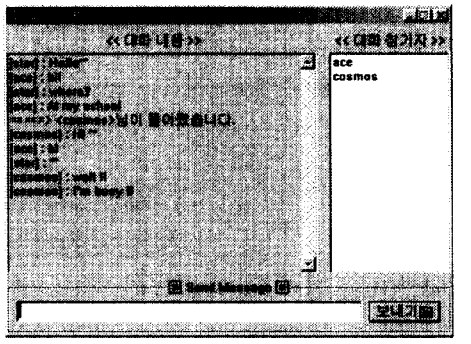
(그림 10) 서버의 LOG 화면

그림 11은 PC용 클라이언트의 로그인 후 서버로부터 현재 접속되어 있는 모든 클라이언트의 정보를 넘겨받은 결과(Refresh)를 나타낸 것이다. 각 클라이언트가 로그인하거나 로그아웃 할 때마다 클라이언트/서버 구조로 해당 정보를 브로드캐스팅하게 된다. 이 때 서버의 데이터 제어기에서 변경전파모듈과 관련성도표를 참조하게 된다.

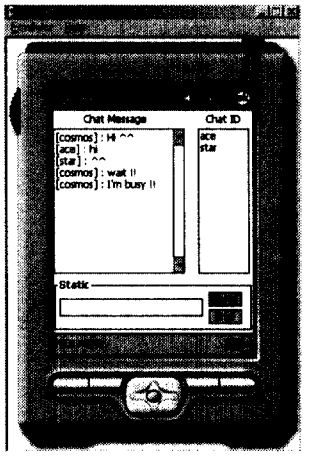


(그림 11) 클라이언트 Refresh 화면

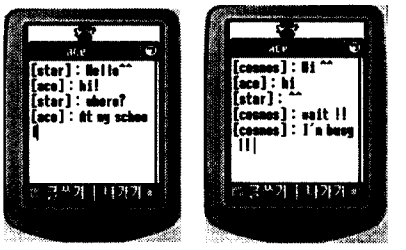
그림 12, 13, 14는 Peer-to-Peer 구조로 각 클라이언트간의 데이터를 송수신하는 과정을 보여주는 것이다. 그림 12는 PC(ID : star) 클라이언트이며, 그림 13은 PDA(ID : cosmos) 클라이언트이며, 그림 14는 휴대폰(ID : ace) 클라이언트이다. 최초 star와 ace가 대화를 하고 있는 상황에서 cosmos가 추가되어 3개의 클라이언트가 N:M 통신을 하고 있음을 보여준다.



(그림 12) PC 클라이언트의 대화장면



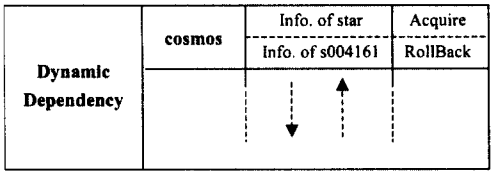
(그림 13) PDA 클라이언트의 대화장면



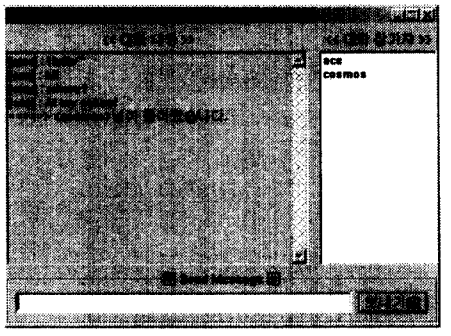
(그림 14) 휴대폰 클라이언트의 대화장면

다음은 이 논문에서 제안한 동시성 제어와 변경 전파 모듈의 수행이 요구되는 상황으로서 두 개의 PC 클라이언트(star, s004161)가 동시에 하

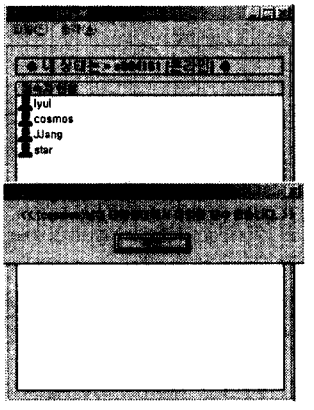
나의 PDA 클라이언트(cosmos)에게 대화 요청을 하였을 경우 서버의 종속성 도표에는 그림 15와 같은 동적 관련성이 생성이 되며, 3.3절의 동시성 제어 알고리즘에 의해 star가 cosmos에 대한 대화 권한을 획득하여 대화 상대로 추가되었고(그림 16), s004161의 수행화면에서는 대화 불가능이란 메시지가 서버에서 즉시 변경 전파됨을 알 수 있다(그림 17).



(그림 15) 동시 요청으로 생성된 동적 관련성



(그림 16) 대화 권한 획득 후 변경 전파 결과



(그림 17) 대화 불가능 정보의 즉시 전파

5. 결 론

이 논문은 최근 그 사용자가 꾸준히 확대되고 있는 PDA, 휴대폰 등의 모바일/무선 단말기와 PC와 같은 기본 고정 단말기 등이 서로 유무선으로 연계되어 응용 프로그램을 수행하는 멀티 플랫폼 온라인 응용의 원활한 서비스를 위해 클라이언트/서버 인프라 구축 기술을 개발하고자 하였다. 이러한 온라인 환경은 각 클라이언트 기기의 사용 가능한 리소스(메모리 용량, 프로세서 처리 능력) 차이가 심하고 통신 대역폭의 차이가 심하여 기존의 데이터 제어 기법을 그대로 적용할 수 없다. 이러한 문제점을 해결하기 위해 이 논문에서는 멀티 플랫폼 기반 다중 사용자의 동시 접속 온라인 응용 프로그램에서 효과적인 데이터 제어를 위해 새로운 동시성 제어 기법과 변경전파 기법 및 객체 관리 기법을 제안하였다.

구체적으로 다양한 컨텐츠를 제공하는 온라인 응용의 동시성 제어를 위해 응답 시간의 지연을 허용하여 항상 일관성 유지를 보장해야 하는 경우와 실시간 처리를 위해 순간적인 일관성 위배를 허용하는 경우로 나누어 설계하였으며, 특히 다양한 클라이언트의 처리 능력 차이 및 네트워크 속도에 관계없이 잠금을 요청한 시간을 기준으로 잠금을 획득하는 알고리즘을 제안하였다.

그리고 변경 정보의 동기화를 위해 시나리오에 따른 종속성 도표를 근거로 변경 정보를 전파하는 방법을 제안하였다. 종속성 도표에서 '연결된 클라이언트'는 현재 개설된 창구에 연결된 모든 클라이언트에 변경 정보를 전파하기 위한 정보이며, '정적 관련성'은 그룹별로 선택적인 변경 정보 전파를 위한 정보이며, '동적 관련성'은 동시성 제어 후의 복귀 정보 전달 등을 위해 유용하게 사용된다.

또한 고정 단말기인 PC에 비해 모바일/무선 클라이언트인 PDA나 휴대폰은 낮은 처리 능력 및 통신 대역폭을 가지고 있으므로 이러한 리소스의 차이를 극복하기 위한 방안으로서 객체 우선 순위 부여, 데이터의 단순화와 일반화, 알고리즘의 단순화를 제안하였다.

또한 이 논문에서 서버, PC 클라이언트, PDA 클라이언트, 휴대폰 클라이언트 모듈을 개발하여 적용해본 채팅 프로그램을 통해 제안한 기법이 적절히 동작함을 알 수 있었다. 이 논문에서 제

안한 기법들은 온라인 게임, 학습 등 다양한 분야에서도 쉽게 적용할 수 있으리라 판단된다. 앞으로는 보다 엄격한 제약 조건을 요구하는 멀티 플랫폼 기반 온라인 게임을 개발하여 본 논문에서 제안한 기법들을 적용하고자 한다.

참 고 문 헌

[1] P.A. Bernstein, N. Goodman(1984), "An Algorithm for Concurrency Control and Recovery in Replicated Distributed Databases", ACM Transactions on Database Systems, vol. 9, no. 4, pp. 596-615

[2] D.R. Brown, M.R. Cutkosky, J.M. Tenenbaum(1990), "Next-Cut : A Computational Framework for Concurrent Engineering", Int. Symposium on Concurrent Engineering

[3] G. Coulouris, J. Dollimore, T. Kindberg(1994), "Distributed Systems : Concepts and Design", Addison-Wesley Publishing

[4] M.R. Cutkosky, R.S. engelmore, R.E. Fikes, M.R. Genesereth, T.R. Gruber, W.S. Mark, J.M. Tenenbaum, J.C. Weber(1993), "PACT : An Experiment in Integrating Concurrent Engineering Systems", IEEE Computer, pp.28-37

[5] P. Dewan, J. Riedl(1993), "Toward Computer Supported Concurrent Software Engineering", IEEE Computer, pp. 17-27

[6] K. Dittrich, R. Lorie(1988), "Version Support for Engineering Database System s", IEEE. Transactions on Software Engineering, Vol. 14, No. 4, pp. 429-427

[7] J. Jing, A. Elmagarmid(1999), "Client-Server Computing in Mobile Environments", ACM Computing Surveys, Vol 31, No 2

[8] A. Kahol, S.Khurana, S. Gupta, P.Srimani(2000), "An Efficient Cache Maintenance Scheme for Mobile Environment", International Conference on Distributed Computing Systems

[9] R. Laurini, D. Thompson(1992), "Fundamentals of Spatial Information Systems", Academic Press

[10] R. Ramakrishnan, J. Gehrke(2000), "Database Management Systems", McGraw-Hill, pp. 523-570

[11] 김미란, 최진오(2001), "Generalization과 filtering을 이용한 무선지도 데이터베이스의 동적 생성 기법", 정보처리학회 논문지, 제8권 제4호

[12] 이만재(2002), "온라인 게임 엔진 기술 동향", 한국정보과학회 학회지, 20권, 1호, pp.12-18

김 진 덕

1993 부산대학교 컴퓨터공학과 (공학사)
1995 부산대학교 컴퓨터공학과 (공학석사)
1995~2000 부산대학교 컴퓨터공학과(공학박사)
1998~2001 부산정보대 정보통신계열 전임강사
2001~현재 동의대학교 컴퓨터공학과 전임강사
관심분야: 데이터베이스, GIS, 모바일 컴퓨팅
E-Mail: jdk@dongeui.ac.kr

진 교 홍

1991 부산대학교 컴퓨터공학과 (공학사)
1993 부산대학교 컴퓨터공학과 (공학석사)
1993~1997 부산대학교 컴퓨터공학과(공학박사)
1997~2000 국방과학연구소 선임연구원
2000~현재 동의대학교 멀티미디어공학과 조교수
관심분야: 컴퓨터 네트워크, 분산 처리 시스템
E-Mail: khjin@dongeui.ac.kr