Evaluation and Refinement of MVC Web Application Architecture

Rajneesh Chaturvedi[1], Prof. Swati V. Chande [2], Dr. Amita Sharma[3]

1. Sr. Assistant Professor, The IIS (Deemed to be University) Jaipur (India).
2. Head, International school & informatics & Management, Jaipur (India).
3. Sr. Assistant Professor, The IIS (Deemed to be University) Jaipur (India).

Abstract

Web applications have become an essential part for the current social and IT scenario. There has been a continuous expansion with respect to dynamic and complex customer requirements while developing such web applications. These web applications are distributed and divided into layers as per the web application architecture and frameworks. The performance of these applications is greatly influenced by the underlying architecture and frameworks and is one of the key issues in the current competitive enterprise environment. The performance can be measured on the basis of response time and throughput parameters. These parameters can be further improved by refining the underlying web application architecture. A variety of architectures such as Model-View-Controller (MVC), Model-View-Presenter (MVP), Presentation–abstraction–control (PAC), Model View and View model (MVVM) etc. have been proposed for developing web applications, but MVC has been the most popular architecture amongst the web development community due to its suitability for large scale applications.

This paper focuses on the performance evaluation of MVC architecture and suggests refinement in the MVC architecture to achieve better performance results in terms of response time and throughput.

Keywords: MVC architecture, Web Application Performance, Response time, Throughput, Performance testing simulation tools, Latency.

Introduction:

Web applications have become an indispensable component covering many domains such as business, education, entertainment, art and many more. In such a scenario, the users of web applications have dynamic and complex requirements. In order to accommodate all such requirements of users, a web application's structure needs to be modular and should be divided into layers/components. This will in turn result in flexible and scalable web application architecture. [1]

The users of web applications require shorter response times; hence performance of the web applications is somewhere directly or indirectly related to user satisfaction amongst the users. The performance of any web application is measured in terms of response time and throughput parameters against the user's requests. It is dependent on various aspects such as application layer, network layer, web server, data layer and operating system policies etc; [2][3].The scope of this study covers the web application architecture as the primary source for improving the performance of web applications. The researchers have worked on improving MVC architecture to enhance the quality of web applications. The reduction of round trips (application latency) was considered as a refinement of MVC and proposed a Flexible Web Application Programming model (FWAP), which could be used on thick client and thin client deployment model.[2], introduced the concept of navigational objects, which meant navigation may be context dependent. This architecture also provided facility to define objects that implement the navigation view interface in generic manner and then refined these objects by adding application specific behavior.[3], A new improved architecture XMVC based on MVC and XML technology, a combination of XML string and Extensible Style sheet Language (XSL) file is sent back to client browser from bean.[4]

There are several aspects for improvement in MVC architecture. According to literature review, performance is one of the most important parameters in assessing the quality of any web application.

This study aims at improving the most popular MVC web application architecture in order to enhance the performance of the respective web application.

Section I describes the parameters of the web application performance; section II discusses classical MVC, section III discusses about the motivation for the proposed refinement, section IV suggests refined MVC, where as section V gives a detailed comparative analysis of performance testing of a web application developed using classical MVC and refined MVC model followed by section VI which concludes the paper.

I Parameters determining performance of web applications:

Web application performance evaluation requires evaluating performance in terms of performance indicators like Response time, throughput, no of concurrent users etc.

1. Response time: Response time is the time in which the user receives a response to the request. As the number of concurrent users increases, load on the web application also increases. As per the different experiments in a simulated environment the response time varies under one second against a certain number of concurrent users. But after reaching a specific number of concurrent users, system resources are occupied and sudden increase in response time is observed. [5]

As the number of requests increases, servers start loading those requests in the waiting queue while increasing the response time.

We can divide this response /waiting time in to two parts [5]:

a. **Network latency**: It is the time taken for transferring request/response between different servers.
b. **Application latency**: It is the time taken by the web application to generate response to the user request. This totally depends upon the application design, coding and application architecture used to develop the web application.

Response time= Network latency (N1+N2+N3+N4) + Application latency (A1+A2+A3)

The solution for decreasing the application latency (A1+A2+A3) is complex; a web application should be designed to minimize round trips to increase its performance. [5]

2. Throughput: Throughput reflects the number of requests that application can handle per unit time. As the number of concurrent users increases, throughput also increases proportionally, but as the system resources are fully utilised, throughput reaches at its peak and starts decreasing while decreasing the application performance.[5]

II MVC Architecture:

MVC (Model-View-Controller) framework is popularly used for web application design. The MVC architecture provides a way for decomposing an application into three parts: model, view and controller. The clean separation of these three components makes web applications more maintainable, scalable and efficient. This architecture is extensively used for separating the user interface from its application data and functionality. [6] Figure 1 shows the basic components of MVC architecture.

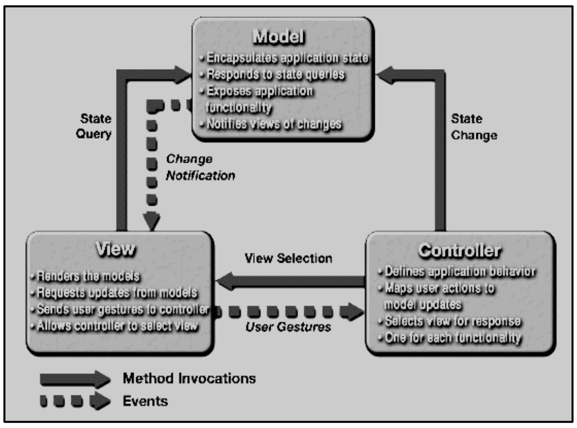
Model
• Encapsulates application state
• Responds to state queries
• Exposes application functionality
• Notifies views of changes
State Query
State Change
Change Notification
View
• Renders the models
• Requests updates from models
• Sends user gestures to controller
• Allows controller to select view
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

Figure 1: MVC Architecture framework [6]

III Motivation for the proposed refinement

The design patterns play an important role in the development of not only the web based applications but also mobile applications [7] The web applications have evolved over the years, but at the same time the inherent complexities need to be addressed with proper selection of development frameworks, architectural patterns etc.[8].This selection may further have a direct or an indirect effect on the flexibility, scalability, performance of the designed web applications. The evaluation of web applications is highly dependent on the performance indicators such as response time and throughput [9]. Hence in order to address these performance requirements, we may choose between different web architectures, web development frameworks, web servers, network layer/data layer level requirements etc.

Every application needs to be loaded on the server for its execution. As the size of the application grows the response time of application is affected adversely. Also, if the large size application is loaded on the server again the response time will increase. Considering this aspect, a refinement in terms of MVC architecture is proposed which considers validation as a separate entity from the model component [10].

The validations form an important part of any web application, wherein data from the user needs to be collected and verified in context with the business logic [11].There are two kinds of validations that can be applied while designing web applications:

a) Client side validation: These validations are generally implemented using scripting languages which first validate data at the client side.

b) Server side validation: Server side validations are implemented at server side to validate data received from client side using server side programming language. Server side validations are necessary part of business logic layer of web application architecture.

This eventually loads the application in segments or only the required part of the application is loaded.

IV Proposed refined model Server side Validation Model View Controller (SVMVC):

Server side model view controller (SVMVC) is a refined model of classical MVC (Model View Controller), which focuses on refinement of the “Model” layer of MVC framework to improve performance. Model is designated for the business logic/application layer of a web application, which contains all components of the server side to handle processing of all client side requests. As model layer has different components of server side processing, it can be further refined as per their different functionalities. Server Side Model View Controller (SVMVC) is refinement of the model layer which introduces server side validation as a new component. The validation component handles all server side validation processing and then redirects the requests to the other components of the model layer. Figure 2 discusses the basic architecture of SVMVC. This refinement will reduce the application latency, thereby improving the performance of web applications. The proposed refinement is verified with the results, achieved after performance testing of the developed web application.

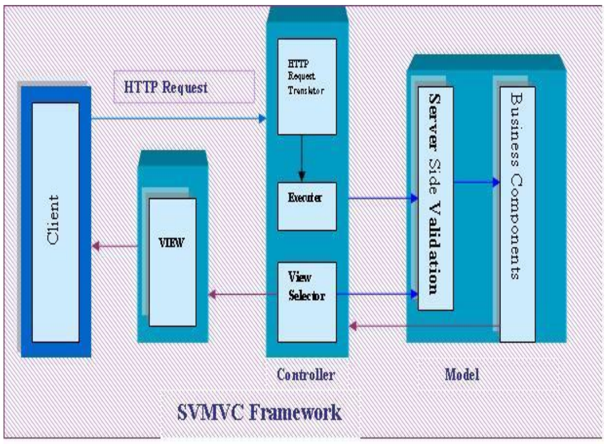

Figure 2: Proposed SVMVC Model

V Results and comparative analysis of MVC and SVMVC

Experimental Setup: This experimental study involves development of two different web applications based on MVC and SVMVC with the same functionality. These web applications with the underlying models have been further tested in a simulation environment using Jmeter.

Configuration:

Hardware: Intel Core to Duo process, 2 GB RAM

Software:

1. Programming language/IDE: Spring, Java
2. Simulation Tool: JMeter 2.3
3. Parameters of study: Response time, Throughput.

Result analysis of SVMVC performance:

The performance has been evaluated and recorded for two major parameters i.e. throughput and response time.

Comparative Response Time: The response time achieved in the simulation environment of both MVC and SVMVC based web applications have been displayed in Table 1.

Table 1: Comparative response time

	MVC	**SVMVC**
Concurrent Users	**Response time (ms)**	**Response time (ms)**
100	2199	1022
200	1186	1147
300	1291	1011
400	1123	821
500	1156	854
600	1247	1136
700	1142	790

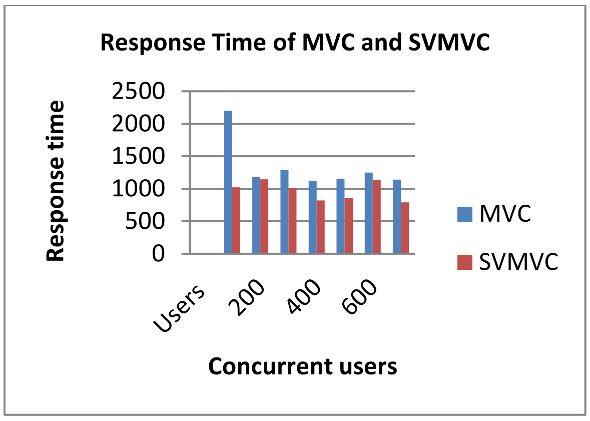

Figure 3: Comparative response time chart

Table 1 displays the response time chart generated by Jmeter for MVC and proposed SVMVC framework. While Figure 3 displays the comparative analysis of response time for MVC and proposed SVMVC framework.

The response time for MVC based web application with respect to hundred concurrent users is 2199 milliseconds whereas for proposed SVMVC web application it is 1022 milliseconds. The response time for MVC based web application with respect to two hundred concurrent users is 1186 milliseconds whereas for proposed SVMVC web application it is 1147 milliseconds The response time for MVC based web application with respect to three hundred concurrent users is 1291 milliseconds whereas for proposed SVMVC web application it is 1011 milliseconds. The response time for MVC based web application with respect to four hundred concurrent users is 1123 milliseconds whereas for proposed SVMVC web application it is 821 milliseconds. The response time for MVC based web application with respect to five hundred concurrent users is 1156 milliseconds whereas for proposed SVMVC web application it is 854 milliseconds. The response time for MVC based web application with respect to six hundred concurrent users is 1247 milliseconds whereas for proposed SVMVC web application it is 1136 milliseconds. The response time for MVC based web application with respect to seven hundred concurrent users is 1142 milliseconds whereas for SVMVC web application it is 790 milliseconds.

Inference: The observed values of response time for MVC and SVMVC based web applications clearly states that the response time of SVMVC based web application is significantly less as compared to the response time of MVC based web application. This means that the user will receive response of their request in lesser amount time if SVMVC based framework is used to develop web application as compare to the classical MVC framework. The fluctuation in the results can be attributed to certain other factors related to web server, operating system policies, data layer structure and network related factors, which is out of the scope of this study.

Comparative throughput analysis: The throughput achieved in the simulation environment of both MVC and SVMVC based web applications have been displayed in the Table 2.:

Table 2: Comparative throughput

	MVC	**SVMVC**
Concurrent Users	**Throughput (sec)**	**Throughput (sec)**
100	9	9.3
200	9.5	9.6
300	9.6	9.7
400	13	12.8
500	16	16.1
600	19	19.2
700	22.4	22.7

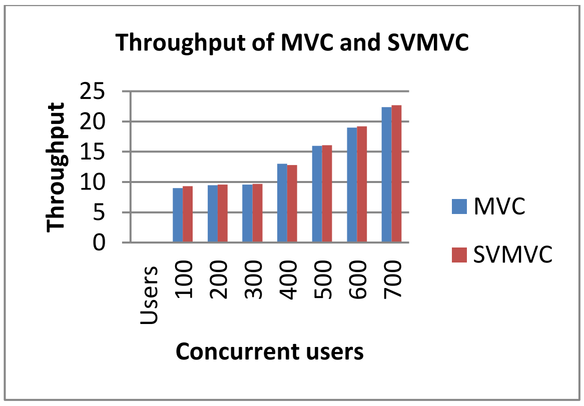

Figure 4: Comparative throughput chart

Table 2 displays the throughput chart generated by Jmeter for MVC and proposed SVMVC framework. While Figure 4 displays the comparative analysis of throughput for MVC and proposed SVMVC framework. The throughput for MVC based web application with respect to hundred concurrent users is 9 whereas 9.3 for proposed SVMVC web application. The throughput for MVC based web application with respect to two hundred concurrent users is 9.5 whereas 9.6 for proposed SVMVC web application. The throughput for MVC based web application with respect to three hundred concurrent users is 9.6 whereas 9.7 for proposed SVMVC web application. The throughput for MVC based web application with respect to four hundred concurrent users is 13 whereas 12.8 for proposed SVMVC web application. The throughput for MVC based web application with respect to five hundred concurrent users is 16 whereas 16.1 for proposed SVMVC web application. The throughput for MVC based web application with respect to six hundred concurrent users is 19 whereas 19.2 for proposed SVMVC web application. The throughput for MVC based web application with respect to hundred concurrent users is 22.4 whereas 22.7 for proposed SVMVC web application.

Inference: The observed values of throughput for MVC and SVMVC based web applications clearly states that the throughput of proposed SVMVC based web application is significantly higher as compared to the throughput of MVC based web application.

This means that the web application based on refined SVMVC model handled more requests as compared to MVC based web application.

VI Conclusion and future work:

The experiment results validate that there is a significant improvement in response time and throughput, with the proposed refinement in MVC framework. Hence it can be concluded that the proposed SVMVC framework can be used to improve the performance of the web applications. The refinement related to application layer parts and other parts of MVC can be taken up as the future work of research.

References

[1] Subraya, B. M., & Subrahmanya, S. V. (2000, October). Object driven performance testing of Web applications. In *Proceedings First Asia-Pacific Conference on Quality Software* (pp. 17-26). IEEE.

[2] Zhu, K., Fu, J., & Li, Y. (2010, June). Research the performance testing and performance improvement strategy in web application. In *2010 2nd international Conference on Education Technology and Computer* (Vol. 2, pp. V2-328). IEEE.

[3] Leff, A., & Rayfield, J. T. (2001, September). Web-application development using the model/view/controller design pattern. In *Proceedings fifth ieee international enterprise distributed object computing conference* (pp. 118-127). IEEE.

[4] Schwabe, D., & Rossi, G. (1995). The object-oriented hypermedia design model. *Communications of the ACM*, *38*(8), 45-46.

[5] Huang, S. Q., & Zhang, H. M. (2008, December). Research on Improved MVC Design pattern based on Struts and XSL. In *2008 International Symposium on Information Science and Engineering* (Vol. 1, pp. 451-455). IEEE.

[6] Thung, P. L., Ng, C. J., Thung, S. J., & Sulaiman, S. (2010, June). Improving a web application using design patterns: A case study. In *2010 International Symposium on Information Technology* (Vol. 1, pp. 1-6). IEEE.

[7] Aljamea, M., & Alkandari, M. (2018). MMVMi: A validation model for MVC and MVVM design patterns in iOS applications. *IAENG Int. J. Comput. Sci*, *45*(3), 377-389.

[8] Kumar, V., & Chopra, D.V. (2017). Design & Implementation of Jmeter Framework for Performance Comparison in Ruby, PHP, & Python Web Applications.

[9] Meier, J., Farre, C., Bansode, P., Barber, S., & Rea, D. (2007). *Performance testing guidance for web applications: patterns & practices*. Microsoft press.

[10] Sunardi ,A., Suharjito(2019),MVC Architecture: A Comparative Study Between Laravel Framework and Slim Framework in Freelancer Project Monitoring System Web Based,Procedia Computer Science,Vol 157, Pp 134-141, ISSN 1877-0509, doi.org/10.1016/j.procs.2019.08.150.

[11] S V N Madupu, Kiran Kumar. (2020). A Review on MVC Architecture Application. International Journal of Innovative Research in Computer and Communication Engineering. 8. 558-566. 10.15680/IJIRCCE.2020.0803065.