Instituto Politécnico Nacional

Escuela Superior de Cómputo

Desarrollo de Sistemas Distribuidos

Prof. **Benjamín Cruz Torres**

Práctica No. 5
Sincronización

Grupo: 4CV3

Equipo: 6

Integrantes:

1. Acosta Rosales Jair Sebastián
2. De Jesús López David
3. Galicia Vargas Gerardo
4. Martínez Marcos Luis Eduardo
5. Octaviano Lima Elvia Jaqueline

Fecha: 09 de noviembre de 2018

Práctica 5: Sincronización de equipos

Objetivo de la Práctica Que el alumno comprenda la importancia de la sincronización en sistemas distribuidos.

Tecnologías a aplicar: Sockets, RMI, SOAP, Hilos (threads), POO, Protocolos de comunicación, Bases de Datos, algoritmos de sincronización.

Actividades

A partir de la práctica 4, desarrollará una aplicación para repartir cartas de baraja francesa a través de dos servidores. Los servidores deberán tener sus relojes sincronizados. De acuerdo a los siguientes requerimientos:

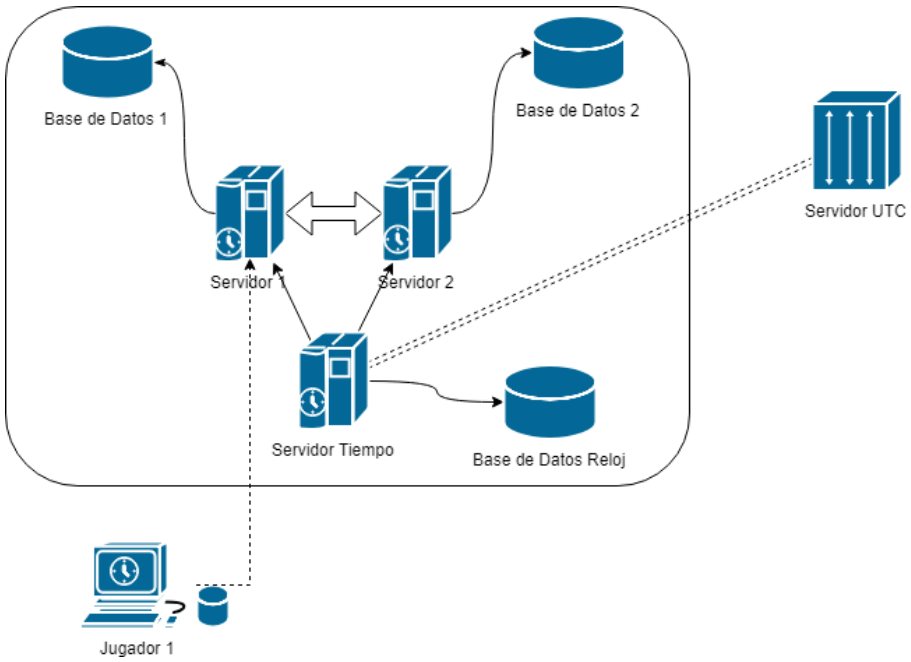


Figura 1. Sincronización usando el algoritmo de Cristian

Requerimientos funcionales

- Los servidores son los únicos conectados y con acceso a las bases de datos.
- Dentro de la interfaz de cada servidor repartidor habrá un botón de “Reiniciar” y un Canvas para poner una imagen, inicialmente vacío.
- Dentro de la interfaz del servidor de tiempo habrá un botón de “Sincronizar”
- En cada computadora Servidor y Jugador está el reloj de la práctica 1.
- En cada computadora Jugador hay un botón de “Pedir carta”. Éste manda una petición al servidor correspondiente, con la cual dicho servidor envía la información de una carta al azar a ese Jugador.
- La información de petición (IP, hora, carta) se guardará en la base de datos.
- El botón “Reiniciar”, permitirá reiniciar la partida del lado de cada servidor.
- La carta elegida se mostrará (en forma de imagen) solamente en la interfaz gráfica del coordinador. En el cliente se mostrará solamente en formato de texto.

- Cuando termina la partida (se repartieron todas las cartas) se le notificará al usuario si quiere salir o reiniciar una nueva partida.
- Al inicio los relojes tienen horas diferentes elegidas al azar.
- Todos los relojes se pueden modificar manualmente por el usuario.
- El Servidor de Tiempo tendrá que sincronizar el sistema cada que se presione el botón “Sincronizar”.
- El formato de hora es de 24 hrs.

Requerimientos no funcionales

- Al pulsar el botón de modificar, el reloj correspondiente se detendrá.
- Tomar en cuenta la latencia al enviar los ajustes por parte del servidor de tiempo.
- Para sincronizar se utilizará el algoritmo de Cristian
- Una vez que se calcula la nueva hora de referencia el coordinador envía los ajustes correspondientes.
- Cada vez que se calcule la hora de referencia se deberá crear un nuevo registro en la Base de Datos Reloj.
- La Base de Datos Reloj tendrá la siguiente estructura.

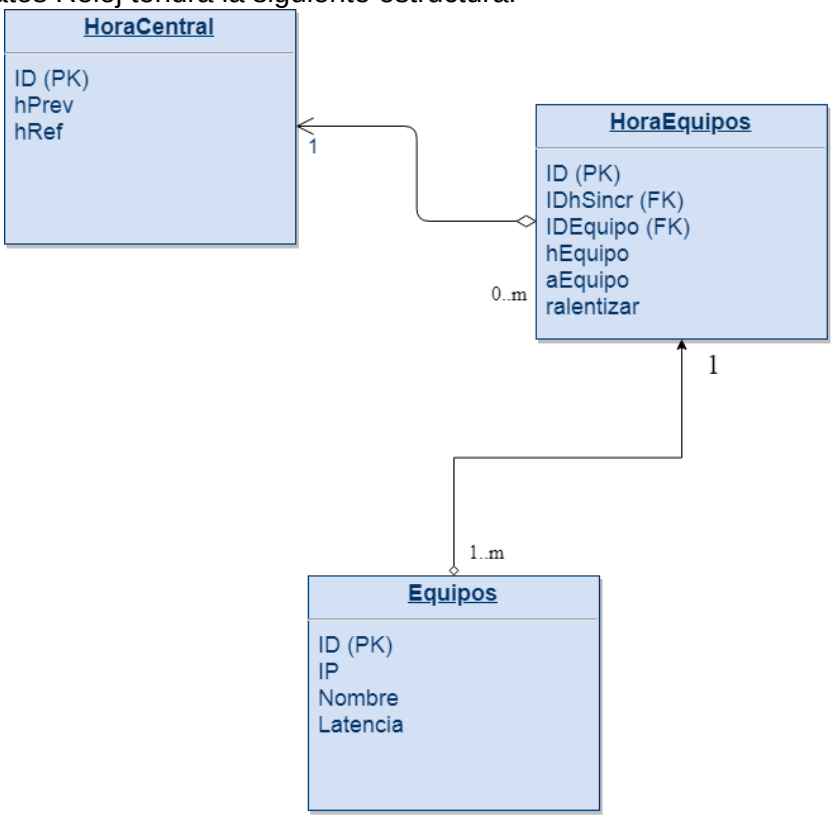


Figura 1. Modelo propuesto

Introducción

En esta práctica se incluye la utilización del mecanismo RMI para emplear métodos remotos como si fueran locales, permitiendo que la base de datos asociada a cada servidor se actualice de tal forma que ambas persistan todas las operaciones realizadas.

Además, se agrega un servidor UTC que implementa un algoritmo de sincronización para que los relojes locales se ajusten gradualmente.

Desarrollo de la práctica

Retomando los avances hechos en las prácticas anteriores, se retoma la funcionalidad del mecanismo RMI, con lo que los clientes solicitan cartas a los servidores y los servidores se mantienen en comunicación persistiendo todas las solicitudes de cartas.

Con respecto de la anterior práctica, se agrega un servidor UTC

{Incluye capturas de pantalla de la aplicación}
En la imagen se muestran las interfaces de los servidores: Servidor J1, Servidor J2, Servidor UTC y de los jugadores: Jugador S1 y Jugador S2.

Se determina el cálculo del ajuste, para lograr la sincronización de los relojes:

$$ajuste = (lhoraUTC - lhoraCliente) - (\frac{latencia}{2})$$

Como se implementó en la práctica anterior, las interfaces de los servidores permiten modificar la hora de los relojes, y reiniciar la partida.

Las interfaces de los jugadores permiten modificar la hora de los relojes y solicitar cartas a sus correspondientes servidores.

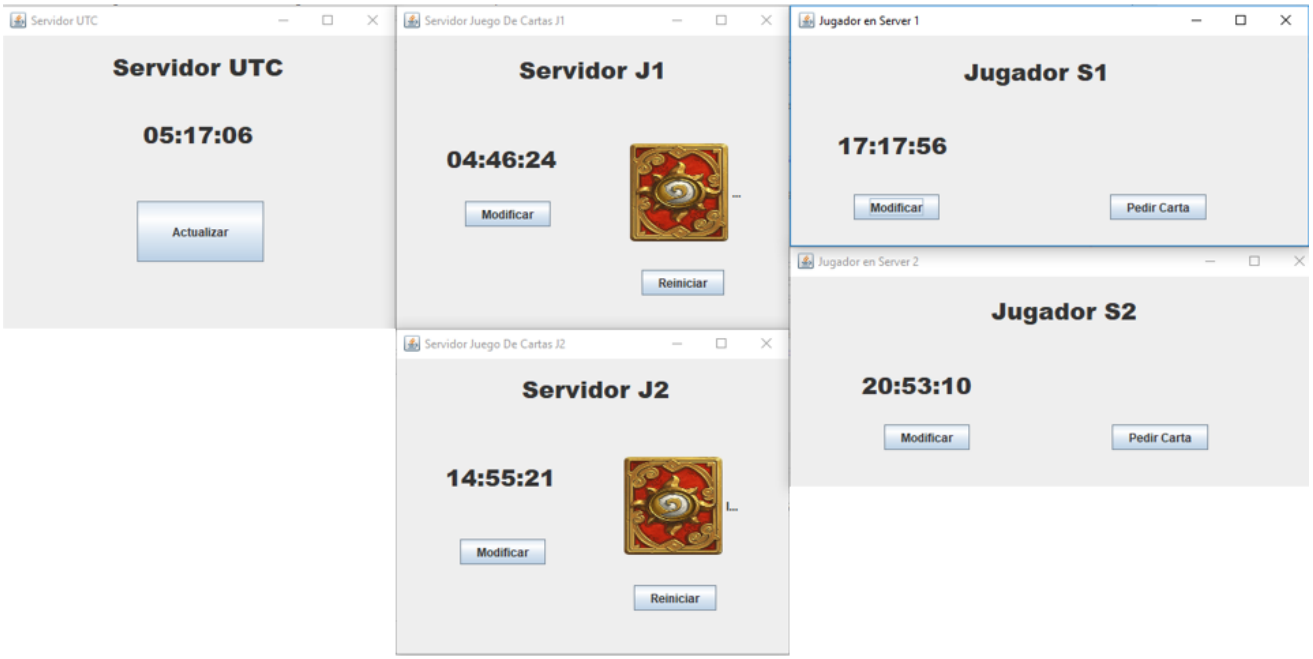


Figura 2. Interfaces de la

A continuación se incluyen las pruebas.

Primero se ejecuta el servidor UTC.

P5ServidorUTC (run) | P5ServidorJ1 (run) | P5ServidorJ2 (run) | JugadorRMI_S1 (run) | JugadorRMI_S2 (run)

```
run:
Escuchando en: us-w10latitt-45/192.168.168.1 En puerto: 3232
referencia a server 1 es null
referencia a server 2 es null
referencia a server 1 es null
referencia a server 2 es null
```

Éste se mantiene a la espera de una referencia a los servidores; una vez ejecutados, el UTC obtiene las correspondientes referencias.

P5ServidorUTC (run) | P5ServidorJ1 (run) | P5ServidorJ2 (run) | JugadorRMI_S1 (run) | JugadorRMI_S2 (run)

```
run:
Escuchando en: us-w10latitt-45/192.168.168.1:3233
Randoms hora:04 minutos:20 segundos:20
```

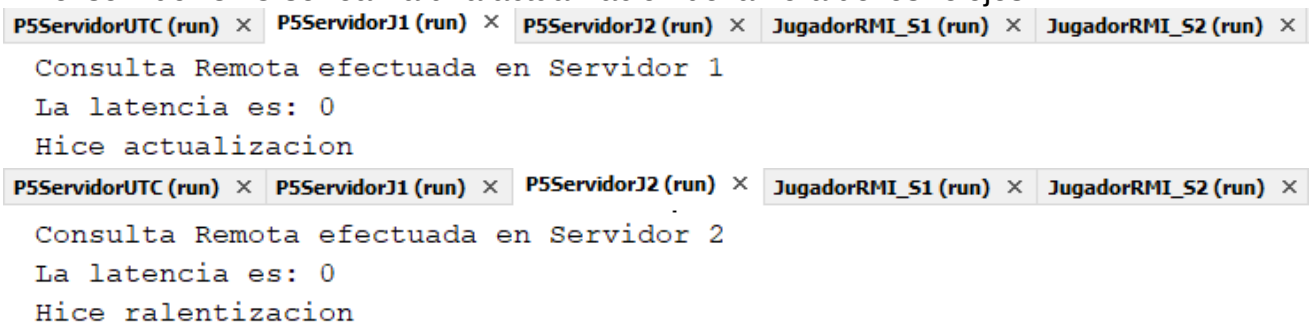
P5ServidorUTC (run) | P5ServidorJ1 (run) | P5ServidorJ2 (run) | JugadorRMI_S1 (run) | JugadorRMI_S2 (run)

```
run:
Escuchando en: us-w10latitt-45/192.168.168.1:3234
Randoms hora:14 minutos:19 segundos:19
fecha: 02/Nov/2018 hora: 14:51:19
Se guardo el registro insert Partida
```

Se completa la comunicación entre los servidores y se realiza el registro en las bases de datos sobre el inicio de la partida.

ID	fecha	hora_inicio	hora_fin
1	02/Nov/2018	04:42:24	NULL
2	02/Nov/2018	14:51:19	NULL

En el servidor UTC se realiza una actualización de la hora de los relojes.



El cálculo de la actualización, la latencia es de 0, por lo que el cálculo de la latencia da como resultado una ralentización de los relojes en los servidores hasta que estos se encuentran sincronizados.

Conclusiones

Los algoritmos de sincronización, aplicados en este caso a través del servidor UTC, permiten mantener mayor consistencia en los datos almacenados, ya que por lo menos se acercan los valores de las horas en que se realizan las operaciones.

Bibliografía

Comunicación entre un servidor y múltiples clientes. (s.f.). Webtutoriales.com. Recuperado de: http://www.webtutoriales.com/articulos/comunicacion-entre-un-servidor-y-multiples-clientes

González-J., Agustín. (2009). *Remote Method Invocation (Invocación Remota de Métodos)*. Recuperado de: http://profesores.elo.utfsm.cl/~agv/elo330/2s09/lectures/RMI/RMI.html