Universidad Simón Bolívar
Departamento de Computación
Redes de Computadores I
Prof. Wilmer Pereira

Proyecto1 (8%)

1. Objetivo

Debe definir una aplicación cliente/servidor usando *sockets*, en el lenguaje de su preferencia, para comunicar clientes con tres servidores que ofrecen libros en formato `pdf`. La funcionalidad básica es que los clientes pueden consultar la oferta de libros de los servidores y solicitar los documentos que deseen.

La interacción será por línea de comandos sobre un terminal de linux donde el cliente puede solicitar información de los libros, recuperarlos y recibir *feedback* del estado de su descarga, sin perder el control del terminal. Desde la cónsola del servidor, el administrador, podrá hacer diversas consultas sobre la información ofrecida y verificar el estado de las descargas en curso.

2. Cliente

El cliente sabrá de antemano la ubicación de los tres servidores de descarga, es decir, sus IP's y puertos de escucha. Una vez que ya solicitó la lista de libros disponibles, puede ocurrir que el libro deseado, esté en más de un servidor. En ese caso, lo descargará de cualquiera de los servidores y la cónsola debe quedar disponible para hacer cualquier otra descarga, es decir, debe haber concurrencia del lado del cliente usando procesos livianos (hilos) o procesos pesados (*fork*).

Los comandos que puede ejecutar el cliente por cónsola son:

1) `ESTADO_DESCARGAS`
2) `LISTA_LIBROS`
3) `SOLICITUD` *`libro`*
4) `LIBROS_DESCARGADOSxSERVIDOR`

`LISTA_LIBROS` será todos los documentos que ofrecen los tres servidores. El cliente puede escribir los comandos por cónsola o valerse de un menú para ofrecer las distintas opciones de consulta.

La información para `LIBROS_DESCARGADOSxSERVIDOR` debe ser guardada permanentemente y conservarse entre sesión y sesión, es decir, aunque se apague el computador o el cliente deje la aplicación, debe mantenerse el histórico de las descargas. Utilice archivos XML o JSON para guardar esta información.

Es necesario considerar el tratamiento de errores y cada eventualidad debe ser capturada por su programa ya que durante la corrección se probarán escenario de caída de servidores y cliente aún durante la recuperación de libros.

Por último, no debe haber pérdida en la descarga si esta ha superado el 50% del libro. En consecuencia, si hay una caída y se ha recibido más del 50% del libro, al levantarse de nuevo el cliente, debe comenzar desde la mitad del archivo para no perder lo descargado hasta ese momento.

3. Servidor

Las solicitudes deben ser respondidas concurrentemente, es decir, varios clientes pueden descargar el mismo o diferentes libros al mismo tiempo. También la cónsola debe estar disponible para estadísticas las cuales son:

```
1) LIBROS_DESCARGADOS
2) CLIENTE_QUE_CONSULTARON
3) NUM_DESCARGASxLIBROxCLIENTE
4) DESCARGAS_EN_CURSO
```

Al igual que en el cliente, debe considerarse el procesamiento de errores. Por ejemplo, ante caídas de un servidor los clientes pueden seguir consultando los otros dos y, sólo si están los tres caídos, habrá un mensaje de imposibilidad del servicio ante el cliente.

4. Condiciones de la entrega

- Cada actor (cliente o servidores) debe correr en máquinas diferentes. La corrección se hará en una máquina con varias ventanas abiertas sobre las máquinas remotas, cada una con un programa cliente o servidor ejecutándose.

- El proyecto es en grupos de máximo dos personas o individual.

- No es necesario implementar interfaz gráfica. Concéntrese en las funcionalidades de redes que será lo evaluado en el criterio de corrección.

- Coloque tiempo de espera en su código, para poder verificar las estadísticas, como `ESTADO_DESCARGA` (cliente) o `DESCARGAS_EN_CURSO` (servidor), es decir, ralentice las descargas para poder hacer seguimiento durante la corrección de lo que ejecutan tanto el cliente como el servidor.

- La corrida es presencial y debe estar presente ambos integrantes del equipo.
-
- Dentro del criterio se evalúa el dominio del código por ambos miembros del grupo

- El informe debe contener:
    - Una muy breve descripción de la estructura del código
    - Los diagramas de secuencia que indiquen claramente como se logra la interacción entre el cliente y los servidores. Esto permite definir las reglas de comunicación o protocolo entre los diferentes actores.
    - Por último, es importante que tenga una bibliografía que indique claramente si usa código de internet y desde donde lo descargo. Evite ser sancionado por copia.
- El código debe tener documentado al menos los métodos o módulos, estilo `javadoc`. En caso de usar un lenguaje diferente a Java, debe respetar esa estrategia de documentación.