Universidad Nacional
Facultad de Ciencias Exactas y Naturales
Escuela de Informática

EIF212 – Sistemas Operativos
Ing. Jose Pablo Calvo Suárez

Tarea Programada #2 – Valor 10%

Objetivos

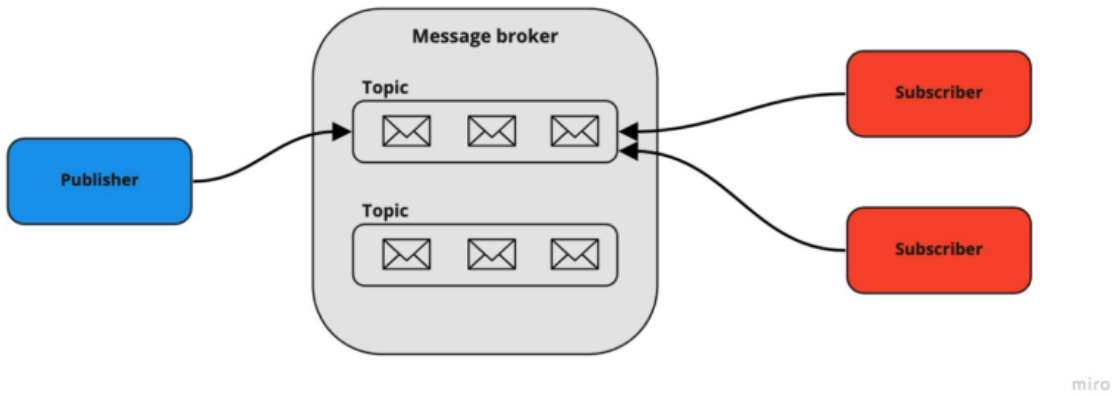
- Describir el uso de diversas API para creación y gestión de hilos
- Implementar técnicas de programación asociadas al uso de APIs del sistema
- Instalar y configurar distintas tecnologías y protocolos de red
- Implementar técnicas de comunicación y sincronización entre procesos e hilos

Descripción

El proyecto consiste en desarrollar una versión simplificada de un "**message broker**" utilizando el modelo **cliente/servidor** – es decir, la aplicación deberá permitir la comunicación en la red. La aplicación utilizará el estilo de mensajes "**publisher/subscriber**", en el cuál un proceso productor publica un mensaje en un determinado tema, mientras que uno o varios subscriptores a ese tema lo consumen. Para más información, pueden acceder al siguiente enlace:

https://www.ibm.com/topics/message-brokers

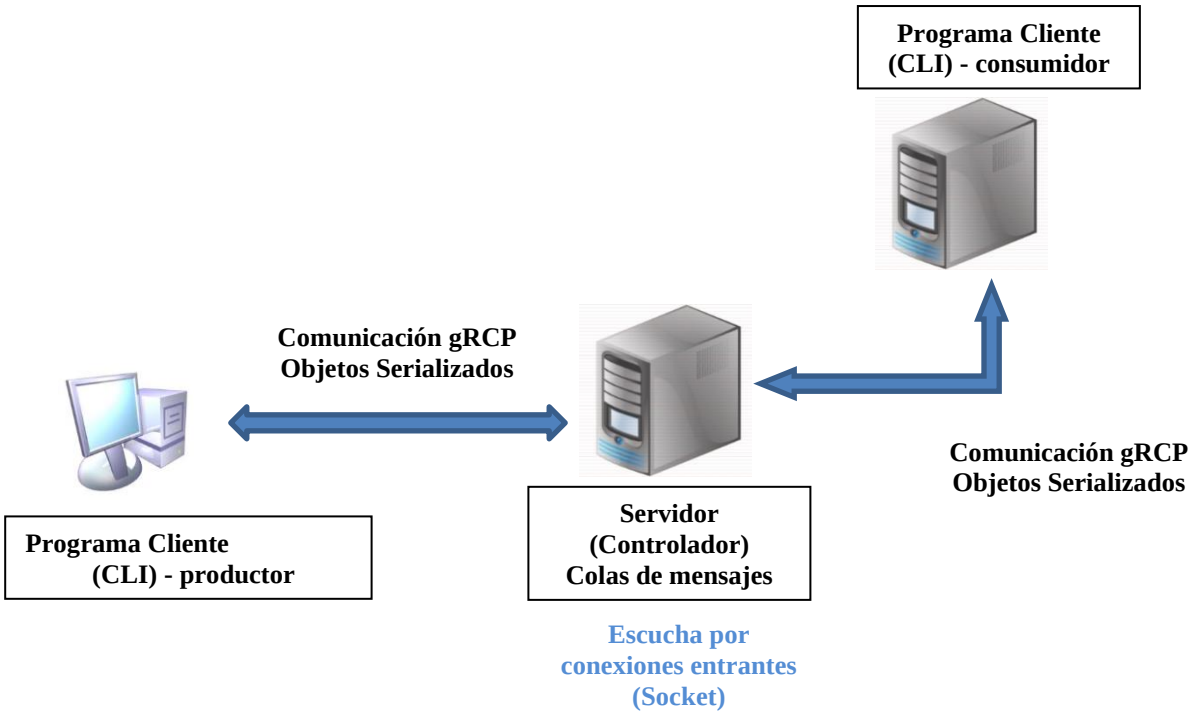
La solución deberá contar con dos programas: el **programa cliente**, que puede enviar mensajes (publicar en un tema) o recibir los mensajes de un tema al que se encuentre subscrito. Mientras que el **programa servidor** se encarga de administrar las conexiones de los clientes y recibir los datos para procesarlos en las colas de mensajes (insertar los datos, eliminar los datos de la cola, manejar la sincronización, etc.). Un diagrama simplificado de la aplicación puede verse de la siguiente manera:



Tome en cuenta que pueden existir múltiples productores/consumidores al mismo tiempo. Eso quiere decir que el servidor deberá crear una cola por tema (para efectos del proyecto, pueden asumir un máximo de tres temas de manera simultánea como máximo). Para la comunicación y el paso de mensajes entre el cliente y el servidor, utilizarán **gRPC**, que es un framework diseñado para el paso de información entre aplicaciones en la red. Para obtener más información de **gRPC**, visiten el siguiente enlace:

https://grpc.io/

En la siguiente imagen se muestra a grandes rasgos el proceso de comunicación entre los clientes y el servidor. Un productor envía datos hacia un tema en el servidor utilizando **gRCP**. El servidor toma los datos y los escribe en la cola correspondiente. Luego, el servidor envía los datos hacia los consumidores que se encuentren conectados en ese momento.



Consideraciones Adicionales

- Debido a que el servidor debe manejar múltiples conexiones concurrentes, se deben implementar hilos (un hilo para manejar cada conexión a cada cliente)

- Debe manejar adecuadamente las situaciones de error de conexión de la red (si un cliente deja de responder o hubo un problema en la red, no debería causar que la aplicación termine abruptamente)

- Implementen una adecuada sincronización de hilos/procesos – utilicen mutex locks, semáforos, variables de condición, etc. (Pueden utilizar más de un mecanismo de sincronización de ser necesario)

- El servidor debe escribir cada evento de la aplicación en un archivo de log (por ejemplo, cuando un nuevo cliente se conectó, cuando se recibió un nuevo mensaje en un tema, cuando se envió a un subscriptor, etc.). El formato del archivo de log debe ser el siguiente:

  <Fecha>:<hh:mm:ss> <Mensaje>

  Por ejemplo:

  30/05/2023:15:09:36 Nuevo cliente conectado al puerto 34000
  30/05/2023:15:10:25 Mensaje enviado al tema Sistemas Operativos

Nota: Ustedes deciden el contenido de los mensajes de log. Existe un único archivo para todos los eventos, así que deben implementar técnicas de sincronización

- Noten que existen diversas situaciones que pueden ocurrir con el **message broker**. Por ejemplo, ¿qué pasa si un cliente publica un mensaje en un tema, pero no existen subscriptores? ¿Se mantiene el mensaje en la cola por cierto tiempo, o hasta que algún subscriptor lo consuma? Para todos esos casos, ustedes diseñan la solución que consideren pertinente

- Las colas de mensajes son de tamaño finito, así que deben tomar en consideración que los consumidores no deben consumir de colas vacías, y los productores no deben insertar nuevos mensajes si la cola está llena (un buen escenario para el uso de semáforos)

- Los tipos de mensajes que puede soportar la aplicación son definidos por ustedes – **gRPC** soporta múltiples tipos de datos. Sin embargo, se recomienda trabajar con texto para simplificar el desarrollo del proyecto (pueden utilizar otros formatos como imágenes, si así lo desean)

Observaciones

- La tarea debe ser realizada en grupos de tres personas como máximo

- Se debe entregar el proyecto con el código fuente, ejecutables y librerías necesarias para su ejecución. Incluya un pequeño archivo .txt con las instrucciones para correr el programa

- Utilice el lenguaje de programación que desee (el único requerimiento es que soporte **gRPC**). Inclusive, podrían crear el cliente en un lenguaje de programación y el servidor en otro lenguaje

- La fecha límite para la entrega es el día domingo 9 de junio a las 11:59pm mediante el aula virtual

- En caso de copia (dos proyectos o más con mucha similitud que lo demuestren) o plagio (códigos descargados de Internet, libros o cualquier otro material), la nota de la tarea es 0

- Debe indicar todos los recursos consultados para le elaboración del proyecto. Pueden incluirlos como parte de los comentarios dentro del código fuente