Tema 4 Sistemas Distribuidos: tiempo y estados globales

La noción del tiempo es una problemática en un sistema distribuido, debido a que no existe un reloj global al sistema. Se pueden realizar algunas aproximaciones, con la sincronización de los relojes de los computadores, con relojes lógicos o con algoritmos para conocer los estados globales de un SD cuando ejecuta.

Cada computador de la red tiene su propio reloj interno. Dicho reloj es usado por los procesos locales, de modo que procesos en computadores distintos pueden tener marcas de tiempo distintas. Es importante saber que aunque todos los relojes del sistema distribuido se sincronicen, estos variarán significativamente con el tiempo.

**Tasa de deriva de reloj:** diferencia por unidad de tiempo en que el reloj del computador difiere del reloj perfecto.

Los relojes de cuarzo ordinarios derivan 1 seg. Cada 11-12 días. Los relojes de alta precisión derivan10^-7 segs.

Un sistema distribuido está distribuido como una colección de procesos. Cada uno de ellos tiene un estado, el cual está formado por todas sus variables u objetos y que puede cambiar en ejecución. Se comunican a través de la red mediante mensajes, pudiendo realizar las acciones de envío, recibimiento y cambio de estado.

**Evento:** ocurrencia de una acción que lleva a cabo un proceso al ejecutar. Los eventos en un proceso pueden ordenarse en relación a los procesos siguientes y a los que le preceden.

**Historia del proceso P:** es una serie de sus eventos ordenador por i. Historia(Pi)=hi=<evento0i,evento1i…>.

# RELOJES

Para establecer las marcas temporales se usa el reloj del computador. En un instante t el SO lee el valor del reloj hardware del computador. En general su cálculo no es completamente exacto, pero si funciona lo suficientemente bien puede ser usado como marcador de los eventos de P.

**Resolución del reloj < intervalo de tiempo entre dos posibles eventos consecutivos. La resolución de un reloj es el periodo entre dos actualizaciones consecutivas del reloj.**

Los relojes no siempre están en perfecto acuerdo. Por ello tenemos los siguientes términos:

**Sesgo:** diferencia de tiempo entre dos relojes en un instante determinado.

**Tasa de deriva:** explicada anteriormente.

**¿Qué puede influir en la tasa de deriva del reloj?**

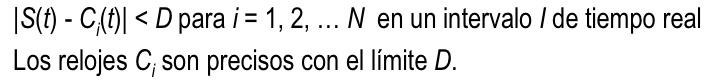
El tiempo atmosférico debido a fenómenos naturales como tormentas o cambios de presión, así como cambios de magnetismo (por ejemplo un caso en el que la gravedad de la tierra se viera afectada).

## TIEMPO UNIVERSAL COORDINADO (UTC)

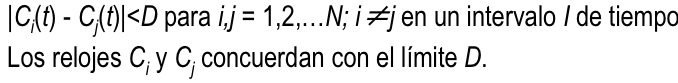
Los relojes se pueden sincronizar con fuentes externas muy precisas: tiempo atómico internacional basado en un reloj con deriva de un segundo cada 300.000 años.

UTC es un estándar internacional de establecimiento y mantenimiento del tiempo transcurrido. Se basa en el tiempo atómico y ocasionalmente ajustado al tiempo astronómico. La señal se difunde mediante estaciones de radio por tierra y mediante satélites. Las computadoras pueden sincronizar sus relojes mediante receptores adecuados.

**Sincronización externa:** un reloj C se sincroniza con una fuente UTC exacta S.



**Sincronización interna:** cualquier par de computadores están sincronizados si sus relojes cumplen:

****

Los relojes sincronizados internamente no necesariamente lo están externamente, puesto que pueden derivar juntos.

**Corrección del reloj:** se dice que un reloj hardware (H) es correcto si su límite de deriva es conocido (tiempo deriva > 0).

Por tanto el error en la medida de dos eventos está limitado para impedir que se produzcan saltos traumáticos en el valor leído. Se puede relajar la condición de error, por ejemplo en make de Unix, alcanzando la monotonicidad en un reloj hardware que funciona rápido, ajustando alfa y beta en C(t) = alfa\*H(t) + beta.

**Un reloj defectuoso:** es aquel que no cumple ninguna de las condiciones de corrección.

**Un fallo de ruptura:** es cuando el reloj se para o no emite tics.

**Un fallo arbitrario:** cuando se produce cualquier otro fallo.

**IMPORTANTE: un reloj no tiene por qué ser preciso para ser correcto.**

## Se dice que un SD es síncrono si están definidos los límites siguientes:

Tiempo máximo y mínimo para ejecutar cada paso de un proceso.

Tiempo máximo y mínimo de recepción de un mensaje.

Los límites de deriva de cada reloj local donde se ejecuta cada proceso son conocidos.

**Primera aproximación(sincronización interna):**

* Un proceso p envía su tiempo local t al proceso p2 en un mensaje m.
* P2 podría poner su reloj a t + Ttransmisión del mensaje.
* Ttransferencia es desconocido, pero min <= Ttrans <= max.
* La incertidumbre u = max-min. Si se establece el reloj a t + (max - min)/2 entonces el sesgo <= u/2.

**Método de Cristian (para sincronizar relojes externamente):**

Un servidor de tiempo S recibe señales UTC.

* El proceso p solicita el tiempo en un mensaje m y recibe t en Mt de S.
* P establece su tiempo a t + Tround/2 (Tround es el tiempo de ida y vuelta).
* Precisión: más o menos será (Tround/2 - min). El min será el mínimo estimado de transmisión.
  + El momento más temprano en que S pone t en Mt es min después de que P enviara Mr.
  + El momento más tardío es min antes de que Mt llegue a p.
  + El tiempo de S cuando Mt llega está en el rango (t + min, t + Tround - min).

**Algoritmo de Berkeley (para sincronizar relojes internamente).**

* Un maestro consulta y recoge valores de reloj del resto de computadores, esclavos.
* El maestro utiliza los tiempos de ida y vuelta de los mensajes para estimar el valor de los relojes esclavos.
* Promedia los resultados incluyéndose y eliminando cualquier valor que no sea consistente.
* Envía la magnitud de ajuste de cada reloj, que puede ser positivo o negativo.

Si el maestro falla, se puede elegir a un nuevo maestro.

## Protocolo de tiempo de red (NTP)

Se trata de un servicio de tiempo para internet. Sincroniza a los clientes con UTC.

Servicio fiable, redundante, reconfigurables si alguno cae, escalables, con autenticación de las fuentes de tiempo.(se puede pensar como una forma de árbol).

* Los servidores primarios están conectados a fuentes UTC.
* Los servidores secundarios sincronizados a los primarios.
* [Subred de sincronización] y en el nivel más bajo de servidores están los PC.

### Sincronización de servidores en NTP.

La subred de sincronización se puede reconfigurar si se produce un fallo:

* un primario que pierde su conexión con UTC puede pasar a secundario.
* Un secundario que pierde a su primario puede seleccionar a otro primario.

Los modos de sincronización son:

* Multidifusión (multicast)
  + En LAN de alta velocidad. Un servidor reparte el tiempo al resto que establecen su tiempo asumiendo un retraso de transmisión.
* Llamada a procedimiento.
  + Similar a de Cristian. El servidor acepta peticiones. Precisión más alta.
* Simétrica.
  + Pares de servidores se intercambian mensajes conteniendo información de tiempo.
  + Usado en los casos en que se necesita muy alta precisión (primeros niveles).

### Intercambio de mensajes entre pares de servidores NTP.

Todos los modos usan UDP. Cada mensaje lleva marcas de tiempo de los eventos recientes (del evento anterior y del tiempo local de envío del mensaje actual).

El receptor anota el tiempo local cuando recibe T.

El protocolo UDP carece de seguridad, y puede haber retraso entre la llegada de un mensaje y el envío del siguiente, o pérdida de mensajes.

### Precisión de NTP.

Para cada par de mensajes entre servidores, NTP estima una compensación entre los dos relojes (deriva) y un retardo d(medida de precisión. Tiempo total de transmisión para los dos mensajes t y t’).

Los servidores NTP mantienen pares del tipo <O , d> estimando la fiabilidad de las variaciones y permitiendo cambiar el propio par.

## Lamport (1978)

No se sincronizan relojes, sino que se ordenan eventos según la relación de orden parcial “suceder antes”.

1. Si los eventos ocurren en Pi (i=1,2,…N) entonces ocurren en el orden observador por Pi, es decir i.
2. Cuando m es enviado entre dos procesos, el envío (m) ocurre antes que recepción(m).
3. La relación es transitiva.

### Lamport. Relojes lógicos.

Un reloj lógico es un contador software monótono creciente. No se debe confundir con un reloj físico.

Cada proceso tiene su reloj lógico que se utiliza para fijar las marcas temporales a los eventos.

## Relojes vectoriales.

Mattern y Fidge [1989 - 1991] los desarrollan para superar la deficiencia de los relojes lógicos de Lamport: L(e) < L(e’) no implica e -> e’.

Un reloj vectorial V en el proceso P es un array de N enteros, que cada proceso utiliza para establecer marcas de sus eventos locales.

# ESTADOS GLOBALES

Se examina si una propiedad particular de un SD es cierta cuando éste se ejecuta. Ejemplos:

* Compactación automática de memoria.
* Detección distribuida de bloqueos indefinidos.
* Detección de terminación distribuida.
* Depuración distribuida.

Ilustran la necesidad de observar el estado del SD globalmente.

**Compactación de memoria:** antes de eliminar información se deben analizar las referencias existentes y los canales de comunicación.

**Bloqueo indefinido:** el abrazo mortal clásico, pero entre procesos no ubicados en la misma máquina.

**Terminación:** dos procesos pueden en un instante ser pasivos, pero no ser susceptibles de ser eliminados. Por ejemplo, P1 a pesar de ser pasivo puede pasar a activo por un mensaje en tránsito.

**Corte consistente:** es un corte tal que si para cada evento que contiene, también contiene todos los sucesos que sucedieron antes de él.

**Estado global consistente:** un estado que corresponde con un corte consistente.

**Caracterización de la ejecución del SD:** transiciones entre estados globales.

**Linealización o ejecución consistente:** ordenación de los sucesos en una historia global que es consistente.

## ALGORITMO DE CHANDY Y LAMPORT

Es un algoritmo de instantánea para determinar estados globales de sistemas distribuidos. Registra un conjunto de estados de procesos y canales de los procesos del sistema de forma que el estado global sea consistente.

Se registra el estado de cada proceso localmente, no hay proceso recolector.

### Regla de recepción del marcador para el proceso P.

Cuando P recibe un mensaje marcador sobre el canal c:

Si (p no ha registrado todavía su estado):

* registra su estado de proceso ahora.
* Registra el estado de c como el conjunto vacío.
* Activa el registro de los mensajes que llegan sobre otros canales entrantes.

Sino

* p registra el estado de c como el conjnto de mensajes que ha recibido sobre c desde que guardó su estado.

### Regla de envío del marcador para el proceso P.

Después de que P haya registrado su estado para cada canal de salida c:

* p envía un mensaje marcador sobre c (antes que envíe otro mensaje sobre c).

En el algoritmo de Chandy y Lamport el estado registrado puede diferir de todos los estados globales por los que ha pasado realmente el sistema.

El algoritmo selecciona un corte de la historia de la ejecución y cuyo estado registrado es consistente.

El algoritmo termina si se cumplen las restricciones de conectividad total e inexistencia de fallo en la comunicación.

## Conclusiones.

Los algoritmos como Cristian o NTP sincronizan los relojes a pesar de sus derivas y el retardo de los mensajes.

La sincronización de relojes no siempre es suficiente para satisfacer los requisitos de ordenación de dos eventos arbitrarios que sucedan en dos computadores.

La relación “suceder antes” es un orden parcial sobre los eventos que reflejan un flujo de información entre ellos.

Los relojes de Lamport son contadores que se actualizan de acuerdo con la relación “suceder antes” entre eventos.

Los relojes vectoriales mejoran los de Lamport, ya que determinan si dos eventos están ordenados por la relación “suceder antes” o son concurrentes, comparando los vectores de marcas.