Programación Distribuida y Tiempo Real
Facultad de Informática
Universidad Nacional de La Plata

Práctica 3

1) Manejo de Errores de Conectividad

Usando como base el programa ejemplo1[1] de gRPC, realice los siguientes experimentos para simular y observar fallos de conectividad tanto del lado del cliente como del servidor:

a) Introduzca cambios mínimos, como la inclusión de exit(), para provocar situaciones donde no se reciban comunicaciones o no haya un receptor disponible. Agregar *screenshots* de los errores encontrados.
b) Configure un **DEADLINE** y modifique el código (agregando, por ejemplo, la función sleep()) para provocar la excepción correspondiente. Agregar *screenshots* de los errores encontrados.

2) Análisis de APIs en gRPC

Describa y analice los distintos tipos de APIs que ofrece gRPC. Con base en el análisis, elabore una conclusión sobre cuál sería la mejor opción para los siguientes escenarios:

a) Un sistema de **pub/sub**.
b) Un sistema de **archivos FTP**.

Nota: Desarrolle una conclusión fundamentada considerando los siguientes aspectos para ambos escenarios (pub/sub y FTP):

- **Escalabilidad**: ¿Cómo se comporta cada API en situaciones con múltiples clientes y conexiones simultáneas?
- **Consistencia vs. Disponibilidad**: ¿Qué importancia tiene mantener la consistencia de los datos frente a la disponibilidad del sistema?

[1] https://github.com/pdytr/pdytr-grpc-demo.git

- **Seguridad**: ¿Qué mecanismos de autenticación, autorización y cifrado se deben utilizar para proteger los datos y las comunicaciones?
- **Facilidad de implementación y mantenimiento**: ¿Qué tan fácil es implementar y mantener la solución para cada API?

3) Desarrollo de un Chat Grupal utilizando gRPC

Implemente un sistema de **chat grupal** simplificado utilizando gRPC, que permita la interacción entre múltiples clientes y un servidor central. El sistema debe incluir las siguientes funcionalidades:

1. **Conectar**: Permite a un cliente unirse al chat grupal.
    - **Entrada**: Nombre del cliente.
    - **Salida**: Confirmación de la conexión (nombre del cliente y mensaje de bienvenida).
2. **Desconectar**: Maneja tanto la desconexión voluntaria de un cliente como la desconexión involuntaria (por ejemplo, mediante el uso de Ctrl-C o por pérdida de conexión).
    - **Entrada**: Nombre del cliente que se desconecta.
    - **Salida**: Confirmación de la desconexión y mensaje de despedida.
3. **Enviar Mensaje**: Permite a un cliente enviar un mensaje al servidor, que se encargará de retransmitirlo a todos los demás clientes conectados al chat.
    - **Entrada**: Nombre del cliente y contenido del mensaje.
    - **Salida**: Confirmación de envío y distribución del mensaje a todos los clientes conectados.
4. **Historial de Mensajes**: Cualquier cliente puede solicitar el historial completo de mensajes intercambiados en el chat mediante el comando especial /historial.
    - **Entrada**: Comando /historial.
    - **Salida**: Archivo de texto o PDF con el historial de mensajes (marcas de tiempo, nombres de los clientes y mensajes).

[20240428 - 12:16:53] Cliente 1: Buen día grupo

[20240428 - 12:18:55] Cliente 2: Buen día, ¿realizaron el punto 3 del tp3 de distribuida?
[20240428 - 12:19:30] Cliente 3: Si, el punto está resuelto.

Requerimientos de Implementación:

- **Servidor**:
    - El servidor debe ser capaz de manejar múltiples clientes concurrentemente, permitiendo la comunicación simultánea entre ellos.
    - El servidor debe llevar un registro actualizado de los clientes conectados, eliminando a aquellos que se desconecten de manera voluntaria o involuntaria.
    - El servidor debe mantener un archivo de historial de mensajes que registre todas las conversaciones, para permitir la consulta posterior.
    - Documente todas las decisiones tomadas durante el proceso de diseño e implementación del servidor, justificando los enfoques utilizados para la concurrencia y la gestión de clientes.
- **Clientes**:
    - Implemente al menos tres clientes que se conecten al servidor, envíen mensajes y reciban los mensajes de otros usuarios.
    - Los clientes deben manejar de manera adecuada las excepciones relacionadas con la desconexión involuntaria o fallos en la conectividad.
- **Concurrente y Escalable**: El sistema debe permitir que varias instancias de clientes se conecten y envíen mensajes de forma concurrente sin que el rendimiento se vea afectado.

Nota: Para cada funcionalidad desarrollada, el código debe estar debidamente comentado y explicado en el informe final. El informe debe detallar las decisiones clave tomadas en el diseño y construcción del sistema, así como cualquier problema encontrado y cómo fue resuelto.

4) Análisis de Concurrencia y Eficiencia

Después de implementar el sistema de chat grupal, realice un análisis sobre la concurrencia y eficiencia del servidor:

- **Concurrencia**: Diseñe un experimento para demostrar si el servidor es capaz de manejar múltiples solicitudes de clientes de manera concurrente. Esto incluye evaluar el comportamiento del servidor cuando varios clientes envían mensajes al mismo tiempo. Si se encuentran problemas de concurrencia, proponga soluciones y documente cómo se podrían aplicar.

Nota: El análisis de concurrencia y eficiencia debe incluir gráficos o tablas que muestren los resultados de las mediciones, junto con una interpretación de los mismos en el contexto del sistema de chat grupal.

5) Medición de Tiempos de Respuesta

a) Diseñe un experimento que permita medir el tiempo de respuesta mínimo de una invocación en gRPC. Calcule el promedio y la desviación estándar.
b) Utilizando los datos obtenidos en la Práctica 1 (Sockets), realice un análisis comparativo de los tiempos de respuesta. Elabore una conclusión sobre los beneficios y complicaciones de cada herramienta.

Entrega de la práctica (individual o en grupos de dos alumnos como máximo):

- Se debe entregar un único informe detallando lo realizado para cada ejercicio. Debe tener un formato bien definido identificando materia, trabajo práctico y autor/es. Se debe entregar en formato electrónico con tipo de archivo .pdf, en tamaño de hoja A4.
- Para cada programa modificado o generado para resolver los ejercicios, debe explicarse el cambio o la implementación realizada. Si bien el programa fuente puede estar comentado, el cambio o la implementación realizada debe explicarse en el texto del informe (no es aceptable “ver código fuente” en el informe).
- Se debe entregar en formato electrónico tanto el informe como todo el código fuente usado/desarrollado.