韓國IT서비스學會誌
第7卷 第2號
2008年 6月

모바일 메인메모리 데이터베이스 시스템을 위한 효율적인 복구 기법

조 성 제*

An Efficient Recovery Method for Mobile Main Memory Database System

Sung-Je Cho*

■ Abstract ■

The rapid growth of mobile communication technology has provided the expansion of mobile internet services, particularly mobile realtime transaction takes much weight among mobile fields. There is an increasing demand for various mobile applications to process transactions in a mobile computing fields. Thus, During transmission in wireless networks a base station failure inevitably causes data loss of the base station buffer. It is required to compensate the loss for communication. The existing methods for a base station failure are not adequate because they all suffer from too much overhead and resolve only the link failure. In this paper, we study an efficient recovry systems for a mobile DBMS. We propose SLL (Segment Log List) that enables the mobile host to compensate data loss efficiently in the case of base station failure. In SLL, a base station deliveries an output information of data cells to a mobile host. when a base station fails, the mobile host can retransmit just next data cells. We also prove the efficiency of new method.

Keyword : Mobile Computing, Log, Segment, Recovery, Database, Mobile Main Memory DB

논문투고일 : 2008년 04월 21일 논문수정완료일 : 2008년 06월 05일 논문게재확정일 : 2008년 06월 09일

* 동방대학원대학교 문화정보학과

1. 서 론

현재 와이브로나 HSDPA 등의 통신망이 발달되고, 컴퓨터 사용이 대중화됨에 따라 노트북, PDA 그리고 UMPC 등이 고성능 모바일 클라이언트 장비로 활용되고 있다. 따라서 초기의 PDA, 핸드폰 등의 모바일 장비들은 한정된 성능으로 인해 읽기 전용 트랜잭션 처리가 대부분이었다. 본 연구에서는 무선통신의 발달과 고성능 모바일 장비를 이용하여 읽기 트랜잭션과 쓰기 트랜잭션을 처리할 수 있는 회복시스템을 연구하고자 한다.

일반적으로 DBMS의 회복 시스템은 회복을 위하여 트랜잭션 처리에 대한 로그정보를 디스크상의 로그파일에 기록하고, 주기적으로 체크포인트를 수행하여 메인 메모리상에 있는 수정된 데이터를 DB에 반영하게 된다. 그러나 회복을 위한 로그 기록과 DB 반영을 위한 디스크 입출력은 시스템 성능을 저하시키는 중요한 요인이 된다.

따라서 시스템 성능을 최대화하기 위해서는 디스크 입출력을 줄일 수 있는 효율적인 회복시스템이 필요하다. 본 연구에서는 메인메모리 DBMS 환경에서의 회복기법을 연구하였다.

최근 들어 모바일 클라이언트들과 방송서버들을 네트워크로 연결시킨 환경을 이용하는 경향이 증가하고 있다. 이러한 환경의 DBMS를 모바일 클라이언트/방송서버 DBMS라고 한다. 모바일 클라이언트는 모바일 컴퓨터나 여러 가지 휴대용 통신기기가 될 수 있고, 모바일 클라이언트 사용자를 이동사용자라 정의한다.

모바일 클라이언트가 다른 모바일 클라이언트와 통신하기 위해서는 방송서버를 경유해야 한다. 모바일 컴퓨팅 환경에서 이동 사용자는 데이터베이스의 검색이나 원하는 작업을 수행하기 위한 방법으로 트랜잭션을 제출한다.

모바일 클라이언트에서 수행되는 트랜잭션을 모바일 트랜잭션이라 하는데[1, 11, 12], 모바일 사이트에서는 주로 판독 전용 트랜잭션이 수행되고, 반면에 방송서버에서는 쓰기연산으로 구성되는 갱신 트랜잭션이 수행된다.

지금까지는 유선통신망 환경에서 중앙집중식 DBMS나 디스크기반 시스템에서의 회복기법에 대해서는 많은 연구가 이루어져 왔다. 그러나 모바일 클라이언트/방송서버 환경에서의 회복기법에 대해서는 연구가 거의 없다. 본 본문에서는 모바일 클라이언트/방송서버 DBMS에서의 회복기법에 대해서 기술한다. 방송서버는 메인 메모리 DBMS 환경이다. 그 이유는 실시간 서비스를 효율적으로 제공하기 위함이다. 기존 디스크 기반 DBMS는 입출력이 전체시스템 성능을 저하하는 중요한 요인이었다.

본 연구에서 사용하는 트랜잭션은 방송서버로부터 전파를 통해 해당 세그먼트를 전송받아 모바일 클라이언트가 직접 트랜잭션을 수행하는 환경으로 가정한다. 기존의 은행업무 및 전자상거래시 사용하는 트랜잭션은 서버가 트랜잭션을 수행 한다.

예를 들면, 모바일 클라이언트/방송서버 DBMS는 클라이언트가 서버에 질의(query)로 요청하는 방법과 클라이언트가 특정 데이터를 서버에 요청하는 방법으로 구분될 수 있다. 서버에 데이터를 요청하는 방법에는 그 성격에 따라 다시 페이지 서버와 객체서버로 구분할 수 있다[13].

클라이언트가 서버에 요청 질의로 신청하는 방법과 특정 데이터를 신청하는 방법 간에는 상대적으로 장점이 있다[13]. 질의로 신청하는 방법은 한 사이트에서 유지되는 중앙 집중식 데이터베이스시스템과 트랜잭션 처리구조가 비슷하다는 장점이 있다.

그래서 현재 사용되고 있는 한 지점에서 유지되는 중앙집중식 데이터베이스 시스템을 클라이언트/서버 환경의 데이터베이스 시스템으로 전환이 용이하다. 이 기법은 클라이언트/서버간의 통신에 있어서, 서버는 클라이언트가 보낸 질의에 대해 만족되는 결과만 클라이언트로 전송하면 되는 통신 오버헤드가 적다는 장점이 있다.

한편, 클라이언트가 서버에게 특정 데이터를 신청하는 데이터베이스시스템에서의 장점으로는 모

바일 클라이언트들의 기억장치 이용율과 처리율을 높임으로서, 서버의 병목현상(bottleneck)을 최소화 할 수 있는 장점이 있다.

이러한 방법은 각각의 모바일 클라이언트의 기억장치와 처리능력을 활용한 방법이며, 그 성능을 고려할 때 효율적이라 할 수 있다. 또 최근에 와이브로나 HSDPA 등의 통신망 발달로 대역폭이 점점 더 향상되는 추세이므로 모바일 클라이언트와 서버간의 통신비용은 점점 더 중요치 않은 요인이 되고 있다.

본 논문에서는 기존의 클라이언트가 특정데이터를 요청하는 환경을 모바일 메인메모리 환경에서의 효율적인 회복기법을 제안하고자 한다. 기존기법을 모바일 환경으로 전환 시 몇 가지 문제점이 발생한다. 첫째, 통신단절로 인한 캐쉬 일관성 문제점을 해결해야한다.

본 연구에서는 기존의 낙관적 동시성제어 기법을 모바일 환경에 적합한 기법을 제한하고자 한다.

둘째, 기존기법은 장애에 대한 회복을 위해 로그일련번호를 서버가 부여한다. 이 때 동기화를 위해 클라이언트간의 통신오버헤드가 발생한다. 본 연구에서는 클라이언트별 로그번호를 부여함으로써 통신오버헤드를 제거하는 기법을 제안하고자 한다.

셋째, 기존 회복 동작은 로그분석을 통해 재수행 동작과 철회수행 동작을 실시한다. 이 때 로그분석에 따른 많은 회복시간이 소요된다. 본 연구에서는 로그분석을 재수행 동작만 실시할 수 있는 모델을 제안하고자 한다.

그리고 모바일 클라이언트가 특정 데이터를 요청하는 환경은 고성능 모바일 클라이언트들의 기억장치에서 객체에 직접 접근이 가능하므로, OODBMS에 잘 적용된다는 장점이 있다.

클라이언트가 질의로 요청하는 구조에서의 회복기법은 중앙집중식인 전통적 회복기법과 비교해볼 때 새로운 문제점은 거의 없다. 그러나 클라이언트가 서버로 특정 데이터를 요청하는 구조에서는 몇 가지 문제점이 발생한다.

이러한 문제점은 다음과 같은 구조적 특징에 기인한다.

첫째, 데이터베이스에 대한 갱신은 클라이언트의 기억장치 내에서 일어나고, 서버는 갱신에 대한 로그와 갱신 발생 전 상태의 데이터베이스를 보유한다.

이 때 데이터의 일관성 문제를 해결하기 위해서는 기존 기법들은 로그일련번호의 동기화 문제를 해결하기위해 복잡한 처리과정이 필요하다[4, 5, 7-10]. 서버는 데이터 갱신 후 다른 클라이언트에게 로그일련번호를 전파하므로 많은 통신오버헤드가 발생하였다.

둘째, 각 클라이언트가 버퍼를 지닌다.

셋째, 클라이언트는 서버와 다른 성능과 신뢰성을 갖는다.

본 연구에서는 이러한 모바일 클라이언트/방송서버 환경에서의 효율적인 회복기법에 대하여 논한다. 또 본 연구의 회복기법은 모바일 클라이언트/방송서버 회복의 대표적인 기법인 Extending Aries-CS기법에 기본을 두고 있다.

본 연구의 기본 개념은 다음과 같다.

첫째, 기존 기법들의 데이터 갱신은 클라이언트의 버퍼에서 이루어지고, 로그와 회복관리자는 서버 내에 존재한다. 서버가 데이터 일관성 및 로그일련번호의 동기화 문제를 해결하기 위해 클라이언트들 간의 병목현상이 많이 발생한다.

본 연구는 클라이언트별 로그일련번호를 부여함으로써 동기화에 따른 병목현상을 감소하고자 한다.

둘째, 기존기법들 서버가 2PL 프로토콜을 사용하여 트랜잭션을 수행한다[6, 8, 9]. 이 때 트랜잭션 철회동작은 서버가 수행한다. 본 연구의 트랜잭션 환경은 다음과 같다.

① 낙관적 동시성제어기법을 사용함을 가정한다.
② 각 모바일 클라이언트는 동시에 하나의 트랜잭션을 수행할 수 있는 환경이다. 클라이언트에서 여러 트랜잭션을 수행시 발생할 수 있는 오버헤드 증가를 해소하기 위한 방안이다.
③ 기존기법들과 같이 클라이언트에서 서버로

부터 해당 페이지를 전송받아 트랜잭션을 수행하는 환경이다. 트랜잭션 완료 후 로그를 방송서버에 전송하면, 방송서버는 세그먼트 로그 리스트(Segment Log List)를 통해 충돌여부를 점검한다. 비충돌이면 완료하고, 충돌이면 클라이언트에게 재 수행을 지시한다.

④ 모바일 클라이언트가 트랜잭션을 재수행하고자 할 때 낙관적 동시성 제어 프로토콜에 의해 해당 데이터 페이지를 서버로부터 재전송 받아 수행해야한다.

⑤ 모바일 클라이언트는 컴퓨터 성능 향상으로 메모리 및 배터리 성능이 우수함을 가정한다.

본 연구는 모바일 클라이언트가 해당 트랜잭션을 철회동작하고, dirty된 페이지를 undo로그를 이용하여 redo하여 트랜잭션을 재수행하는 환경이다. 그 이유는 서버의 병목현상을 감소하기 위해서다.

논문의 구성은 먼저 제 1장에서 연구의 목적 제시하고, 제 2장에서는 관련연구와 문제점 제시 하였다. 제 3장에서는 모바일 컴퓨팅 환경에서의 완료트랜잭션 로그를 새로운 회복 기법을 제안하였고, 제 4장에서는 회복기법에 대한 내용을 언급한다. 제 5장에서는 기존방법과 비교분석하고, 제 6장에서는 성능평가를 알아본다. 마지막으로 결론 및 향후 연구에 대해 언급한다.

2. 관련 연구

2.1 모바일 캐슁 기법

Man Hon Wong은 모바일 컴퓨팅 환경에서 판독전용 트랜잭션을 지원하기 위한 캐쉬 정책을 제시하였다[10]. 이 기법의 단점은 이동성으로 인하여 캐쉬 내의 일관성을 유지하기가 어렵다. 캐쉬 내의 일관성을 유지하기 위한 방법으로 무효화 방송전파 메시지를 송수신하는 방법이다. 무효화 전파는 방송 서버에서 갱신된 항목을 모바일 사이트에 알리는 방송메시지이다. 캐쉬를 사용할 때 모바일 사이트가 통신 단절로 인하여 이 메시지 수신이 불가능한 경우에는 캐쉬 내 데이터 일관성은 보장하기 어렵다. 이러한 경우에 캐쉬 내 일관성을 유지하기 위하여 다중 버전을 제공하고 있고, 버전관리를 위해 방송 서버와 호스트사이에 통신 오버헤드가 발생한다. 이 기법은 쓰기연산의 경우에는 전혀 사용할 수 없는 문제점이 있다. Kam-Yiu Lam에서는 캐쉬를 사용하는 환경에서 주기적인 방송으로 캐쉬 내 일관성을 유지하는 두 가지 방법을 제안하였다[12]. 첫 번째는 모바일 클라이언트의 상태나 캐쉬 내의 내용을 파악하고 있는 방송 서버와 모바일 클라이언트의 어떤 정보도 소유하고 있지 않은 방송 서버로 분류하였다. 두 번째는 모바일 클라이언트에 방송 무효화 전파 방법을 일정시간 동안 갱신정보를 모아서 정기적으로 전파하는 동기식과 갱신이 발생하는 즉시 전파하는 비동기식으로 분류하였다. 이 방법은 직렬성에 위배되는 경우가 발생한다.

방송 서버에서만 갱신작업을 하므로 모바일 클라이언트에서 쓰기 연산하는 경우에는 적용이 어렵고, 방송전파에 따른 오버헤드가 존재한다. 이 기법은 쓰기 연산에 따른 회복기법에는 전혀 고려되고 있지 않다.

2.2 Extending ARIES-CS

Extending ARIES-CS은 ARIES 고장 기법을 클라이언트-서버 환경으로 확장하였다[11]. EXODUS 기법과는 다르게 클라이언트에서만 LRC를 이용하여 WAL 규약을 구현하였다[2, 5]. 로그 레코드와 해당 데이터 페이지의 서버 도착 시각의 차이로 인한 비일관성 극복을 위해 EXODUS기법과 같이 조건부 철회를 사용하였다. Extending ARIES-CS은 로그 레코드가 서버에 도착했을 때 해당 페이지가 dirty된 것으로 간주한다. 이 기법은 여러 클라이언트에서 발생하는 로그 일련번호 부여에 많은 메시지교환에 의해서 생성한다. 그 결과 동기화에 따른 오버헤드가 발생 한다.

2.3 Callback Locking기법

Callback Locking기법은 모바일 클라이언트가 데이터(lock mode)를 함께 캐쉬하고 있어, 트랜잭션이 종료 되더라도 로그를 소유하고 있어 그 페이지에 대한 유효성을 보장하는 기법이다[3, 11]. 이 기법은 모바일 클라이언트가 데이터 페이지에 접근시 매번 서버에게 데이터 페이지 유효성을 검사할 필요가 없다. 그러나 읽기 전용로그(shared lock)를 가진 데이터 페이지에 트랜잭션을 변경하고자 할 때 전용로그(exclusive lock)를 요청해야 한다. 이 경우 서버는 해당 데이터가 캐쉬 되어 있는 다른 모든 모바일 클라이언트 사이트에 메시지를 보내어 캐쉬된 로그를 회수(callback)해야 하는 문제점이 있다.Daniel Barbara에서는 읽기 로그 회수기법(callback read locking)을 사용하였고[3], Jin Jing에서는 읽기 및 변경용 로그를 캐쉬하는 쓰기 로그 회수 기법(callback write locking)을 사용하였다[11]. 이 기법은 회복시 로그분석 동작에 많은 오버헤드가 발생한다.

2.4 No-waiting Locking 기법

이 기법은 모바일 클라이언트 사이트에 캐쉬된 데이터의 정확성 점검을 서버의 응답을 기다리지 않고 바로 처리에 들어가는 기법이다[8]. 이 경우은 여러 사이트에 캐쉬되어 있을 때 트랜잭션 철회율이 많아진다.

3. 제안하는 모바일 시스템 모델

본 장에서는 모바일 DBMS를 위한 전체시스템의 구조와 프로세스에 대해서 언급한 후 본 연구에서 제안하는 회복개념을 설명한다.

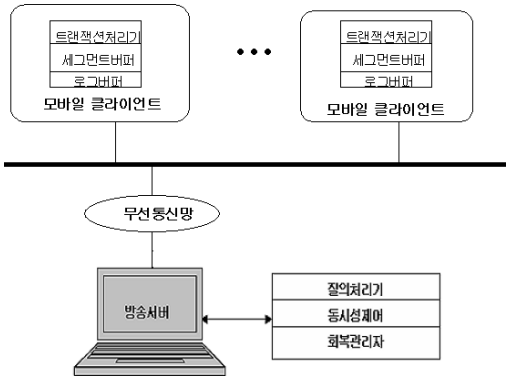
[그림 1]에서와 같이 방송서버, 메인 메모리 데이터베이스, 모바일 클라이언트, 그리고 모바일 통신망으로 구성하고 있다.

전체 시스템의 구조를 설명하면 다음과 같다.

① 방송서버는 모바일 클라이언트에서 공급받은 로그 데이터를 데이터베이스에 반영하기 위해 트랜잭션간의 직렬성 검증을 담당한다.

그리고 방송서버는 일정한 주기로 모바일 클라이언트에게 방송되며, 완료 정보를 디렉터리 형태로 구성하여 신속한 전역검증을 담당한다. 디렉터리 구성을 세그먼트 로그 리스트(Segment log list : SLL)라고 명명한다.

② 모바일 클라이언트는 하나의 트랜잭션을 처리하기 위하여 세그먼트 버퍼와 로그버퍼 지니고 있다. 그 이유를 기술하면, 방송 서버와 모바일 클라이언트의 병목현상 감소를 위해 방송 서버의 간섭 없이 모바일 클라이언트에서 직접 트랜잭션을 수행하고, 완료 정보만 방송 서버로 전송하므로 병목현상을 감소시켰다.



[그림 1] 전체 시스템 구성도

본 연구에서는 낙관적 동시성제어 기법을 사용하고, 일관성과 신속한 회복을 위해 SLL 기법을 제안 하였다.

③ 각 모바일 클라이언트는 트랜잭션 완료 후 재수행 로그를 이용하여 완료 트랜잭션 로그레코드(Commit_Transaction Log List : CTLL)를 작성하여 서버 전송한다. 서버는 SLL의 해당 세그먼트 로그정보를 찾는다. 정보가 없으면 낙관적 동시성 기법에 의해 로그(CTLL)를 SLL에 기록하고, Dirty 세그먼트를 메인 메모리 데이터베이스에

반영한다.

그리고 해당 모바일 클라이언트에게 완료 메시지를 전송한다. 정보가 있으면 해당 모바일 클라이언트에게 재수행 메시지를 보낸다. SLL 정보는 주기적인 체크포인트 방법에 의해 관리한다. 구체적인 프로토콜은 제 4장에서 기술한다.

3.1 방송 서버의 구성

모바일클라이언트/방송서버 환경에서 방송 서버의 역할은 데이터베이스 관리, 로그버퍼와 세그먼트 완료 리스트(Segment Log List : SLL), 동시성 제어, 체크 포인트 기법, 그리고 백업처리기를 관리 담당한다. 기존 기법은 데이터베이스를 디스크에 저장하여 관리함으로서 디스크 입출력으로 빠른 응답을 할 수 없다는 문제점이 제기된다. 본 연구는 이런 문제점을 제거하기 위해 메인 메모리 장치에 전체 데이터베이스를 관리하고 디스크에는 백업용 데이터베이스를 저장 유지 하도록 구성 하였다.

본 연구에서의 로깅 기법은 지연기법을 사용한다. 지연기법은 갱신된 연산을 가지므로 재 수행만을 실시하는 장점을 가지고 있다.

3.1.1 로그버퍼와 SLL

각 모바일 클라이언트들이 트랜잭션을 완료한 후, 방송 서버에게 완료 트랜잭션 로그 레코드(Commit Transaction Log Record : CTLR)를 전송한다. 이때 방송서버는 모바일 클라이언트로부터 전송받은 CTLR를 기록하기 위해 SLL 구조가 필요하다. SLL는 트랜잭션 충돌여부의 체크 수행 동작 때 필요하고, 또한 체크포인트에 의해 시스템 백업 후 제거된다.

3.1.2 회복 관리기(recovery manager)

모바일 클라이언트-방송서버 환경에서 예기치 못한 파손이 발생할 경우 데이터베이스의 일관성이 유지되게 회복동작이 반드시 필요하게 된다. 회복시스템 모델은 다음과 같다. 첫째, 방송서버는 로그파일에서 세그먼트별 로그를 재구성한다.

둘째, 로그를 분석하여 재수행 로그만 메인 메모리 데이터베이스에 반영하면 된다. 그 이유는 모바일 클라이언트에서 트랜잭션 수행 완료 후 재수행 로그만 서버에 전송하기 때문이다.

3.1.3 동시성 제어

동기화 및 일관성을 위한 동시성제어 개념은 다음과 같다.

첫째, 본 연구에서는 낙관적 동시성제어 기법을 사용한다. 둘째, 모바일 클라이언트가 트랜잭션 수행완료 후 서버에게 완료 트랜잭션 로그 레코드를 전송한다. 셋째, 서버는 동시성제어 및 회복관리를 위해 완료 트랜잭션 로그 레코드를 가지고 SLL을 구축한다.

SLL 구축시 다른 모바일 클라이언트에서 수행한 완료 트랜잭션 로그 레코드가 있는지 조사한다. 이 때 비 존재(비 충돌)시 모바일 클라이언트에게 완료를 통보한다. 존재(충돌)시 낙관적 동시성제어 알고리즘에 따라 처리한다. SLL과 낙관적 기법의 프로토콜은 제 4장에서 기술한다.

넷째, SLL의 로그파일은 체크포인트기법에 의해 관리된다.

3.1.4 백업 처리기

주기적으로 데이터베이스를 안정된 보조기억 저장 장치에 기록하여야 한다. 이때 백업 처리기는 버퍼크기와 SLL크기를 고려하여 일정한 주기로 백업처리를 담당한다.

3.2 모바일 클라이언트의 구성

모바일 클라이언트는 로그버퍼와 세그먼트 버퍼, 그리고 세그먼트버퍼로 구성된다.

모바일 클라이언트 로그버퍼는 구성은 <표 1>과 같다.

그 로그버퍼는 철회수행 로그레코드(undo log

〈표 1〉 모마일 클라이언트 로그버퍼

항 목	설 명
로그 종류	재수행 로그, 철회 로그
연산 종류	실행, 완료
세그먼트 아이디	연산이 수행된 세그먼트의 식별자
세그먼트 위치	연산이 수행된 세그먼트의 데이터 위치
변경 전 데이터	연산 수행 전의 데이터
변경 후 데이터	연산 완료 후의 데이터

record)와 재수행 로그 레코드(redo log record)를 유지 관리하며, 세그먼트 버퍼는 방송 서버에서 전송 받는 세그먼트를 저장하기 위한 기억 공간이다. 철회수행은 트랜잭션을 철회시 로그 레코드를 이용하여 모바일 클라이언트가 철회를 직접 수행한다. 본 연구 환경은 메인 메모리 DB이고, 낙관적 기법을 사용함으로 DIRTY 세그먼트를 철회동작 할 수 있도록 서론에서 가정하였다. 재수행 로그 레코드는 트랜잭션 수행 완료시, 그 결과의 로그정보를 방송 서버에 전송하기 위한 레코드이다.

3.3 모바일 컴퓨팅 회복 시스템 모델

3.3.1 트랜잭션 정의

본 연구에서 가정하는 모바일 클라이언트-방송서버 데이터베이스 시스템에서는 전송단위가 세그먼트이고, 트랜잭션은 모바일 클라이언트에서만 처리하는 것으로 가정한다.

한 트랜잭션의 처리가 시작될 때 로그에 <Ti, starts>라는 로그레코드를 기록하고, 트랜잭션처리 시 마다 데이터의 갱신을 기록하기 위해 <Ti, data item, new value>의 로그 레코드 형식을 로그에 추가 한다. 그 트랜잭션이 부분 종료시 <Ti, commit>라고 로그에 기록 한 뒤, 그 다음에 변경된 값이 모바일클라이언트 로그버퍼에 기록한다. 이 과정을 하나의 트랜잭션이라 한다.

3.3.2 회복 모델

모바일 컴퓨팅 환경에서 발생할 수 있는 고장유형은 모바일 클라이언트 고장, 방송서버의 고장, 네트워크의 고장, 트랜잭션 철회 등으로 분류할 수 있다.

모바일 클라이언트와 방송서버 고장은 배터리 정전 등으로 인한 작동하지 못하는 경우에 발생한다. 네트워크의 고장은 통신망의 문제로 인하여 메시지를 전송할 수 없는 경우에 발생한다.

그리고 제안하는 기법에서 트랜잭션 철회는 기존의 모바일 클라이언트가 데이터 세그먼트를 캐쉬에 따른 많은 복잡성을 제거하기 위하여 캐쉬하지 않고 하나의 트랜잭션 수행에 필요한 데이터 세그먼트만 방송서버로부터 전송받아 수행한다.

이때의 트랜잭션 종류는 두 가지로 분류한다. 하나는 판독전용(read only) 트랜잭션이고, 다른 하나는 쓰기연산(update operation) 트랜잭션으로 수행된다.

기존의 기법들은 방송서버에서 트랜잭션 처리 후 모바일 클라이언트로 전송함으로써 방송서버의 병목현상과 캐쉬 일관성 문제 등의 문제점이 내포하고 있다.

본 회복모델의 모든 연산은 모바일 클라이언트가 수행하고, 판독전용 트랜잭션 완료 후 종료하고, 갱신연산 트랜잭션은 수행완료 그 결과 중에서 재수행 로그만 서버로 전송한다. 그 결과 병목현상과 캐쉬에 따른 알고리즘의 복잡성을 감소시켰다.

네 가지 고장의 유형에 따른 회복방법을 살펴보면 다음과 같다. 첫 번째, 모바일 클라이언트 회복은 재가동 후 방송서버로부터 트랜잭션 처리에 필요한 세그먼트를 전송받아 ST에 저장한 후 트랜잭션을 수행한다.

두 번째, 방송서버 회복은 모바일 클라이언트들은 정상적으로 수행중인 트랜잭션을 수행 완료 후에 로그 정보를 버퍼에 보관한다. 방송서버는 재가동 후 로그정보를 이용하여 SLL를 구성한다. 그리고 방송서버는 모바일 클라이언트들에게 복구완료메시지를 전송한다. 모바일 클라이언트는 메시지를 받으면 보관 중인 정보를 서버로 전송한

후 정상적으로 수행한다.

세 번째는 네트워크 회복은 방송서버와 모바일 클라이언트는 진행 중인 트랜잭션은 수행 완료 후 대기한다. 네트워크의 정상회복 ACK메시지를 받으면 정상적 트랜잭션을 수행한다.

네 번째는 트랜잭션 회복은 판독전용 트랜잭션과 갱신 전용 트랜잭션이 철회(Abort)된 경우 방송서버로부터 필요한 세그먼트를 재전송 받아 다시 수행한다.

본 논문에서는 고장에 대한 회복기법을 연구 하였다.

3.3.3 회복 트랜잭션 가정

제안하는 회복알고리즘을 위해 사용되는 가정들은 다음과 같다.

① 데이터베이스 일관성을 위한 각 사이트 수행 로그정보는 방송서버에만 저장한다. 이렇게 함으로써 중복저장에서 발생하는 많은 메시지 교환과 알고리즘의 복잡성 문제를 해결할 수 있다.

② 모바일클라이언트에서는 트랜적션 수행과 철회 연산만 수행하고, 방송서버는 각 사이트에서 수행 완료한 세그먼트 로그정보를 대상 으로 한다.

본 논문에서는 모바일 클라이언트와 방송서버의 트랜잭션의 회복방법에 대하여 기술한다.

③ 고장은 메시지를 보낸 후 일정한 시간동안 응답이 없으면 고장으로 인식하거나 일정한 고장 탐지 알고리즘에 의해 인식될 수 있다.

④ 본 논문에서는 모바일 컴퓨팅 환경에서의 발생 가능한 네 가지 고장유형 가운데 모바일클라이언트와 방송서버 고장에 대한 회복기법을 다룬다.

4. 제안하는 회복시스템 구조

4.1 자료구조

4.1.1 모바일클라이언트내의 로그구조

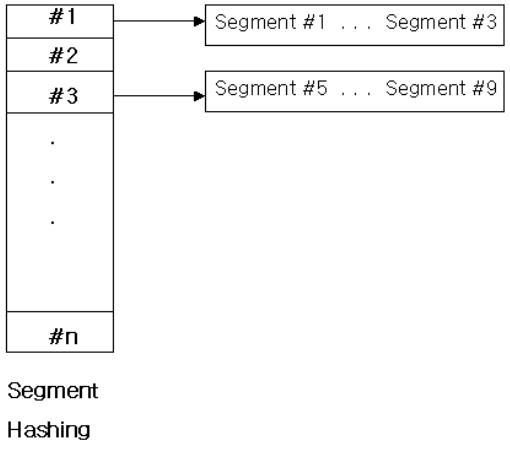
모바일 클라이언트 내에는 하나의 트랜잭션을 저장할 수 있는 로그버퍼가 존재하고 있고, 로그버퍼는 재수행 로그와 철회수행로그로 구성되어 있다. 그 구조는 다음과 같다.

〈표 2〉 로그레코드 구조

항 목		설 명
재수행 로그	LSN	로그일련번호
	세그먼트 번호	트랜잭션 수행 중인 세그먼트 번호
	OFFSET	트랜잭션 수행 중인 데이터
	NEW VALUE	갱신 데이터
철회수행 로그	LSN	로그일련번호
	세그먼트 번호	트랜잭션 수행 중인 세그먼트 번호
	OFFSET	트랜잭션 수행 중인 데이터
	OLD VALUE	원시 데이터

4.1.2 데이터 세그먼트 구조

본 논문에서 제안하는 회복방법을 위해 유지되는 로그구조는 다음과 같다.



[그림 2] 세그먼트 버퍼 구조

세그먼트 버퍼 구조는 [그림 2]같은 형태로 방송서버의 메인 메모리 데이터베이스에 유지되며, 방송서버는 텍스트 데이터와 멀티미디어 데이터를 유지할 수 있는 세그먼트 단위로 저장한다.

즉 하나의 데이터 아이템이 여러 세그먼트에 분산 저장될 수 있도록 허용함으로써, 멀티미디어

데이터를 저장할 수 있고 또한 단편화 현상을 제거하였다.

그리고 세그먼트 테이블에는 각 세그먼트별로 dirty bit을 두어 데이터베이스 백업 받을 때 갱신된 페이지만을 디스크에 백업을 받는다. 이와 같이 할 경우 전체 데이터베이스를 백업받을 때 걸리는 시간에 대한 오버헤드를 제거할 수 있다.

4.1.3 완료 트랜잭션의 로그구조

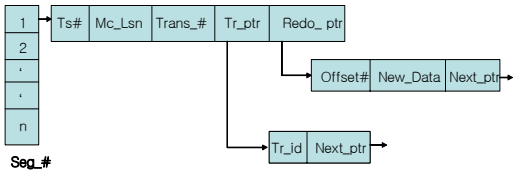
모바일 클라이언트는 <표 3>와 같은 구조로 트랜잭션이 성공적으로 완료한 후 재수행 로그를 이용하여 완료 트랜잭션 레코드를 작성한다. 이 리스트는 먼저 트랜잭션수행을 시작하면서 세그먼트 번호(Seg#)와 트랜잭션을 수행하는 모바일 클라이언트 로그일련번호(Mc_Lsn), 그리고 회복제어를 위한 타임 스템프(Time_stamp)를 구성한다. 그리고 완료된 트랜잭션 번호(Tr_ptr), 완료된 트랜잭션에 의해 수행된 재수행 로그정보(Redo_ptr)로 구성된다.

〈표 3〉 Commit_Transaction 로그 레코드

항 목	설 명
세그먼트번호	트랜잭션에 사용된 세그먼트번호
모바일 클라이언트 로그번호	해당 클라이언트별로 부여한 로그일련번호
트랜잭션 번호	세그먼트에 수행된 트랜잭션 일련번호
트랜잭션 포인터	트랜잭션에서 사용된 데이터 아이템별 번호
재수행 번호	재수행 로그레코드 일련번호

4.1.4 세그먼트 로그 리스트(Segment Log List)

각 모바일 클라이언트에서는 트랜잭션 수행 완료될 때마다 방송서버로 [그림 3]과 같은 완료 트랜잭션 로그 레코드를 만들어 전송한다. CTR을 전송받은 방송서버는 [그림 3] Segment Log List에 해당 엔트리의 맨 뒤에 삽입 한다. 여기서는 엔트리는 하나의 세그먼트에 대응한다. 이 저장구조를 SLL



[그림 3] Segment Log List

이라 한다. 이 구조는 세그먼트별로 디렉터리를 구성한다. SLL은 각각 5개의 필드로 구성된다. 첫 번째 필드는 트랜잭션의 충동여부를 위한 타임스탬프를 나타낸다. 두 번째 필드는 모바일클라이언트에서 트랜잭션 수행완료 로그레코드의 LSN 값을 나타낸다. 세 번째는 필드 트랜잭션 번호이고, 네 번째 트랜잭션 일련번호를 기록한다. 마지막 다섯 번째 필드는 재 수행 로그정보를 기록한다.

4.2 회복 알고리즘

기존의 모바일 클라이언트-방송서버 환경에서의 회복 기법은 모바일 클라이언트에서 생성된 로그 레코드와 페이지를 WAL 규약을 이용하여 방송서버에 전송하였다. 이렇게 함으로써 발생하는 문제점들을 해결하기 위해 방송서버에서는 ARIES의 무조건 철회 동작을 수정한 조건부 철회 동작을 수행해야 한다[5, 11]. 또한 dirty된 페이지의 방송서버 도착 사실로부터 해당 페이지가 dirty되었음을 SLL에 기록하기 때문에 발생하는 문제점도 해결해야 한다[5]. 이와 같은 문제점 해결이 필요하게 된 것은 모바일 클라이언트는 로그 레코드와 세그먼트를 방송서버에 전송하기 때문이다. 항상 로그 레코드는 그 레코드의 생성과 관련된 해당 세그먼트들을 함께 방송서버로 보내 주어야 한다. 본 논문은 로그 레코드와 해당 페이지를 방송서버에 보냄으로 발생하는 문제들을 제거하기 위하여 모바일 클라이언트는 방송서버에 로그 레코드만을 전송하도록 제안한다. 또한 기존의 회복 기법에서는 철회동작을 방송서버가 함으로써 시스템의 병렬성을 충분히 사용하지 못하였다[5].

본 논문에서는 시스템 병렬성을 충분히 사용하기 위하여 모바일 클라이언트에서 철회(abort)동작을 수행하도록 제안한다.

모바일 클라이언트와 방송서버환경에서 회복동작은 서버가 책임지며 모바일 클라이언트 파손의 경우와 방송서버 파손의 경우로 분류할 수 있다.

방송서버가 고장일 발생할 경우 모바일 통신네트워크를 통하여 모바일 사이트에 방송전파 메시지를 보낸다. 방송서버 고장이 발생한 후 다시 회복하게 되면, 고장으로부터 회복되었음을 모바일 클라이언트에게 방송전파로 알린다. 복구 메시지를 받은 모바일 클라이언트들은 완료 트랜잭션 로그 레코드가 있으면 방송서버로 전송한다. 그러면 방송서버는 고장동안 발생한 데이터베이스 갱신을 반영하여 일관된 데이터베이스 상태를 유지한다.

서버고장은 트랜잭션의 수행결과에 따라 세 가지 유형으로 분류할 수 있다. 첫째는 모바일 클라이언트에서 트랜잭션을 수행 중에 서버가 고장이 발생하는 경우이고, 둘째는 모바일 클라이언트가 트랜잭션 수행 완료 후 완료 트랜잭션 로그 레코드를 방송서버로 전송하여 서버가 SLL을 통해 완료/철회(COMMIT/ABORT)를 체크 한 후 서버로부터 철회 결정 받은 후 재제출하기 전에 방송서버가 고장이 발생한 경우이고, 세 번째는 방송서버에서 모바일 클라이언트로 완료(COMMIT) 결정을 전송 후모바일클라이언트에게 도착하기 전에 서버 고장이 발생한 경우이다.

이런 경우 해결책은 다음과 같다. 첫 번째는 모바일 클라이언트에서 계속 수행 중인 트랜잭션을 수행 완료 후 완료 트랜잭션 로그 레코드를 작성하여 로그버퍼에 저장한 후, 방송서버가 복구메시지를 받으면 정상적으로 작업을 수행하면 된다. 두 번째일 경우는 방송서버로부터 복구 메시지를 받으면 재제출하여 정상적인 작동을 수행한다. 세 번째는 방송서버가 복구직후 SLL을 통하여 고장직전의 완료(COMMIT) 정보를 재구성하여 완료메시지를 모바일 클라이언트에게 재전송한다.

모바일 클라이언트와 방송서버환경은 다수의 모바일 사이트들과 하나의 서버로 구성된다. 그리고 모바일 클라이언트는 핸드오버와 제한된 배터리 전력으로 인한 파손 발생률은 방송서버의 파손 발생률에 비해 상대적으로 훨씬 높다고 할 수 있다. 그러므로 모바일 클라이언트 파손에 대한 효율적 회복 알고리즘이 필요하다. 다음은 구체적 회복 알고리즘이다.

4.2.1 정상적인 상태에서 알고리즘

1) 방송서버알고리즘

방송서버의 알고리즘은 다음과 같다.

① SV1 : While (CTR = commit) Do
send the TR_commit result
to the Transaction mobile client.
② SV2 : While (CTR = Abort) Do
send the TR_abort result to
Transaction mobile client.
③ Broadcasting cycle :
While (BR_time = true) Do
sent data_segment to mobile clients.
④ time = time + count ; goto SV1.

2)모바일 클라이언트

모바일 클라이언트의 알고리즘은 다음과 같다.

① MC1 : Put data_seg to segment buffer
② Begin transaction
write redo record to log_buffer
write undo record to log_buffer
End transaction.
③ Write ctr
④ Submit ctr to the Server
⑤ Do until receive a result from server
Wait ;
⑥ IF (receive a result from the server)
then switch = result flag
⑦ While result flag = commit Do
remove the log_buffer and data segment
⑧ While result flag = abort Do

goto mc1 ;

4.2.2 방송서버고장 상태에서 알고리즘

1) 모바일 클라이언트 알고리즘

장애 후 회복 알고리즘은 다음과 같다.

① server failure is detected
② Begin transaction
write redo record to log_buffer
write undo record to log_buffer End transaction
③ Write ctr
④ Do until recovery_message from the server Wait

2)방송서버 알고리즘

장애 후 회복 알고리즘은 다음과 같다.

① IF (server is recovered from failure) THEN send the recovery_message to all mobile_clients ;
② Restructure Directory SLL ;
③ Produce ctr_list ;
④ Send ctr_ list to all mobile_clients ;
⑤ Restart operation ;

4.3.3 모바일 클라이언트 고장회복 알고리즘

장애 후 회복 알고리즘은 다음과 같다.

① While Receive = recovery message Do
② cache data_seg to segment buffer
③ Begin transaction
write redo record to log_buffer
write undo record to log_buffer
End transaction
④ Write ctr
⑤ Submit ctr to the Server ;
⑥ Do until receive a result from server Wait ;
⑦ While receive a result from the server Do switch = result flag
⑧ commit :
remove the log_buffer and data segment
⑨ abort : goto mc1 ;

5. 비교 분석

본 장에서는 제안한 모바일 컴퓨팅 회복제어 기법과 기존의 관련된 연구 기법을 비교 분석한다.

본 논문이 제안한 회복제어 알고리즘의 특징을 살펴보면 다음과 같다.

1. 본 논문에서 방송서버는 수행 완료된 트랜잭션들의 정보를 SLL에 저장하여 관리한다.
2. 모바일 클라이언트는 수행 완료된 세그먼트는 전송하지 않고, 완료 트랜잭션 로그 레코드를 방송 서버로 전송한다.
3. 기존의 방법 에서는 서버에서 트랜잭션을 철회 동작 하지만 본 논문에서는 모바일 클라이언트에서 철회 동작을 한다.
4. 본 논문에서 모바일 클라이언트들이 동시에 같은 세그먼트를 사용할 수 있다.
5. 모바일 클라이언트는 수행 완료된 트랜잭션을 방송 서버에 있는 SLL을 이용하여 회복동작을 신속히 처리한다.

본 논문에서 제안한 방법을 기존의 기법과 비교할 때 다음과 같은 장점을 갖게 된다.

1. 트랜잭션 종료 시 방송서버에게 완료로그 레코드만 전송하므로 WAL규약이 필요하지 않다. 그 결과 회복 시간을 단축하였다.
2. 기존 기법과 달리, 즉 타임 스템프 체크포인트 방식이 아닌 모바일 클라이언트별로 로그 일련번호를 부여함으로써 재수행 동작에 대한 오버헤드가 감소하였다.
3. 기존기법 들은 데이터와 로그를 함께 캐쉬하고 있어 쓰기연산 때 다른 모바일 클라이언트에 캐쉬된 록을 회수(callback)해야 한다. 이렇게 할 경우 발생하는 록 오버헤드를 감소시켰다.

6. 성능평가

본 장에서는 논문에서 개발한 모바일 메인 메모리 DBMS를 위한 회복시스템의 성능 평가 결과를 살펴본다. 성능평가를 위해 사용된 시스템의 하드웨어 사양은 인텔 CPU 2.40GHz, 2GB RAM을 사용하였다. 그리고 실험 모델에 대하여 기본 매개변수를 선정하고, 설정된 매개 변수를 사용하여 다음과 같은 순서로 진행한다.

먼저 Extending ARIES-CS기법을 본 연구 환경에 맞도록 변화시켜 로깅 기법, 회복 기법 그리고 동시성 제어 기법 등에 대해서 모델링을 위한 분석을 한다. 그리고 본 제안기법을 트랜잭션 양의 변화에 따른 로드, 세그먼트 크기 변화에 따른 로드, 고장에서 회복하여 서비스하는데 걸리는 시간으로 나누어 분석을 한다.

6.1 실험 성능 측정

본 실험은 CSIM 라이브러리를 이용하여 데이터 충돌이 발생할 수 있는 파라미터를 변경하면서 그것이 성능에 미치는 영향을 측정하였다. <표 4>는 시뮬레이션 매개 변수를 제시하였다.

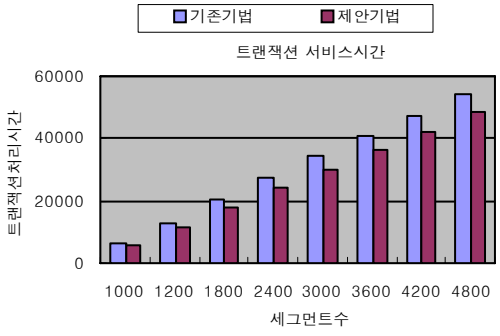
〈표 4〉 시뮬레이션 매개 변수

매개 변수	값
데이터베이스 크기	256Mbyte
레코드 크기	32 words
세그먼트 크기	8192 words
세그먼트 요청	EXPON(10)
세그먼트를 주 기억 장치로 load 하는데 걸리는 시간	87.8ms
트랜잭션 처리시간 + 로그 생성 시간	2ms
로그 디스크에 쓰는 시간	87.8ms
로그 분석 + 재 수행	2ms

6.2 성능 비교 분석

6.2.1 회복하여 서비스시간 성능평가

[그림 4]는 본 논문에서 제시한 기법을 사용하는 회복기법과 기존기법을 사용한 회복시스템과의 시스템 오류 발생 후 회복성능을 비교한 그래프이다.



[그림 4] 트랜잭션 서비스시간

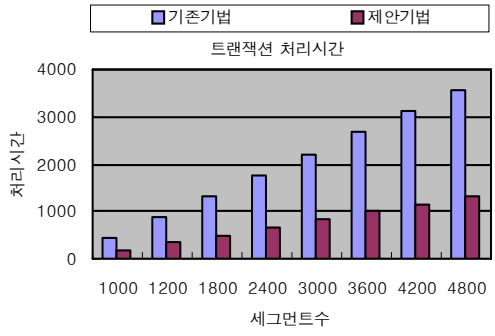
입력된 세그먼트의 수는 1000에서 4800을 발생시켰다. 회복하여 서비스시간(초 단위)이 0에서 60000까지 증가됨에 따른 처리시간을 나타낸 것이다. 본 비교에서는 트랜잭션 처리시간은 고려하지 않고, 단지 시스템이 파손 후 방송서버를 회복하여 모바일 클라이언트에게 해당 세그먼트를 전송하는데 까지 걸리는 시간을 분석하였다. 이 실험 결과 전체적으로 제안기법이 기존기법 보다 우세한 것으로 분석되었다. 본 기법의 성능이 훨씬 뛰어난 이유는 기존기법의 회복 동작은 로그 분석, 재수행, 철회 동작으로 이루어져 처리 시간이 너무 많이 소요되기 때문이다. 본 기법은 간단한 로그 분석과 재수행 동작만 이루어진다. 따라서 입력되는 세그먼트수가 증가함에 따라 상당한 시간이 단축되었다.

6.4.2 로깅 기법을 이용한 트랜잭션 처리 시간 로드 비교

[그림 5]는 입력되는 세그먼트수의 크기가 변화됨에 따라 부가되는 처리시간을 성능평가 하였다. 트랜잭션의 처리 시간(초 단위)은 0에서 4000까지 증가됨에 따라 처리시간을 나타낸 것이다.

본 비교에서는 트랜잭션 처리 시간 동안 시스템 파손 없이 정상적으로 작동됨을 가정하였다. 그

결과 제안된 기법이 기존기법보다 우수한 것으로 나타났다.



[그림 5] 트랜잭션 처리시간

그 이유는 크게 세 가지로 생각 할 수 있다. 첫째, 모바일 클라이언트에서 트랜잭션이 발생하면 해당 페이지를 서버에게 요구한다. Exodus기법은 서버에 있는 DPT(Dirty Page Table)를 이용하여 전송하므로 DPT에 대한 로드가 많은 반면 제안기법은 SLL을 이용하므로 신속히 모바일 클라이언트에게 전송 할 수 있다. 둘째, 모바일 클라이언트에서 트랜잭션을 수행할 때 로그부여방법이 다르다. 기존기법은 로그 일련 번호(Log sequence Number : LSN)와 LRC(Log Record Number)를 사용하므로 로드(Load)가 많은 반면 본 기법은 모바일 클라이언트별로 로그를 부여함으로써 오버헤드가 적다. 셋째, 트랜잭션 종료 후 서버로 트랜잭션 결과를 전송시 Dirty 여부 문제이다. 기존기법은 로그 레코드와 해당 페이지를 전송하지만 본 기법은 로그 레코드만 전송하므로 시간이 단축되었다.

7. 결 론

본 논문에서는 모바일 메인 메모리 데이터베이스 환경에서 발생하는 여러 가지 문제점들을 제시하고, 이 문제점을 해결을 위한 효율적인 회복기법을 제안하였다.

제안하는 효율적인 회복기법은 세그먼트 디렉토리를 이용한 회복모델을 설계하였다. 이 모델에서의 방송서버는 메인 메모리 데이터베이스 구축하고, 각 모바일 클라이언트에게 세그먼트를 제공한다. 그 결과 제안된 대역폭으로 인한 병목현상을 줄이고, 분산 트랜잭션을 수행 할 수 있도록 설계하였다. 제안하는 회복기법은 세그먼트 로그 리스트 방법이므로 회복의 재수행 시간을 단축하였다.

이런 환경에서 성능평가를 위해 시뮬레이션을 실시하였다. 그 결과 살펴보면, 제안된 기법이 타 연구에 비하여 약간 우수함으로 나타났다. 앞으로의 연구는 동시성제어에 대한 연구가 필요하다.

참 고 문 헌

[1] Bernstein, P., V. Hadzilacos, and N. Goodman, “Concurrency control and Recovery in Database Systems”, *Addition-Wesley*, (1987), pp.217-240.

[2] Carey, M., D. Dewitt, J. Richardson, and E. Schekita, “Storage Management for Objects in EXODUS”, *in Object-Oriented Concepts, Databases, and Applications, W. Kim F. Lochovsky, Addition-Wesley*, (1989), pp.341-369.

[3] Daniel Barbara, Tomasz Imielinski, “Sleepers and Workaholics : caching Strategies in Mobile Environments”, *VLDB*, (1995), pp.1-12.

[4] Daniels, D., A. Spector, and D. Thompson, “Distributed Logging for Transaction Processing”, *Proc. ACM SIGMOD Conf.* San Francisco, (1987), pp.13-25.

[5] Franklin, M., M. Zwilling, C. Tan, M. Carey, and D. Dewitt, “Crash Recovery in Client-Server EXODUS”, *TR #1081, Comp Sci Dept., Univ. of Wisconsin-Madison*, (1992), pp.165-174.

[6] Kam-Yiu Lam, Tei-Wei Kuo, Wai-Hung Tsang and Gray C. K. Law “Concurrency

Control in Mobile Distributed Real-Time Database Systems", *Information Systems*, Vol25, No.4(2000), pp.261-322.

[7] James J. and Kistler, M. Satyanayanan, "Disconnected Operation in the Coda File System", *ACM*, (1992), pp.213-225.

[8] Wang, Y. and L. Rowe, "Cache consistency and Concurrency Control in a Client-Server DBMS Architecture", *Proc. ACM SIG MOD*, Denver, (1991), pp.367-376.

[9] The Committee for Advanced DBMS Function, "Third Generation Data Base System Maifesto", *SIGMOD Record*, Vol.19, No.3 (1990).

[10] Man Hon Wong, and Wing Man Leungm, "A Caching Policy to Support Read-only Transaction in a Mobile Computing Environment", *CS-TR-95-07*, 1995.

[11] Jin, Jing and Omran, Bukhres, "Distributed lock management for mobile transaction", *proceedings of the 15th IEEE international conference on Distributed computing systems*, (1995), pp.118-125.

[12] Kam-Yiu, Lam and Mei-Wai, Au, "Broadcasting Consistent Data to Read-only Transaction from Mobile Clients", *The Computer Journal*, Vol.45 No.2(2002), pp.129-146.

[13] Dewitt, D., P. Futtersack, D. Maier, and F. Velez, "A Study of Three Alternative Workstation-Server Architectures for Object-Oriented Database Sytems", *Pro., 16th VLDB conf.,* (1990), pp.35-49.

◈ 저 자 소 개 ◈

조 성 제 (chosj715@yahoo.co.kr)

현재 동방대학원대학교 문화정보학과 교수로 재직 중이며, 홍익대학교 전자계산학과 박사를 취득하였다. Journal of Microelectronics and Reliability(SCI) 등의 국제학술지 및 한국통신학회지, 한국정보처리학회지, 인터넷정보학회지 등의 국내학술지에 논문을 게재한 바 있다. 주요 관심분야는 메인 메모리 DBMS 및 모바일 컴퓨팅, 문화 콘텐츠 개발, 정보보안, ERP (Enterprise Resource Planning)시스템 구현, S/W개발 프로젝트관리 등이다.