Desarrollo WEB en Entorno Servidor:

Explorando Modelos de Ejecución en Servidores Web

Alumno: Fara Santeyana, María Guillermina.
NIA: 13298631.
Curso: 2do DAW.
Instituto: I.E.S La Sènia.
Año: 2025-2026.

Investiga los siguientes modelos de ejecución (**basado en procesos, basado en hilos, dirigido por eventos o en kernel**) utilizados por servidores web. Para cada uno, completa una tabla con:

- Definición del modelo
- Ejemplo de servidor que lo utiliza (Apache, Nginx, Node.js, etc.)
- Ventajas e inconvenientes
- Casos de uso recomendados

Usa una analogía para explicar cada modelo (por ejemplo, restaurante con camareros, etc.). Redacta una breve conclusión (máx. 200 palabras) sobre qué modelo consideras más eficiente para una aplicación web con alta concurrencia y por qué.

RA1.a, RA1.c, RA1.e, RA1.g

1. Como analogía para explicar cada modelo tomando el funcionamiento de los Contact-centers
    a. Modelo basado en procesos: Este modelo es el más parecido (con algunas diferencias) a la modalidad utilizada en la realidad, donde la empresa contrata a un representante por cada llamada, con su propio cubículo, su ordenador, CRM y sistema de llamada. Cada llamada que entra es gestionada por el asesor, si el trabajador falla no afecta al resto de operadores ni clientes.
    b. Modelo basado en hilos: En este modelo los representantes utilizan los mismos recursos informáticos y atienden a múltiples clientes en simultáneo, cada uno realiza una acción. Es más veloz, pero deben coordinarse muy bien para no interferir en las llamadas y procesos.
    c. Modelo basado en eventos: Esto ocurre en los sistemas de atención al cliente vía whatsapp/instagram/escrito. El representante, atiende varias consultas de distintos clientes en simultáneo, dejándolo en espera mientras atiende otras consultas o gestiona las soluciones, cuando está lista la respuesta ésta es devuelta al cliente.
    d. Modelo basado en Kernel: En este caso por encima de estos asesores se encuentra un sistema que organiza que dirige el tráfico, qué asesor atiende cada llamada según su "skill", cual espera, cuales son prioritarias (clientes premium) y se les destina más recursos. Así los clientes más importantes son atendidos con mayor rapidez que los clientes de menor importancia.

2. Considero que para una aplicación web con alta concurrencia el modelo basado en eventos es el óptimo ya que no es necesario crea un proceso o hilo para cada actividad, el proceso principal se queda a la espera del evento lanzado por el cliente o por ejemplo de la base de datos. Cada señal se procesa de manera independiente, asíncrona, que incluso en inactividad es posible reactivarlos. Esto nos deja una arquitectura altamente escalable adaptándose a niveles de concurrencia y disponibilidad altos. Sin inicialización de múltiples hilos ni espera a la finalización de procesos.

	Definición del modelo	Servidor que lo utiliza	Ventajas/inconvenientes	Usos recomendados
Basado en procesos	Las actividades y peticiones realizadas como procesos (separados pero intercomunicados) se consiguen de forma más eficiente.	Apache.	*Minimiza errores y fallos. *Mejora tiempos. *Estructuras independientes <u>Desventajas:</u> Podría ser más costoso	Actividades que busquen estar separadas pero interrelacionadas, mejorando la seguridad.
Basado en hilos	Las actividades se ejecutan como múltiples hilos dentro de un proceso, paralelizando la ejecución. Basado en sockets.	Servidor http en Java.	<u>Ventajas:</u> *Mejor distribución de tareas, mejorando el rendimiento. *Gestión rápida. *Mejor control de bloqueos. <u>Desventajas:</u> *Sobrecarga en aplicaciones complejas. *Muy complejo para aplicaciones sencillas. *Su correcta programación puede ser compleja.	Recomendado para aplicaciones de carga media por su complejidad.
Dirigido por Eventos	Las estructuras y su ejecución están determinados por eventos que los define el programador o el usuario. Se maneja a tiempo real y permite comunicación asíncrona.	Ngnix, node.js.	<u>Ventajas:</u> *Mayor eficiencia *Menor curva de aprendizaje *Mayor interactividad *Flexibilidad, adaptable a distintas aplicaciones. * Escalabilidad, asincronía. <u>Desventajas:</u> *Complejidad en su mantenimiento.	Aplicaciones que necesitan comunicar cambios y soluciones en tiempo real en los sistemas entre servicios desacoplados

			*Portabilidad limitada, depende de las plataformas. *Flujo que escapa al control del programador. *Pueden surgir bloqueos	
Kernel	Es el núcleo de sistema operativo, gestiona el acceso a memoria, procesador. Es la base de la interacción entre el hardware y software. Controla ejecución de programas, seguridad. 3 tipos de Kernel, monolítico, microkernel, híbrido.	Linux, Windows NT, macOS	Ventajas: *Mayor rendimiento. *Mayor eficiencia Desventajas: *Mantenimiento complejo *Diseño complejo, aumentando el tiempo de desarrollo	Sistemas complejos como S.O. En servidores robustos con múltiples conexiones.

Bibliografía

https://iveconsultores.com/enfoque-basado-en-procesos/

https://www.vadavo.com/blog/que-es-nginx-como-funciona/

https://www.arcesio.net/practicas/WebServer1.htm

https://webhosting.de/es/nodejs-javascript-entorno-de-ejecucion-servidor/

https://es.wikipedia.org/wiki/Programaci%C3%B3n_dirigida_por_eventos

https://keepcoding.io/blog/que-es-la-programacion-orientada-a-eventos/#Ventajas_de_la_programacion_orientada_a_eventos

https://www.redhat.com/es/topics/integration/what-is-event-driven-architecture

https://www.ceac.es/blog/que-es-la-programacion-multihilo-y-que-ventajas-tiene

https://sisoperativoluis.wordpress.com/2016/09/24/concepto-de-hilo-de-ejecucion/

https://www.ibm.com/es-es/topics/event-driven-architecture

www.lenovo.com/es/es/glossary/thread/

https://keepcoding.io/blog/que-es-el-kernel/

https://datascientest.com/es/kernel-todo-sobre

https://www.ionos.es/digitalguide/servidores/know-how/que-es-el-kernel/

https://dev.to/cristobal_g/event-driven-architecture-eda-2m39