RESEARCH ARTICLE OPEN ACCESS

Web Application Framework: Review

Rimjhim Mishra [1], Dr.Pankaj Kumar [2]
Department of Computer Science and Engineering
SRMGPC, Lucknow
India

ABSTRACT
Architecture has its roots in civil engineering. Civil engineers lay down blueprints before starting construction of a building. The importance of architecture to web applications is the same as it is to a building (Grady Booch, Chief Technical Officer and Vice-President, Catapulse, 2001). A good architecture is the essential predictor for a cost effective, scalable, and continuously evolving website. In this paper we will discuss the existing architectures of web development.

Keywords *:- **Web framework, MVC***

I. INTRODUCTION

Architecture has its roots in civil engineering. Civil engineers lay down blueprints before starting construction of a building. The importance of architecture to web applications is the same as it is to a building (Grady Booch, Chief Technical Officer and Vice-President, Catapulse, 2001). A good architecture is the essential predictor for a cost effective, scalable, and continuously evolving website.

The canonical web architecture follows the basic server-client architecture. It has an application server and database interacting with each other in the same way as in any server software application. The web server and the web clients belong to the web space and they interact with both the application server and the file system. The simple canonical web architecture can be presented as follows.

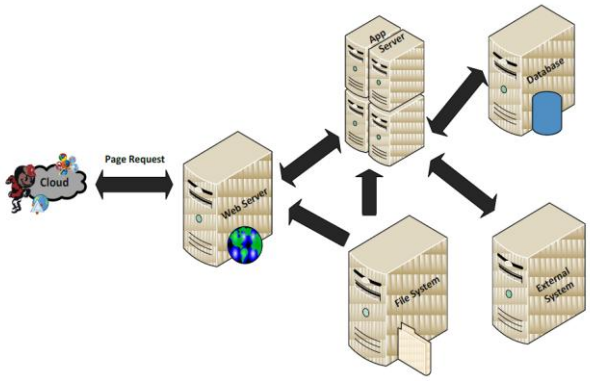

Figure 1: Simple Canonical web application Architecture

Though the architecture diagram resembles traditional client- server architecture, there are several architectural differences when the elements interact with each other.

For example, web application has to decide between maintaining a state and not maintaining a state and this affects all elements in the architecture unlike the traditional client-server architecture. There are several challenges in the implementation of web application architecture.

Different communication protocols are used in different layers of a web application. For example, communication between a web client and a web server would follow an Internet Protocol (IP) whilst a file server and an application server will communicate like a normal software application. Integration between these different communication protocols is a challenge in web application.

The decision between thin client and fat client is a very important architectural tradeoff for web applications. A web application developer should understand the advantages and disadvantages of each of these architectures. Thin clients are simple to use and easy to implement. Hence, sophisticated solutions cannot be expected out of it, whereas, thick clients offer locality of reference, sophistication of distribution and interactivity. However, the implementation and communication overhead with the thick client might not be suitable for many web applications. Therefore, understanding the demand of the web application and choosing between the thick and the thin client requires a

learning curve.

Considering all these challenges, detailed common web architecture could be derived. The Patterns and Practices Application Architecture Guide 2.0 discussed about one such common framework and this section refers to this more standard common framework. The representation of this framework is as follows (Meier, et al.).

II. ARCHITECHTURAL VIEWS

When designing a web application architecture, it is essential to consider different views as described in the 4+1 View Architecture model (Kruchten, 1995) (Shklar, Rosen, & Jones).

The design view of the WAF focuses on the actual design of different elements in a web application and the interactions between them. This includes the functional requirements of a system.

The process view is mainly used for scalable web applications, which requires manual thread implementation. If the underlying architecture framework does not have a mechanism to implement threads, then the web application architecture should be designed to accommodate concurrency and synchronization mechanism and therefore avoiding deadlocks or starvation.

The implementation view is the configuration setup in a web application. Apart from coding the functionality of a web application, proper setup has to be done for the communication protocols to work as expected. In addition, files and components integrated with the system should be assembled and marked for release. Effectively, the final product should have a standalone code with a complete functionality.

The deployment view maps the architectural components to the system hardware. This mostly involves the operational engineers' design document. The topology has to be understood and coordinated with the document before preparing the web applications for release. This ensures distribution, delivery, and successful installation of a web application on the server.

While designing a web application, use case view aids in understanding different perspectives of different users in the system. The system is used internally by testers, analysts and externally by various end users. Understanding their perspectives helps the developer to build a modular component oriented web application.

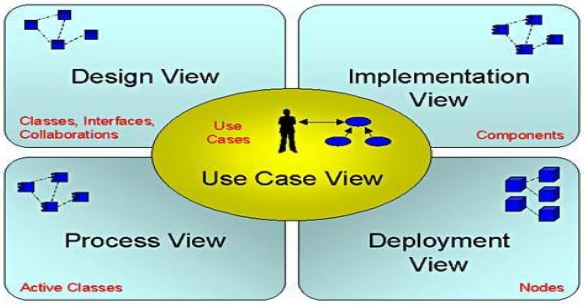

Figure 2: The 4+1 View Architecture Model

III. 3-TIER ARCHITECTURE

In a 1-tiered architecture, presentation, business and logical layers are inside a single hardware and in a 3-tiered architecture, these layers are in different physical platforms.

The initial tiered application was a 1-tier architecture, which had all the code in a single machine. The presentation, data and the logical layers were all tightly coupled and were indiscernible. It caused many problems with respect to scalability. Even with a multiprocessor system, the system was not scalable enough to cater the needs of increasing users, due to the connection limitations to a single server. Even if the server was powerful enough to accept many connections, code has to be changed in all the three layers to make the web application scalable. In 1-tier architecture, porting the application to another machine means rewriting the code from the scratch.

The 2-tier architecture was a separation of presentation layer and logic layer from the data layer. This made it easy for developers to forget the backend implementation while writing logic or html. Yet, problems that were described in the 1-client architecture remained between the logical and database layer.

The 3-tier architecture has three tiers with each tier belonging to one layer, presentation, logical and database.

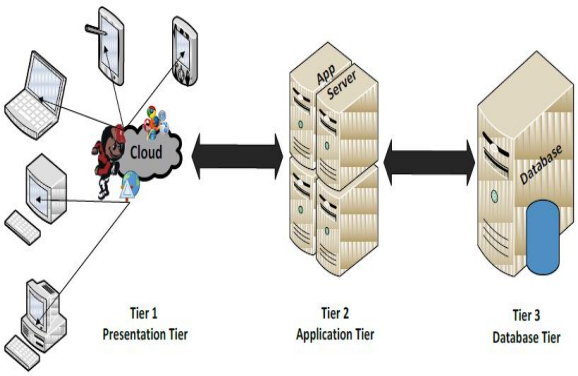

FIGURE 3: 3-TIER ARCHITECTURE

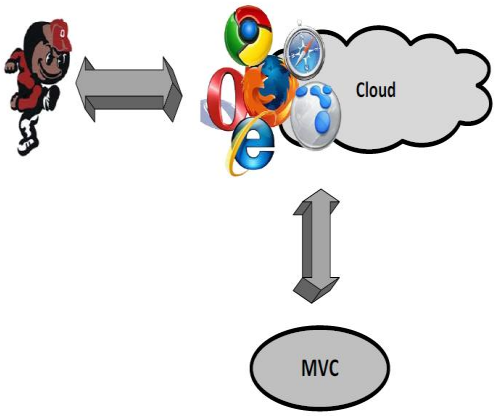

FIGURE 4: MVC Architecture

It has been one of the famous WAFs for quite a long time. As the names suggest, presentation layer deals with the HTML, static content, styles and all the front end applications, business later deals with the business logic and the data objects, which will be responsible to act as an interface to the database layer. The database layer has the actual data objects, which are updated, based on the business logic. The implementations of these three layers should be independent. When connected, these tiers should communicate smoothly without glitch

IV. MODEL VIEW CONTROLLER ARCHITECTURE

The MVC pattern was first implemented in Smalltalk and was called as the Smalltalk MVC Framework. It was primarily used for standardizing the UI of Smalltalk- 80. Three types of objects evolved out of this MVC Paradigm. The Model models the real world entity, which includes the entities, characteristics of the entities, state of the entities, application domain data and mapping between the application state and the entity state. The view is responsible for the visual content, from text to high level graphics. The controller reacts to the user inputs and manages both the view and the model. MVC is successful because of the division of responsibilities that lets an individual work on individual concerns. The figure below illustrates a typical .NET MVC.

The model holds the business logic of a web application and encapsulates all the system actions. The view represents the presentation layer and the controller focuses on control flow between view and model. It ensures appropriate actions are triggered when an event occurs.

V. WEB FRAMEWORK

Website Definition: A website is a collection of web pages served from a registered web domain via Hyper Text Transfer Protocol (HTTP). A website is hosted on a web server where both content and code resides and is accessed via a URL from anywhere on the internet. The web pages are either plain text or Hyper Text Markup Language (HTML) or Extensible Hyper Text Markup Language (XHTML).

Web application Definition: A web application is the dynamic version of the static websites. They have run able application scripts along with HTML and these scripts execute when the web pages are accessed via URLs.

Web application framework definition: A WAF is a software framework that gives flexibility of filling customizable code and generating web applications.

VI. ANALOGY FOR WAF

WAF is like a blueprint or frame for construction. Based on the requirements document, a WAF is chosen by the developer. Generally, WAF contains the template code and pre-implemented control flow. For example, WAF is like the frame for a hut, which

cannot be used to build a multi-storey building, but can be used to build a thatch roof hut, brick hut and so many other variations of hut.

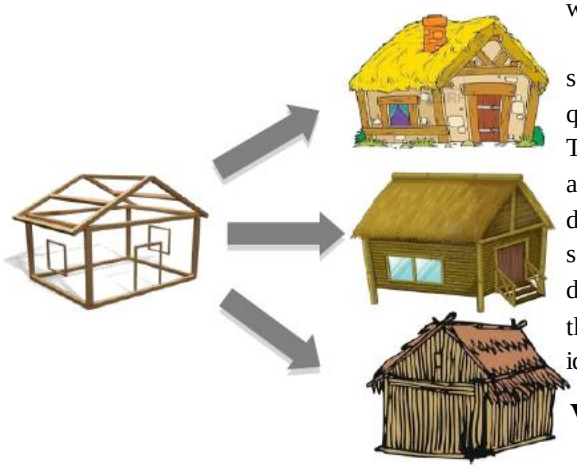

FIGURE 5: HUT FRAMEWORK ANALOGY

The constructor need not worry about how to place the wood one after another or how to define an outline. The framework takes constructor off the responsibility to redefine the shape of a hut, which is already been defined. Similar to the above analogy, a WAF is a framework chosen by the web developer to create web applications whose outline conforms to WAF standards.

VII. WAF CHARACTERISTICS

The primary purpose of WAF's existence is to promote reusability (Schwabe, Rossi, Esmeraldo, & Lyardet, 2001). Web design frameworks are conceptual approaches to avoid redundancy and to maximize code reuse. The importance of abstracting and reusing such design components and structures are explained in detail in this paper (Schwabe, Rossi, Esmeraldo, & Lyardet, 2001). Any WAF should be able to provide the four primary goals of web applications: scalability, maintainability, availability, and reliability. In addition, every WAF should have goals of its own. Different web applications will need different architecture and hence different WAF implementations. Some might have decoupled components of navigation, design and UI and some might not focus on the navigation completely.

Web developers choose different web design and development frameworks based on the web application that they are going to develop and this requires a learning curve. The learning curve adds up every time a different type of web application is being developed. Apart from spending time in actual design and development, there is an interval spent in discussing and deciding which web design or development framework will best suit the application.

Web applications are different from conventional software in the sense that they should have fewer bugs, quick development time, and continuous evolvement. Therefore, improving the development of web application should not only focus on the code development but also on the testing and debugging strategies. Very few frameworks enhance the testing and debugging factors of web development. A framework that could enable decoupling of all components will be ideal for component and unit testing.

VIII. COMPONENTS OF WAF

A web framework has two main components. One is the actual concept of the web framework where the application coding resides and the other is the navigation code. The conceptual model of the web application contains annotations, which are placed as placeholders for the developers to fill in the web application code, and the navigational model is the control hand offs that happen within a web application. For example, MVC framework defines navigational model within each controller where the controller is the manager handing off control to the next controller, essentially the next URL and hence allowing navigation within a site. These two components are the backbone of a WAF.

IX. APPRAOCHES TO WAF

Websites were initiated with static pages, images, and content in the first generation. Today, web applications have lot more complex structures and components entailed to them. For example, enterprise web applications show dynamically created pages, use relational databases to store enterprise data, implement database transactions, use the content management systems, and handoff controls between components. If a web developer has to go in detail into all of these components then it might be difficult to create an enterprise web application. A WAF separates SoC from the development cycle to keep web development process simple (Yaldiz) (J, 2002).

OOP has been one of the biggest advents in the software industry. Most programmers used OOP within each SoC layer. However, the development of relational

databases and Object Relational Mapping (ORM), has given rise to OO frameworks designed for web applications. ORM frameworks like Object Relational Bridge (ORB) and hibernate have become quite popular in software development. They give developers the flexibility to develop software applications that are less error prone and less deviating from the standard practices.z

X. CONCLUSION

Web applications have entered the research arena too where they are the primary data collection tools. Researchers use the "learn and adapt" strategy on web applications, which has required much better extensibility and modifiability. Here we discussed various architectures of web designing and web application framework.

REFERENCES

[1] Adam, P. (2007). Post object oriented paradigms in software development: a comparative analysis. Proceedings of the International Multi-conference on Computer Science and Information Technology, (pp. 1009-1020).

[2] Manal, M.Y. "A 'cloud-free' security model for cloud computing", International Journal of Services and Standards, Vol.5, No.4, pp. 354-375, 2009.

[3] An Approach to User-Behavior-Aware Web Applications. (2005). 5th International Conference on Web Engineering (ICWE) (pp. 417-428). Springer-Verlag. Balasubramanian, K., Schmidt, D. C., Molnár, Z., & Lédeczi, Á. (n.d.). System Integration using Model-Driven Engineering. Institute for Software Integrated Systems, Vanderbilt University, Nashville.

[4] Biggerstaff, T. J. (1989). Software Reusability: Concepts and models (Vol. 01). ACM Press.

[5] Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P., & Stal, M. (1996). Pattern-Oriented Software Architecture: A System of Patterns. John Wiley & Sons,

[6] Clarke, S., Harrison, W., Oscher, H., & Tarr, P. (1999). Subject Oriented Design: Towards Improved Alignment of Requirements, Design and Code. OOPSLA,

[7] Simon, D.R. "On the power of quantum computation", SIAM J. Comput., Vol.26, pp.1474-1483, 1997