닷넷 스마트클라이언트 기술을 이용한 3-티어 환경에서의 학사시스템 아키텍쳐 설계

Design of University Management System Architecture of 3-Tier Environment Using .Net Smart Client Technology

김현준, 이준환, 현득창, 조한진
극동대학교

Kim hyun-jun, Lee Jun-Whan, Hyun Duk-Chang, Cho han-jin
Far East Univ.

요약

인터넷 보급률이 증가함에 따라 동시 접속자 수가 크게 증가한 요즘 기존에 소규모 시스템 및 소수의 사용자에게 적합한 클라이언트와 서버 방식인 2-티어 환경에서의 시스템은 성능저하의 요인이 되었고 이는 대규모 프로젝트 수행시 좀더 높은 성능으로 처리하게 되었다. 본 논문에서는 대규모 프로젝트의 최적의 성능 및 유지보수 편의성을 제공하기 위해 닷넷 리모팅 서비스와 스마트 클라이언트를 기반으로 한 3-티어 환경에서의 학사시스템 아키텍쳐를 설계하였다.

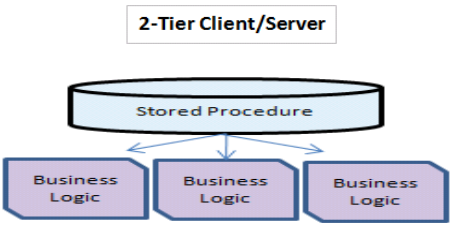
Ⅰ. 서론

본 논문에서는 학사시스템 아키텍쳐를 설계하면서 초기 2-티어 시스템에 의한 사용자 수 증가에 따른 성능저하, 애플리케이션 분산에 따른 유지관리의 어려움 등의 문제에 해결방안으로 대규모 프로젝트 수행시 최적의 성능 및 유지보수 편의성을 제공하는 클라이언트와 서버 사이에 미드웨어를 둔 3-티어 환경을 요하게 되었고 이는 닷넷 리모팅 서비스와 스마트 클라이언트를 기반으로 한 3-티어 환경에서의 좀 더 효율적인 학사시스템 아키텍쳐를 설계하고자 한다[1].

Ⅱ. 관련기술

1. 2-티어 아키텍쳐

2-티어는 클라이언트가 서버에 있는 데이터를 액세스하는 방식이다. 클라이언트에 사용자 인터페이스와 비즈니스 로직이 위치하게 되는 클라이언트를 fat client라고 한다. 처리가 클라이언트에서 이뤄지기 때문에 클라이언트에 부하가 크고 많은 사용자의 동시 접속에 의한 성능이 저하되는 단점이 있다[1].



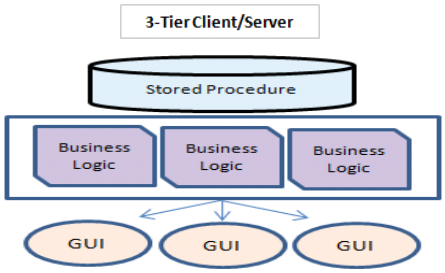
▶▶ 그림 1. 2-티어 아키텍쳐

2. 3-티어 아키텍쳐

2-티어 구조에서는 여러 문제가 발생할 수 있는데 이러한 문제의 근본적인 이유는 클라이언트와 서버가 논리적으로는 일 대 일의 서비스 관계를 유지해야 하기 때문이다. 이러한 문제를 개선한 구조가 3-티어 클라이

언트-서버 모델인데, 3-티어 구조에서는 응용 처리를 전담하는 중간계층을 두고 클라이언트는 모든 서비스를 중간계층에게 요구하고 중간계층은 데이터베이스 서버와 같은 최종 서버와 통신을 한 후 그 결과를 클라이언트에게 전달한다[2].

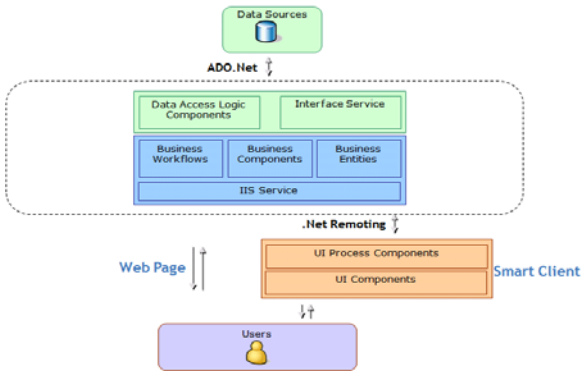
3-티어 구조는 서비스 확장성과 안정성, 그리고 효율면에서 유리하므로 대용량 서비스에 적합하고, fat-Client 보다 서버에 부하가 적기 때문에 성능이 좋아지고, thin-Client 보다 프리젠테이션 로직에 어플리케이션로직이 포함 안 되므로 관리가 보다 쉽게 된다[3].



▶▶ 그림 2. 3-티어 아키텍쳐

Ⅲ. 아키텍쳐 설계

3-티어 아키텍쳐는 재사용성과 유지보수를 위해 비즈니스 로직 계층을 별도의 서버에 둔 것이다. 이렇게 둔 경우 비즈니스 로직 변경시 분리된 비즈니스 로직 계층만 변경하면 되고 시스템 자원을 보다 효율적으로 관리할 수 있다. 〈그림 3〉은 3-티어 아키텍쳐가 적용된 학사 시스템에 데이터 전송방식이다.



▶▶ 그림 3. 학사 Application 아키텍쳐 시나리오

1. 아키텍쳐 특징

독립 실행형의 스마트 클라이언트로 구현하고, 닷넷 리모팅은 IIS에 호스팅되는 바이너리 방식의 닷넷 리모팅으로 구현한다. 비즈컴포넌트는 닷넷 컴포넌트로 구성된다. 트랜젝션의 관리는 System.Transaction으로 구현한다. 비즈니스 로직은 주고 스토어드 프로시져에 포함된다.

2. 계층구분

〈표 1〉은 실제 학사 시스템에 구현되는 아키텍쳐 계층에 적용된 기술을 구분하였다.

표 1. 학사 Application 아키텍쳐 계층

프리젠테이션 계층	UI(User Interface)
미들웨어	· Remoting(.Net Remoting) · Biz(Business Service Layer) · DAL(Data Access Layer)
데이터베이스	DataBase(MS_SQL 2008)

2.1 프리젠테이션 계층(User Interface)

데이터의 조회/입력/수정/삭제 등의 이벤트 처리를 담당한다. 폼베이스를 상속받아 구현하고 입력 데이터의 유효성을 체크한다. 공통에서 제공하는 리모팅 개체를 통해서만 서버상의 비즈 컴포넌트를 호출할 수 있다. 개발자가 개발을 담당하는 부분이다.

2.2 미들 웨어(Business Layer)

2.2.1 Remoting(.Net Remoting)

사용자 인터페이스와 통신하는 계층으로 사용자의 요청을 비즈니스 컴포넌트로 전달하는 역할을 한다. 개발자가 개발을 담당하는 부분이다.

2.2.2 Biz(Business Service Layer)

비즈니스 컴포넌트와 직접 연관되는 데이터 엑세스 코딩은 DAL의 호출로 구현한다. 트랜잭션 관리를 위해

System.Transaction을 사용하고 필요한 부분만 트랜잭션 처리를 하도록 최소화한다. 개발자가 개발을 담당하는 부분이다.

2.2.3 DAL(Data Access Layer)

스토어드 프로시져만 호출 가능하다. 데이터베이스 커넥션 정보는 암호화 되어 있다.

2.3 DataBase(Stored Procedure)

MS_SQL2008 스토어드 프로시져를 사용하고 비즈니스 로직의 변경 및 추가의 경우에 있어서 모듈 재컴파일 및 배포의 과정이 필요 없어 유지보수 관리가 보다 수월하다. 산술적인 로직 구현은 중간계층을 활용한다. SQL보다 C#언어가 훨씬 성능이 뛰어나기 때문이다.

Ⅳ. 결론

본 논문에서는 C# Visual Studio 2008과 MS_SQL 2008 기술을 이용한 학사관리 시스템 프로젝트에 초점을 맞추어 아키텍쳐를 설계하였다. 대형 프로젝트에 맞는 아키텍쳐 구현을 제시하고 닷넷의 데이터셋 개념을 사용해 서버의 부하를 줄이고 개발에 편이성을 중심으로 설계할 것이다.

향후에 3-티어에 환경에서, 더 나은 시스템 환경을 구축하고 동시에 개발에 대한 편이성 연구가 이루어져야 할 것으로 생각한다.

■ 참 고 문 헌 ■

[1] 왕정휘 “기업 환경에 최적화된 닷넷기반 솔루션 구현”, 고려대학교 컴퓨터정보통신대학원, 석사 학위 논문, 2006.

[2] 김대근 “닷넷 웹 기반의 개발 생산성 향상에 필요한 패키지 설계 및 구현” 부경대 교육대학교 석사 학위논문, 2007.

[3] 나윤지 “.NET을 기반으로 한 효율적인 전자상거래 시스템에 관한 연구”, 학국콘텐츠학회논문지, 2002.