TP 모니터를 이용한 계정계 업무 분산 시스템 구축

이 성 주[†] · 최 완 규[††] · 나 영 남[†††]

요 약

DBMS을 이용한 온라인 트랜잭션 처리 시스템에 비해 TP 모니터를 부가하여 사용할 경우 시스템의 성능이 증가한다. 아울러 이러한 TP 모니터를 이용한 OLTP 시스템 구축이 늘어가고 있는 실정이다. TP 모니터를 이용할 때 고려해야 할 점들이 많으나 기본적으로 트랜잭션의 ACID외에 빠른 응답 시간과 확장성에 초점을 맞춰 TP 모니터를 적용하여 분산 시스템을 구현 하였다. 본 논문에서는 분산 시스템에 의한 다운 사이징을 구현하고 그 성능을 검증한다.

Implementation of a Distributed System for Banking Accounting with the TP monitor

Sung-Joo Lee[†] · Wan-Gyu Choi[††] · Young-Nam Na[†††]

ABSTRACT

We can get very enhanced performance in case of adding the TP monitor to the on_line transaction processing system rather than just using the DBMS. Futhermore, It is getting boomed to establish a OLTP system with the TP monitor. We designed and implemented a distributed processing system with the TP monitor focused quick response time and expansionality and basic the ACID of transaction. In this paper, We explain the implementation about downsizing and upon distributed system and verified of performance.

1. 서 론

트랜잭션 처리(Transaction Processing:TP)는 조직과 산업체 전반에 걸쳐서 업무활동의 중심을 이루는 정보의 관리로, 오늘날 TP 기술은 분산 시스템과 클라이언트/서버 구조에서 OLTP(On_Line Transaction Processing)를 수행하는 경향 때문에 더욱 그 중요성이 요구된다.

※본 논문은 1994년도 조선대학교 학술연구비에 의해 연구되었음.

† 정 회 원:조선대학교 전자계산학과
†† 준 회 원:조선대학교 전자계산학과
††† 정 회 원:조선대학교 전산통계학과

논문접수:1997년 6월 19일, 심사완료:1997년 9월 14일

통신 네트웍을 이용한 많은 컴퓨터의 연결은 사용자들에게 유용한 분산 자원을 접근하고 공유할 수 있게 하고, 작업량을 효율적으로 분산 시킬 수 있다.

단일의 메인 프레임에서 여러 대의 분산된 호스트로의 분산 시스템 구축을 고려할 때 은행업무나 항공권 예약 등 빠른 응답 시간과 많은 단말기를 보유한 환경에서는 제한된 시간내에 많은 양의 트랜잭션을 처리하기 위해 필요하다.

TP 모니터는 기본적으로 트랜잭션의 ACID(Atomicity, Consistency, Isolation, Durability)를 제공 함으로써 자료의 분산으로 인하여 발생할 수 있는 트랜잭션의 투명성 문제를 해결할 수 있다.[18]

또 정형화된 트랜잭션 포맷과 사용하기 쉬운 사용자

인터페이스의 제공, 이름 서버에 의한 응용 서버의 위치 투명성 제공으로 응용 프로그램의 작성을 쉽게 한다.

프로세스와 단말기의 관리, 프로세스간 부하의 분산, 장애의 극복, 주어진 시간내에 트랜잭션의 완료등 전체 시스템에 대한 통합적인 관리가 용이 하도록 한다.

본 논문은 TP 모니터를 이용하여 온라인 계정계 업무를 분산 시스템으로 구축하여 다운 사이징에 의한 시스템의 성능 향상에 있다. 2장은 분산 시스템과 TP 모니터에 대한 고찰, 3장은 은행의 분산 시스템 설계 및 구축, 4장은 구현한 시스템의 성능평가 및 분석, 5장은 본 연구를 통하여 얻어진 결론 및 향후 연구 방향을 서술한다.

2. 분산 시스템과 TP 모니터

2.1 분산 시스템(Distributed system)

분산 시스템은 응용 처리를 지원하는 초보적인 호스트 기반 처리(Host-base) 환경으로부터 발전하여 처리 분배가 주 컴퓨터에 연결되어 주 컴퓨터가 지시하는 관련기능을 수행하는 주-종속 처리(Master-slave)를 거쳐, 공유 기기를 자연스럽게 확장한 개념으로 요청과 처리가 클라이언트와 서버에 분배되는 클라이언트/서버 처리, 참여한 모든 시스템이 대등하여 서로에게 서비스를 요청하고, 제공하는 동배간 처리(peer to peer)로 발전 하였다.

단일의 중앙 집중화된 시스템 내에서 모든 업무를 수행하는 대신 지역별 기능별로 자료를 중심으로 여러 대의 소형 시스템에 분산하여 처리하는 방식이다.

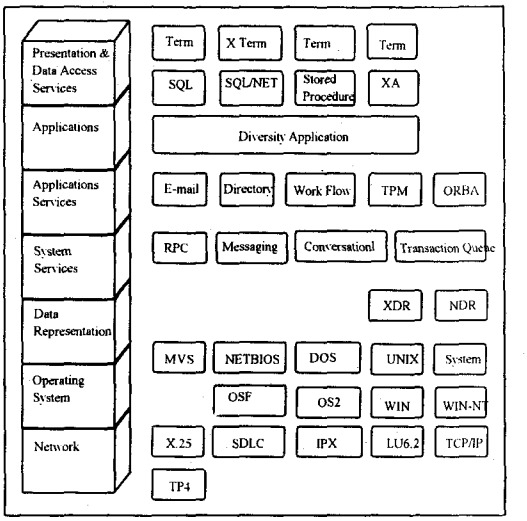
이는 네트웍을 이용한 계산(Network computing)을 근간으로, 전체 시스템의 동시성(Concurrent processing)을 높이고 가용성(Availability), 처리율(Performance) 그리고 투자 대 효과를 증대 시키며 반무장애 시스템(Near Fault-Tolerance)을 구성할 수 있다[17].

2.2 개방 시스템(Open system)

상호 운용성과 응용 이식 요구 사항을 충족 시키기 위한 인터페이스, 서비스, 그리고 지원 형식을 명시하는 국제 정보 기술 표준안과 기능적인 표준안 윤곽의 포괄적이고 일관성 있는 집합으로,

- 시스템간의 응용 이식성
- 응용의 성능과 처리율의 확장성
- 시스템간의 상호 운용성의

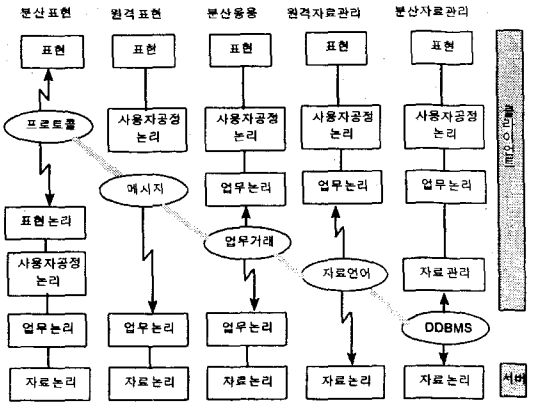
특징을 가지며, 개방 시스템의 구성 요소는 (그림 1)과 같다.



(그림 1) 개방 시스템의 구성요소
(Fig. 1) Components of open system

2.3 클라이언트/서버(Client/Server)

클라이언트/서버 모델은 공유기기 처리를 확장한 개념으로 여러 모습들을 포괄하며, 이와 관련된 문제



(그림 2) 클라이언트/서버의 기능 분산
(Fig. 2) Functional distribution of client/server

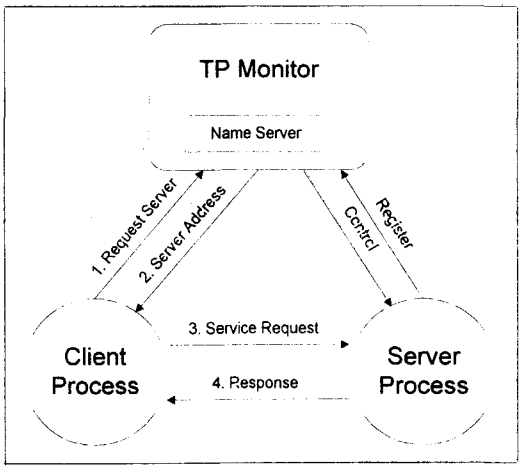
는 통신망, 분산 데이터, 분산 처리, 분산 트랜잭션 처리와 관리 등 이다.

클라이언트의 성능 향상과 더불어 기존 중앙 서버에서 처리하던 많은 처리 로직을 대폭 클라이언트에 이행하여 처리하는 방식, 즉 응용 서버는 자료와 관련된 처리 로직과 자료 접근만을 담당하고 입출력 자료의 검증 및 처리, 표현 클라이언트가 하도록 하는 방식으로 클라이언트/서버의 기능 분산은 (그림 2)와 같다.

2.4 TP 모니터

TP 모니터는 대형의 DTP(Distributed Transaction Processing) 시스템을 구축할 때 단순히 DBMS만을 이용하는 대신 분산된 시스템간의 트랜잭션을 모니터하여 전체적인 트랜잭션의 처리율을 향상 시키기 위한 미들웨어다[7].

TP 모니터를 사용 함으로써 첫째, 응용프로그래머는 트랜잭션의 ACID를 쉽게 구현할 수 있고 트랜잭션의 스케쥴링, 작업 큐의 관리, 고장 허용, 보안, 비정상 종료된 트랜잭션의 회복에 관한 우려에서 벗어날 수 있다. 둘째, 단말기의 수를 증가시킬 수 있으며, 응답 시간도 DBMS을 사용한 시스템보다 응용 프로그램의 차이에 따라 달라질 수는 있으나 현격하게 빨라진다[4, 5, 6, 7].



(그림 3) TP 모니터의 기능과 거래 흐름
(Fig. 3) Function and transaction flow of TP monitor

대형 분산 시스템을 구축할 때 TP 모니터는 트랜잭션 처리 성능을 향상 시키며, 웍스테이션 수준의 소형 시스템에서는 오히려 TP 모니터의 트랜잭션 모니터링 기능에 의해 과부하를 유발할 위험이 있다[18].

(그림 3)은 클라이언트/서버 환경에서 TP 모니터의 역할과 트랜잭션의 흐름을 도식화한 것으로, 서버 프로세스는 기동시 TP 모니터에 자신의 이름, 주소, 자격, 보안에 관한 정보를 등록한다.

등록된 서버에 대한 주소를 클라이언트는 서버의 이름을 가지고 TP 모니터에 요청하며, 등록된 서버에 대해 TP 모니터는 트랜잭션 처리에 필요한 적절한 제어를 할 수 있다. 클라이언트는 서버의 주소를 가지고 해당 서버에게 서비스를 요청하여 응답을 받음으로써 트랜잭션은 종료된다.

이러한 TP 모니터를 사용 함으로써 얻게되는 장점으로는 1)여러 대의 단말기 사용 가능 2)트랜잭션의 단위 처리량 증가 3)장애율 감소 4)응용 관리비 및 개발비 절감 등이다[18].

2.4.1 ACID

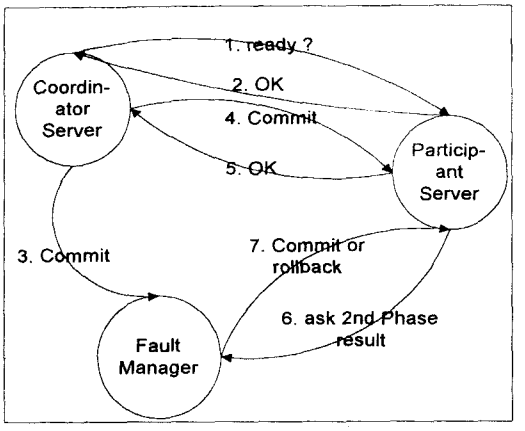
TP 모니터는 트랜잭션이 갖는 다음의 ACID 특징을 만족할 수 있는 지원 기능을 제공 한다.

- ⊙ 원자성(Atomicity): 트랜잭션의 수행 결과가 데이터베이스에 부분적으로 반영 되어서는 안됨 (All or Nothing property).
- ⊙ 일관성(Consistency): 트랜잭션 수행 후에도 데이터베이스는 계속 일관성을 유지 해야 함.
- ⊙ 불간섭(Isolation): 복수의 트랜잭션이 동시에 수행시 상호 간섭하지 않아야 함. 즉, 불완전한 트랜잭션의 어느 부분이 다른 트랜잭션에 영향을 주어서는 안됨.
- ⊙ 지속성(Durability): 트랜잭션의 정상적인 종료 후, 어떤 장애의 발생 시에도 정상적 결과가 지속 되어야 함.

2.4.2 Two Phase Commit(2PC)

클라이언트/서버가 각각의 자원을 성공적으로 수정할 때 트랜잭션 자원에 대한 변경을 완료하고 원자성과 무결성을 보장 받는다.

ACID중 원자성과 무결성의 보장을 위해 TP 모니터

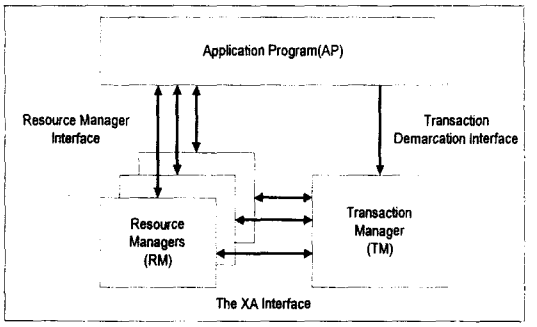


(그림 4) 2 Phase commit
(Fig. 4) 2 Phase commit

는 (그림 4)와 같이 2 Phase Commit 방법을 사용한다.

2.4.3 X/OPEN DTP Model

트랜잭션 처리 응용 프로그램의 호환성을 위한 X/OPEN DTP 모델[12]은 하드웨어와 소프트웨어에 무관하게 응용 프로그램을 작성할 수 있도록 하며, 기능적 구조는 (그림 5)와 같다.



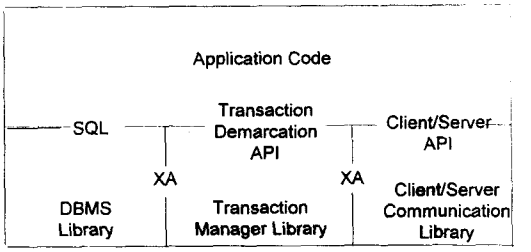
(그림 5) X/Open 구조
(Fig. 5) X/Open architecture

⊙ Application Program(AP): 최종 사용자의 요구를 처리하는 응용 로직으로서 자원 관리자(RM)와 트랜잭션 관리자(TM)와의 인터페이스를 담당한다.

⊙ Transaction Manager(TM): AP에 의하여 정의된 트랜잭션과 자원 관리자 사이에서 트랜잭션의 완료를 관리한다. 이 요소는 시스템에서 트랜잭션의 고유 이름을 할당하고 자원 관리자와 함 2PC(2 Phase Commit)를 지원하며 장애 후의 복구를 관장한다.
TM은 AP에게 트랜잭션의 경계인 트랜잭션의 시작(begin), 완료(commit) 그리고 원위치(roll-back)를 정의한다.

⊙ Resource Manager(RM): 응용 프로그램의 특정한 공유 자원을 관리하여 트랜잭션이 필요로 하는 여러 종류의 자원을 AP가 다수의 자원 관리자를 직접 호출 하게한다.

(그림 5)의 X/Open 구조를 구성하는 요소들 간의 관계를 블럭화하면 (그림 6)과 같다.



(그림 6) X/Open의 표준 인터페이스
(Fig. 6) Standard interface of X/Open

(1) DTP 기능

분산 트랜잭션 처리 시스템(Distributed Transaction Processing System)은 다음과 같은 기능을 제공하여 시스템의 가용성과 성능 및 호환성의 증대를 도모한다.

① 벤더로부터의 독립

TP 모니터는 다양한 하드웨어 및 소프트웨어 제품으로부터 독립적 이어야 한다.

하드웨어 부분은 특정한 하드웨어 제품으로 부터의 종속을 탈피하기 위해 오픈 아키텍처를 지원하는 유닉스 운영체제 시스템이 주로 사용되나, 소프트웨어의 경우 TP 모니터의 부분 부분이 특성상 번들(bundle)로 제공되기 때문에 이를 해결하기 위하여 XA 인터페이스를 지원 하도록 하여 응용 프로그램 작성자들이 하드웨어나 특정 TP 모니터의 종속에서 탈피할 수 있다.

②분산의 투명성

시스템의 분산 규모가 커질수록 위치 투명성의 중요성이 증대 하므로, OLTP 시스템에서 통신 관리자는 응용 프로그래머에게 반드시 위치의 투명성을 제공 하여야 한다. 이를 위하여 트랜잭션에 근거한 라우팅 기능이 제공되어, 응용 프래그머는 데이타의 위치에 무관하게 TP 모니터는 트랜잭션의 내용에 의해 해당 자료가 위치한 시스템으로의 트랜잭션 전달을 보장한다.

③성능과 확장성

OLTP 응용 프로그램은 높은 처리율과 짧은 응답 시간을 요구한다. 이를 위하여 OLTP 시스템은 응용 프로그램들이 각기 병렬적으로 수행될 수 있도록 모듈화를 제공한다. 동시에 응용 프로그램들이 첨가되는 시스템에서 탄력적으로 가동될 수 있도록 확장성도 제공한다.

④연결성과 상호 작용성

다양한 전송 인터페이스를 갖는 네트워크(XTI, Socket, Netbios 등)와 프로토콜(TCP/IP, X.25 등)의 지원으로 응용 프로그래머의 요구에 부응하는 다양한 연결성을 제공한다. 이러한 API 인터페이스는 응용 프로그램에 반드시 독립적으로 지원되어 다른 TPM (Transaction Processing Monitor)을 사용하는 시스템과의 상호작용성을 증대한다.

⑤신뢰성

장애 후에도 장애 이전의 결과를 지속적으로 유지하고, 트랜잭션이 병렬적으로 수행될 때, 어느 한 시스템에서 장애가 발생 하더라도 트랜잭션의 결과는 원자성(Atomicity)이 보장되는 신뢰성을 제공한다.

⑥보안성

부여된 권한을 갖는 자가 부여된 서비스만을 받을 수 있도록 응용 프로그램에게 보안성을 지원한다.

⑦감시(Monitoring)와 관리(Administration)

시스템이 분산되어 있으므로 한 곳에서 전체 분산된 시스템을 단일 시스템 처럼 감시하여 시스템의 부하를 고려한 작업순위 산정, 지정된 시간에 응용 프로그램의 수행 등의 기능을 제공하여 시스템 관리자의 작업을 손쉽도록 지원한다.

3. 분산 온라인 시스템 구축

3.1 시스템 도입 배경 및 목표

분산 온라인 시스템 구축 이전의 시스템은 NCR 9800XL-2004에 TOX/CS 운영체제 시스템으로, 시스템의 노후화로 시스템의 처리 능력이 한계에 이르렀고 응용 소프트웨어의 기본 구조의 수정이 요구 되었다. 아울러 사용자의 복잡. 다양한 요구에 부응하기 위한 정보 기술에 대한 시스템의 한계 등 도입 배경 요인은 〈표 1〉과 같다.

시스템 구축의 기본 목표인

- 경제적인 시스템
- 유연성 및 확장성을 고려한 시스템
- 신뢰성있는 시스템
- 국제 지향의 표준화된 시스템의 설정과 기술적인 측면에서는
- 지역 분산시스템 : 단일의 메인 호스트에서 모든 업무를 처리하는 대신 지역적으로 호스트를 두어 분산 처리가 가능한 시스템
- 오픈 시스템 : 국제 표준화된 정보 기술을 손쉽게 접목할 수 있는 표준, 개방화된 시스템

〈표 1〉 분산 온라인 시스템 도입 배경

〈Table 1〉 Background of distributed online system implementation

구 분	내 용
중앙집중식 문제점	· 전산 투자 비용 과다 · 공급 업체에 예속 · S/W 자산화의 어려움 · 사용자 요구 S/W 개발 부진(전산 back-log 증대) · 환경 변화에 대한 유연성 부족 · 많은 전산 인력의 요구 · 장애발생 지역의 광범위
정보기술의 발전	· H/W 성능대비 가격의 하락 · 개방형 시스템 구현 · 사용자 컴퓨팅 가능 · 통신 기술의 발달 · 다양한 사용자 접속 기술의 발달 · 클라이언트/서버 구조의 유효 · 분산 정보 기술의 성숙

· 신 정보기술 사용:클라이언트/서버, 객체지향 등 신 정보기술의 사상을 손쉽게 응용할 수 있는 시스템을 기본 목표로 한다.

3.2 시스템 개요

분산 시스템 구축을 위한 온라인 시스템은 7대의 유닉스 호스트(HP70, 128 MIPS, 256MB)를 4개의 지역에 분산시켜 다수의 응용 프로그램이 가동된다. 부하에 따라 응용 서버의 수를 조절할 수 있으며 온라인 시작과 함께 일정량의 응용 프로그램이 항상 대기 상태로 가동되어 이벤트에 따른 기동/처리 방법이 지니고 있는 Start up 오버 헤드를 피하게 한다.

네트웍은 TCP/IP(3개지역:56Kbps Dual, 2개지역: T1 및 56Kbps)를 사용하며 부분적으로 X.25를 이용한다.

클라이언트 단말기는 약 1000여대 이며 네트웍 부하와 트랜잭션 처리의 분산 및 온라인 로그를 저장하기 위해 영업점에 웍스테이션(32M, 1.3G Disk)을 설치한다.

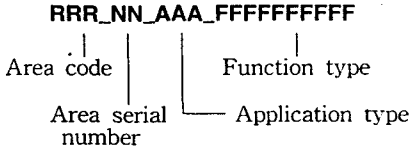
텔러 단말용 클라이언트 PC(33MHz, 4M, 80M Disk)는 MS-Window에서 약 400본의 DLL(Dynamic Link Library) 프로그램이 GUI 형태의 프리젠테이션과 입력 자료의 초기 검증 및 서버로의 서비스 요청과 응답 결과의 출력을 담당 하도록 한다.

응용 프로그램은 C++ 및 C 언어로 작성하여 시스템 자원의 이용 및 호환성과 소프트웨어 재사용의 문제를 해결 하도록 한다.

3.3 응용서버의 이름 제정 및 서버 분리

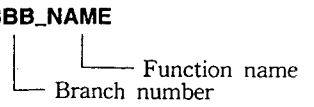
클라이언트는 서버의 서비스를 받을 때 서버의 이름을 가지고 찾아가기 때문에 서버의 이름은 TP 모니터를 사용하는 환경에서는 매우 중요하다. 편의/관리의 효율을 위한 이름 제정 규칙(Naming Rule)은 다음과 같이 한다.

①지역 서버명



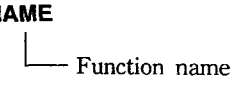
ex) **KWJ_01_CK _UPDATE**
Area code : **KWJ**(Kwangju)
Area serial number : **01**
Application type : **CK**(Checking)
Function type : **UPDATE**(Update)

②영업점 서버명



ex) **001_BCS_CONTROL**
Branch number : **001**
Function name : **BCS_CONTROL**

③특수목적 서버명



ex) **CHANGE_DATE**(Change Date)

서버 이름 제정 규칙과 아울러 조회만을 담당하는 응용 서버와 데이터 베이스를 갱신하는 응용 서버로 분리하여 서버의 관리와 시스템의 성능 향상을 추구한다.

3.4 요청 및 응답 포맷의 정형화

프로세스간 트랜잭션의 전달은 금융 환경의 특수성을 참작하여 포맷을 정형화 함으로써 유지/보수를 용이하게 하도록 한다.

클라이언트로 서비스를 요청할 때는 TITA(Transaction Input Text Area) 포맷을, 응용 서버간의 글로벌 트랜잭션의 이동에는 MACTA(Multiple Account Text Area) 포맷을, 트랜잭션의 로그를 작성하기 위한 TXLOG(Transaction LOG area), 트랜잭션간 공통부분을 정의하기 위한 TXCOM(Transaction COMmon area), 응용서버가 클라이언트로 서비스 응답을 보내기 위한 TOTA(Transaction Output Text Area)를 사용하여 트랜잭션을 발생시키고 받을 수 있도록 한 각종 인터페이스 포맷은 다음과 같다.

TITA(2048B)

HEADER (41B)	BASIC LABEL (151B)	TEXT (1856B)

TOTA(490B)

BASIC LABEL (68B)	TEXT (422B)

MACTA(2049B)

REGION -ID(8B)	SERVER NAME(30B)	LENGTH (4B)	LABEL (151B)	TEXT (1856B)

TXLOG(532B)

BASIC LABEL (28B)	TEXT (504B)

TXCOM(310B)

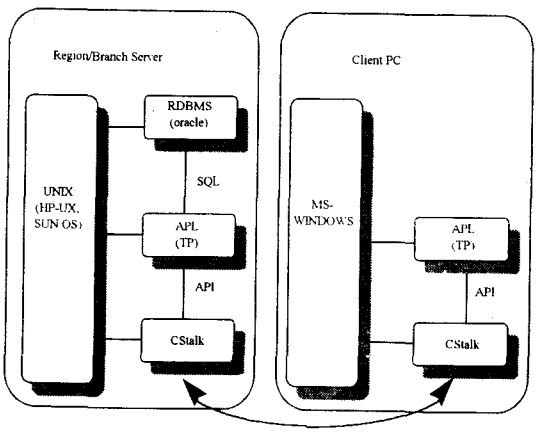
TEXT(310B)

3.5 하드웨어와 소프트웨어적 구조

하드웨어적 구조는 7개의 지역 서버와 본부 지역에 응용 업무별 기능 서버를 두며, 영업점은 영업점 서버가 있는 관리 모점(母店)과 서버가 없는 출장소로 구분한다.

각 지역의 호스트는 복수 개를 두어 주 호스트와 장애시 신속한 백업 호스트로의 전환을 가능케 한다.

소프트웨어적 구조는 클라이언트에 업무의 많은 부분을 이관 함으로써 응용 서버의 부하를 경감 시키는 것을 원칙으로 한다. 즉, 클라이언트는 GUI 화면 처리, 입력된 자료의 검증, 계산 기능, 각종 출력 장치의 관리를 담당하는 기능만을 부여하여 서버의 부하를 경감하는 방법으로 (그림 7)과 같다.



(그림 7) 소프트웨어 구조
(Fig. 7) Software architecture

응용 프로그램은 TP 모니터를 통하여 클라이언트로부터 요청된 서비스를 받고 이를 SQL을 이용하여 데이터 베이스 저장 하거나 적절한 처리를 하게 한다. 처리 결과 또한 TP 모니터를 통하여 클라이언트 PC에 전달되며, 클라이언트에 있는 TP 모니터는 이를 클라이언트 APL(Application Program Library)에 전달하고 클라이언트 APL은 이를 가공하여 MS-Windows에서 제공하는 각종 기능을 호출하여 GUI 형태로 사용자에게 제공하게 한다.

TP 모니터는 서버와 클라이언트 사이에 트랜잭션이 이동하는 동안에도 지속적인 감시를 통하여 예기치 못한 장애나 돌발 사태에도 트랜잭션의 투명성을 보장하기 위한 조치를 취하게 한다.

4. 시스템 성능 평가 및 분석

4.1 TPC 벤치마크 테스트

성능 측정 방법으로는 온라인 트랜잭션처리 시스템의 성능 측정에 사용되고 있는 TPC 벤치마크 프로그램을 사용한다.

TPC-A(Transaction Processing Council-A)는 은행을 가상하여, 한 개 이상의 지점을, 각 지점은 여러 대의 현금 자동 인출기와 계좌를 보유 하면서 많은 고객들이 유일한 계좌를 가지고 있는 응용 환경을 가정한다. TPC-A에서 사용하는 데이터 베이스는 은행, 단말기, 계좌, 거래내역으로 구성된다.

TPC-B는 TPC-A와 동일하나 10대의 터미날을 사용하는 대신 배치 트랜잭션 생성 프로그램을 터미날 대신 이용하는 것이 다르다.

실제적으로 10대의 터미날을 이용하여 단말기의 사고(think)하는 시간을 부여하는 것은 시스템의 성능 측정과 무관하다. 성능 측정의 편의를 위해 본 논문에서는 LAN 환경을 포함하는 TPC-B Local을 선택 하였으며, TPC-B 성능 측정에 사용된 서버 프로세스의 처리 알고리즘은 다음과 같다.

Read 100 Bytes including Aid, Tid, Bid, Delta from terminal
BEGIN TRANACTION
Update Account where Account_ID = Aid:
Read Account_Balance from Account

```
    Set Account_Balance = Account_Balance +
                          Delta
    Write Account BaLANCE TO aCCOUNT
  Write to History:
    Aid, Tid, Bid, Delta, Time_stamp
  Update Teller where Teller_ID = Tid;
    Set Teller_Balance = Teller_Balance + Delta
    Write Teller_Balance to Teller
  Update Branch where Branch_ID = Bid:
      Set Branch_Balance = Branch_Balance +
                           Delta
      Write Branch_Balance to Branch
  COMMIT TRANSACTION
Write 200 bytes including Aid, Tid, Bid, Delta,
Account_Balance to terminal
```

Aid: Account ID, Tid: Teller ID, Bid: Branch ID

위와 같은 기능을 수행하는 서버 프로세스는 오라클 SQL*PLUS 및 PROC를 이용하여 작성 하였으며, 〈표 2〉와 같은 TPC-A 1tps 환경에서 10개의 트랜잭션 생성 배치 프로그램이 10초당 1개의 트랜잭션을 15분 동안 발생하여 처리된 트랜잭션 수, 평균 및 최대 응답시간, 처리율(Transactions Per Second)을 사용한다.

〈표 2〉 TPC-A의 기본 크기
〈Table 2〉 Basic size of TPC-A

Account 테이블	100,000레코드
Teller 테이블	10레코드
Branch 테이블	1레코드
History 테이블	충분한 공간
트랜잭션 생성 프로세스	10개

성능 측정은 1tps와 2tps 환경에서 실시 하였으며 2tps는 〈표 2〉의 모든 항목의 값을 두배 하여 실시했다.

〈표 3〉과 같이 TP 모니터를 사용하지 않을 경우, 평균 응답 시간이 0.983, 최대 응답 시간이 4.317으로, 응답 시간의 폭이 매우 큰 반면, TP 모니터와 DBMS를 함께 사용하였을 경우는 최대 응답 시간을 나타낸 트랜잭션도 평균 응답 시간에서 크게 벗어나지 않으므로 DBMS만을 사용한 시스템에 비해 빠르고 고른 응답 시간을 보임을 알 수 있다.

〈표 3〉 TP 모니터의 시험 결과
〈Table 3〉 The result of TP monitor testing

환경	시스템	처리된 트랜잭션수	TPS	평균 응답 시간	최대 응답 시간
1tps	DB Only	1032	1.147	0.983	4.317
	DB +TP	1404	1.560	0.238	0.837
2tps	DB Only	1507	1.669	1.352	8.739
	DB +TP	2410	2.677	0.570	1.035

환경	시스템	1초	2초	3초	3초이상
1tps	DB Only	993	35	3	1
	DB +TP	1403	0	0	0
2tps	DB Only	637	820	32	18
	DB +TP	2337	73	0	0

4.2 분산 시스템 구축에 의한 거래량 및 응답 시간 분석

일정기간 동안의 온라인 Audit Log에서 추출한 거래량과 응답 시간은 〈표 4〉와 같다.

〈표 4〉 응답 시간별 거래량과 백분율
〈Table 4〉 Transaction's number and percentage in response time

응 답 시 간	트랜잭션 수	백분율(%)
1초 미만	306,602	72.515
1초-2초미만	96,805	22.895
2초-3초미만	10,654	2.520
3초-4초미만	4,288	1.014
4초-5초미만	1,934	0.457
5초 이상	51,283	10.875
총 계	471,566	100.000

〈표 4〉와 같이 트랜잭션의 90% 이상이 5초 이내에 처리됨을 알 수 있으며, 5초 이상이 소요되는 트랜잭션은 온라인 마감후 처리되는 뱃치형 트랜잭션이다.

이들은 메모리의 증설이나 응용 서버의 컴펙트화, 데이터의 효율적인 분산으로 지역간의 트랜잭션을 줄임으로서 해결할 수 있다.

4.3 지역 서버별 트랜잭션 종류의 분석

지역 서버별 트랜잭션 종류의 분석 결과는 〈표 5〉와 같다. 〈표 5〉와 같이 크라이언트로 부터의 트랜잭션이 전체의 40% 이상을 차지하고 있으며, 두 개 이상의 응용 서버를 거치면서 시스템 자원을 과다 사용하는 2PC 트랜잭션이 28%임을 알 수 있다.

이들은 지역적으로 분산된 시스템의 연동 트랜잭션만 2PC를 적용하고 동일 시스템내의 트랜잭션은 UNIX 운영체제의 공유 라이브러리 개념을 이용하여 트랜잭션의 감소와 응답 속도의 증대, 메모리의 절약, 서버의 가동율을 향상시킬 수 있다.

〈표 5〉 지역 서버에서의 거래 종류
〈Table 5〉 Kinds of transaction in branch server (%)

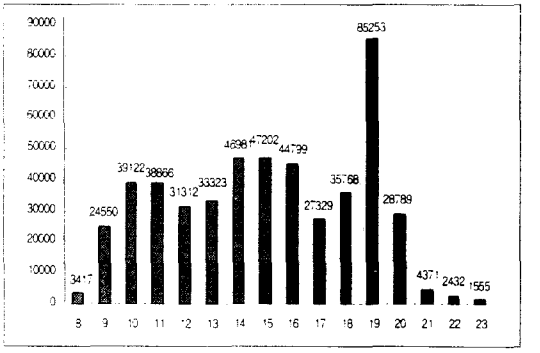
지 역	건 수	단 말	Batch	CD/ATM	2PC
지역1	60,260 (13.32)	24,115 (40.02)	5,927 (9.84)	1,436 (2.23)	28,872 (47.91)
지역2	89,304 (19.73)	37,146 (41.60)	19,454 (21.87)	8,432 (9.44)	24,272 (27.18)
지역3	81,533 (18.02)	36,597 (44.88)	14,146 (17.35)	7,460 (9.15)	23,350 (28.63)
지역4	96,565 (21.34)	43,233 (44.77)	16,845 (17.44)	9,450 (9.79)	27,037 (28.00)
지역5	37,437 (8.27)	14,883 (39.75)	11,427 (30.52)	1,294 (3.46)	9,833 (26.27)
지역6	44,533 (9.85)	19,235 (43.17)	12,271 (27.54)	1,961 (4.40)	11,086 (24.88)
지역7	42,870 (9.47)	8,352 (19.48)	27,540 (64.24)	444 (1.04)	6,534 (15.24)
총 계	452,542 (100)	183,561 (40.56)	107,610 (23.78)	30,387 (6.71)	130,984 (28.94)

4.4 시간대 별 트랜잭션의 분석

다음은 하루중 매 시간마다의 거래량을 나타낸 그래프이다.

(그림 8)과 같이 하루 중 시간대 별 트랜잭션 량은 오전 10시와 오후 2, 3시대에 트랜잭션의 최대치를, 12시대는 중간치를 나타낸다. 오후 7시대의 거래량은 공공요금, 센터 컷 등의 본부에서 투입하는 뱃치성 거래로, 단말기의 거래와는 성질이 다르다.

영업 시간 중에 처리 해야하는 뱃치성 트랜잭션은 시스템의 부하가 적은 시간대를 선택해야 함을 알 수 있다.



(그림 8) 시간대 별 거래량
(Fig. 8) Amount of transaction in time

4.5 메인 프레임과 분산 시스템의 비교

하드웨어의 성능 등 전체 시스템의 성능에 영향을 미치는 요소들이 많아서 동일 조건을 만들 수 없기 때문에 정확한 성능 측정은 어렵다. 그러나 TPC (Transaction Processing Council) 벤치마크 테스트 등

〈표 6〉 메인 프레임과 분산 시스템 비교
〈Table 6〉 Comparison of main-frame and distributed system

구 분	메인 프레임	분산 시스템
특 징	· 거래량 증가에 따른 확장의 제약 · 장애시 시스템 전체 기능 중단 · 순간 과부하시 집중 부하 부담 · 시스템 확장시 교체 방식 · 개발 난이도가 낮음 · 운영 난이도가 낮음 · 집중 데이터 처리 간편	· 거래량 증가에 따른 확장의 유연 · 장애시 시스템 일부 기능 중단 · 순간 과부하시 부하 분산 · 시스템 확장시 부가 방식 · 개발 난이도가 높음 · 운영 난이도가 높음 · 집중 데이터 처리 복잡
투자 비용	· 초기 투자보다 상회	· 1개 지역 센터 증설 비용 증가
시스템 접속	· 주종관계 접속(효율 저하)	· 동등관계 접속(개방형)
다운 사이징	· 대응이 고려되지 않음	· 최 적
투자 효율	· 성능의 고정으로 선투자	· 소규모 시스템으로 대응 용이
부대 시설	· 과다한 공간 소요 · 부대 설비 비용의 과다	· 최소의 공간 소요 · 부대 설비 비용의 절감 · 소비 전력이 적음
정보 기술	· 기존 정보 기술 의존	· 최신 정보 기술 수용

의 성능 측정 프로그램을 통해 응답 시간이나 제한된 시간 동안의 트랜잭션 처리량 등은 측정이 가능하나 본 논문은 전체 계정 업무의 구축에 주안을 두었으며, TPC의 성능 측정치는 다량의 단순 거래를 처리하여 성능을 측정하는 것이므로 고려하지 않았다.

단일 메인 프레임과 분산 시스템에 대한 비교. 분석은 〈표 6〉과 같다.

5. 결론 및 향후 연구방향

단일 메인 프레임과 분산 시스템의 성능에 관한 비교 분석은 시스템의 이질성 때문에 특별한 의미를 부여할 수 없다. 그러나 TP 모니터를 이용한 분산 오픈 시스템의 구축 선례가 없는 국내 환경에서 본 연구에서 구축한 시스템이 다른 시스템 산정의 기준이 되리라 생각한다.

실제로 본 연구에서 구축한 시스템의 자료는 말레지아 흑화은행의 시스템 산정 자료로 이용되어 그 효율성이 입증 되었다.

TP 모니터는 제한된 시간 내에 처리가 되지 않을 경우 이를 롤백 시킬 수 있는 트랜잭션의 타임 아웃 기능이 있지만 서버간의 부하 분산 알고리즘이 고려되지 않아 최악의 경우 시스템의 성능이 급작 스럽게 떨어지는 단점을 가지고 있다.

향후 효율적인 부하 분산 알고리즘의 채택으로 준실시간 처리를 가능토록 하여야 할 것이며 효율적인 부하의 분산을 위한 알고리즘 개발이 지속 되어야 한다.

데이터 베이스와의 인터페이스를 위한 X/Open DTP XA 인터페이스의 지원도 구현 하여야 한다. 그러나 부하 분산만을 고려하면 전체 응답 시간이 따라서 늦어지게 되므로 서버간 부하 분산시 이의 고려가 필요하다.

참 고 문 헌

[1] C.J. Date, An Introduction to Database Systems Volume I, Addison-Wesely Publishing Company, 1990.

[2] David Bell and Jane Grimson, "Distributed Database Systems", Addison-Wesely Publishing Company, 1992.

[3] D. Lomet, "Using Timestamping to Optimize Two Phase Commit", Proc. of the 2nd Int'l Conf. on Parallel and Distributed Information Systems, pp. 48-55, Jan. 1993.

[4] James H. Johnson and David M. Hudson, "Open System OLTP Monitors", UniForum Conference Proceedings, pp. 155-169, 1992.

[5] Juan M. Andrade and Mark T. Carges, "On-Line Transaction Processing In Open Systems", UniForum Conference Proceedings pp. 123-135, 1992.

[6] Mary R. Hesselgrave, "Considerations for Building Distributed Transaction Processing Systems on UNIX System V", UniForum 1990.

[7] Philp A. Berstein, "Transaction Processing Monitors," CACM Vol. 33, No. 11, pp. 76-86, Nov. 1990.

[8] P. Triantafillou, "Recovering in Large Distributed Systems with Replicated Data", Proc. of the 2nd Int'l Conf. on Parallel and Distributed Information Systems, pp. 39-47, Jan. 1993.

[9] Robert Abbott and Hector Garcia-Molina, "Scheduling Real-time Transaction, a Performance Evaluation", Proc. of the 14th VLDB Conf. pp. 1-12.

[10] Sape Mullender, "Distributed Systems", ACM press Addison-Wesley Publishing Company, 1992.

[11] Transaction Processing Performance Council, "TPC Benchmark A", Transaction Processing Performance Council, 1992.

[12] X/Open, "The X/Open reference Model for Distributed Transaction Processing", X/Open Company Ltd., Guide, 1991.

[13] Yihjye Jack Hwang, Jiehyeong Sun, "CStalk, Distributed On-Line Transaction Processing for Banking System Integration," ICSI IEEE Volumn 2. 1994.

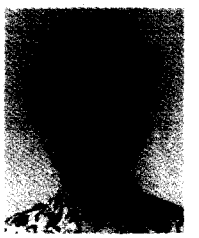
[14] 서필교, "분산처리시스템에서의 파일할당위치선정", 경영정보학회 pp. 379-386, 1994.

[15] 시에치노 시스팀컨설팅, "클라이언트/서버, Downsizing 정보시스팀 설계방법 및 핵심구현기술", 1993.

[16] 유호동, 임성채, 김명호, "고성능 온라인 트랜잭션 처리 모니터의 설계 및 구현", 한국정보과학회 논문지, pp. 816-825, 1994.
[17] 이기현, "클라이언트/서버 구조", 이한출판사, pp. 29-42, 1995.
[18] 이기현, "클라이언트/서버 환경 구축과 활용2," 정보과학회지 제12권 제2호, pp. 75-91, 1994.
[19] 홍장의,송운호,김영찬, "분산시스템에서 데드라인이 있는 트랜잭션의 원자성 제어 프로토콜", 한국정보과학회 논문지, 1993.

이 성 주

1970년 한남대학교 물리학과(학사)
1981년 조선대학교 경제학과(석사)
1992년 광운대학교 전자계산학과(석사)
1988년~1990년 조선대학교 전자계산소 부소장
1981년~현재 조선대학교 전자계산학과 교수
1995년~1997년 2월 조선대학교 산업대학장
1997년~현재 조선대학교 정보과학대학장
관심분야: 소프트웨어 공학, 프로그래밍 언어, 객체지향 시스템, Rough Set



최 완 규

1988년 서울대학교 인문대학 종교학과(문학사)
1992년~1993년 (주)공성통신 전산실
1993년~1994년 (주)한양 시스템 전산실
1994년 8월 조선대학교 대학원 전자계산학과(이학석사)
1994년 9월~현재 조선대학교 대학원 전자계산학과 박사과정
관심분야: 소프트웨어 공학, 프로그래밍 언어, 객체지향 시스템, Rough Set

나 영 남

1991년 조선대학교 전자계산학과(이학사)
1993년 조선대학교 대학원 전자계산학과(이학석사)
1994년 9월~현재 조선대학교 대학원 전산통계학과 (박사과정 수료)
관심분야: 지능정보, 퍼지시스템, 소프트웨어 공학, 프로그래밍 언어, Rough Set