PREPARACIÓN PRUEBA2

Hilos:

- Comparten el mismo recurso (Como memoria)
- Existen dos tipos:
  - Flujo Único: utiliza un único hilo para controlar la ejecución
  - Flujo múltiple: utilizan varios para realizar su ejecución
- Multi-hilo: contiene un hilo principal que a su vez contiene otros hilos en ejecución
- Un hilo presenta una secuencia única, por lo que un programa puede tener varias secuencias
- Multi-hilo: permite acceder a recursos libres de CPU mientras se realizan otras tareas

Sincronización de relojes:

- Relojes Lógicos:
  - Lamport: "la sincronización no debe ser absoluta, si dos procesos no interactúan entre si, no requiere sincronización"
  - Distorsión de reloj: diferencia entre los valores de los relojes en tiempo de forma local
  - Es importante el orden de ocurrencia de eventos, no la hora exacta ("al contar 3 saltamos")
- Relojes físicos:
  - Se utiliza el tiempo atómico internacional y de coordenadas universal (TAI y UTC)
  - Es posible sincronizarlos por onda de radio cortas o un satélite

- Algoritmo de Lamport:
  - Relación temporal, donde A es el envío y B la recepción = “a→b”
  - se entiende que a<b (en tiempo) por lo tanto C(a)→C(b) C creciente
  - si son del mismo o distinto proceso y se llega a a→b, es verdadero
  - es concurrente si a→b = F y b→a = F
  - no puede pasar que C(a) distinto a C(b)

- Algoritmo de Christian:
  - Existe un servidor quien administra el tiempo al usuario (es como las weas)
  - Periódicamente el usuario solicita el tiempo del servidor, responde en t (tiempo)
  - el usuario mantiene su tiempo en t(propio)+t(transmisión)
  - el problema radica en que t(propio) podría ser mayor al de transmisión
  - otro problema es en el tiempo de propagación t(transmisión) el cual varía por red
  - la propagación puede estimarse como (tiempo de envío - tiempo de solicitud)/2
  - si se conoce el tiempo en que el servidor tarda en manejar la interrupción (ti) se puede mejorar la estimación como (tiempo de envío - tiempo de solicitud - ti)/2
  - Christian sugiere realizar varias mediciones para mejorar la precisión (un estilo de promedio) donde (tiempo de envío - tiempo de solicitud)

- Algoritmo de Berkeley:
  - Contiene un reloj maestro y varios relojes esclavos en otras máquinas

- El reloj maestro consulta los valores a las máquinas esclavas y realiza un promedio entre ellas para sincronizarse
- El reloj maestro envía los valores de desfase de los esclavos para que se ajusten a la nueva hora (este puede ser + minutos o -minutos según sea necesario)

Exclusión Mutua:

- Existen situaciones donde hay recursos compartidos que no puede utilizarse por más de un proceso al mismo tiempo
- En los sistemas donde se comparte memoria puede usarse semáforos, en estos sistemas ya no se comparte memoria física por lo que se debe utilizar algoritmos de exclusión mutua
- Se deben cumplir requisitos:
  - **Inanición:** cuida cuando un proceso desea entrar en la región crítica, dándole un tiempo para no producir bloqueos
  - **Seguridad**: garantiza que se ejecute un solo proceso en la región crítica
  - **Orden:** se debe seguir un orden de lamport al entrar a la región crítica "sucedió antes"
- Para garantizar que ningún proceso entre en la región crítica mientras esté en uso:
  - **Algoritmo Centralizado:**
    - simula un solo procesador para realizar la exclusión
    - nombra un proceso como coordinador
    - el coordinador da token a los procesos que necesiten entrar a la región crítica
    - al salir entra el siguiente con token
    - PROBLEMAS:

      → si el coordinador falla GG

      → si no posee token GG

→ un solo coordinador puede producir cuello de botella GG

- **Algoritmo distribuido (Ricart y Agrawala):**
  - Los procesos construyen un mensaje con:

    → Nombre, número de proceso y hora actual
  - Estos mensajes son enviados a todos los procesos como método de confianza
  - El proceso al recibir un mensaje puede ocurrir lo siguiente:

    → si el receptor no se encuentra en región crítica y no quiere entrar = OK

    → si el receptor se encuentra en región crítica = no responde

    → si el receptor no se encuentra en región crítica y quiere entrar:

    - compara su etiqueta con la del enviado en tiempo
    - si es menor gana y entra, lo mismo al contrario
  - PROBLEMAS:

    → requiere muchos mensajes al servidor GG

    → no existen tolerancias a fallos GG?
- **Algoritmo de anillo de token:**
  - Los procesos se conectan en forma de anillo
  - cada proceso se le asigna una posición
  - cada nodo conoce la dirección del vecino
  - al iniciar, el proceso 1 recibe un token el cual va circulando por el anillo
  - si el proceso k no necesita entrar a la región crítica, circula el token con un mensaje de k + 1
  - si el proceso necesita entrar a la región crítica, mantiene el token y entra
  - al salir pasa el token al siguiente proceso del anillo
  - PROBLEMAS:

→ de perder el mensaje con token no puede continuar el proceso GG

→ si un proceso falla en la región crítica (se lleva el token) GG

Replicación:

- Significa mantener una copia de información en diferentes pc´s
- Altamente usado ya que proporciona:
  - mejor rendimiento:
    - la replicación se implementa a través de cachés en clientes o servidores
    - es importante mantener copia de resultados para evitar copias idénticas
    - se evita el tiempo de latencia del cálculo del resultado y consultas a otros sv
    - el uso contínuo genera un costo por intercambio y actualización por uso de protocolos
  - alta disponibilidad:
    - la proporción de tiempo que un servicio está accesible con tiempos de respuesta razonable que debe ser cercana al 100%
    - la pérdida de disponibilidad puede ser debido a:
      - fallas de sv
      - desconexiones
  - tolerancia a fallas:
    - alta disponibilidad no implica corrección (puede existir datos no actualizados u organizados)
    - ante una caída el servicio sigue funcionando
- Ejemplos:
  - almacenamiento en caché de servidores web y servidores proxy web

    - servicio de nombres DNS
    - centro de datos de buscadores web
- Requisitos:
    - Transparencia: cliente inconsciente de la existencia de múltiples copias del recurso
    - Consistencia: las operaciones deben dar resultados según lo especificado
- Tipos de replicación:
    - **Pasiva:**
        - Existe un gestor primario y varios esclavos
        - El forntal se comunica con el primario y envía réplicas a los esclavos
        - Si el primario falla, un esclavo lo sustituye
        - Fases:
            - petición: frontal envía petición al gestor primario
            - coordinación: gestor ejecuta peticiones con identación fifo
            - ejecución: almacena las respuestas
            - acuerdo: gestor envía la actualización a los respaldos
            - respuesta: gestor envía respuestas al frontal
        - el proceso es propenso a cuellos de botella en el gestor primario
    - **Activa:**
        - Todos los gestores tienen el mismo rol
        - El frontal difunde la info a todos por igual
        - Los frontales procesan la petición de manera idéntica pero independiente
        - Fases:
            - petición: frontal multidifunde a los gestores mediante multidifusión fiable (no se envía otra petición hasta que se reciba la respuesta a la petición actual)

        - coordinación: entrega la petición a todos los gestores según ordenación total
        - ejecución: gestor ejecuta la petición
        - acuerdo: no es necesaria por la multidifusión
        - respuesta: cada gestor envía su respuesta al frontal
    - las fallas se toleran por la multidifusión fiable y ordenada