Journal of Computers Vol. 28, No. 6, 2017, pp. 151-162
doi:10.3966/199115992017122806013

The Study and Application of the New Control Layer for Enterprise-Class Web Applications

Bi Liang[1*]

[1] School of Computer Science, Sichuan University of Arts and Science,
No. 400, Nanba Road, Dachuan District, Dazhou City, Sichuan Province, China
liangbi6@126.com

Received 02 August 2016; Revised 31 August 2016; Accepted 07 February 2017

Abstract. With the arrival of the big data era, due to the fact that enterprise-class web applications need to handle with more user-submitted requests so the effective control for a large amount of requests becomes increasingly important. In this paper, we propose the new control layer for enterprise-class web applications by properly combining with the function of C (Controller) component in MVC (Model View Controller) pattern, the cross-cutting idea of AOP (Aspect Oriented Programming) and the layering thought of multi-layer architecture, which is specifically used to receive, parse and dispatch the user-submitted request from the presentation layer. Then we design and implement the control layer mainly through Spring MVC framework and apply it to the case – the control module of the MCRIMS (Movable Cultural Relics Information Management System). At last we compare it with the same case realized by the current popular controller technology-Struts2 under the same circumstance. This case shows that the control layer constructed in this paper is not only practical and feasible, but it also has advantages of easy implementation, good expansibility and fabulous adaptability. What's more, compared with Struts2, the development cycle of the case based on the new control layer is shortened by about 35.71% and the performance turns out to be much better.

Keywords: design and implementation, enterprise-class web applications, multi-layer architecture model, MVC pattern, Spring MVC

1 Introduction

In the current big data era, with the widespread popularity of internet and web applications, the number of users' access to web applications is increasing rapidly, the large and complex enterprise-class web applications become more and more popular, which not only makes it cumbersome to develop web applications in its early stage, but also stands a severe test in extendibility, maintainability, stability and other performances in its later stage. Luckily, the appearance of multi-layer architecture and framework technology effectively alleviates these problems. What's more, faced with thousands of user-submitted requests, it is very necessary for enterprise-class web applications to rapidly accept and handle with these requests, improve the response speed and enhance the user's experience, so the effective control for a large number of requests becomes more and more important in enterprise-class web applications [1].

In this paper, a novel layer (i.e. the control layer) is separated from the traditional business logic layer by effectively combining with the function of C component in MVC pattern, the cross-cutting idea of AOP and the layering thought of multi-layer architecture model, which is specifically used to receive, analyze and forward the user-submitted request from the presentation layer [2-4]. Thus, to some extent, the speed of enterprise-class web applications in response to the request can be increased, the heavy burden of the business logic layer in web applications can be reduced, and the performance of enterprise-class web applications can be effectively improved.

Moreover, according to the role and the task that the control layer plays and shoulders in this paper,

* Corresponding Author

there are a lot of technologies that can be used to implement it at present, such as Servlet, Struts1, WebWork2, Struts2, Spring MVC, Tapestry, ASP, NET MVC, Zend Framework, etc. [5]. Among them, Struts2 becomes the popular implementation tool for the control module of web applications in China, due to its mature, stable, easy-to-test and easy-to-extend and some other advantages as well as rich interceptors. But Struts2 also has some disadvantages, such as hard-to- process requested parameters, being complicated in data validation, low security and poor performance, which have limited the scope of its use in enterprise-class web applications. The emergence of Spring MVC has attracted much attention from developers at home and abroad, because of its simple structure, flexible operation, powerful function and excellent performance. So in this paper we adopt Spring MVC framework to construct the new control layer for enterprise-class web applications, and apply it to the control module of the MCRIMS [6]. And as a result, much more satisfactory results are obtained.

2 Related Theories

The new control layer proposed in this paper involves theories mainly including MVC pattern, AOP idea and multi-layer architecture model.

2.1 MVC Pattern

MVC is a software design pattern, which was invented by Xerox PARC (Xerox Palo Alto Research Center) for the programming language Smalltalk-80 in 1980s and has been widely used at present. In this pattern, the Model processes the data logic of applications, the View handles with the data presentation, and the Controller deals with the interaction between users and web applications [7]. The purpose of using MVC is to make the code separated from the M to the V, so that the same program can use different forms of representation, such as a batch of statistical data can be respectively represented by histogram, pie chart or line graph and etc. The C is to ensure the synchronization between the V and the M, namely, once the M is changed, the V will be updated accordingly. The relationship among the three components (the M, the V and the C) is shown in Fig. 1. The core idea of MVC is to separately implement the view function, model function and control function of the application in different parts (also called layers), which can increase the reusability of the code, reduce the coupling degree between the data description and the application operation, and as a result improve the readability of the code and the expansibility and maintainability of the application.

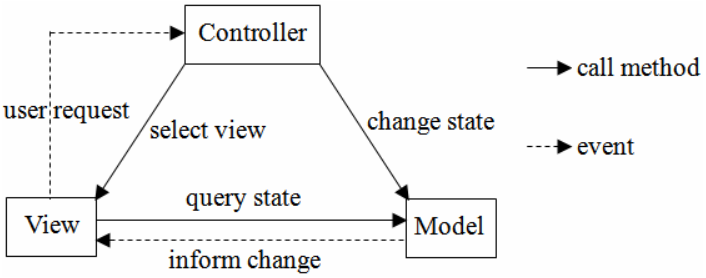

Fig. 1. MVC pattern

2.2 AOP Idea

Based on OOP (Object Oriented Programming), AOP is a design pattern and a programming method proposed to make up for the deficiency of OOP in dealing with the problem of scattering and entanglement and to solve other problems such as poor object assembling and inability of handling with cross-cutting relationships and the like. The core idea of AOP is to extract the program concerns crosscutting in the software system, and then regard it as a module of the software system that can be completed independently in the design and the implementation stage, so it can improve the code's abstraction and modularity. Fig. 2 shows the relationship between cross-cutting concerns and the business logic [8]. From Fig. 2, using AOP can isolate various parts of the business logic, so as to reduce the coupling degree between each part of the business logic, and improve the reusability and the development efficiency of application program. Like OOP, AOP itself is not a programming language,

but a kind of software design idea, whose appearance is not to replace OOP, but to complement and make OOP perfect.

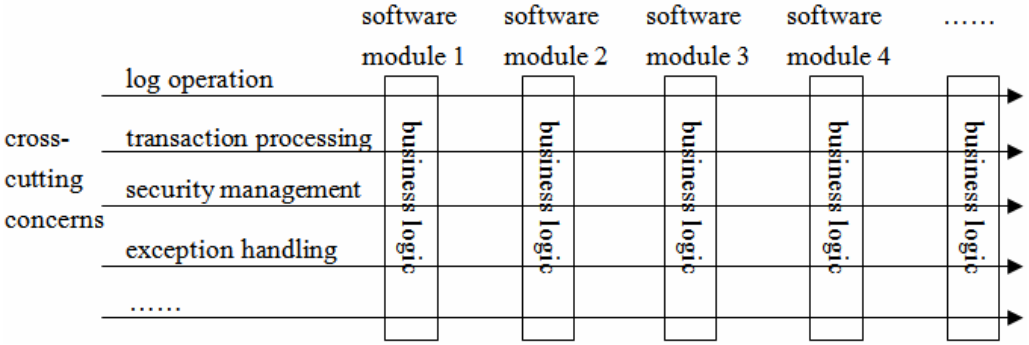

Fig. 2. The relationship between cross-cutting concerns and the business logic

2.3 Multi-layer Architecture Model

Multi-layer architecture is a kind of strategies based on the isolation-control, which is adopted by developers who face the complex and easy-to-change users' needs in the process of developing software, and each isolation layer can be deployed independently. Three-layer architecture is a classic multi-layer architecture model, including the presentation layer, the application layer and the data layer. Among them, the presentation layer is the layer that users can directly get access to and interact with, such as the desktop UI (User Interface), web pages and so forth. The application layer encapsulates the business logic, the domain concept, the data access logic and the like. The data layer is used to get access to the external data source of the application data like the database service, the CRM (Customer Relationship Management) system, the host and so on. Some layers in three-layer architecture can be further divided to makes it multi-layer architecture, for example, the application layer can be further divided into the business logic layer and the data access layer. Presently, the mainstream multi-layer architecture is shown in Fig. 3 [9]. Of course, all of these logical layers can be deployed on the same computer.

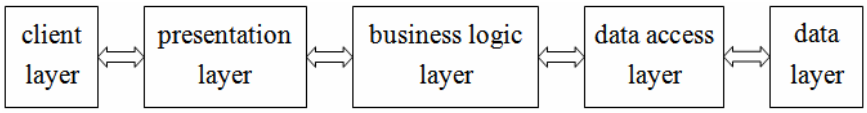

Fig. 3. Multi-layer architecture model

3 The Key Technology

The key technology used to realize the new control layer in this paper is Spring MVC, which is a kind of Java web framework technology.

3.1 Introduction to Spring MVC

Spring MVC belongs to the follow-up product of Spring framework, and has been integrated into Spring Web Flow. It is a detailed and complete MVC framework, through which MVC pattern is implemented and as a result the data, the business as well as the presentation are effectively separated. Among them, the Controller is mainly responsible for coordinating the control among components, and DispatchServlet is responsible for reading the configuration file and finding the corresponding Controller component with HandlerMapping. The Model is constructed by the BeanForm of the application system state or the JavaBean of the business logic. The View is realized by Jsp, Velocity or other presentation layer technology [10]. In addition, Spring MVC adopts an elegant method of view analysis, its controller returns a ModelAndView object, and the object's attribute only is the name of the view and the model, where the model provides the name of the bean and the corresponding relationship with the object. Moreover, this technology makes full use of the advantages of Spring's Non-intrusive programming so that the developer only needs to configure the XML (Extensible Markup Language) file without

implementing any related Spring interfaces. At the same time, it can greatly reduce the amount of code of web applications by effectively using Spring's Annotation. Because of these gorgeous features, Spring MVC has been widely applied by domestic developers.

3.2 The Core Component of Spring MVC Framework

Spring MVC framework is mainly composed of DispatchServlet, HandlerMapping, Controller, ModelAndView, ViewResolver, View, HandlerAdapter, Handler, HandlerInterceptor and other components, in which the core components are as follows [11].

DispatcherServlet. It is implemented by the traditional Servlet technology, and used to control and navigate the complete flow of the application [12]. Besides, it is the first controller which interacts to the request, so the web.xml receives the request and transfers the request to it.

HandlerMapping. It maps incoming requests to handlers and a list of pre-processors and post-processors (handler interceptors) based on some criteria, the details of which vary by HandlerMapping implementation [13]. The most popular implementation supports annotated controllers, but other implementation exists as well.

Controller. Being responsible for handling with requests that are dispatched by DispatcherServlet, it encapsulates the user-requested date after the business logic layer is processed into a Model, and then returns the Model to the corresponding View for display [14].

ModelAndView. Being created by the Controller, ModelAndView associates the View to the request and stores the business logic and Model data [15]. When a Controller calls it, it will execute and return the data and name of the View.

ViewResolver. ViewResolver resolves logical String-based View names to actual View types. It identifies and implements what is the output media and how to display it.

View. It is responsible for rendering output. Different Views can be selected for different types of output based on the results, the viewing devices and the communication devices.

Moreover, DispatcherServlet is not only the control center of Spring MVC framework and the unique access entrance of all HTTP (HyperText Transfer Protocol) requests, but also the center of other designs.

3.3 The Processing Flow of Spring MVC

Spring MVC belongs to the web framework driven by the request. Its main processing flow is shown in Fig. 4 [16], when it receives the user-submitted HTTP request from the presentation layer.

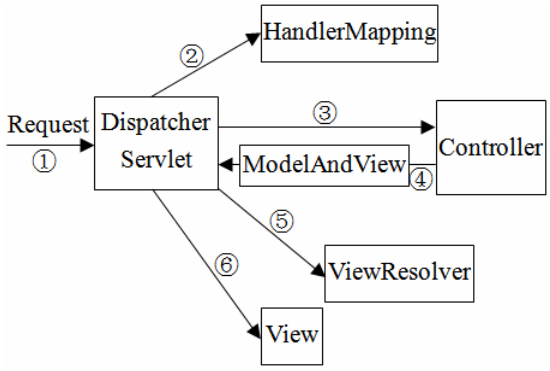

Fig. 4. The main processing flow of Spring MVC

From Fig. 4, the concrete procedures of the Spring MVC' processing requests is as follows:

(1) Spring MVC submits all HTTP requests to the DispatcherServlet, and delegates other modules of web applications to process these requests really.

(2) The DispatcherServlet queries one or more HandlerMapping to find the specific Controller for processing request.

(3) The DispatcherServlet submits the request to the target Controller.

(4) The Controller calls the corresponding business logic method for processing business and returns a ModelAndView.

(5) The DispatcherServlet queries one or more ViewResolver to find the View specified by the ModelAndView.

(6) The View is responsible for rendering and showing results to the user of the presentation layer.

4 The Study of the New Control Layer

The study on the new control layer for enterprise-class web applications is the key point of this paper, because a good design of the control layer can not only improve the development efficiency of enterprise-class web applications, but it can also improve the running speed of enterprise-class web applications. The four aspects listed below, namely the function, the design, the implementation and the sequence, are used for demonstrations.

4.1 The Function of the Control Layer

With the rapid growth of the number of user requests, the controller is becoming increasingly popular in web applications, and it is also getting more and more attention from web designers or web developers both at home and abroad. In this paper, the new control layer for enterprise-class web applications proposed by properly combining with MVC pattern, AOP ideal and multi-layer architecture model is the controller itself, and it is also a cross-cutting concern in enterprise web applications. Besides, it is a special layer, which is between the presentation layer and the business logic layer. So it not only reduces the coupling of components in web applications, but also acts as a "bridge" role between the user request and the business logic processing. The control layer is mainly responsible for controlling the request flow, completing the function of the C component in MVC pattern itself, which is responsible for receiving, analyzing and distributing all user-submitted HTTP requests from the presentation layer [17]. According to different requests, the control layer calls the corresponding business logic processing method to deal with the request, and selects the corresponding view to return to the user based on the current state data and the result of the business logic processing, thus it successfully completes the interaction between users and enterprise-class web applications.

4.2 The Design of the Control Layer

According to the function of the control layer, it mainly focuses on controlling the request flow of enterprise-class web applications and completing the correct jump of the request. In addition, the specific logic of web applications is encapsulated in JavaBean of the business logic layer, which can improve the flexibility and reusability of enterprise-class web applications. In order to enhance the rationality and feasibility of the control layer, in this paper, appropriately combining with MVC pattern, AOP ideal & multi-layer architecture model, we use the DispatcherServlet, the Controller, the spring-servlet.xml and other components in Spring MVC framework to construct it. The concrete architecture design is shown in Fig. 5 [18].

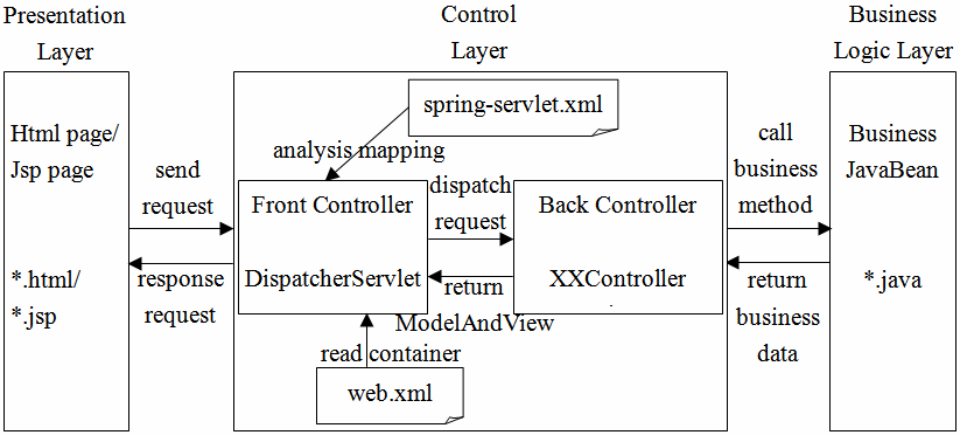

Fig. 5. The architecture of the control layer based on Spring MVC framework

In Fig. 5, the Front Controller is the core controller of the control layer, which uniformly distributes all user-submitted HTTP requests from the presentation layer by the DispatcherServlet, and sends the requests to the corresponding Controller Class in the Back Controller according to the spring-servlet.xml's configuration information. The Controller Class acts as a role of "adapter" between the HTTP request and the business logic processing. This Class handles the request by calling the business method of the business logic layer, so it mainly controls the flow of enterprise-class web applications rather than really realizes the business logic functions of web applications. Besides, the web.xml is the container that is used to initialize the configuration information of enterprise-class web applications.

So, based on Spring MVC framework, we reasonably construct the control layer for enterprise-class web applications. It can correctly control the user-submitted request, and reduce the burden of the business logic layer in web applications. Moreover, it is effectively separated from the upper presentation layer and the lower business logic layer, and realizes high cohesiveness in layers and low coupling between layers, which is conducive to constructing enterprise-class web applications with high flexibility, good reusability, easy deployment, extension and maintenance.

4.3 The Implementation of the Control Layer

In this paper, Spring MVC and related technologies are adopted to implement the constructed web control layer. First of all, the Front Controller, responsible for managing the distribution assignment of all user-submitted HTTP requests, is configured in the web.xml using <servlet> element and <servlet-mapping> element, which is implemented by org.springframework.web.servlet.DispatcherServlet. As long as the service starts, the spring-servlet.xml is loaded, and the environment of web applications is configured automatically. At the same time, classes marked @Service, @Compoments, @Repository and @Controller are scanned in the class path, and they are dynamically injected into Spring container for the unified management. After the container packages requested parameters, the user's request will be distributed to the specified Back Controller. Then Spring MVC maps the URL and the request method by @RequestMapping, obtains the parameters in URL by @PathVariable and @RequestParam, and returns the JSON (JavaScript Object Notation) string through @ResponseBody. At last, the standard string, which is realized after the Jackson package serializes the Java object, is returned to the presentation layer. Thus the function of the control layer in enterprise-class web applications is realized successfully.

4.4 The Request Processing Sequence of the Control Layer

According to the design of the control layer for enterprise-class Web applications and the request processing flow of Spring MVC, the request processing sequence of the new control layer is shown in Fig. 6 [19].

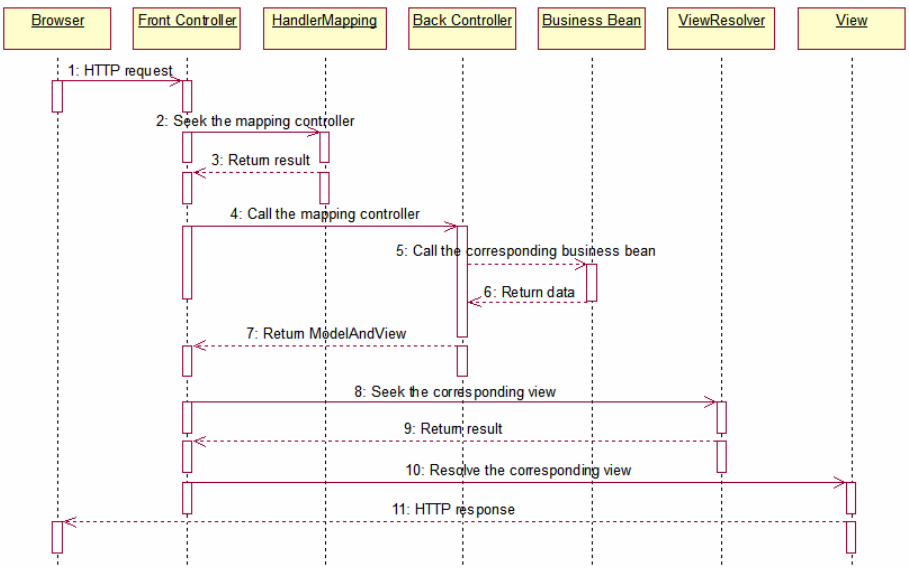

Fig. 6. The request processing sequence diagram of the control layer

5 The Application of the New Control Layer

The application of the new control layer is another key point of this paper, as only the control layer that meets the development demands of enterprise-class web application is of the practical use.

5.1 A Case Description

Cultural relics are artifacts and sites, which have historical, artistic and scientific values about the development of human history. It can be divided into the movable cultural relics and the immovable cultural relics. Between them, the movable cultural relics refers to the collection of cultural relics, that is, the important object, art, literature, manuscripts, books, representative objects and so on in different periods of the history. China with a long history of civilization is abundant in the movable cultural relics, such as the Hongshan Culture ERON, the Horse-riding Chebi, the Stepmother Wu Ding, the Riverside Scene at Qingming Festival, the Heshibi and the Lanting Pavilion Sequence, which are of high archaeological research value and collection value. For the government, the department for cultural relics management, and the archaeological and cultural relics researchers, using the traditional manual way to manage a large amount of the data about the movable cultural relics cannot meet their needs. So it is necessary to develop the MCRIMS by the modern computer technology, the network technology and the web technology. In this paper, the constructed control layer architecture is applied to the control module of the MCRIMS realized by Spring, Spring MVC, XML and other technologies, and through this case, more details about the concrete implementation process of the control layer are declared.

5.2 The Implementation of the Control Layer of the Case

To build the development environment. According to the implementation process of the control layer constructed in this paper, we firstly need to build the development environment for the control layer of the MCRIMS, and take MyEclipse 8.5 as the development platform, Tomcat 7.0 as the server, Java as the basic language, Spring as the technical basis, Spring MVC as the key technology and so on to develop it. The specific software and hardware environment is shown in Table 1.

Table 1. Basic development environment

Category	Basic Configuration
Hardware	Asus 15.6" Laptop, Intel Core i5-5200U 2.2GHz, 4GB RAM, 500GB HDD, 100 Mb/s Ethernet, etc.
Software	Microsoft Windows 7, SQL Server 2008, JKD 1.7, MyEclipse 8.5, Tomcat 7.0, Spring 3.2, Spring MVC 3.2, Struts2.3, Dreamweaver 8, etc.

Then, we import required Jars into the MCRIMS, such as spring-core-3.2.0.RELEASE.jar, spring-beans-3.2.0.RELEASE.jar, spring-context-3.2.0.RELEASE.jar, spring-tx-3.2.0.RELEASE.jar, spring-web-3.2.0.RELEASE.jar, spring-webmvc-3.2.0.RELEASE.jar, spring-webmvc-portlet-3.2.0.RELEASE.jar and so on, which will play a role when the MCRIMS is running.

To configure the Front Controller of the case. The web.xml file is used to initialize the configuration information in the MCRIMS, such as welcome page, servlet, servlet-mapping, filter, listener, etc. At the same time, it can also include the configuration of some initialization parameters, but some parameters specify the additional XML configuration files to be loaded, which can affect its behavior. In the control layer of the MCRIMS, We mainly use the web.xml file to complete the configuration of the Front Controller. It is the central controller for the entire web application, which is used to intercept all user-submitted HTTP requests, and the task is really finished with the DispatcherServlet. Its core configuration is shown as follows [20]:

```
configure <servlet>:
begin
  <servlet-name>:='springMVC'
  <servlet-
class>:='org.springframework.web.servlet.DispatcherServlet'
    configure <init-param>:
    begin
      <param-name>:='contextConfigLocation'
```

```
        <param-value>:='classpath:*/spring-servlet.xml'
      end
    <load-on-startup>:=1
  end
  configure <servlet-mapping>:
  begin
      <servlet-name>:='springMVC'
      <url-pattern>:='/ or /*'
  end.
```

To develop the Back Controller of the case. The Back Controller of the MCRIMS, known as the business logic controller, is a single HTTP request processing controller. It is also the main task that the developer personally needs to complete. The Developer codes the corresponding Controller Class based on different user-submitted HTTP requests, and uses @Controller to define a Back Controller, uses @RequestMapping to map the specific request to the Back Controller. The Controller Classes of the MCRIMS mainly are ImageControl.java, Document Control.java, VideoControl.java, CulturalRelicControl.java, NewsControl.java, MessageControl.java, ViewControl, UserControl.java, AdminControl.java, etc. Among them, the pseudo code of the Video Control class is shown as follows.

```
open Controller Service: @Controller
class VideoControl
begin
  open Autowired Service: @Autowired
  var videoService: VideoService;
  open RequestMapping Service:
  @RequestMapping(value:='/showVideo/id:={id}')
  method getContextByid(id: Integer, request: HttpServletRequest)
  begin
    var getVideo:=videoService.getVideoByid(id);
    request.setAttribute('getVideos', getVideo);
    return ModelAndView;
  end
end.
```

To configure the spring-servlet.xml of the case. The spring-servlet.xml of the MCRIMS specifies the process to be executed for the HTTP request, such as the path, the class, the view and others, and then completes the correct navigation of the request. In this configuration file, the ViewResolver is mainly configured. It provides a mapping from the view name to the actual view, which means a process to add a prefix to the view name. The View handles with the preparation work of the request, and submits the request to a specific view technology, such as Jsp. The main spring-servlet.xml configuration of the MCRIMS is shown as follows [21]:

```
begin
  <beans xmlns:mvc:='http://www.springframework.org/schema/mvc',…
  xsi:schemaLocation:='http://www.springframework.org/schema/mvc,
  http://www.springframework.org/schema/mvc/spring-mvc-3.2.xsd',… />
  open <mvc:annotation-driven>
  open <context:component-scan base-package:='com.web' />
  configure HandlerAdapter:
  <bean class:='org.springframework.web.servlet.mvc.annotation.
  AnnotationMethodHandlerAdapter' />
  configure resources:
  <mvc:resources location:='/resource/' mapping:='/**' />
  configure ViewResolver:
  <class:='org.springframework.web.servlet.view.
  InternalResourceViewResolver' p:prefix:='/' p:suffix:='.jsp'… />
  …
end.
```

Finally, according to the configuration information and the requirement of the MCRIMS, the pages of the presentation layer are realized by Jsp, HTML5 and other technologies. The pages are used to display the response results from the control layer of the MCRIMS, such as index.html, success.html, fail.html, login.jsp, register.jsp, show.jsp, image.jsp, news.jsp, video.jsp, message.jsp and etc.

6 The Analysis of the Experiment Results

For the development environment in Table 1, we adopt the control layer presented in this paper to design the control module of the MCRIMS (that is the control layer of the MCRIMS), and implement it by Spring MVC's related technologies, finally make a comparison with the same scale and function of the case developed in the same development environment by Struts2 in aspect of the development cycle, the number of code lines and the software size and etc, the specific data obtained are shown in Table 2.

Table 2. Comparison of the case

The control module of the MCRIMS	Number of developers	Number of configuration file types	Development cycle (days)	Number of code lines (KLOC)	Software size (GB)
Based on Struts2	1	2	14	1.01	0.21
Based on the control layer	1	2	9	0.49	0.10

It can be seen from Table 2, compared with Struts2, the number of code lines of the case based on the control layer becomes much less and also the development cycle of the case is shortened by about 35.71%.

Moreover, we use the HP' LoadRunner tool to test their performance respectively [22], and the testing results are shown in Table 3.

Table 3. The results of performance testing of the case

The control module of the MCRIMS	Total number of requests	Number of concurrent users	Min response time (s)	Max response time (s)	Average response time (s)	Throughput (Req/Sec)	CPU utilization (%)	Transaction pass rate (%)
Based on Struts2	6000	300	4.15	6.34	5.12	1118	72	91
Based on the control layer	6000	300	3.20	5.37	4.24	1406	63	99

From Table 3, we can know that the results of performance testing of the case based on the control layer is better than that of the case based on Struts2 under the same conditions. Besides, the throughout of the former is about 1.26 times than that of the latter.

And for different numbers of concurrent users, the results of their performance testing on average response time, throughput, CPU utilization and transaction pass rate are shown in Fig. 7 to Fig. 10.

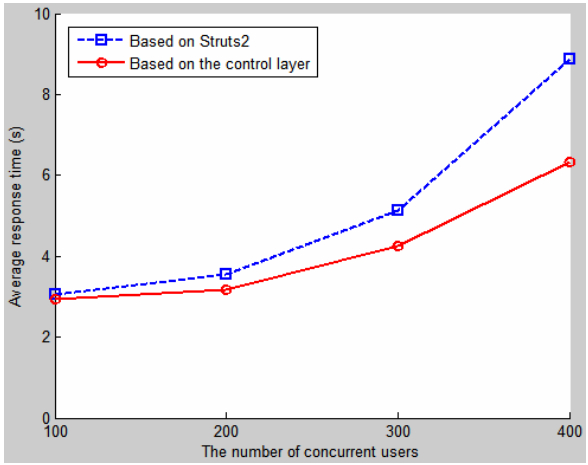

Fig. 7. The average response time for different numbers of concurrent users

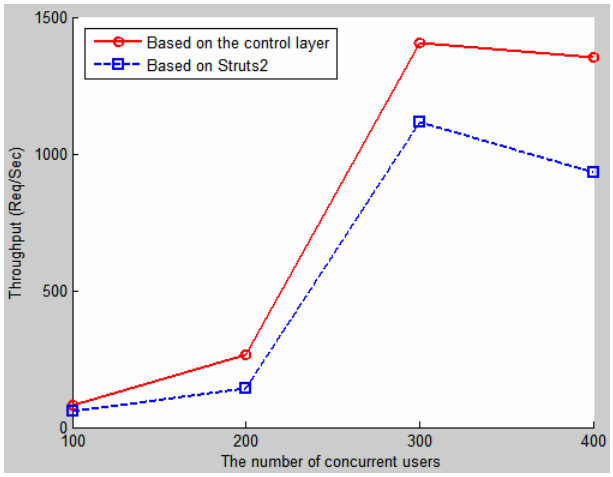

Fig. 8. The throughput for different numbers of concurrent users

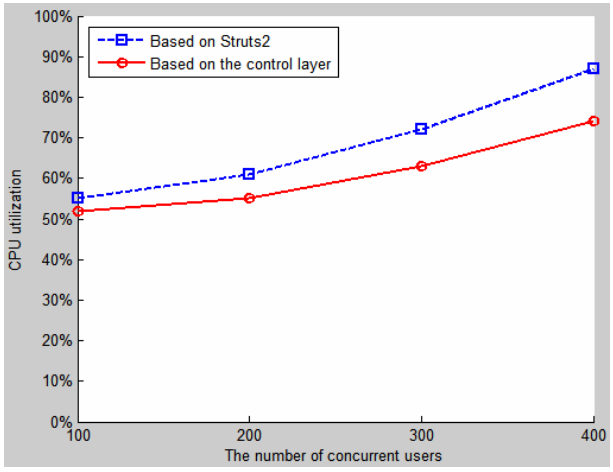

Fig. 9. The CPU utilization for different numbers of concurrent users

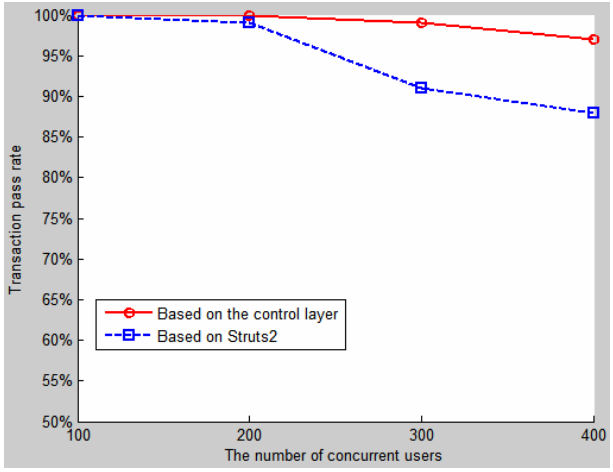

Fig. 10. The transaction pass rate for different numbers of concurrent users

From Fig. 7 to Fig. 10, we find that in the performance aspect the control module of the MCRIMS based on the control layer is better than that based on Struts2. In addition, the similar results of performance testing are able to be obtained by applying this new control layer to other enterprise-class web applications, such as online shopping system. From here we know that the control layer declared in this paper can effectively deal with a large amount of concurrent requests, so it is very suitable for the

requirement of enterprise-class web applications for mass users now.

7 Conclusions

Presently, with the arrival of the big data era and the popularity of enterprise-class web applications, the explosive growth of the amount of users' accessing to web applications happens, which makes the function of the control layer become more and more important. In this paper, we not only design and implement the new control layer for enterprise-class web applications, but also apply it to the control module of the MCRIMS, as well as compare with the same case developed by Struts2. The practical development proves that the control layer constructed in this paper is not only feasible, but it also has the following advantages to a certain extent:

(1) Clear division of roles: In this control layer architecture, there are the Front Controller, the Back Controller, the request mapping and other roles, and they complete their respective tasks alone.

(2) Simple configuration: For the application of the control layer, using Annotation, IoC/DI (Inversion of Control/Dependency Injection) and other Spring MVC's technologies can simplify the configuration file and the management of Beans.

(3) Fabulous adaptability: According to different users' requests, the developer can create the Back Controller needed by any request, which can improve the adaptability of web applications.

(4) Great extensibility: Different roles are encapsulated as components, and provide developers with interfaces, which is conducive to the extension of web applications in its later stage.

(5) Excellent performance: From the Table 3, the web applications based on the control layer in this paper perform very well, respond at fast speed and meet the needs of users in a better way.

But, the following disadvantages also exist:

(1) The Back Controller depends on the Servlet API (Application Programming Interface), which makes it inconvenient to perform the unit test of the controller.

(2) The emergence of the control layer makes the traditional three-layer architecture into the four-layer architecture, which increases the design complexity of enterprise-class web applications.

(3) The enterprise-class web application based on the control layer reduces the configuration of the XML file mainly through Annotation, which reduces the readability of the system code to some extent.

In a word, pros and cons exist together if we develop enterprise-class web applications based on the control layer proposed in this paper, but it reduces the code, shortens the development cycle, and the developed web applications has high flexibility, good performance and other advantages, which provide superiorities for the development of large-scale or medium-scale enterprise-class web applications. As a result, the web applications based on the control layer better meet the requirements of the current enterprise-class and multi-layer web applications in the age of big data.

Acknowledgements

This work is partially supported by the Science and Technology Project of Archives Bureau in 2014 (2014-X-65), and the General Project of Education Department of Sichuan Province in 2016 (16ZB0362).

References

[1] W.-X. Li, X. Wang, Research and application of web system development architecture based on Extjs and Spring MVC, Journal of Yunnan University 35(2)(2013) 110-115.

[2] M.-M. Zhou, Y.-B. Dong, M.-L. Zhu, The design of a model for developing web controller based on MVC, Computer Application and Software 22(12)(2005) 96-104.

[3] K. Hokamura, N. Ubayashi, Sh. Nakajima, A. Iwai, Aspect-oriented programming for web controller layer, in: Proc. the 2008 15th Asia-Pacific Software Engineering Conference, 2008.

[4] Y.-J. Zhang, Design and realization of the project management system based on multilayer architecture, [dissertation] Wuhan, China: Wuhan University of Science and Technology, 2015.

[5] Y. Zhang, Y.-H. Wang, X.-N. Zhang, Design and implication of MVC framework based on spring, Computer Engineering 36(4)(2010) 59-62.

[6] G. Shao, Optimization and improvement of MVC controller based on spring framework, [dissertation] Jinan, China: Shandong University, 2011.

[7] E. Gamma, R. Helm, R. Johnson, J. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley Professional, New York, 1995.

[8] J. Cao, The global reconstruction of enterprise information system based on aspect oriented programming, [dissertation] Shanghai, China: Shanghai Jiao Tong University, 2014.

[9] Z.-G. Yang, Design and implementation of enterprise office automation system based on three Layer architecture, [dissertation] Chengdu, China: University of Electronic Science and Technology of China, 2015.

[10] L. Liu, Construction the traveling electronic commerce platform based on Spring MVC, [dissertation] Shanghai, China: Fudan University, 2009.

[11] G.-L. He, Design and implication of the vocational college internship management system based on Spring MVC, [dissertation] Kunming, China: Kunming University of Science and Technology, 2014.

[12] W. Lin, The design and implication of data synchronization system based on Spring MVC structure, [dissertation] Jilin, China: Jilin University, 2015.

[13] F. Xue, F. Liang, S.-X. Xu, B.-R. Wang, Research on Spring MVC framework based on web and its application, Journal of Hefeil University of Technology 35(3)(2012) 336-340.

[14] P. Gupta, M.C. Govil, Spring Web MVC framework for rapid open source J2EE application development: a case study, International Journal of Engineering Science and Technology 2(6)(2010) 1684-1689.

[15] H. Yuan, The development of campus shop system based on Spring Web MVC, [dissertation] Guangzhou, China: South China University of Technology, 2014.

[16] Z.-Y. Yu, Design and implementation of information management system based on Spring MVC, [dissertation] Chengdu, China: University of Electronic Science and Technology of China, 2013.

[17] L. Zhang, Application of paging component based on Model-View-Controller pattern, Computer Engineering 37(21)(2011) 255-257.

[18] B. Liang, D.-J. Liu, L.-L. Xiao, Research on the construction of immovable historical relics digital platform based on five-layer architecture, Computing Technology and Automation 4(4)(2014) 115-118.

[19] Y.-C. Ren, Z.-F. Xing, T. Xing, J.-C. Zheng, Application research for integrated SSH combination framework to achieve MVC mode, in: Proc. the 2011 International Conference on Computational and Information Sciences, 2011.

[20] T. Zhang, Research and realization of web system development architecture based on Extjs and Spring MVC, Computer Technology and Development 23(1)(2013) 147-149.

[21] D.-D. Zhang, Z.-Q. Wei, Y.-Q. Yang, Research on lightweight MVC framework based on Spring MVC and MyBatis, in: Proc. 2013 Sixth International Symposium on Computational Intelligence and Design, 2013.

[22] L. Yuan, Performance testing and optimization for online reservation web system, [dissertation] Xidian, China: Xidian University, 2015.