Journal Of Harmonized Research (JOHR)

Journal Of Harmonized Research in Engineering
1(2), 2013, 110-114

ISSN 2347 – 7393

Review article

J2EE AND MODEL-VIEW-CONTROLLER ARCHITECTURE – A REVIEW

Naveen Malik, Naeem Akhtar, Pankaj Sharma, Rahul, Hardeep Rohilla

Dronacharya College of Engineering, Khentawas,
Farukhnagar, Gurgaon, India

Abstract
J2EE architecture has various complicated layers which includes client, presentation, business logic and data persistence layer. There is respective data presentation in applicable layer of J2EE, with strictly rules to access and transform data objects in between these layers. In J2EE, it should control the access to data persistence layer to protect important enterprise data and avoid showing the database pattern to client directly. The data persistence layer lies behind the business logic layer and provides business interfaces via the business logic layer to communicate with the presentation layer to package the data persistence layer, separates the data persistence layer from the presentation layer. This paper contains middle-ware technology which is the kernel and key to the solution of the dynamic E-business system through analyzing J2EE architecture, MVC design pattern, the component technology of JSP, Servlet and EJB etc. J2EE technology and MVC design pattern can simplify the software development, improve the software performance and quickly construct the dynamic E-business system of the good extensibility, maintainability, dependability and high usability.

Key-Words: – Persistence Layer, Presentation layer, Business logic layer, Model, View, Controller etc.

1. Introduction

This paper uses J2EE architecture and MVC (Model-View-Controller) design pattern to design and implement a dynamic E-business system or a web based application. On the Internet today, with the enterprise level application is developing quickly, and the electronic commerce market is inflating fast, more and more enterprise level projects presents the development trend that they take the Web technology as the main technology. At the same time, the dependence to the server end technology like the middle-ware is also increasing. So the information and technology department of the Enterprise

For Correspondence:
malik2008ATgmail.com
Received on: October 2013
Accepted after revision: December 2013
Downloaded from: www.johronline.com

requires a possible way to develop the application programs, and make the application programs be related with those middle-ware which are flexible and can be transplanted.

These application programs should be able to concurrently deal with tens of thousands of users continuously all day long. One of the problems to build such a complex application program is how to design it, develop it and maintain it. Using J2EE can not only simplify the foundation of the enterprise level application program, but also cause the designers and the programmers to distribute the function in each discreteness of the server end when using J2EE to establish the application programs. As the J2EE adopts the layered thought, and subdivides the process of the corresponding response or the resources that obtains from the user into multilayer to deal with, and each layer is realized by the component technology, therefore regarding a large-scale enterprise level application, it needs to coordinate the relations of each layer well, and increases the connection of each layer, but maintains the incompact coupling. The key of the managing to each layer in J2EE is to control the interior business data and the customer data in the J2EE architecture system. The multi-layered architecture of J2EE

J2EE (Java 2 Platform, Enterprise Edition) is an architecture which uses Java platform to simplify many enterprises on the development, deployment and management of the related complex questions. The basic J2EE technology is the core of Java platform or Java 2 platform standard. Not only does it consolidate many advantages of the standard and more convenient JDBC, JNDI, RMI, CORBA API, but also provides the protected data security pattern in Internet applying and comprehensively supports EJB, Java Servlet, Java Mail and XML technologies.

2. The Multi-layered Architecture of J2EE

The multi-layered architecture of J2EE [2] mainly includes the client layer (the behave layer), the presentation layer (the Web layer), the business logic layer (the application layer) and the data persistence layer (the EIS layer). Each layer has its specific function. Through providing components to the application program, J2EE realizes the multi-layered architecture function [3].\The client layer J2EE supports many kinds of client types [4]. Because of the B/S structure [5], the main body of the program runs at the server end, and the client server does not need to undertake the complex calculate duty (this is the so-called “the thin client server”). The major function of the client layer is to enable the users to carry on the communication smoothly with the server through the user interface. There are main three kinds of technologies on client layer: HTML (a general e-commerce network, and it is also the most universal kind), the radio equipment (it can get information anytime/anywhere), and the Applet or the Java application program (which can be used to realize the complex & fast user interface) and so on.

2.1 The presentation layer

This is also called the Web layer [6], it runs in the J2EE Web vessel. Its main function are to deal with the HTTP request comes, and reply the HTTP dynamically according to the Servlet and JSP in the Web server.

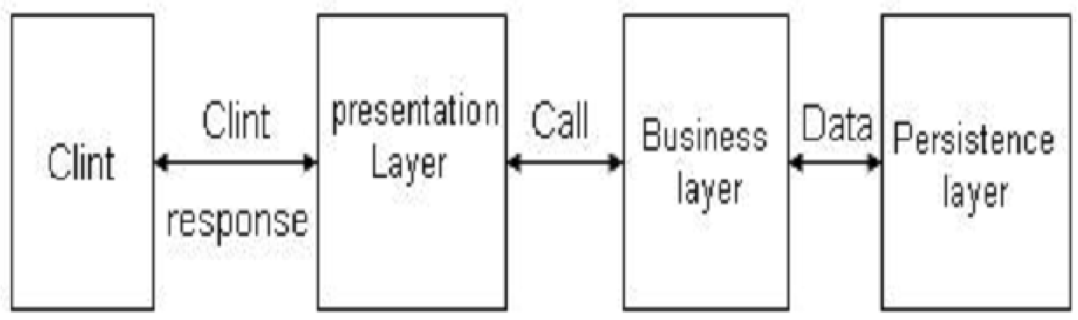

Fig 2.1 Interaction between layers

2.2 The business logic layer

The diverse technologies the business logic layer [7] use run in the corresponding vessels. They are simply used to recognize the business logic in the enterprise application program. The so-called business logic refers to the processing demand to the data carried by the specific enterprise (client). For example, in the telecommunication enterprise, there are a series of operations like as inquiring the phone bill, filling the account information, calculating the cost and so on. Every enterprise has its unique and single business logic because of its own typical quality, so it requires to be realised in the business logic layer. In the actual project development, the business logic layer is very important; it provides concurrency, flexibility, life cycle management, and fault tolerance and so on for the entire project.

2.3The persistence layer

The persistence layer defines the business data of the application program, and simulates the real entity and the business flow of the organizations, so it is the foundation of the enterprise application.
The business object is the simple software abstract of the real world entity, it represents some concrete object in the business domain.

3. J2EE Architecture

Java 2 Enterprise Edition programming is very simple with using J2EE architecture based on the component, delaminating and the platform independent, because the business logic is encapsulated into reused component, and J2EE server provides the background services for all components types through using container, it is not required to develop this kind of service by oneself. Therefore we may concentrate the energy on solution handy business problems. J2EE provides a many of developing technologies, there are more commonly used in the following:

- JSP
- Servlets

3.1 JSP

JSP (Java Server Page) technology is a new dynamic web application technology. JSP web page contains the traditional HTML web page files (*.html, *.htm), which are inserted into Java program files (Scriptlet) and JSP tags. Therefore, it comes into being a dynamic page on the server according to the client request.

3.2 Servlet

Servlets are the small Java program on the server side and it should realize HttpServlet interface. It can respond and deal with client request through Servlet API. Interaction between different layers In the J2EE multi-layered architecture, it should strictly control the visit to the persistence layer. One of the reasons that J2EE adopts the layered thought is to protect the important business data, and avoid exposing the database pattern directly to the client server. Generally, the presentation layer interacts with the database by the business layer and the persistence layer, as the Fig. shows.

The word Façade in English means the frontage of the building. But in the design pattern, it means to hide the facility behind the interface (the positive pattern Façade Pattern) by the uniform and simple interface. According to this thought that hides the persistence layer behind the business logic layer, and connects with the presentation layer through the interface that the business layer provided, to realize the separation between the persistence layer and the presentation layer.

4 HISTORY OF MVCMVC was first described in 1979 by Trygve Reenskaug, then working on Smalltalk at Xerox PARC. The original implementation is described in depth in the influential paper “Applications Programming in Smalltalk-80: How to use Model–View–Controller.

There have been several derivatives of MVC. For example, Model View Presenter is used with the .NET Framework, and the XForms standard uses a “model-view-controller-connector architecture. However, standard MVC remains popular.
Though MVC comes in different flavors, control flow is generally as follows:

i. The user interacts with the user interface in some way (for example, presses a mouse button).
ii. The controller handles the input event from the user interface, often via a registered handler or callback and converts the event into appropriate user

action, understandable for the model.

iii. The controller notifies the model of the user action, possibly resulting in a change in the model's state. (For example, the controller updates the user's shopping cart.)

iv. A view queries the model in order to generate an appropriate user interface (for example, the view lists the shopping carts contents). Note that the view gets its own data from the model. The controller may (in some implementations) issue a general instruction to the view to render itself. In others, the view is automatically notified by the model of changes in state (Observer) which require a screen update.

v. The user interface waits for further user interactions, which restarts the cycle. Some implementations such as the W3C XForms also use the concept of a dependency graph to automate the updating of views when data in the model changes.

By decoupling models and views, MVC helps to reduce the complexity in architectural design and to increase flexibility and reuse of code.

4.1 MVC(Model-View-Controller)

MVC is a software design pattern. Using this pattern, the system is divided into three modules, every module has own function. This is a typical multi-tier structure designing ideas. There are three categories in the following [3]:

i. The Model
ii. The View
iii. The Controller

Which are further defined as following:

i. The model

The model is the principal part of application program. It expresses business data, it is clearly said that one model is a record in database. It is independent of the data form, in other words one model may provide the data for many views, in this way it reduces the repeated code for our application program.

ii. The view

The view is an interrelated part of User Interface (UI) in application program and a seeing and exchanging interface by users. One of advantages is that it can process many different views for your application program using MVC. Actually does not have the true processing to take place in the view, and the view is only regarded as a means of output data and user operation.

iii. **The controller**

The controller controls UI data displaying and updates the model object state according to the users' input. The controller accepts the user input and calls the model and view to complete the user demand. So when we click the hyperlinks in the web page and send HTML table list, the controller itself does not have any output and make any processing. It only receives the request and determines calling which model to deal with the request, and it confirms to use which view to display the returning data of the model processing.

There is the relationship of the model, the view and the controller [4], shown as Fig.

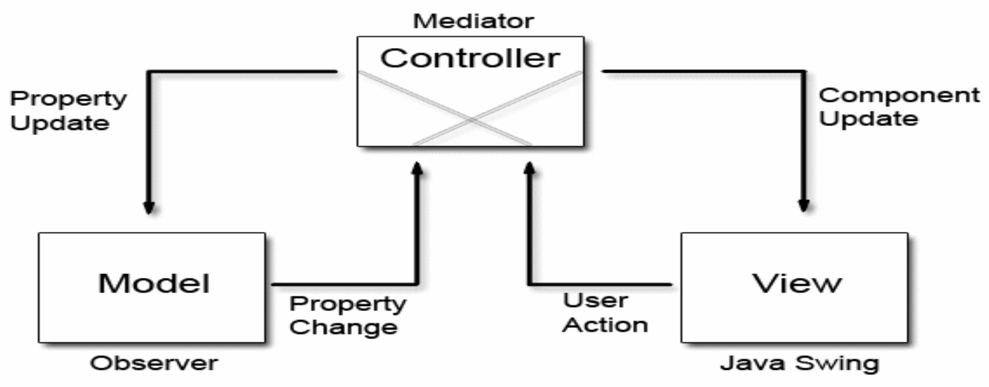

Fig.4.1 MVC design pattern

Web server is an important part of J2EE architecture. The main technologies are Servlet and JSP; its function implements UI. Because the essential of JSP and Servlet is a small Java program on the server side, it can exchange Java Applet and HTML with the client. Not only is implementing speed fast and is UI generation flexible, but also is the security very good, at the same time, it can realize some simple application logic.

5. CONCLUSION

The component technology of JSP, Servlet, EJB etc based on J2EE platform and MVC design pattern can simplify the developing process, improve the software performance. Moreover, Based on the development architecture composed by the Struts and the Hibernate, it passes the data by the value object which the layers corresponding to, and strictly controls the visit to the persistence layer by the users. In this way, it can protect the business data effectively. Besides, it is useful to the upper debug and maintenance by separating the business logic and the data expression, and also by separating the data in different layers.

6. References

[1] Hangzhou, China, April 6-8, 2008 Research on Data Expression in J2EE Architecture system

[2] Michael Girdley, J2EE Applications and BEA WebLogic Server, Publishing House of Electronics Industry. 2002.

[3] Wu Wei, Lu Jian-de, Research of Layer Pattern on Developing of J2EE Application, Microcomputer Development, Vol.15, No.1, 2005.1, pp.125-127.

[4] Chuck Cavaness, Jakarta Struts, O'REILLY Publishing, 2004, 151-184.

[5] Bambara J.J, J2EE Unleashed, Beijing: China Machine Press, 2002.

[6] Zhu Xiaoyi, The 3-tier B/S Enterprise Information System Based on J2EE, Journal of Taiyuan University of Technology, Vol.36,

[7] Jim Keogh. J2EE Complete Reference, Publishing House of Electronics Industry. 2003.

[8] Mike Brevoort, Sr. Associate, Mike Steinley, Sr. Associate, Heather Spinnenweber, Sr. Associate, A Distributed Component Architecture Model, October 2004.

[9] The J2EE 1.4 Tutorial, http:// ava.sun.com/j2ee/1.4/docs/tutorial/doc/