Diseño de sistemas web y sus fundamentos

La arquitectura cliente-servidor sigue ahora más vigente que nunca pero ya no es más como la conocimos en el siglo pasado. Ahora, en busca de la universalización de plataformas (Unix, GNU/Linux, Android, MS Windows, Mac OS, etc.) debemos programar con nuevos paradigmas impuestos por el padre de la web, Sir Tim Berners Lee. ¡Sirva entonces esta entrada en el blog para ponernos al día con esta tarea pendiente!

En el siguiente vídeo denominado ""System Design 101" (por que las clases en las universidades comienzan su numeración desde cien) del canal "The Road Map" en YouTube describen pero que muy bien el mundo de la programación de aplicaciones web.

Ya las aplicaciones que conectaban en una red de área local un grupo de ordenadores a un servidor central donde se almacenaban bases de datos y ficheros poco a poco han quedado cortos (el concepto) con la potencia de las nuevas máquinas y la llegada de nuevos dispositivos como los teléfonos móviles con sistemas operativos como Android, Ubuntu y Mac.

El vídeo está en una resolución de 854 por 480 píxeles, ocupa 10 megabytes audio original en inglés, tiene subtítulos incrustados en castellano y en inglés y pueden descargar haciendo clic secundario para que aparezca el menú emergente y seleccionar "descargar".

"System Design 101" por The Road Map, audio en idioma inglés
(descargar y abrir en reproductor favorito para subtítulos en castellano e inglés)

Hacemos un breve repaso, minuto a minuto:

- Minuto 0:15: para los sistemas web necesitaremos, obviamente, los navegadores web, al momento de escribir estas líneas, en orden de velocidad de ejecución y consumo de recursos:
    - Navegador web Opera: con bloqueador de publicidad integrado, una interfaz agradable y lo mejor de todo ¡gran administrador de memoria y recursos de procesador central! llega este software desde el frío norte de Noruega... En un viejo ordenador de 32 bits doble núcleo con un gigabyte de memoria RAM es capaz de correr de manera decente. Redes sociales: Twitter @Opera, YouTube @Opera.
    - Navegador web Chromium: y su versión comercial Chrome tienen al poderoso

Google y su red de ordenadores a nivel mundial para tener acceso al caché de páginas web, así que parte de la carga de ejecución es delegada de nuestro computador. Por razones de trabajo con Google Drive hemos tenido que usarlo y curiosamente acá es donde viene la lentitud con YouTube y todos los demás sitios relacionados con Google. Sin embargo funciona y debemos instalar aparte AdBlock Plus para evitar la publicidad; ¡pongan atención porque tienen un proyecto de sistema operativo! Twitter @ChromiumDev.

    - Mozilla Firefox: el veterano navegador quedó para programadores serios y con potentes ordenadores. Siempre fiel a las nuevas normas de navegación de la World Wide Web Consortium (W3C), dicta cátedra en la materia pero exige hardware moderno y del bueno. Twitter @Firefox, YouTube @FirefoxChannel.

- Minuto 0:28: DNS (*Domain Name System*), hemos publicado acerca de cómo es el funcionamiento para poder "resolver" el nombre de un dominio en dirección IP.
- Minuto 0:58: una vez el navegador web contacta nuestro servidor web comienza la transferencia de datos (texto, imágenes, vídeo, etcétera).
- Minuto 1:17: CDN (*Content Delivery Network*) la forma como un servidor copia nuestros recursos y los aloja en sitios físicamente cercanos a nuestros visitantes.
- Minuto 2:37: *Horizontal Scaling* versus *Vertical Scaling*, las diferencias entre ampliar las capacidades de un servidor o, mejor aún, ampliar a varios servidores con un mecanismo en el siguiente punto.
- Minuto 3:35: *Load Balancers*, o balanceadores de carga, a medida que nos visitan serán redirigidos cada vez a un servidor distinto y al llegar al último se comenzará de nuevo por el primero.
- Minuto 3:50: *Application Architectures* (*Microservices*, SOA) importantes en la seguridad, para autentificar usuarios, se comunican por medio de APIS y pueden, por ejemplo, mandar correos o mensajes de texto con códigos de verificación de manera independiente de la aplicación principal.
- Minuto 4:23: *Storage* (*Database*, *Caching*, *Cloud Storage*) en el área de almacenamiento bien podemos usar a las tradicionales bases de datos pero con al añadido de que podemos almacenar nuestras consultas en algo como Redis y tener un *caché* a la mano para realizar cálculos. Ni hablar del almacenamiento en Internet tal como hace DigitalOcean con MySQL.
- Minuto 5:41: *Logging* porque debemos de tener un almacenamiento de los registros generados por nuestra aplicación, guardarlos como por ejemplo con a Pandora FMS con ElasticSearch.
- Minuto 5:57: *Messaging/Queues* para almacenar las tareas pendientes por hacer en una piscina que pueda ser atendido por varios servidores (por ejemplo un servidor de correo que revise la cola de mensajes y si encuentra una orden de enviar correo-e a alguna persona, pues hacerlo).
- Minuto 6:29: *Search Engines*, señoras y señores, motores de búsqueda como DuckDuckGo y Google, esas maquinarias no se van a meter en nuestras bases de datos de manera directa. Por ello debemos tener nuestro propio método para buscar, en el vídeo

nombran a ElasticSearch pero yo agrego también a Redis.

¿Qué es un DNS y cómo funciona?

https://www.youtube.com/watch?v=Wj0od2ag5sk

Registros DNS

https://www.youtube.com/watch?v=7lxgpKh_fRY

¿Cómo funcionan los balanceadores de carga?

https://www.youtube.com/watch?v=galcDRNd5Ow

¿Cómo funcionan las Redes de Entrega de Contenido?

https://www.youtube.com/watch?v=6DXEPcXKQNY

¿Cómo funciona el caché de las cabeceras HTTP?

https://www.youtube.com/watch?v=HiBDZgTNpXY