REVIEW OF COMPUTER ENGINEERING STUDIES

ISSN: 2369-0755 (Print), 2369-0763 (Online)
Vol. 3, No. 2, June 2016, pp. 43-46
DOI: 10.18280/rces.030204

A publication of IIETA

IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering
http://www.iieta.org/Journals/RCES

The Design Method of Embedded Web Based on Model-View-Controller Pattern

Wei Dengfeng

Computer Science College, Yangtze University, Jingzhou 434023, China

Email: weidengfeng@126.com

ABSTRACT

In order to improve development efficiency of embedded Web application and code reusability, a kind of development mode of embedded Web based on MVC pattern is presented and a kind of practical application template engine is designed based on this pattern. This design introduces MVC method and Web application development is divided into model, view and controller to achieve the separation of logistic and presentation layer in the process of embedded Web design, thus it achieves the modularization of embedded Web application design and clear hierarchical structure. Developing Web application based on this method can improve code reusability and strengthen expandability and maintainability of system. This mode is a kind of common design method and can be applied to various system platform according to actual application environment.

Keywords: Model-view-controller, Embedded web, Template engine.

1. INTRODUCTION

With the development of embedded technology and gradual popularity of embedded equipment, there are more and more network product based on embedded system that provides with Web and CGI interface for user to make user query or set up systematic parameters through the Internet of remote browser [1]. At present, these CGI applications are achieved by using one CGI language (such as various script languages and C language). Their disadvantages include staggered confusion of codes between static page and dynamic data and distortions of program structure, which are bade for independent maintenance of pages and business model and are difficult to satisfy variation requirements of user. This paper achieves a kind of design template engine based on embedded system and MVC. Through a set of insert HTML scripts, interface design and dynamic data operation is separated and separates business model from page and improves reusability of model [2].

2. MVC DESIGN CONCEPT

MVC namely is Model-View-Controller (model-view-controller). The model is application object, includes core development can compulsively separate data (model) and interface (view) [7]. At present, there are some Web program development tools based on MVC. Although these tools are widely applied to website design, two problems limit the application of it on embedded system. Firstly, such generated Web applications are based on server-side script [8]. data of problem and is independent of external forms of display. View obtains information from model to present data in the model and logistic relationship in different manner. Controller defines response mode of user interface on user input, properly transfers business logic according to client request of user and displays selected view to user based on result. Among them, there are most processing tasks in view [3].

MVC separates view and model by establishing one "order/notice" protocol [4]. View must accurately reflect status of model. In the event of changes of data in the model, the model will inform view of update in a manner. The model returns unformatted data, namely, the model is not associated with data format. There are different views in one model. When establish new views, model is not rewrote [5]. MVC is intended to increase code reusability and decreases coupling degree of data expression, data description and application operation. Figure 1 describes the relationship of view, controller and model [6].

3. THE APPLICATION OF MVC ON WEB PROGRAMMING

Applying MVC design concept to Web application However, due to smaller storage space, embedded system generally selects simple and compact Web server but does not support server-side script. Secondly, using server-side script means to dynamically analyze and explain when call page and the running speed on the embedded equipment with lower performance is low.

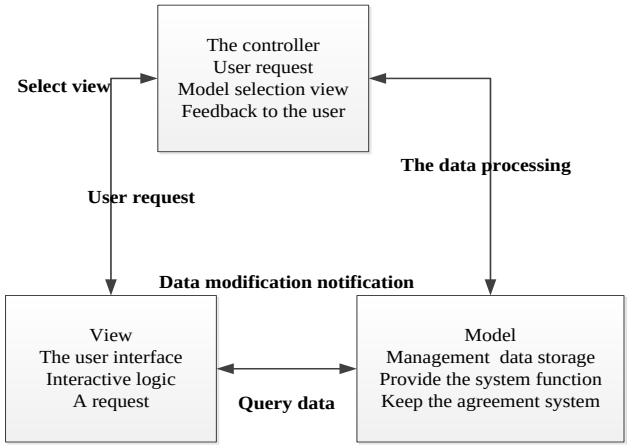

Figure 1. Architecture of MVC and relation between MVC components

Thus, to achieve MVC in embedded Web design, another method must be adopted [9]. Through analysis, this paper chooses CGI program as purpose to achieve CGI program template engine based on MVC. CGI is s up ported by most Web server and execution speed is higher so that it is suitable for embedded Web application. This design adopts traditional template to complete the design of view, controller and model. The view then is transmitted into corresponding CGI program. At last, it is combined with model and controller to form final CGI program. The design structure is as shown in Figure 2.

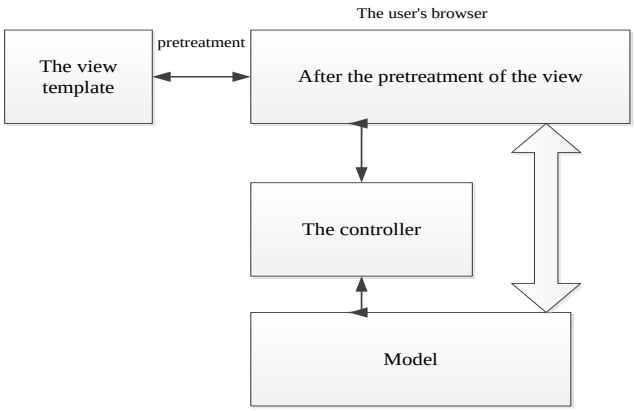

Figure 2. Architecture of web application based on MVC

4. THE DESIGN OF WEB TEMPLATE ENGINE BASED ON MVC

The biggest disadvantage of CGI program which is written in any language is that the code is not easy to read or maintain [9]. Due to static content in the page, dynamic data and control flow and function call in the program are mixed together. So code readability is poor and code is difficult to identify by DreamWeaver graphical design tool and to maintain interface [10].

Introducing MVC concept into CGI program design is to separate the interface, control and dynamic data to the maximum extent when design and maintain CGI program so as to make interface designer and programmer concentrate on code on the HTML layer and business layer without mutual influence. The code integration is automatically completed by template engine.

```
<? pseudocode
  print_table($
source,$col,$ border)
  while$count>0
    do
  print($ count)
  $count=$ count—1
    done
  ?>
```

Figure 3. Sample of built in script

4.1 View design

View namely is user interface. Using HTML layer language to encode view is mainly intended to help art designer use graphical tools to design page layout and static elements. Built in script is used to design dynamic content and a pair of specific symbols (hereinafter known as script domain) is included. It is better to select symbols neglected by HTML, such as “<?---->”and “< !! >”. In this case, the content of script domain will not affect appearance of other components when design page. The script content shall include sentence of variable definition and reference, function call and flow control etc.

(1) Variable. Variable is used for storage of dynamic data and includes data shown in interface and used inside script. The application of variable includes assignment and reference. In the following example, pseudo-code name is used to present variable. In general, the assignment of other variable other than variable used inside view is processed in the control module and the variable is only used in the view design.

(2) Function. Function is used to achieve a set of special function. For example, print form, read and write file and calculate expression. The function is achieved in the controller and model design and the usage of function can make code structure clear and easy to read.

(3) Flow control. Flow control is used for conditional choice and control of process procedure and data display. Flow control includes sentences such as for, while, until, break, continue, if/elif/else. Flow control can achieve the goal of page content choice and data flow branch.

Control including dynamic data in the web page view, such as text field, button and form, can be achieved through function. The print_table in the pseudo-code as shown in Fig 3 is a simple table function and its parameter is respectively data source, column number and margin. The function will print the col column table taking source as data source, the column number of table is calculated according to the dynamic of data volume in the source. An important feature of MVC view is nest. For example, a table can nest elements such as pull-down menu and radio button. The nest of function can be used to achieve view nest.

4.2 Controller design

The main mission of controller is to submit data process request of user to model and select proper data to return to user. It is mainly intended to schedule coordination between view and model. Controller call data processing interface and save result for reference of view. Concurrently, it is responsible for choice of view. To design better controller, better interface between view and model shall be provided to decrease direct access to model from view.

Fig. 4 shows a simple example of view-controller. The view quotes variable name and the controller call GetUserName function to assign returned value to name, thus view obtains value of name from controller and shows in the user interface. In this mode, only interface protocol between view and controller needs to be maintained to quot variable from view and separate usage and assignment of variable.

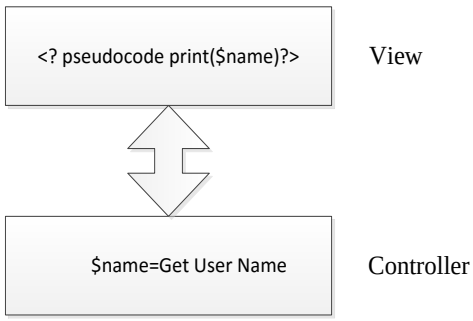

Figure 4. View and its controller

4.3 Model design

The model covers business data and business rule. Business data includes user profile, database and other data file.

Math Business rule is responsible for processing the user request passed from control module, mainly completes data calculation, access, transition and I/O operation and return result to control module. Model is the core of business system and provides interface for view and controller. The process procedure processed in the model is associated with business data, generally called by controller module and output unformatted original data. The data will be transferred to view to format after processed by control module. At last, they will display on the web page.

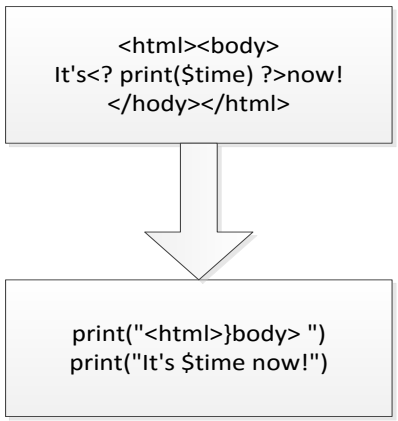

Figure 5. View preprocessing

4.4 Module integration

After completing 3 modules above, the most important is how to integrate them to form complete Web application. The preprocessing of view in the process as shown in Fig. 2 is the key step. The original view is code of HTML layer so that built in script is not yet performed and must be modified as CGI program. In this case, it can connect with controller and model. Business rule in this process namely read in each line of view file and output the content of non-script domain as Print Statement; after analyzing grammar of script domain, it is written to CGI procedure. The preprocessing of view involves text matching, replacement and simple syntax analysis. The whole process shall be complete automatically by template engine. In order to decrease difficulty of achievement, built in script can use same language with object CGI program. Fig. 5 is a simple example of preprocessing. The CGI procedure is generated after transition of original view file. After integration of CGI procedure, controller and model, the CGI web page which user access through browser shall be obtained.

It can be seen that the features of achievement is the preprocessing for template, namely, create CGI procedure through reading template file. After that, in the process of operation, template file is not reanalyzed so as to increase efficiency.

5. CONCLUSION

The application of MVC design mode on development of Web program of embedded Linux increase reusability of module and improve maintainability of system. Concurrently, it provides with better mode for development of Web application system and division of developer and with standard interface for integration among modules. In short, MVC is better software patter to establish Web application. The layer of Web application can make programmer carefully consider the additional complexity of application so as to increase expansibility and to make program more sound, more flexible and more individual.

REFERENCES

[1] L. Taihuan and F. Wei, "Design and implementation of blood management system based on B/S*," Journal of Capital Medical University*, vol. 31, no. 6, pp. 821-823, 2010.

[2] W. Daping and W. Bingwen, "Design and implementation of dam safety monitoring platform based on B/S," *Computer & Digital Engineering*, vol. 38, no. 12, pp. 145-147, 2010.

[3] ROBEY D., "User attitudes and management information system use," *Academy of Management Journal*, vol. 22, no. 3, pp. 527-538, 1979.

[4] X. Caifeng and Z. Yueheng, "Design and implementation of enterprise purchase sale storage management information system based on ASP.NET," *China Management Information*, vol. 14, no. 3, pp. 35-36, 2011.

[5] M. Dexiang and L. Rongge, "Web application program three layers models based on ASP.NET technology," *Microcomputer Applications*, vol. 18, no. 3, pp. 26-29, 2002.

[6] L. Qingsong, "Research and design of hospital management system," *Software Development and Design*, vol. 34, no. 1, pp. 55-59, 2010.

[7] Matsumoto M., "Assurance technology of system test based on operators' aspects*," High Assurance Systems Engineering Symposium*, no. 5, 2008.

[8] Matt J., *Web Programming with ASP and COM, Boston*, MA: Addison Wesley, vol. 32, no. 4, pp. 65-66, 2006.

[9] Gatrel M., Counsell S. And Hall T., "Design patterns and change proneness: a replication using proprietary

C# software," *Reverse Engineering*, vol. 23, no. 2, pp. 160-164. 2009.
[10] Eunmi Choi, Yoojin Lim, and Dugki Min, "Performance Comparison of Various Web Cluster Architectures," *Lecture Notes in Computer Science*, vol. 3398, pp. 617-624, May 2005.
[11] Fouzi Semchedine, Louiza bouallouche-Medjkoune and Djamil Aissani, "Task assignment policies in distributed server systems: A survey," *Journal of Network and Computer Applications*, vol. 34, no. 4, pp. 1123-1130, 2011.
[12] Md.Firoj Ali and Rafiqul Zaman Khan, "The study on load balancing strategies in distributed computing system," *International Journal of Computer Science & Engineering Survey (JICSES)*, vol. 3, no. 2, pp. 19-30, April 2012.
[13] Ahmad I., Ghafoor A. and Mehrotra K., "Performance prediction of distributed load balancing on multicomputer systems," *ACM*, pp. 830-839, 1991.
[14] Archer C. J., Sidelnik A. and Smith B. E., "Parallel computing system using coordinator and master nodes for load balancing and distributing work," *United State Patent*, vol. 12, no. 5, pp. 219-230, 2010.
[15] Jain P. and Gupta D., *An Algorithm for Dynamic Load Balancing in Distributed Systems with Multiple Supporting Nodes by Exploiting the Interrupt Service*, Academy Publisher, pp. 232-236, 2009.
[16] D. Mosedale, W. Foss and R. McCool, "Lessons learned administering Netscape's internet site," *IEEE Internet Computing*, vol. 1, no. 2, pp. 28-35, 1997.
[17] V. Cardellini, M. Colajanni and P.S. Yu., "Dynamic load balancing on web-server systems," *IEEE Internet Computing*, pp. 28-39, May 1999.
[18] Kwok Y., Ahmed, "Benchmarking and comparisons of the task graph scheduling algorithms," *Parallel Distrib. Comput*, vol. 59, no. 3, pp. 381–422, 1999.
[19] Thanalapati T. and Dandamudi S., "An efficient adaptive scheduling scheme for distributed memory multicomputers," *IEEE Trans. Parallel Distrib. Syst*., vol. 12, no. 7, pp. 758-768, 2001.
[20] Hiess H. and Schmitz M., "Decentralized dynamic load balancing: the particles approach," *Inform. Sci.*, vol. 48, no. 1-2, pp. 115-128, 1995.
[21] LeMair M., Reeves, "Strategies for load balancing on highly parallel computers," *IEEE Trans. Parallel Distrib. Syst*., vol. 4, no. 9, pp. 970-993, 1993.
[22] Watts J. and Taylor, "A practical approach to dynamic load balancing," *IEEE Trans. Parallel Distrib. Syst*., vol. 9, no. 3, pp. 235-248, 1998.
[23] Yagoubi B. and Slimani Y., "Task load balancing strategy for grid computing," *Comput. Sci*., vol. 3, no. 3, pp. 186-194, 2007.
[24] Donaldson V., Berman. F. and Paturi R., "Program speedup in a heterogeneous computing network," *Journal of Parallel and Distributed Computing*, vol. 21, no. 3, pp. 316-322, June 1994.
[25] Casavant T. L., "A taxonomy of scheduling in general purpose distributed computing systems," *IEEE Transactions on Software Engineering*, vol. 14, no. 2, pp. 141-153, 1994.
[26] Qian X. and Yang Q., "An analytical model for load balancing on symmetric multiprocessor systems," *Journal of Parallel and Distributed Computing*, vol. 20, pp. 198-211, 1994.
[27] Taylor V., Schwabe E. and Holmer B., "Balancing load versus decreasing communication: parameterizing the tradeoff," *Parallel Distrib. Comput*., vol. 61, no. 5, pp. 567-580, 2001.
[28] Eager D., Lazowska E. and Zaharjan J., "Adaptive load sharing in heterogeneous distributed systems," *IEEE Trans. Softw. Eng.*, vol. 12, no. 5, pp. 662-675, 1986.
[29] Tout W., "Distributed load balancing for parallel main memory hash join," *IEEE Trans. Parallel Distrib. Syst.*, vol. 6, no. 8, pp. 841-849, 1995.

NOMENCLATURE

MVC	Model-View-Controller
CGI	Common-Gateway-Interface
HTML	Hyper Text Mark-up Language
k	Thermal conductivity, W.m-1. K-1