Universidad Técnica Federico Santa María
Sistemas Distribuidos
Segundo Semestre 2020

Laboratorio 2

Profesora: Erika Rosas
Ayudantes de Tarea: Sebastián Alvarado **sebastian.alvaradoa@sansano.usm.cl**
Sebastián Ávalos **sebastian.avalosh@sansano.usm.cl**
Franco Zalavari **franco.zalavari@sansano.usm.cl**

4 de noviembre de 2020

1. Objetivos

- Familiarizarse con algoritmos de coordinación:
  - Exclusión mutua centralizada.
  - Exclusión mutua distribuida.
- Implementar y coordinar diferentes componentes de un sistema distribuido.

2. Introducción

La comunicación es primordial en la construcción de los sistemas distribuidos. Como se abordó en el primer laboratorio, constituye un principio fundamental para poder conectar servicios y desplegar arquitecturas que alimenten los sistemas a producir. En la medida que estos sistemas aumentan en complejidad se hace presente un nuevo problema, la coordinación, servicios que comparten recursos o que se distribuyen tareas, además de poder comunicarse, deben ser capaces de ponerse de acuerdo para trabajar en conjunto.

En este laboratorio se propone implementar un sistema en el cual se deba hacer uso de lo aprendido en la tarea anterior, utilizando el lenguaje de programación GO y la tecnología de comunicación de gRPC, además de los algoritmos vistos en clases como los asociados a exclusión mutua, tanto centralizados como distribuidos para coordinar el acceso a un recurso compartido.

3. Tecnologías

- El lenguaje de programación a utilizar es **Go**.
- Para la comunicación se utilizará **gRPC**.

4. Tarea

Una red de bibliotecas está experimentando con métodos de distribución de sus existencias digitales a sus suscriptores que no han podido acercarse a sus dependencias producto de las restricciones sanitarias. Dado lo anterior, han decidido digitalizar sus existencias e invertir en un sistema de distribución de libros

que garantice el acceso de estos a sus clientes. Dado lo anterior, han decidido contactar a los estudiantes del curso de Sistemas Distribuidos para que les presten soporte mediante sus conocimientos en el área.

Para tener una mejor distribución de sus recursos y accesos más rápidos, los encargados de la biblioteca han decidido fragmentar los libros y distribuirlos en un conjunto de nodos. Los nodos deben coordinarse para acceder a un log el cual registrará cada escritura realizada en el sistema. En ningún caso, más de un nodo debe acceder a este registro, por lo que han decidido pedirles a los alumnos de Sistemas Distribuidos que implementen dos algoritmos de exclusión mutua: El algoritmo centralizado y el algoritmo distribuido. Se les solicita que comparen los sistemas y les entreguen un informe con los resultados obtenidos.

5. Arquitectura del sistema

El sistema de biblioteca online tendrá que ser implementado como un sistema de archivos distribuidos. Para esto, se consideran 3 tipos de nodos. Estos nodos son: Cliente, DataNode y NameNode. Algunos de los componentes funcionarán de una manera distinta dependiendo del tipo de algoritmo de exclusión mutua que se esté implementando, sin embargo, compartirán ciertas características en ambas implementaciones.

5.1. Cliente

Son los encargados de cargar y descargar archivos del sistema, por ello se distinguen 2 tipos:

- Cliente Uploader: Se encarga de cargar los libros en la red, para ello los divide en chunks de tamaño **250 kB** los cuales son enviados a un DataNode en particular. Libros de dominio público pueden ser obtenidos desde `https://www.elejandria.com/coleccion/libros-llevados-al-cine`.
- Cliente Downloader: Se encarga de recuperar los libros de la red, se conecta al NameNode para obtener la ubicación de los chunks, los cuales descarga de los DataNodes que le son indicados y reconstruye el libro a partir de los chunks.

Pueden encontrar un tutorial de como descomponer un archivo en fragmentos de tamaño definido (chunks) y luego volver a reconstruirlo en `https://www.socketloop.com/tutorials/golang-recombine-chunked-files-example`.

5.2. DataNode

Son los encargados de almacenar los chunks que fueron generados por el cliente uploader. En primera instancia, uno de los DataNodes se encarga de distribuir los fragmentos entre sus pares, sin embargo, previo a eso debe producir una propuesta de como se repartirán las partes. Esta propuesta debe ser aprobada por los otros DataNodes o por el NameNode dependiendo el tipo de algoritmo que se esté implementando. De esta manera, en el NameNode quedará registro de la ubicación de todos los fragmentos (chunks) que componen cada libro para que sean recuperados posteriormente.

- Algoritmo Exclusión Mutua Distribuida:
  - El algoritmo a utilizar es el de Ricart y Agrawala.

- La propuesta de distribución debe ser aprobada por los otros nodos, si estos no la aprueban debe volver a plantearse una nueva propuesta.
- Si la propuesta es aceptada el nodo en cuestión realiza la documentación en el registro del NameNode.
- Si más de un nodo intenta escribir en el registro del NameNode puesto que este es un recurso compartido, deben resolver su conflicto siguiendo algoritmo de Ricart y Agrawala.

▪ Algoritmo Exclusión Mutua Centralizada:

- La propuesta de distribución debe ser aprobada por el NameNode, en caso de ser rechazada, debe volver a producirse una nueva propuesta.
- En caso de que la propuesta sea aceptada, el nodo ingresa la información pertinente en el registro del NameNode y distribuye los chunks.
- Si más de un nodo intenta acceder al registro del NameNode al mismo tiempo, este último debe encargarse de resolver el conflicto.

5.3. NameNode

Será el encargado de almacenar el LOG dentro del cual se registrarán los metadatos de las partes de cada archivo, además de mostrar el listado de libros disponibles al cliente.

Las tareas del NameNode dependerán principalmente del tipo de algoritmo que se está implementando, como se vio anteriormente, en la versión Distribuida este solo se encargará de mantener el registro de libros, mientras que para la versión centralizada deberá tener un papel de coordinador en caso de que se quiera realizar modificaciones de manera concurrente en el registro. Esto sumado a sus labores de intermedio entre la biblioteca y los usuarios que quieran acceder a algún libro.

5.3.1. Estructura del Log del NameNode

Para mantener un orden que permita conocer la ubicación de los chunks de cada libro, el registro del NameNode debe tener la siguiente estructura:

```
Nombre_Libro_1 Cantidad_Partes_1
parte_1_1 ip_maquina
...
parte_1_n ip_maquina
Nombre_Libro_2 Cantidad_Partes_2
parte_2_1 ip_maquina
...
parte_2_n ip_maquina
...
```

6. Informe

En el informe deben describir qué hicieron y cómo lo hicieron. Además deben presentar los resultados obtenidos, analizarlos y discutir sobre estos. Finalmente, deben presentar sus conclusiones. Las métricas que deben considerar para evaluar cada algoritmo son el número de mensajes enviados y el tiempo que demora en escribir en el log.

7. Diagramas de Secuencia del Problema

Para los diagramas que se muestran a continuación se considera una sola solicitud de un cliente único.

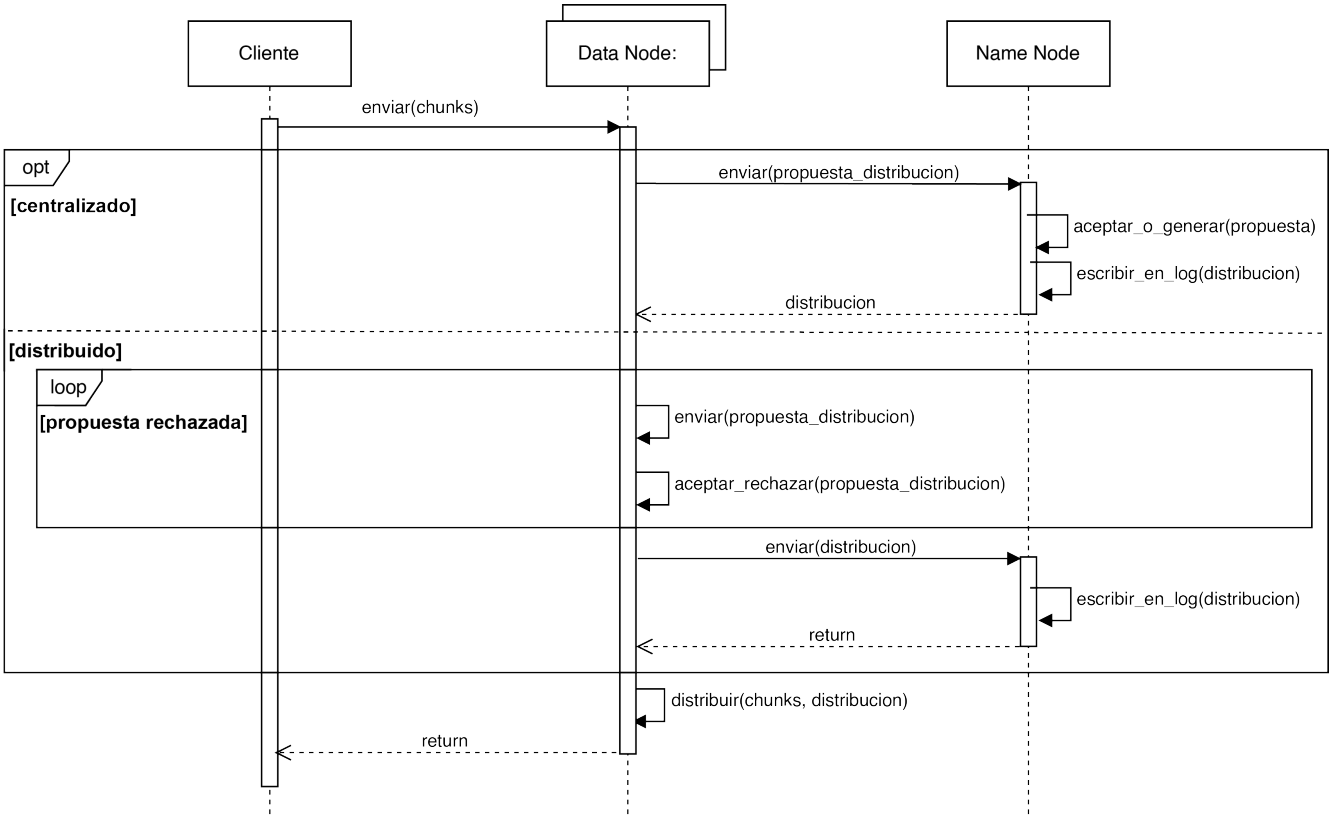

Figura 1: Diagrama de secuencia para el envío de un libro de un cliente uploader

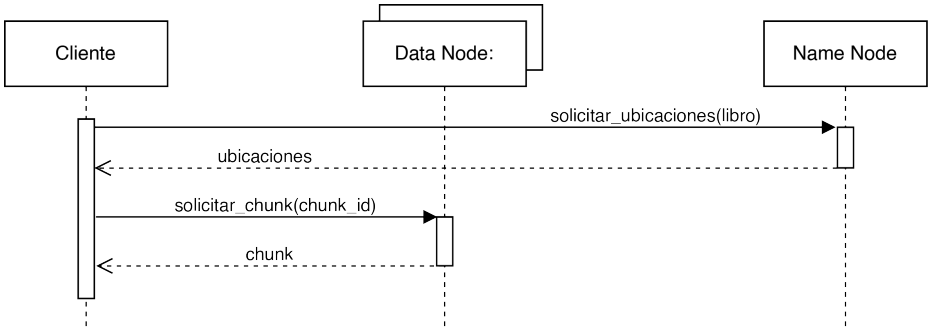

Figura 2: Diagrama de secuencia para la solicitud de un libro de un cliente downloader

8. Restricciones

- Todo uso de librerías externas que no se han mencionado en el enunciado debe ser consultado con los ayudantes.
- La distribución de las máquinas debe seguir el siguiente orden: 3 máquinas para DataNodes, 1 para NameNode, los clientes pueden estar en cualquiera de las 4 máquinas.
- El informe deberá tener una extensión **máxima** de 4 páginas.

9. Consideraciones

- Se realizará una ayudantía con el fin de presentar y resolver dudas de la tarea.
- Todas las consultas sobre la tarea se deben realizar en el foro de Moodle y se responderán sólo hasta dos días antes de la entrega de la tarea. Las respuestas a las consultas en fin de semana y después de las 18hrs queda a disponibilidad de los ayudantes. Si usted hace una pregunta un viernes en la noche puede no ser respondida hasta el lunes en la mañana.
- En caso de problemas particulares enviar un correo.
  sebastian.alvaradoa@sansano.usm.cl, **sebastian.avalosh@sansano.usm.cl** o **franco.zalavari@sansano.usm.cl** y con copia a **erosas@inf.utfsm.cl**
- Se fijará un horario para la revisión de la tarea.

10. Reglas de Entrega

- La tarea se entrega en **grupos de 2 personas**.
- La fecha de entrega es el día **2 de Diciembre a las 23:55**.
- La tarea se revisará en las máquinas virtuales, por lo que los archivos necesarios para la correcta ejecución de esta debe estar en ellas. Recuerde que el código debe estar identado, comentado, sin warnings y sin errores.

- Se aplicará un descuento de **5 puntos** al total de la nota por cada Warning, Error o Problema de Ejecución.
- Debe dejar un **MAKEFILE** o similar en cada máquina virtual asignada a su grupo para la ejecución de cada entidad. **Tareas sin MAKEFILE no se revisarán**.
- Debe dejar un **README** en cada máquina virtual asignada a su grupo con nombre y rol de cada integrante, además de la información necesaria para ejecutar los archivos. Si este no contiene la información pertinente para poder ejecutar su tarea, esta no será revisada.
- No se aceptan entregas que no puedan ser ejecutadas desde una consola de comandos. Incumplimiento de esta regla significa **nota 0**.
- En las máquinas virtuales deben encontrarse tanto la implementación del sistema como el informe solicitado. La ausencia de cualquiera de estos significará **nota 0**.
- Cada hora o fracción de atraso se penalizará con un descuento de **5 puntos**.
- Copias serán evaluadas con **nota 0** y serán notificadas a la profesora y las autoridades pertinentes.