.NET기반의 저부하형 웹 애플리케이션 설계 및 구현

최동우, 안현식
동명정보대학교 로봇시스템공학과

Design and Implementation of light loaded Web Application based on .NET

Dong-Woo Choi, Hyunsik Ahn
Dept. of Robot System Engineering, Tongmyong Univ. of Info. Tech.

요 약

온라인상에서의 혼합형 서비스의 추세와 함께 보다 많은 정보들이 처리됨에 따라 웹 애플리케이션의 경우 서버의 부하를 가능한 줄이고, 처리속도를 향상시키기 위한 필요성이 대두되고 있다. 본 논문에서는 이러한 웹 애플리케이션의 문제점을 분석하고, 최근에 등장한 .NET 기반의 저부하형 웹 애플리케이션 설계 및 구현 방법을 제안한다. 다중 접속시의 서버의 부하를 최소화하기 위하여, 관계형 데이터베이스를 설계하고 최소한의 모듈을 생성하였으며, SP를 이용하여 복잡한 SQL문을 단순화 하여 데이터베이스의 부담을 줄여 서버의 처리능력을 향상 시켰다. 또한 User Control을 활용하여 페이지를 구성하여 페이지 처리 속도를 향상시켰으며, 스크립트를 활용하여 서버 언어를 이용한 작업을 최소한으로 줄였다. XML/EDI를 이용한 전자문서교환 방식을 채택하고 관리비용을 줄일 수 있도록 하였다. 본 논문에서는 제안한 저부하형 시스템을 학사관리 시스템 상에서 구현하여 기존의 방법에 비해 보다 효율적 시스템임을 보인다.

1. 서론

온라인상에서의 글로벌 서비스의 추세와 함께 보다 많은 정보들이 온라인상에서 송수신되고 있으며 보다 체계적인 관리 및 유지가 요구되고 있다. 웹을 기반한 애플리케이션의 경우 서버의 부하와 호출 속도를 줄이기 위한 노력이 이루어지고 있다.[1-3] 특히 많은 사용자들이 동시에 접속하면 애플리케이션이 동시에 실행이 되므로 서버의 부하가 커지며, 다양한 기능을 지원하기 위해 지나치게 많은 모듈을 생성하면 한 모듈의 실행시에 파생하는 모듈이 연이어서 실행되게 되므로 역시 부하를 가중시키며 처리 속도가 저하되는 문제가 존재하게 된다.

본 논문에서는 이러한 웹 애플리케이션의 문제점을 구체적으로 분석하고, 최근에 등장한 .NET 기반으로 다중 접속시의 서버의 부하를 최소화하며 최소한의 모듈 생성을 위한 설계 방법을 제안하고, 이를 학사관리 시스템 상에서 구현하여 보다 효율적 시스템임을 보인다. 제안한 웹 애플리케이션 시스템은 최근에 MS에서 개발한 ASP.NET을 기반으로 한다. 기존의 ASP (Active Server Page)는 사용자가 접속할 때 마다 서버에서 컴파일과정을 통해서 접속자에게 결과값을 넘겨주는데 비해 ASP.NET은 처음 요청이 있으면 컴파일하고, 파일이 변경되기 전까지는 그 다음의 요청이 있을 경우 이전에 컴파일 된 내용을 실행한다. 따라서 서버에서 접속자의 의한 파일요청에 대한 응답 속도를 향상시킬 수 있다.[11] 또한 DB는 관계형 DB로 설계하였는데 저장 프로시저(Stored Procedure)를 이용하여 복잡한 SQL 문을 단순화 시켜주고, 보안성을 높임과 동시에 DB의 부담을 줄여 주므로 서버의 성능을 향상시켰다. 또한 본 논문에서는 XML/EDI을 이용하여 전자문서화 함으로서 기존의 EDI는 변환 소프트웨어가 필요했던 단점을 개선하고, 인터라넷 방식이 아닌 웹으로 서비스함으로써 설치 비용절감과 함께 공간의 제약을 줄였다.

본 논문의 구성은 2장에서는 본 논문에서 제시한 웹 애플리케이션 시스템의 설계를 자세히 서술하였고, 3장에서는 제안한 시스템을 학사관리 시스템에 적용한 결과와 고찰을 언급하고, 4장에서 결론을 맺는다.

2. 저부하형 웹애플리케이션의 설계 방법

2.1 관계형 데이터베이스 설계

본 논문에서는 데이터베이스 설계를 관계형 데이터베이스를 설계하였다. 관계형 데이터베이스 설계는 정규화를 거친 후 비정규화를 통해 데이터 모델링을 실시하는 방법으로서 테이블의 컬럼을 수정하고자 할 경우 빠른 대처를 할 수 있으며 재사용성을 높일 수 있다. 그림 1은 학사관리 시스템을 예로 들어 웹 애플리케이션의 데이터베이스의 개념설계인 ER(Entity Relationship) 모델링의 결과를 보여주고 있다.

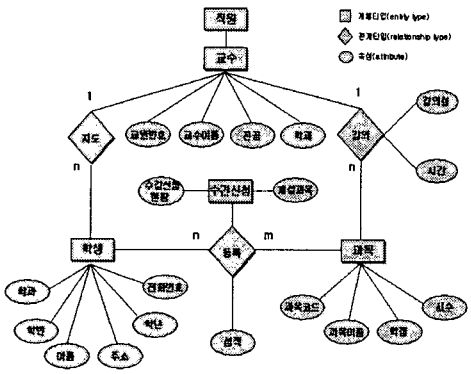


그림 1. 관계형 데이터베이스 개체 모델 구성

먼저 정규화 과정을 통해 그림 2와 같이 불필요한 테이블의 생성을 줄이고 중복성이 없는 컬럼으로 구성된 테이블을 설계하였다. 중복성이 필요한 컬럼에는 PK(Primary Key)를 지정하였으며, 연관성 있는 테이블 간의 컬럼들의 관계를 만들기 위해서 FK(Foreign Key)를 생성하였다. 예를 들어 '과목'에서는 PK와 FK를 동시에 설정하고, 수강에서는 FK로 설정하였다. 만약 수강에서 임의의 과목코드를 입력시 제약조건에 의해서 입력된 과목코드가 과목의 과목코드에 존재하는지를 체크하고 그 과목이 존재하지 않으면 True를 반환됨으로 그림 3(a)와 같이 웹서버와 데이터베이스연결이 하나의 과정으로 종료된다. 그러나 이러한 설정이 없을 경우 임의로 입력된 과목코드가 과목의 과목코드에 존재여부 체크를 위한 별도의 과정이 필요하게 되므로, 그림 3(b)와 같이 연결과정 늘어나게 된다. 따라서 데이터베이스의 규모가 줄게 되며 데이터의 처리속도가 향상됨으로서 저부하형 처리를 더욱 용이하게 할 수 있다.

2.2 저장프로시저 구성

본 논문에서는 저장프로시저를 이용하여 복잡한 SQL문을 단순화함으로써 네트워크 트래픽을 줄인다. 그림 4(a)는 저장프로시저 생성구문이다. 그림 4(b)는 SP를 생성하여 데이터베이스에 저장하는 부분으로 학번별 수강신청리스트의 경우이다. 학번컬럼(snumber)을 변수 @snumber로 정의하고 학번의 값을 받아서 쿼리(query)문에서 실행된 결과값만을 반환한다. 그림 4(c)는 데이터베이스에 저장된 SP를 ASP.NET에서 호출하는 부분이다. 메소드에서 string형으로 받은 snumber값을 SP의 snumber변수에 넘겨주며 SP에서 실행된 쿼리문의 값을 반환하여 사용하는 부분이다. 여기서는 하나의 쿼리문이 여러 부분에 사용되어지므로 특정 쿼리문을 수정시 사용되어진 모든 부분을 각각 수정해야 한다. 그러나 SP를 사용하게 되면 SP로 저장된 부분만을 수정하므로 사용되어진 모든 부분에 적용될 수 있어 재사용이 가능하다. 또한 SP를 이용함으로써 기존의 컴파일 된 결과를 다시 호출하게 됨으로써 빠른 속도를 얻을 수 있다.

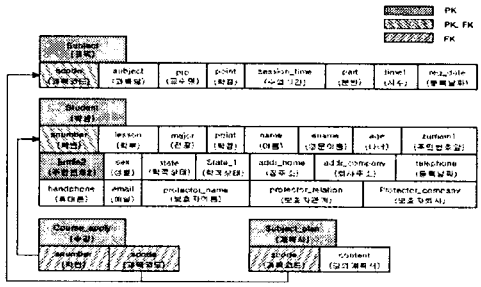
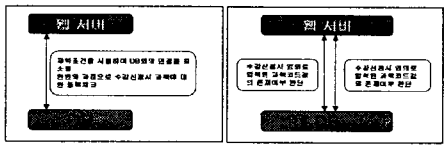


그림 2. Table design



(a) 제약조건으로 데이터베이스 구성시

(b) 제약조건 없이 데이터베이스 구성시

그림 3. 그림 2의 table design 효과

2.3 저부하형 프레임처리

제안한 시스템은 저부하형 프레임처리를 위하여 ASP.NET에 포함되어 있는 User Control을 사용하였다. 각 페이지에 로딩 될 때 프레임으로 나누어져 있던 것을 User Control로 포함 시키면 프레임으로 구성되어 각각의 프레임을 로딩하는데 필요한 시간을 제거할 수 있다. 그림 5는 User Control을 사용하고 있는 예로서 로그인 페이지의 좌편 하단의 굵은선으로 표시된 부분의 서브메뉴를 menu.ascx로 구성하였다. 메인페이지에서 header부분에 User Control을 정의하고, 서브메뉴가 사용되어지는 위치에 삽입하여 사용하였다. 따라서 기존의 프레임으로 구성된 페이지는 프레임에 속한 모든 페이지들을 전부 로딩하게 된다.

그러나 User Control을 사용하게 되면 프레임으로 구성된 페이지처럼 각각의 페이지를 전부 로딩하지 않고 프레임을 나눈 것과 같은 효과가 있으며, 페이지의 로딩수가 줄게 되어 페이지 로딩 속도가 향상된다.

```
저장프로시저 생성 구문
CREATE PROC[EDURE] procedure_name [;number]
    [
        {@parameter  data_type}  [VARYING]  [=  default]
[OUTPUT]
    ]
    [,...n]
[WITH
    {
        RECOMPILE
        | ENCRYPTION
        | RECOMPILE, ENCRYPTION
    }
]
[FOR REPLICATION]
AS
    sql_statement [...n]
```

(a) 저장프로시저 생성구문

```
Create proc tbl_sugang
        @snumber int
As
        select distinct * from course_apply c, subject s
where
        sunmber = @snumber and c.scode = s.scode
go
```

(b) 학번별수강신청리스트 tbl_sugang SP생성

```
public string Login(string snumber, string scode) {
SqlConnection myConnection = new SqlConnection
(ConfigurationSettings.AppSettings["ConnectionString"]);
SqlCommand myCommand = new SqlCommand("tbl_sugang",
myConnection);
myCommand.CommandType = CommandType.StoredProcedure
.....
```

(c) tbl_sugang SP를 호출해서 쓰는 부분

그림 4. 저장프로시저

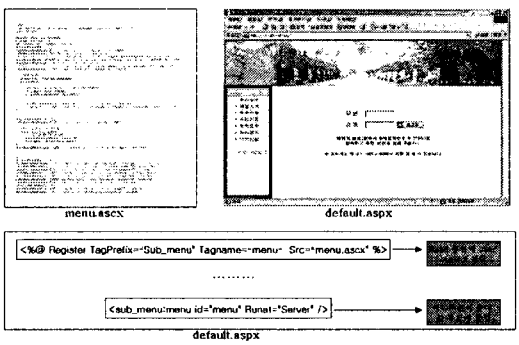


그림 5. User control 사용 예

또한 그림 6과 같이 각 페이지 마다 페이지에 로딩되는 메뉴의 연결 정보나 동적 데이터들을 XML에 저장하여 마우스의 이벤트가 발생시에만 XML에서 불러오게 하였다. XML은 페이지 리로드(reload)가 없기 때문에 데이터를 읽을 때 생기는 화면의 리로드가 필요 없으며, 페이지의 전체의 내용을 다 로딩 하지 않고 마우스 이벤트 발생시만 데이터를 읽어 오므로 페이지 로딩속도가 향상 될 수 있다. 또한 서버의 작업량을 줄여서 속도의 향상을 위해서 서버 언어의 사용을 최대한으로 줄이고, 스크립트를 많이 사용하였다. 스크립트는 클라이언트에서 컴파일된 상태로 저장되어 있다가 실행되므로, 서버작업량이 줄어들게 되며 처리속도가 향상되게 된다.

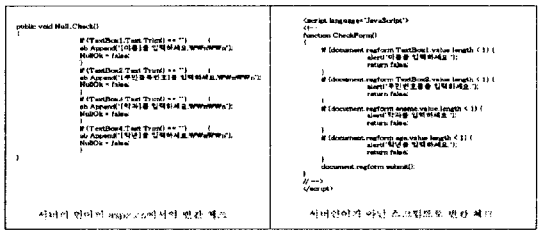
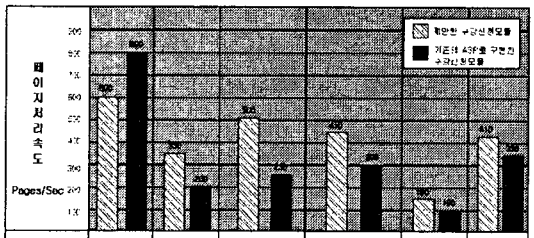


그림 6. 스크립트 사용 예

3. 성능 평가 및 고찰

본 논문에서는 제안한 시스템의 성능을 평가하기 위해 Window 2000 server에 내장되어 있는 성능 모니터링 콘솔을 사용하였다. 이 툴은 클라이언트 로드, 연결 수, 쿠키형식, HTTP 응답 헤더등을 모니터링 할 수 있다. 성능 평가를 위해 ASP 기반의 수강신청 시스템을 별도로 구성하였고, 구현한 ASP.NET기반의 수강신청 시스템과 비교하였다. 성능 모니터링 콘솔은 기존의 ASP의 경우 서버 컴퓨터에 대한 성능 카운터는 전체적으로 평가되지만 APS.NET에서는 응용프로그램별로 평가가 가능하다. 본 논문에서는 동일한 조건하에서 평가하기 위하여 ASP.NET의 경우를 전체적으로 평가가 될 수 있도록 ASP에 들어 있는 로드, 연결 수, 쿠키형식, HTTP응답헤더 등의 기능을 추가하였다. 두가지의 경우에 대해 동일한 서버에서 수강신청 서비스를 실행하면서 페이지 항목별 페이지 처리 속도를 비교하였다. 그림 7은 두가지 경우를 비교한 결과를 도표로 보여주고 있다. 로그인의 경우 첫 페이지를 로딩 하는 시간은 ASP로 구성된 시스템에 비해서 본 논문에서 제안한 방식이 처리 속도가 좀더 많이 소요되는 것을 볼 수 있다. 그러나 수강신청 및 과목코드 조회 등 접속 후 페이지별 이동에 대한 페이지처리 속도는 기존의 ASP기반 시스템보다 성능이

향상된 것을 볼 수 있다. 특히 데이터베이스에서 많은 데이터를 가져와야하는 과목코드 조회시에는 본 논문에서 제안한 관계형 데이터베이스설계와 SP를 활용함으로서 높은 성능향상을 볼 수 있다. 전체적인 페이지 처리 속도가 원인을 분석해 보면 본 논문에서 제안한 User Control을 활용함으로 기존의 프레임으로 구성되어 있던 ASP기반 시스템에 비해 페이지 처리속도가 향상된 점을 지적할 수 있다. 또한 서버의 작업을 줄이기 위해 서버언어의 작업량을 줄이고, 스크립트를 활용하므로 ASP기반 시스템에 비해 많은 페이지속도가 향상되었다고 볼 수 있다.



항목	로그인	수강신청	과목코드조회	수강신청 내역	삭제	평균
1인당 평균 Page View	2	2	3	1	2	20
View 반도수	20%	20%	30%	10%	20%	100

그림 7. 페이지항목별 성능비교

4. 결론

본 논문에서는 온라인상의 웹 애플리케이션의 경우 다중접속의 부하에 대한 대처와 보다 빠른 실행을 위해 .NET을 기반으로 한 저부하형 웹 애플리케이션 설계 방법을 제안하였으며, 이를 학사관리 시스템에 적용하여 구현하였다. 제안한 시스템은 관계형 데이터베이스를 설계하여 재사용성이 가능하며, 모듈을 최소화 하도록 생성하였으며, SP를 이용하여 복잡한 SQL문을 단순화하여 처리 속도가 향상되며, 관리가 보다 용이하게 하였다. 또한 ASP.NET에 포함되어 있는 User Control을 활용하여 페이지를 구성하여 프레임으로 나누어져 있던 페이지에 비해 페이지처리속도를 향상시켰고 스크립트를 활용하여 서버언어의 작업을 최소함으로써 서버의 부하를 줄일 수 있었다. 또한 XML을 활용하여 이벤트가 발생하지 않을 경우 페이지의 동적데이터를 로딩하지 않도록 하여 페이지의 로딩 속도를 향상시켰다. XML/EDI를 활용하여 기존의 EDI에 비해 전자문서전송을 웹상에서 이루어지게 하여 공간의 제약을 제거하였으며, 변환소프트웨어가 불필요하도록 구성하였다.

본 논문에서 제안한 시스템은 그룹웨어나 학사관리 시스템과 같이 대형 웹 애플리케이션을 설계시 서버의 부하를 줄이고 처리 속도를 향상시키기 위한 설계 방법으로 활용될 수 있다. 보다 진보된 기능의 향상을 위해서는 새로운 플랫폼에 대한 연구가 지속적으로 이루어 져야 할 것이다.

[참고문헌]

[1] J. Song, A. Iyengar, E. Levy-Abegnoli, and D. Dias, "Architecture of a Web Server Accelerator", Computer Networks, Vol.38, No.1, pp.75-97, Jan. 2002.

[2] B. Krishnamurthy and C. E. Wills, "Analyzing Factors that Influence End-to-End Web Performance", Computer Networks, Vol.33, no.1, pp.17-32, Jun. 2000.

[3] 문진용, 구용완, "인터넷 상에서 PHP를 이용한학사관리 시스템의 설계 및 구현", 한국정보처리학회 논문지 제7권 제10호, 2000.

[4] 이용훈, 한판암, "웹기반 가상연수 시스템 설계 및 구현", 한국정보처리학회 논문지 제7권 제9호 2000.

[5] 김현순, "Web 기반의 수강 시스템 설계에 관한 연구", 연세대학교 석사학위논문, 2001

[6] 신인재, "Web 기반에서의 학사행정 통합 운영을 위한 정보서비스 시스템 설계 및 구현", 충북대학교 석사학위논문, 2001

[7] 이승관, "Web 기반의 대학 수강신청 최적탐색기 개발에 관한 연구", 경희대학교 석사학위논문, 1999

[8] R. Anderson et. al., 'Professional ASP.NET 1.0', Wrox Press, 2002.

[9] N. Sundaresan and R. Moussa, "Algorithms and Programming Models for Efficient Representation of XML for Internet Applications", Computer Networks, Vol.39, no.5, pp.681-697, Aug. 2002.

[10] R. Agrawal, R. J. Bayardo Jr., D. Gruhl, and S. Papadimitriou, "Vinci: a service-oriented architecture for rapid development of Web applications", Computer Networks, Vol.39, No.5, pp.523-539, 2002.

[11] A. Banerjee et. al., ' C# Web Services', Wrox Press, 2002.