네트웍 트랜잭션 처리를 위한 부하 균등 알고리즘 설계 및 구현

이충석 김성후 박규석
경남대학교 컴퓨터공학과

Design and Implementation of a Load Balancing Algorithm for Network Transaction

Chung-Seok Lee, Sung-Ho Kim, Kyoo-Seok Park
Dept. of Computer Science, Kyungnam University

요 약

인터넷 이용자들의 증가로 인해 개방된 네트워크에서의 실시간 분산환경이 고려되고 있으며, 이들 분산된 정보 및 컴퓨터 자원들에 대한 접근 요구가 증대됨에 따라 네트워크 상호연결 요구 또한 커지고 있다. 이러한 정보제공에 대한 이용자들의 요구를 충족시키기 위해 다중 서버를 두고 이들 서버간에 네트워크의 성능을 효율적으로 감시하고 제어하기 위한 모니터링 서버를 제공함으로써 분산 시스템의 성능 향상은 물론 이용자 입장에서의 정보에 대한 응답시간과 반환시간을 최소화하고, 시스템의 전반적인 측면에서의 작업 처리율과 자원의 활용도를 최대화 할 수 있으며, 동적인 상황에 따라 스스로 판단하고 적절하게 대응할 수 있는 지능형 이동 에이전트 템플릿을 설계 및 구현하였다.

1. 서론

인터넷에서 웹을 기반으로 한 다수의 이용자가 필요로 하는 정보에 대한 요구를 충족시키고 원활한 정보 전송을 위한 개방성 및 상호 운용성을 고려한 트랜잭션 처리가 요구되며, 이러한 트랜잭션 처리는 분산된 정보 제공서버로부터 공유 데이터베이스에 접근하여 이용자가 원하는 정보에 대해 효율적으로 처리해주는 모니터링을 기반으로 하는 프로그램의 수행이라 할 수 있다.

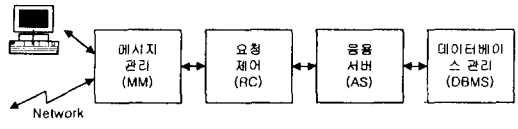
기존의 트랜잭션 처리 기술은 이용자와 정보서버간의 다대일 방식에 기반을 둔 것으로 네트웍 트래픽과 서버의 부하 처리에 부족한 점이 있었다.

본 연구에서는 이러한 부분들에 대해 보완해줄 수 있는 시스템으로 다대다 방식을 기반으로 하는 분산된 환경에서의 다중 정보서버를 구축하고, 복잡하고 다양한 정보통신 서비스를 제공하기 위한 방법으로 다중 정보 제공서버들간의 네트웍 트래픽과 각 정보 제공서버들에 대한 부하에 대한 처리를 모니터링 함으로써 분산된 컴퓨터 시스템의 성능 데이터들을 모니터하고 그 상관 관계에 따른 분석을 기반으로 하여 효율적인 정보를 제공 하려고 한다. 또한 플랫폼의 독립적인 처리를 위해 특정 하드웨어나 OS에 영향을 받지 않는 객체기반 프로그램으로 구성하여 원활하게 인터넷 이용자가 개방된 네트워크에서 실시간으로 정보에 대한 요구를 수행할 수 있도록 실시간 분산 환경을 고려한 미들웨어 적용 방안을 제안하며, 종래의 사용자에 의해 입력된 지시사항이나 동작 스크립트대로만 행함으로써 복잡한 작업을 수행하거나 예기치 못한 예외적 상황이 발생하였을 때 스스로 판단하여 적절한 대응조치를 수행할 수 없었던던 비 지능형 에이전트를 개선한 동적으로 적응할 수 있는 분산 네트웍 서버들간의 지능형 이동 에이전트 템플릿을 제공하고자 한다.

2. 관련연구

2.1 TP Monitor

TP 모니터는 트랜잭션 응용 프로그래머가 트랜잭션의 ACID 특성을 쉽게 구현할 수 있도록 도움을 주는 프로그램으로서 시스템 자원의 할당을 관리하는 인터페이스와 프로시저들로 이루어지며 트랜잭션 스케줄링, 큐 관리, 고장 허용, 보안 및 정상적으로 종료되지 못한 트랜잭션의 회복 기능 등을 지원하며, TP 모니터를 사용하지 않은 시스템에 비해 모니터가 사용된 시스템의 경우 트랜잭션의 처리율이 향상되고, TP 모니터를 사용하는 경우 그렇지 않은 경우에 비해 보다 많은 수의 터미널들을 지원할 수 있으며, 장애율의 감소 및 응용의 관리, 개발비용 등을 경감시켜 준다. 그림 1은 전형적인 TP 모니터 모형으로서 대부분의 TP 모니터가 이러한 모형을 기반으로 만들어졌다.



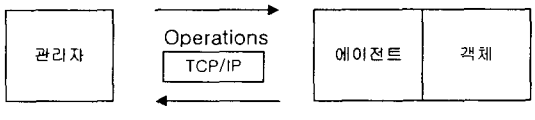
[그림 1] TP 모니터 모형

TP 모니터가 수행해야 하는 역할을 요약하면 다음과 같다.

- 트랜잭션을 동기화 시키고 관리해야 한다.
- 충분한 수준의 수행 능력과 보안을 보장해야 한다.
- 서버들 사이의 작업을 균등하게 분배해야 한다.
- 이종의 DBMS와 상호 연결 기능을 지원해야 한다.

2.2 SNMP

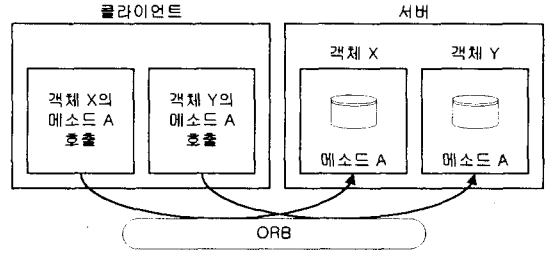
TCP/IP 망에서의 이기종 네트워크 관리를 위한 Internet Protocol인 SNMP(Simple Network Management Protocol)는 안정되어 있고 단순하며, 오버헤드가 적은 것이 장점이다. SNMP는 표준 응용단계 프로토콜이고, 이를 사용한 상업용 소프트웨어도 있다. SNMP는 관리자가 원격 호스트나 게이트웨이에게 네트워크 상태에 대해서 물어보고 원격 호스트 동작을 제어하도록 구성되어지며, 그 구성은 그림 2와 같다.



[그림 2] SNMP 구조

2.3 CORBA Object Model

OMA 코어 객체 모델을 구체화한 것으로 클라이언트는 요구의 대상이 되는 객체를 특별히 지정하여 연산을 호출한다. 본 연구에서는 단순히 원격 메소드를 호출하는 것이 아니라 하나의 목표 객체의 메소드를 호출함으로써 같은 이름의 메소드 호출이 그 요청을 받는 서버 객체에 따라 다른 결과를 가져올 수 있음을 의미하는 CORBA ORB(object bus), 즉 객체를 기본으로 하는 추상화된 방식의 분산 컴퓨팅 방식이다.



[그림 3] ORB의 통신 메커니즘

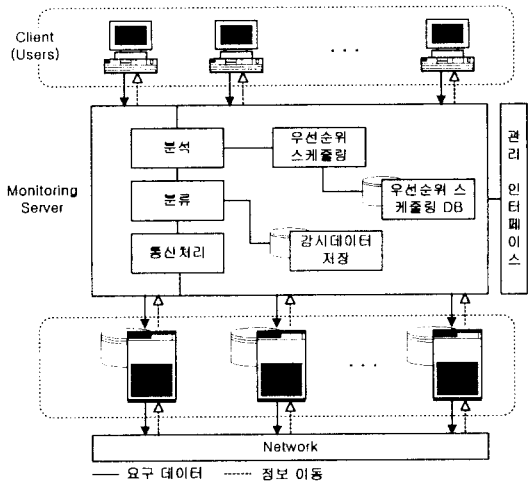
3. 시스템 설계 및 구현

3.1 시스템 설계

시스템 구성은 그림 4와 같이 Web 기반에서의 정보를 요구하는 Client와 정보를 제공하는 Information Server, 이들 관계를 원활하게 처리하기 위한 모듈인 본 연구에서 제안한 Monitoring Server로 구성되어 있다. Monitoring Server는 Web에서의 관리 인터페이스에 의해 Remote로 제어할 수 있으며, 트랜잭션 처리를 위해 실시간으로 분석, 관리하게 된다.

이러한 실시간 트랜잭션은 처리 후에 데이터의 정확성을 유지하기 위한 시간적 제약과 트랜잭션 스케줄링을 고려한 각 정보 제공서버에 대한 우선순위 할당에 대한 스케줄링과 우선순위를 고려한 데이터베이스 동시성 제어 기법을 필요로 한다. 실시간 트랜잭션 처리에 대한 문제는 주로 OS 및 DBMS 관점에서 연구되어 왔으며, 근래에는 분산 네트워크 처리에서 이용되어 지고 있다. 즉, 실시간 제약에 따른 프로세스 스케줄링 및 데이터 접근 경쟁시 동시성 제어 등과 같은 해결 방법 등에 대해 연구되어 지고 있다.

본 연구의 Monitoring Server 모듈에 대한 각 구성은 분산 네트웍에서의 각 정보 제공서버들로부터 실시간 제약을 고려한 우선순위 스케줄링 처리모듈과 공유 데이터베이스에 대한 동시성 제어에 대한 실시간 트랜잭션 처리 성능 향상에 기여할 수 있는 부분을 고려한다.



[그림 4] 부하균등 처리를 위한 분산네트웍 알고리즘

3.2 모듈정의 및 알고리즘 구성

분석, 분류 처리 모듈은 이용자의 요구처리에 대해 우선순위를 결정하기 위한 모듈의 전 단계로 각 정보 제공서버에 대한 상태 처리와 통신처리를 하게된다. 각 서버의 상태에 따라 오류 발견시 자동적으로 오류 수정을 요구하게 된다. 또한 구성된 데이터베이스에 대한 동시성 제어 처리를 하며, 동시성 제어를 위한 알고리즘은 다음과 같다.

```
//ID...working signal
JobID_M.SetJobID($Target_ID,this.getName());
//Sorting Schedule
if($Job_Code!=5&&$Job_Code!=6) {
  this.setPriority(5-$Job_Code); }
yield();
int count=1;
//Over ID...working check
while(JobID_M.ChkJobID($Target_ID,this.getName())) {
  //접속한 서브모듈에 대기신호 전송
  //System.out.println('\t'+this.getName()+"->Waitting...");
  if(count==1) OUT.println("WAIT");
  count++; }
OUT.println("ACK");
switch($Job_Code) {
  case 0:{  //User delete
```

우선순위 스케줄링 처리 모듈은 각 정보 제공서버에 대한 네트웍 트래픽과 서버들에 대한 부하률에 따른 것으로 오름차순으로 정렬되며, 관리 인터페이스에 의해 정해진 시간 또는 예약된 시간에 의해 분류별 이동 서버에 대한 정보 데이터베이스를 구축하게 되며, 이에 대한 데이터베이스 구성과 스케줄링 알고리즘은 다음과 같다.

```
Sorting_Schedule(int Time) {
  //Server List Store Class
  Web_Server List;
  //Current time run... Server list Store
  Get_Server_List(List);
  //시간대별 각 서버의 Network traffic Store
  Get_Ping(List, Time);
  Get_Users(List, Time);
  //네트웍 트래픽과 부하률을 가지고 오름차순 정렬
  Sort_Schedule(List, Time);
  //이용자 요구 처리를 위한 우선순위테이블 생성
  List = Create_Table(List); }
```

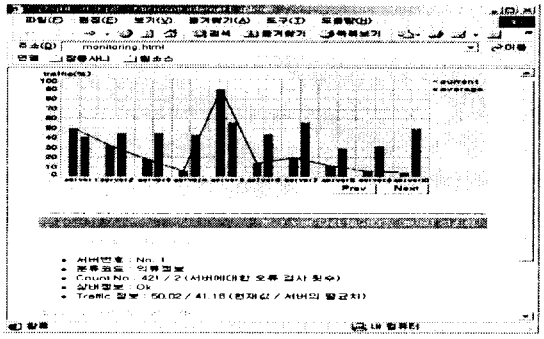
<표 1> 우선순위스케줄러의 데이터베이스 구성

서버번호(SevNo)	각 정보 제공서버에 대한 고유번호
분류코드(SCode)	각 정보 제공서버가 가지는 항목 분류 코드
URL	각 정보 제공서버의 URL No.
Count	각 정보 제공서버의 분석 결과에 따른 횟수
상태(State)	각 정보 제공서버의 상태

3.3 인터페이스 구현

본 연구에서 각 정보 제공서버들에 대한 상태를 분석하기 위한 실시간 모듈 인터페이스는 다음과 같으며, 그것은 각 정보 제공서버에 대한 현재 Traffic 비율과 현재까지의 Traffic에 대한 평균치로 구성된다.

또한 10개 이상의 서버가 등록된 경우 10개의 서버 단위로 나타나게 되며, 각 서버에 대한 정확한 상태정보를 문서 형식으로 확인 할 수 있으며, 그림 5는 분산 서버들에 대한 모니터링 결과를 나타내고 있다.



[그림 5] 분산 서버들에 대한 모니터링 결과

4. 결론

본 논문에서는 분산된 네트웨크 환경에서 다중 서버에 대한 트랜잭션 처리 및 트랜잭션 처리 모니터에 대해 고찰 하였으며, TP 모니터 기반으로 한 분산 서버간의 트랙픽과 서버 부하에 대한 스케줄링으로부터 실시간 트랜잭션 처리 지원 기능의 일부분으로서 우선순위 처리에 대한 기능과 관리 기능을 포함하였다. 각 정보 제공서버로부터 구성된 데이터베이스에 대한 일관성 유지를 위한 동시성 제어를 고려하여 TP 모니터를 사용하는 분산 서버간의 트랜잭션 처리 시스템의 성능 향상 효과를 보였다.

향후 연구방향으로는 분산 트랜잭션 처리 기능 및 회복 기능 및 TP 모니터 인터페이스 기능 보강과 Java Security 등에 대한 지속적인 연구가 필요하다.

[참고문헌]

[1] J.Davin, "A Simple Network management Protocol(SNMP)," Networking Group Request For Comments 1157 (May 1990).

[2] K.McCloghric, M.Rose, "Structure and Identification of Management Information for TCP/IP based Internals," Networking Group Request For Comments 1155 (May 1990).

[3] K.Harrenstien, E.Feinler, "DoD Internet host table Specification," Networking Group Request For Comments 952 (October 1985).

[4] Clement H. C. Leung, "Quantitative Analysis of Computer Systems," Jhon Wiley & Sons, 1988.

[5] Raj Jain, The Art of Computer systems Performance Analysis:Techniques for Experimental Design, Measurement, Simulation, and Modeling," John Wiley & Sons, 1991.

[6] Johnson J. H. and Hudson D. M., "Open Systems OLTP Monitors," UniForum 1992 Conference Proceeding, pp. 157-169.

[7] Abbott, R. and H. Garcia-Molina, "Scheduling Real-Time Transactions : a Performance Evaluation," Proc. of the 16th Int'l Conf. on Very Large Data Bases, Los Angeles, Aug., 1988.

[8] Gray, J., "The Benchma가 Handbook: for Database and Transaction Processing systems, Morgan Kaufmann publishers, Inc., 1991.

[9] Philp A. Berstein, "Transaction Processing Monitors," CACM, Vol. 33, No. 11, Nov. 1990, pp.75-86.