Faculty of Engineering & Applied Science

SOFE4790U – Distributed Systems

Group Work

Homework – Web Server Software

Due Date: 10/17/2022

First Name	Last Name	Student ID
Abdul	Bhutta	100785884
Alexander	Campbell	100703650
Mamun	Hossain	100553073
Owen	Musselman	100657709

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

Physical layer is used to have the medium transfer information over when a request has been made. The data link layer formats the data so that the request is placed into packet so that is can be sent. The data link layer is where the connections are terminated. In the Network, Transport and Session layers are where the packets are being sent between the server and client, and the packets can be resent, and that the connection can be an open communication channel between both the server and the client. The presentation and application layers contain the information that was sent from the server to the client which will be displayed via html or other means.

HTTP is used to get information, using get requests, and for uploading using post requests. Additionally, the HTTP url has the ability to contain various parameters that can provide more information like searches, the browser the client is using, etc. HTTP is stateless meaning that each request is handled independently, so the server does not store any information about that request. If information needs to be stored for a request, it can be done through cookies. A new cookie is made and then stored in the database, and then sent to the client. Whenever the client reconnects, the client will send the cookie to the server, where the server checks the database for that cookie and displays the page that contains the cookie information. It is important to note that HTTP is used mainly for data transfer over IP.

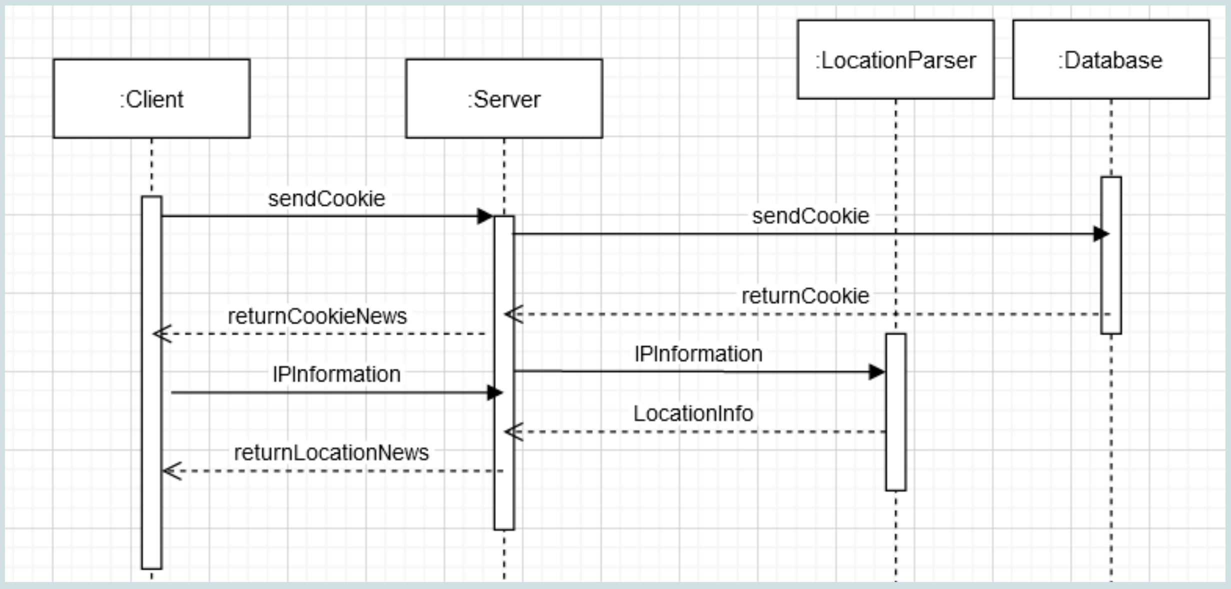
DNS's purpose is to assign the server's 8-bit IP address to a that is chosen by the server owner, as to make the site more easily remembered by clients. In essence, it maps names to IP addresses.

HTML is used to display the web page, the webpage can be be retrieved to display state information of a previous state using cookies.

Take CNN.Com for example, their webserver updates the information displayed on each client? How can a web server do this? Draw UML sequence diagrams? What functions/ components need to be added to the described system?

Using cnn.com as an example, their web server updates the information that is displayed on each client's browser by possibly getting their general location by their IP address, and displaying news that is more applicable to that location. Another their web server can display more personalized information by using cookies, and displaying news that is similar to what searches have been made, or previous visits to the cnn site.

UML sequence diagram of CNN web server



The functions/components added are the location parser and the database. Where the database holds the cookie information of the clients, where it is used to validate the clients accessing the site with previous sessions. The web server will present the client with a page that reflects their previous session information. The location parser is used to get the general location of the client as to have a location based web page presented to the client upon access.

How are transparencies provisioned within the described image sharing system?

The following transparency is provisioned within the image-sharing web server,

- Access Transparency: The data of many images stored on multiple servers are hidden from the client.

- Location Transparency: The data returned to the user, such as an image or a document, will be displayed, but no information regarding the physical location of the file will be accessed by the user.

- Migration Transparency: The image being moved or transferred onto a new physical server, or the directory of the image being updated within the server is changed. The user should not be impacted when they are accessing the website.

What is synchronous messaging" or request reply messaging? what is the role of a web server in this type of messaging? How is that different from Asynchronous Messaging Primer?

Synchronous messaging is best described as two-way communication among two systems where one system sends a message, and the other is waiting for the message. The web server and the client use transmission control protocol (TCP) and TCP handshake to communicate. When the client visits a website (CNN.com), it will send a request to the server, and the client will wait. Once the connection is established, the server will send an acknowledgment to the client. The client will request the information, and the database will retrieve the data (CNN webpage with the images). Afterwards, it will transmit the message to the client with an acknowledgment. An example of synchronous messaging is when a user purchases an item during online shopping. At the same time, the application waits to check if there is enough inventory before proceeding to check out. Asynchronous messaging is one-way communication between two systems where the sender sending the data or information does not need acknowledgment from the receiver. An example of asynchronous messaging is an application sending NBA score updates throughout the day.

Synchronous messaging/request reply messaging will block the execution of code in the browser, which inherently pauses the user experience until a response is received. This is done when the client requests to visit a page on a web server, where the client requests the page, the client will then wait for the request to be served. The server will respond to the request with the requested page. The client then gets the page it requested and the user experience is resumed. Asynchronous messaging is different as there is an intermediate messaging broker which provides functionality of moving messages from the producer to the consumer. The broker can assist in async messaging processing when there are operations being executed that take longer to complete, so if a command is sent, the producer will not have to wait for the consumer to complete with the inclusion of the broker.

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

A queue-based load-levelling pattern allows for a message queue which stores many tasks from different users. The queue behaves as a buffer for the tasks to be stored and waits until the service is ready to execute that task. The webserver runs the queue-based load level pattern on the server and continuously keeps the tasks in the queue until the service is ready to process the next task. This allows the webserver to accept many incoming requests and level the load for each service.

What can you do with CGI ? what are the pros and cons? How can we enhance CGI? Read about three-tier architecture and multi-tier architectures ... How did this functionality evolved from CGI?

CGI is a software interface that allows programs to communicate with programs on different machines. This is done by receiving and forwarding data.

Pros:

- Gives a standardized way to access data from different sources, which can be useful in many applications like file access of different computers, and web browsing, etc.
- Provides communication between networks.
- Allows for heterogeneous systems, which increases the efficiency of resources and performance.

Cons:

- Can cause security risks because it provides an easy way to access data which can be used maliciously.
- Will need changes in the network architecture that may not be feasible for some networks or organizations.

CGIs can be enhanced through the utilization of server specialized gateways that have separate data-forwarding capabilities. This evolved into multi-tier architecture. CGIs are necessary for this process to forward and receive information to various machines throughout the layers. CGIS can be enhanced by plugins like Fast CGI where the plugin enables web servers to work safely in conjunction with python and Perl (content generation technologies). CGI can be enhanced to be more secure so that if the user knows the path to a python file on the server, they will not be able to execute it. Essentially, CGI is implemented through the business layers since it acts like the middleman, that is able to communicate with external resources.

A web server is just one piece of the "enterprise architecture" what are other pieces? How can such architecture handle very large amounts of data? Where would you fit a Database? How can the operating system support such a webserver?

The other pieces of the enterprise architecture is the business architecture. Next it drives down the information architecture that prescribes itself onto information systems architecture. Next it translates its data to data architecture, which is supported by the delivery systems architecture that all work cohesively with the web server.

The architecture can handle large amounts of data through the use of a bottom up approach where the raw data is stored in the system and recording the metadata for the website analytics. It uses the information architecture that will always have real time data sources that are capturing messages in real time. The architecture is also able to handle large amounts of data due to the utilization of the cloud for containerization, and virtualization allowing for load balancing, request splitting, distributed mass storage, and resource scalability and elasticity.

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.

A web server is usually installed on the server Operating system, which allows the user to deploy or install business layer and web applications.

====================== END OF GROUP WORK ======================

Homework 3: Web Server Software
Member: Abdul Bhutta
Changes: RED

Webserver Summary

Each program on the internet will communicate with IP protocol (32-bit number), while the TCP/IP is used to communicate with two different systems. When two computers communicate with each other at each endpoint are called sockets. At each socket, the computer will have an IP address and a port number where the IP address is the source, and the port identifies the specific application. A DNS (Domain Name Service) translates the IP address into a name service to allow for easy access.

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

Functional Requirements

- The server must have an IP
- Must have a socket for communication
- Must support TCP/IP
- Each socket must have a port number associated with it.
- Parse each request and what's associated with the request
- Fetch the data
- Format the data in HTML format
- Transmit the data
- CGI Protocol
- Data Fetch
- Listing Directories
- Format Data

At what layers of the OSI models does this system operate? What is the role of HTTP, DNS, HTML in the described system?

Layer	Operation	Protocol
Application Layer	The Applications such as DNS will run here.	HTTP

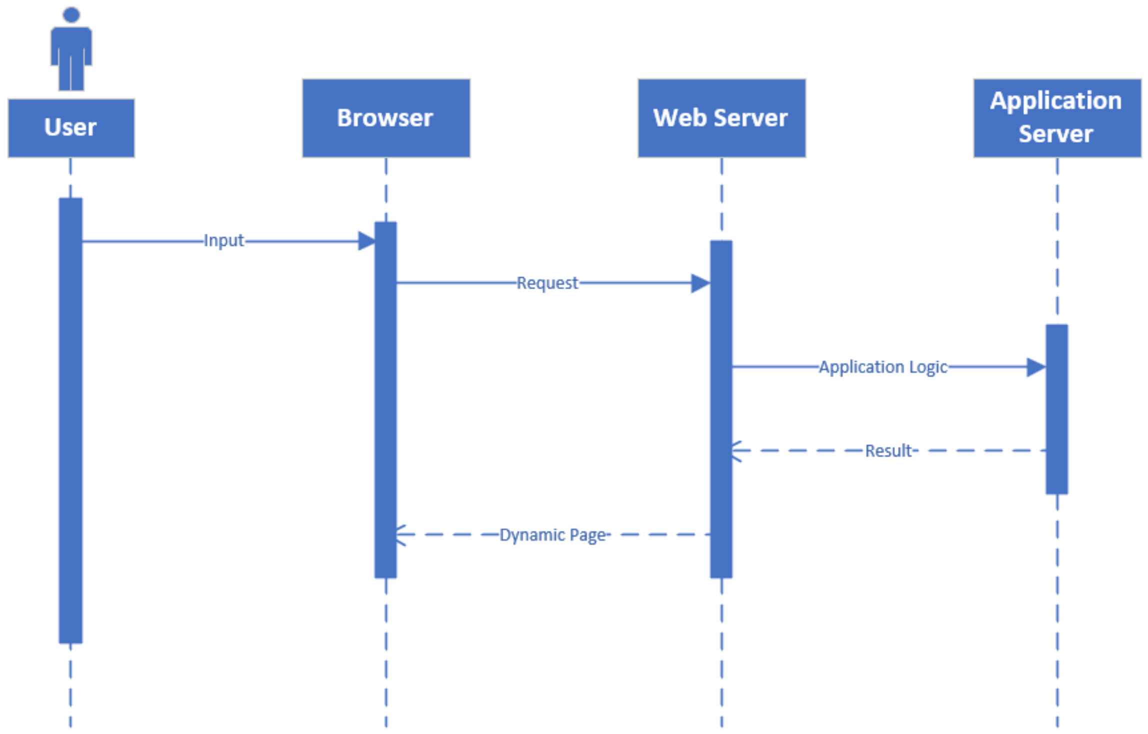
Presentation Layer	If the web server is sending or receiving files.	HTTP
Session Layer	The socket will use IP and port number to establish a connection with the application.	HTTP
Transport Layer	It is used to communicate with other systems through an end-to-end connection.	TCP/IP
Network Layer	The webserver must be connected to the outside world through a router to communicate with other client requests.	IP
Data Link Layer	For a web server to communicate to clients over the internet, it must have a NIC card and a MAC address.	Ethernet
Physical Layer	Each web server must be connected through an Ethernet cable.	Ethernet

Protocol	**Role**
HTTP	HTTP (Hyper Text Transfer Protocol) is used to communicate between the client and server while allowing them to exchange data. The role of the HTTP protocol in the simple web server application is it uses this protocol while allowing the client to make requests to the server and the server responds back with the required data.
DNS	The DNS is used to map IP addresses to a name such as google.ca, which helps user to remember the name rather than the 4-byte IP address.
HTML	HTML (HyperText Markup Language) is a standard language to display a document on a web browser.

Take CNN.Com for example, their webserver updates the information displayed on each client? How can a web server do this? Draw UML sequence diagrams? What functions/ components need to be added to the described system?

CNN uses an application server as a middleware to update the information on the client side. The web server allows communication between the client while the application server implements business logic and produces dynamic content for multiple users. The data provided is based on the region or location of the request through the IP address, then displays the image or content. The application server component must be added to the simple web server, and a UML sequence diagram is provided below.

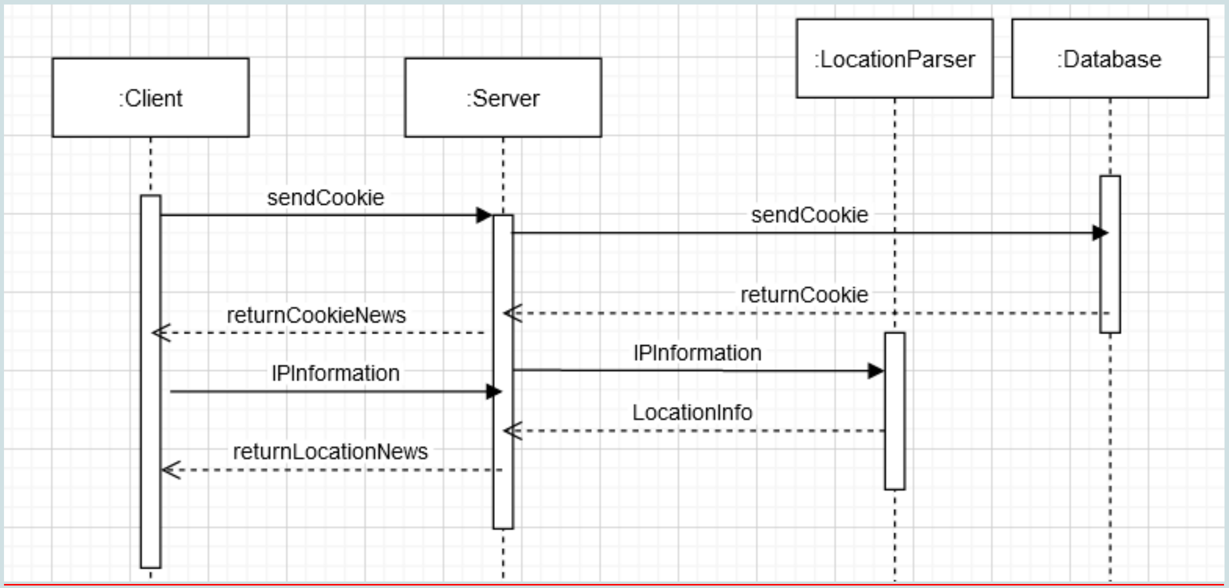
Sequence Diagram



//Don't know which one is right! VAGUE QUESTION

Using cnn.com as an example, their web server updates the information that is displayed on each client's browser by possibly getting their general location by their IP address, and displaying news that is more applicable to that location. Another their web server can display more personalized information by using cookies, and displaying news that is similar to what searches have been made, or previous visits to the cnn site.

UML sequence diagram of CNN web server



The functions/components added are the location parser and the database. Where the database holds the cookie information of the clients, where it is used to validate the clients accessing the site with previous sessions. The web server will present the client with a page that reflects their previous session information. The location parser is used to get the general location of the client as to have a location based web page presented to the client upon access.

How are transparencies provisioned within the described image sharing system?

The following transparency is provisioned within the image-sharing web server,

- Access Transparency: The data of many images stored on multiple servers are hidden from the client.
- Location Transparency: The data returned to the user, such as an image or a document, will be displayed, but no information regarding the physical location of the file will be accessed by the user.
- Migration Transparency: The image being moved or transferred onto a new physical server, or the directory of the image being updated within the server is changed. The user should not be impacted when they are accessing the website.

What is synchronous messaging" or request reply messaging? what is the role of a web server in this type of messaging? How is that different from Asynchronous Messaging Primer?

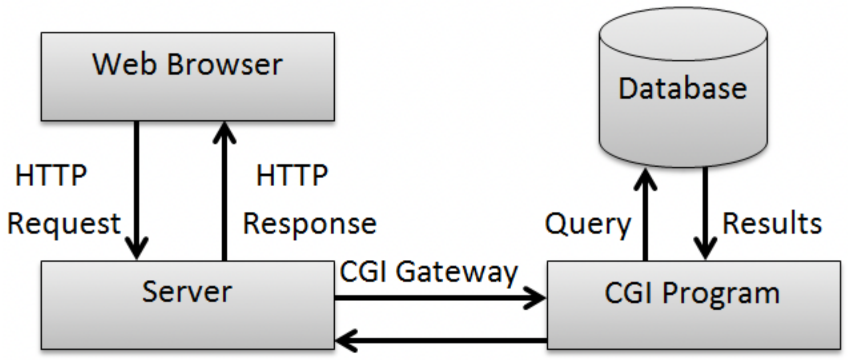
Synchronous messaging is best described as two-way communication among two systems where one system sends a message, and the other is waiting for the message. The web server and the client use transmission control protocol (TCP) and TCP handshake to communicate. When the client visits a website (CNN.com), it will send a request to the server, and the client will wait. Once the connection is established, the server will send an acknowledgment to the client. The client will request the information, and the database will retrieve the data (CNN webpage with the images). Afterwards, it will transmit the message to the client with an acknowledgment. An example of synchronous messaging is when a user purchases an item during online shopping. At the same time, the application waits to check if there is enough inventory before proceeding to check out. Asynchronous messaging is one-way communication between two systems where the sender sending the data or information does not need acknowledgment from the receiver. An example of asynchronous messaging is an application sending NBA score updates throughout the day.

Synchronous messaging/request reply messaging will block the execution of code in the browser, which inherently pauses the user experience until a response is received. This is done when the client requests to visit a page on a web server, where the client requests the page, the client will then wait for the request to be served. The server will respond to the request with the requested page. The client then gets the page it requested and the user experience is resumed. Asynchronous messaging is different as there is an intermediate messaging broker which provides functionality of moving messages from the producer to the consumer. The broker can assist in async messaging processing when there are operations being executed that take longer to complete, so if a command is sent, the producer will not have to wait for the consumer to complete with the inclusion of the broker.

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

A queue-based load-levelling pattern allows for a message queue which stores many tasks from different users. The queue behaves as a buffer for the tasks to be stored and waits until the service is ready to execute that task. The webserver runs the queue-based load level pattern on the server and continuously keeps the tasks in the queue until the service is ready to process the next task. This allows the webserver to accept many incoming requests and level the load for each service.

What can you do with CGI ? what are the pros and cons? How can we enhance CGI? Read about three-tier architecture and multi-tier architectures ... How did this functionality evolved from CGI?



The common gateway interface (CGI) allows static HTML pages to run external programs such as scripts to generate dynamic web pages. It acts as a middleware between the web server and databases. A three-tier architecture consists of a presentation layer, a business layer, and a data layer. The presentation layer is where the user interface is displayed, which shows the application's functions that the client can understand. The business layer is where the calculations or processing are done. Lastly, the data layer is used to store or retrieve data from a database which is used in the logic layer for processing and forwarded to the presentation layer. The common gateway interface is between the database and the server, as shown in the image above. This architecture runs various programs for the user or some logic to display on the webpage, which helped evolve into an n-tier architecture where the CGI is evolved to a business (application) layer. In the business layer, it performs various processes and rather than just producing dynamic web pages, it also decides what is accepted by the application and what functionality to perform. The pros and cons of CGI is displayed below.

Pros	Cons
Allows the program to be written with any language	Overheads from loading page into memory.
Provides an interactive dynamic webpage for the user	Consumes a lot of processing time
CGI runs on the servers which are secure	Hackers can view the host system through CGI allowing them to access confidential files.

A web server is just one piece of the "enterprise architecture" what are other pieces? How can such architecture handle very large amounts of data? Where would you fit a Database? How can the operating system support such a webserver?

An enterprise architecture contains many pieces besides a web server. A few components integrated into an enterprise architecture are an application server, storage, OS, load balancer, database, and gateway. In the current era, cloud computing is the center of most enterprises. Much data is stored in a data center where all the databases reside. It is extremely difficult to process all the data on one machine, where cloud computing is integral to managing big data. It helps deal with “big data” and processing through resources being pooled together in the cloud, and the cloud also provides you with a database built in the cloud (DBaaS) that can be managed on a virtual OS. A web server is usually installed on the server OS, which allows the user to install or deploy business layer applications and web applications.

Homework 3: Web Server Software
Member: Owen Musselman

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

- Parse the incoming requests to figure out what method is within the request.
- Create or find the page that the client is requesting.
- Send the page to the client that was requesting it.
- Check if the client requested page exists and send it to them if it exists.
- Need an error handler which sends the client errors if pages they request do not exist.
- Require a server exception handler which will signal if there are any internal errors in the server code.
- Need an IP and port so that it can be assigned a domain name.

At what layers of the OSI models does this system operate? What is the role of HTTP, DNS, HTML in the described system?

The OSI layers used in this system are 5, 4, 3, 2, 1. This is due to this system needing to communicate with the client which is done at layer 5, but in order for that to happen a session needs to be made which is done at this layer. Layer 5 will setup, coordinate and terminate the communication between the application involved. Layer 4 is used as it is responsible for for coordination of the data transfer between the server and the client, like where the client is, and at what rate the data needs to be transferred to the client. Layer 4 is where the TCP/IP protocol is. Layer 3 is used as it forwards the packers along a route. Layer 2 is used as it is responsible for error correction, along with node to the node data transfer. Layer 1 is used as it is the physical medium by which the data is transferred, where requests come in, and where responses go out.

The role of HTTP in this system is to request information using get requests. For the role of uploading information to the server, post requests are used. The HTTP url has the ability to have parameters included in it which provides even more information, like searches, and client browsers, etc. Additionally, HTTP is stateless, which means that each individual request is handled independently, with the server not storing anything about that request, if that information is needed it can be done in the form of cookies. Cookies are a short string of characters in which a server sends to a client. If the client performs actions that require the state of the client to be saved, a new cookie is made and then stored in a database, and then sent to the client. Any time the client reconnects the client sends the cookie to the server and the server checks the database for it and displays the page according to the saved state. HTTP is mainly used for data transfer over IP.

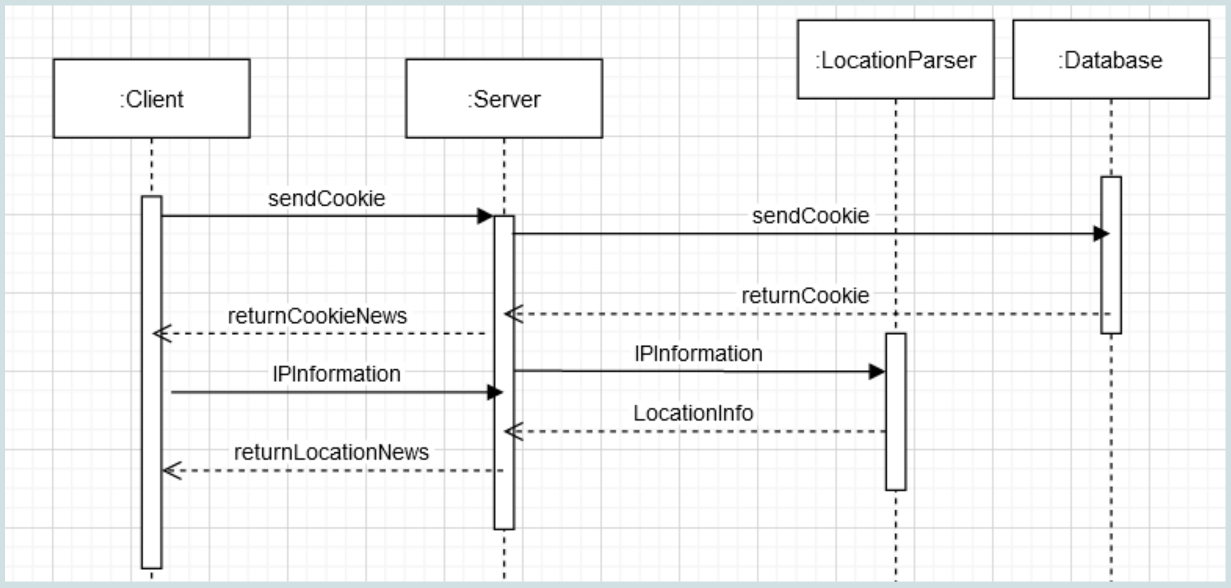
The role of DNS in this system is to just assign the server's 8-bit IP address to a name that is chosen by the server owner so that the address to the server's site can be easily remembered by clients. For example an IP of 24.224.35.149 is assigned to foobar.com and instead of people remembering 24.224.35.149, they can just remember foobar.com.

The role of HTML is to display the generated and the retrieved page that the client had requested from the server to the client's browser. Headers are used to tell the client to interpret the received data from the server as HTML.

Take CNN.Com for example, their webserver updates the information displayed on each client? How can a web server do this? Draw UML sequence diagrams? What functions/ components need to be added to the described system?

Using cnn.com as an example, their web server updates the information that is displayed on each client's browser by possibly getting their general location by their IP address, and displaying news that is more applicable to that location. Another their web server can display more personalized information by using cookies, and displaying news that is similar to what searches have been made, or previous visits to the cnn site.

UML sequence diagram of CNN web server



The functions/components added are the location parser and the database. Where the database holds the cookie information of the clients, where it is used to validate the clients accessing the site with previous sessions. The web server will present the client with a page that reflects their previous session information. The location parser is used to get the general location of the client as to have a location based web page presented to the client upon access.

How are transparencies provisioned within the described image sharing system?

The transparencies provisioned within this system is that the client has no idea how the requests are being sent to the server, or how the responses are being sent. That information is hidden from the client's view. Another transparency is the Error handling, where the client only sees the status messages: informational responses (100-199), successful responses (200-299), redirect responses (300-399), client error responses (400-499), server error responses (500-599). Additionally the client is presented with the responses information like http version, header field names they are only presented with the body which is usually html. When the client gets a response with html it is interpreted by the browser and displays it accordingly, it doesn't just display it as html code.

The following transparencies are provisioned within the image-sharing web-server:

- Access Transparency: The data of many images stored on multiple servers are hidden from the client.
- Location Transparency: The data returned to the user, such as an image or a document, will be displayed, but no information regarding the physical location of the file will be accessed by the user.
- Migration Transparency: The image being moved or transferred onto a new physical server, or the directory of the image being updated within the server is changed. The user should not be impacted when they are accessing the website.

What is ' synchronous messaging" or request reply messaging? what is the role of a web server in this type of messaging? How is that different from Asynchronous Messaging Primer?

Synchronous messaging/request reply messaging will block the execution of code in the browser, which inherently pauses the user experience until a response is received. This is done when the client requests to visit a page on a web server, where the client requests the page, the client will then wait for the request to be served. The server will respond to the request with the requested page. The client then gets the page it requested and the user experience is resumed. Asynchronous messaging is different as there is an intermediate messaging broker which provides functionality of moving messages from the producer to the consumer. The broker can assist in async messaging processing when there are operations being executed that take longer to complete, so if a command is sent, the producer will not have to wait for the consumer to complete with the inclusion of the broker.

Synchronous messaging is best described as two-way communication among two systems where one system sends a message, and the other is waiting for the message. The web server and the client use transmission control protocol (TCP) and TCP handshake to communicate. When the client visits a website (CNN.com), it will send a request to the server, and the client will wait. Once the connection is established, the server will send an acknowledgment to the client. The client will request the information, and the database will retrieve the data (CNN webpage with the images). Afterwards, it will transmit the message to the client with an acknowledgment. An example of synchronous messaging is when a user purchases an item during online shopping. At the same time, the application waits to check if there is enough inventory before proceeding to check out. Asynchronous messaging is one-way communication between two systems where the sender sending the data or information does not need acknowledgment from the receiver. An example of asynchronous messaging is an application sending NBA score updates throughout the day.

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

The role of the web server in the queue-based load leveling pattern is that the web server will direct the various tasks to a service bus queue, which the queue then decouples the tasks from the services, so that the services are able to handle messages now, even if there are a large number of requests coming from the web server.

What can you do with CGI ? what are the pros and cons? How can we enhance CGI? Read about three-tier architecture and multi-tier architectures ... How did this functionality evolved from CGI?

CGI is used to grant a way for a web server to execute programs externally to satisfy a request. Some pros of CGIs are: give a simple interface for creating HTTP requests to a web server. You do not need special libraries/APIs to make a CGI. CGIs can be written in many languages, support for CGIs is very extensive. Some cons of CGIs: Lots of processing time taken. HTTP requests to the server can become slow. It is harder to cache data in memory when going between pages. CGI can be enhanced through the use of plugins like FastCGI where the plugin enables a web server to safely work in conjunction with content generating technologies like python and perl. We can also enhance CGI to be more secure so even if a user knows the path to a python file stored on the server they cannot execute it, or only some sections of it. multi -tier architectures evolved from CGI as, like the name suggests there are multiple layers in an architecture, and the CGI is the middle man interface which web servers pass the client requests to the databases and other resources. In essence CGI is implemented through the business logic layer as it acts as the middleman to communicate with the external resources.

A web server is just one piece of the "enterprise architecture" what are other pieces? How can such architecture handle very large amounts of data? Where would you fit a Database? How can the operating system support such a webserver?

There are various parts of an enterprise system, with there being security, application, infrastructure, information, networks, integration. The web server fits into the infrastructure portion, with the other components of a system falling into other sections. Enterprise architecture can handle large amounts of data due to the ability to utilize the cloud for containerization, and virtualization, which allows for load balancing, request splitting, and resource scalability and elasticity. The architecture can handle large amounts of data through the use of a bottom up approach where the raw data is stored in the system and recording the metadata for the website analytics. It uses the information architecture that will always have real time data sources that are capturing messages in real time.

The other pieces of the enterprise architecture is the business architecture. Next it drives down the information architecture that prescribes itself onto information systems architecture. Next it translates its data to data architecture, which is supported by the delivery systems architecture that all work cohesively with the web server.

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.

A web server is usually installed on the server Operating system, which allows the user to deploy or install business layer and web applications.

Homework 3: Web Server Software
Member: Alexander Campbell

Software Requirements of a Web Server

- Internet Protocols: A family of communication protocols that are used by the internet for programs and machines to communicate to each other
- Sockets: A socket is an end-to-end communication device that uses a port assigned to it on the host computer and an IP address.
- HyperText Transfer Protocol: This is one protocol that uses text to transfer data between machines or programs. It involves the client sending a request for some information and the server responding to it with the information or some other request depending on the success of retrieving the information.

At what layers of the OSI models does this web server system operate? What is the role of HTTP, DNS, HTML in the described system?

In a web server, technically all layers of the OSI model are utilized in some form. At the lowest level, the Physical layer, there are actual physical devices that transfer the information when any request is being made. At the Data Link layer we have the formatting of the data occurring, where the request is put into a packet that can be sent off to the server it's requesting from by the Physical layer. This is also where connections to the server from the client may be terminated. In the Network, Transport, and Session layers these information packets that are being sent between the client and the server can have these packets resent and ensure there's an open communication channel between the client and server. Finally in the Presentation and Application layer we have the information that has been sent to the client from the server being displayed in some manner.

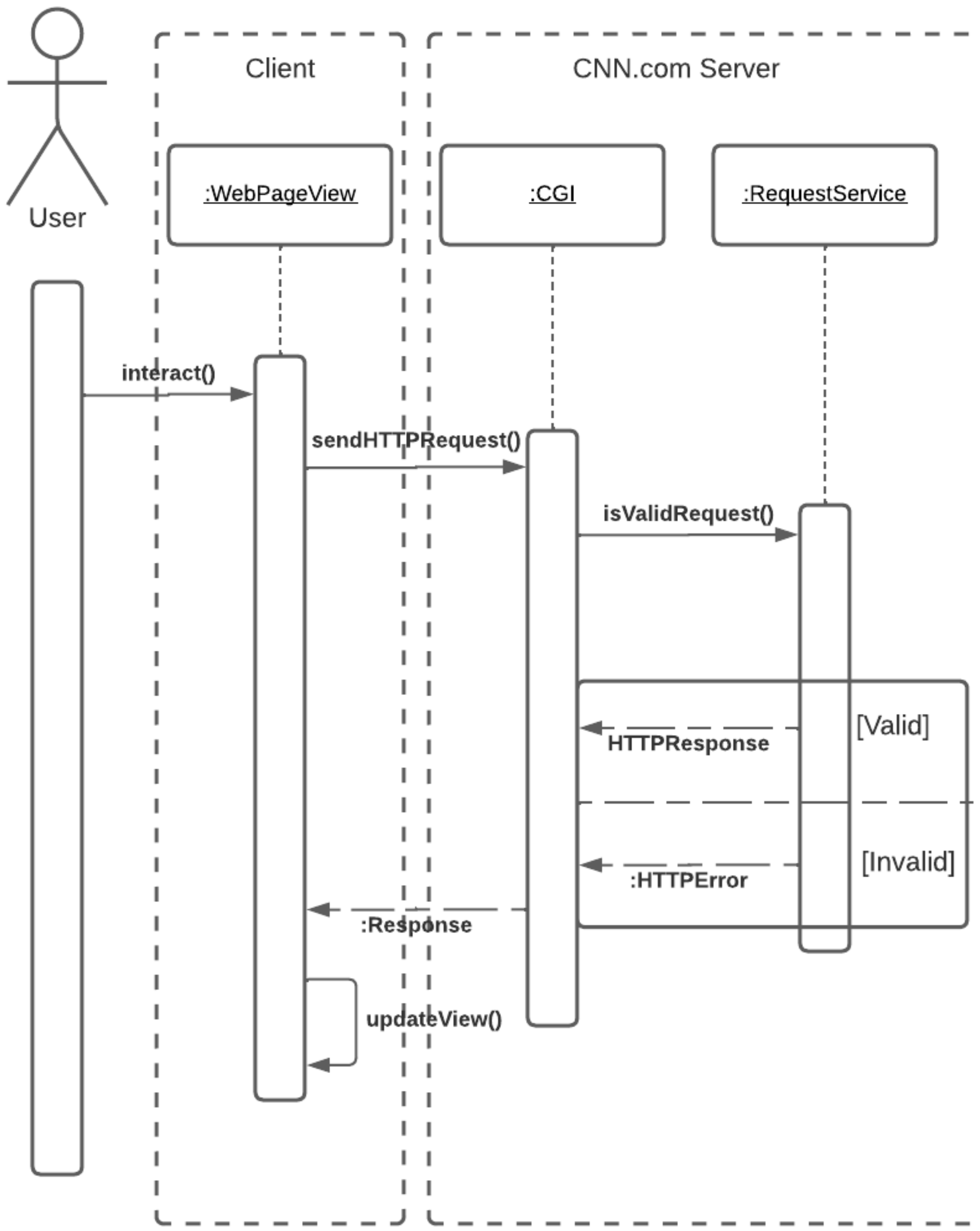
Both HTTP and DNS are both in the Application layer. HTTP at its core are just information requests from a client to server, these requests allow the user to interact with the server and create requests thus belonging to the Application layer. DNS is a domain name service, which maps names, such as google.com, to connectable IP Addresses. This is also done at the Application layer since the client is requesting the specified domain name that is converted to an IP address when communicating with the server.

HTML is at the Presentation layer of the OSI model. The reason is HTML is the formatted data request that the client can view in a readable format.

How are transparencies provisioned within the described image sharing system?

Replication transparency is very important in a web server. It ensures that the image sharing system's data is readily available but also hidden from the users that want to access the data that corresponds to the web server. Access transparency is also provisioned where user's don't need to know the physical location of the web server, just the DNS.

Take CNN.Com for example, their web server updates the information displayed on each client? How can a web server do this? Draw UML sequence diagrams? What functions/ components need to be added to the described system?



What is ‘synchronous messaging’ or request reply messaging? What is the role of a web server in this type of messaging? How is that different from Asynchronous Messaging?

Synchronous messaging is a type of request-reply messaging in which a response is required from the recipient of a message. It can be used in many different scenarios, such as to send notifications or to provide updates on a process. The role of a web server in request reply messaging is to provide a service to users who are requesting data from it. The user sends out information to the server and then receives information back from it. Asynchronous messaging refers to a message-passing system in which messages are not sent or received in real time, meaning that messages may be sent at any time with no guarantee the recipient has received them. This is different from synchronous messaging in which the recipient is guaranteed to receive it as there is a response after receiving the information.

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

In Queue-Based Load Leveling Pattern, web servers are the components that have the responsibility to take the request load of the application, receiving requests and then queuing them up. This way, they can distribute the load to multiple servers which will avoid overloading any single server and provide better performance to users.

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

A common gateway interface (CGI) is a software interface that allows programs on one machine to communicate with programs on another machine by receiving and forwarding data.

Pros:

- Allows for communication between networks, which can lead to new innovations in the field of networking and distributed systems.
- Allows for the integration of heterogeneous systems, leading to more efficient use of resources and better performance.
- Provides a standardized way to access data from different sources, which can be useful in many applications such as web browsing or accessing files on your computer.

Cons:

- Introduces some security risks because it provides an easy way for hackers to access your data if you have not taken steps to secure it properly.
- Requires changes in the network’s architecture that may not be feasible for some networks or organizations.

How can we enhance CGI? Read about three-tier architecture and multi-tier architectures. How did this functionality evolve from CGI?

CGIs can be enhanced by utilizing several specialized gateways that have separate data forwarding capabilities. This later evolved into multi tier architectures where data management, processing, and presentation are separated in a program. CGIs are needed in this process to receive and forward information to various machines to the various layers.

A web server is just one piece of the "enterprise architecture", what are other pieces?

The components that make up enterprise architecture are as follows:

- Data management
- Networking
- Databases and data storage
- Operating systems
- Virtualization

How can such architecture handle very large amounts of data? Where would you fit a database? How can the operating system support such a web server?

The enterprise architecture can handle large amounts of data through the use of data management components. These components allow for efficient and safe storage of data such that the business can easily scale their databases up or down depending on their need.

Homework 3: Web Server Software
Member: Mamun Hossain

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.

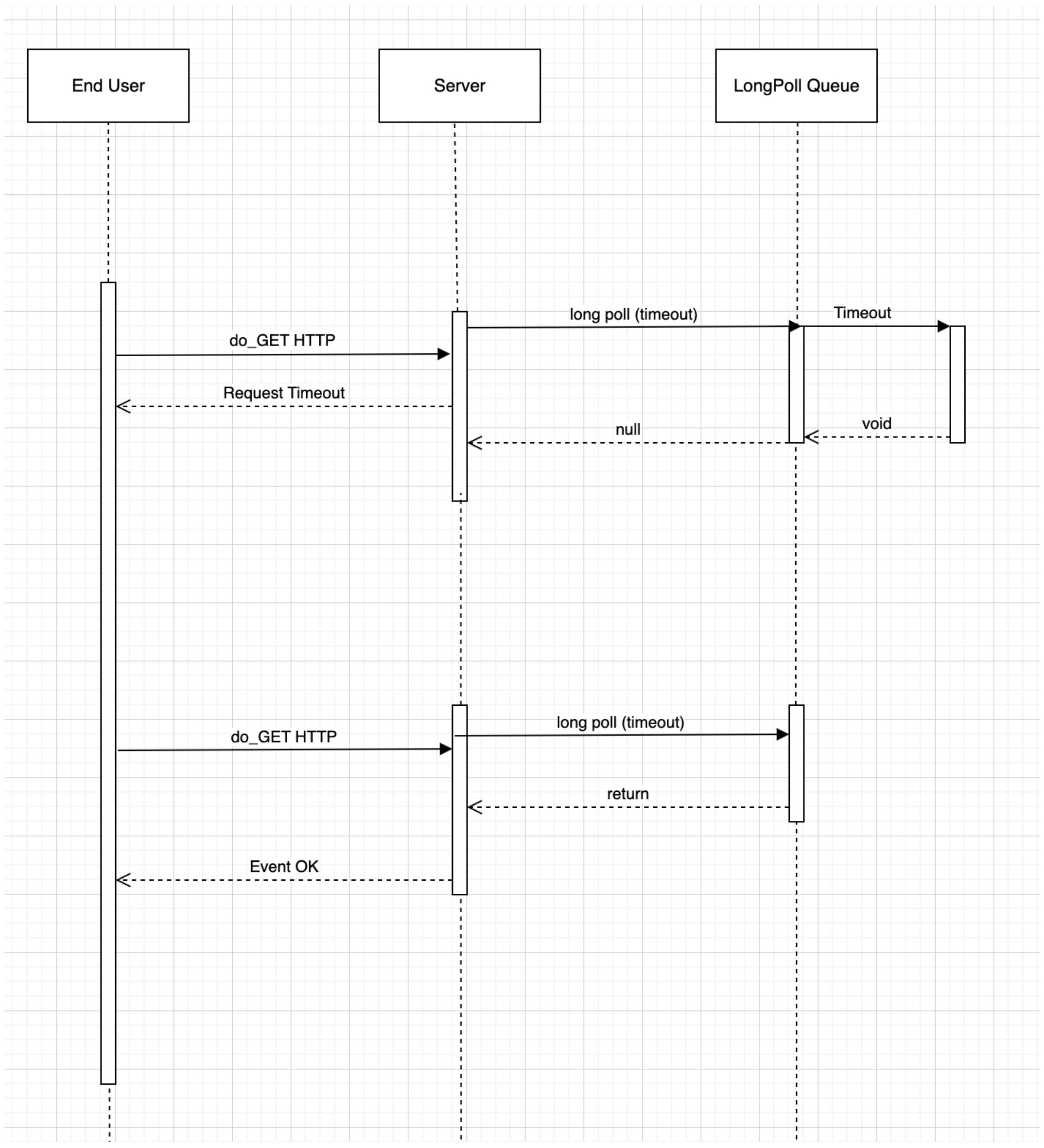
Draw a UML sequence diagram

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.