Techniques for Scaling Components of Web Application

Ademola Adenubi, Olanrewaju Lewis, Bolanle Abimbola

***Abstract*— Every organisation is exploring the enormous benefits of the Internet to launch their respective web application as a means of integrating into the global village. For large organisation, the web application is expected to respond to increasing number of concurrent requests from clients. This extends a greater workload on the web server that hosts the web application and could stretch the machine beyond the designed capacity. Overloading the web server could break down the machine and compromise the business objective of the organisation. Thus, there is a need to scale the development of the web application in such a way that it can adapt to expansion and sustain high workload. This research integrates different techniques (tiered architecture, load balancing, caching and database replication) to create a model of a scalable web application that can be easily adapted to respond to increasing workload.**

***Index Terms*— scalability, web, application, techniques, component**

I. Introduction

INTERESTINGLY, the growth in the capability of computer terminal has challenged the rapid evolution of browser based application [1]. Thus, most organisations are exploring the opportunity of the internet to host their website as a platform to promote their products and services. Information about their product and services is arranged into web pages and delivered from the web server to the client browser over the HTTP (HyperText Transfer Protocol). However, recent developments in sustaining user's experience on the internet, coupled with the need to define means of seamlessly expanding the capacity of the website to process higher volume of client requests, have necessitated the development of web application that display dynamic information stored in a backend database. For organisation that adapts their website has a web application, content and application are provided in such a way that information are dynamically and automatically delivered to a site visitor in the most convenient and applicable manner [2]

For instance, a web site for selling products could have a web application that stores its products and their prices in the database. A potential buyer can select different products from the product page and move to a payment page where the prices of all the products selected are listed with the total amount due. The same payment page will have different information for different buyers based on the quantity and price of products they selected. Thus, web applications do not have pages structured like web sites; adding more data to the database increases its page effective count without necessarily adding more code or mark-up to it [3]

More so, web applications are pushing attention away from the client browser to the server. The advent of AJAX (Asynchronous JavaScript and XML) has changed the interaction model of web applications [3]. With Ajax, web pages can be modified by requesting information from the server and using the HTTP POST to submit the changes, returning another page either confirming the changes or displaying the modified data. The technique is that the server extracts the scripts and processes it against the database or any other applications it calls. The resulting data from the query is then returned to the client. In this case the workload is on the server. But for a single server, a high volume of requests will result in network congestion and application failure [4]

For a web application that is expected to service huge number of requests from numerous site visitors, managing the greater workload that is generated as a result of the rate of requests and concurrent requests by clients on the server requires that the key components (software and hardware) of the web application is designed to scale so that it can withstand the new requirements of greater workload. Scalability is referred to as "a component's ability to adapt readily to a greater or lesser intensity of use, volume, or demand while still meeting business objectives" [5]. Thus, organisation with initial simple web application can seamlessly integrate scalability techniques to grow its web application to the complexity and functionality of a large web application.

There are various techniques for scaling the component parts of a web application. A holistic approach to creating a scalable web application is to integrate the key scalability techniques together for scaling the workload on the web application. These techniques will be integrated together to create a model of a scalable web application.

II. Web Application

Unlike a web site that has static content on web pages, a web application contains the data but with a separate delivery mechanism [3]. In this case, content on the web

Manuscript received in September 25, 2013; revised January 25, 2014. This work is presented by the following authors:

Ademola Adenubi is a lecturer in the Computer Science Education Department, Tai Solarin University of Education, Ijagun, P.M.B 2118, Ijebu Ode, Ogun State, Nigeria (e-mail: aadenubi@ymail.com).

Olanrewaju Lewis is a lecturer in the Physics Department, Tai Solarin University of Education, Ijagun, P.M.B 2118, Ijebu Ode, Ogun State, Nigeria (e-mail: olanrewaju.lewis@googlemail.com).

Bolanle Abimbola is a lecturer in the Computer Science Education Department, Tai Solarin University of Education, Ijagun, P.M.B 2118, Ijebu Ode, Ogun State, Nigeria (e-mail: bolaikotun@yahoo.com).

ISBN: 978-988-19252-5-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

page is separated from the mark-up; it is dynamically generated and delivered on the page based on user interaction and request. Content on the web page is stored in the backend database and can be queried to deliver separate or different content on the same page. As with a static web site, data is transported over the HTTP from the web server and presented on the client browser. However, static content can as well be generated by web application with content management system (CMS) [2].

One of the key elements of a web application is its architecture, which defines the interaction between the major components of the web application and their respective functions [6]. The architecture makes web application to have series of application layers or tiers with each tier responsible for specific tasks of storing, processing and presenting data.

III. Components of Web Application

Within the context of the development and deployment of a functional internet based application that delivers on the business objective of an organisation, a web application is seen as an enterprise solution that is hosted on a machine but accessible by other machines on the network through the HTTP requests. Thus, Software and Hardware components represent the holistic view of efforts at creating a web application.

The software component of a web application represents the content, data and functions, which delivers the client's HTTP requests through the POST or GET functions. It is possible to layer the architecture of the software component by separating the data from the functions so as to make it easy to scale. Meanwhile, the Hardware components represent the hardware machine or computer cluster on which the software component is hosted. The clustering of the machines can as well be architected to scale.

IV. Measures of Scalability

The major objectives to scalability are to improve capacity of the components, enhance the efficiency of the components and to shift or distribute load on the components among several machines [5]. In achieving the key objectives, there are two measures to scale the major components of the web application: Horizontal and Vertical Scalability. The key difference is that horizontal scalability is about replacement while vertical scalability is about upgrading of components.

V. Scaling the Software Component

Scaling the software component of the web application is done by dividing the web application into logical layers: Presentation, Logic or Application and Database Layers. The presentation layer is concerned with page generation; the logic layer represents the core for processing all the business rules of the web application; the database layer stores the data used by the web application and does not receive direct connection from the presentation layer but from the application layer. Dividing the web application into logical layers, partitioning and replicating the database on cluster machines are techniques of scaling the software component of the web application.

A. Tiered Architecture

A well architected web application separates its application functionality into manageable block of tasks that utilise the hardware resources for easy management, reusability and scalability [5]. In this form, the web application has a series of application layers or tiers with each tier responsible for specific tasks. Possible architecture could be a single tier, two tier or three tier (also referred to as multi-tier) web application architectures. However, it is important to note that the tier represents logical divisions of the web application services and not necessarily a physical division of the software and hardware components [7].

In a single tier web application, also referred to as a flat or combined architecture, all the layers of the applications are deployed on a single machine. In which case, a single machine hosts the web application. A single server or multiple servers can be deployed, in which case the architecture becomes a single tier, single server or single tier, multiple server architecture respectively [7].

In two-tier web application architecture, there is a logical separation of the business logic from the presentation that is running on the client while the database is running separately on the server. However, the server can also store and execute the business logic with the aid of stored procedures and SQL [6].

The three-tier architecture clearly separates the three layers of the web application with each layer defined by the specific function it performs [7]. Thus, each tier can be modified without resulting in the overall change of the web application. There is also the N-tier architecture (N = 4, 5 …); a form of the three- tier architecture that further sub-divides the three layers [6].

Table 1: Comparing the three different tiered architectures

Parameters	Single-Tiered	Two-Tiered	Three-Tiered
Ease of Administration of Hardware	High	High	Low
Ease of Administration of Software	Low	Low	High
Cost of Deployment and Maintenance	Low	High	High
Connection to Remote Services	Low	High	High
Concurrent Database Connections	Low	Low	High
Scalability	Low	Low	High
Modularity of Application	Low	High	High

ISBN: 978-988-19252-5-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

For effective approach to scalability of large web application, the three-tiered architecture is appropriate, though requiring a higher degree of expertise and cost.

B. Replication and Partitioning of Database

For a high volume web application, a single database server is insufficient to handle greater workload. Using multiple machines to form a cluster of database servers with good redundancy to scale the workload on the database is appropriate. The database server can either be scaled horizontally where several other database servers are added or scaled vertically where the database server is replaced with a higher configuration server each time there is the need to scale. With this scalable architecture at the database layer, the data can either be replicated or partitioned to scale on the multiple machines providing several points from which data can be read.

With the replication approach, data is replicated on multiple machines within the cluster to expand the read capability of the database and thereby providing multiple points for reading data. Within the cluster are the master and the slave servers and replication takes place between them; the master records the write queries in the binary log and the slaves read the queries from the binary log. However, there are various modes of replication of the data between the master and the slaves and this includes Master-Slave, Tree and Master-Master Replications [3]. Another mode of replication is the dynamic multi-versioning approach which scales the database tier using distributed concurrency control mechanism [8]. By this approach, the database cluster is connected to a scheduler that distributes incoming requests. The scheduler is preconfigured to understand the transaction types, read or write, used by different applications and the tables in the database that they access and thus send the queries to the respective master node. The master distributes the read activities across the slaves and updates the replica on all slaves. In this way, concurrent transactions can operate on different replica.

The partitioning approach to scaling the database involves the division of the database into manageable partitions; each partition (containing a subset of tables from the database) may be spread over multiple machines. Each machine can then handle separate transaction to increase the read and write capacity of the database through a modified database dispatching layer that understands the right machine to execute a database transaction. Thus, a new partition can be added to increase the read and write capacities of the database making it possible to scale the database. Database partitioning is in two ways: vertical partitioning also known as clustering and horizontal partitioning also known as federation [3]. Clustering or horizontal partitioning of the database requires that the database is partitioned into multiple chunks or clusters of tables; a database dispatching layer is modified to understand the machine with the cluster containing the tables being required by a query. Meanwhile, a vertical partitioning or federation scales the database by splitting the data in the tables into chunks that reside on multiple machines called shards; a conceptual federation logic layer is added to accept the requests from the application layer and determines the appropriate shard to query data from. Vertical partitioning thus involves the creation of tables with fewer columns, a process common in Normalisation, and having the tables stored on multiple machines

Table 2: Comparing database partition and database replication

Parameters	**Replication**	**Partitioning**
Extent of Scalability - write	Low	High
Extent of Scalability - read	High	High
Complexity of Data Transaction	Low	High
Integrity of Data Transaction	High	Low
Time to Complete a Transaction	High	Low

Table 3: Comparing Vertical and Horizontal database partitioning

Parameters	Vertical Partitioning	Horizontal Partitioning
Extent of Scalability	High	Low
Ease of Administration	High	Low
Complexity of Data Transaction	High	Low
Integrity of Data Transaction	Low	High

The replication approach is subjected to redundancy and put greater workload on the master as the web application grows bigger. On the other hand, partitioning approach attempts to distribute loads to all machines but with more administrative efforts. For a MySQL database system, the choice of a replication approach is good for a web application with limited non linear read capacity, while a choice of partitioning approach is good for an architectural linear scaling [3].

VI. Scaling the Hardware Component

There are two options available for managing the hardware platform of a large web application. One alternative is to upgrade the web server from single processor to multiple processors and the other alternative is

ISBN: 978-988-19252-5-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

to add more servers. These two alternatives are described as vertical and horizontal scalability respectively [3]. Other techniques to scale the hardware include the use of load balancer to distribute workload among multiple machines and caching for storing frequently accessed data to reduce read requests

A. Horizontal and Vertical Scaling of Web Server

Scaling the web server from the hardware point of view is about expanding the capacity of the web server so that it can be efficient and reliable to support the growth of the web application and its greater workload as a result of high number of concurrent requests. To achieve this, two options are feasible which is either to scale vertically or horizontally. While a vertical scaling of the web server is about replacing the hardware with more powerful configuration, horizontal scaling is about adding more hardware of similar configuration, each time the web application is to grow in response to greater workload

Table 4: Comparing horizontal and vertical scaling of Web Server

Parameters	Vertical Scaling	Horizontal Scaling
Ease of Administration	High	Low
Cost of Implementation	High	Low
Power Consumption	Low	High
Physical Space Required For Deployment	Low	High
Support for Higher Workload	Low	High
Redundancy	Low	High
Limitation to Scalability	High	Low

The point at which horizontal scaling is consuming too much space, power and high cost as a result of redundancy is determined and vertical scaling is introduced. That is, when the cost of adding new hardware is becoming expensive, a replacement of the hardware with more powerful configuration is introduced to ensure that the web application is able to sustain greater workload at a relatively minimal cost, moderate physical space and easier administration.

B. Load Balancing

This is a technology of achieving the distribution of loads across multiple machines in a machine cluster. Various methods and strategies are used to achieve effective load balancing. However, load balancing technology is usually provided by a dedicated software or hardware tool.

Achieving load balancing with hardware tool is by placing a dedicated machine before the multiple servers in a cluster to create a layer before a tier in the web application architecture. The machine is configured to manage traffic and distribute incoming requests among the machines in the cluster. The load balancer machine is able to effectively detect new or failed server by periodically checking all the connected real servers to determine their status [3].

For a software based load balancer, the load balancing software acts as a frontend server distributing requests among the backend servers that are connected to its ports. With this technology, server software is installed on a regular machine to offer load balancing solutions that runs from a range of simple to super-complex operating systems as against the use of a dedicated load balancing hardware machine. Quite a couple of common choices of software solutions for load balancing are available and they include Perlbal, Pound and Linux Virtual Server.

An alternative method to a software or hardware load balancing methods is the round robin DNS (Domain Name Service). This method assigns a multiple IP (Internet Protocol) addresses to a single domain. When a client makes a DNS request by using the URL address of the web application, the DNS server resolves the domain name and responds with the list of server IP for the web application domain, and a connecting client is free to decide on which server to connect. The choice of this method is based on the extent of security and control over the DNS that is required for the implementation of the load balancing.

Table 5: Comparing Hardware and Software load balancers

Parameters	Hardware Load Balancer	Software Load Balancer
Ease of Administration	Low	High
Cost of Deployment	High	Low
Effectiveness and Reliability	High	Low

With the three-tier web application architecture, a load balancer can be added in front of a required tier to handle the number of concurrent requests going to the multiple servers. The choice is either in the use of a software load balancing or a dedicated hardware load balancing.

C. Caching

Caching is a way of reducing read requests and a performance related issue but has direct bearing on scalability because it expands the capacity of a web application component [3]. It suffices to establish that a scalable caching technology can be implemented at every level of the web application, especially at the layers close to the final output from read activity, perhaps, at the web server and database layers of a web application to improve their read capability.

Various caching technologies are used to scale a component of the web application for improved performance. The adaptive page caching, patented by IBM,

ISBN: 978-988-19252-5-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

supports the caching of a complete web application page [2]. In this technology, the web application dynamically generates cacheable web page after the page was initially accessed. A portlet caching technology is used to put the static content of the web application in a portlet cache after the initial page generation for a limited time (while the session has not expired) and ensure that only the dynamic content provided by interactive portlet is requested any other time the page is requested [2]. Like the adaptive page caching, this caching technology also reduces the markup or page generation time of portlets. The Memcached is by far the most popular and unrestricted (it is open source) caching technology. It is able to cache data and objects in memory; for that it is faster than most other caching technologies.

Table 6: Comparing Adaptive Page, Portlet and Memcache Cache

Parameters	Adaptive Page	Portlet	Memcache
Speed of Response to Request	High	High	High
Availability of Use due to Patent Issue	Low	Low	High
Effectiveness in the ability to drop Cache Data at any Time	High	Low	High
Possibility of Caching all Page Types	Low	Low	High

Putting a cache at the presentation layer will enhance the processing time of the web server during page generation. Putting it at the application layer and the database layer will also improve the time to process the business logic and the database transaction respectively while also reducing overhead or workload on the machines.

VII. Steps in Scaling Web Application

Though, there are challenges in design, implementation and management of high volume web applications with various methods of implementations [9]. But with a clear understanding of the architecture of the web application, it is possible to take an informed decision on the implementation strategy for applying the techniques to scale the web application. The following steps are suggested, by IBM Experts, as a model for building a scalable web site that is optimised for performance:

1) Understanding the application environment: this step requires an analysis of all the components in the existing infrastructure to determine their respective levels of significance and requirements for upgrade. The length of a submitted paper should be commensurate with the importance, or appropriate to the complexity, of the work. For example, an obvious extension of previously published work might not be appropriate for publication or might be adequately treated in just a few pages.
2) Categorizing the workload: this step requires an evaluation of the web application to determine all the transaction complexities it handles which include data, security and others
3) Determining the component that is most affected: this step requires that the most important characteristic of the web application should be mapped with each component to determine the best scaling characteristics that requires protection.
4) Selecting the scaling technique to apply for scaling the workload: this step requires that, based on the information gathered so far, the best scaling techniques is selected; a technique that does not compromise on any of the critical factors in achieving scalability.
5) Applying the technique: this step requires that the chosen technique(s) is/are applied in a test environment to evaluate the resulting performance and scalability impact of the changes in each component to the surrounding infrastructure – other components.
6) Re-evaluating the process: this step involves an evaluation of the entire scaling process over and over again to eliminate avoidable bottleneck in order to scale to manageable status, while leaving those bottlenecks that could not be avoided.

VIII. Model of a Scalable Web Application

For a classic J2EE based web application that is implemented on three–tier architecture, the web application is a combination of the logically separated presentation and business logic layers. The Servlet or JSP at the presentation layer accepts a browser HTTP request and returns HTML file if the request is for static web page but a request for dynamic content is sent to the business logic layer for processing through Java RMI (Java Remote Method Invocation). Another part of the Servlets or an EJB (Enterprise Java Bean) at the application or business logic layer performs the business logic and connects with a JDBC (Java Database Connection) to the MySQL database (it can be MS SQL or ORACLE) at the database layer to query and update the database. The database returns a table data to the Servlets at the presentation layer through the EJB at the business logic layer; the Servlets then render the processed request as HTML file to the browser. A diagrammatic description of the interactions in a three-tier web application architecture using Java is shown in figure 1.

ISBN: 978-988-19252-5-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

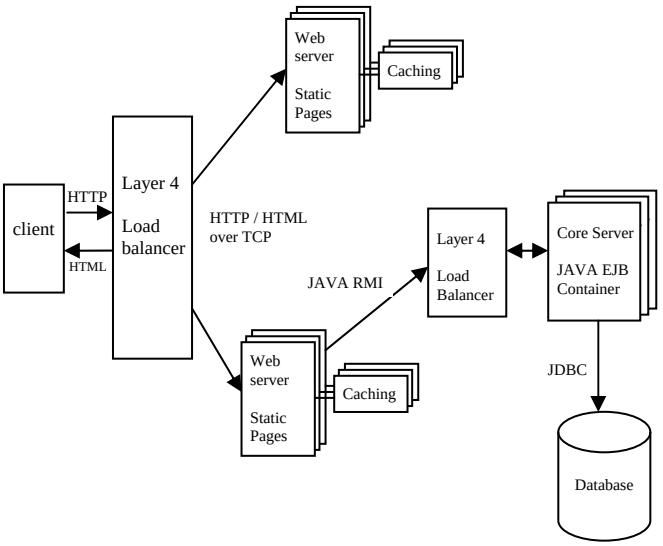

Figure 1: Scalable 3-Tiered Web Application Architecture

IX. Conclusion

Scaling a web application is indeed a means of managing workload and improving performance but it comes at a trade-off to the organisation. There is an increased cost of implementing scalable web application; more experts and machines are required and these come at a cost that might be beyond the reach of an average organisation. Related to cost is the administrative work that is involved in maintaining a big scalable web application.

There is also a relationship between scalability and performance of the web application. Though a scalable web application is indeed a relatively high performing web application but the reverse is not always the case. Thus, scalability and performance are two separate issues.

While efforts were made to review some of the major techniques and technologies for scaling the software and hardware components of web application, there are such other techniques like the Cascaded Style Sheet for scaling the User Interface Design and the Object Oriented Programming approach for developing the core application as modules to allow for code re-use and thus adapting the web application for scalability.

References

[1] J. V. Gurp, A. Karhinen and J. Bosch, “Mobile Service Oriented Architectures (MOSOA),” Nokia Research Centre, Finland.

[2] Liesche and Stefan, “Develop high performance Web sites with both static and dynamic content using WebSphere Portal V5.1,,” IBM WebSphere Developer Technical Journal, [Online]. Available: http://www.ibm.com/developerworks/websphere/techjournal/0506_liesc. [Accessed 24 4 2013].

[3] C. Henderson, Building Scalable Websites, O’Reilly Media Inc, 2006.

[4] T.-S. Chen and K.-L. Chen, “Balancing Workload based on Content Types for Scalable Web Server Clusters,” in *18th International Conference on Advanced Information Networking and Application, Department of Information Management*, Chang Jung University, 2004.

[5] W. Chiu, “Design for Scalability - an Update,” [Online]. Available: http://www.ibm.com/developerworks/websphere/library/techarticles/hipods/scalability.html . [Accessed 24 4 2013].

[6] A. Homer and e. al, “Components and Web Application Architecture,” 2008. [Online]. Available: http://technet.microsoft.com/en-us/library/bb727121.aspx. [Accessed 14 November 2012].

[7] Anon., “Cluster Architectures,” 2008. [Online]. Available: http://e-docs.bea.com/wls/docs81/cluster/planning.html. [Accessed 14 November 2012].

[8] M. Kaloian and A. Cristiana, “Scaling and Continuous Availability in Database Server Clusters through Multiversion Replication,” University of Toronto, [Online]. Available: http://www.eecg.toronto.edu/~amza/papers/manassiev-scaling.pdf . [Accessed 2 June 2013].

[9] W. Chiu, “Design for Scalability - an Update,” IBM, [Online]. Available: http://www.ibm.com/developerworks/websphere/library/techarticles/hipods/scalability.html . [Accessed 2 June 2013].

ISBN: 978-988-19252-5-1
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)