The Design and Realization of Library MIS Based on MVC

Jianhua Li, Fanxing Meng, Xiumei Wen
Hebei Institute of Architecture Civil Engineering
Zhang Jia Kou, China
xiumeiwen@163.com

***Abstract*—MVC is a typical software architecture model The paper states the development principle and advantages of the MVC pattern, discusses the software development process based on the MVC pattern, and designs a Web application—library management system. On the basis of introducing books management system, this paper describes the system design and implementation method based on MVC pattern. The use of MVC model improves maintainability and reusability of the system.**

Keywords- MVC model; JSP; Servlets; Javabeans

I. INTRODUCTION

With the rapid increase of the network applications, MVC paradigm is a very advanced design idea for Web application development. No matter which language you choose, no matter how complex the applications are, it can provide the most basic analysis method for the application model structural, provide a clear designing for products, and provide canonical reference for software engineering. This article raises management system based on MVC paradigm on the basis of research on library management system. MVC separates the input, processing, and output of an application according to the Model, View and Controller, that is, three-tier design mode (namely model layer, view layer and control layer). It is now widely used in software architecture model. The paper uses the JSP+JavaBeans+Servlet+JDBC technologies to realize the MVC architecture. Displaying Web pages is using JSP; realizing business models is using JavaBeans, Controlling the data transmission and function call between foreground and background is using Servlet; Accessing database is using JDBC; storing data is using the database.

II. MVC DESIGN MODEL

MVC was first described in 1979 by Trygve Reenskaug, then working on Smalltalk at Xerox PARC. Model–View–Controller (MVC) is a software architecture, currently considered an architectural pattern used in software engineering. The pattern isolates "domain logic" (the application logic for the user) from the user interface (input and presentation), permitting independent development, testing and maintenance of each (separation of concerns).

A. the basic structure of MVC

In a real application, the three components of MVC are independent and interrelated. The structure and functions of MVC model are as shown in Fig. 1 .

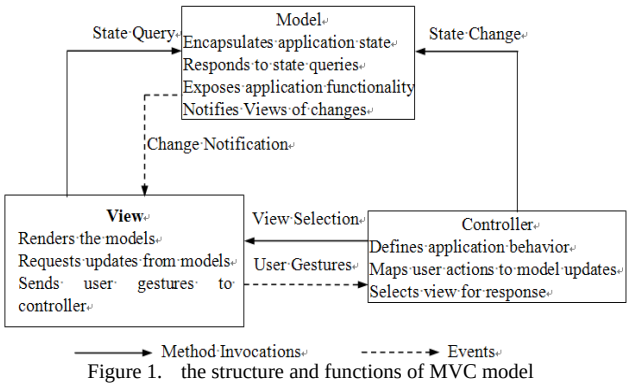

Figure 1. the structure and functions of MVC model

The model manages the behavior and data of the application domain, responds to requests for information about its state (usually from the view), and responds to instructions to change state (usually from the controller). In event-driven systems, the model notifies observers (usually views) when the information changes so that they can react. Design of business model can be said to be the main core of MVC. One of the important models of business model is the data model. Data model is the data saving of the entity objects. For example, an order is saved to the database or is obtained from the database. We can list this model separately, all operations related to the databases are limited to the model.

The view represents the user interface. It can be described as HTML, XHTML, XML and Applet interface for the Web application. The view renders the model into a form suitable for interaction, typically a user interface element. Multiple views can exist for a single model for different purposes. A viewport typically has a one to one correspondence with a display surface and knows how to render to it.

The controller accepts the user request, match model with view to finish the user's request. The function of dividing the control layer is clear, it is a dispatcher for choosing what kind of models, choosing what kind of views and finishing what kind of user requests. The controller receives input and initiates a response by making calls on model objects. A controller accepts input from the user and instructs the model and viewport to perform actions based on that input. The controller cannot do any data processing. For example, users click on a link, the controller accepts request, and it does not handle the business information. It passes the user's information to the model, tells the model what to do, chooses the view that meets the requirements to return to the user.

Published by Atlantis Press, Paris, France.

Therefore, a model may correspond to more views, and a view may correspond to more models

B. the control flow of MVC

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

(1) The user interacts with the user interface in some way (usually the view, but not necessarily), 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 cart's contents).

(5) The view gets its own data from the model. In some implementations, the controller may 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.

The user interface waits for further user interactions, which restarts the cycle.

C. the advantages of MVC

(1) Benefit to division deployment. To develop application using MVC design pattern, personnel division of the different functions is very clear. To use the example of the development of Web application in JAVA language, Java programmers focus on business logic, and interface programmer (HTML and JSP developers) will just concentrate on the page performance forms.

(2) Increase the reuse of the application. For the same Web application, the clients may use multiple ways to access. They can access not only by HTML browser of the computer, but also by WAP browser of the mobile telephone. But no matter by what kind of access mode, the business logic and business process of the application is the same, you need to change the concrete implementation way of the presentation layer. MVC can solve this problem easily because it realized the separation of the business and view.

(3) Reduce coupling and improve the maintainability. MVC design pattern separate the presentation layer and business layer effectively, it reduces the coupling close degree between them. Therefore, any alteration of the business logic cannot affect the code of the presentation layer; the developers can modify the code of the presentation layer freely, and need not rebuild the code of the model and the controller.

D. the disadvantages of MVC

(1) MVC is not suitable for the development and design of small application. MVC developing personnel divide the application fully based on the model, view and control. This can increase some unnecessary works to a few small applications; this affects the exploitation efficiency.

(2) Programming based on MVC design pattern demands the developers must well design the program structure before the programming; because dividing a whole application into three components in the designing process, this increases the number of the file need to manage.

III. SERVELET

Servlets are the Java platform technology of choice for extending and enhancing Web servers. Servlets provide a component-based, platform-independent method for building Web-based applications, without the performance limitations of CGI programs. And unlike proprietary server extension mechanisms (such as the Netscape Server API or Apache modules), servlets are server- and platform-independent. This leaves you free to select a "best of breed" strategy for your servers, platforms, and tools.

Servlets have access to the entire family of Java APIs, including the JDBC API to access enterprise databases. Servlets can also access a library of HTTP-specific calls and receive all the benefits of the mature Java language, including portability, performance, reusability, and crash protection.

Today servlets are a popular choice for building interactive Web applications. Third-party servlet containers are available for Apache Web Server, Microsoft IIS, and others. Servlet containers are usually a component of Web and application servers, such as BEA WebLogic Application Server, IBM WebSphere, Sun Java System Web Server, Sun Java System Application Server, and others.

The Servlet is a Java class, and it communicates and interacts with the model but does not need to generate HTML or XHTML output; the JSPs do not have to communicate with the model because the Servlet provides them with the information—they can concentrate on creating output.

The servlet lifecycle consists of the following steps:

(1) The servlet class is loaded by the Web container during start-up.

(2) The container calls the no-argument constructor.

(3) The Web container calls the init() method. This method initializes the servlet and must be called before the servlet can service any requests. In the entire life of a servlet, the init() method is called only once.

(4) After initialization, the servlet can service client requests. Each request is serviced in its own separate thread. The Web container calls the service() method of the servlet for every request. The service() method determines the kind of request being made and dispatches it to an appropriate method to handle the request. The developer of the servlet must provide an implementation for these methods. If a request for a method that is not implemented by the servlet is made, the method of the parent class is called, typically resulting in an error being returned to the requester.

(5) Finally, the Web container calls the destroy() method that takes the servlet out of service. The destroy() method, like init(), is called only once in the lifecycle of a servlet.

IV. DESIGN AND REALIZATION OF LIBRARY MANAGEMENT SYSTEM BASED ON MVC

A. *Organic combination of JSP＋Servlets＋JavaBeans*

JSP + Servlets + JavaBeans is application example of MVC pattern in JSP development, Organic Combination of three can implement system function well of MVC model.

Library management system is based on MVC pattern; the whole management system uses three-tier architecture of Browser / Web / Database. The clients send requests to Web server through browser, the servlet running in server accept the requests. The servlet in the system can be used as the controller of the application, JSP page is used as view and JavaBean is used as model. Servlet receives all requests sent by the clients; it will call the different JSP page based on the accepted requests. In order to facilitate the operation, sometimes Servlet makes a JavaBean instance used by JSP page based on JSP page's needs. JSP page calls directly method or uses the user-defined labels to obtain the data of JavaBean instance. This method allows the operation results returning to the users in time when users borrow and return the books; this meets the system real-time requirements. In this design model, data transmission of every layer is as shown in Fig. 2 .

In this system, the model is a collection of Java classes that form a software application intended to store, and optionally separate, data. A single front-end class that can communicate with any user interface (for example: a console, a graphical user interface, or a web application).

The view is represented by a JavaServer Page, with data being transported to the page in the HttpServletRequest or HttpSession.

The Controller servlet communicates with the front end of the model and loads the HttpServletRequest or HttpSession with appropriate data, before forwarding the HttpServletRequest and Response to the JSP using a RequestDispatcher.

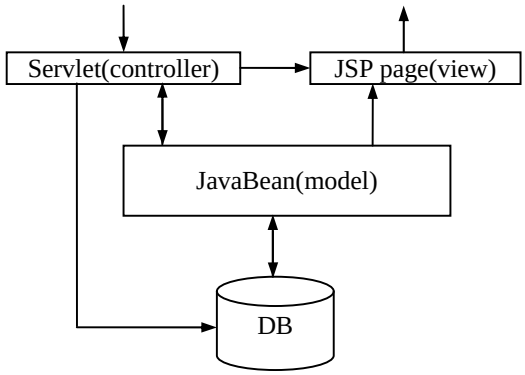

Figure 2. Data transmission in MVC

B. *design and realization of library management system*

The functional requirements block diagram of the library management system is as shown in Fig. 3 ; the information query module is as shown in Fig 4.

The usecase is the interaction between the system and the users; it pays attention to the external users. The users can understand the system functions by usecase. In this system, the usecase is as shown in Fig. 5 .

The codes of the database connection is as follows:

```
// creating the database connection
public Connection getConnection()throws Exception{
Class.forName("com.microsoft.jdbc.sqlserver.SQLServer
Driver");
con=DriverManager.getConnection("jdbc:microsoft:sqlse
rver://localhost:1433;DatabaseName=MyLibrary","sa","123"
);
return con;
}
```

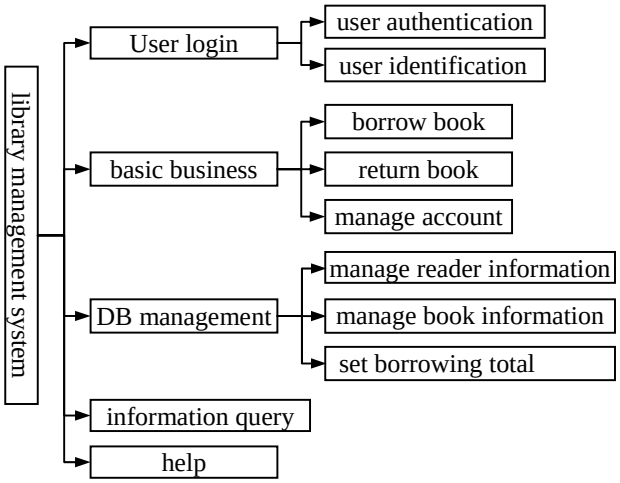

Figure 3. the block diagram of the library management system

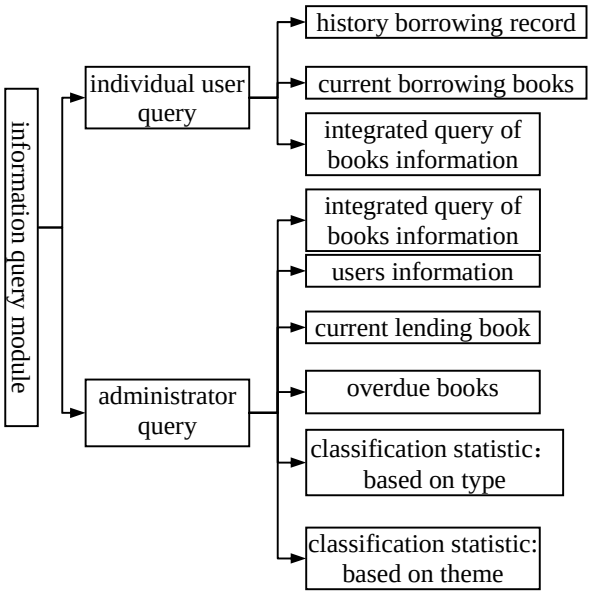

Figure 4. the information query module

V. CONCLUSIONS

MVC design pattern meets the interface demand; separate the calculating model of the software from the interface constitute. It can build and use multi-views when the model is running. The model has portability, and is a potential frame structure. The library management system realized the modules of user login, basic business, DB management and information query, implemented paperless filing, saved the cost. The use of MVC pattern reduces the complexity in architectural design and increases flexibility and maintainability of code, enhanced the development efficiency of the system.

REFERENCES

[1] LI Wei-hua, YAN Jian-jun. Based on the MVC Pattern's Personnel Management System's Design and Realization. JOURNAL OF SANMING UNIVERSITY, 2009,26(4):416-419

[2] Li Rong. Study on the Web Application Based on MVC Mode. Software Guide, 2010,9(1):19-21

[3] Qiuhui Yu. Deisgn and Implemenatiton of Attndance Management Sysetm Based on MVC. Software Guide, 2009,8(10):51-53

[4] Lifeng Fan, Shiquan Qiao, Wenbin Cheng. JSP Programming[M]. Beijing:POSTS & TELECOM PRESS, 2009,5:211-221

[5] Zhen Guo, Guohui Wang. Programming in JavaServer Pages[M]. Beijing:POSTS & TELECOM PRESS, 2009,12:18-19,120-143

[6] YANG S heng-quan, LIU Hai-quan , LIU Ping-ping, YAO Quan-zhu. Design of Electro-archive Management System of MVC Mode. Journal of Xi'an Technological University, 2009,29(3):253-257

[7] Yi Jungang. The Application of Servlet to the Library's Web System. Library and Information Service, 2001,2:55-60

[8] LIN Wen-peng, WANG Chang-yao, CONG Pi-fu. Analysis and Design of the WebGis of Mobile Communicaiton Network Management System Based on Servlet. Application Research of Computers, 2005,2:139-141

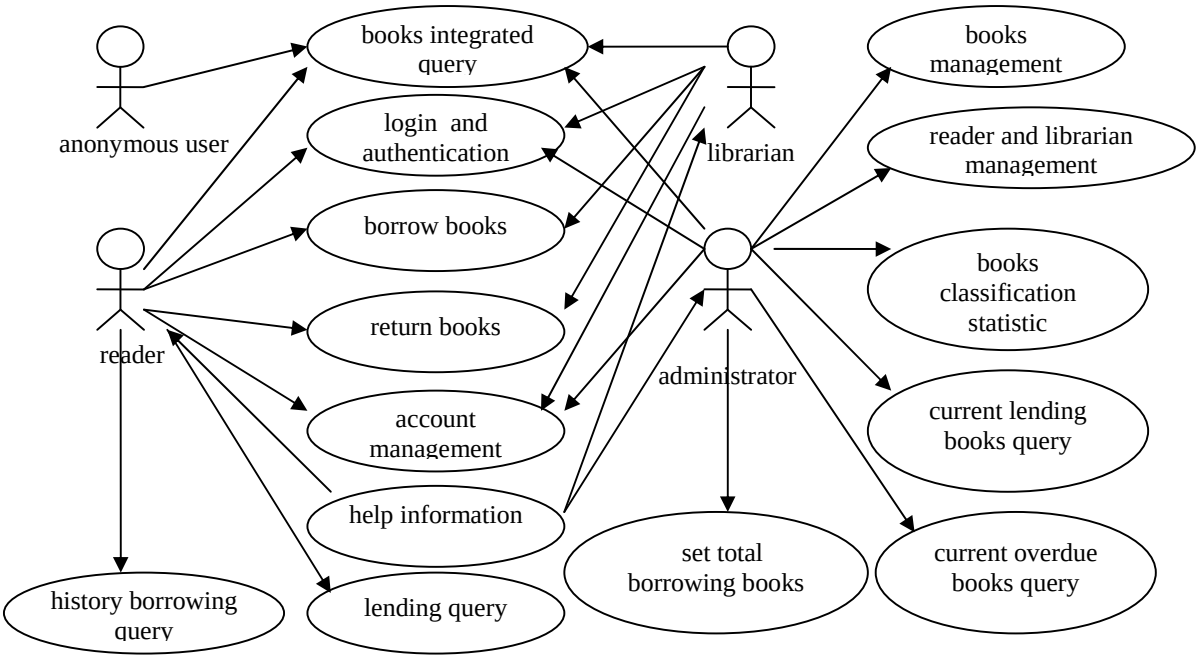

Figure 5. the usecase of library management system