CBTis#42

Implementa el diseño físico de la base de datos.

Actividad 4-Unidad 2

Programación

MÓDULO IV:
DESARROLLA SOFTWARE DE APLICACIÓN WEB CON ALMACENAMIENTO PERSISTENTE DE DATOS.

SUBMÓDULO 1:
CONSTRUYE BASES DE DATOS PARA APLICACIONES WEB.

Elaborado por: Abraham Castañeda Quintero
Docente: Alejandra Serrano Luna
Semestre y grupo: 5° “B”

Índice.

Contenido

Introducción.

En esta la actividad No. 4 unidad 2 del modulo IV de la carrera de programación, Es necesario saber como funciona el modelo de cliente-servidor y sus distintas características además de sus ventajas y desventajas esto para entender como es que se distribuyen los sitios web ya sean estáticos o dinámicos funcionan utilizando el modelo de cliente-servidor.

Desarrollo.

-¿Qué es el modelo cliente servidor?

La arquitectura cliente-servidor es un modelo de aplicación distribuida en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes. Un cliente realiza peticiones a otro programa, el servidor, quien le da respuesta. Esta idea también se puede aplicar a programas que se ejecutan sobre una sola computadora, aunque es más ventajosa en un sistema operativo multiusuario distribuido a través de una red de computadoras.

Algunos ejemplos de aplicaciones computacionales que usen el modelo cliente-servidor son el Correo electrónico, un Servidor de impresión y la World Wide Web.

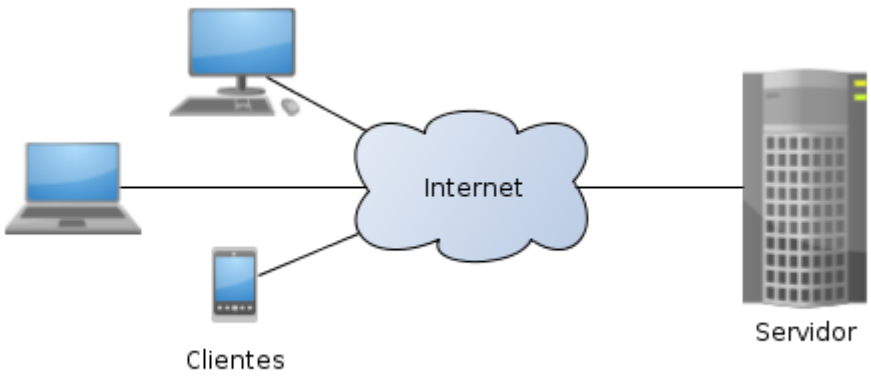
Segunda definición:

En esta arquitectura la capacidad de proceso está repartida entre los clientes y los servidores, aunque son más importantes las ventajas de tipo organizativo debidas a la centralización de la gestión de la información y la separación de responsabilidades, lo que facilita y clarifica el diseño del sistema.

La separación entre cliente y servidor es una separación de tipo lógico, donde el servidor no se ejecuta necesariamente sobre una sola máquina ni es necesariamente un solo programa. Los tipos específicos de servidores incluyen los servidores web, los servidores de archivo, los servidores del correo, etc. Mientras que sus propósitos varían de unos servicios a otros, la arquitectura básica seguirá siendo la misma.

Tercera definición:

Este modelo es uno de los principales usados en muchísimos servicios y protocolos de Internet, por lo que para todos aquellos que quieren aprender más sobre la web y cómo funciona, entender el concepto de modelo cliente servidor se vuelve algo indispensable.



Comparación de la arquitectura C/S con otras arquitecturas de red

Comparación con las redes de pares

Las **redes de pares**, también conocidas como redes **par-a-par** o **peer-to-peer** (abreviado con las siglas **P2P**) son otro tipo de arquitectura de red.

Comparación con la arquitectura Cliente-Cola-Cliente

Si bien la clásica arquitectura C/S requiere uno de los puntos terminales de comunicación para actuar como un servidor, que puede ser algo más difícil de aplicar, la arquitectura Cliente-Cola-Cliente habilita a todos los nodos para actuar como clientes simples, mientras que el servidor actúa como una cola que va capturando las peticiones de los clientes (un proceso que debe pasar sus peticiones a otro, lo hace a través de una cola, por ejemplo, una consulta a una base de datos, entonces, el segundo proceso conecta con la base de datos, elabora la petición, la pasa a la base de datos, etc.). Esta arquitectura permite simplificar en gran medida la implementación de software. La arquitectura P2P originalmente se basó en el concepto "Cliente-Cola-Cliente".

Arquitecturas multi-capas

La arquitectura cliente/servidor genérica tiene dos tipos de nodos en la red: clientes y servidores. Consecuentemente, estas arquitecturas genéricas se refieren a veces como arquitecturas de dos niveles o **dos capas**.

Algunas redes disponen de tres tipos de nodos:

- Clientes que interactúan con los usuarios finales.
- Servidores de aplicación que procesan los datos para los clientes.
- Servidores de la base de datos que almacenan los datos para los servidores de aplicación.

Esta configuración se llama una arquitectura de tres-capas.

- Ventajas de las arquitecturas n-capas:

La ventaja fundamental de una arquitectura **n-capas** comparado con una arquitectura de dos niveles (o una tres-capas con una de dos niveles) es que separa hacia fuera el proceso, eso ocurre para mejorar el balance la carga en los diversos servidores; es más escalable.

- Desventajas de las arquitecturas de la n-capas:
- Pone más carga en la red, debido a una mayor cantidad de tráfico de la red.
- Es mucho más difícil programar y probar el software que en arquitectura de dos niveles porque tienen que comunicarse más dispositivos para terminar la transacción de un usuario.

-Características.

para la arquitectura C/S el remitente de una solicitud es conocido como cliente. Sus características son:
Es quien inicia solicitudes o peticiones, tienen por tanto un papel activo en la comunicación (dispositivo maestro o amo).
Espera y recibe las respuestas del servidor.
Por lo general, puede conectarse a varios servidores a la vez.
Normalmente interactúa directamente con los usuarios finales mediante una interfaz gráfica de usuario.
Al receptor de la solicitud enviada por el cliente se conoce como servidor. Sus características son:
Al iniciarse esperan a que lleguen las solicitudes de los clientes, desempeñan entonces un papel pasivo en la comunicación (dispositivo esclavo).
Tras la recepción de una solicitud, la procesan y luego envían la respuesta al cliente.
Por lo general, acepta las conexiones de un gran número de clientes (en ciertos casos el número máximo de peticiones puede estar limitado).
En la arquitectura C/S sus características generales son:
El Cliente y el Servidor pueden actuar como una sola entidad y también pueden actuar como entidades separadas, realizando actividades o tareas independientes.
Las funciones de Cliente y Servidor pueden estar en plataformas separadas, o en la misma plataforma.
Cada plataforma puede ser escalable independientemente. Los cambios realizados en las plataformas de los Clientes o de los Servidores, ya sean por actualización o por reemplazo tecnológico, se realizan de una manera transparente para el usuario final.
La interrelación entre el hardware y el software están basados en una infraestructura poderosa, de tal forma que el acceso a los recursos de la red no muestra la complejidad de los diferentes tipos de formatos de datos y de los protocolos.
Su representación típica es un centro de trabajo (PC), en donde el usuario dispone de sus propias aplicaciones de oficina y sus propias bases de datos, sin dependencia directa del sistema central de información de la organización.
Los servidores pueden ser apátridas o stateful. Un servidor apátrida no guarda ninguna información entre las peticiones. Un servidor stateful puede recordar la información entre las peticiones. El alcance de esta información puede ser global o sesión-específico. Un servidor del HTTP para las páginas estáticas del HTML es un ejemplo de un servidor, apátrida mientras que Apache Tomcat es un ejemplo de un servidor stateful.
La interacción entre el cliente y el servidor se describe a menudo usando diagramas de secuencia. Los diagramas de secuencia se estandarizan en el UML. Es importante que los clientes no interactúen entre sí ni que lo hagan clientes de capas bajas hacia otros de capas más altas, por eso todo tiene que pasar por el servidor.
Otro tipo de arquitectura de red se conoce como arquitectura del par-a-par porque cada nodo o caso del programa es un “cliente” y un “servidor” y cada uno tiene responsabilidades equivalentes. Ambas arquitecturas están en uso amplio.

-Ventajas y desventajas.

Ventajas:

- Centralización del control: los accesos, recursos y la integridad de los datos son controlados por el servidor de forma que un programa cliente defectuoso o no autorizado no pueda dañar el sistema. Esta centralización también facilita la tarea de poner al día datos u otros recursos (mejor que en las redes P2P)..
- Escalabilidad: se puede aumentar la capacidad de clientes y servidores por separado. Cualquier elemento puede ser aumentado (o mejorado) en cualquier momento, o se pueden añadir nuevos nodos a la red (clientes y/o servidores).
- Fácil mantenimiento: al estar distribuidas las funciones y responsabilidades entre varios ordenadores independientes, es posible reemplazar, reparar, actualizar, o incluso trasladar un servidor, mientras que sus clientes no se verán afectados por ese cambio (o se afectarán mínimamente). Esta independencia de los cambios también se conoce como encapsulación.
- Existen tecnologías, suficientemente desarrolladas, diseñadas para el paradigma de C/S que aseguran la seguridad en las transacciones, la amigabilidad de la interfaz, y la facilidad de empleo.
- En las redes C/S los demás clientes no tienen acceso a las IP's por lo que se dificulta el rastreo y/o hackeo de los usuarios

Desventajas:

- La congestión del tráfico ha sido siempre un problema en el paradigma de C/S. Cuando una gran cantidad de clientes envían peticiones simultáneas al mismo servidor, puede ser que cause muchos problemas para este (a mayor número de clientes, más problemas para el servidor). Al contrario, en las redes P2P como cada nodo en la red hace también de servidor, cuanto más nodos hay, mejor es el ancho de banda que se tiene.
- El paradigma de C/S clásico no tiene la robustez de una red P2P. Cuando un servidor está *caído*, las peticiones de los clientes no pueden ser satisfechas. En la mayor parte de redes P2P, los recursos están generalmente distribuidos en varios nodos de la red. Aunque algunos salgan o abandonen la descarga; otros pueden todavía acabar de descargar consiguiendo datos del resto de los nodos en la red.
- El software y el hardware de un servidor son generalmente muy determinantes. Un hardware regular de un ordenador personal puede no poder servir a cierta cantidad de clientes. Normalmente se necesita software y hardware específico, sobre todo en el lado del servidor, para satisfacer el trabajo. Por supuesto, esto aumentará el coste.
- El cliente no dispone de los recursos que puedan existir en el servidor. Por ejemplo, si la aplicación es una Web, no podemos escribir en el disco duro del cliente o imprimir directamente sobre las impresoras sin sacar antes la ventana previa de impresión de los navegadores.

Cuadro Sinóptico.

CLIENTE-SERVIDOR

- -¿Qué es?
  - es un modelo de diseño de software en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes. Un cliente realiza peticiones a otro programa, el servidor, quien le da respuesta. Esta idea también se puede aplicar a programas que se ejecutan sobre una sola computadora, aunque es más ventajosa en un sistema operativo multiusuario distribuido a través de una red de computadoras.
- -Características
  - Es quien inicia solicitudes o peticiones, tienen por tanto un papel activo en la comunicación (dispositivo maestro o amo).
  - Espera y recibe las respuestas del servidor.
  - Por lo general, puede conectarse a varios servidores a la vez.
  - Normalmente interactúa directamente con los usuarios finales mediante una interfaz gráfica
- -Ventajas
  - Centralización del control: los accesos, recursos y la integridad de los datos son controlados por el servidor de forma que un programa cliente defectuoso o no autorizado no pueda dañar el sistema. Esta centralización también facilita la tarea de poner al día datos u otros recursos (mejor que en las redes P2P)..
  - Escalabilidad: se puede aumentar la capacidad de clientes y servidores por separado. Cualquier elemento puede ser aumentado (o mejorado) en cualquier momento, o se pueden añadir nuevos nodos a la red (clientes y/o servidores).
- -Desventajas
  - La congestión del tráfico ha sido siempre un problema en el paradigma de C/S. Cuando una gran cantidad de clientes envían peticiones simultáneas al mismo servidor, puede ser que cause muchos problemas para este (a mayor número de clientes, más problemas para el servidor). Al contrario, en las redes P2P como cada nodo en la red hace también de servidor, cuanto más nodos hay, mejor es el ancho de banda que se tiene.

Conclusión.

En conclusión, en esta actividad aprendí sobre como funciona el modelo cliente servidor asi como sus características y sus ventajas y desventajas además pude ver como es que se comporta y que beneficios tiene con respecto a otros modelos, a mi paracer el modelo cliente-servidor es el mas mejor porque es como funciona la mayoría de la red de internet y también es así como las empresas crean su modelo de negocios invirtiendo en infraestructura de servidores y desarrollo para los mismo, para que los usuarios comunes que este caso son los clientes puedan acceder con mayor facilidad a los servicios.

Referencias.

colaboradores de Wikipedia. (2020, 8 noviembre). *Cliente-servidor*. Wikipedia, la enciclopedia libre. https://es.wikipedia.org/wiki/Cliente-servidor

MODELO CLIENTE SERVIDOR. (s. f.). Mi sitio. Recuperado 12 de noviembre de 2020, de https://redespomactividad.weebly.com/modelo-cliente-servidor.html

Schiaffarino, A. (2019, 7 agosto). *Modelo cliente servidor*. Infranetworking. https://blog.infranetworking.com/modelo-cliente-servidor/