바다/웹:웹과 객체지향 데이터베이스 관리시스템의 통합

김 완 석† · 이 장 선†† · 송 영 기††† · 박 진 섭†††† · 김 명 준††††† · 오 길 록†††††††

요 약

정보서비스 관점에서 대규모 정보 서비스 시스템을 개발하는 가장 좋은 방법중의 하나가 웹의 서비스 능력과 데이터베이스 관리시스템의 유용한 기능을 상호 보완적 측면에서 통합하는 것이다. 웹과 데이터베이스 관리시스템 통합의 핵심 요소가 웹-DBMS 통로이다. 웹-DBMS 통로는 데이터베이스 관리시스템에 대한 요청을 웹의 기술을 이용하여 데이터베이스 관리시스템에 접근하고, 데이터베이스 관리시스템의 처리 결과를 웹서버에게 전한다. 바다/웹은 웹의 서비스 능력과 객체지향 데이터베이스 관리시스템인 바다-III를 서로 보완적 측면에서 통합한 웹-DBMS 통로이다. 이 논문에서는 웹과 바다-III를 통합하기 위한 설계와 구현상의 경험을 설명하고 데이터베이스에 대한 질의들을 통해 바다/웹의 효과를 분석했다.

BADA/Web : Integration of The Web and An OODBMS

wan-seok Kim† · jang-sun Lee†† · young-kee Song††† · jin-sub Park†††† · myung-joon Kim††††† · kilnok O†††††††

ABSTRACT

We believe in terms of information service systems that one of the best ways to develop a large scale database service system is to integrate the service capability of the Web and the data management facility of database management systems in a complementary fashion. In such integration a database gateway is the core component. the web-database gateway accesses database management systems to serve the requests represented by using the Web technology. We designed BADA/Web be independent from the Web and DBMS as much as possible, which minimizes the performance overhead caused by connecting database management systems and makes BADA/Web portable. BADA/Web incorporates TCL into a library of it and handles concurrent requests efficiently. In this paper we describe our design and implementation experience in integrating the Web and BADA-III. We evaluate the performance of BADA/Web by measuring and comparing the latency and average response time for a simple query and also explore the effects of BADA/Web with some synthetic queries.

1. 서 론

1991년 CERN에서 월드 와이드 웹(WWW : World Wide Web)[1]이 개발된 뒤, 웹은 인터넷을 기반으로 하는 하이퍼미디어 정보 시스템으로서 지속적인 성장을 이루고 있다. 웹은 서버와 클라이언트 구조를 가지고 있다(그림 1). 웹 서버는 CGI (Common Gateway Interface)[5]와 HTML(Hyper Text Markup Language)[2]로 쓰여진 하이퍼 미디어 문서를 포함하고 있다. HTML은 태그를 기반으로 하여 많은 이기종간의 플랫폼에서 문서를 나타내는 것을 가능하게 하는 언어이

† 정 회 원 한국통신연구원 인터넷서비스 책임연구원
†† 정 회 원 인제대학교 정보통신공학과 교수
††† 정 회 원 친안외국어대학 산업정신과 교수
†††† 정 회 원 대전대학교 컴퓨터공학부 교수
††††† 종신회원 한국통신연구원, 선임/책임연구원, 컴퓨터·소프트웨어 기술연구소 소장
††††††† 종신회원 한국통신연구원 연구위원, 한국정보처리학회 회장
논문접수 : 1999년 11월 11일, 심사완료 : 2000년 11월 7일

다. HTML문서들은 관계되는 다른 데이터 혹은 다른 하이퍼미디이 문서와의 링크를 포함하고 있다.

웹서비는 서비와 클라이언트사이의 통신 규약으로 HTTP(HyperText Transfer Protocol)[3]를 지원한다. 웹서버는 클라이언트가 보낸 URL(Uniform Resource Locator)[4]로 요구하는 문서나 멀티미디어 데이터에 접근하고, 필요하다면 CGI를 통해 HTML 문서 형태로 변환하여 클라이언트에게 보낸다 클라이언트인 웹 브라우저는 전달 받은 문서를 해석해서 사용자들에게 나타내 준다.

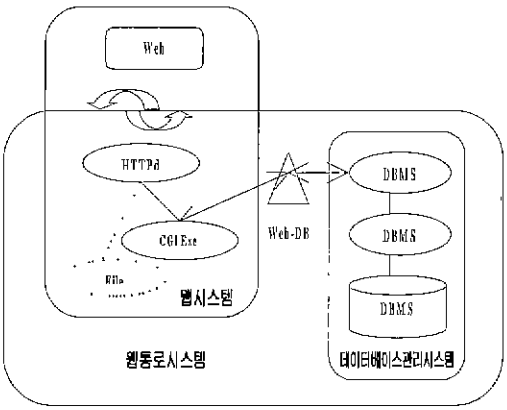
웹은 전 세계에 걸친 정보 공유 환경을 제공하기 위해 다음과 같은 기술적인 문제들을 해결했다 1) 전 세계에 퍼져 있는 데이터의 유일성(Identity)[4], 2) 하이퍼미디어 문서 기술[2], 3) 네트워크를 통한 문서 전송을 위한 통신 규약[3].

그러나 웹은 여전히 인터넷을 기반으로 하는 대규모 하이피미디이 정보 시스템으로서 해결해야 할 다음과 같은 문제들을 안고 있다. 첫째, 많은 이기종 데이터들을 조직하고 관리하기가 쉽지 않다. 왜냐하면 웹의 주요 저장장치가 파일 시스템이기 때문이다. 즉, 파일 시스템에 의해 많은 제한이 야기되기 때문에 대용량 데이디 서비스 시스템을 구축하기는 쉽지 않다. 들째, HTML의 질의 표현 능력은 제한적이다. 따라서 복잡한 질의 형태를 HTML로 나타내기란 쉽지 않다 셋째, HTTP 통신 규약의 속성상 상태 지향 서비스(state-oriented service) 지원이 용이하지 않다.

반면에 데이터베이스 관리시스템은 많은 데이터를 조직하고 관리하는 다양한 기능을 제공한다. 그러나 인터넷 기반 대규모 하이퍼미디어 정보 시스템상에서 데이터의 전달과 표현을 위한 충분한 기능은 가지고 있지 않다. 그러므로 (그림 1)과 같이 정보 제공 시스템의 관점에서 대규모 정보 서비스 시스템을 개발하는 가장 좋은 방법중의 하나가 데이터베이스 관리시스템의 유용한 기능과 웹의 서비스 능력을 상호 보완적으로 통합하는 것이다

웹과 DBMS를 상호 보완적으로 통합하는 핵심 요소인 웹-DBMS 통로는 웹인터페이스를 사용하여 데이터베이스 처리 요청들을 데이터베이스 관리시스템에 전달하고 처리 결과를 웹서버에게 전한다.

바다/웹은 웹의 서비스 능력과 객체지향 데이터베이스 관리시스템인 바다-III를 서로 보완적 측면에서 통합한 웹-DBMS 통로이다. 웹을 데이터베이스 관리시스템에 연결함으로 생길 수 있는 성능감소를 최소화하기 위해, 바다/웹을 가능한 한 웹과 데이터베이스 관리시스템을 독립적으로 구현하였다. 한편 바다/웹은 TCL (Tool Command Language)을 라이브러리화하여 바다/웹 명령을 사용한 프로그래밍을 가능하게 했다



(그림 1) 웹과 DBMS의 연동

이 논문에서는 웹과 바다-III를 통합하기 위한 설계와 구현상의 경험을 설명하고, 또한 데이터베이스 질의를 통해 바다/웹의 효과를 분석했다

이 논문의 나머지 부분은 다음과 같이 구성된다 2장에서 기존의 웹-DBMS 통로를 살펴보고, 강단점을 분석한다. 그리고 3장에서 통로 설계 원리를 포함하여 바다/웹의 구조와 설계 내용을 설명한다. 그리고 구현상의 세부 사항들과 실험적 결과들을 각각 4, 5장에서 언급한다. 마지막 장에서 결론을 맺는다.

2. 웹-DBMS 통로에 대한 조사

웹-DBMS 통로 구조에 관한 분류가 [9]의 논문에서 제시 되었다. 이 분류는 데이터베이스에 대한 접근이 이디에서 이루어지는가에 중점을 두고 웹-DBMS 통로의 범주를 나누고 있다 데이터베이스는 서버쪽 혹은 클라이언트쪽에서부터 접근할 수 있다. 즉 서버나 클라이언드를 확장함으로써 데이티베이스 관리시스템과 웹을 통합할 수 있다.

서버쪽을 확장하는 경우 웹-DBMS 통로는 CGI나 API를 이용하여 구현할 수 있다 CGI를 이용한 통로의 경우는 두가지 형태로 나눌 수 있다. 첫째, 웹-DBMS 통로 자체가 실행 가능한 CGI 실행 가능(CGI executable)인 경우이다. 둘째, CGI와 데이터 응용 서비로 구성된 구조에서 CGI는 데이터 응용 서버에게

단지 요청을 넘겨 주기만 하는 CGI 응용 서버(CGI application server)의 경우이다.

클라이언트쪽을 확장하는 통로의 경우에는, 데이터베이스 관리시스템은 브라우저를 확장하거나 혹은 MPEG 플레이어와 같은 외부 뷰어를 이용함으로써 직접 접근할 수 있다.

본 논문은 고비용의 클라이언트측의 데이터베이스 관리시스템 연결 구조에 대해서는 언급하지 않는다. 저비용 고효율성이 보장되는 서버측 구조만을 간단히 살펴 보기로 한다.

CGI 실행 가능 구조는 기존의 웹과 데이터베이스에 대한 개발 환경을 그대로 사용하여 웹-DBMS 통로를 구현할 수 있기 때문에 웹과 데이터베이스 관리시스템의 통합은 단순하고 직접적이라 웹서버나 브라우저측의 변경이 필요 없다. 이러한 구조는 각각의 데이터베이스 요청을 통로 프로세스가 데이터베이스 서버에게 전달하고 데이터베이스 서버를 통해 요구 처리를 수행한다. 웹서버는 통로 프로세스로부터 결과들을 받고난 이후에 통로 프로세스와 데이터베이스와의 연결을 종료한다. 이 방법은 데이터베이스 관리시스템의 응용(클라이언트)이 매 처리요구 마다 초기화되어 동작되기 때문에, 데이터베이스 관리시스템이 연결된 상태에서 반복적인 질의를 최적화하여 서비스하는 일반적인 장점을 소용없게 만들어 버린다. 그뿐만 아니라 데이터베이스 관리시스템을 사용하기 위해 동시에 들어 오는 많은 수의 요청을 처리해 내야 하는 대규모 서비스 환경에서 시스템 자원 제약과 웹서버와 웹-DBMS 통로간의 통신과 프로세스 생성 비용으로 인하여 심각한 성능저하를 초래할 수 있다. WebDBC [13], Cold Fusion [14], Web/Genera [15], GSQL [16], WDB [17]는 모두 CGI 실행 가능 구조를 가지는 대표적 제품들의 예들이다.

CGI 응용 서버 구조의 웹-DBMS 통로는 통로 부분(gateway)과 응용 서버(application server), 이 두가지 모듈로 나누어 진다. 통로 모듈은 CGI형의 실행 가능 프로세스로 웹서비스 요청을 휘지상태의 응용 서버에게 전달한다. 응용 서버 모듈은 보통 데이터베이스 관리시스템의 클라이언트 응용으로서 요청을 기다리는 데몬 프로세스로 동작한다. 응용 서버 프로세스의 수는 가용 시스템 자원을 고려하여 적절히 결정할 수 있으나 프로세스간 통신으로 야기되는 별도의 비용을 부담해야만 한다. UniWeb[10], O2Web[11] 그리고 MARS [18]는 이러한 구조를 가지는 제품들이다.

Netscape의 NSAPI와 Internet Information Server의 ISAPI경우에는 사용자가 웹서버의 기능을 확장할 수 있는 API를 지원한다. 그러므로 웹-DBMS 통로는 확장 가능한 API로 구현될 수 있어서 웹서버 프로세스에 동적으로 연결된다. 이는 웹서버의 CGI 프로세스 관리 비용을 감소시키 주는 반면에 이와 같이 특정한 웹서버의 API에 의존하면 호환성이 문제가 된다.

한편, 웹-DBMS 통로는 UniWeb[10] 등과 같이 데이터베이스 클라이언트 프로그램으로 구현될 수도 있다. 이 경우, 클라이언트 프로그램을 데이터베이스 관리 시스템을 위한 특정 웹클라이언트로 간주되며 특정 데이터베이스 관리 시스템에 의존한다.

따라서 바다/웹은 웹-DBMS 통로에 대한 자료조사를 통해 지적된 기존의 웹-DBMS 통로의 문제점을 해소하기 위해 첫째, 저비용 고효율성을 보장하는 서버측 CGI 응용서버 구조를 채택했다. 둘째, 웹서버와 웹-DBMS 통로간의 통신과 프로세스 생성 부담을 다수의 응용서비를 데몬화함으로서 최소화했다.

3. 바다/웹의 구조

바다-III는 한국전자통신연구원에서 개발한 멀티미디어 데이터베이스 관리시스템이다. 바다-III는 객체지향 데이터베이스 관리시스템으로서 자동색인과 전문에 대한 내용기반 검색이 가능한 정보 검색 기능이 밀결합(Tightly Coupled)되어 있는 것이 특징이다[12].

2장의 웹-DBMS 통로에 대한 조사를 통해 파악된, 웹-DBMS 통로 구조에 따른 비용과 효율성의 차이, 웹서버와 웹-DBMS 통로간의 통신과 프로세스 생성에 의한 성능 저하 초래, 시스템 의존도에 따른 호환성의 문제 등을 고려하여 바다/웹은 다음과 같은 설계 원리를 택하였다.

3.1 바다/웹의 설계 원리

정보서비스의 관점에서 보면 대규모 정보 서비스 시스템을 개발하는 가장 좋은 방법중의 하나가 데이터베이스 관리시스템의 유용한 기능과 웹의 서비스 능력을 상호 보완적으로 통합하는 것이다. 바다/웹은 웹과 바다-III를 통합하는 통로이다. 기존의 웹-DBMS 통로들의 문제점과 바다-III의 특징들을 조사하여, 바다/웹은 다음과 같은 요소들을 고려하여 설계했다

- 웹과 데이터베이스 관리시스템과의 독립성 : 바다/웹의 설계에서 주된 원리중의 하나가 바다/웹을 웹, HTTP 통신규약[3], 데이터베이스 관리 시스템으로부터 가능한 한 분리 시키는 것이다. 바다/웹의 주목적이 웹과 데이터베이스 관리 시스템을 통합하는 것이라는 점을 고려하면 웹과 데이터베이스 관리 시스템으로부터 독립적인 웹-DBMS 통로를 구현하는 것은 모순처럼 생각될 지 모른다. 그러나 두 시스템에 대한 독립성은 급속히 성장하고 있는 웹과 통신규약으로 인해 야기되는 수정에 대한 부담을 피할 수 있을 뿐만 아니라, 표준화된 API를 사용하는 웹-DBMS 통로를 개발하면 기존의 웹서버에 자연스럽게 연결할 수 있다. 이러한 요구사항은 바다/웹을 상호 분리된 모듈들로 설계함으로써 만족시킬 수 있다. 1) 데이터베이스 관리시스템에 관한 지식이 없이도 구현 가능한 모듈-통로(CGI 수행가능) 프로그램, 2) 웹환경에 대한 지식이 없이도 구현 가능한 모듈-데이터베이스 관리시스템 응용 서버, 3) 두가지 모듈을 연결해 주는 모듈-메시지 관리 및 모니터링 데몬

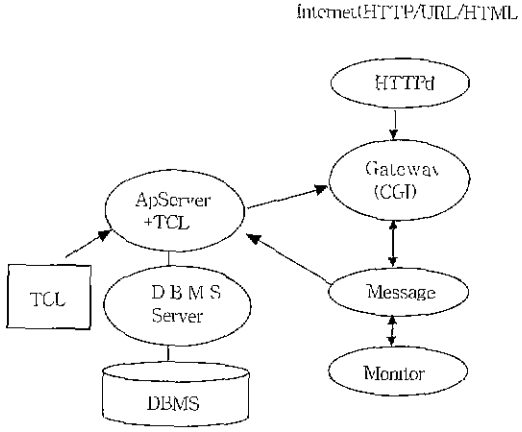
- 성능상의 추가 부담 최소화 동일한 데이터베이스에 접근하는 동일한 응용 프로그램이 클라이언트의 모든 요청 마다 실행된다면, 매 요청 마다 데이터베이스 관리시스템의 초기화 과정에 의한 프로세스의 생성과 소멸 부담이 시스템 성능을 저하시킨다. 그래서, 프로세스의 생성 소멸에 대한 부담을 데이터베이스 응용 서버와 모니터를 데몬 프로세스로 구동함으로써 최소화한다 즉, 모니터가 항상 클라이언트로부터의 요청을 기다리고, 데이터베이스 응용 서버는 초기화된 데이터베이스 관리시스템과 연결된 상태에서 모든 요구 사항을 수행함으로써 프로세스 생성 소멸 비용을 줄인다

- 동시 요청에 대한 효과적 처리 : 복수의 클라이언트로부터 동시에 일어나는 요청은 모두 처리되어야 한다 동일한 데이터베이스 관리시스템에 접근하는 하나의 응용 서버만으로 동시에 발생하는 요청들을 효과적으로 처리하지 못한다. 그래서, 동시에 발생하는 요청들을 복수의 데이터베이스 응용 서버를 구동하여 각 요청들을 모든 응용 서버들에게 고르게 분배하여 처리한다

- 스크립트 언어의 통합 : 바다/웹 메소드를 사용하여 사용자가 원하는 프로그래밍이 가능하도록, TCL이나 Perl과 같은 스크립트 언어를 지원한다

3.2 바다/웹의 설계 구조

바다/웹의 주된 요소들은 Gateway, Monitor, Apserver (그림 2)이며 하나의 메시지큐를 공유한다. Gateway는 CGI 실행 가능 구조이며, 웹서버를 통해 전해지는 클라이언트로부터의 요청들을 메시지큐에 적재해 놓는다. 메시지큐에 요청을 적재한 Gateway는 Apserver로부터의 요청의 수행 결과인 데이터베이스에 대한 데이터를 기다리게 된다. 즉, 데이터베이스 관리 시스템의 처리 결과로 얻은 데이터를 분석하고, 데이터에 대한 MIME(Multi-Purpose Internet Mail Extension) 타입을 설정해서 웹서버에게 MIME 타입과 함께 데이터를 보낸다.



(그림 2) 바다/웹의 구조

바다/웹이 구동을 시작할 때, Monitor가 생성된다 Monitor는 응용서버의 이름과 응용 서버의 갯수, 그리고 서버의 경로와 같은 시스템 환경에 관련된 파라메터를 참조하여 Apserver라는 응용 서버를 생성한다. Monitor는 종료하지 않는 데몬으로, 주기적인 상태 점검을 통해 응용 서버를 모니터링하여 서버가 정상적이지 못할 경우에는 오류를 복구하는 일을 한다 또한, Monitor는 메시지큐에 있는 내용을 점검하여 요청 자체가 상태를 나타내고자 하는 것인지 모든 응용 서버

들에게 브로드캐스팅해야 할 지를 고려한다. 만약 요청이 모든 응용 서버들에게 브로드캐스트 되어야 한다면, n이 응용 서버의 갯수라고 했을 때, Monitor는 각각의 응용서버에 맞게 요청을 수정하여 n개의 복사본을 만들어 메시지큐에 저재한다 만약 요청이 트랜잭션 처리와 같이 상태를 지향하는 것이라면, Monitor는 각각의 요청을 식별하기 위해 쿠키(cookie)를 생성하고, 생성된 쿠키에 대응하는 새로운 응용서버를 생성한다(다음장에서 상태지향적인 요청을 바다/웹이 이떻게 처리하는가에 대해서 자세히 설명한다). 그리고 새롭게 생성된 응용서버는 Monitor의 개입없이 동일한 쿠키를 가지고 있는 요청을 처리하며 상태지향적인 요청이 처리되고 난후에 종료하게 된다

Apserver는 바다-III 데이터베이스 관리시스템의 응용 프로그램일뿐만 아니리 바다/웹의 서비이다. 모든 Apserver들은 바다/웹이 시작할 때 Monitor에 의해 생성되이 데몬으로 존재한다. 각각의 Apserver는 능동적으로 메시지큐로 부터 한번에 하나의 요청을 읽어 들어 처리하고, 처리후 특정 위치('/tmp/cgi_pid')에 처리결과를 적재한후 Gateway에게 요청의 종료를 사용자 신호(user signal)을 사용하여 Gateway에게 알려 준다. 각각의 Apserver는 메시시큐에 요청이 존재할 때 이러한 과정을 반복한다

4. 바다/웹의 구현

바다/웹은 웹과 바다-III를 상호 보완적 측면에서 통합하여 데이터베이스 질의 결과를 HTML 문서 형태로 변환시켜주는 용이함과 데이터 접근 관리, 다양한 실패로부터의 자동화된 데이터 복구, 색인과 검색, 트랜잭션 처리, 복잡한 데이터 형태와 관리, 보안을 위한 데이터베이스의 기능들을 웹시스탬에 제공한다

4.1 웹-DBMS 통로

바다/웹은 바다-III 데이터베이스관리시스템을 관리하고 데이터를 검색하기 위한 웹-DBMS 통로 명령어들을 제공해 준다. 즉, 웹-DBMS 통로 명령어들은 데이터베이스 스키마 구축과 갱신, 트랜잭션 관리, 클래스, 오브젝트, 메소드, 속성, 색인, 사용자, 데이터 접근 체계 관리 등의 데이터베이스관리시스템의 기능을 웹을 통하여 사용자가 쓸 수 있도록 해준다. <표 1>에 바다/웹의 웹-DBMS 통로 명령어(이들 명령어는 TCL 스크립터와 함께 사용 가능함, 기술방법은 상이함)들을 요약해 놓았다.

〈표 1〉 바다/웹 통로 명령어

구분	명령어
클래스	CreateClass, DeleteClass, ConnectClass, DisconnectClass, SubclassCounter, GetSchemaCounter, GetsubclassName
속 성	AddAttribute, UpdateAttribuitename, DeleteAttribute, UpdateAttribuitetype, AttributeCounter, GetAttributename
메스드	AddMethod, DeleteMethod, GetMethodname UpdateMethodname, UpdateMethodpath, MethodCounter,
색 인	CreateIndex, GetIndex, DeleteIndex, IndexCounter
권 한	PrivilegeCounter, PutPrivilege, GetPrivilege, UpdatePrivilege
객 체	AddObject, DeleteObject, DeleteAllObject, GetObjectPoid, UpdateObject, ObjectsCounter, GetAttributevalue
질 의	QueryResultcounter, GetQueryresult
트랜잭션	TxBegin, TxCommit, TxRollback
사용자관리	RegisterUser, DeleteUser, UserCounter, GetUsername, UpadatePassward

웹-DBMS 통로 명령어들은 HTTP의 GET이나 POST 메소드를 사용하여 HTML 문서 혹은 URL형태 안에서 나타날 수 있다.

다음에 보여주는 예제는 GET 메소드를 이용하여 URL의 질의 문자열에서 웹-DBMS 통로 명령어를 나타내는 형식을 보여주고 있다

예를 들어, http : //bada.etri.re.kr/Gateway?function=Create Class&classname=new class& 에서와 같이, 해당 URL에 접근하여 bada.etri.re.kr에서 구동하고 있는 HTTP 서버에게 질의 형태로 포함된 데이터를 웹-DBMS 통로에게 보낸다. 질의 문자열에서 CreateClass 통로 명령어는 새로운 클래스인 new_class의 생성 요청을 나타낸다

4.2 데이터 변환

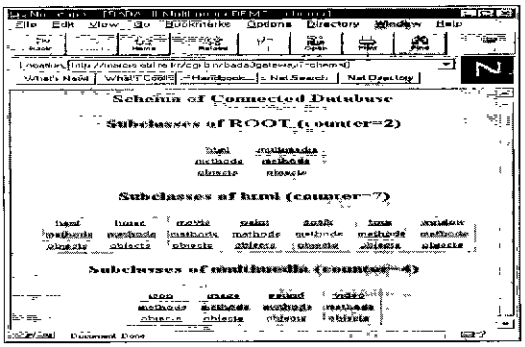
웹 클라이언트 프로그램은 많은 다른 서버들을 통해 데이터에 접근한다. 그리고 데이터는 서버에서 클라이언트로 전송하기 전에 프로그래머의 의도에 따라 HTML 문서로 형태가 변환될 수 있어야 한다

요청된 데이터베이스의 데이터는 사용자가 정의한 문서 형태로 변환될 수 있다. 바다/웹은 두가지 변환

방법을 지원하기 위해 데이터베이스 관리자 명령어와 TCL 스크립트 언어를 제공하며, 바다/웹의 Apserver 안에서 HTML 문서 형태로 변환된다.

바다/웹의 데이터베이스 관리자 명령어들은 데이터베이스관리시스템 조작을 위한 것들로 schema(), class(), object(), attribute(), method(), query() 등이 있다. 데이터베이스 관리자 명령어들은 HTTP의 GET 메소드만을 지원한다 명령어의 실행 결과는 HTML 문서의 정해진 양식으로 변환된다.

http : //bada etri re.kr/Gateway?schema()와 같이 URL로 접근함으로써, 사용자는 데이터베이스 스키마에 관한 정보를 얻을 수 있다. Apserver는 schema()를 실행하여 정보를 가져 오고, HTML 문서의 정해진 양식으로 변환한다. (그림 3)에서 결과 문서를 한 예로 보여준다.



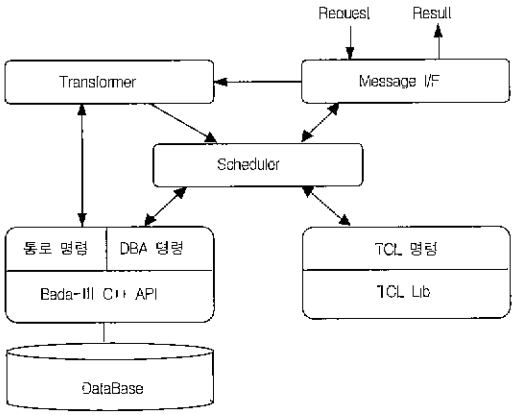
(그림 3) 바다/웹을 통한 DB 스키마 표시

바다/웹은 TCL을 지원하여 사용자가 원하는 HTML 문서의 양식을 서버에서 정의하도록 해준다 TCL은 고수준 스크립트 언어이며 C쉘이나 Perl과 유사하다. TCL은 인터프리티더 언어(Interpreted Language)로서 응용안에서 확장 및 내장가능하다. TCL 인터프리터를 수행함으로써 야기될 수 있는 성능상 감소와 TCL의 응용으로써 응용서버를 구현해야 하는 것을 피하기 위해서 TCL1.17b1의 쉘 구조를 제거하여 라이브러리형태로 변형하여 인터프리터의 주된 기능들을 Apserver (그림 4)안에서 일반적인 함수 기능들로 재구현했다.

Apserver의 커널은 메시지 인터페이스(message interface), 데이터 출력 변환기(data output transformer), 명령어 스케줄러(command scheduler) (그림 5)와 같이 세가지의 주된 모듈로 구성된다.

응용서버커널		
통로명령	DBA명령	TCL명령
바다-III C++ API		TCL Lib

(그림 4) 바다/웹 응용서버와 TCL의 결합



(그림 5) 바다/웹 응용서버 커널의 구조

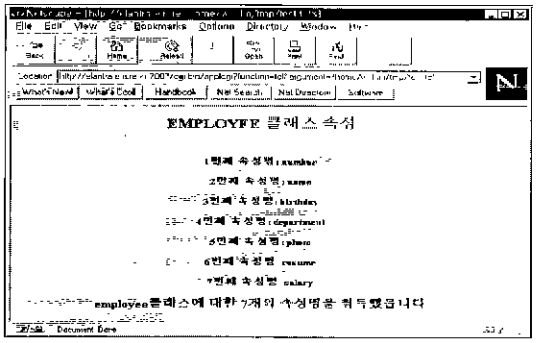
메시지 인터페이스는 메시지큐로부터 한번에 하나의 요청을 읽어 들여 데이터 출력 변환기로 보낸다. 그리고 Gateway에게 요청의 종료를 알려주는 일도 담당한다. TCL과 응용서버 내의 데이터의 출력 방법이 상이한데, 응용서버의 데이터 출력 변환기가 Apserver와 TCL의 처리 결과 출력을 제어 한다. 명령어 스케줄러는 TCL 파일을 제외한 모든 요청들은 바다-III C++ API를 통해 처리한다. TCL 파일의 경우 TCL 라이브러리 함수를 사용한다.

(그림 6)은 바다-III 데이터베이스관리시스템으로부터 employee라는 클래스 속성의 이름값들을 얻은 정보를 HTML문서 형태로 변환하는 과정을 나타내는 TCL 파일의 예제이다.

```
# file name :- getattribute.tcl
# tcl scripter file for get attributes
puts stdout "<center><p><h2>employee클래스 속성"
set max [AttributeCounter employee]
for { set i 1 } { $i <= $max } { incr i } {
    puts stdout "<p><h4>$i번째 속성명.."
    puts stdout [GetAttributename employee $i]
}
puts stdout "<p><h4>employee클래스에 대한 $i개의
            속성명을 획득했읍니다."
puts stdout "</center>
```

(그림 6) TCL 파일 기술 예

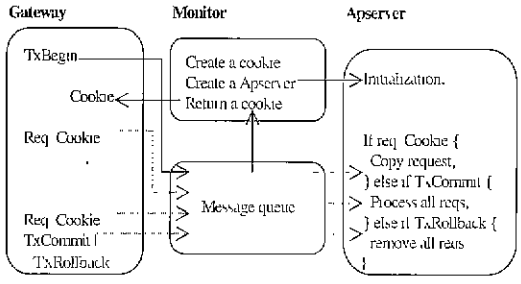
(그림 6)은 http://bada.etri.re.kr/Gateway?function=tcl&filename=/getattribute.tcl&의 URL에 접근하여수행하는 파일 내용을 보여준다.
그 결과는 (그림 7)과 같다



(그림 7) TCL파일 실행 결과

4.3 트랜잭션 지원

바다/웹은 트랜잭션 처리를 지원한다. Gateway에 의해 메시지큐에 적재된 요청이 상네를 지향하는 것이라면 Monitor는 트랜잭션 처리를 위하여, 이 요청을 다른 것들과 구분하기 위해 쿠키를 생성한다. Monitor는 Gateway에게 쿠키를 되넘기는 한편, 프로세스의 인자로서 쿠키를 가지는 새로운 Apserver 프로세스를 생성한다. 쿠키를 인자로 새로 생성된 서버는 Monitor의 간섭없이 동일한 쿠키를 가지는 요청들을 처리한다. Apserver는 배치(batch)의 형태로 트랜잭션의 모든 요청들을 한번에 처리한다. 다시 말해, Apserver는 트랜잭션의 모든 요청들에 대한 복사본을 만들어 놓는다. 그리고, Apserver가 트랜잭션의 허락 혹은 철회를 얻게 되면, 복사되어 있는 요청들의 수행을 시작한다. 프로세스의 종료는 트랜잭션을 처리하고 난 후 이루어진다. (그림 8)은 바다/웹에서 트랜잭션이 어떻게 처리



(그림 8) 바다/웹 트랜잭션 처리

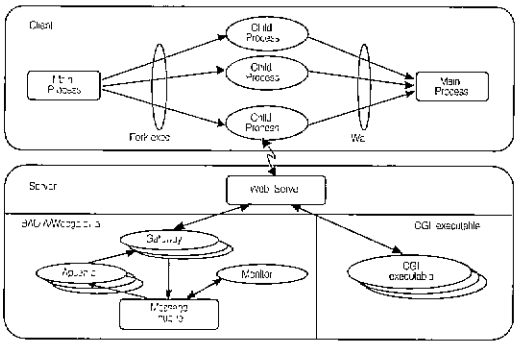
되는 가를 보여주고 있다

5. 바다/웹의 성능평가

5.1 실험을 위한 설계

본 실험은 (그림 9)에서 보여주듯이 웹서버를 구동하는 서버 머신, 바다/웹, 바다-III DBMS, 서버에게 요청을 생성하기 위한 클라이언트 머신을 가지는 시스템으로 환경을 설정했다

<표 2>는 본 실험에 사용한 머신을 나타낸다



(그림 9) 성능평가 프로세스 구조

〈표 2〉 성능평가를 위한 시스템 규격

	Client	Server
System(Sun)	Ultra Enterprise 3000	Sparc 20
CPU	UltraSparcX4(167MHz)	TIX1(40MHz)
Memory	1GB	256MB
OS	SunOS 5.5.1	SunOS 5.4
NetWork	10Mbps	10Mbps

본 실험을 위해, 클라이언트 머신에서 주 프로세스(main process)가 자식 프로세스(child process)를 생성한다. 클라이언트의 주 프로세스는 모든 자식 프로세스의 종료를 기다린다. 클라이언트의 주 프로세스는 각각의 자식 프로세스에 의해 생성된 요청의 갯수, DBMS와 네트워크의 성능에 대한 각각의 요청에 대한 평균 응답시간(response time), 서비스 실패율 등과 같은 통계 수치들을 모은다. 각각의 자식 프로세스는 임의의 URL()로 HTTP 프로토콜을 이용하여 웹서버에게 요청을 보내고, 웹서버로부터 결과를 받고 난후에 다시 또 다른 하나의 요청을 보낸다. 각각의 자식 프로세스는 이러한 과정을 64번 반복한다

바다/웹의 성능을 평가하기 위한 가장 좋은 방법은, 시험을 위한 결과가 잘 나타나는 데이터베이스관리시

스템 클라이언트 CGI를 구현하여 바다/웹의 경우와 성능을 비교하는 것이다 그래서 본 실험에서는 CGI 응용 서버 구조를 가지는 바다/웹통로와 CGI 실행 가능한 프로그램을 구현하였다

바다/웹에서 메시지큐는 Gateway, Monitor, Apserver들 사이에 공유된다. Gateway는 CGI 실행 가능한 형태이고, Monitor는 Apserver의 감독자이다. 그래서 서버에서 동시에 처리되는 요청수는 Apserver수에 달려 있다.

본 실험에서는 Apserver의 갯수를 1, 2, 4, 8, 16, 32로 다르게 해 보았다. 다시 말해 서버에게로 동시에 들어 올 수 있는 요청의 수를 자식 프로세스의 수만큼 다르게 해 보았다 이는 요청에 대한 상호간의 도착시간을 다르게 해 보는 것과 유사하다.

5.2 결 과

각각의 요청을 위한 평균 응답시간은 주로 데이터베이스관리시스템과 네트워크의 성능에 달려 있다. 그래서 바다/웹과 CGI 실행 가능 구조에 대한 잠복시간(latency)을 측정하는 것이 더욱 합리적이다 본 실험에서는 데이터베이스관리시스템 응용서버에게 요청을 보내고 응답을 받기까지 걸리는 시간을 바다/웹의 잠복시간으로 정의하였다. 네트워크의 영향을 없애기 위해 동일한 머신(Sun Ultra Enterprise 3000)상에서 자식 프로세스, 바다/웹, 웹서비를 구동하였다. 실험에서 바다/웹의 Apserver는 데이터베이스관리시스템의 영향을 제거하기 위해 바다-III에 접근하지 않고 요청자에게 요청을 받았다는 확인만을 넘겨준다.

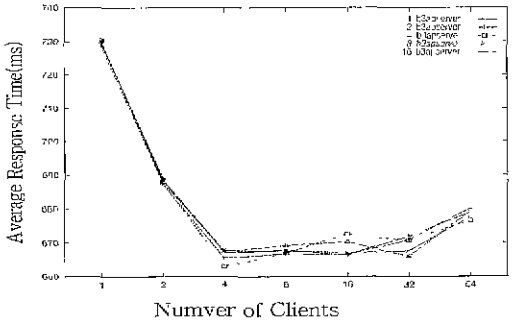
(그림 10) 은 바다/웹과 CGI 실행 가능 구조의 평균 잠복시간을 보여준다. 모든 경우에 있어 바다/웹의 성능이 CGI 실행 가능 구조보다 나았다 평균 잠복시간은 클라이언트가 증가함에 따라 감소했는데, 이것은 서버에서의 동시에 처리가능한 요청의 수가 증가하기 때문이다. 물론 잠복시간은 Apserver의 수가 증가함에 따라 감소한다.

그러나 (그림 10)에서 클라이언트의 수가 16에서 64로 달라졌을 때 성능상의 변화를 나타내기가 힘들었지만 두가지 구조를 가지는 형태가 다 진진적으로 성능이 감소하였다고 말할 수 있을 것이다 이것은 아마도 프로세서가 문맥을 교체하는데 걸리는 시간과 자원의 제약 때문에 일어난 것으로 본다.

5.3 민감성

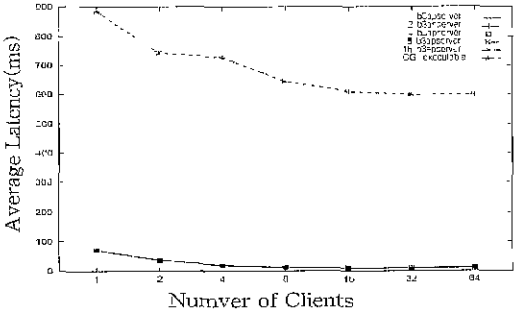
네트워크와 데이터베이스관리시스템에 대한 바다/웹의 민감한 정도를 평가해 보기 위해, 네트워크와 데이터베이스관리시스템의 성능을 다르게 해 보는 것이 가능할 것이다. 본 실험에서 네트워크의 성능을 바꾸지는 않았고, 그 대신 바다-III에 접근하기 위한 간단한 질의어를 적용해 보았다 본 실험에서 테스트 데이터베이스는 (그림 7)에 있는 employee라고 하는 하나의 클래스를 구성했다. 그리고 다음과 같은 질의어를 본 실험에 사용했다

(그림 11)은 질의에 대한 평균 응답시간을 보여준다 우리는 이미 잠복시간에 있어 바다/웹이 CGI 실행 가능 구조의 경우보다 나은 것을 알 수 있었기 때문에 여기에서 두가지 형태의 성능을 굳이 비교해 보지는 않았다. (그림 11)의 그래프는 (그림 10)에서 보여주는 것과 비슷한 양상을 띄고 있다. 즉, 클라이언트의 수가 증가함에 따라 평균 응답시간이 낮아지고 있었다. 왜냐하면 서버에서 동시에 처리되고 있는 요청의 수가 증가하기 때문이다. 그러나 클라이언트의 수가 32를 넘어갔을 때 응답시간은 증가하였다 이것은 메시지큐에 있는 요청의 수가 요청을 처리해 내는 Apserver의 수보다 훨씬 많기 때문이다. 그래프에서는 다소 불안



(그림 10) 바다/웹과 CGI 실행의 평균 잠복시간

select * from employee where salary > 80000



(그림 11) 바다/웹의 평균 응답시간

정한 곡선의 움직임을 볼 수 있는데, 이것은 아마 네트워크의 불안정한 상태로 인해 야기된 것으로 본다.

6. 결론과 향후과제

정보서비스의 관점에서 보면 대규모 정보 서비스 시스템을 개발하는 가장 좋은 방법중의 하나가 상호 보완적 측면에서 데이터베이스관리시스템의 유용한 기능과 웹의 서비스 능력을 통합하는 것이라고들 한다. 그러한 통합에 핵심 요소가 웹-DBMS 통로이다. 웹의 기술을 이용함으로써 생기는 요청들을 처리하기 위해 웹-DBMS 통로는 데이터베이스관리시스템에 접근한다. 바다/웹은 웹의 서비스 능력과 객체지향 데이터베이스관리시스템인 바다-III를 서로 보완적 측면에서 통합하는 통로이다.

우리는 가능한 한 웹과 DBMS로부터 독립적인 바다/웹을 설계하였다. 이를 통해 데이터베이스관리시스템에 연결함으로써 생길 수 있는 성능상의 추가 부담을 최소화 하고 바다/웹을 이동가능(portable)하게 설계하였다. 바다/웹은 TCL을 하나의 라이브러리로 병합하여 프로그래머의 동적 데이터 표현 요구를 효과적으로 해결했다. 이 논문에서 우리는 하나의 간단한 질의어에 대한 잠복시간과 평균 응답시간을 측정하고 비교함으로써 바다/웹의 성능을 평가해 보았다. 성능비교수치는 웹-DBMS 통로 설계 구조의 중요성을 강조해 주고 있다.

현재 바다/웹은 일괄적으로 상태지향적인 요청들을 처리해 내고 있으나 향후 더욱 효과적으로 그러한 요청들을 지원하도록 할 계획을 가지고 있다. 우리는 또한 바다/웹을 CORBA 통로와 함께 통합하기 위해, 다른 데이터베이스관리시스템에 접근하기 위해 확장, 수정할 계획을 가지고 있다.

참 고 문 헌

[1] T. Berners-Lee, R Cailliau, J.-F. Groff, and B Pollermann, "World-Wide Web : The Information Universe." *Electronic Networking : Research, Applications, and Policy*, Vol.1, No.2 Westport CT, 1992, pp.52-58

[2] T Berners-Lee and D Connolly, "Hypertext Markup Language Specification - 4.0," W3C's Recommendation, Dec 1997.

[3] T. Berners-Lee, "Hypertext Transfer Protocol-HTTP/1.1," Internet RFC 2068, May 1997.

[4] T. Berners Lee, "Uniform Resource Locators," Internet RFC 1808, Dec. 1995

[5] D. Robinson, "The WWW Common Gateway Interface Version 1.1," Internet Draft, Jan. 1996

[6] J. Ousterhout, *Tcl and The Tk Toolkit*. Addison-wesley, 1994.

[7] Larry Wall and Randal L. Schwartz, *Programming Perl*, O'Reilly & Associates, Inc, 1990.

[8] Martin R Arick, *Unix C Shell Desk Reference*, John Wiley & Sons, 1993.

[9] P.-C Kim, "A Taxonomy on The Architecture of Database Gateways for The Web," Proc. *of the 13th ICAST and 2nd ICMIS*, Shaumburg, IL, Apr 1997

[10] P.-C. Kim, "UniWeb - A Database Gateway for The Web," *Database Journal*, Vol.3-1, 1996, pp 65-84.

[11] L Latham, "Client/Server Computing - Strategic Directions, Tactical Solutions," *InSide Gartner Group This Week*, Vol.X, No.20, Gartner Group, May 18, 1994, pp.1-5

[12] M. O. Choi, M. Y. Lee, S. T. Jun, J Kim, O. J. Cho, Y K. Kim, and K. H. Hong, "Design of The Object Kernel of BADA-III - An Object Oriented Database Management System for Multimedia Data Services," *Proc. of the Workshop on Network and System Management*, 1995.

[13] Bo Frese Rosmusen and Beno Pirenne, "WDB - A WWW to Sybase Interface," *Proc. of the Workshop on Network and System Management*, 1995

[14] "Allaire ColdFusion Overview," 1995, http : // www allaire com

[15] Jason Ng, "GSQL A Mosaic SQL Gateway," Dec. 1993, http : //www.ncsa.uiuc.edu/SDG/People/jason/pub/gsql/starthere.html.

[16] Mars : "Internet Transacuon Processing," http : // www progress com/beta/internet/mars

[17] "Internet Server API Overview," http : //www.microsoft com/win32dev/apiext/isapimrg.html.

[18] URL : http : //www stormcloud com.

[19] URL : http //gdbdoc.gdb org/letovsky/genera/genera.html

[20] URL : http //www.netscape.com.

김 완 석

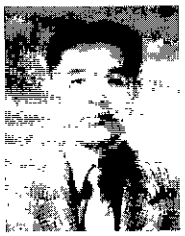
e-mail : wskim@etri.re.kr
1982년 영남대학교 물리학과 졸업(학사)
1992년 계명대학교 전자계산학과 졸업(석사)
1988년~현재 한국전자통신연구원 인터넷서비스 책임연구원
관심분야 : 실시간 운영체제, 데이터베이스와 웹 연동, 멀티미디어 DBMS, 내장형 DBMS.

이 장 선

e-mail : sunny@sanux.com
1983년 경북대학교 전자공학과 (공학사)
1985년 한국과학기술원 전산학과 (공학석사)
1997년 미국 Syracuse University (공학박사)
1985년~2000년 한국전자통신연구원, 책임연구원
2000년~현재 인제대학교 정보통신공학과, 조교수
관심분야 : 멀티 입출력, Cluster file systems for SAN, 자료저장 시스템, 운영체제, 멀티미디어 시스템

송 영 기

e-mail : yksong@eagle.chonan-c.ac.kr
1977년 서울대학교 계산통계학과 (학사)
1981년 한국과학기술원 전산학과 (석사)
1977년~1985년 국방과학연구소 선임연구원
1985년~1999년 한국전자통신연구원 책임연구원
1993년~1994년 미국, 텍사스 알링톤 대학, 방문연구원
1999년~현재 친안외국어대학 산업전산과 교수
관심분야 : 데이터베이스, 소프트웨어공학, 정보보호, 이동 에이전트



박 진 섭

e-mail : jspark@drgon.taejon.ac.kr
1900년 대전대학교 전산지원센터장 (현)
1900년 대전대학교 컴퓨터전자통신공학부 교수(현)
1900년 중앙대학교 대학원 공학박사
1900년 연암공업대학 조교수
1900년 한국기계연구원 연구원
관심분야 : 컴퓨터 네트워크, 시스템 모델링 및 성능평가, 네트워크 보안, 정보보호정책.

김 명 준

e-mail : joonkim@etri.re.kr
1978년 서울대학교 자연과학대학 계산통계학과 (이학사),
1980년 한국과학기술원 전산학과 (이학석사),
1986년 프랑스 Nancy 제1대학교 응용수학 및 전산학과 (이학박사)
1980년~1981년 아주대학교 종합연구소 연구원
1981년~1986년 프랑스 Nancy 전산학 연구소(CRIN) 연구원,
1986년~현재 한국전자통신연구원, 선임/책임연구원, 컴퓨터·소프트웨어 기술연구소 소장
1993년~1993년 프랑스 Univ. of Nice Sophia-Antipolis 방문 교수
관심분야 : 데이타베이스, 분산 처리, 실시간 처리, 소프트웨어 공학

오 길 록

e-mail : groh@etri.re.kr
1968년 서울대학교 문리대 천문기상학과(학사)
1975년 한국과학기술원(KAIST) 산업공학과(석사)
1981년 프랑스 국립응용과학원(INSA) 컴퓨터공학과(박사)
1969년~1978년 한국과학기술연구원(KIST) 전산개발부 선임연구원
1978년~1987년 한국전자통신연구원 컴퓨터연구부장
1987년~1988년 한국전자통신연구원 행정전산망주전산기 개발 본부장
1988년~1996년 한국전자통신연구원 컴퓨터연구단장 및 정보기술개발단장
1996년~1998년 ETRI 부설 시스템공학연구소 소장
1997년~1999년 한국컴퓨터그래픽스학회 회장
1998년~2000년 한국전자통신연구원 컴퓨터·소프트웨어 기술연구소 소장
2000년~현재 한국소프트웨어컴포넌트컨소시엄 회장
2000년~현재 한국정보처리학회 회장
2000년~현재 한국전자통신연구원 연구위원