UniWeb 2.0 - 웹을 이용한 클라이언트-서버 데이타베이스 응용 개발 환경

김평철*

UniWeb 2.0 - A Web-based Client-Server Database Application Development Environment

Pyung-Chul Kim

〈요 약〉

웹을 이용한 클라이언트-서버 데이타베이스 시스템은 웹의 서비스 능력과 데이타베이스 시스템의 데이타 관리 기능을 상호 보완적으로 통합함으로써 인터넷과 같은 대규모 환경에서 데이타베이스 업무 환경을 구축하는 데 매우 적합한 것으로 알려져 있다. 데이타베이스 통로는 이러한 통합의 가장 핵심적 구성 요소이다. 본 논문에서는 먼저 클라이언트-서버 데이타베이스 응용을 위한 데이타베이스 통로의 고려사항으로서, 고성능 실행구조, 응용 프로그램 개발 환경, 그리고 상태 및 트랜잭션 관리에 대해 기술하고, 이어서 UniSQL/X용 데이타베이스 통로인 UniWeb 2.0의 설계와 구현에 대하여 소개한다. UniWeb 2.0은 CGI 응용 서버 방식을 채택하여 다양한 플랫폼을 지원하고, 고성능 그리고 확장성을 제공한다. 또한 프로그래머가 HTML 문서에 SQL/X문장이 포함된 Tcl 스크립트를 끼워 넣을 수 있도록하여 응용 프로그램 개발 생산성을 향상시키고 있다. UniWeb 2.0은 여러 웹 페이지에 걸친 상태 와 트랜잭션을 지원하고 있다.

* 충남대학교 정보통신컴퓨터공학부

1. 서론

은행의 예입/인출 업무, 항공사의 좌석 예약 업무 등으로 대표되는 클라이언트-서버 데이타베이스 시스템은 크게 세 개의 소프트웨어 모듈로 분리된다 [10]. 첫째는 데이타 관리 모듈로서, 특정 응용과 독립되어 데이타베이스의 구축, 검색 등을 지원한다. 일반적으로 DBMS(database management system) 엔진이 여기에 해당한다. 둘째는 응용 업무 모듈로서, 데이타 관리 모듈의 인터페이스를 통하여 각 업무의 데이타 처리 절차를 구현하고 있다. 세째로 사용자에게 대화형으로 자료를 입력 받고 결과를 출력하는 실연 모듈이다.

종래의 클라이언트-서버 데이타베이스 시스템을 인터넷과 같은 대규모 서비스에 환경에 적용하면 크게 두 가지의 문제가 발생한다. 첫째로 실연 모듈의 관리가 힘들다. 실연 사양(예를 들면, 화면 배치)은 클라이언트의 응용 프로그램에 코드화 되어 있거나 폼 개발 도구에 의해 작성된 폼 화일에 담겨져 있다. 즉, 응용 프로그램이 수행되거나 폼 화일이 번역되면서 필요한 자료를 서버의 데이타베이스에서 불러와 실연하게 된다. 클라이언트 수가 많거나 지리적으로 분산되어 있는 경우, 클라이언트에 관리되고 있는 실연 모듈을 변경하는 것은 매우 힘들고 많은 비용이 소요된다.

둘째로, 클라이언트-서버 데이타베이스 시스템에서 클라이언트 환경은 서버와의 통신 규약, DBMS와 접속하는 소프트웨어 그리고 실연에 사용되는 번역기 등 연결된 서버에 종속된다. 그런데, 일반적으로 클라이언트는 자신의 환경을 각각에 알맞도록 자율적으로 배치하는 경향이 있다. 예를 들면, 서로 다른 운영체제, 윈도우 시스템, 통신망 등을 이용할 수도 있다. 즉, 서비스 가입자의 환경이 모두 같다고 가정하기는 힘들다. 따라서, 클라이언트 서버 데이타베이스 시스템을 그대로 대규모 인터넷에 적용하면 지원할 수 있는 클라이언트 수에 제한을 받게 될 것이다.

월드 와이드 웹(World-Wide Web)은 스위스의 CERN에서 개발된 하이퍼미디어 방식의 대규모 정보 서비스 시스템으로서, 1992년 발표된 이후 인터넷을 중심으로 급속히 성장하여, 현재는 상업적인 응용에 사용되고 있기도 하다 [4]. 웹에서는 정보가 멀티미디어 자료나 다른 문서를 링크로 갖는 하이퍼미디어 문서로 표현된다. 클라이언트의 웹 브라우져는 HTML(hypertext markup language)로 쓰여진 개별 하이퍼미디어 문서를 번역하여 사용자 화면에 실연(presentation)한다 [5]. 이때, 각 멀티미디어 자료의 실연을 위해 적절한 외부 뷰어(external viewer)를 이용하기도 한다. 웹 브라우져는 사용자가 원하는 문서의 주소를 식별하여 해당 웹 서버를 연결한다. 전세계에 퍼져 있는 각 하이퍼미디어 문서 및 멀티미디어 자료는 고유의 식별자인 URL(uniform resource locator) [3]을 갖는다.

웹 서버는 클라이언트의 요구에 의해 화일 시스템에 저장된 하이퍼미디어 문서와 여기에 링크된 멀티미디어 자료를 전송한다. 이때, 웹 서버와 웹 브라우져 간에는 HTTP(hypertext transmission protocol) [2]라는 통신 규약이 적용된다. 웹 서버는 운영 체제의 일반 프로그램을 호출할 수 있는 CGI(common gateway

interface)를 지원한다[12]. 이는 검색, 위치 지정이 가능한 이미지, 폼(form), 그리고 동적으로 생성되는 하이퍼미디어 문서를 다루는 데 자주 이용된다. CGI는 웹 서버로부터 여러 가지 입력 인자를 환경 변수 형태로 받고, 프로그램 수행 결과를 HTML 문서로 변환하여 웹 서버에 전달한다.

웹에서 HTML 문서는 모두 서버에서 관리된다. 따라서, 서비스되고 있는 정보를 변경하는 것이 매우 용이하다. 또한, HTML, URL, HTTP 등의 중심 기술들이 이미 인터넷 표준으로 등록되어 있고, 이를 지원하는 상용 웹 브라우저가 광범위하게 출시되어 있기 때문에, 웹을 클라이언트-서버 데이타베이스의 실연 모듈의 관리에 적용하면, 앞에서 기술한 기존의 클라이언트-서버 데이타베이스 시스템을 인터넷에 적용할 때 발생하는 문제점이 해결될 수 있다.

즉, 모든 데이타는 데이타베이스 시스템에 의해 저장, 관리, 처리되고, 클라이언트측의 사용자들은 웹 기능을 통해 질의를 요구하고 또한 그 결과를 볼 수 있다. 데이타베이스 통로(database gateway)는 위와 같은 통합의 핵심 요소로서 웹으로 표현된 사용자의 요구를 DBMS와 연동하여 처리한다.

우리는 UniSQL/X 객체 관계 DBMS에 저장된 자료를 웹을 통하여 검색할 수 있는 기능을 지원하는 데이타베이스 통로인 UniWeb 1.0을 개발한 바 있다 [1]. 이를 클라이언트-서버 데이타베이스 응용 개발 환경으로 확장한 것이 UniWeb 2.0이다. UniWeb 2.0은 CGI 응용 서버 방식을 채택하여 다양한 플랫폼을 지원할 뿐 아니라 고성능, 높은 확장을 지원한다. 또한 프로그래머가 HTML 문서에 SQL/X문장이 포함된 Tcl 스크립트를 끼워 넣을 수 있도록 하여 응용 프로그램 개발 생산성을 향상시켰다.

다음 장에서는 클라이언트-서버 데이타베이스 응용을 위해 데이타베이스 통로를 설계하는 데 고려해야 하는 중요한 몇 가지 이슈와 이에 대한 기존의 방법을 설명한다. 3, 4장에서는 UniWeb 2.0의 실행 구조와 구현에 대해 자세히 설명한다. 마지막으로 5장에서는 이 논문을 요약하고 앞으로 계획을 기술한다.

2. 데이타베이스 통로의 고려 사항

2.1 실행 구조

데이타베이스 통로는 웹의 어느 위치에서 데이타베이스를 접근하는가에 따라 여러 구조로 분류되고, 각 구조에 따라 성능이 달라진다. 데이타베이스 통로의 구조는 우선 크게 **서버쪽 확장, 클라이언트쪽 확장**으로 나뉘어진다. 서버쪽 확장은 **CGI 이용 방식, 서버 API** (Application Programmers Interface) **이용 방식**, 그리고 **전용 서버 방식**으로 나뉘어진다. 클라이언트쪽 확장은 **외부 뷰어를 이용하는 방식**과, 브라우저가 데이타베이스를 직접 접속할 수 있도록 하는 **브라우저 확장 방식**으로 나뉘어진다. CGI 이용 방식은 데이타베이스 통로가 CGI 실행 화일 그 자체인 **CGI 실행 화일 방식**과 CGI가 적절한 데이타베이스 응용 서버에 사용자의 요구를 전달하는 **CGI 응용 서버 방식**으로 다시 나뉘어진다.

각 구조 분류의 기능 및 성능 측면의 특징과

개발 사례의 자세한 내용은 [1, 8]에서 이미 기술하였기 때문에 본 논문에서는 UniWeb 2.0의 구조를 이해하는 데 도움이 될 수 있는 CGI 응용 서버 방식에 대해서만 간략히 기술한다.

CGI 응용 서버 방식에서는 웹 표준기술인 CGI를 이용함과 동시에 DBMS의 최적화 기능을 활용하기 위해 데이타베이스 통로를 디스패처(dispatcher)와 데이타베이스 응용 서버로 나눈다. 이 구조에서 디스패처는 CGI 실행 화일로서 단순히 데이타베이스 응용 서버에 웹 서버의 요구를 전달한다. 데이타베이스 응용 서버는 데몬(demon) 프로세스로서, 데이타베이스 응용이 CGI 실행 화일 자체로 실행되는 CGI 실행 화일 방식과는 큰 차이가 있다.

디스패처는 데이타베이스 시스템과 관계없이 구현된다. 일단 웹 서버의 CGI로 생성된 디스패처는 웹 서버의 요구를 분석한 후, 그 요구를 처리할 수 있는 응용 서버를 찾아 전달한다. 응용 서버로부터의 결과는 HTML로 표현된다. 하나의 디스패처 프로세스의 크기는 대략 수십 킬로바이트로 매우 작기 때문에 대규모 서비스 환경에서도 실행 중인 모든 디스패처 프로세스의 총 크기는 문제가 되지 않는다.

CGI 실행 화일 방식과는 달리 데이타베이스 응용 서버는 CGI 환경과 관계없이 구현될 수 있다. 응용 서버는 DBMS와 접속한 후 디스패처로부터 요구를 기다린다. 특히, 하나의 요구를 처리한 후에도 프로세스가 종료되지 않고 새로운 요구를 기다린다. 따라서, 이 방식은 앞서 설명한 DBMS 최적화 기술의 이점을 충분히 이용할 수 있다.

대규모 서비스에서 응용 서버 프로세스의 수는 이용 가능한 시스템 자원에 따라 다를 수 있다. 이것은 동시 요구의 수와 데이타베이스 응용 프로세스의 수가 같은 CGI 실행 화일 방식과는 다르다.

2.2 응용 프로그램 개발 환경

데이타베이스 통로는 응용 프로그램 개발의 생산성을 극대화하기 위한 환경을 제공해야 한다. 현재 사용하고있는 응용 프로그램 개발 환경으로는 **일반적인 프로그램 언어 이용, 확장 HTML 태그 이용, 확장 스크립트 언어의 삽입 방법**으로 분류할 수 있다.

C/C++와 같은 일반적인 프로그램 언어를 이용할 경우에는, DBMS에서 제공하는 SQL이나 API를 삽입하여 응용 프로그램을 개발하게 된다. 프로그램 언어는 일반적으로 강력한 프로그래밍 구조체를 제공하며 일단 실행 코드로 컴파일되면 실행 속도가 매우 빠르다. 그러나 프로그램 언어를 배우고 이용하는 것이 어렵기 때문에 응용 프로그램 개발의 생산성이 떨어진다. 더구나, 컴파일된 실행 코드는 하드웨어나 운영체제에 따라 달라지므로 다양한 플랫폼을 지원하기 힘들다.

확장 HTML 태그는 기존의 HTML에 DBMS접속을 위한 태그 변수, 반복 구조, 제어 구조와 같은 몇 가지 프로그래밍을 위한 태그를 추가한 것이다. 이 방식에서는 확장 태그를 이용한 HTML 문서 작성이 바로 데이타베이스 응용 프로그램의 작성을 의미한다. 따라서, 응용 개발

환경이 매우 단순하기 때문에 배우기 쉽다. 그러나, 확장된 태그가 지원하는 프로그래밍 기능은 실세계의 응용을 개발하기에는 너무 많은 제약을 갖고 있다. 사용자가 입력한 값을 이용하여 SQL 문장을 형성하고 SQL 질의 수행 결과를 HTML로 변환하고, 특히 복잡한 업무 논리를 구현하기 위해서도 프로그래밍 언어의 거의 모든 기능이 필요하게 된다.

XQML [7], Cold Fusion [14], WebDBC [13], WebBase [15], Microsoft IDBC [20] 등이 각각 고유의 확장 HTML 태그를 지원하는 데이타베이스 통로이다.

일반 프로그램 언어를 이용할 때의 생산성 저하 문제를 해결하고 동시에 확장 HTML 태그 방식의 기능 제약 문제를 해결하기 위해 데이타베이스에 접속할 수 있도록 확장한 스크립트 언어를 HTML 문서에 직접 삽입하도록 할 수 있다. 스크립트 언어는 컴파일 방식의 프로그래밍 언어와 비교하여 높은 수준의 프로그래밍 구조체를 지원하기 때문에 배우고 사용하기 쉽다. 또한, 대부분의 스크립트 언어가 인터프리트 방식으로 수행되기 때문에 동적인 응용 프로그램 개발이 가능하여 높은 개발 생산성을 얻을 수 있다. 반면에, 기계어 수준의 실행화일을 생성하는 컴파일 방식에 비해서도 상대적으로 낮은 응답 시간을 초래한다.

Netscape LiveWire [19]는 ODBC와 몇 가지 대표적인 DBMS에 접속할 수 있는 확장된 JavaScript를 지원하고, WebStar[16]는 Tcl [11]을 확장하여 HTML 문서에서 CORBA 서비스를 접속할 수 있도록 하고 있다.

2.3 상태(state) 관리

여러 코너가 있는 전자 쇼핑 응용과 같이 웹을 이용한 많은 데이타베이스 응용에서 사용자가 여러 웹 페이지를 방문하면서 데이타베이스 트랜잭션을 형성하는 것이 일반적이다. 하나의 데이타베이스 트랜잭션에는 사용자와의 대화가 여러 번 있을 수 있고, 이때 각 대화는 하나의 웹 페이지에 대한 요구에 해당하게 된다.

여러 대화로 구성된 트랜잭션을 지원하기 위해서는 트랜잭션 내의 연속적인 요구들간의 정보를 관리할 필요가 있다. 이 정보가 바로 상태(state)이다. 그러나 현재 HTTP가 비상태보존(stateless) 프로토콜을 이용하고 있기 때문에 각 요구는 그 이전의 요구와는 독립된 환경에서 처리된다. 따라서 웹에서 요구간의 상태를 관리하는 것은 간단하지 않다.

웹에서 상태 관리에 관련된 이슈들은 다음과 같다.

- 상태 유지 : 데이타베이스 통로는 앞선 요구의 수행 결과를 이후의 요구에 의해 참조될 수 있도록 유지하여야 한다.
- 상태 해제 : HTTP가 비상태보존 프로토콜이므로 일반적으로 웹 서버는 언제 상태가 해제되어야 할지를 알지 못한다. 그러므로 데이타베이스 통로는 적절한 방법을 이용하여 불필요한 상태를 해제할 수 있어야 한다.
- 상태 보호 : HTTP는 두 개의 연속적인 요구

사이에 어떤 연결도 제공하지 못하기 때문에 원하지 않는 사용자가 다른 사용자의 상태를 악용할 수도 있다. 데이타베이스 통로는 한 사용자의 상태를 다른 사용자로부터 보호할 수 있어야 한다.

- 비정상 상태 전이 방지 : 웹 브라우저는 전진(forward), 후진 (backward) 탐색 방법으로 지역 캐쉬에 저장된 이력(history) 정보를 탐색할 수 있다. 이때, 캐쉬에 복사본이 없을 경우 이력 정보 탐색은 다시 웹 서버에게 새로운 요구를 하게 되고 이는 원하지 않는 상태 전이를 초래할 수 있다. 데이타베이스 통로는 이런 비정상 상태 전이가 발생하지 않도록 해야 한다.

상태는 클라이언트쪽에 유지하게 할 수 있다. 이 경우, 앞선 요구의 상태가 클라이언트로 보내지고, 클라이언트는 다음 요구에 이전 상태 정보를 같이 넘겨준다. 상태 정보는 URL의 일부분 또는 Cookie[18]를 이용하여 전송할 수 있다. 클라이언트쪽에 상태를 유지하는 방법은 불필요하게 통신량을 증가시킨다. 최악의 경우 상태 정보의 양이 최대 요구 크기보다 커질 수도 있기 때문에 적용할 수 있는 응용에 제한이 있다.

이에 대한 대안으로, 서버쪽에서 상태 정보를 유지할 수 있다. 상태 정보는 주기억 장치나 혹은 디스크 화일에 저장한다. 저장된 상태 정보로 각각 유일한 상태 식별자를 갖고 클라이언트의 요구를 참조하고자 하는 상태 식별자를 요구와 같이 넘겨준다.

이때, 상태 식별자는 URL의 일부를 Cookie를 이용하여 전송할 수도 있고, 클라이언트의 IP주소를 바로 상태 식별자로 이용할 경우도 있다. 서버쪽에서 상태를 유지하는 경우에는 상태 정보의 크기에 제한 받지는 않으나 데이타베이스 통로의 구현이 복잡해진다.

불필요한 상태의 해제는 일반적으로 시간제한(time-out) 방식을 이용한다.

상태 보호를 위해서는 Cookie를 이용하거나, 난수를 발생하여 상태 식별자를 생성하거나, 혹은 클라이언트의 IP주소를 이용할 수 있다. 비록 Cookie를 이용하는 방법이 상대적으로 안전하기는 하지만 어떤 방법도 상태의 악용을 완전하게 차단하지는 못한다.

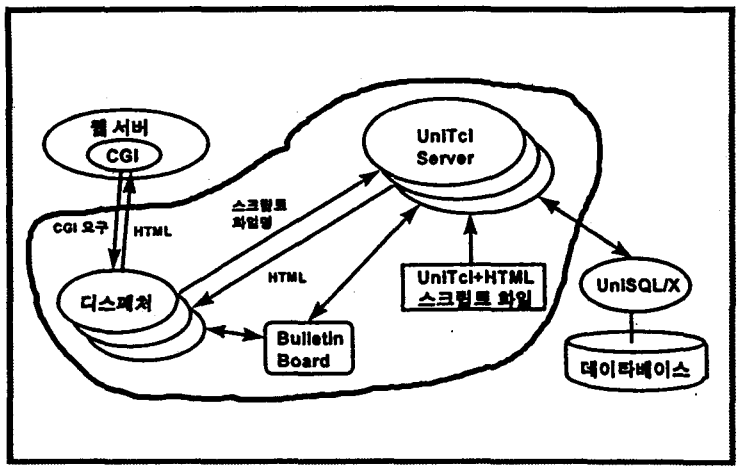
이력 정보 탐색에 의해 발생되는 비정상 상태 전이를 막기 위해 서버는 각 요구마다 이것이 현재의 웹 페이지에서 발생되었는지를 확인해야 한다. 한 가지 해결책은 각 요구마다 단조증가하는 순차 번호를 부여하여, 만약 후진 탐색을 할 경우 서버가 감지할 수 있도록 하는 것이다. 그러나 이를 감지했다 하더라도, 이를 잘못된 요구로 처리할 것인지, 데이타베이스 트랜잭션의 부분 철회로 간주할 것인지를 판별해야 한다. 이에 대해서는 더 연구가 필요하다.

Netscape LiveWire [19]는 위에서 소개했던 상태 유지, 해제, 보호에 대한 모든 대안을 지원하고 있다. WebObject [17]는 각 요구에 순차 번호를 부여하여 이력 정보 탐색을 막을 수 있도록 하고 있다.

3. UniWeb 2.0의 실행구조

UniWeb 2.0은 UniSQL /X용 객체 관계

DBMS와 웹을 CGI 응용 서버 방식으로 통합한 데이타베이스 통로로서 디스패처, 게시판 (bulletin board), UniTcl 서버, 그리고 UniTcl 스크립트 화일로 구성되어있다. 〈그림 1〉은 UniWeb 2.0의 실행 구조를 보인다.



〈그림 1〉 UniWeb 2.0의 실행 구조

디스패처는 CGI 실행 화일이다. 사용자의 요구가 발생했을 때, 웹 서버는 프로세스 파이프를 통해 각 요구를 위한 디스패처를 생성한다. 디스패처는 게시판을 참조하여 요구를 처리할 수 있는 UniTcl 서버를 찾아서 그 요구를 전달한다. 디스패처는 GET, POST 방법을 모두 지원한다.

게시판은 UniTcl 서버들의 정보를 갖고 있으며, 디스패처 프로세스들과 UniTcl 서버들에 의해 상호 배타적인 방법으로 공유된다. 게시판은 Unix System V IPC의 공유 메모리 기법으로 되었다. 게시판에는 각 UniTcl 서버에 대해 다음과 같은 정보를 유지한다.

- 프로세스 정보: 프로세스 식별자, 프로세스 크기 등.
- 서버 상태: 서버가 현재 요구를 처리 중인지를 나타내는 플래그, 서버가 현재 관리하고 있는 상태 식별자 등.
- 통신 채널 정보: 서버와의 통신을 위한 소켓 (socket) 이름.
- 통계 정보: 시작 시각, 처리 건수 등.

UniTcl 서버는 UniSQL/X 엔진과 접속하는 응용 서버로서 요구된 스크립트 화일을 적재하고 그것을 수행하여 HTML 화일로 변환한다. 스크립트 화일은 UniSQL/X를 접속할 수 있도록 Tcl을 확장한 UniTcl 스크립트가 삽입된 HTML 문서이다.

UniTcl 서버의 구현에 대해서는 다음 장에서 자세히 설명한다. UniTcl 서버는 스크립트 화일을 해석하면서 UniSQL/X에 접속한다. 또한, 상태와 트랜잭션을 관리하며 Unix 소켓을 통해 디스패처와 접속한다. 관리자는 시스템의 부하에 따라 UniTcl 서버의 수를 적절히 조절할 수 있다.

UniTcl 서버는 데몬 프로세스로서 일단 하나의 UniSQL/X 데이타베이스와 접속하면, 다른 UniSQL/X 데이타베이스가 요구될 때까지 접속을 계속 유지하기 때문에 UniSQL/X 엔진과의 반복적인 접속을 피할 수 있고, UniSQL/X 질의 최적화와 객체 캐쉬의 장점을 충분히 활용할 수 있다.

다음은 UniWeb 2.0에서 사용자 요구를 처리하는 단계를 보여주고 있다.

1. 아래와 같은 형식의 사용자 요구가 웹 서버에게 전달된다.

   〈디스패처 경로〉/〈스크립트 화일명〉[/〈상태 식별자〉]?QS

   〈디스패처 경로〉는 디스패처의 실행을 위한 CGI 경로이다. 〈스크립트 화일명〉은 요구된 UniTcl 스크립트 화일의 이름이다. 〈상태 식별자〉는 참고하고자 하는 상태 식별자로서, 새로운 상태를 시작할 때는 필요 없다. QS는 HTML의 FORM 태그를 통해 입력한 질의 값으로서 환경 변수인 QUERY_STRING에 저장된다.
2. CGI 실행 화일인 디스패처 프로세스가 생성되고 프로세스 파이프를 통해 웹 서버로 부터 요구를 받는다.
3. 디스패처는 게시판을 참조하여 해당 요구를 처리할 수 있는 UniTcl 서버를 찾아 소켓을 통해 요구를 전달한다.
4. UniTcl 서버는 UniSQL/X 엔진을 접근하여 요구된 스크립트 화일을 번역한다. 또한 게시판에 자신의 실행 상태를 기록한다. 스크립트 화일의 실행 결과는 HTML 문서(보다 일반적으로는 MIME [6] 자료)이다. UniTcl 서버는 실행 결과를 소켓을 통해 디스패처에 전달한다.
5. 디스패처는 최종결과를 웹 서버에 전달하고 종료된다.

4. UniTcl의 구현

4.1 왜 Tcl인가?

UniTcl은 UniSQL/X를 접근하기 위한 Tcl을 확장한 스크립트 언어이다. 우리는 다음과 같은 이유로 Tcl 스크립트 언어를 선택하였다.

- Tcl은 간단하지만 강력한 스크립트 언어로서, 변수, 제어 구조, 사용자 정의 프로시져 등과 같은 필수적인 프로그래밍 기능을 가지고 있다.
- Tcl 인터프리터는 그 자체가 C 언어에 의해 호출될 수 있기 때문에 우리가 개발하고자 하는 인터프리터에 Tcl 인터프리터를 포함시키는 것이 매우 간단하다.
- 기능 확장이 용이하다. 데이타베이스 접근을 위한 새로운 명령어를 기존의 Tcl 인터프리터에 첨가하는 것이 쉽다. 특히, 새로운 명령어를 C 언어로 구현할 수 있다.
- JavaScript, VBScript 등과 달리, Tcl은 소스 수준의 공용 소프트웨어이다.
- Unix, Windows 시리즈, 맥킨토시 등의 다양한 플랫폼에서 사용 가능하다.
- 1990년 처음으로 발표된 이후 많은 사용자 그룹이 확보되어 있다.
- 웹 환경의 응용 개발에 가장 적합한 언어 중 하나로 부각되고 있다. 예를 들면, Tcl/tk 플러그인(plug-in)이 이미 개발되었다 [21].

4.2 UniSQL/X 접근을 위한 Tcl의 확장

〈그림 2〉는 UniTcl의 구성 요소를 보여주고 있다. UniTcl은 Tcl 인터프리터의 확장성을 충분

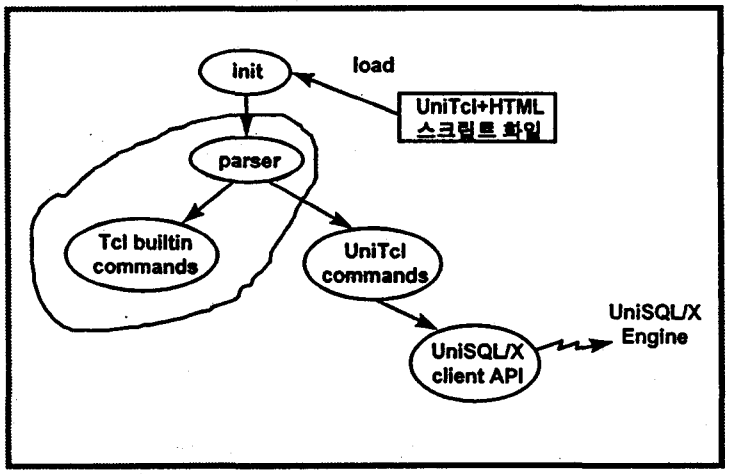
히 활용하여 구현하였다. UniTcl 인터프리터는 HTML 화일로부터 UniTcl 스크립트를 뽑아내어 Tcl 파서에게 전달하고, Tcl 파서는 내장 명령어(builtin command)와 확장된 UniTcl 명령어를 이용하여 스크립트를 해석한다.

UniTcl 명령어는 UniSQL/X API 함수를 통하여 UniSQL/X DBMS를 접근한다. 각 UniTcl 명령어는 C 언어로 구현되어 Tcl 파서에 등록되어 있다. 응용 프로그래머는 이와 같은 방식으로 응용에 필요한 부가적인 명령어를 추가할 수도 있다.

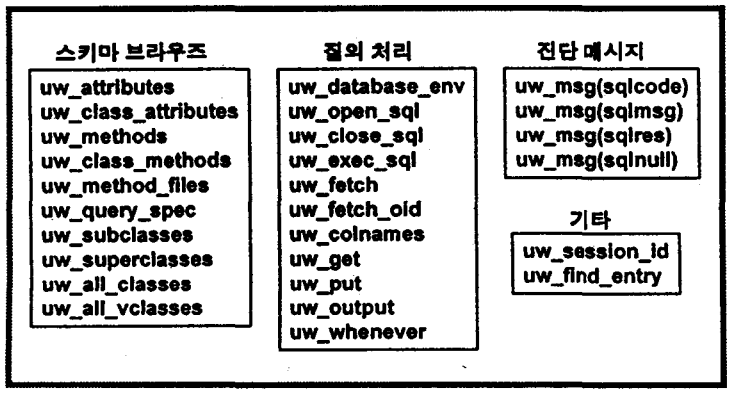
UniTcl 명령어는 UniSQL/X DBMS를 접근하기 위한 기능과 (예, 스키마 브라우즈, SQL/X 문 처리, 오류 진단 등) 상태를 관리하는 기능을 모두 지원하고 있다. 〈그림 3〉은 UniTcl 명령어를 요약하고 있다.

UniTcl은 삽입 SQL (Embedded SQL) 언어에서 널리 사용되는 커서(cursor) 개념을 구현하고 있다. 다음은 UniTcl 커서를 이용한 예이다.

```
set sql "select * from person"
set sql_handle [uw_open_sql $sql]
set tuple [uw_fetch $sql_handle]
```



〈그림 2〉 UniTcl 구성 요소



〈그림 3〉 UniTcl 명령어 요약

변수 sql_handle은 UniTcl 커서를 나타내는데, 이는 내부의 커서 구조체를 식별하는 Tcl 문자열을 갖도록 구현되었다. 내부의 커서 구조체는 커서 식별자, SQL/X의 결과에 대한 포인터, 그리고 기타 관리 정보로 이루어진다. 위의 예제에서, sql_handle은 uw_open_sql 명령어로 생성된 후, uw_fetch 명령어에 의해 결과 튜플(tuple)을 가져오는데 사용된다.

사용자는 Tcl의 문자열 조작 기능을 이용하여 임의의 SQL/X 문장을 만들 수 있다. 질의 결과 또한 Tcl 문자열로 표현된다. 특히, 튜플은 Tcl 리스트로 표현되는데, 리

스트의 각 문자열은 하나의 속성값을 나타낸다.

UniSQL/X는 문자, 비트 문자열, 수치, 시간, 집합, 객체 등, 다양한 기본 자료형을 제공하는 반면, Tcl은 오직 문자열만 지원한다. 따라서, Tcl에서 데이타베이스 결과를 사용할 수 있도록 하기 위해서는 UniSQL/X 자료형을 Tcl 문자열로 변환할 필요가 있다.

UniSQL/X의 문자, 수치 등을 Tcl 문자열로 변환하는 것은 간단하다. 다만, 어떤 값에 대한 표현 형식이 여러 가지 있는 경우에는 시각적으로 가장 적절하고 본래 값의 정보 손실이 없는 표현 형식을 채택하였다.

응용 프로그램 측면에서 볼 때 UniSQL/X 객체 자료는 메모리에 복사된 해당 객체의 포인터 값을 갖는다. 이 포인터 값은 웹의 연속되는 요구에서 항상 유효하다고 볼 수는 없다. 예를 들면, 하나의 요구를 수행하면서 복사된 객체가 다음 요구가 발생하기 전에 쓰레기 수집기(garbage collector)에 의해 삭제될 수 있고, 이 경우 앞 요구에서 얻은 포인터를 이용하여 해당 객체를 다시 접근할 수 없게 된다. 그러나, 웹과 같이 하이퍼텍스트 방식으로 자료를 검색하는 응용에서는 연속되는 요구들 사이에 같은(혹은 연결된) 객체를 참조하는 경우가 매우 빈번하다.

많은 복합 객체(composite object)로 이루어진 객체 관계 데이타베이스를 웹을 통해 접근하는 경우를 생각해보자. 각 복합 객체는 관련된 다른 객체에 대한 참조를 가지고 있을 것이다. 이러한 객체 참조는 HTML 페이지의 앵커(anchor)로 자연스럽게 표현될 수 있다. 그리고, 사용자는 이러한 앵커의 호출을 통해 하이퍼텍스트 방식으로 복합 객체를 검색할 수 있게 된다. 한편, 이 방식을 이용하면 간단하지만 강력한 하이퍼텍스트 데이타베이스 브라우져를 구현할 수 있다.

일반적으로 객체의 접근은 SQL을 이용하는 것보다 객체 참조를 이용하는 것이 훨씬 빠르다. 특히 UniSQL/X는 한번 접근한 객체를 응용 프로세스의 메모리에 저장하여 같은 객체의 반복적인 접근시에 매우 높은 성능을 지원한다 [9].

UniTcl에서는 연속되는 요구에서 객체 참조가 재사용될 수 있도록 하기 위해 UniSQL/X 객체 포인터 대신 객체 식별자를 사용하였다. 즉, 한 객체 참조가 HTML의 앵커로 표현될 때,해당 객체의 식별자를 Tcl 문자열로 변환하여 나타낼 수 있도록 하였다. 객체 식별자는 객체 포인터와는 달리 데이타베이스 내에서 유일하고 영구적이다.

UniSQL/X의 집합형 자료는 Tcl 리스트로 변환된다. 집합의 각 원소 값은 자료형에 따라 앞에서 언급한 방법을 적용하여 역시 Tcl 문자열로 변환된다. Tcl 리스트는 기존의 Tcl 내장 명령어를 이용하여 다양하게 조작될 수 있다.

데이타베이스 결과에서 각 속성 값이 널(null)인지 아닌지는 UniTcl의 결합변수(associative variable), uw_msg(sqlnull)를 통하여 판별될 수 있다. 즉, 하나의 튜플이 uw_fetch에 의해 추출되면, 각 속성의 널 여부를 나타내는 값이 uw_msg(sqlnull) 리스트에 채워진다. 이 리스트의 한 원소 값이 음수 값이면 이에 해당하는 속

성 값이 널임을 뜻한다. 또한, uw_msg는 일반적인 내포 SQL 언어의 SQLCA(SQL Communication Area)의 용도로도 사용된다. 즉, uw_msg (sqlcode)와 uw_msg (sqlmsg)는 각각 데이타베이스를 접속하여 얻은 반환 코드와 오류 메시지를 갖고 있다.

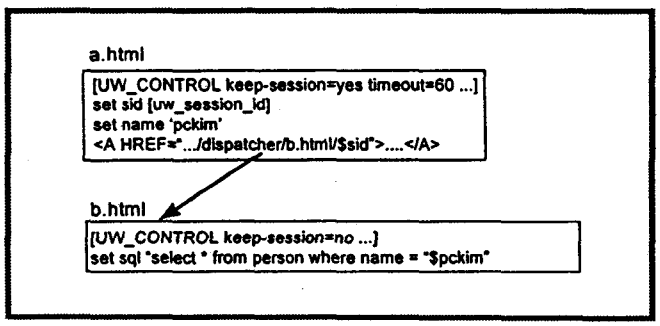
<그림 4>는 UniTcl을 삽입한 HTML 문서의 예제를 보인 것이다. 개발자는 표준 HTML 문서를 작성하되, 필요할 경우, [와]를 사이에 임의의 UniTcl 명령어를 삽입할 수 있다. 이 예제에서는 SELECT 문과 커서를 이용하여 데이타베이스 저장된 객체를 표형식으로 출력하고 있다.

```
<BODY>
[ uw_database_env testdb
  set class [uw_find_entry class]; set sql "select * from $class"
  set handle [uw_open_sql $sql]
  uw_output $handle; # use default TABLE format output
]
<P><TABLE BORDER=1>
[ uw_cursor first $handle;          # TABLE output using iteration
  while {$uw_msg(sqlcode) == 0} {
    set tuple [uw_fetch $handle]
    puts "<TR>"
    foreach col $tuple {
      puts "<TD>$col"
    }
    uw_cursor next $handle
  }
  uw_close_sql $handle
]
</TABLE>
```

<그림 4> UniTcl이 삽입된 HTML 예제

4.3 상태 관리

<그림 5>는 두 개의 UniTcl 스크립트가 연속하여 요구되는 예를 보여준다. name은 a.html에 의해 생성된 Tcl 변수이고 b.html에서 참조된다. 이러한 경우, b.html에 대한 요구에 a.html에서 생성된 상태 식별자($sid)를 결부시켜, b.html이 a.html에 의해 생성된 상태를 참조한다는 것을 나타내게 된다. UW_CONTROL 문은 스크립트 화일의 처음에 나타날 수 있고, 상태의 유지 및 해제에 대한 제어 정보를 기술한다. 이 예제에서 a.html의 수행 상태는 유지되고, b.html의 상태는 유지되지 않고 있다.



<그림 5> 상태 사용의 예

연속적인 스크립트 화일 요구가 같은 상태 식별자로 연결되면, 한 스크립트 화일에서 생성된 어떤 데이타라도 다음 스크립트 화일에 의해 참조될 수 있다. 상태 정보는 Tcl 구조체와 데이타베이스 정보로 구성된다. Tcl 구조체는 변수와 프로시져 등을 포함하고, 데이타베이스 정보는 현재 사용 중인 데이타베이스 접속 정보, 트랜잭션, 그리고 커서 정보 등을 포함한다.

모든 상태 자료는 UniTcl 내부 메모리에 유지된다. 그러나 UniTcl이 상태 자료를 저장하기 위해 특별히 구현하고 있는 자료 구조는 없다. 즉, 상태 자료 중 Tcl 정보는 Tcl 인터프리터에 의해 자동으로 관리된다. Tcl 인터프리터는 Tcl 변수와 프로시져의 빠른 탐색을 위해 메모리 해쉬 테

이블을 구현하고 있다. 한편, 상태 자료 중 데이타베이스 정보는 UniSQL/X 라이브러리를 통해 관리된다.

현재 버전의 UniSQL/X 라이브러리는 하나의 응용 프로세스가 동시에 여러 데이타베이스를 접속하거나 두 개 이상의 트랜잭션을 관리할 수 없는 제한을 가지고 있다. 이러한 제약 때문에 UniTcl 서버는 한 순간에 하나의 상태만 관리할 수 있도록 구현되어 있다. 즉, UniTcl 서버가 하나의 그 상태가 종료될 때까지 다른 상태를 서비스할 수 없다.

요구를 처리하면서 새로운 상태가 생성되면 UniTcl 서버는 그 상태 식별자를 게시판에 기록한다. 그리고, 디스패처는 사용자 요구에 상태 식별자가 결부되어 있을 경우, 이를 관리하고 있는 UniTcl 서버를 게시판에서 찾아 그 요구를 전송하게 된다. 상태 식별자가 결부되지 않은 요구는 현재 상태를 관리하고 있지 않는 UniTcl 서버 중 하나에 전송된다.

생성된 상태는 사용자의 명시적인 요구에 의해 해제될 수도 있으나, 주어진 시간 동안 더 이상 요구가 없을 경우에는 시스템이 강제로 해제시킬 수도 있다. 상태가 강제로 해제되는 경우에는, 이 상태에 결부된 Tcl 구조체와 사용 중인 커서들은 삭제되고 트랜잭션은 복귀(rollback)된다.

4.4 트랜잭션 지원

UniSQL/X는 SQL/X 명령어 수준에서 순서화 가능한(serializable) 데이타베이스 트랜잭션을 제공한다. UniTcl 명령어는 UniSQL/X의 모든 기능을 포함하고 있기 때문에, 사용자는 UniTcl 스크립트로 작성된 하나의 웹 페이지 안에 UniSQL/X 트랜잭션을 자유롭게 삽입할 수 있다. 또한, UniWeb 2.0의 상태 관리 기능을 이용하면 여러 웹 페이지에 걸치는 데이타베이스 트랜잭션도 발생시킬 수 있다.

UniWeb 2.0의 트랜잭션 관리 기능은 다음과 같다.

- 하나의 상태에 복수의 트랜잭션을 지원한다.
- 하나의 UniTcl 스크립트 화일에 복수의 트랜잭션을 포함시킬 수 있다.
- 하나의 트랜잭션이 복수의 UniTcl 스크립트 화일에 걸쳐 생성될 수 있다.
- 완료(commit), 복귀(rollback), 저장점(savepoints), 분리도(isolation level)와 같은 UniSQL/X의 트랜잭션 관련 모든 기능을 사용할 수 있다.

5. 결론

웹을 이용한 클라이언트-서버 데이타베이스 시스템은 웹의 서비스 능력과 데이타베이스 시스템의 데이타 관리 기능을 상호 보완적으로 통합함으로써, 인터넷과 같은 대규모 환경에서 데이타베이스 업무 환경을 구축하는데 매우 적합한 것으로 알려져 있다. 데이타베이스 통로는 이러한 통합의 가장 핵심적 구성 요소이다. 현재, 많은 데이타베이스 통로가 공용 또는 상용 제품으로 이미 구현되어 있다.

이 논문에서는 클라이언트-서버 데이타베이스

응용에 적합한 데이타베이스 통로를 개발하는 데 고려해야 하는 이슈로서, 고성능 실행 구조, 응용 개발 환경 그리고 상태 관리 등에 대해 현재까지 알려진 해결 방법을 검토하였다.

또한, 우리는 최근에 개발한 UniSQL/X DBMS와 웹을 통합하는 데이타베이스 통로인 UniWeb 2.0의 설계 및 구현에 대하여 상세하게 기술하였다. UniWeb 2.0은 UniWeb 1.0을 클라이언트-서버 데이타베이스 응용 개발 환경으로 확장한 것으로서, CGI 응용 서버 방식을 채택하여 고성능, 확장성 그리고 다양한 플랫폼에 대한 적용성을 지원하고 있다. UniWeb 2.0에서는 UniTcl 스크립트를 HTML에 직접 삽입함으로써 응용을 개발할 수 있도록 하고 있다. 이는 기존의 C/C++ 언어를 이용하거나 확장 태그를 이용하는 방식에 비하여 높은 생산성을 지원한다. UniTcl은 UniSQL/X DBMS를 접속할 수 있도록 Tcl 스크립트 언어를 확장한 것이다. UniTcl 인터프리터는 응용 서버로서 동작하며 웹 요구에서 발생하는 상태와 데이타베이스 트랜잭션을 관리한다. UniWeb 2.0은 1996년 7월에 개발 완료되었고, 현재 Solaris 2.4 이상에서 작동한다. 아울러, UniWeb 2.0 관련 자료들은 http://grigg.chungnam.ac.kr/~uniweb 에서 찾을 수 있다.

현재 우리는 다음과 같은 측면에서 UniWeb 2.0을 확장하고 있다.

- 웹 응용에 적합한 새로운 트랜잭션 관리 기법: 이 기법은 웹 페이지에 기반을 둔 새로운 트랜잭션 정확성 기준(correctness criteria)을 제시할 것이고, 웹 환경에서의 트랜잭션 프로그래밍 지침으로 활용될 것이다.
- 이력 정보 탐색으로 인한 비정상 상태 전이를 해결할 수 있는 보다 포괄적인 상태 관리 방법.
- 대표적인 웹 서버들의 API 지원.
- 다른 DBMS 지원.
- 성능 감시 및 조정을 위한 시각적인 인터페이스.

〈참고문헌〉

[1] 김평철, "UniWeb - 웹을 위한 UniSQL/X 데이타베이스 통로", 데이타베이스 저널, 제3권 1호, 1996, pp. 65-84.

[2] T. Berners-Lee, Hypertext Transfer Protocol - HTTP/1.0, Internet RFC 1945, May 1996.

[3] T. Berners-Lee, Uniform Resource Locators, Internet RFC 1738, Dec. 20, 1994.

[4] 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.

[5] T. Berners-Lee and D. Connolly, Hypertext Markup Language Specification - 2.0, Internet RFC 1866, Nov. 1995.

[6] N. Borenstein and N. Freed, MIME (Multipurpose Internet Mail Extensions) Part One: Mechanisms for Specifying and Describing the Format of Internet Message Bodies, Internet RFC 1521, Sep. 1993.

[7] R. Chiang, XQML: A Gateway for WWW and RDBMS, Technical Document, Infobahn Computing Technology

Co., Ltd.,1995.

[8] P.-C. Kim, "A Taxonomy on the Archtecture of Database Gateways for the Web", *Proc. of the 13th Intl Conf. on Advanced Science and Technology (ICAST97)*, Apr. 1997. (to appear).

[9] W. Kim, "Object-Oriented Database Systems: Promises, Reality, and Future", *Modern Database Systems*, Addison Wesley, 1995, pp. 255-280.

[10] 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.

[11] J. Ousterhout, *Tcl and the Tk Toolkit*, Addison-Wesley, 1994.

[12] D. Robinson, The WWW Common Gateway Interface Version 1.1, Internet draft, Jan. 1996.

[13] URL: http://www.ndev.com/ndc.

[14] URL: http://www.allaire.com.

[15] URL: http://www.webbase.com/docs/WB17/HowWorks.htm

[16] URL: http://webstar.cerc.wvu.edu/

[17] URL: http://www.next.com/WebObjects/

[18] URL: http://www.illuminatus.com/cookie

[19] URL: http://home.netscape.com/comprod/server_central/product/livewire/livewire_datasheet.html

[20] URL: http://www.microsoft.com/infoserv/docs/PROGRAM.HTM#12632

[21] URL: http://www.sunlabs.com/research/tcl/