Global Research
Researcher Forum

Journal of Global Research Computer Science & Technology
(JGRCST)
Vol-I, Issue-II, July 2014
ISSN: 2349 - 5170

J2EE and MVC Architecture

Manish Bhatt
Banaras Hindu University (BHU)
Varanasi (UP)

Abstract: J2EE architecture has many complicated layers, including client. These layers are presentation layer, business logic layer and data persistence layer. There is respective data presentation in relevant layer of J2EE, with strictly rules to access and transform data objects between these layers. This paper brings forward middleware technology which is the kernel and key to the solution of the dynamic E-business system through analyzing J2EE architecture and the most important MVC (Model View Controller) design pattern. It includes 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 expansibility, maintainability, dependability and high usability. The traditional system of industry report is highly influenced by the database security and has low efficiency on report. In order to solve these problems, this paper studies J2EE and MVC architecture, combine them and propose an industry report system which based on J2EE and MVC architecture. The system which makes full advantages of both architecture, has solved the problems such as avoid showing databases pattern to client directly and separating the data persistence layer, business logic layer and presentation layer from each other. It obtains a good result in the web based practical application.

Keywords: 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 growing fast, more and more enterprise level projects presents the development trend that they take the Web technology as the central technology. At the same time, the dependence to the server end technology like the middleware is also increasing. So the information and technology department of the Enterprise needs a feasible way to develop the application programs, and make the application programs be related with those middleware which are flexible and can be transplanted.

These application programs should be able to concurrently deal with tens of thousands of users uninterruptedly 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 multi-layer 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 cohesion 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.

2. History of J2EE Architecture

How J2EE came about is quiet interesting. Java, originally named Oak, was conceived as a software language for developing applications for household appliances and other such devices. With the internet revolution, Java gradually evolved into a language for client-side development with capabilities such as applets and JavaBeans. Along the way, several Java APIs, such as JDBC, were developed to address market needs for generic access and usage of resources typically required by enterprise software applications.

Journal of Global Research Computer Science & Technology (JGRCST)
Vol-I, Issue-II, July 2014
ISSN: 2349 - 5170

It was clear soon after Java's introduction that the use of Java on the client side in a browser-based systems environment faced some serious challenges, such as the latency involved in the loading of Java libraries over the internet before a client-side Java application could start up. However, Java's relative simplicity, platform independent architecture, and rich set of API's as well as its Web enabled nature were strong positives for its use in enterprise software development.

3. 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 applications. 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. The multi-layered architecture of J2EE 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.

a. The client layer

J2EE supports many kinds of client types. Because of the B/S structure, 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 three kinds of technologies on the client layer: HTML (a general electronic commerce network, and is also the most universal kind), the radio equipment (can get information anytime and anywhere), and the Applet or the Java application program (can be used to realize the complex and fast user interface) and so on.

b. The presentation layer

The presentation layer is also called the Web layer, it runs in the J2EE Web vessel. Its major function is to deal with the HTTP request, and produce the response of HTTP dynamically according to the Servlet and JSP in the Web server.

c. The business logic layer

The different technologies the business logic layer use run in the corresponding vessels. They are mainly used to realize 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 is a series of operations such as inquiring the phone bill, filling the account, calculating the cost and so on. Each enterprise has its unique business logic because of there-own distinctive quality, so it needs to be realized 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.

d. The persistence layer

The persistence layer describes 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 domain.

J2EE programming is extremely simple through using J2EE architecture based on the component and the platform irrelevant, because the business logic is encapsulated into reused component, and J2EE server provides the background services for all components types through using container, it need not develop this kind of service by oneself.

Journal of Global Research Computer Science & Technology (JGRCST)
Vol-I, Issue-II, July 2014
ISSN: 2349 - 5170

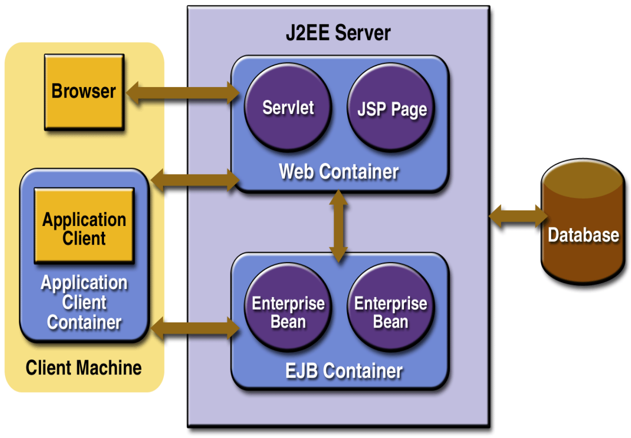

Fig.1 J2EE Architecture

Therefore we may concentrate the energy on solution handy business problems. J2EE provides a variety of developing technologies, there are more commonly used in the following:

i) JSP: JSP technology is a new dynamic web application technology standard. JSP web page is composed of the traditional HTML web page files (*.html, *.htm), which are inserted Java program files (Scriptlet) and JSP tags. Consequently, it comes into being a dynamic page on the server according to the client request.

ii) Servlet: Servlet is a small Java program on the server side and it must realize HttpServlet interface. It can respond and deal with client request through Servlet API, and even it can bring a dynamic HTML page.

e. 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.

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 MVCistory of MVC

Trygve Reenskaugh introduced MVC into SmallTalk-76 while visiting Xerox Parc in the 1970s. In the 1980s, Jim Althoff and others implemented a version of MVC for the SmallTalk-80 class library. It was only later, in a 1988 article in "The Journal of Object Technology", that MVC was expressed as a general concept.

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, MVC remains popular.

5. MVC (Model-View-Controller)

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:

5.1 The Model

The model is the principal part of application program. The model is another name for the application logic layer. 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.

5.2 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.

5.3 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, as shown in Fig.

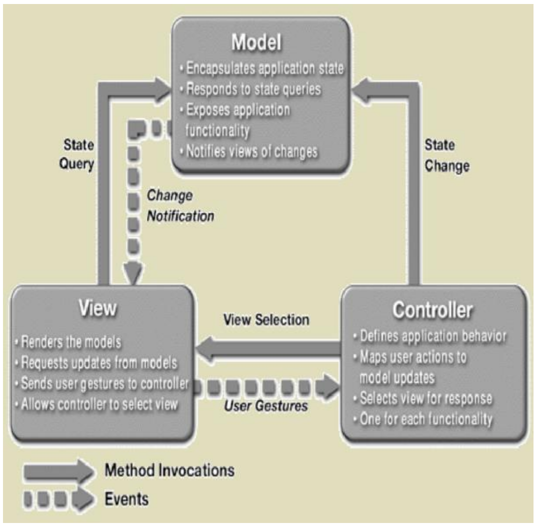
Model
• Encapsulates application state
• Responds to state queries
• Exposes application functionality
• Notifies views of changes
State Query
State Change
Change Notification
View
• Renders the models
• Requests updates from models
• Sends user gestures to controller
• Allows controller to select view
View Selection
User Gestures
Controller
• Defines application behavior
• Maps user actions to model updates
• Selects view for response
• One for each functionality
Method Invocations
Events

Fig. MVC design pattern

Though MVC comes in different flavours, control flow is generally as follows:

1. The user interacts with the user interface in some way (for example, presses a mouse button).
2. 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.

3. 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.)
4. 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.
5. 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.

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.

6. 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.

References

1) Bambara J.J, J2EE Unleashed, Beijing: China Machine Press, 2002.
2) Chuck Cavaness, Jakarta Struts, O'REILLY Publishing, 2004, 151-184.
3) Hangzhou, China, April 6-8, 2008 Research on Data Expression in J2EEArchitecture system
4) Jim Keogh. J2EE Complete Reference, Publishing House of Electronics Industry. 2003.
5) Michael Girdley, J2EE Applications and BEA WebLogic Server, Publishing House of Electronics Industry. 2002.
6) Mike Brevoort, Sr. Associate, Mike Steinley, Sr. Associate, Heather Spinnen weber, Sr. Associate, A Distributed Component Architecture Model, October 2004.
7) The J2EE 1.4 Tutorial, http:// java.sun.com/j2ee/1.4/docs/tutorial/doc/
8) 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.
9) Zhu Xiaoyi, The 3-tier B/S Enterprise Information System Based on J2EE, Journal of Taiyuan University of Technology, Vol.36,