**Tolerancia a fallos.**

1. ¿Cuáles son los componentes de un sistema de cómputo que pueden fallar?

Procesador, memoria, un dispositivo E7S, un cable.

1. ¿Qué es una falla y cuál puede ser causa?

Es un desperfecto causado por error del diseño error de fabricación un error de programación

1. ¿En qué consisten las fallas transitorios intermitentes permanentes?

Transitorio: si la operación se repite, la falla ya no representa intermitentes reaparece.

Permanente: continúa existiendo hasta reparar el componente con el desperfecto.

1. ¿Por qué la confiabilidad es particularmente importante en un sistema distribuido?

Debido a la gran cantidad de componentes, da aun la posibilidad que falle uno de ellos.

1. ¿Cuáles son las fallas existentes y cuáles bizantinas?

Salientes un procesador que falla solo se detiene y no responde sistemática un proceso que falla continuamente de ejecución proporcionando respuestas incorrectas.

1. ¿Cuáles son los sistemas síncronas?

Un sistema que tiene las propiedades de responder siempre a un mensaje dentro de un límite concuerdo.

1. En qué consisten la redundancia de la información física y del tiempo.

Se agregan algunos bits para recuperar los bits devueltos se realiza una acción y entonces se vuelve a realizar.

1. ¿Qué es la réplica activa?

Es una técnica muy conocida para proporcionar la tolerancia de fallos mediante la redundancia física.

1. ¿En qué consiste la redundancia modular simple?

Cada votante a un circuito tiene 3 entradas y una salida.

1. ¿Cuál es la idea esencial del método de respaldo primario?

En cualquier instante un servidor con el primario realiza todo el trabajo.

1. ¿Cuáles son las 2 ventajas que tiene el método de respaldo primario sobre la réplica activa?
2. Es sencilla mediante la generación manual en la práctica, se requieren maquinas puesto que en cualquier instante se necesita un respaldo.

**Asignación de procesadores.**Sistemas distribuidos.

Por definición un sistema distribuido consta de varios procesadores.

Estos se pueden organizar como estaciones de trabajo o pila de procesadores.

En ambos casos se necesita de un criterio (algoritmo) para decidir cuál proceso de la lista hay que ejecutar y en cual computadora.

Para el modelo de estación de trabajo la pregunta es cuándo ejecutar un proceso de manera local y cuando buscar una estación inactiva.

Se genera un nuevo trabajo cada vez que un proceso en ejecución se bifurca o crea un subproceso.

En la mayoría de los casos el proceso que se bifurca es el intérprete de comandos Shell que inicia un nuevo trabajo como respuestas a una petición de usuario.

En otros casos, el propio proceso de usuario crea uno o más hijos para tener un mejor desempeño con una ejecución en paralelo.

Las estrategias de asignación de procesadores se pueden dividir en dos categorías.

En la primera, llamada no migratoria, al crearse un proceso, se toma una decisión acerca de dónde colocarlo. Una vez colocado el proceso permanece ahí hasta que termina, no se puede mover no importa la sobrecarga que exista.

Con los algoritmos migratorios los procesos se pueden trasladar aunque haya iniciado su ejecución.

Los algoritmos migratorios permite un mejor balance de la carga de trabajo, son más complejos, y tienen un efecto fundamental en el diseño del sistema.

Los algoritmos para asignar procesos a los procesadores llevan implícito el intento por optimizar las tareas del sistema. De lo contrario haría la asignación en forma aleatoria o en orden numérico.

Una posible optimización seria maximizar el uso de cada CPU, es decir, el número de ciclos de cada CPU que se ejecutan con algún beneficio para las tareas del usuario.

El termino de maximización del uso del CPU se refiere a que hay que evitar a toda costa el tiempo inactivo de las CPU, es decir, dar a cada CPU algo que hacer.

Otro aspecto a optimizar, es minimizar es la tasa de respuesta.

La tasa de respuesta se define como la cantidad de tiempo necesaria para ejecutar un proceso en cierta computadora dividido entre el tiempo que tardaría en ejecutarse en un procesador de referencia.

En segundo aspecto es el diseño centralizado contra el distribuido.

La recolección de toda la información en un solo lugar permite tomar una mejor decisión, pero que es menos robusta y genera una carga pesada en la maquina central.

Son preferibles los algoritmos descentralizados pero se ha propuesto algunos centralizados por la carencia de soluciones descentralizadas adecuadas.

**Aspectos del diseño de algoritmos de asignación de procesadores.**

* Los algoritmos deterministas son adecuados cuando se sabe de antemano el comportamiento de los procesos. Por ejemplo, tener una lista de todos los procesos junto con sus necesidades de procesamiento, de archivos, de comunicaciones, etc.
* Con toda esa información disponible sería posible hacer una asignación perfecta.
* En ningún de los sistemas se tiene un conocimiento total pero se puede obtener una aproximación razonable: en la mayoría de los sistemas un día de trabajo es similar al trabajo del día anterior.
* De ese modo la carga de trabajo se puede predecir con cierta precisión, al menos de forma estadística.
* En los sistemas en los que la carga es totalmente impredecible las asignaciones de trabajo dependen de los que cada procesador este haciendo en ese momento.
* Las cargas de trabajo pueden variar de manera drástica en periodos menores de una hora o incluso un minuto.
* La asignación de procesadores en tales sistemas no se puede hacer de manera estadística por lo que se utilizan técnicas llamadas heurísticas.

**Actividad: sistemas operativos en tiempo real (pag 239)**

1. ¿A diferencia de la mayoría de los programas, cómo funcionan los sistemas de tiempo real?

Interactúan con el mundo exterior de una manera que implica al tiempo.

1. Indica por lo menos 4 ejemplos de sistemas de tiempo real.

Reproductor de discos compactos, computadoras incluidas en la televisor, grabadoras de video, subsistemas de automóviles.

1. ¿Cuáles son los estímulos? A) periódicos. B) aperiódicos C) esporádicos.
2. Modo que un estímulo ocurre de que recibe un cuadro nuevo cada 1/60 segundos.
3. Son recurrentes pero no regulares, como en la llegada de un avión al espacio aéreo de un controlador de tráfico aéreo.
4. Inesperados, como el sobrecalentamiento de un dispositivo.
5. ¿Cuáles son los dos tipos en lo que se clasifican los sistemas de tiempo real?

Sistema de tiempo real suave y duro.

1. ¿Qué significa el tiempo real suave?

Que no existe problema si se rebasa un tiempo límite. Por ejemplo, un conmutador telefónico bajo condiciones de sobrecarga podría perder o equivocar de ruta tales llamadas y seguir cumpliendo sus especificaciones.

1. ¿A que conduce un tiempo límite excedido en un sistema de tiempo real duro?

Que falla toda la actividad actual, pero que la consecuencia no es fatal.

1. Señala por los tres mitos acerca de los sistemas en tiempo real.

Sistemas de tiempo real tratan de la escritura de controladores de dispositivos en código ensamblador.

El cómputo de tiempo real es rápido.

Las computadoras rápidas harán que el sistema de tiempo real sea obsoleto.

1. ¿En los sistemas activados por eventos, ¿Quién detecta cuando ocurre un evento significativo en el mundo exterior?

Un sensor, lo que entonces provoca que el CPU conectado tenga una interrupción

1. ¿Cuál es el principal problema con los sistemas activados por eventos?

Pueden fallar bajo condiciones de carga pesada, es decir, cuando muchos eventos ocurran a la vez.

1. ¿En los sistemas activados por el tiempo que sucede cada vez que ocurre una interrupción del reloj?

Ocurre una interrupción del reloj AT milisegundos.

1. ¿Cuál es la diferencia entre los diseños activados por eventos y los activados por el tiempo?

Los diseños activados por eventos dan respuesta rápida con carga baja, pero tienen mayor costo y probabilidad de fallar con carga alta. Los diseños activados por tiempo tienen las propiedades opuestas y solo son adecuados en un ambiente relativa estático en donde se conozca mucho y de anteaño acerca del comportamiento del sistema.

Java sockets

Clases tiles en comunicaciones.

Socket

Es el objeto básico en toda comunicación a través de internet, bajo el protocolo TCP. Esta clase proporciona métodos para la entrada/salida a través de streams que hacen la lectura y escritura a través de sockets muy sencilla.

Serversocket

Es un objeto utilizando en las aplicaciones servidor para escuchar las peticiones que realicen los clientes conectados a este servidor. Esta objeto no realiza el servicio, sino que crea un objeto socket en función del cliente para realizar toda la comunicación a través de él.

Datagramsocket

La clase de sockets datagrama puede ser utilizada para implementar datagramas no fiables (sockets UDP) no ordenados. Aunque la comunicación por estos sockets es muy rápida porque no hay que perder tiempo estableciendo la conexión entre cliente y servidor.

Datagrampacket

Clase que representa un paquete datagrama conteniendo información de paquete, longitud de paquete, direcciones internet y números de puerto.

Multicastsocket

Clase utilizada para crear una versión multicast de las clases socket datagrama. Múltiples clientes/servidores pueden transmitir a un grupo mutlicaast (un grupo de direcciones IP compartiendo el mismo número de puerto).

Networkserver

Una clase creada para implementar métodos y variables utilizadas en la creación de un servidor TCP/IP.

NetworkClient

Una clase creada implementar métodos y variables utilizadas en la creación de un cliente TCP/IP

socketImpl

Es un interface que nos permite crearnos nuestro propio modelo de comunicación. Tendremos que implementar sus métodos cuando la usamos. Si vamos a desarrollar una aplicación con requerimientos especiales de comunicaciones, como pueden ser la implementación de un cortafuegos (TCP es un protocolo no seguro), o acceder a equipos especiales (como un lector de código de barras o un GPS diferencial) necesitaremos nuestra propia clase socket.

Página 272 uso de archivos

**Que representas las mediciones estáticas y como se realizan.**

Una toma instantánea del sistema en cierto momento y se realiza al examinar el disco. Y ver lo que hay en el

**Que mediciones se encuentran entre las mediciones estáticas**

Las mediciones estáticas se realizan al examinar el disco y ver lo que hay en el.

**Como se llevan a cabo las medidas dinámicas.**

Al modificar el sistema de archivos

**Que proporcionan los datos obtenidos en las medidas dinámicas.**

Proporciona información con respecto a la frecuencia relativa de varias operaciones

**Por qué un problema en las mediciones es saber que tan típica es la población observada.**

Nadie lo sabe, hasta que estos sistemas se instrumenten y se realicen las mediciones en ellos.

**Pagina 274 estructura del sistema**

**Porque en algunos sistemas no hay distinción entre cliente y servidor.**

Todas las maquinas ejecutan el mismo software básico, de modo que una máquina que desee dar servicio de archivos al público general es libre de hacerlo

**En que consideran las dos formas de estructurar los servicios de archivos y directorios.**

Colocar a un servidor junto y otra es separarlos

**Que es esencial para que funcione el ocultamiento de los nombres**

Es esencial que cuando se utilice de manera inadvertida un nombre binario absoluto, se le informe de esto al cliente de alguna manera.

**A que se refiere que los servidores no deben contener los estados (ser sin estado) de los clientes.**

Cuando un cliente envía una solicitud a un servidor, este la lleva a cabo, evia la respuesta y elimina de sus tablas internas toda la información relativa

**Cuáles son las ventajas de los servidores sin estado y la de los servidores con estado.**

Ventajas de los servidores sin estado: no necesita llamadas open/close, no se desperdicia el espacio del servidor en tablas, no existe límite para el número de archivos abiertos

Ventajas servidores con estado: mejor desempeño, es posible la lectura adelantada, es más fácil la idempotencia

Los sockets son un sistema de comunicación entre procesos de diferentes máquinas de una red.

Más exactamente, un socket es un punto de comunicación por el cual un proceso puede emitir o recibir información.

Fueron popularizados por berckley software distribution, de la universidad norteamericana de berkley.

Los sockets utilizan el protocolo TCP (Transfer Control Protocol) y UDP (User Datagram Protocol).

Utilizan una serie de primitivas (funciones básicas) para establecer el punto de comunicación, para conectarse a una maquina remota en un determinado puerto que esté disponible, para escuchar en él, para leer escribir y en él, y para desconectarse.

Con todas las primitivas se puede crear un sistema de dialogo muy completo.

Funcionamiento genérico.

Un servidor se ejecuta sobre una computadora específica y tiene un socket que responde en un puerto específico.

El servidor espera, escuchando a través del socket a que un cliente haga una petición.

El cliente conoce el nombre de host de la maquina en la cual el servidor se encuentra ejecutando y el número de puerto en el cual el servidor está conectado.

Para realizar una petición de conexión, el cliente intenta encontrar al servidor en la maquinas servidora en el puerto especificado.

Java sockets

El paquete java.net de la plataforma java proporciona una clase socket, la cual implementa una de las partes de la comunicación bidireccional entre un programa java y otro programa en la red.

La clase socket se sitúa en la parte más alta de una implementación dependiente de la plataforma, ocultando los detalles de cualquier sistema particular al programa java.

Usando la clase java.net.socket los programas java pueden comunicarse a través de la red de una forma totalmente independiente de la plataforma.

Java.net incluye la clase serversocket, la cual implementa un socket el cual los servidores pueden utilizar para escuchar y aceptar peticiones de conexión de clientes.

Por otra parte, si intentamos conectar a través de la web, la clase URL y clases relacionadas (URL Connection, URLEncoder) son probablemente más apropiadas que las clases de sockets.

Las clases URL no son más que una conexión a un nivel más alto a la web y utilizan como parte de su implementación interna los sockets.

**Servitor**

Serversocket(port#timeout)

Accept()

Outputstream

Inputstream

Close()

**Cliente**

Socket(host port#)

Outputstream

Inputstream

Close()

El modelo de sockets mas simple es:

El servidor establece un puerto y espera durante un cierto tiempo (timeout segundos), a que el cliente establezca la conexión.

Cuando el cliente soliciten una conexión, el servidor abrirá la conexión socket con el método accept().

El cliente establece una conexión con la maquina host a través del puerto que se designe en puerto#

El cliente y el servidor se comunican con manejadores InputStream y OutStream.

Apertura de sockets

Si estamos programando un cliente, el socket se abre de la forma:

Socket miCliente;

miCliente = new Socket(“maquina”, numeroPuerto);

Donde maquina es el nombre de la maquina en donde estamos intentando abrir la conexión y numeroPuerto es el puerto del servidor que está corriendo sobre el cual nos queremos conectar.

Cuando se selecciona un numero de puerto, se debe tener en cuenta que los puertos en el rango 0-1023 estan reservados para usuarios con muchos privilegios (superusuarios o root)

Estos puertos son los que utilizan los servicios estándar del sistema como email, ftp o http.

Para las aplicaciones que se desarrollen, asegurarse de seleccionar un puerto por encima del 1023.

Ejemplo de la declaración de apertura de un socket cliente:

Socket miCliente;

Try{

miCliente=new Socket(“maquina”, numeroPuerto);}

catch(IOException e){

System.out.println€;

}

278

1. En un Sistema cliente servidor ¿cuáles son los cuatro lugares donde se pueden almacenar los archivos?

El disco del servidor, la memoria principal del servidor, disco del cliente, memoria principal del cliente.

1. ¿Por qué el disco del servidor es el lugar más directo para almacenar archivos?

Porque ahí existe mucho espacio y los archivos serian accesibles a todos los clientes

1. ¿Que debe hacerse antes de que un cliente pueda leer un archivo?

El servidor estable un puerto y espera durante un cierto tiempo a que el cliente establezca la conexión. Cuando el cliente solicite una conexión, el servidor abrirá la conexión socket con el método accept.

1. ¿Por qué se puede lograr mejor desempeño si se ocultan (conservan) los archivos de más reciente uso en la memoria principal del servidor?

El cliente establece una conexión con la maquina host a través del puerto que se designe el puerto#

El cliente y el servidor se comunican con manejadores InputStream y OutputStream.

1. ¿Cuáles son los dos problemas que debe resolver el algoritmo para determinar los archivos que deben permanecer en cache?

Es el tamaño de la unidad que administra el cache.

1. ¿Por qué el mantenimiento de un cache (en la memoria principal de un servidor) es fácil de lograr y transparente para los clientes?

Es el algoritmo debe decidir que hacer si se utiliza toda la capacidad del cache y has eliminar a alguien.

1. ¿Cuáles es la única forma de deshacerse del acceso a la red?

Es hacer el ocultamiento en el lado del cliente que es donde aparecen los problemas.

1. Si los diseñadores colocan el cache en la memoria principal, ¿Cuáles son las tres opciones que existen para una posición precisa?.

Consiste en ocultar los archivos en forma directa dentro del propio espacio de direcciones de un proceso usuario.

Es el nucleo

Es un proceso administrativo del cache.