MVC 프레임워크를 적용한 웹 기반의 시스템 관리 도구의 설계 및 구현

김지연[O] 안창원
한국전자통신연구원
{jiyeonkim[O], cwahn}@etri.re.kr

Design & Implementation of System Management Tool based on the internet developed by MVC

Jiyeon Kim[O] Changwon Ahn
Operating System, Computer System Dept. Electronics and Telecommunication Research Institute

요 약

스몰토크-80의 사용자 인터페이스는 MVC (Model/View/Controller) 프레임워크에 입각하여 디자인 되었다. 그 이후 MVC 프레임워크는 많은 시나리오에서 사용되고 있다. MVC 프레임워크를 적용하여 웹 어플리케이션을 개발하면 시스템 개발 및 유지 보수가 쉬워지며 확장성과 성능 향상에 도움을 준다. 본 논문에서는 MVC 프레임워크를 적용하여 시스템 관리 도구를 설계하고 구현한다. 더 나아가 MVC 프레임워크의 단점을 보완한 방법을 제시하고 이를 시스템 관리 도구에 적용하여 본다.

1. 서론

네트워크 환경이 발전함에 따라 시스템 관리 도구는 단일 시스템의 상태뿐만 아니라 네트워크로 연결되어 있는 서버들을 일률적으로 관리하게 되었다. 로컬 어플리케이션으로 제공되던 시스템 관리도구는 웹의 발달로 인하여 어플리케이션의 웹 지원 능력이 요구되고 있다. 로컬 어플리케이션 기능을 웹으로 변환하는 과정에 있어서 발생되는 문제점과 제약점들로 인하여 웹 어플리케이션 개발 프로세스의 효율성에 대한 관심이 증가하고 있다.

초기 JSP 스펙[1]에서는 JSP 기술을 사용하는 웹 어플리케이션을 제작하는 두 가지 방법을 제시했다. JSP 1 모델은 JSP 페이지가 모든 요청을 처리하고 클라이언트에 출력하는 방식을 가진다. JSP 2 모델은 클라이언트의 요청을 컨트롤러 역할의 서블릿[2]이 먼저 받아들이고, 수신한 요청을 처리하여 해당 JSP 페이지를 결정한다. JSP 2 모델 방식은 웹 어플리케이션에서 역할 분리와 밀접하게 관련되어 있다. JSP 페이지가 요청수신, 비즈니스 로직 수행, 다음에 보일 뷰 결정 등의 역할을 모조리 담당하게 된다면 유지보수나 확장성에 문제가 생기는 것은 물론 JSP 자체도 매우 지저분해진다. 어플리케이션 개발과 유지를 쉽게 하려면 웹 어플리케이션의 컴포넌트들이 명확하게 개별적인 역할을 담당하도록 해야 한다. 두 모델 사이의 주요 차이점은 모델 2 구조에서는 단일 진입점을 제공하기 때문에 클라이언트의 요청을 효율적으로 처리할 수 있으며, 모델 1 구조보다 비즈니스 로직과 프리젠테이션 로직을 명확히 구분할 수 있기 때문에 뛰어난 재사용성과 확장성 및 유지보수성을 제공한다. 이와 같은 JSP 2 모델의 구조는 MVC 프레임워크[3]를 기반으로 하고 있다. 하지만 모델 2 방식에도 한계점이 있다. 본 논문에서는 MVC 프레임워크의 단점을 보완한 아이디어를 제시하고 이를 적용하여 웹을 지원하는 시스템 관리 도구를 설계하고 구현하고자 한다.

2. 관련 연구

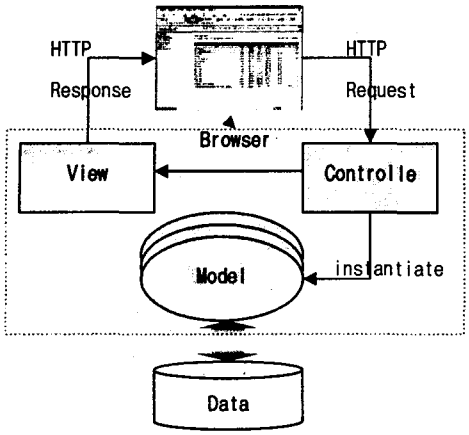
웹 초창기에 HTML 문서가 정적이었던 것과는 달리 근래에 와서는 동적인 컨텐츠를 생성할 수 있는 능력이 요구되고 있다. 이러한 요구에 맞추어 처음 등장한 것이 CGI(Command Gateway Interface)이다. CGI는 동적으로 클라이언트의 요청을 처리할 수 있게 해주지만 프로그래밍 언어와 HTML를 명확히 구분하기 어렵다. 또한 CGI에서는 Message 메커니즘이나 Response 메커니즘과 같은 MVC 프레임워크를 지원하는 기술들이 부족하다.

이후 CGI의 문제점들을 해결하고 자바 언어의 강력한 기능성을 기반으로 자바 서버 페이지 스펙 설계자들은 웹 어플리케이션을 위한 두가지 모델 JSP 1과 JSP 2 모델을 제공했다. JSP 2 모델은 MVC 프레임워크의 이론적인 구현을 효과적으로 가능하게 함으로써 개발과 유지보수의 용이성 및 확장성 등을 제공하므로 웹 어플리케이션 개발에 널리 쓰이고 있다

MVC 프레임워크는 다음과 같은 세 개의 컴포넌트로 구성된다.

- 모델(Model): 엔터프라이즈 자바빈즈로 캐슐화된다.
- 뷰(View): JSP로 프리젠테이션을 책임진다.
- 컨트롤러(Contoller): 자바 서블릿으로 클라이언트로부터 HTTP Request를 수신한다.

그림 1의 컨트롤러 컴포넌트는 클라이언트로부터 요청을 가로채서 각 요청을 수행할 특정 비즈니스 오퍼레이션으로 바꾸거나 비즈니스 오퍼레이션을 직접 호출하고 이를 핸들러에게 넘겨준다. 또는 클라이언트에 보여줄 뷰를 선택하는 작업을 돕는다. 모델 컴포넌트는 뷰 컴포넌트에서 사용될 데이터를 캡슐화 하는 작업을 한다. 뷰 컴포넌트는 캡슐화된 빈을 이용하여 클라이언트에게 프리젠테이션을 한다.



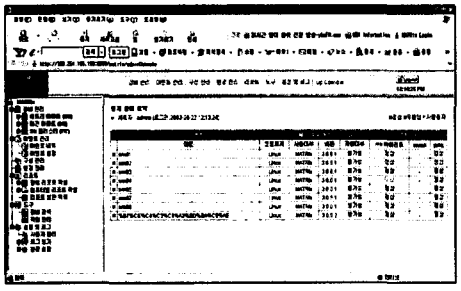
[그림 1] MVC 프레임워크 기반 웹 어플리케이션

각 컴포넌트의 역할에서 보듯이 MVC 프레임워크는 비즈니스 로직과 프리젠테이션 로직을 구분하여 역할 분담이 명확하므로 개발자의 웹 어플리케이션 개발이 쉽도록 해준다. 그러나, MVC 프레임워크를 적용하여 웹 어플리케이션을 개발할지라도 여전히 한계는 존재하고 있다. 컨트롤러 컴포넌트는 클라이언트의 모든 요청을 수신해야 하고 비즈니스 오퍼레이션을 호출하거나 선택하는 작업 모두를 수행해야 한다. 또한 모델에서 제공되는 데이터를 뷰에 전달할 책임도 지니게 된다. 즉, 컨트롤러 컴포넌트는 매우 복잡한 로직을 지니게 된다.

3. 웹 시스템 관리 도구 개발 구조

시스템 관리 도구는 네트워크로 엮여있는 수많은 시스템들을 일률적으로 관리할 수 있는 기능들을 제공한다. 관리 대상이 되는 시스템으로부터 정보를 가져와 이를 가공해 관리자가 시스템의 상태를 파악할 수 있도록 정보를 화면에 뿌려준다. 시스템 관리 도구는 관리 대상이 되는 시스템의 CPU, 메모리, 디스크 등과 같은 하드웨어 자원 상태, 로그와 같은 소프트웨어 상태 및 관리 대상들을 효율적으로 다루기 위한 그룹 관리나 노드 관리와 같은 다양한 기능을 제공한다. 이와 같은 기능들을 개발하는 데는 여러 개발자의 공동의 노력이 요구되므로 효율적인 개발 프로세스가 요구된다.

본 장에서는 웹 기반의 시스템 관리 도구를 MVC 프레임워크로 설계하고 구현한 아이디어를 설명하고자 한다. MVC 프레임워크의 각 컴포넌트를 시스템 관리 구조를 설계하는 데 적용시켜 설명하고 각 컴포넌트의 성능을 향상 시킬 수 있는 아이디어를 제시한다.

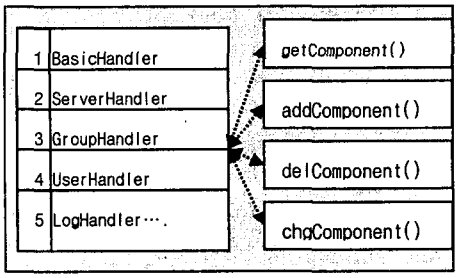


[그림 2] 웹 시스템 관리 도구 메인 화면

구현된 시스템 관리 도구는 그림 2와 같다. GUI는 HTML인 JSP로 이루어져 있고 관리자의 작업 요청을 받아들이는 역할은 서블릿이 맡게 된다. 서블릿이 클라이언트의 작업 요청을 받아들여 요청을 처리할 수 있는 모듈을 호출하도록 되어 있다. 모듈의 결과는 엔터프라이즈 자바 빈즈 개체로 생성되고 이는 컨트롤러 컴포넌트에 의해 HTTP Response로 클라이언트에게 전송된다.

3.1 컨트롤러 (Controller)

시스템 관리 도구에서 컨트롤러 컴포넌트의 역할을 담당하는 것은 서블릿이 수행하고 있다. 서블릿은 클라이언트의 요청을 수신하고 요청에 따라 요청을 처리 할 트랜잭션 로직을 찾는다. 시스템 관리 도구가 제공하는 기능이 다양하므로 트랙잰션과 로직이 복잡하다. 본 논문에서는 컨트롤러 컴포넌트를 기능의 특징 별로 세분하고 있다. 예를 들어 그룹 관리는 그룹 핸들러가 처리하고 사용자 관리는 사용자 핸들러가 처리한다. 각각의 핸들러는 스레드로 생성되므로 병렬적으로 작업을 수행할 수 있다. 작업을 해당 컴포넌트로 분배하는 작업은 HTTP Request 인자에 따라 이루어진다. 이를 Transaction Distributor라고 불리는 컴포넌트가 담당하게 된다. Transaction Distributor는 그림 3과 같이 해쉬 테이블 구조로 이루어져 있어서 HTTP Request 인자로 쉽게 해당 컴포넌트를 검색할 수 있다. 또한 새로운 기능이 추가될 때마다 이를 처리할 트랜잭션과 로직을 가진 새로운 컴포넌트를 추가하고 삭제하는 것이 용이하다. 시스템 관리자가 도구에 로그인 하는 경우, 암호화된 ID와 패스워드를 HTTP Request에 실어서 서블릿인 *AdminServlet*으로 보낸다. 이를 받은 *AdminServlet*은 인증 과정을 처리하는 BasicHandler를 호출하게 된다. 본 논문에서 구현한 시스템 관리 도구의 모든 기능은 위와 같은 흐름을 따르게 되므로 새로운 특징의 기능이 추가될 때마다 이를 처리하는 컴포넌트를 생성하고 이를 Trascaction Distributor에 등록하여 사용할 수 있다. 여러 개발자가 작업을 분할할 때 각 컴포넌트 별로 작업을 수행할 수 있는 장점도 있다.



[그림 3] Transaction Distributor의 구조

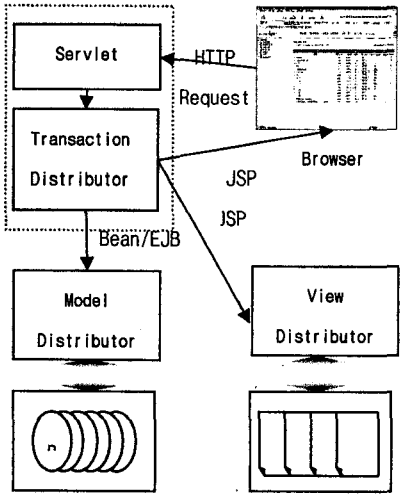
3.2 모델 (Model)

컨트롤러의 해당 작업 컴포넌트의 요청 처리 결과를 받아 들여 View Distributor에게 보낼 데이터를 가공하는 역할을 한다. 컨트롤러로부

터 수신된 결과는 메시지 큐를 이용해서 저장되고 메시지 큐로부터 결과를 꺼내 자바 빈즈 개체를 생성한다. 메시지 큐는 요청을 버퍼링 하고 있으므로 여러 클라이언트의 요청을 모두 처리할 수 있는 장점이 있다. 빈즈 개체는 결과 값을 여러 번 전송하는 과정을 거치지 않도록 모든 데이터를 담고 있다. 빈즈 개체를 사용하면 네트워크의 부하를 줄이고 응답 시간을 향상 시키는 장점이 있다. 또한 비즈니스 객체와 프리젠테이션을 분리하는 것을 도와 주고 유지 보수와 기능 향상을 쉽게 해준다. 빈즈 개체는 두 가지 경로를 통해 생성된다. 앞서 설명한 인증 기능은 Transaction Distributor에 의해 로드된 핸들러에 의해 넘겨받은 ID와 패스워드를 복호화하고 이를 데이터베이스의 사용자 테이블의 정보와 비교한다. 그 결과를 빈즈 개체로 생성하고 이를 뷰 컴포넌트로 넘겨준다. 그리고 CPU나 메모리와 같은 주기적인 모니터링이 필요한 기능은 ID 인증 기능과 달리 결과를 바로 뷰 컴포넌트로 보내지 않는다. 주기적으로 추출된 결과 정보는 데이터베이스에 저장되고 요청을 완료한다. 이후 관리자의 모니터링 요청이 AdminServlet으로 수신되면 로드된 핸들러는 데이터베이스로부터 결과를 추출해 이를 빈즈 개체로 생성하고 이를 뷰 컴포넌트로 넘기게 된다. 실시간 정보를 추출하지 못하는 단점을 해결하기 위해 모니터링 폴링 주기를 최소 1초까지 설정할 수 있다. 폴링 주기에 따라 핸들러는 에이젠트로부터 정보를 추출해 데이터베이스에 저장하므로 실시간에 가까운 정보를 모니터링 하는 것이 가능하다.

3.3 뷰 (View)

시스템 관리 도구의 뷰 컴포넌트는 View Distributor 컴포넌트로 이루어져 있다. View Distributor는 클라이언트의 요청에 따라 JSP 페이지를 검색하고 추출한다. 선택된 JSP가 프리젠테이션을 하기 위한 정보는 모델 컴포넌트에서 생성된 빈즈 개체이다. 해당 JSP 페이지 컨트롤러에 의해 HTTP Response로 전송된다.



[그림 4] MVC 기반의 웹 지원 시스템 관리 도구 성능 향상 구조

지금까지 클라이언트로부터 요청이 들어왔을 때 이를 처리하고 결과를 프리젠테이션 하는 흐름을 MVC 프레임워크 기반으로 설명하였다. 주기적으로 정보가 갱신되어야 하는 CPU나 메모리와 같은 정보는 웹 기반의 어플리케이션 개발에서는 구현이 매우 어렵다. 시스템 관리 도구는 시스템의 상태를 통지하는 어플리케이션이므로 주기적으로 서버의 정보를 가져와 그 정보를 프리젠테이션하는 기능들이 많다. 이를 고려하여 시스템 관리도구에서 주기적으로 업데이트 되어야 하는 정보에 대한 클라이언트의 요청은 쓰레드를 이용하여 비동기적으로 처리한다. 컨트롤러는 서버의 정보를 추출하는 에이젠트로 요청을 보낸 후 리스너에 요청을 등록한다. 해당 컨트롤러는 에이전트로부터 결과를 받으면 그 결과를 클라이언트에게 보여준다. 리스너 핸들러 구조를 통해 MVC 프레임의 워크를 이용하여 웹 기반의 어플리케이션을 개발하였을 때 발생할 수 있는 모델 컴포넌트와 뷰 컴포넌트간의 정보 업데이트 문제를 해결하도록 구현하였다.

4. 결론

본 논문에서는 MVC 프레임워크를 적용하여 웹 시스템 관리 도구를 설계하고 구현하였다. MVC 프레임워크를 적용하여 웹 시스템 관리 도구를 설계하면 MVC 세 개의 컴포넌트의 역할을 명확히 구분할 수 있으므로 확장과 유지 보수가 쉬워지며 기능을 추가하거나 삭제하기가 쉽다. 컨트롤러 컴포넌트는 새로운 트랜잭션이 요구되는 기능을 처리하기 쉽도록 기능 별로 컴포넌트를 세분하고 있으며, 모델 컴포넌트는 컨트롤러 컴포넌트의 결과를 캡슐화시켜서 뷰 컴포넌트에서 사용하기 쉽도록 구현되고 있다. 또한 본 논문의 시스템 관리 도구는 모델의 정보가 갱신될 때마다 이를 뷰 컴포넌트에게 알릴 수 있도록 비동기화 모드로 설계되어 있다. 뷰 컴포넌트는 모델 컴포넌트의 빈즈 개체를 효율적으로 프리젠테이션 하도록 이루어져 있다.

향후에는 각 컴포넌트들의 성능 측정을 통해 성능 향상 지점을 찾아내고 자 한다. 다양한 컴포넌트들이 얼마나 빠르게 동작하는지, 사용자의 요청에 얼마만큼의 응답속도를 내면서 성능을 유지할 수 있는지 알고자 한다.

참고 문헌

[1] Sun Microsystems Inc. Java TM Servlet 2.3 and JavaServer PagesTM 1.2 Specifications

[2] 최종명. 사이버 강의로 배우는 웹 개발자를 위한 서블릿. 이한디지털

[3] Erich Gamma. Design Patterns. U.S. Addison-Wesley Pub Co.

[4] Chuck Cavaness. Programming Jakarta Struts. U.S. O' Reilly

[5] 서순모, MVC Archictecture 기반의 EC System용 상품 전시 컴포넌트의 설계 및 구현, 한국정보과학회 학술발표논문집(가을), Vol 1, p.412-414, 2001