Mamun Hossain - 100553073

Can you compile a list of software requirements of a web-server as described in the article?

ID	Software System	Category	Description	Risk Control?
1	Internet Protocol	Server	Allows for communication between computers that might look like reading and writing files.	No
2	Socket	Server	Point to point communication channels like a phone.	No
3	HTTP	Server	Allows for data to be transferred over IP address	No
4	Data Fetch	End User	Using the HTTP *GET* to fetch information regarding the page from the server that the end user initiates.	No
5	Format Data	End User	Format the data to code the data so it can be read and recognized by different applications and programs.	Yes
6	Displaying values	System	Allows to modify web pages to display values provided by HTTP requests.	No
7	Listing Directories	System	Uses do_GET to teach web server to display a directory’s content when path is in the URL rather than in the file	No
8	CGI Protocol	System	Provides a standard way for a web server to run an external program to satisfy a request.	Yes

At what layers of the OSI models does this system operate?

This system would operate at three different layers: Layer 2 (Data Link), Layer 3 (Network), and Layer 7 (Application). The reason why I would say falls under these layers is because of the following:

Layer 2 (Data Link): The server needs to have a hardware interface to be able to have the values to be displayed to the end user. The data link would be the connection to the end user to the web server that is made in the process.

Layer 3 (Network): Because we are using an IP address that is being embedded with a socket, which directly communicates with the HTTP that transfers the data over the IP address, we can be confident that this layer is crucial as the Internet Protocol is an integral protocol at this layer.

Layer 7 (Application): Due to our usage of the server and how we are using it, especially with using HTTP to transfer the data information using the do_GET methods, it is safe to say because of its dominance in layer 7, and the wya HTTP manages website traffic on the internet, the web server operates at this layer.

What is the role of HTTP, DNS, HTML in the described system?

HTTP: In this system, HTTP is described as a means of the client sending a request that specifies and asks for what it wants over the socket connection and in response sends the server data. The important thing about the HTTP in this described system is that the request it makes is just text, and the information the program that desires it needs to parse the information, but that being said HTTP responses are formatted like HTTP requests.

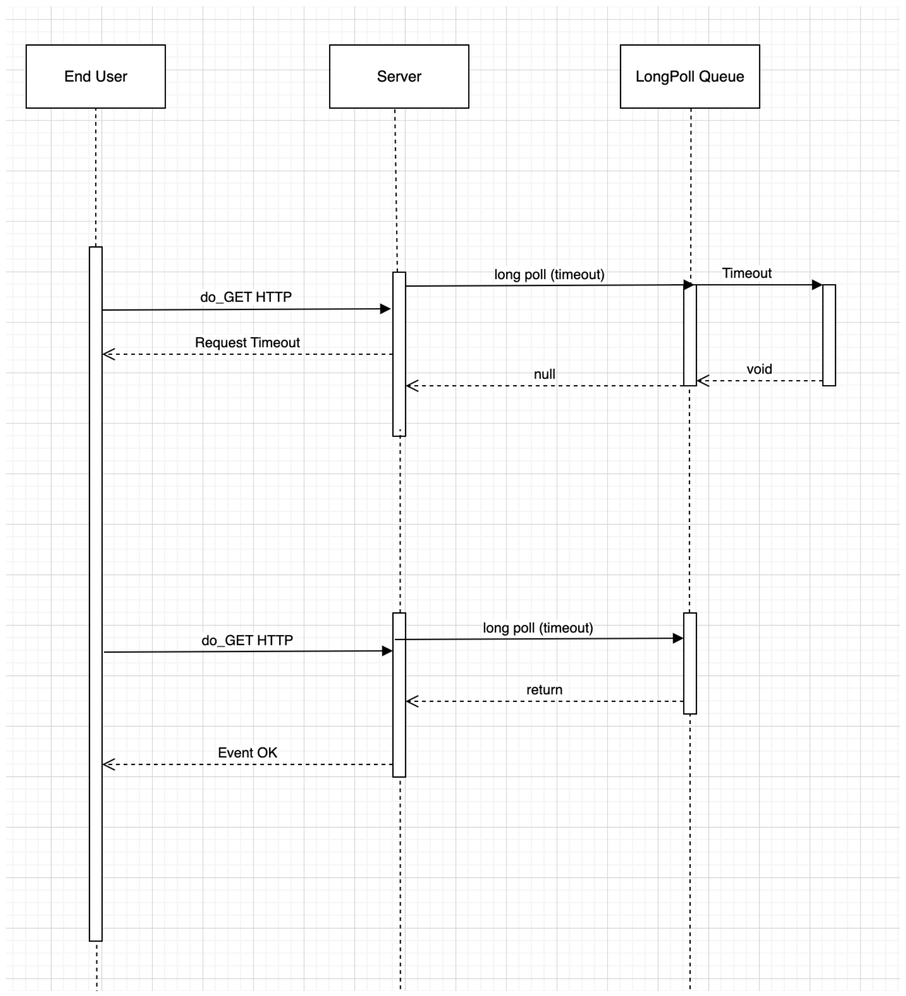
DNS: In the described system, the main goal of the DNS is to provide the IP address with its four 8-bit addresses to make it easier for us humans to use.

HTML: In this system, the role of the HTML files are to be formatted and send that information back by creating a header to tell the client to read the data in HTML. Doing this will be able to relay the information in a readable way, and will allow the display to work in cohesion for the end user.

Take CNN.Com for example, their web server updates the information displayed on each client. How can a web server do this?

The way this works is because it uses an event driven framework that supports WebSockets and HTTP. This event driven framework contains different client and server side components and creates an asynchronous web application which in turn supports the WebSockets, the Server Sent Events, and the Long-Polling that allows for a web server to do this.

Draw a UML sequence diagram



What functions/components need to be added to the described system?

The main functions that need to be added to the described system is implementing a blocking queue and an event manager. With these two functions, we can add in our components, such as a long polling component that can time out when necessary but also return the poll event when needing to update the server. The server sent events will then work in cohesion with the event manager to use a PUT event into the HTTP client and back into the server.

How are transparencies provisioned within the described image sharing system?

The transparencies are being provisioned in the Internet Protocol layer as it is being put in the socket and being used with the HTTP file structure to then relay that information and format it as a client. There is also transparency located in the servers point of view, as when the connection becomes transparent, the server will be able to identify the IP which in turn will allow for the seamless interactions with the CGI protocol and to display the web server's response after importing the requests.

What is synchronous messaging or request reply messaging? What is the role of a web server in this type of messaging?

Synchronous messaging: A type of messaging that needs to have a request and a response. The benefits of this are that it is done in real-time, and uses an obvious model, but that being said not all problems can be solved synchronously.

Request Reply Messaging: When the client acknowledges the receiving of reply messages and when the server gets back with the acknowledgement the client will only delete the information from its cache after.

The role of a web server in this type of messaging is simple. We want to have a reliable system that can use as minimal resources as possible but to give us the highest outcome. Some web services can take a longer time to process, and using synchronous messaging is where our downfall will occur. We need to have a messaging system that allows for multiple concurrent processes, but the caveat behind this is that there are more overhead and reliability issues, but that being said an asynchronous messaging system will always be better in distributed systems as they are more reliable, and there is less room for error as there is human support in the process. Because of these differences between a synchronous messaging system, we can see how an asynchronous messaging primer would be more beneficial to our web server.

What is the role of a web server in Queue-Based Leveling Pattern:

The role of a web server in a Queue based leveling pattern is to have the queue act as a buffer between a task and a service to ensure that heavy loads will be scalable and reliable elsewise there is a chance that the request can time out. An example is the Azure web server stores data using a separate container but if many instances run on a single container, then there is a chance of the server failing. This is where asynchronous messaging primer comes into play where the tasks can be sent to the message queue and be processed at a more consistent rate before it is sent off to the service.

What can you do with CGI? What are the pros and cons? How can we enhance CGI?

The things we can do with CGI is allow people to not have to edit the source of their web server to add new functionality, so the mechanism called Common Gateway Interface (CGI) eliminates this issue by providing a way for a web server to run an external program in order to satisfy a request.

Some pros and cons are as follows:

Pros	Cons
Is language independent, allows user to write programs in the language of their choice (must be valid language for web server)	Cannot easily access data between page loads
Easy to use UI for sending GET requests from HTTP	Can impose high level of security issues regarding data access
Does not need any special libraries	Uses many resources and is not time efficient
Captures whatever the subprocess sends to standard output	GET and PUT requests are slow

The way we can enhance CGI protocols is by assigning health parent classes for the RequestHandler and the other case handlers as well. However we never want to have a method to have more than one handler. Doing so will increase the efficiency of the do_GET class and will allow for uploading files per case handler to take its own individual task rather than have one do everything and in turn, will eliminate many of the time related cons that we had previously listed.

A web server is just one piece of the “enterprise architecture” . What are the other pieces?

The other pieces of the enterprise architecture, starting from the top, is the business architecture. After that, it drives down to the information architecture that then prescribes itself onto the information systems architecture. After it is identified, it can translate its data to the data architecture, which is supported by the delivery systems architecture that all work in cohesion with the web server.

How can such architecture handle very large amounts of data?

The reason it can handle large amounts of data is because of the current architecture allows it to enable the development capabilities to create values from the technology. With a bottom up approach, the raw data was kept within the system and recording the metadata for the website analytics while it uses the information architecture to constantly have real-time data sources capture real-time messages.

Where would you fit the database?

Using different database models, and individualizing them to use different data sources. For example, having a model expose itself to an API template or to a web server, would allow us to fit the information and spread it across the database as required.

How can the operating system support such a web server?

The operating system would be able to have it by actively being scalable in response to the database that is working with the web server. After that, the rest of the caching can simply be done by the VM itself without putting any load on the operating system. As long as the operating system remains scalable and available for the static files that get used, the operating system can consistently delegate them.