Sistemas operativos tolerantes a fallas y de tiempo real

Un sistema de cómputo falla cuando deja de cumplir con su especificación

Una falla es un desperfecto, que puede ser causado por un error de diseño, de fabricación, de programación o un daño o deterioro físico.

Tipos de fallas

Transitorias	Ocurren una vez y desaparecen. (interferencias electromagnéticas, ruido, ...)
Intermitentes	Aparecen y desaparecen periódicamente. Difíciles de detectar (falso contacto, calentamiento, ...)
Permanentes	Continúa existiendo hasta que el sistema se repara el componente o desperfecto (cto. quemado, ruptura de cables, ...)

.

El objetivo de diseño de un sistema tolerante a fallas (stf) es que continúe funcionando de manera correcta incluso en presencia de fallas (no sólo tratar de evitarlas).

Arquitectura de sistemas tolerantes a fallas

(Fallas en: componentes, procesadores, desempeño)

Fallas de componentes (Analizado comunmente por fabricantes):

p .- probabilidad de fallo (desperfecto) en un segundo dado

(1-p).- probabilidad de no fallo en un segundo dado

$p(1-p)^k$.- probabilidad de fallo después de k segundos

Tf .- tiempo promedio de fallo:

$$Tf = \sum_{k=1}^{\infty} kp(1-p)^{k-1} = \frac{1}{p}$$

usando

$$\sum_{k=1}^{\infty} \alpha^k = \alpha(1-\alpha)$$

$$\alpha(1-p), \frac{\partial}{\partial p}, (-p)$$

Ejm. Si $p = 10^{-6}$ por segundo, $Tf = 10^{6}$ segundos o aprox. 11.6 días

En un sistema de cómputo, las **fallas de un procesador** (o proceso) son muy importantes. Se pueden distinguir dos tipos:

Fallas silentes (fallas de detención).- El procesador se detiene y no responde a entradas subsecuentes. Puede anunciar que ya no funciona.

Fallas bizantinas.- El procesador sigue funcionando, pero entrega salidas incorrectas. Los errores de software frecuentemente son fallas bizantinas. (Más difíciles de detectar y corregir). [Imperio Bizantino (330-1433), los Balcanes y Turquía. Por las intrigas y conspiraciones, Lamport 1982]

Fallas de desempeño

Dependen del tipo de sistema:

Sistemas síncronos.- Si las comunicaciones se realizan con temporizadores finitos (acknowledge, timers, ...)

Sistemas asíncronos.- Si las comunicaciones no utilizan temporizadores (más difíciles de tratar).

Sistemas redundantes

La redundancia es el método general para implementar tolerancia a fallas.

Tres tipos de redundancia:

De información.- Se agrega información adicional para poder recuperar bits erróneos (Bits de paridad, Código Hamming, CRC, etc.)

Del tiempo .- Las acciones se realizan varias veces, hasta que sean exitosas. Muy útil con fallas transitorias o intermitentes (Transacciones atómicas)

Física.- Se agrega un equipo adicional para que el sistema tolere la falla de componentes y pueda seguir trabajando.

Dos técnicas: réplica activa (trabajo conjunto) y respaldo primario (trabajo individual)

Consideraciones:

1. Grado de réplica requerido
2. Desempeño en el caso promedio y peor caso sin fallas
3. Desempeño en el caso promedio y peor caso con fallas

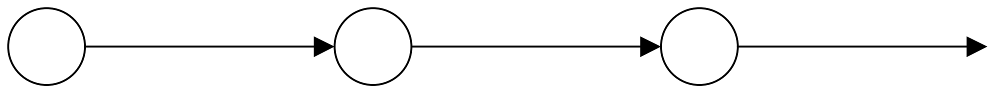
Sistemas replicados

Réplica activa

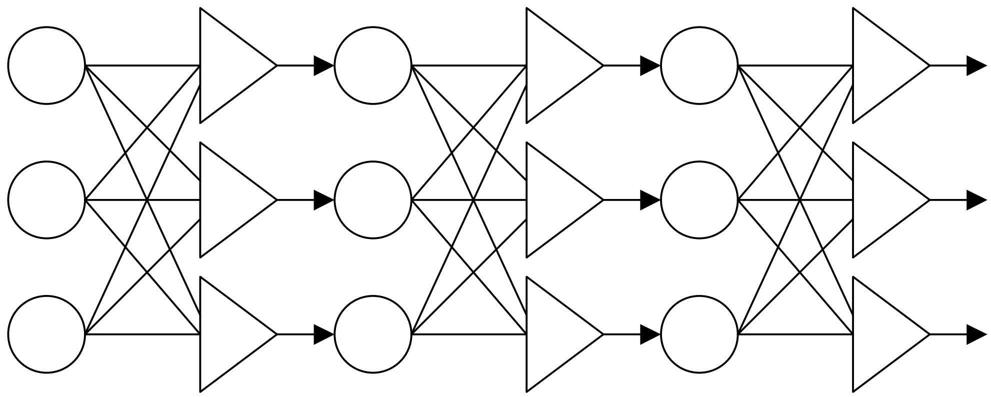
Existen varios dispositivos similares trabajando conjuntamente para obtener un resultado y ocultar las fallas. Usado en sistemas biológicos (ojos, pulmones, extremidades, ...)

Ejm. Redundancia modular triple (TMR)

Procesadores y votantes



Original



Respaldo activo

Estos sistemas se declaran como tolerantes a fallas de orden k, o tolerante a k fallas, si pueden sobrevivir a fallas de k componentes simultáneos.

En fallas silentes se requiere un mínimo de k+1 procesadores para que el sistema siga cumpliendo sus especificaciones y lograr una tolerancia a fallas de orden k.

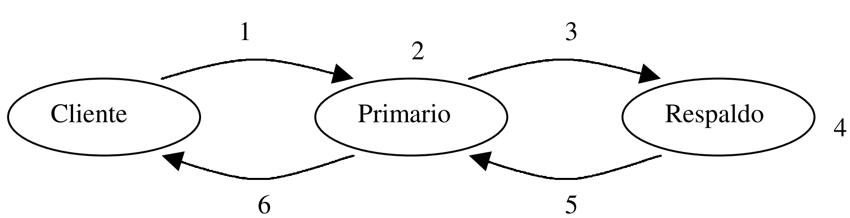
En fallas bizantinas se requiere un mínimo de 2k+1 procesadores para lograr una tolerancia de k fallas

La réplica activa no es un asunto trivial y es útil en sistemas de tiempo real.

Respaldo primario

Aquí, en cualquier instante un servidor es primario y realiza todo el trabajo. Si este falla, un secundario ocupa su lugar.

Es más sencilla y menos costosa que la réplica activa. Trabaja mal con fallas bizantinas. La recuperación de una falla puede requerir mucho tiempo. Requiere protocolos de acuerdo entre el primario y el respaldo.



Aspectos de diseño de sistemas tolerantes a fallas

- Importancia de la información procesada por el sistema (misión crítica, baja prioridad)
- Rapidez con la que el sistema debe recuperarse
- Costo del esquema de tolerancia a fallas
- Factibilidad de la implementación

Conceptos de procesamiento en tiempo real

Generalmente se considera que un programa funciona correctamente si entrega resultados que dependen sólo de la secuencia de instrucciones ejecutadas y no del momento en que se realicen.

En un sistema de tiempo real, el tiempo de respuesta también es un factor que determina el correcto funcionamiento. Los programas deben responder a una entrada o estímulo en un cierto tiempo límite para que la respuesta sea correcta. El momento en que se produce la respuesta es tan importante como aquello que se produce.

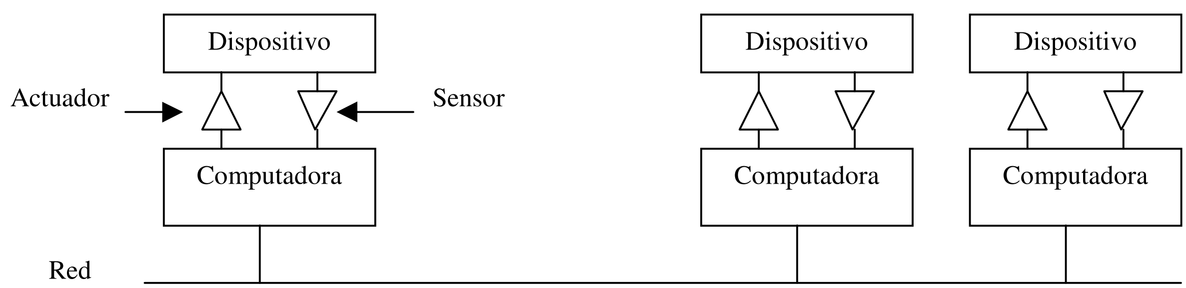
Muchos son los sistemas con estas características en el mundo real: control de tráfico aéreo, fábricas automatizadas, robots, equipos médicos, equipos de audio/video computarizados, etc.

Todos estos sistemas deben responder a estímulos de alguna clase, que pueden ser:

Periódicos.- Ocurren de manera regular data ΔT segundos (audio/video)
Aperiódicos.- Son recurrentes, pero no regulares (Llegada de aviones o vehículos)
Esporádicos.- Inesperados (Calentamiento de dispositivos, fallas de energía)

A pesar de que los eventos sean periódicos, pueden existir varios flujos de ellos que pueden superponerse en el tiempo (varios eventos de diferente tipo ocurren al mismo tiempo)

Debido a la complejidad de tener una sola máquina que controle un sistema grande de tiempo real, es útil pensar en sistemas distribuidos de tiempo real, donde las acciones se realicen de manera descentralizada.



Los sistemas de tiempo real se clasifican por lo general en dos tipos dependiendo de lo estricto de sus tiempos límite y las consecuencias de omitir uno de ellos:

1. Sist. de tiempo real suave.- No hay problema en rebasar un tiempo límite (un conmutador telefónico)
2. Sist. de tiempo real duro.- En estos sistemas, perder un tiempo límite es inaceptable pues tendría consecuencias graves (sist. médicos, tráfico aéreo)

Aspectos de diseño de sistemas en tiempo real

Sincronización del reloj.- Es importante sobre todo en sistemas distribuidos, donde existen varios relojes independientes.

Sistemas activados por eventos vs. activados por tiempo.- Los s.a.p.e. funcionan a través de interrupciones, pero pueden fallar cuando se presentan “lluvias de eventos” que sobrecargan al sistema En un s.a.p.t. se muestrean los sensores a intervalos de tiempo regulares y esto evita la sobrecarga en casos extremos, pero debe escogerse cuidadosamente el valor de este intervalo.

Predictibilidad.- El sistema debe tener un comportamiento predecible en todo momento. Deben conocerse la frecuencia esperada de generación de eventos y el tiempo de atención a cada uno.

Tolerancia a fallas.- En muchos sistemas de tiempo real la seguridad es crítica. Algunos pueden congelarse durante una falla seria para evitar peligros, y se dice que son seguros contra fallas.

Soporte del lenguaje.- Existen lenguajes especializados de tiempo real. (no soporten ciclos while generales ni recursión)

Comunicación en tiempo real.- La predictibilidad y el determinismo son cruciales.(Anillo y token, en vez de ethernet, comunicaciones con calidad de servicio)

Planificación en tiempo real.- Generalmente se base en hilos con funciones bien definidas y tiempos de ejecución acotados.

Los algoritmos de planificación pueden caracterizarse por los sig. parámetros:

Tiempo real duro vs. suave (gravedad de una falla)
Planificación con prioridad vs. sin prioridad (Las tareas pueden interrumpirse o no)
Dinámico vs. estático (Cambian los parámetros de decisión en el tiempo)
Centralizado vs. distribuido (La decisión es local o concensada)