International Journal of Enhanced Research in Science Technology & Engineering, ISSN: 2319-7463
Vol. 2 Issue 10, October-2013, pp: (60-64), Available online at: www.erpublications.com

SUPRA-MVC: A novel algorithmic approach to handle large scale bioinformatics applications

Muhammad Irfan Khan[1*], Muhammad Umer[2],
Muhammad Faraz Arshad Malik[3], Mohammad Ayyaz Azeem[4]

[1]Software Engineer, R&D department, eJuicy Solutions and Biosciences Department, COMSATS Institute of Information and Technology, Islamabad, 44000, Pakistan
[2]Project Manager, eJuicy Solutions Islamabad, 44000, Pakistan
[3]Senior Scientific Officer, Biosciences Department, COMSATS Institute of Information and Technology, Islamabad, 44000, Pakistan
[4]Center for Advanced Studies in Engineering (CASE), Islamabad, 44000, Pakistan

[1]khanirfanbioinformatics@gmail.com, [2]umer@ejuicysolutions.com, [3]famalik@comsats.edu.pk, [4]ayyaz.maju@gmail.com

Abstract: The challenge to manage massive applications arises from weak application design strategy. The objective role of this article is to propose an algorithmic approach to engineer high performance bioinformatics softwares. The technique proposed in this article aids in building efficient and effective large applications which become difficult to manage otherwise. This framework methodology is designed to implement application development using codeigniter 2.1.0 framework.

Keywords: supra-MVC, Bioinformatics applications, Coding tools and Techniques.

Introduction

The concept of MVC (model-view-controller) is quintessential to modular programming[1]. But the rising complexity of web environment based projects suggest that for "modern software" the traditional MVC scope falls short. Considering reduction in server processor load and efficient handling of foundational scalability issue of the application, the MVC pattern has its restrictions [2]. So, the practical implementation of MVC for large scale applications has some limitations. Whereas, supra-MVC architecture design is a generic structure hciihw can support application development with nested modules. Absence of nested modules support in codeigniter HMVC (Hierarchical model–view–controller) demands that we adopt Supra-MVC as an alternate coding pattern [3]. This algorithm provides skeletal infrastructure for implementation of bioinformatics related web-services after post genomic revolution. The aforementioned inadequacy points towards the requirement of modularized architecture design pattern that does not have HMVC restrictions.

Methodology

A stable version of codeigniter is required in order to follow the steps and understand the working principle of Supra-MVC. Public access is available on the internet as well[4] . The routing scheme of codeigniter framework is well documented and is publically available at [5]. To summarize, the business logics are maintained in models, whereas controllers are responsible for data trafficking and models are required for the interface design. A detailed documentary description of MVC is available which is required to understand Supra-MVC approach[6][7]. However, the workflow of Supra-MVC is explained in the following section

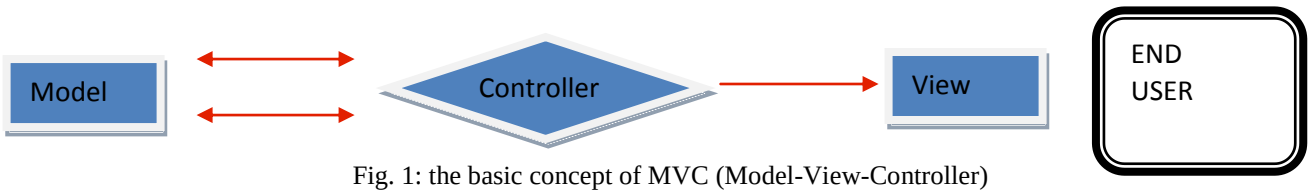

Fig. 1: the basic concept of MVC (Model-View-Controller)

Working Principle of Supra MVC

Application following Supra-MVC contain supra controller in the main controller subdirectory of the application folder. All the URL requests are pointed towards supra controller. Supra Controller in the application runs MY_Controller.php which is present in the core subdirectory of the application folder with any URL Request.

The controllers in the supplementary module controllers are to be designed in a separate module directory. Each controller has to extend Supra Controller instead of CI_Controller. This ensures that MY_Controller is called first and the URL request to any module is entertained via the MY_Controller.

MY_Controller file present in core subdirectory of the application folder of the project calls functions, Load_supra_view() and Load_supra_model(). The functions defined in "loader.php" file in the core subdirectory of the of the system folder of the project has two functions defined; supra_view() and supra_model() that are responsible for calling the views and models in the supplementary modules in the module subdirectory.The workflow of application is depicted in the following flowsheets

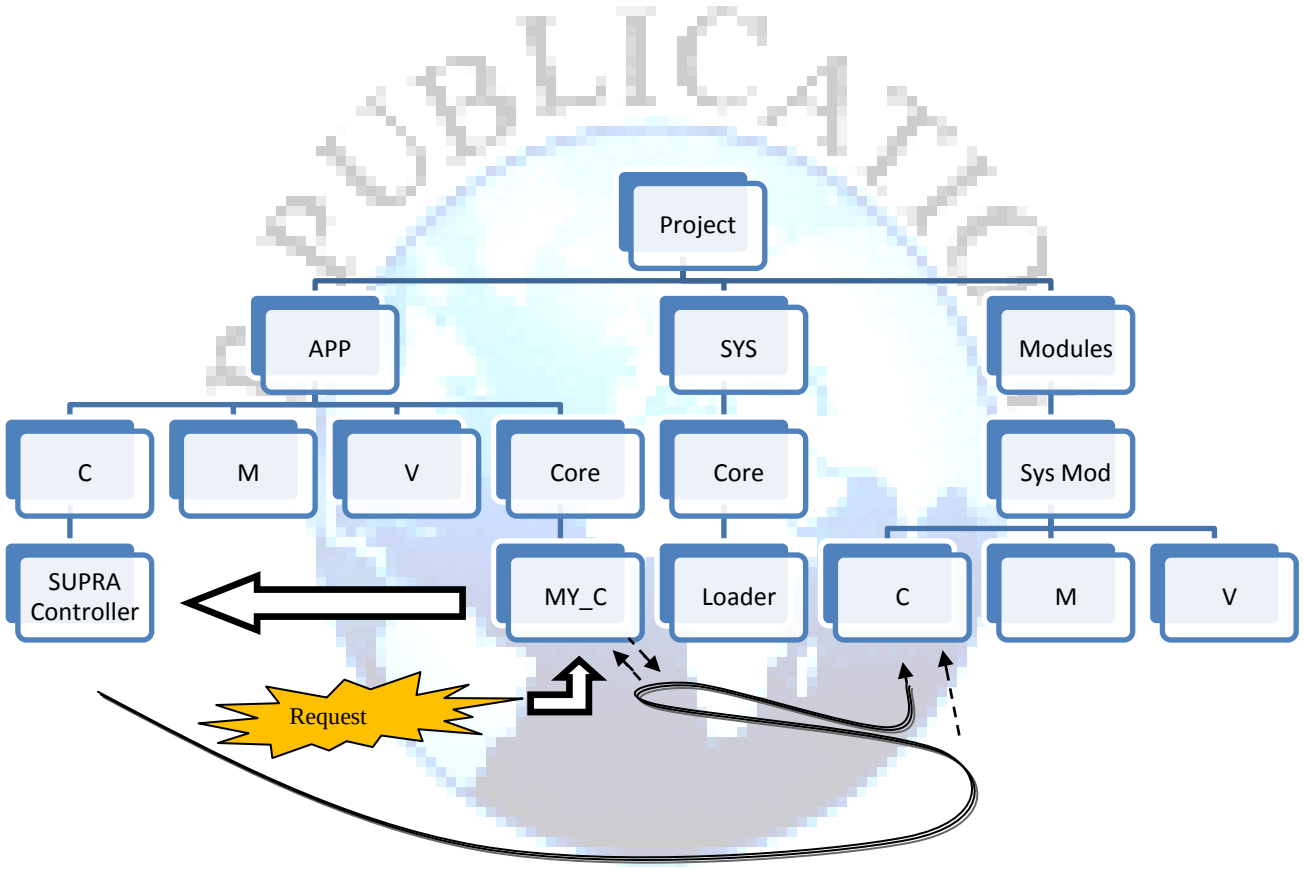

Fig 2: The codeigniter project has a file hierarchy system as depicted in the figure above.

Whenever a request in the URL is made the application runs MY_Controller.php in the core subdirectory of the application folder of the project which is called because the main default controller in the main controller subdirectory of the application folder of the project inherits this controller. The 2nd step of is to call the controller of the any module in the modules subdirectory of the project. As all the controllers in the modules subdirectory have to extend the Supra controller so the MY_Controller is called first and the URL request to any module is entertained via the MY_Controller controller in the core subdirectory of the application folder in the project.

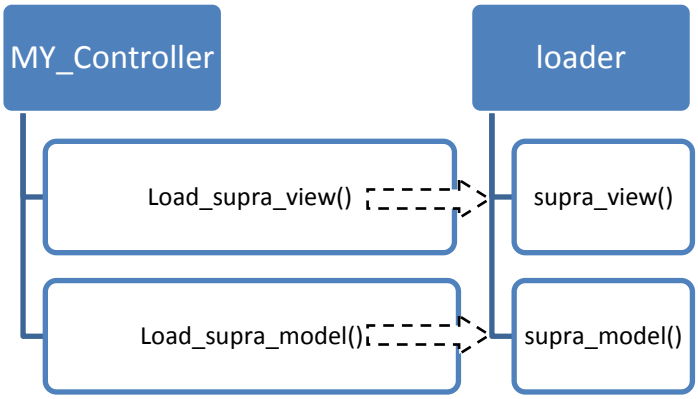

Fig 3: MY_Controller Controller that is contained in the core subdirectory of the application folder of the project calls both these functions, Load_supra_view() Load_supra_model() And the "loader.php" file in the core subdirectory of the of the system folder of the project has two functions defined; supra_view(), supra_model()

Configuration of an Application

To successfully design applications that implement the SUPRA-MVC pattern, it is critical that all of the application features follow the following rules.

- Download supra-MVC pack from http://dev.ejuicysolutions.com/supramvc
- Set the default controller to the supra controller and use $route[':any'] = "supra"; in routes.php so that every request in the URL is handled by supra.php. This is the only controller in the controller subdirectory of the application folder in the project.

This is of use as the modules are stored in the database with a module identifier and the Directory address where they are saved in is also stored in the database. Both these values are of key importance as module name is specified in the URL: request in the first URI segment. The first URI segment is then queried to database so as to retrieve the directory and the identifier. The controller name that is the class is specified in the second uri segment of the module path. The object of the path is made and the method that is mentioned in the third uri segment is now called via the object name.

- Add the following snippet of code to the end of config.php file in the config subdirectory of the application folder in the project. This snippet of code is useful to configure that the application has a custom controller in the core subdirectory of the application folder of the project.

```
function __autoload($class){
if(strpos($class, 'CI_') !== 0){
@include_once( APPPATH . 'core/'. $class . EXT );
}}
```

Fig 4: This function is provided in the supraconfig.php. The developers have to add this code in config.php file

Module Development Rules

Rules mentioned in this section are required to implement the supra-MVC architecture. (see supplementary material)

- Copy Supra.php file in the main controller directory of the application directory of the project.
- Include MY_Controller controller in core subdirectory of the application folder in the application folder
- Replace the Loader.php file in the core subdirectory of the system subdirectory of the application
- Run the modules.sql file and create modules table in the database
- Run the module_key_files.sql file and create table for key files of each module in the database

- save the modules in the database with a unique identifier and start to call the controllers as in supra.php controller file
- Create Module directory in the main directory of the project and start to write customized modules.
- The Developer will have to extend custom controller instead of CI_Controller
- Do not do $this->load->model for loading the models instead call $this->supra->model(MODULES_MODEL_PATH) method
- Do not do $this->load->view(MODULES_VIEW_PATH) for loading the views instead call $this->supra->view(MODULES_VIEW_PATH) method

Comparative study and discussion

The entire application runs on index.php then the controller mentioned in the URL is searched in the controller subdirectory in the application folder of the project[7] . This protocol is followed because of the routing scheme mentioned in the codeigniter.php and router.php in the "core" subdirectory of the system folder of the project. Controllers can be further separated in sections as subdirectories. But, Controllers cannot be taken out from the controller directory.

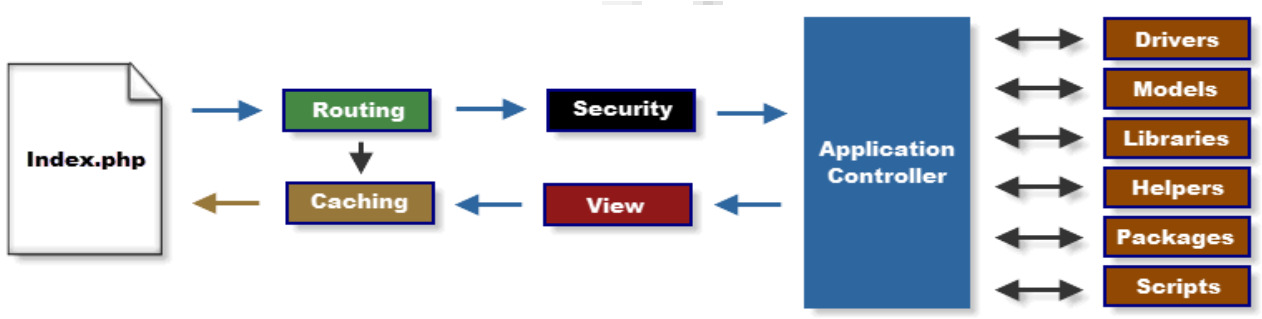

Fig 5: this fig explains the application functionality it is publically available at [7]

Whereas in the HMVC architectural design pattern that allows extensive modularization has one restriction that the folder name should be the same as the controller name so if we intend to encrypt the controller files name then there is a logic layer involved but the trade-off is that the application is not Search Engine Optimized and is not considered SEO friendly. So, this argument reiterates that there should be an alternative that does not have the restrictions of both the coding structures.

Features of Supra-MVC

Supra-MVC has the added advantages which are enlisted in the following figure as well. The Security is enhanced because the database operations are performed based on the module slug which is generated for each module added. The extensive support of modularity is augmented as the integration of additional modules is made easier. The Program organization is also incremented which leads to code reusability and hence, the performance is made robust and the applications developed become robust in efficiency.

Table 1: this table elaborates the robust features of supra-MVC

Modularity
• Modularity (Nested Modules Support) leads to easy integration of added features.
Organization
• Increased capacity for program organizational management has the improved application maintenance.
Reusability
• Code Reusability and code amendment is enhanced.
Performance
• Comparative SEO performance of application is enhanced.
Security
• Secutiry is the most robust feature of of the applicaiton following supra-MVC architecture

Conclusive Analysis

The pragmatic scope of this technique is generalized. The altered routing scheme can be used to develop commercial as well as R&D applications. So, the whole discussion points towards the requirement of an alternative pattern that does not have the restrictions of both the patterns the encryption for the controller names without compromising the search engine optimization. Hence, the scope of this research article elaborates the application aspect of this approach.

Future Perspective

An application using the Supra-MVC methodology shall be soon made public. The application interface design and development is currently under-development. The application will be a testbed workbench for scientists in general and bioinformaticians in particular.

Supplementary Material

The readme.txt file provided along with the core php files at http://dev.ejuicysolutions.com/supramvc guides the users to follow the rules in steps as well.

Acknowledgments

The support provided by Aahid Naeem, Melad-ul-Jabbar, Khurrum Liaqat, Adnan Ahmad Ansari , Majid Majeed Chaudry, Sibt ul hassan, Umar Butt, Amir waqas, Junaid naeem, Raja Farakat, Khasta Gula, Wasif, Mateen , Qasim, Hafiz hamid, Adnan Yusuf, zubair Qaiser, Mussadiq, Atif Hussain, Imtisal Akhtar, Aqil Mehmood, zeeshan javed, Rizwan Hayat and Gufran Bulbul (Department of Biosciences, CIIT, Islamabad, Pakistan) is highly acknowledged

References

[1]. http://www.jdl.co.uk/briefings/MVC.html
[2]. http://somethingstatic.com/hierarchical-model-view-controller-planning-future/
[3]. http://ellislab.com/forums/viewthread/204669/#954521
[4]. http://ellislab.com/codeigniter/user-guide/installation/downloads.html
[5]. http://ellislab.com/codeigniter/user-guide/
[6]. http://msdn.microsoft.com/en-us/library/ff649643.aspx
[7]. http://ellislab.com/codeigniter/user-guide/overview/appflow.html

Muhammad Irfan Khan is a bioinformatician with software engineering experience. His research interests involve solving system biology problems using computational techniques.

Muhammad Umer is a Microsoft certified, sql certified experienced IT professional.

Muhammad Faraz Arshad Malik is working as a Senior Scientific Officer, COMSATS Institute of Information Technology, Islamabad. He is interested in Artificial Intelligence, Machine Learning, Software Engineering.

Mohammad Ayyaz Azeem is a associated with case university, Islamabad. He also serves as support engineer for blackberry- warid in Islamabad, Pakistan. His research interests include algorithm development and software applications.