**Software Engineering Program**

**School of Science and Computer Engineering**

**University of Houston Clear Lake**

**Spring-2016**

**SWEN 5233, Section 01 - Software Architecture**

**Project Milestone 2**

**Part 1**

**Web Server**

**Mohan Ajay Kumar Yaramati - Apache**

**Sudha Mounika Jaripiti - Nginx**

**Dr. Sheena Judson Miller**

**05/02/2016 11:59AM**

# Introduction

A Web Server is any Machine that gets demands from a customer machine and can pivot handle the request and send back a reaction. This is usually the most important part as far as a Web Server to send back a Web pages when individuals what to go explore to a site page that is facilitated on that server now one may ask how it would send and get the data to and from the server. This is the place HTTP starts to assume a part into how this all goes about.

# Apache Web Server

## Overview

Apache is the main web server with more than 60% share in the market as per the Netcraft study. The Apache HTTP server software keeps running out of sight under suitable working framework, which supports multi-tasking, and gives administrations to different applications that associate with it. It is an open source application, BSD-like permit that takes into account both business and non-business uses of Apache. Skilled group of designers with an assortment of foundations and an open development process in view of specialized benefits. Apache keeps running on about all kinds of Windows, UNIX, mainframes systems. Apache web server consists of a modular approach. Each of the module has a particular usefulness that is structures a part of the consolidated list of capabilities of the Apache. A percentage of the standard modules mostly accompany the install of the Apache, while alternate modules must be downloaded independently that includes up the usefulness depending on client’s need.

Some features of apache web server

* It is a free and open source software which has the capability to incorporate various add-ons that are available.
* Reduction of size of web pages
* Apache web server installation can serve numerous websites simultaneously which can be referred to as concurrency feature.
* It is easy to specify KeepAliveTImeout in milliseconds.
* It is presently conceivable to define variables in the design, permitting a clear representation if the same quality is utilized at numerous spots as a part of the setup.
* It supports serving multiple protocols. For example, mod\_echo
* It is faster and stable on all the platforms like windows, BeOS, OS/2.

### Code Organization:

The source directory consists of bin directory where all the batch files exist. The configure files which are necessary for the server exist under conf directory. The library files are necessary for the execution of web pages which exist in source directory under lib folder. Information related to connections, requests made by the client are stored under logs directory. The src directory consists of java source files which leads to generate servlets and other java classes. Once the Apache is build successful an output directory will exist under source directory. The static content of the websites like JSP pages, HTML pages, CSS stylesheet files are located under web directory. A few important configuration files including web.xml file, some library files necessary for the application exists under WEB-INF directory. During the building process a couple of directories are created on a temporary basis. They are build directory which contain exact image of the file in web application and dist directory which will create an exact image of the web application which includes documentation, license file and read me file. General purpose expression parser allows to mention complex code using syntax like RewriteCond, Header, <If>, SetEnvIfExpr. AllowOverrideList directive is used to override configuration which are allowed in .htaccess files.

### Architectural style

Service oriented architecture style is one style that is observed in Apache web server. There will be a response to every request received from the client to the server. It not only responds to one particular client, it responds to all large number of user request simultaneously. If the request cannot be served by a single module, it gets transferred to other modules using apache core component. Once the request is served its time to frame the response that need to be sent to client browser. At this point Apache core components play a vital role in designing the response using functions like merging, formatting, splitting and some other security techniques. Modules communicate with each other to process the request accordingly. Registry is maintained to record the incoming requests from the clients. As many features like service, registry, interaction with other components, authentication, concurrency relate to Apache web server, Service Oriented architectural style is best suited.

## Module View

Module can be referred as a piece of code that extend functionality of Apache HTTP Server. Modules can be either progressively or statically involved with the center. For static consideration, the module's source code must be added to the server's source appropriation and to aggregate the entire server. Progressively included modules add usefulness to the server by being stacking as shared libraries amid start-up or restart of the server. For this situation the module mod\_so gives the usefulness to include modules progressively. In a present conveyance of either Apache 2.0 or Apache 1.3, everything except exceptionally fundamental server usefulness has been moved to modules. Modules associate with the Apache server by means of a typical interface. They enlist handlers for snares in the Apache center or different modules. The Apache center calls all enlisted snares when pertinent, that implies while setting off a snare. Modules again can connect with the server center by means of the Apache API. Utilizing that API every module can get to the server's information structures, for instance for sending information or apportioning memory. Every module contains a module-data, which contains data about the handlers gave by the module and which setup mandates the module can prepare. The module information is key for module enlistment by the center.

The configuration directives can be prepared by the apache modules. The module data is an extremely huge perspective as it is vital for module enlistment by the center.

A handler is a callback technique enrolled for a specific occasion. At the point when the occasion happens, the occasion dispatcher will call all handlers in a particular request. In Apache, the occasions amid solicitation preparing are set apart by snares. Apache Modules register Handlers for certain hooks.

Optional Functions are like hooks. An Apache 2.0 module can likewise enroll optional functions and filters. The distinction is that the center overlooks any arrival esteem from a discretionary capacity. It calls all discretionary capacities paying little mind to mistakes. So discretionary capacities ought to be utilized for assignments that are not essential to the request response process by any mean.

A Filter forms information by perusing from an information and keeping in touch with a yield channel. An arrangement of channels where one channel forms the yield of another is called channel chain. Apache utilizes an information channel chain to handle the HTTP request for and a yield channel chain to prepare the HTTP response. Modules can progressively enlist channels for both chains.

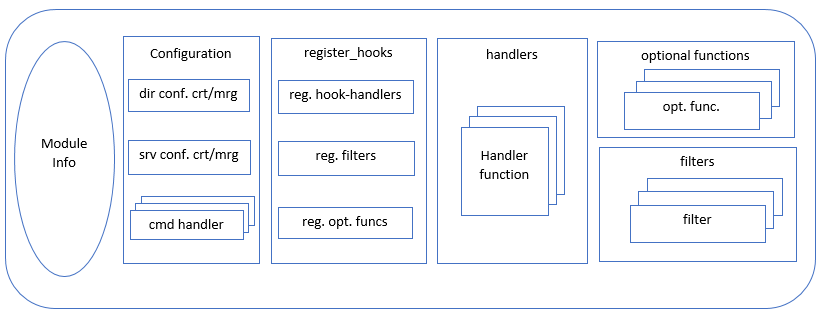


Figure 1 Module View [2]

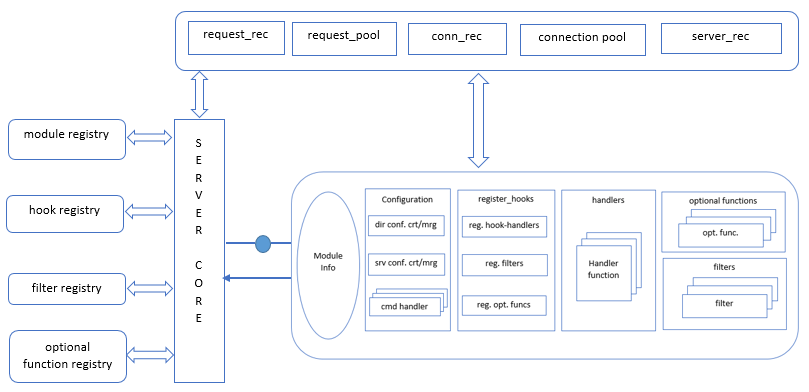


Figure 2 Interaction of core and module [2]

request\_rec: It is one of the most important data structure that exist in the web server. It will be passed through the module handler during the request/response phase. All the request and reference information connects it to registries.

server\_rec: Apache maintain this to keep the configuration information about the server. It comprises of data such as server name, port name, port number, time out and keep alive settings.

conn\_rec: Information related to the current existing connection is available in this structure. It also helps in storing user data.

### Realization of source code:

The module directory contains the subdirectories with all modules like ssl, mod\_proxy, mod\_auth\_form, mod\_buffer, mod\_data and many other modules required by the server to sustain. The server directory consists of all the core source files of the Apache web server along with Multi-Processing Modules. The os directory contains platform dependent files so that it can run on any platform it is compatible. The source library directory contains additional libraries like apr-util (Apache Portable Runtime utilities), apr(subdirectories of Apache portable Runtime), pcre(subdirectories for the files dealing with Perl-compatible regular expression). The build and test directories deal with building and testing Apache respectively. Attributes, parameters, cookies, parsers, packages exists in the http directory that makes the client request to navigate to apache web server and send back the response. Server core which is important part in the web server exists in the websocket directory.

## Component and connector view

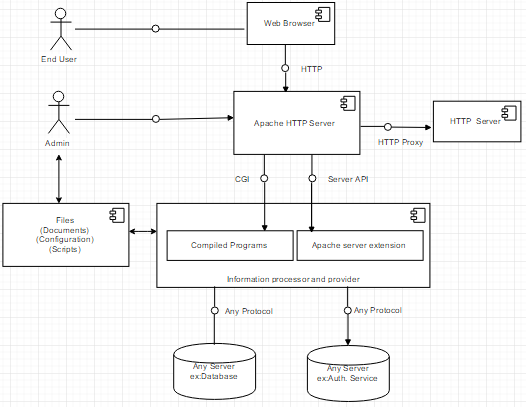


Figure 3 Component & Connector View [2]

### Relation between component connector view and module view

The segments of server lke database in Unified Modeling Language outlines associates to module which are request\_rec, conn\_rec, association pool, server\_rec, request\_pool that has all data about the server, the present and past associations. Modules are bits of code that can be utilized to give the usefulness of web server. The interface that exists in the apache server core and modules are given by the server to use by the modules.

Users raise requests using http and the apache receives it using HTTP protocol. Apache makes use of this services. The registry pool is shown in module view and service component in Unified Modelling Language diagram similar to registry pool in apache web server. Providing services to modules, and yield support to modules is the main goal.

### Sequence Diagrams

#### Sequence diagram to process HTTP requests

When the request is received the socketProcessor will process the socket wrapper to connection handler which handles all the request received. It again does the process with the socket wrapper. Processor prepares the request and serves the request and response. It then invokes standard engine, standard Context, standard wrapper and filter chain classes. Filters then call the servlet and serves the HTTP request.

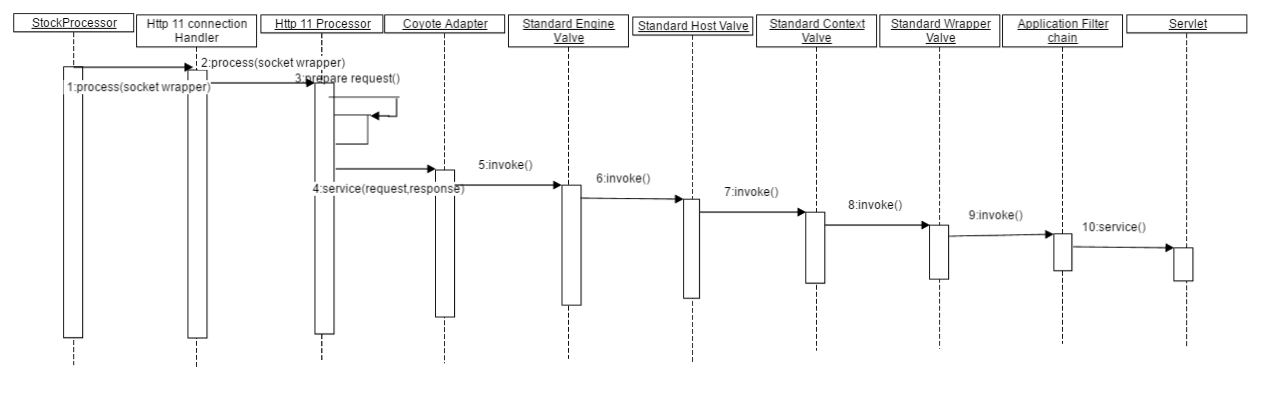


Figure 4 Sequence diagram to Process HTTP request [5]

#### 2.3.2.2 Sequence diagram illustrating flow of execution that results from browser request

The home page can be retrievedc from the browser when user gives a command. Ths browser establishes TCP connection with server and it sends a request to web server. Tomcat uses patterns of url in web.xml to find the process for these requests. In this case the doget method is called by tomcat in the home servlet. The home servlet is set for an attribute in request to the object passed by Tomcat. Once the home.jsp results an HTML page that returns to browser in a HTTP response message. As the browser gets the HTML, the TCP connection with server is closed.

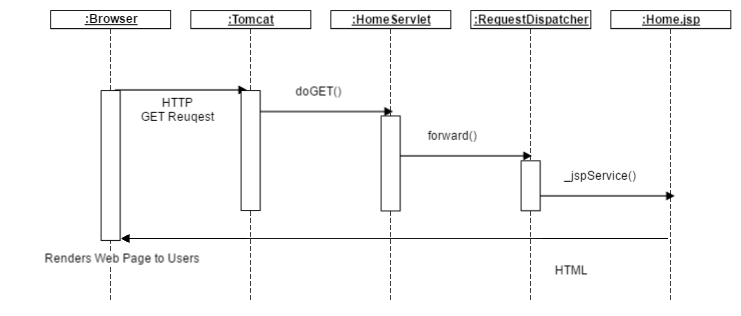


Figure 5 Sequence diagram Flow of Execution[4]

## Connectors:

A protocol is a set of special rules which allows the communication between each elements and accomplishes certain requirements of the elements. It includes JDBC and ODBC

JDBC: It is equivalent of ODBC just after the release of JAVA. It is a method of connection but drives off few databases than ODBC.

ODBC: It is the most common database. It was developed by IBM, Microsoft and many. The components in ODBC are driver and can be bought for common database platform.

CGI: Common Gateway Interface is a standard way for web server to interface which has programs installed in server and generates web page. These programs are usually done in script and can also be written in programming language.

HTTP: It is a server which receives request from web and makes the request to Internet. It also returns the result to the browser.

API: Application Programming Interface is an interface which is direct to the web server or a web browser. It is web development concept that is limited in web application, which does not include the web server or browser unless APIs are public. It is accessible by a remote web application.

# Allocation View:

User interacts with the web browser to run his application on Apache web server. Apache web server makes use of available resources like server, compiled files and some data components. It also makes use of server core which is important part in the web server. Apache server core interacts with registry pool and also with different modules like registry hook, configuration, handlers, filters. The server core consists of processing unit where the client request is processed with the help of data extracted from the database.

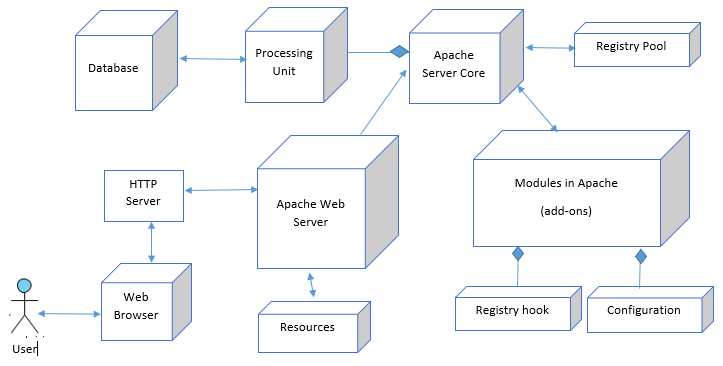


Figure 6 Allocation View

### Link allocation view to module view and C&C view

Requests are initiated by the users using web browser and can be received by apache core through http protocol. The apache uses resources, registry pools in allocation view of apache web server. They mainly focus on providing the services to different modules and also provide some support to other modules.

An allocation view type relates software to non-software unit. Example view relates a module with a hardware component which will run, like the hardware and communication device.    Work assignment view show how modules are assigned and are responsible for development. Implementation view may show the way modules are packed in library.

The way modules run on their hardware platform and the relation between are different compared to static structures. The C&C view shows clarity to the components and the connectors that transfer data and have control.

Modules like conn\_rec, server\_rec, request\_pool, request\_rec in the module view interact with the server core in Unified Modelling Language and the connector diagram. It also relates to database server in allocation view. They carry out the data about the server, existing and previous connections

## Conclusion:

The system architecture of the apache web server is caused by The Apache Software Foundation. It is concurrent and this feature of concurrency allows to meet large number of users with add on functionality which has a main role. It provides a very important role in providing performance, Usability, modularity features are quite some reasons which make it huge success.

# NGINX Web Server

## Overview:

Nginx is the most popular free open source web server which acts as a reverse proxy server for HTTP, HTTPS, SMTP, POP3 protocols that focus on concurrent connections with high performance. It is the third most popular web server after the Apache Web server and Microsoft’s IIS which operate as an IMAP/POP3 mail proxy server. In addition to that, it also serves as a load balancer and HTTP cache server which runs on Linux, Solaris, Mac OS X and Microsoft Windows. The nginx codebase is original which is written completely in C programming language. It is capable to handle more than 10,000 simultaneous connections with a low memory footprint, high performance and efficiency.

In the year 2002, Igor Sysoev initiated working on nginx web server. By the year 2004, public launch took place under the two-clause BSD license. The main benefit of deploying nginx is its high concurrency and high performance with even more benefits associated with it. Nginx can solve the problem of many connections running at the same time parallely. Many architectures and operating systems such as Linux, Solaris, Mac OS X, Microsoft Windows use nginx. Nginx was developed similar to apache initially where both the servers support static files like .jpg and .html. Nginx added more features compared to apache over the course of development.

### Features:

* Nginx provides features like Latency Processing, SSL (Secure Socket Layer), compression and caching, static content, requests throttling.
* Nginx is compatible with many operating systems such as Linux, Solaris, Mac OS X, FreeBSD and Microsoft windows.
* It handles high concurrency with high performance and efficiency.
* It uses multiplexing and event notifications and dedicates specific tasks to separate processes.
* Request processing rate limiting.
* With low memory, it can handle more than 10,000 simultaneous connections.
* It also includes TLS/SSL support.
* Web page access authentication.

### Code Structure:

The worker code of nginx incorporates the modules of core and functional. The nginx core is in charge of keeping up a run loop and implementing proper segments of module’s code on every phase of processing a request. Modules are composed of a large portion of presentation and application layer functionality. When proxy is enacted, modules read and write to the network and capacity, change the content, apply server side incorporate activities and transform the requests to the upstream servers. The modular architecture of nginx permits the developers to broaden the features of web server without changing the nginx core.

Dynamically loaded modules are not bolstered by nginx which means modules are compiled during the built stage including the core. Nginx utilizes event notifications, and various disk I/O execution enhancements in Solaris, Linux, BSD-based operating systems, like kqueue, events ports in order to take care of activities such as accepting, processing and managing the connections of network. The main objective is to give number indications to the operating system in concern to obtain asynchronous feedback for traffic, operations of disk, reading and writing to sockets etc. The use of various strategies for multiplexing and advanced I/O operations is intensely enhanced for each Unix-based operating system nginx keeps running on.

Nginx is an open source web server which follows an event-driven architecture. Apache web server has a problem in accommodating large number of clients synchronously which proposed to develop the basic architecture for nginx system. Hence, the base for introducing nginx architecture was taken from the Apache architecture. Nginx architecture uses ‘Uses’ category in Module view and ‘Pipe and filter’ category in Component and Connector view.

## Module View:

The Module view of nginx is shown in the below diagram:

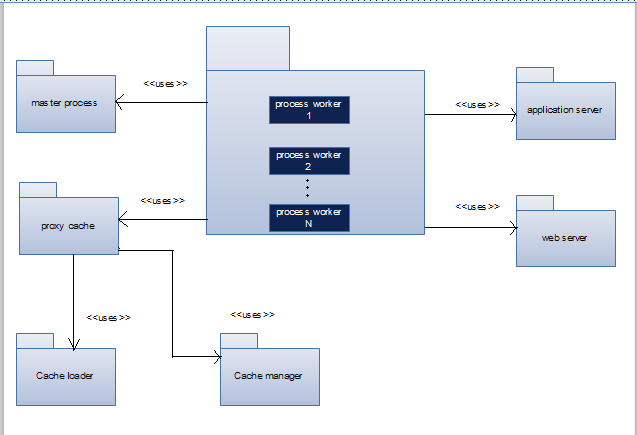


Figure 7: Module view of Nginx web server

The module view of nginx uses the ‘uses’ style. There are six modules which are described in the above view. They are: master process, proxy cache, cache loader, cache manager, application server and web server. Nginx has one master process and one or N number of worker processes. As we can enable caching here, we can also run cache manager and cache loader processes initially. The fundamental reason for running the master process is to read and assess configuration files and additionally maintain the worker processes. The processing of requests is done by the worker processes. To proficiently distribute the requests among all the workers in order to process it, the nginx depends on OS mechanisms. A configuration file called nginx.conf setup the number of worker processes. It can be consequently changed in accordance with the quantity of available CPU cores or it can be fixed to the given configuration.

A proxy cache known as web cache acts as a mediator between the client and the server which duplicates all the contents of the process and stores it. The client can request the proxy cache directly and then it returns the content to the client without contacting the main server. This enhances the performance because the clients can easily access the proxy servers more productively uses the application servers since they don't need to take every necessary step of creating pages from scratch every time. When the size of the cache reaches its limit, the cache manager process removes the least recently used files to bring the cache size back under the utmost. After the process starts, the cache loader runs only once which loads metadata (cached data) into the shared memory.

## Component and connector view:

The Component and connector view of nginx is shown in the below diagram:

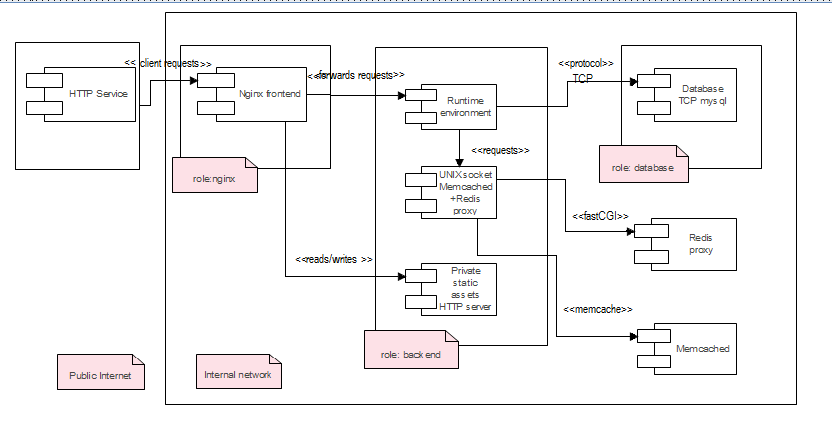


Figure 8: Component and connector view of Nginx web server

The component and connector view of nginx uses the pipe and filter style which is described in the above figure. The following are the components in it: HTTP service, Nginx frontend, runtime environment, UNIX socket memcached and redis proxy, private static assets HTTP server and database. It shows all the run time components and their behaviour in this type of component and connector view. The roles such as nginx server, backend and database constitute the internal network. The memcached is an extremely basic and simple API and you can have memcached components more than one. The database server is Mysql 5.1 which is connected to the run time environment by a pipe. We can use the Redis proxy component for simple caching which is described in the module view as proxy cache or web cache module. The HTTP service is an external service which provides the capacity to make the HTTP requests easily and it serves as an initial flow of data. Nginx frontend serves as a reverse proxy and provides HTTPS to the client by getting the content through HTTP from number of backends.

### Sequence diagrams:

#### Sequence diagram1:

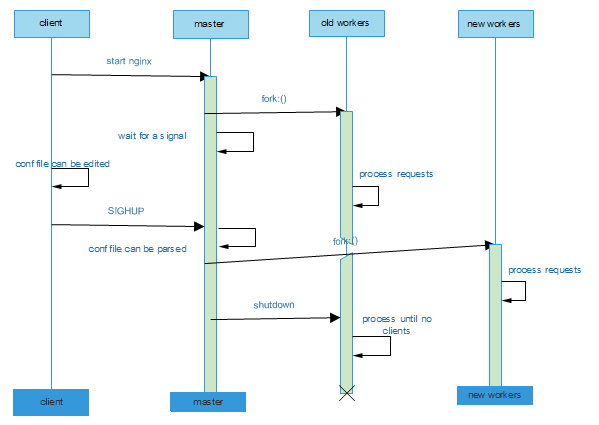


Figure 9: Sequence diagram of master and worker processes of nginx web server

#### Sequence diagram 2:

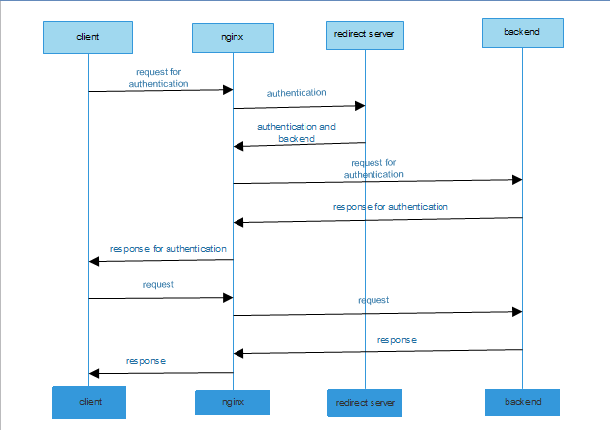


Figure 10: Sequence diagram of nginx requests and responses

Apache inspires nginx by implementing memory allocations inside its worker process. The nginx memory management description will be as follows: As there are multiple connections, for each connection nginx dynamically allocated the needed memory buffers and they are linked for saving and manipulating the body and header when these is a request and a response. And after the connections are released, the memory buffers which are allocated, stored and linked are freed. It is critical to note that nginx tries to abstain replicating the data which is in the memory as much as possible by sending the data using pointer values. There are two modules http and mail which are said to be core and the real functional modules and they play a prominent role in the working of nginx. These two modules give an extra level of abstraction between the core and the lower-level components. The sequence of events are handled by a connection with an application layer protocol such as HTTP, SMTP or IAMP in these type of modules.

## Allocation view:

The Allocation view of nginx is shown in the below diagram:

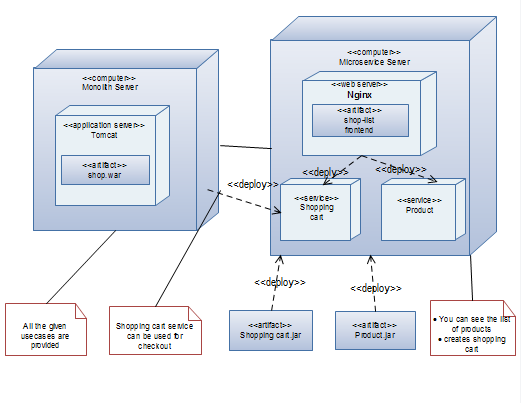


Figure 11: Allocation view of nginx web server

The allocation view of nginx uses the deployment style which is described in the above figure. The user interactions can be stated by considering the example of how nginx works with online shopping. Nginx web server logs all the requests from the clients which request it. This allocation view is given as an example to the component and connector view where the nginx frontend which is shown in both the figures takes the requests from the client and provides the HTTPS to the client by getting the content via the hypertext transfer protocol message. The application server which is shown in the allocation view and module view furnish its clients with access to the business logic which creates dynamic content which means the code that changes data to provide the specific functionality offered by an administration or an application. Communication between the application server and its customers may take the form of HTTP messages and the connection from HTTP service to the nginx frontend is shown in the component and connector view.

## Conclusion:

Nginx was intended to be a particular tool to accomplish performance, density and use economical resources of the server while empowering dynamic development of a website, so it has taken after an alternate model. It was really inspired by the continuous improvement of advanced event-based techniques in various operating systems. The foundation of nginx code resulted in a modular, event-based, single threaded, asynchronous architecture with high performance and high concurrency. Nginx utilizes multiplexes and event notifications intensely and allots specific and fixed tasks to some processes. A pre-determined number of single threaded processes are known as ‘workers’ process connections in a highly effective run loop. Nginx is able to handle thousands of simultaneous connections within each worker.

# References:

[1] The Apache Software Foundation Apache Tomcat[Online]. Available https://tomcat.apache.org/tomcat-7.0-doc/connectors.html

[2] Home of Fundamental Modelling Concepts Apache Modeling Project[Online]. Available http://www.fmc-modeling.org/projects/apache

[3] The Apache Software Foundation Overview of New Features in Apache HTTP Server[Online]. Available https://httpd.apache.org/docs/2.4/new\_features\_2\_4.html

[4] California State University San Bernardino Java Server Page[Online]. Available http://cse.csusb.edu/turner/java\_web\_programming/jsp/

[5] Programering Tomcat Introduction[Online]. Available http://www.programering.com/a/MjM4cjMwATY.html

[6] Home of Fundamental Modelling Concepts Apache Source Distribution[Online]. Available http://www.fmc-modeling.org/category/projects/apache/amp/4\_2Structure\_Source.html

[7] The Apache Software Foundation Connectors[Online]. Available https://tomcat.apache.org/tomcat-7.0-doc/connectors.html

[8] Nginx. Welcome to Nginx Wiki’s documentation [Online]. Available: <https://www.nginx.com/resources/wiki/>

[9] Andrew Alexeev. The Architecture of Open Source Applications [Online]. Available: <http://www.aosabook.org/en/nginx.html>

[10] Quora. How can nginx handle concurrent requests with single worker process [Online]. Available: <https://www.quora.com/How-can-nginx-handle-concurrent-requests-with-a-single-worker-process>

[11] Felix Bachmann and Mark klein. Understanding quality attributes [Online]. Available: <http://www.win.tue.nl/~wstomv/edu/2ii45/year0910/Software_Architecture_in_Practice_2nd_Edition_Chapter4.pdf>

[12] Nginx. Configuring HTTPS servers [Online]. Available: <http://nginx.org/en/docs/http/configuring_https_servers.html>