UNIVERSIDAD AUTÓNOMA METROPOLITANA
UNIDAD AZCAPOTZALCO

DIVISIÓN DE CIENCIAS BÁSICAS E INGENIERÍA

PROYECTO TERMINAL DE INGENIERÍA EN COMPUTACIÓN

“SINCRONIZACIÓN DE RELOJES FÍSICOS”

ALUMNO:
MANUEL DURÁN LÓPEZ
MATRÍCULA: 206300353

ASESOR:
M. En C. JOSÉ ALFREDO ESTRADA SOTO
Nº ECO.: 22003

TRIMESTRE 11-O

RESUMEN

El concepto de sincronización es aplicable para muchos aspectos de la vida cotidiana, como la entrada al trabajo, las citas con otras personas, las horas de comida, entre otras. Por supuesto que para el área de Ingeniería es sumamente importante ya que permite llevar a cabo muchos procesos de producción.

La sincronización permite que dos o más eventos puedan coincidir en un espacio de tiempo determinado, la finalidad de generar coincidencias en el tiempo ha tenido su origen en la necesidad de marcar instantes, tales como el inicio o fin de un proceso, una probable interrupción o la introducción de una nueva actividad. Por supuesto que las marcas de tiempo son la herramienta principal para poder definir medidas de tiempo.

La sincronización en los sistemas de cómputo es muy importante, ya que lo anterior mencionado permite tomar intervalos de tiempo que medirán la duración de uno o varios procesos en diferentes tipos de sistemas, lo que permitirá tener un acercamiento tangible al rendimiento de los procesos y posteriormente nace la posibilidad de hacer optimizaciones. Establecer una sincronización en un sistema distribuido es más complicado que hacerlo para un sistema centralizado, ya que en un sistema centralizado los procesos se ejecutan en un solo equipo, mientras que en un sistema distribuido los procesos se ejecutan en equipos diferentes y cada uno tiene su propio reloj autónomo. Existen varias maneras de sincronizar un sistema, dichas maneras se ven reflejadas en algoritmos, algunos utilizan una referencia de tiempo para obtener un tiempo en común.

En este trabajo se muestra cómo llegar a la sincronización de los relojes físicos de tres equipos de cómputo. El sistema se define como heterogéneo, ya que en él se encuentran dos PC´s y un equipo móvil (PDA). Es decir, las plataformas hardware, los sistemas operativos y los lenguajes de programación sobre los que se trabaja son completamente diferentes.

Se emplea el algoritmo de Berkeley para la sincronización de los relojes con una granularidad con base en los milisegundos.

ÍNDICE GENERAL

I. INTRODUCCIÓN 5

II. OBJETIVOS 6

III. DOCUMENTACIÓN FINAL 7

IV. SINCRONIZACIÓN 8

V. RELOJES 8

VI. SINCRONIZACIÓN DE RELOJES FÍSICOS 9

VII. ALGORITMOS DE SINCRONIZACIÓN 10

VIII. DESARROLLO DEL SISTEMA 12

IX. MANUAL DE INSTALACIÓN Y DE USUARIO 18

X. CONCLUSIONES 29

XI. BIBLIOGRAFÍA 30

INDICE DE FIGURAS

Figura 1. Algoritmo de Cristian *10*

Figura 2. Sistema de sincronización de relojes físicos *13*

Figura 3. Equipo 1 ..... *14*

Figura 4. Equipo 2 ..... *15*

Figura 5. Equipo 3 ..... *16*

Figura 6. Switch Kingston *16*

Figura 7. Router Belkin *16*

Figura 8. Cable UTP *18*

Figura 9. Cable UTP Cat 5e *19*

Figura 10. Red Inalámbrica *19*

Figura 11. Código de colores B *19*

Figura 11. Topología en Estrella *20*

Figura 12. Modelo Cliente Servidor (Sockets) *23*

Figura 13. Diagrama de solicitud de tiempo *25*

Figura 14. Conectividad de plataformas *25*

Figura 15. Comunicación C – Java *28*

I. INTRODUCCIÓN

La computación ha tenido la tendencia de mostrar la capacidad de hacer procesos y tareas con dispositivos cada vez más pequeños e innovadores, cada uno de los lanzamientos de dichos dispositivos implica el retroceso en cuanto a la complejidad de tareas que el dispositivo previo realizaba, pero un adelanto en cuanto a la portabilidad y diseño. Sin embargo, al pasar del tiempo estos dispositivos incrementan la capacidad para realizar tareas complejas y sofisticadas que se asemejan a las que puede hacer una computadora personal, lo que hace necesaria la interconexión entre los dispositivos y el intercambio de información en todos los niveles, es decir: texto, audio, video, entre otros, por lo que la variedad de computaciones, protocolos, lenguajes de programación y algoritmos es necesaria para cumplir con las expectativas de la transferencia de información a las velocidades y con la calidad que el mundo actual exige.

Para que la computación de los dispositivos que van desde una PC, que hoy en día se encuentra en muchos hogares, hasta dispositivos que se encuentran dentro de las nuevas tendencias como las computadoras portátiles, o las tablets, es necesario que dichos dispositivos sean capaces de ejecutar procesos que cuenten con una sincronía en el tiempo del sistema local. Por supuesto que para fines de interconexiones e intercambio de información entre varios dispositivos del mismo tipo o diferente es necesario establecer métodos por los cuales los procesos de cada uno de los equipos del sistema lleguen a una sincronía en el tiempo de la ocurrencia de los procesos que ejecutan. Una de las maneras más sencillas de tener armonía en el tiempo de los procesos es establecer un sistema centralizado en el que toda la información y procesos se hospeden en una máquina maestra o un servidor definido, y que los demás equipos únicamente sean clientes o terminales tontas que solicitan información a ese servidor; sin embargo, cada una de las sesiones que ejecutará procesos en un mismo servidor o maestro carecen de autonomía pues no cuentan con un registro ni una administración de procesos propia. Por otro lado, puede mencionarse el enfoque de un sistema centralizado en el cual cada nodo de la red hace que su tiempo dependa de un maestro (Computadora), en este caso los procesos de cada una de las computadoras que representan a cada uno de los nodos en la red, tienen autonomía en los registros y administración de sus procesos; sin embargo, para sincronizarse en el tiempo en que estos procesos se ejecutan, únicamente heredan la hora de la máquina maestro.

La sincronización de relojes físicos mediante el algoritmo de Berkeley es una metodología en la cual el proceso mismo de sincronización es dinámico, ya que no hay herencias directas de instantes ni horas en el tiempo, es decir, no hay un UTC en el sistema, en su lugar existe un coordinador que puede ser electo por diferentes métodos, una vez que está definido dicho coordinador este solicitará la hora de los demás equipos del sistema, los promediará y enviará la nueva hora a cada uno de los equipos para que la definan como la hora del sistema, entonces se podrá decir que el sistema está sincronizado.

La sincronización de relojes físicos va más allá de la teoría y los algoritmos, tiene tanta importancia que por medio de esta herramienta se puede mejorar la eficiencia del trabajo de toda una empresa u optimizar sus costos; esto es gracias a que en muchos de los casos impide la duplicación de información esto implica ahorro de tiempo y de espacio para archivar dicha información, en el aspecto energético también tiene un impacto porque si se logra tener una sincronización entre la maquinaria utilizada una empresa con el consumo general de energía eléctrica de una empresa así como la adecuación a las horas productivas, entonces para las horas no productivas se puede llegar a una

armonía en la cual hay ahorro o disminución considerable en el consumo de energía. La telefonía IP es otro claro ejemplo de un sector que necesita de la sincronización de relojes, pues cada uno de los equipos maneja un reloj independiente, además de que hay un conmutador que los coordina y que también tiene su propio reloj; más aún, el software que gestiona la telefonía cuenta con otro reloj. Así, es importante tener sincronizados todos los relojes porque de lo contrario sería muy complicado hacer un respaldo de la información de audio recabada automática o manualmente de los servicios que se hayan configurado para la telefonía, por ejemplo, para un servicio de buzón de voz o contestadora automática sería muy importante, ya que si la hora no es la misma en cada uno de los nodos de la red ni en la central que los gestiona entonces nunca podría conocerse una hora real en la que se esta recibiendo o escuchando un mensaje, esto traería como consecuencia una desincronización en horas laborales, eventos, entre otros.

En diversos sistemas informáticos de gestión de información también es muy importante la sincronización de relojes, ya que de no existir todas las operaciones de los sistemas como altas, bajas, cobros, devoluciones, pagos, modificaciones jamás confirmarían sus cambios de forma correcta. Además entre una operación y otra se tendría que dejar un lapso de tiempo relativamente grande y eso sería pérdida de producción.

II. OBJETIVOS

Objetivo General

Realizar un sistema con la capacidad de llegar a la sincronización de relojes físicos por medio del método de Berkeley.

Objetivos particulares

- Analizar la granularidad en tiempo de acuerdo a los diversos sistemas operativos y lenguajes de programación que se van a usar.
- Implementar un algoritmo de elección, de un proceso maestro, a partir de un conjunto de procesos corriendo en distintos procesadores.
- Diseñar e implementar un mecanismo de comunicación entre procesos que determine tiempos locales de reloj de los esclavos.
- Estimar la precisión del protocolo de acuerdo a los resultados del punto anterior.
- Construir un mecanismo simple tolerante a fallos para eliminar las lecturas de los relojes defectuosos.
- Diseñar e implementar el mecanismo para actualizar los relojes, tanto del maestro como de los esclavos.

- Inducir casos para desincronizar los relojes de los esclavos.

- Realizar un comparativo de los distintos diseños elaborados para cada módulo según el dispositivo en el cual se implementó (computadora personal, PDA).

- Determinar el ritmo de deriva de los relojes locales de acuerdo a los resultados obtenidos.

III. DOCUMENTACIÓN FINAL

En el presente documento se muestra el preámbulo del proyecto, su finalidad, los módulos que lo integran, así como el desarrollo de los mismos, el manual de usuario y finalmente se analizan y discuten las pruebas que se realizaron en el sistema para llegar a una conclusión respecto a los objetivos establecidos. Al ser un sistema físico compuesto por varios computadores es difícil mostrar su funcionamiento de forma práctica e ilustrativa por lo que se anexa una animación hecha en Adobe Flash que muestra el funcionamiento del sistema.

IV. SINCRONIZACIÓN

La sincronía es un concepto que se puede llevar a diferentes parámetros como pueden ser: instantes, lugares, longitudes, contenidos, entre otros. Por ejemplo, dos personas u objetos pueden estar sincronizados si se encuentran en el mismo lugar, dos objetos llegarían a la sincronía y armonía en un arreglo de objetos si estos compartieran el tamaño y el color, los dispositivos móviles están sincronizados si contienen los mismos archivos de audio o video entre otros, pero en las Ciencias de la Computación el tipo de sincronización que es realmente importante es el de los eventos: en términos de programación y de sistemas operativos, podemos hablar de la sincronía de los procesos durante el tiempo en el que se ejecuta un programa.

La diferencia entre un sistema centralizado y uno distribuido es el lugar en donde se almacena toda la información y desde donde se ejecutan los procesos y programas. Para el caso de un sistema centralizado toda la información se almacena en un servidor o en un arreglo local de servidores que suministran información a cada uno de los clientes que lo solicitan, de igual forma si los clientes necesitan utilizar algún programa o aplicación quien arrancará la ejecución de los mismos y tendrá los procesos que esto genere será el servidor que centraliza la información, lo que causa que toda la administración de recursos la lleve dicho centro de datos y tenga más riesgo de fallar y que la suministración de información se vea considerablemente afectada o suspendida. Un sistema distribuido

el almacenamiento de la información y la ejecución de programas sen encuentran en los diferentes nodos del sistema, esto implica que la administración de los recursos de los que se hace uso necesita de una coordinación de eventos y esto parte de una sincronización. Es muy importante definir los intervalos de tiempo en que uno de los nodos comienza a usar un recurso y hasta que momento lo hace para que el recurso este disponible para los demás nodo, o bien, si este ya se encuentra solicitado por otro nodo, inmediatamente que se termine de proporcionar el servicio al primer solicitante seguirá el segundo en forma sucesiva hasta poder cumplir el número de solicitudes pendientes.

La sincronización de los procesos en un sistema distribuido no es trivial ni automática como para el caso de los sistemas centralizados, en los cuales para saber la hora del sistema o el tiempo de ejecución de un programa únicamente se solicita el tiempo del sistema o se calcula el intervalo de tiempo respectivamente. Para un sistema distribuido es necesario hacer un análisis sobre la arquitectura que se tiene en la red que conecta a cada uno de los nodos, sobre la fuente de información, su colocación, la arquitectura de servidores y finalmente el análisis de las plataformas y los procesadores con los que trabaja cada uno de los equipos de los nodos. Es preciso diseñar procedimientos para obtener la hora del sistema de cada uno de los nodos y para poder tener un punto de partida, posteriormente ya que no hay un centro de datos coordinador hay que diseñar el procedimiento para la elección de un coordinador en el sistema, con los relojes sincronizados en todos los nodos del sistema todas las ejecuciones y transferencias de datos estarán sincronizadas, por lo menos en el tiempo que es el objetivo primordial de este proyecto terminal. Es importante mencionar que los relojes se sincronizarán acelerando o retrasando una variable llamada Deriva del Reloj según sea necesario partiendo de la información del tiempo que tenga el nodo coordinador o el UTC.

V. RELOJES

El principio de la sincronización u ordenamiento en el tiempo son las marcas de tiempo en donde hay dos variables importantes: el día y la fecha. Los equipos de cómputo disponen de un reloj del sistema que toma la hora en la configuración que el usuario le asigne por defecto, por modificación o por instalación, ese es el inicio del intervalo de tiempo del sistema, pero la forma en la que ese reloj obtiene la medición del tiempo es mediante el reloj físico de la computadora que es un oscilador de cristal, cada una de sus oscilaciones está asociada a un segundo para el sistema.

Los relojes de las computadoras por defecto, al igual que los relojes convencionales con los que los seres humanos vivimos día a día, tienden a estar en desacuerdo con otros relojes, tienen una diferencia en tiempo que puede ser considerable o no, y también puede ser en diferentes unidades de tiempo dependiendo de la granularidad del reloj, a tal diferencia se le llama *Sesgo del Reloj*. La razón de la divergencia de los relojes de las computadoras es, porque como se dijo antes, dependen de un oscilador de cristal y la convergencia de cada una de las oscilaciones de un reloj respecto a las de otro es casi imposible. Las oscilaciones de un reloj físico son almacenadas en un contador, el cual es el encargado de transformarlas en segundos y aquí se establece un nuevo contador el cual cada 60 segundos se forma un minuto y así sucesivamente hasta las horas. El sesgo entre los relojes es extremadamente pequeño, sin embargo no es una oscilación la que provoca una diferencia entre relojes

significativa y visible, sino la acumulación de estas oscilaciones a lo largo de un periodo de tiempo determinado, la hora con la que fueron inicializados los relojes es independiente a las variaciones que puede tener el tiempo en un futuro por consecuencia del sesgo del reloj.

El ritmo de deriva del reloj es la compensación necesaria para la sincronización de dos o más relojes en un sistema distribuido. Dicha variable es la que se altera para llegar a la sincronía en un sistema independientemente del número de nodos que haya, para poder realizar esta operación es necesario tener un reloj de referencia, no importa si el tiempo sea mayor o menor, para elegir el reloj de referencia se utilizan otros criterios que lo conviertan en el coordinador del proceso de sincronización. Una vez que se tiene un reloj de referencia se altera al resto de los relojes del sistema, se comienzan a hacer comparaciones con el reloj que fue determinado como patrón referencial, es necesario saber que relojes tienen tiempos más elevados y cuales inferiores a los del reloj principal y calcular proporciones respecto a la magnitud del sesgo, pues esto es lo que va a determinar la medida en que debe alterarse el ritmo de deriva de los relojes para llegar a la sincronía, para los relojes que tienen tiempo superior al del reloj de referencia debe bajar el ritmo de deriva, mientras que para aquellos que tengan tiempo inferior debe acelerarse el ritmo de dicha variable.

Las fuentes de tiempo externas de alta precisión como los relojes atómicos son una herramienta fundamental para llegar a la sincronización del tiempo mediante la alteración del ritmo de deriva de los relojes a partir del sesgo que estos tengan con el reloj atómico. Sin embargo para el caso de este proyecto terminal no se utilizará ninguna fuente externa de tiempo, también conocido como Tiempo Universal Coordinado (UTC), ya que la referencia será un nodo dentro del sistema.

VI. SINCRONIZACIÓN DE RELOJES FÍSICOS

La medición del tiempo tiene su importancia en la necesidad de conocer con precisión los instantes en que suceden los eventos y cuál es su duración, con que frecuencia se realizan, como deben hacerse mejores y ajustes en el tiempo sobre estos eventos. Las personas a diario necesitan saber la hora para poder organizar su tiempo en el trabajo, en el hogar, en los estudios e incluso en su descanso, cada una de las personas debe poseer un reloj que le permita saber la hora en todo momento, y dicho reloj debe estar sincronizado con el resto de los relojes de la zona horaria en donde se encuentre una persona, el UTC en este caso son los días solares que determinan el inicio y el fin de un día, las personas se rigen con horas con formato de Horas, Minutos y Segundos, en donde en la mayoría de las ocasiones los segundos son despreciables y la granularidad común para la coordinación de eventos y actividades entre las personas es de 15 minutos o media hora que equivale a 30 minutos solares.

Para el caso de las computadores el principio es el mismo, ya que se necesita tener una referencia para llevar al resto de los relojes a la sincronía general, la diferencia con los humanos es la granularidad que es necesaria para esta sincronización y el sesgo del reloj que pudiera existir entre dos relojes que están ejecutando procesos en común. En este proyecto terminal en particular la granularidad que se va a manejar es la de milisegundos ya que el recurso primordial para el que se esta buscando la sincronización entre los relojes que tiene cada uno de los equipos que representa a cada uno de los nodos es el acceso a los procesos, entonces, si dos o más nodos del sistema quieren ejecutar un proceso

en común tiene que ponerse en cola de espera aquel nodo que haya solicitado la ejecución del proceso como un P2, mientras que P1 termina de ejecutarse, así el intervalo de tiempo que existe entre P1 y P2 va a ser el tiempo de espera para P2, si la granularidad es demasiado grande, entonces P2 va a esperar demasiado tiempo para poder ejecutarse aunque P1 ya haya terminado, esa es la razón por la cual la granularidad es tan pequeña, el resultado va a ser una sincronía robusta.

El hecho de que la granularidad en un sistema de sincronización de tiempo sea una variable conocida y con valor constante hace posible que el uso de un UTC pueda ser despreciado, ya que el UTC se utiliza para partir de una granularidad específica hasta encontrar el ritmo de deriva del reloj necesario para sincronizar relojes. Existen diversos algoritmos para sincronizar relojes físicos en sistemas de cómputo, algunos de ellos son:

- Algoritmo de Cristian
- Algoritmo de Lamport
- Algoritmo de Berkeley

VII. ALGORITMOS DE SINCRONIZACIÓN

VII.1 Algoritmo de Cristian

La propuesta de Cristian es la utilización de un servidor que proveerá de un punto de tiempo a sus clientes mediante un servidor de tiempo que está relacionado con un UTC, la descripción del algoritmo es la siguiente:

- El proceso P envía mensaje mr solicitando el tiempo al servidor S
- S envía el tiempo en mt
- P ajusta su reloj utilizando el tiempo recibido en mt

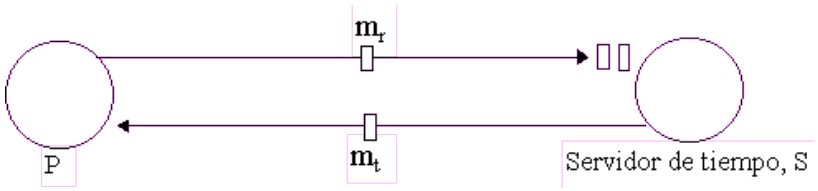

Figura 1. Algoritmo de Cristian

Tciclo= mr + Tproc.S+ mt, en donde tfinal = t + Tciclo/2

VII.2 Algoritmo de Lamport

El algoritmo de Lamport en grandes rasgos consiste en que se tienen una fila de procesos o hilos a los que se les asocia un identificador, el cuál no puede ser igual ya que no pueden generar ni ejecutar al mismo tiempo, por eso los procesos deben liberar los recursos para poder atender una nueva petición, la descripción del algoritmo es la siguiente:

- El proceso Pi desea entrar a una SC, envía el mensaje Tm al resto de los nodos y pone al mensaje en la cola de peticiones. Tm representa la marca de tiempo del mensaje.
- Cuando el proceso Pj recibe el mensaje Tm lo coloca en su cola de peticiones y envía un mensaje de conocimiento al proceso Pi, con su respectiva marca de tiempo.
- El proceso Pi obtiene el recurso cuando se cumplen las siguientes condiciones que son probadas localmente por Pi:
    - Existe un mensaje Tm en la cola de peticiones.
    - El proceso Pi ha recibido un mensaje de todos los procesos con una marca de tiempo mayor a Tm.
- Para liberar el recurso, el proceso Pi quita cualquier mensaje Tm de su cola de peticiones y envía un mensaje Pi libera recurso con marca de tiempo al resto de los proceso.
- Cuando el proceso Pj recibe un mensaje Pi libera recurso, quita cualquier mensaje Tm de su cola de peticiones.

VII.3 Algoritmo de Berkeley

El algoritmo de Berkeley fue propuesto por Gusella y Zatti quienes describieron la sincronización interna entre las colecciones de computadoras por medio de sus relojes físicos, en donde es necesario establecer una computadora maestra o coordinadora, lo que se logra por medio de la generación de procesos. A dichos procesos es necesario asociarles un identificador de proceso, esta variable es única para cada uno de los procesos, se debe localizar al proceso con mayor ID y establecerlo como el coordinador. La descripción del algoritmo es la siguiente:

- Elección de un coordinador por ID.
- Solicitud del coordinador a los nodos para conocer la hora de su reloj.
- El coordinador se solicita la hora de forma local.

- Se hace un promedio de hora.
- La hora resultante del promedio de envía a cada uno de los nodos y se impone al reloj.
- Se tiene sincronía en el sistema.

VIII. DESARROLLO DEL SISTEMA

VIII.1 Visualización y descripción general del sistema

El sistema cuenta con tres nodos:

1) Computadora portátil
2) Computadora portátil
3) PDA

La conexión de estos nodos se lleva a cabo por medio de dos dispositivos de red:

1) Switch
2) Access Point

El sistema propuesto es el siguiente:

Figura 2. Sistema de sincronización de relojes físicos.

VIII.2 Disposición Física del Sistema

El sistema consta únicamente de tres nodos; dos laptops y un PDA, los dispositivos de red que ayudan a intercomunicarlos son un switch y un access point.

VIII.2.1 *Descripción de los equipos:*

Equipo 1:

Lap top HP dv 6220la

- Procesador AMD turion 64 bits
- 120 GB en disco duro y
- 1 GB en memoria RAM
- Sistema Operativo Ubuntu Linux 11.04

Figura 3. Equipo 1

Equipo 2:

Lap top HP dv5-2232la

- Procesador AMD Turion
- 640 GB en disco duro
- 4 GB en memoria RAM
- Sistema Operativo Ubuntu Linux 11.04

Figura 4. Equipo 2

Equipo 3

HP IPAQ 6945, cuenta con conectividad bluetooth, GPRS, así como Windows Mobile 5.

Figura 5. Equipo 3

VIII.2.2 Dispositivos de Red

Switch Kingston de 5 puertos

Figura 6. Switch Kingston

Router Belkin 54g (Modo Access Point)

Figura 7. Router Belkin

VIII.3 Comunicación de lo dispositivos

Es evidente que para tener la posibilidad de sincronizar los relojes de todas las computadoras que haya en un sistema es una necesidad encontrar un mecanismo para que se pueda establecer una comunicación, para que sean visibles unos con otros y para que puedan interactuar entre si al nivel que se desea que lo hagan. En este caso en particular es necesario que las computadoras reciban y envíen información de forma correcta, lo que se desea conocer es la hora del sistema de cada una de las

computadoras del sistema. Los sockets, en esencia son mecanismos que dos procesos hablen entre sí aunque se encuentren en diferentes máquinas por medio de un canal de comunicación.

En términos estrictos de programación un socket es un archivo que se abre de forma especial, una vez que este archivo o canal de comunicación está abierto se puede leer y escribir con las funciones read() y write() respectivamente; para saber el socket que va a leer, el que va a escribir, así como cual es el que va a ejecutar alguna tarea en especial se diferencian mediante identificadores únicos llamados PID y mediante la identificación con una dirección IP para saber a qué computadora pertenecen. Los sockets tienen varias propiedades como son:

- Fiabilidad de transmisión
- Orden de los datos
- No replican datos
- Límites de los mensajes
- Prioridad de mensajes

Los sockets tienes tres diferentes tipos de atributos: dominio, protocolo y tipo; el dominio especifica el medio de comunicación de la red que utilizará el socket, el protocolo indica el protocolo que se utilizará, el tipo indica el servicio de protocolo que se dará, que puede ser de flujo o de datagramas.

Atributos del dominio:

AF_UNIX: Sockets internos de UNIX

AF_INET: Protocolos de Internet ARPA

AF_ISO: Protocolos estándar ISO

AF_NS: Protocolos de redes Xerox

Atributos de los tipos

SOCK_STREAM: Se utiliza para el flujo TCP/IP

SOCK_DGRAM: Se utiliza para procesos que no mantienen ninguna conexión y flujo por datagramas.

Definición de un socket

Para definir un socket es suficiente el número de IP a la que se va a conectar y el número de puerto por el que se va a conectar.

VIII.4 Descripción del Software

En el sistema hay tres equipos que representan tres nodos de información, dos de ellos; la laptop HP y la netbook ACER tienen instalado como sistema operativo Ubuntu Linux 9.04, mientras que la HP IPAQ tiene como sistema operativo Windows Mobile 5. Sin embargo hay que hacer algunas adecuaciones e instalaciones sobre los correspondientes sistemas de cada uno de los equipos y son las siguientes:

VIII.4.1 *Cómputadora HP dv5-2232la (Coordinador)*

- Instalación del paquete Build Essentials
- Instalación de compilador gcc
- Instalación de IDE Netbeans 7.0

VIII.4.2 Computadora HP DV 6220la (Esclavo 1)

- Instalación del paquete Build Essentials
- Instalación del compilador gcc

VIII.4.3 HP IPAQ (Esclavo 2)

- Mysaifu JVM 4.8.0

IX. MANUAL DE INSTALACIÓN Y DE USUARIO

a) Conexión de los nodos del sistema.

Los equipos se conectan a un switch físicamente por medio de cable UTP categoría 5, para la conexión correcta se hace un escalamiento ya que la conexión desde el switch al PDA no es por medio de un cable, la conexión por cable únicamente se hace del switch al router Belkin y esté hace una conexión inalámbrica hasta el PDA.

Características de red:

Tipo de cable: UTP

Categoría: 5e

Tipo de red: Alámbrica /Inalámbrica

Código de colores: Tipo B

Topología de red: Estrella

a.1) Cable UTP: Por sus siglas en inglés (*Unshielded Twisted Pair*) es un medio de comunicación físico que consta de 8 hilos que están trenzados por pares y aislados por plástico, juntos el aislamiento y el trenzado disminuyen las interferencias de comunicación que pudieran existir.

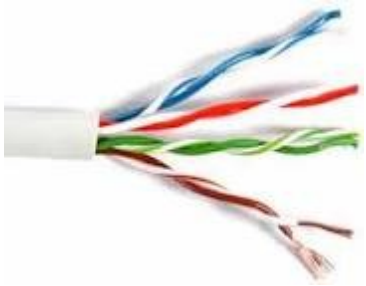

Figura 8. Cable UTP

a.2) Categoría 5e: Permite procesar señales de alta integridad y alcanzar velocidades de 100Mbps hasta 100Mhz.

a.3) Red alámbrica/inalámbrica: La red alámbrica es por medio de cables de red que interconectan los equipos mediante un switch o conmutador y garantiza la correcta transmisión de paquetes entre los equipos a una velocidad buena, mientras que la red inalámbrica es un recurso que se utiliza con equipos que están lejos de un switch y que no son fijos.

Figura 9. Cable UTP Cat 5e

Figura 10. Red Inalámbrica

a.4) Código de colores B: Establece que la conexión haga un recorrido del Pin 1 al Pin 8 con el siguiente orden: Blanco/Naranja, Naranja, Blanco/Verde, Azul, Blanco/Azul, Verde, Blanco/Café, Café y en el otro extremo del cable será la conexión pin a pin debido a que la interconexión se está haciendo con un switch y no equipo a equipo.

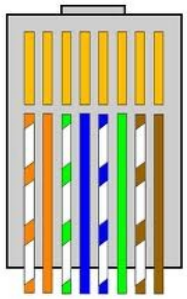

Figura 11. Código de colores B

a.5) Topología de Estrella: Se define como tal debido a que el diseño de la red necesario y que más conviene para lo que se desea hacer demanda un nodo en medio representado por un switch que comunicará al resto de los nodos o equipos en la periferia.

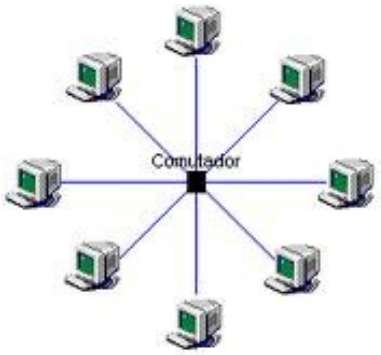

Figura 11. Topología en Estrella

b) Instalación del software

b.1) Instalación del paquete Build-Essentials

b.1.1) Abrir una terminal en el sistema operativo Ubuntu Linux y ejecutar el siguiente comando:

sudo apt-get install build-essentials

b.2) Instalación del compilador gcc

b.2.1) Abrir una terminal en el sistema operativo Ubuntu Linux y ejecutar el siguiente comando:

sudo apt-get install gcc

b.3) Instalación del IDE Netbeans 7.0

b.3.1) Desde cualquier explorador de Internet (por ejemplo Microsoft Internet Explorer, Mozilla Firefox, Google Chrome, etc.) acceder a la dirección http://www.oracle.com y en el apartado “Downloads” descargar el paquete Netbeans IDE 7.0.1.

b.3.2) Ejecutar el archivo descargado desde la página de ORACLE, instalar el IDE.

b.4) Instalación de la máquina virtual Mysaifu

b.4.1) Desde cualquier explorador de Internet (por ejemplo Microsoft Internet

Explorer, Mozilla Firefox, Google Chrome, etc.) acceder a una fuente en donde sea posible descargar el software, como sugerencia http://www2s.biglobe.ne.jp/~dat/java/project/jvm/index_en.html, ir al área de descargas para descargar el paquete JVM Mysaifu.

b.4.2) Vía cable o Bluetooth transferir el paquete comprimido desde a PC en donde se descargo el paquete hacia el PDA.

b.4.3) Abrir la carpeta que se transfirió, seleccionar el archivo CAB y ejecutarlo, de esta manera se iniciará la instalación de Mysaifu.

c) Configuración de la red

c.1) Configuración en Ubuntu Linux. Situarse en Administración -> Conexiones de Red -> Ipv.4, posteriormente seleccionar la interfaz Auto eth0 y dar click en "Editar" y seleccionar la opción de IP Manual, dar click en agregar e introducir los parámetros IP, Default Gateway y Mascara de Subred.

c.2) Configuración en Windows Mobile 5. Situarse en Inicio -> Configuración -> Conexiones -> LAN Inalmbrica -> introducir una IP.

d) Configuración del punto de acceso.

- Acceder a la dirección IP 192.168.2.1 desde un explorador de Internet para tener acceso a la configuración del Router.
- Seleccionar la opción "Use as Access Point"
- El sistema solicitará una contrasea, dejarla en blanco y dar click en "Submit".
- Aparecer una pantalla de advertencia que nos hace saber que el Router se está configurando exclusivamente como punto de acceso, habilitar la opción "Enable" y dar click en el botón "Apply Changes".
- Solicitará una dirección IP y una submáscara de red, introducir dichos valores.
- Se mostrará una pantalla de advertencia que notifica que la IP será cambiada por la indicada y para acceder al panel de configuración del equipo se deber acceder a esa IP.
- Finalmente se debe esperar 25 segundos para que el equipo asigne los parámetros indicados y los cambios puedan ser aplicados, se deberá asignar un nombre a la conexión de red del punto

de acceso.

d.1) Asignación de IP's

- Computadora Coordinador (HP Pavilion dv 2232la): 192.168.1.28
- Computadora Esclavo 1 (HP Pavilion dv5-2232la): 192.168.1.29
- Hand Held Esclavo 2(HP IPAQ): 192.168.1.30
- Router Punto de Acceso (Belkin 54g): 192.168.1.31

Nótese que todos los equipos pertenecen al mismo segmento de red y todos tienen una submáscara de red tipo C (255.255.255.0).

e) Breve descripción de las plataformas de programación y sistemas operativos.

e.1) Lenguaje C (Programación). Es un lenguaje de programación de propósito general que ofrece flexibilidad sintáctica, control de flujo, una cantidad considerable de operadores y estructuras sencillas. C trabaja con varios tipos de datos reconocidos por la mayoría del hardware de las computadoras, estos datos pueden ser tratados o manipulados mediante operadores, estructuras o algoritmos mediante instrucciones por el lenguaje mismo. Un programa en C se compone de una o más funciones. Una de las funciones tiene que ser obligatoriamente main o principal. Una función en C es un grupo de instrucciones que realizan una o ms acciones. Así mismo un programa contendrá una serie de directivas #include que permitirán incluir en el mismo archivo de cabecera que a su vez constarán de funciones y datos predefinidos en ellos.

e.2) Lenguaje Java (Programación). Es un lenguaje de programación de alto nivel que hereda mucha de la sintaxis y características del lenguaje C, pero elimina herramientas de bajo nivel e información basura ya que cuenta con un recolector de basura automático y su modelo de objetos es mucho más simple, el estilo de programación en el que entra Java es una programación con orientación a objetos y esta basada en clases.

e.3) Ubuntu Linux 11.04 (Sistema Operativo). Es un sistema operativo que utiliza el núcleo de Linux y tiene orígenes en la familia de Debian, tiene una orientación hacia el usuario promedio con fácil uso, manejo y configuración y se rige bajo una licencia libre o de código abierto que permite a la comunidad de programadores y desarrolladores hacer cambios o refinamientos al sistema y compartirlos con el resto de los usuarios. Su interfaz de ventanas puede ser GNOME o KDE.

e.4) Windows Mobile 5 (Sistema Operativo). Es un sistema operativo utilizado en teléfonos celulares (Smartphones) y Pocket PCs. Windows Mobile está basado en el kernel de Windows CE 5.2, el uso del sistema se popularizó porque el usuario se familiariza fácilmente con un sistema que maneja en su computadora y generalmente el uso de Windows Mobile se asocia con el uso de un stylus en el dispositivo.

f) Estructura del Sistema de Sincronización de Relojes

f.1) Comunicación entre procesos

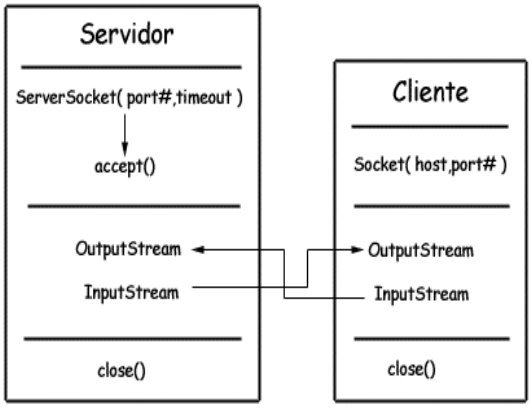

Figura 12. Modelo Cliente Servidor (Sockets)

Como se muestra en la Figura 12 el modelo en el cual se basa el sistema de sincronización de relojes es en el Cliente Servidor mediante programación de Sockets para el paso de mensajes entre el cliente y el servidor, para el caso especifico de este sistema el coordinador es un cliente y los dos esclavos son servidores, ya que el coordinador necesita saber la hora de los esclavos y es el que realiza la solicitud de información.

La secuencia del modelo con respecto a los sockets en el sistema de sincronización de relojes es la siguiente:

- Servidor 1 (Esclavo 1) se levanta
- Servidor 2 (Esclavo 2) se levanta
- Cliente (Coordinador) se conecta a Esclavo 1 y Esclavo 2
- Cliente hace solicitud a Esclavo 1 y Esclavo 2 de hora del sistema
- Cliente envía mensaje con la nueva hora a Esclavo 1 y Esclavo 2

f.2) Compilación de código en C.

Desde la terminal en Ubuntu Linux situarse en la carpeta donde se encuentra el archivo fuente en C. A continuación ejecutar el siguiente comando: gcc -o archivo archivo.c, en donde archivo es el nombre del ejecutable y archivo.c el del programa en C.

f.3) Compilación del código en Java

Utilizando el IDE Netbeans situarse en la clase main del proyecto y seguir la siguiente ruta: Ejecuta -> Ejecutar project.

g) Funcionamiento del sistema

Hay un coordinador fijo que en el modelo cliente servidor es el cliente de dos servidores que en el sistema son los esclavos, el coordinador por medio de un paso de mensajes le va a solicitar a cada uno de los servidores la hora que tienen en su sistema y también se pedirá así mismo la hora de su sistema, entonces va a contar con tres horas diferentes:

1) Hora del sistema del Coordinador
2) Hora del sistema del Esclavo 1
3) Hora del sistema del Esclavo 2

El coordinador va a tener estás tres horas con granularidad milisegundos, es decir, con el formato 00:00:00:000, una vez que tenga los datos va a hacer un promedio entre las tres horas, entonces tendrá una hora resultante, la cual va a enviar por medio de paso de mensajes a los esclavos 1 y 2 para que modifiquen la hora de su sistema, de igual forma lo hará el coordinador, en ese momento el sistema estará sincronizado con granularidad milisegundos.

El algoritmo de sincronización que utiliza el sistema es el algoritmo de Berkeley el cuál realiza uno de los pasos antes descritos; por medio de un coordinador conocer las horas de los demás equipos, promediarlas e imponerlos en cada uno de los equipos del sistema así como en él mismo.

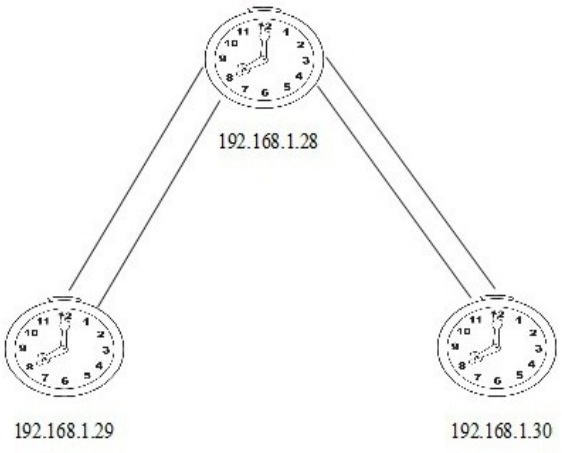

Figura 13. Diagrama de solicitud de tiempo

Como se ilustra en la Figura 13 el coordinador hace dos funciones, la de cliente primero cuando solicita la hora del sistema de los esclavos y como servidor cuando les hace saber que hora deben colocar en su sistema.

h) Conectividad de plataformas y lenguajes.

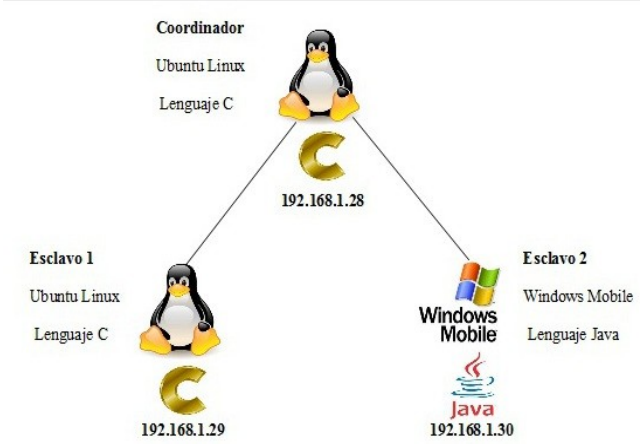

Figura 14. Conectividad de plataformas

Como se muestra en la Figura 14 el coordinador está en un sistema operativo Linux, al igual que el esclavo 1, ambos tienen programados el paso de mensajes mediante sockets y el algoritmo de Berkeley en lenguaje C. Por otro lado el esclavo 2 tiene sistema operativo Windows Mobile y la programación de sockets y el algoritmo están en lenguaje Java, para el caso de la comunicación por sockets entre el coordinador y el esclavo 2 y viceversa se hace un escalamiento que se precisa más adelante en la descripción del pseudocódigo.

Para el caso especifico del esclavo 2 que trabaja con Windows Mobile es necesario utilizar un paquete antes mencionado que es la máquina virtual de Java, ya que Windows Mobile no la tiene instalada por defecto, la JVM (Java Virtual Machine por sus siglas en inglés) que se utilizará será Mysaifu, permite compilar clases en Java o ejecutar archivos .JAR, en este caso desde una PC se hace la programación necesaria en Java y por medio del IDE Netbeans se generan los archivos .JAR, los cuales se envían por medio de Bluetooth al hand held para ejecutarlos desde Mysaifu.

i) Pseudocódigos

i.1) Cliente (Coordinador)

Crear Socket TCP

SI el Socket se creó correctamente solicitar al Sistema Operativo conexión con una IP por medio de un Puerto.

De lo contrario marcar Error de Socket

SI la IP está dentro de la red y el puerto está disponible CONECTADO

OBTENER la hora del sistema local HORA_LOCAL

ESCRIBIR al Servidor1 ¿Cuál es la hora?

LEER del Servidor1 HORA1

ESCRIBIR al Servidor2 ¿Cuál es la hora?

LEER del Servidor2 HORA2

HACER HORA_BERKELEY = (HORA_LOCAL + HORA1 + HORA2)/3

HACER HORA_LOCAL = HORA_BERKELEY

ESCRIBIR al Servidor1 HORA_BERKELEY

ESCRIBIR al Servidor2 HORA_BERKELEY

CLIENTE se DESCONECTA

i.2) Servidor (Esclavos 1 y 2)

Crear Socket TCP

SI el Socket se creó correctamente solicitar al Sistema Operativo conexión con una IP por medio de un Puerto.

De lo contrario marcar Error de Socket

OBTENER la hora del sistema local HORA1

LEER del Coordinador ¿Cuál es la hora?

ESCRIBIR al Coordinador HORA1

LEER del Coordinador HORA_BERKELEY

HACER HORA1 = HORA_BERKELEY

SERVIDOR se DESCONECTA

j) Precisiones de tiempo en el sistema

Dentro del sistema hay procedimientos que hacen que el sesgo de los relojes del sistema sea mayor, uno de ellos es el hecho de que un procesador no puede realizar dos operaciones instantaneas sino una detrás de otra en una cola de prioridad por ocurrencia, por lo que la solicitud del cliente a cada uno de los servidores no se da exactamente al mismo tiempo sino con fracciones de segundo despues entre el primer servidor y el segundo. Sin embargo este sesgo no afecta para obtener un tiempo promedio que se imponga a todos los equipos de la red. El otro caso es más significativo para la obtención de tiempo promedio ya que para que el cliente (coordinador) con plataforma Linux y lenguaje de programación C se comunique con el servidor (esclavo 2) con plataforma Windows Mobile y lenguaje de programación Java es necesario hacer una escala en la comunicación entre los dos como se muestra en la siguiente figura:

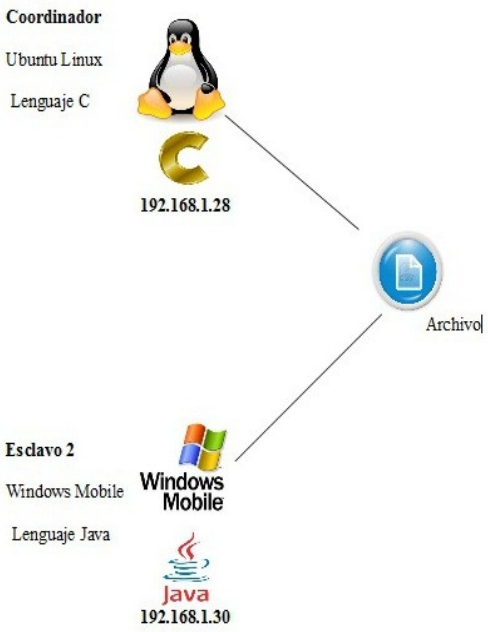

Figura 15. Comunicación C – Java

En la Figura se muestra la escala que se hace para comunicar el cliente se un lenguaje con el servidor del otro, en este caso el servidor con Java crea un archivo de texto en el sistema operativo del cliente y en él escribe la hora de su sistema, por lo que el cliente no se comunica directamente con el servidor, sino que tiene acceso al archivo que el servidor creó en su sistema y lee la hora del servidor, pero esta tarea significa una pérdida de tiempo ya que la obtención de la información no es inmediata, por lo que hay que implementar un método de compensación de tiempo para que no haya pérdida o esta no sea significativa, para lograr esto una vez que el cliente tiene la hora del servidor le va a sumar el tiempo que llevo crear el archivo y escribir en él, de esta forma la precisión no se ve afectada y el sesgo de los relojes será real.

La ecuación es la siguiente:

$$T_{rsj} = t_{sj} + t_{comp}$$

En donde Trsj es el tiempo real del servidor java, t_{sj} es el tiempo del servidor java y t_{comp} es el tiempo de compensación

Después de hacer varias pruebas de ejecución del modelo de cliente – servidor solicitando la hora del sistema se llega a la conclusión de que el tiempo promedio que es necesario para crear el archivo, escribir en él y darle lectura es: 941 milisegundos, indicándolo con esta precisión porque es la granularidad que se definió para el sistema.

X. CONCLUSIONES

Un sistema es capaz de realizar una tarea específica para sí mismo obteniendo información que pueda ser la materia prima para comenzar un proceso más en el mismo sistema, pero los sistemas autónomos no son de gran utilidad sino pueden comunicarse con otros, ya que caen en la ambigüedad, y al no tener contacto con otros sistemas también carece de actualizaciones y datos reales por lo que como consecuencia un sistema pierde confiabilidad de sus resultados. Los sistemas de cómputo necesitan interactuar entre sí, de esta manera hay muchos más lugares de donde se puede obtener información, a donde dirigir información y un área más grande que puede ser controlada, ya que justamente esa es la finalidad de los sistemas de cómputo, establecer un control sobre los procesos y tareas entre cada uno de los equipos que pertenezcan a ella para distribuir actividades como impresiones, administración de procesos, realizar funciones como servidor, hacer escaneos de estado de los equipos, etc, para el caso de los sistemas de tiempo real hay muchos procesos encargados de mantener a los sistemas en un paralelismo en cuanto a tiempo para que pueda existir la interacción en un espacio de tiempo que pueda proporcionar una realidad soportada en intercambio de operaciones, audio y voz sobre todo.

Los sistemas de información están soportados en infraestructura como la que se plantea en este proyecto terminal, ya que necesitan un medio para poder proponer un flujo de información que beneficie a quien administra el sistema, pero sin duda alguna a quien más debe beneficiar es a los usuarios que a diario y en todo momento consultan y registran información en el sistema, porque es conveniente que dichos usuarios que están en contacto siempre con tecnologías que les brindan muchas opciones de adaptación a sus necesidades personales de comunicación. El control es la herramienta por la cual los seres humanos tienen especial interés en desarrollar sistemas de tiempo real cada vez más sofisticados ya que una tarea no tiene sentido si se realiza tarde o después del justo momento en el que se desea o se requiere realizar, los sistemas de tiempo real no son más que el instrumento con el que el ser humano cuenta para hacer sus actividades diarias ayudado por un avión que lo transporta, un cajero que le proporciona o le solicita dinero, un teléfono celular que lo mantiene informado de su alrededor y le permite reportar acerca de su situación actual, de la televisión misma que depende de la hora del día y la duración de los programas, las fabricas que producen y dan trabajo a millones de personas.

Las necesidades de las personas son los requisitos que debe tener un sistema para existir y funcionar de forma correcta, cuando varias personas trabajan para una empresa o en un mismo lugar, entonces un sistema debe de tener la capacidad de administrar los recursos en un periodo de tiempo bien definido para un usuario bien definido, así el primer recurso con el que un usuario cuenta es con el tiempo que solicite y le sea asignado por el sistema. Los componentes de un sistema de tiempo real son principalmente los siguientes:

- Integración de aplicaciones
- Manejo de interrupciones
- Bases de datos de tiempo real
- Sistemas operativos de tiempo real

- Sincronización de relojes lógicos y físicos

En este proyecto terminal se hace una aportación sobre dos de estos componentes, que son los sistemas operativos de tiempo real y sobre la sincronización de relojes físicos, en ambos el tiempo es un principio fundamental y cuando este es paralelo entre dos o más equipos se puede hablarde que el sistema está trabajando en tiempo real, pero dentro hay diferencias en el tiempo que pueden ser perceptibles o no para la ejecución de una tarea, para el caso de este proyecto el tiempo es significativo hasta el grado de los milisegundos, ya que es a esta granularidad donde se sincronizan los relojes físicos de los equipos en el sistema, pero no sin antes tener la estructura en los sistemas operativos correspondientes. Dicho proyecto terminal puede tener continuidad dentro de un sistema informatico de facturación, de cobro, de consulta de saldo en el que se ejecutan diferentes instrucciones y transacciones dentro de un sistema distribuido que requiera que todos los equipos sin importar tipo y ubicación tengan la misma hora con una granularidad muy pequeña, incluso en nanosegundos. Así mismo puede utilizarse como base de un sistema mecánico que utilice temporizadores para movimientos mecánicos, cambios de temperatura o de estado.

XI. BIBLIOGRAFÍA

[1] Gusella, R. y Zatty, S. The accuracy of clock synchronization achieved by TEMPO in Berkeley UNIX 4.3BSD. *IEEE Transactions Software Engineering*, vol. 15.

[2] Coulouris, G., Dollimore, J y Kindberg, T. Sistemas Distribuidos Conceptos y Diseño. Tercera edición, Editorial Pearson.

[3] Tanenbaum, A., Van Steen, M., Sistemas Distribuidos Principios y Paradigmas, Segunda Edición, Editorial Pearson.

[4] Deitel, P.J., Deitel, H.M., Java Cómo Programar, Séptima Edición, Editorial Pearson.

[5] Joyanes Aguilar, L., Zahonero Martínez, I., Programación en C Metodología, Algoritmos y Estructura de datos, Segunda Edición, Editorial McGraw Hill.

[6] Tanenbaum, A., Sistemas Operativos Modernos, Tercera Edición, Editorial Pearson

[7] Tanenbaum, A., Redes de Computadoras, Tercera Edición, Editorial Prentice Hall.

[8] Sistemas Operativos-Coordinación y Acuerdo. [En línea] http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/MonogSO/COORD02_archivos/SO-CyA-Elecciones.htm