메인 메모리 DBMS P*TIME 기반 WAP 서버 구현

윤용식, 노재윤, 송창빈, 차상균
서울대학교 전기컴퓨터공학부
{dandy79, sincere6, tsangbi, chask}@kdb.snu.ac.kr

Implementation of WAP Server using Main Memory DBMS: P*TIME

Yong-Sik Yoon, Jae-Yun No, Chang-Bin Song, Sang-Kyun Cha
School of Electrical Engineering and Computer Science, Seoul National University

요 약

최근 개인 휴대폰의 대중화에 따라 기존의 유선 인터넷 환경과 휴대폰 서비스가 결합된 무선 인터넷 서비스가 각광을 받고 있다. 현재 웹을 기반으로 WAP 인터페이스를 제공하는 회사들은 디스크 기반 DBMS를 사용한다. 데이터의 전송 속도가 유선 상에 비해 떨어지고 회선이 불안정한 무선 인터넷의 특성상, 응답 시간이 느릴 경우 사용자는 서비스 이용에 큰 불편을 느낄 수밖에 없고, 이용상에 오류가 생길 가능성이 높아진다. 이에 비추어 볼 때 응답 시간은 기존의 유선 인터넷 서비스에 비해 더욱 중요한 요소로 작용한다. 디스크 기반 DBMS는 이와 같은 요구 조건을 만족하지 못한다. 그에 반해 구조가 간단하면서 높은 성능을 낼 수 있는 메인 메모리 DBMS는 무선 인터넷 서비스에 적합한 환경을 제공한다.

본 논문에서는 2세대 고성능 메인 메모리 DBMS인 P*TIME을 이용하여 무선 인터넷 서버를 구현하고, 그 성능을 상용 DRDBMS인 Oracle을 이용했을 때의 성능과 비교 분석해 본다. WML, WML Script, JSP를 이용하여 PIMS 서비스를 구현해 본 결과, 동시 접속자 수가 늘어남에 따라 Oracle과 P*TIME의 성능 차이는 더욱 크게 벌어져 동시 접속자 수가 50명일 경우 P*TIME이 100배 이상의 성능을 보였다.

1. 서론

최근 개인 휴대폰의 대중화에 따라 기존의 유선 인터넷 환경과 휴대폰 서비스가 결합된 무선 인터넷 서비스가 각광을 받고 있다. 이에 따라 기존의 인터넷 서비스와 차별화된 무선 인터넷 서비스에 최적화된 환경에 대한 연구가 활발히 이루어지고 있다. 무선 인터넷 환경에서 데이터의 접근과 관리를 담당하는 DBMS는 서비스의 품질을 결정짓는 가장 중요한 부분이다.

현재 웹을 기반으로 WAP(Wireless Application Protocol) 인터페이스를 제공하는 회사들은 대부분 DBMS로 디스크 기반 DBMS(Disk Resident DBMS, DRDBMS)인 Oracle등의 DRDBMS를 사용한다. 데이터 전송 속도가 유선 상에 비해 떨어지고 회선이 불안정한 무선 인터넷의 특성상, 서버에서 응답 시간이 느릴 경우 사용자는 서비스 이용에 큰 불편을 느낄 수밖에 없고, 이용상에 오류가 생길 가능성이 높아진다. 이에 비추어 볼 때 무선 인터넷의 경우 응답 시간은 유선 인터넷의 그것에 비해 더욱 중요한 요소이다. 현재 널리 사용되는 DRDBMS는 이와 같은 요구 조건을 만족하지 못하므로 고성능 하드웨어를 이용하거나 서버를 여러 대 운용하는 방법을 택하고 있다. 그러나 이 방법은 경제적으로 부담도 크고, 가격에 비해 만족할 만한 성능을 얻지 못하는 경우가 많다. 이에 대안으로 구조가 간단하면서 높은 성능을 낼 수 있는 메인 메모리 DBMS(Main Memory DBMS)에 대한 관심이 높아지고 있다.

또한 DRAM 가격이 지속적으로 하락하여 최근에는 서버용 메모리 모듈 가격이 기가 바이트 당 1000 달러 이하로 형성되었고, 64비트 어드레싱이 일반화되면서 수십 GB의 메모리를 장착한 멀티 프로세스 서버 플랫폼을 쉽게 갖출 수 있게 됨에 따라 MMDBMS가 현실적인 대안으로 떠오르고 있다.

본 논문에서는 MMDBMS의 효용성을 확인해 보기위한 WAP 어플리케이션으로 비교적 데이터 접근이 많이 일어나는 PIMS(Personal Information Management System) 서비스를 직접 구현해 보고 성능을 분석하였다. DBMS로는 2세대 고성능 메인 메모리 DBMS인 P*TIME(Highly Parallel Transact In Memory Engine)을 이용하였다.

본 논문의 구성은 다음과 같다. 2절에서 WAP 서버의 구조에 대해 설명하고, 3절에서 구현 시스템의 DBMS로 MMDBMS인 P*TIME을 이용할 경우와, 상용 DRDBMS인 Oracle을 이용할 경우의 성능을 비교 분석하고, 4절에서 결론을 낸다.

2. P*TIME 기반 WAP 서버 구조

2.1. 개요

WAP서버는 클라이언트의 입력 내용을 DBMS에 저장하고, 클라이언트의 요청이 있을 때 DBMS로부터 정보를 받아서 클라이언트에 돌려주는 역할을 한다. 무선 환경은 유선에 비해 느리고, 불안정하다. 그러므로 서버는 최대한 빠른 속도를 제공할 수 있도록 디자인되어야 하며, 그와 동시에 데이터 송수신 중 일어날 수 있는 많은 예외 사항을 처리할 수 있도록 설계되어야 한다.

본 논문은 DBMS로 2세대 메인 메모리 DBMS인 P*TIME을 사용하였다. P*TIME은 L2 캐쉬에 최적화된 인덱스, 동시성 제어 기법 등을 사용하여 검색 성능이 뛰어나다. 뿐만 아니라 고유 기술인 디퍼렌셜 로깅을 이용하여 로깅과 회복을 완전히 병렬적으로 수행할 수 있으므로 갱신 성능이 다른 디스크 및 메인 메모리 DBMS에 비해 월등히 뛰어나, 높은 갱신 성능을 요구하는 분야에서 유일한 경제적 대안으로 부각되고 있다.

2.2. 서버-클라이언트 인터페이스

WAP 스펙에서는 서버와 클라이언트 인터페이스를 위하여 WML을 제공한다. HTML과는 달리 컴파일된 코드를 전송하는 WML은 코드 전체를 전송하는 HTML보다 전송 데이터 양이 적으므로 무선 통신에 적합하다는 장점을 가진다. 하지만 WML만으로는 단지 정적인 페이지만을 생성할 수 있다는 단점을 가지고 있다. 이러한 단점을 보안하기 위해서는 스크립트 언어가 병용되어야 한다. 스크립트 언어로는 JSP를 사용하였다. JSP 엔진으로는 Tomcat 서버를 사용하였다.

2.3. 서버-DBMS 연결

JSP와 Java Beans가 Java 기술을 기반으로 하는 것이므로 JDBC를 사용할 경우 별도의 레이어가 불필요하다. P*TIME에서도 JDBC를 완벽히 지원하여 코드의 변경 없이 Oracle에 이식이 가능하다.

각각의 트랜잭션에 대하여 새로운 Connection을 open하고 close할 경우, 다수의 사용자가 이용하는 과정에서 상당한 병목 현상이 발생할 것이다. 이러한 병목 현상의 해소 방안으로서 사용자가 접속할 때 Connection을 open한 후, 서비스 종료 직전에 Connection을 close하는 방법이 있다. 이렇게 하면, Connection을 open, close하는 회수가 크게 줄어들기 때문에 시스템의 성능 향상을 기대할 수 있다. 이러한 Connection Pool 로직을 사용하여 여러 명이 동시 접속했을 시의 수행 환경을 크게 개선시켰다.

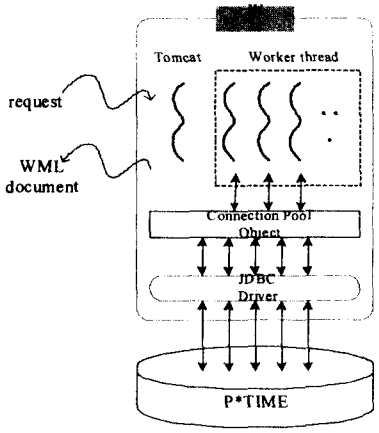

그림 1. WAP 서버의 전체적인 구조

전체적인 서버의 구조는 그림 1에 나타낸 것과 같이 Connect Pool Object를 두어 Connection 을 관리해 주도록 했다.

2.4. PIMS 서비스 구성

무선 인터넷 환경에서는 출력 스크린의 크기가 매우 작고, 그래픽 또한 크게 떨어진다. 이 때문에 일반 웹 상의 프로그램과 같은 기준으로 사용자 인터페이스를 구성할 경우 사용자는 큰 불편을 느끼게 된다. 즉 무선 인터넷의 유저 인터페이스의 경우 작은 화면에 최대한의 정보를 효율적으로 전달함과 동시에 최대한 이용 방법을 단순화하여 유저가 사용하기 편리하도록 설계하여야 한다.

PIMS Service에 접속하게 되면 우선 로그인 또는 등록을 선택할 수 있는 첫 페이지가 나타나게 된다. 이 때 이미 등록이 된 사용자이면 로그인을 거쳐 메인 페이지로 가게 되고, 등록

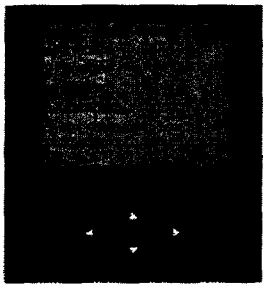

그림 2. 클라이언트 사용자 인터페이스

되지 않은 사용자이면 등록 과정을 거치고 난 뒤에 메인 페이지로 이동한다. 메인 페이지는 그림 2와 같다.

PIMS 서비스를 구현하는데 있어서 JSP의 장점을 최대한 살리는데 중점을 두었다. JSP를 사용하면 표현 부분과 프로그램 구현 부분의 분리가 용이하다. 이는 Java Beans 코드의 이용을 통해 가능하다. 데이터베이스에 접근하는 부분에 있어서는 Java Beans를 적극 활용하였다. 그럼으로써 JSP 페이지에서의 데이터베이스 접근은 단순히 Java Beans상의 매쏘드 호출로 이루어지게 된다.

무선 인터넷은 스크린의 크기가 작기 때문에, 한 화면에 다른 서비스로의 이동을 구현할 경우 많은 스크롤 과정을 거쳐야 한다. 스크롤하는 데 많은 시간이 소요되므로 사용자는 큰 불편을 느낄 수밖에 없다. 때문에 메인 페이지 상에서 자신이 참조하고자 하는 서비스를 먼저 선택하도록 구성하였다. 전체적인 서비스 구성은 그림 3과 같다. 이와 같이 구성할 경우 Events에서 ToDos등의 다른 Service로 전환을 위해서는 메인

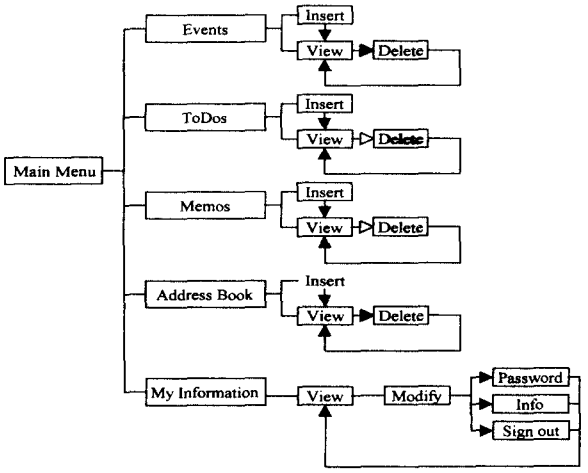

그림 3. PIMS 서비스 구성도

페이지로 돌아가야 한다는 단점이 있지만, 이로 인해 생기는 불편은 많은 스크롤로 인해 오랜 시간을 소모해야 하는 데서 발생하는 불편함보다 작다.

2.5. 테이블 구조

테이블은 총 6개로 구성된다. 로그인 관련 테이블인 User, 사용자 정보 테이블인 UserInfo, 일정 관련 테이블인 Events, 할일 관련 테이블인 Todos, 메모 관련 테이블인 Memos, 주소록 관련 테이블인 Address 테이블 등이 있다.

Table	Column
User	login, password, name
UserInfo	user_id, name, gender, birthdate, home_address, zip_code, company_name, company_address, office_phone, home_phone, fax, mobile_phone, e-mail, e-mail2
Events	user_id, id, title, contents, begin_time, end_time, type, updated, alarm, repeat_time, repeat_1, repeat_2
ToDos	user_id, id, parent_id, title, note, due, priority, done, updated
Memos	user_id, id, parent_id, title, contents, updated, event_id, todos_id
Address	user_id, id, name, e-mail, phone_number

표 1. 테이블 스키마(Table schema)

3. 성능 평가

구현된 PIMS 서버의 성능을 직접 테스트하였다. 테스트의 목표는 현재 구현한 PIMS Service가 초기의 목표대로 기존의 서비스보다 무선 인터넷에 더 적합한지를 알아보는 것이었다.

서론에서 밝힌 바와 같이 무선 인터넷에서 가장 중요한 점은 빠른 응답 시간을 보장하는 것이었다. MMDBMS인 P*TIME을 기반으로 만들어진 이 서비스를 DRDBMS의 대표라고 할 수 있는 Oracle에 이식 한 후 각각의 응답 시간을 측정하여 예상대로 MMDBMS가 무선 인터넷에 더 적합한 구조인지를 확인하였다.

3.1. 실험 환경

서버: CPU: 2 * UltraSPARC-II 400Mhz
OS: SunOS 5.7
Main Memory: 1GB
DBMS: P*TIME & Oracle
JAVA: JDK 1.3.1
JSP engine: Tomcat

실험 data 정보: 등록된 사용자 수: 100000명
테이블 당 레코드 수: 1000000개

3.2. 실험 시나리오 구성

제대로 된 성능 테스트를 위해서는 다수의 사용자가 동시에 접속한 것과 같은 유사한 환경을 시뮬레이트 해 주어야 한다. 이를 위해 HTTP 요청을 동시에 cast해주는 Java 쓰레드를 짧은 시간 간격으로 여러 개 생성하는 방법을 택하였다. Java 가상 머신(JVM)은 CPU 상에서 같은 우선 순위를 가진 쓰레드를 라운드 로빈(round robin) 방식으로 스케쥴링한다. 그런데 각각의 라운드 로빈 시간 간격(round robin time term)은 매우 작으므로, 실제 많은 클라이언트들이 동시에 접속하는 것과 유사한 효과를 낼 수 있다.

동시 접속하는 쓰레드들은 표 1의 Events 테이블을 access하도록 하였고, 클라이언트에서 Events 테이블에 일정을 입력하는 insert operation을 수행하는 데 걸리는 시간을 측정하였다.

3.3. 실험 결과

시나리오에 따라 실험을 수행하여 평균 응답 시간과 동시 접속자 수 간의 관계를 그래프로 나타내면 그림 4와 같다.

동시 접속자 수가 10명일 경우, P*TIME은 Oracle에 비해 10배 정도의 성능을 보여주었다. 동시 접속자 수가 50명인 경우에는 100배 정도 높은 성능을 나타내었다. 동시 접속자의 수가 증가할수록 성능의 차이가 현저해짐을 알 수 있었다.

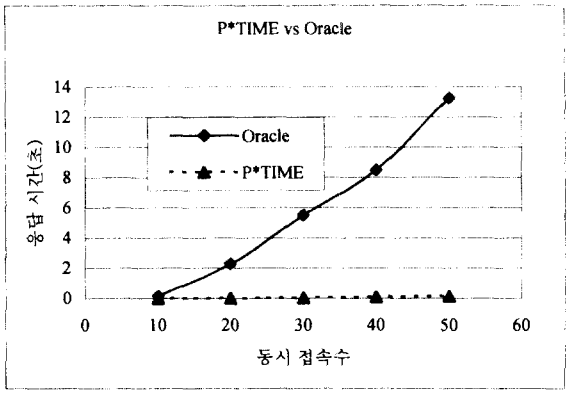

그림 4. P*TIME 과 Oracle의 응답 시간 비교

4. 결론

무선 인터넷의 경우 유선 인터넷에 비해 주고받는 데이터 양이 상대적으로 적고, 접속 상태가 불안정하다. 이런 상황에서는 빠른 시간 내에 데이터의 전송이 이루어 져야 한다. 본 논문에서 데이터의 빠른 처리와 서버의 부하를 줄이기 위해 Connection Pool 로직을 사용하여 여러 명이 동시 접속했을 시의 수행 환경을 크게 개선시켰다.

본 논문에서는 무선 인터넷 환경의 단점을 보완하기 위해 MMDBMS를 사용하였고 그 결과 빠른 시간 내에 데이터를 처리 할 수 있어서 응답 시간을 크게 줄였다. 그리고 본 논문에서 적용한 방법들의 타당성을 확인하기 위해 실험을 수행하였다. MMDBMS의 사용의 이점을 알아보기 위해 DRDBMS인 Oracle과의 비교 실험을 수행 하였다. 본론에서 볼 수 있듯이 실험 결과, MMDBMS의 사용은 수행 성능에 있어서 큰 효과를 나타냄을 확인하였다.

5. 참고 문헌

[1] Jeffrey D. Ullman, Jennifer Widom, *A First Course In database Systems*, Prentice Hall International, Inc., 1997.
[2] Danny Ayers, *Professional Java Server Programming*, WROX Press, Chicago, 2000.
[3] Karl Avedal, *Professional JSP*, WROX Press, Chicago, 2000.
[4] WAP Forum, http://www.wapforum.org/.
[5] Silberschatz, Galvin, Gagne, *Applied Operating Systems Concepts*, John Wiley & Sons, 2000
[6] 차상균, 송창빈, et al., "P*TIME: 고성능 메인 메모리 DBMS 구조와 활용", In *Proceedings of Korean Database Conference*, 2001.
[7] Juchang Lee, Kihong Kim, Sang K. Cha, "Differential Logging: A Commutative and Associative Logging Scheme for Highly Parallel Main Memory Databases," In *Proceedings of IEEE ICDE Conference*, 2001.
[8] Kihong Kim, Sang K. Cha, Keunjoo Kwon, "Optimizing Multidimensional Index Trees for Main Memory Access," In *Proceedings of ACM SIGMOD Conference*, 2001.
[9] Sang K. Cha, Sangyong Hwang, Kihong Kim, Keunjoo Kwon, "Cache-concurrency Control of Main-Memory Indexes on Shared-Memory Multiprocessor Systems", In *Proceedings of VLDB Conference*, 2001.