Enunciado-Prueba.pdf

Juandf03

Redes y Sistemas Distribuidos

2º Grado en Ingeniería Informática

Escuela Técnica Superior de Ingeniería Informática
Universidad de Málaga

E.T.S. INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

Redes y Sistemas Distribuidos
2º curso de los Grados de Ing. Informática, Ing. del Software, Ing. de Computadores e Ing. Informática y Matemáticas

Prueba de evaluación del bloque II

Número del Equipo: _____
Apellidos, Nombre: _____

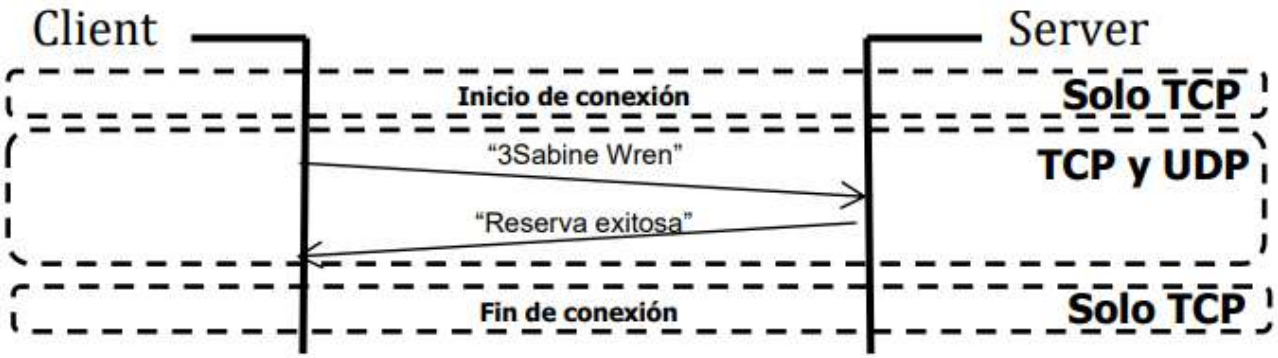
En el campus virtual se ofrece un fichero **`bloque2.zip`** que incluye: un servidor TCP (`ServerTCP.java`), un cliente TCP (`ClientTCP.java`), un servidor UDP (`ServerUDP.java`) y un cliente UDP (`ClientUDP.java`).

Visto el éxito de festivales internacionales como Coachella, la ciudad de Málaga quiere organizar "The Axarquía Music Festival" con cantantes de renombre como el Conejo Malo, Clavelía, Monoz, los Hemanos Químicos… Para ello necesita un sistema de venta de entradas. Para probarlo quieren un sistema que funcione tanto con TCP como UDP y decidir la mejor opción.

Tanto en TCP como UDP se desarrolla el mismo servicio en el que el servidor recibe un texto (como mucho 200B) donde la primera letra es un valor entre `'1'` y `'9'`, correspondiente al número de entradas deseadas y luego un texto, con el nombre de la persona al que corresponde la reserva. Por ejemplo, un posible envío del cliente podría ser **"`3Ezra Bridger`"** indicando que quiere 3 entradas para el festival a nombre de Ezra Bridger. El servidor responderá con **"`Reserva exitosa`"** si todo fue bien o **"`Reserva fallida: XXX`"** si hubo algún problema (`XXX` indica el motivo del error).

En ambos casos (TCP y UDP), el cliente leerá un par de número + texto de teclado, compone el mensaje apropiado, lo envía y tras recibir la respuesta la muestra. En el siguiente diagrama se muestra un ejemplo de los mensajes intercambiados:

En ambos casos (TCP y UDP), el cliente leerá un par de número + texto de teclado, compone el mensaje apropiado, lo envía y tras recibir la respuesta la muestra. En el siguiente diagrama se muestra un ejemplo de los mensajes intercambiados:



En su IDE favorito cree un proyecto Java y copie esos cuatro ficheros incluidos en **`bloque2.zip`**. Resuelva los siguientes ejercicios (el orden de realización puede ser cualquiera).

Ejercicio 1 (Programación UDP) – 3.5 puntos: En los ficheros `ServerUDP.java` y `ClientUDP.java` se encuentra implementando parcialmente el servicio descrito previamente se pide finalizarlo y extenderlo de la siguiente forma:

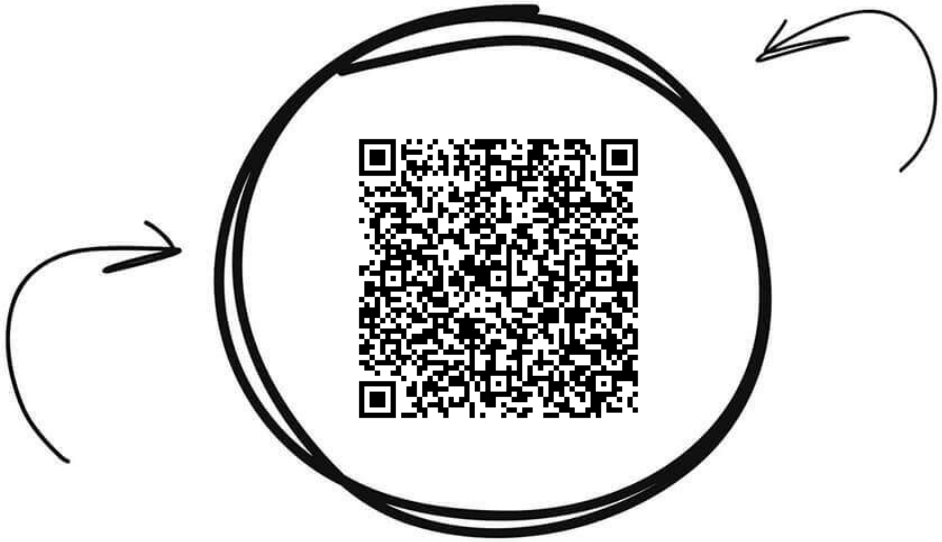
- [1A] El servidor debe usar como puerto el 15000.

ColaCao

APROBASTE LA COURSE NAVETTE, SUPERASTE A TU EX E HICISTE NUEVOS AMIGOS. ESTE EXAMEN NO ES NADA PARA TI. TÚ PUEDES.

#NOPARESDESER única

Redes y Sistemas Distribuidos



Banco de apuntes de la

WUOLAH

Comparte estos flyers en tu clase y consigue más dinero y recompensas

1. Imprime esta hoja
2. Recorta por la mitad
3. Coloca en un lugar visible para que tus compis puedan escanar y acceder a apuntes
4. Llévate dinero por cada descarga de los documentos descargados a través de tu QR

- [1B] El servidor facilitado recibe los mensajes, genera la respuesta, pero no la envía. Complete el código para enviarla.
- [1C] El cliente debe enviar los datagramas al servidor (dirección IP la de `localhost` y puerto 15000).
- [1D] El cliente podrá enviar varias reservas hasta que el usuario pida un número de entradas igual a 0.
- [1E] El servidor ya controla cuántas entradas pide cada cliente. Modifique el código del servidor para comprobar que si las entradas que ya llega solicitadas más las entradas que pide ahora superan 10 entradas envíe como mensaje: "`Reserva fallida: Ha superado el máximo número de reservas`".
- [1F] Note que como usamos UDP, un equipo malicioso podría monitorizar las comunicaciones y al enviar el cliente, respondernos más rápido que el servidor y el cliente aceptaría esa respuesta. Para evitar esto, en el código, la recepción de la respuesta está dentro de un bucle. Complete/modifique la condición del bucle para que solo permita salir del bucle si la respuesta es realmente del servidor. NOTA: los objetos de la clase `InetAddress` disponen del método `equals` para comparar.

Las clases modificadas (`ServerUDP.java` y `ClientUDP.java`) súbalas a la tarea **`Ej. 1 - Programación UDP`** del campus virtual.

Ejercicio 2 (Programación TCP) – 3.5 puntos: En los ficheros `ServerTCP.java` y `ClientTCP.java` se encuentra implementando totalmente el servicio descrito previamente se pide extenderlo de la siguiente forma:

- [2A] El servidor debe usar como puerto el 25000 y con un tamaño de cola de 10.
- [2B] El cliente debe conectarse al servidor (dirección IP la de `localhost` y puerto 25000).
- [2C] El cliente puede hacer varias reservas en la misma conexión de la siguiente forma: (a) primero pedirá un número `N` al cliente y lo leerá, (b) enviará ese valor (como texto) al servidor, (c) leerá pares número de entradas + nombre y harás las peticiones al servidor tal como se hacían de forma original.
- [2D] El servidor debe ser adaptado de igual forma, es decir, primero recibirá del cliente un texto que representa un número (`N`) y luego leerá `N` peticiones y responderá de forma oportuna cada una de ellas.
- [2E] El servidor ya controla cuántas entradas pide cada cliente. Modifique el código del servidor para comprobar que si las entradas que ya llega solicitadas más las entradas que pide ahora superan 10 entradas envíe como mensaje: "`Reserva fallida: Ha superado el máximo número de reservas`"

- [2D] El servidor debe ser adaptado de igual forma, es decir, primero recibirá del cliente un texto que representa un número (`N`) y luego leerá `N` peticiones y responderá de forma oportuna cada una de ellas.
- [2E] El servidor ya controla cuántas entradas pide cada cliente. Modifique el código del servidor para comprobar que si las entradas que ya llega solicitadas más las entradas que pide ahora superan 10 entradas envíe como mensaje: "`Reserva fallida: Ha superado el máximo número de reservas`" y cierre la conexión con el cliente.
- [2F] Cuando el cliente reciba "`Reserva fallida: Ha superado el máximo número de reservas`" debe parar de enviar peticiones y cerrar de forma apropiada la conexión.

Las clases modificadas (`ServerTCP.java` y `ClientTCP.java`) súbalas a la tarea **`Ej. 2 - Programación TCP`** del campus virtual.

Ejercicio 3 (Análisis UDP) – 1.5 puntos: En la traza de wireshark **`UDP.pcapng`** facilitada en el campus virtual, se recogen los intercambios de mensajes de varios clientes con un servidor que ofrece el servicio anteriormente descrito (el puerto del servidor puede variar). Abra el fichero y rellene el cuestionario **`Ej. 3 - Análisis UDP`** del campus virtual.

Ejercicio 4 (Análisis TCP) – 1.5 puntos: En la traza de wireshark **`TCP.pcapng`** facilitada en el campus virtual, se recogen los intercambios de mensajes de varios clientes con un servidor que ofrece el servicio anteriormente descrito (el puerto del servidor puede variar). Abra el fichero y rellene el cuestionario **`Ej. 3 - Análisis TCP`** del campus virtual.