# Distributed Scraper

**Marian S. Álvarez Suri - C412**

**Carlos A. Bresó Sotto - C412**

## 1. Arquitectura

Nuestro sistema distribuido tendrá una arquitectura dividida en tres componentes principales:

**- Nodos de Cola (Queues)**

Los nodos de cola son responsables de almacenar temporalmente las solicitudes de scraping hasta que un nodo de scrapper esté disponible para procesarlas. Implementaremos múltiples nodos de cola y usaremos replicación para asegurar que cada solicitud esté almacenada en al menos tres nodos distintos. Utilizaremos un sistema de monitoreo para verificar el estado de los nodos de cola y redistribuir las solicitudes en caso de fallos de los scrappers mediante temporizadores de visibilidad de la solicitud.

**- Nodos de Scrapper (Scrapper Nodes)**

Los nodos de scrapper se encargan de procesar las solicitudes de scraping y extraer la información requerida de las páginas web. Emplearemos un balanceador de carga para distribuir las tareas equitativamente entre los nodos de scrapper activos. En caso de fallo de un nodo scrapper, las colas se encargarán de reasignar las solicitudes a alguno de los nodos funcionales.

**- Nodos de Almacén (Storage Nodes)**

Los nodos de almacén estarán distribuidos en un anillo de CHORD y almacenarán los resultados del scraping. Los datos scrappeados serán almacenados utilizando hashing consistente para distribuirlos equitativamente entre los nodos de almacén.

**- Nodos de Coordinación (Coordination Nodes)**

Los nodos de coordinación serán los encargados de gestionar las solicitudes de los clientes hacia las colas y de enviar las respuestas una vez obtenidas.

**- Nodos de Monitoreo (Diagnose Nodes)**

Existirá un nodo de monitoreo principal que se encargará de verificar constantemente el estado en que se encuentran el resto de componentes del sistema y enviará indicaciones en caso de fallos.

## 2. Procesos

**- Tipos de Procesos dentro del Sistema**

En nuestro srapper distribuido podremos encontrar los siguientes tipos de procesos:

- Procesos de Coordinación: Gestionan solicitudes de los clientes desde y hacia el sistema distribuido.
- Procesos de Encolado: Manejan la recepción y distribución de solicitudes en las colas.
- Procesos de Scrapping: Ejecutan las tareas de scraping y procesamiento de datos.
- Procesos de Almacenamiento: Gestionan el almacenamiento y recuperación de datos scrappeados.
- Procesos de Monitoreo: Detectan y gestionan fallos en nodos y servicios.

**- Organización o Agrupación de los Procesos**

Nuestros procesos estarán agrupados en clusters de nodos, donde cada cluster maneja un tipo específico de proceso (colas, scraping, almacenamiento). Los procesos de coordinación y monitoreo se manejarán de forma independiente a los clusters de nodos.

**- Patrón de Diseño y Desempeño**

Para el diseño y desempeño del sistema, utilizaremos una combinación de hilos y procesos para balancear la carga y mejorar la eficiencia. Por ejemplo, el servicio de cola podría usar procesos para manejar diferentes solicitudes mientras usa hilos para procesar múltiples solicitudes en paralelo dentro de cada proceso.

## 3. Comunicación

La comunicación dentro del sistema de scrapper web distribuido se facilita a través de dos protocolos principales: gRPC y HTTP. Estos protocolos sirven para diferentes propósitos y se eligen según sus fortalezas y adecuación para tareas específicas.

### - gRPC para Comunicación Interna (Servidor - Servidor)

Utilizaremos gRPC para manejar tareas relacionadas con CHORD, como encontrar los nodos sucesores y predecesores, estabilizar la red y mantener el anillo de hashing consistente. Al utilizar gRPC sobre HTTP/2, el sistema se beneficiará de una comunicación de alto rendimiento entre servidores y baja latencia con una eficiente serialización y deserialización de mensajes. Este protocolo asegura que los nodos distribuidos en el anillo CHORD puedan comunicarse rápida y confiablemente entre sí, manteniendo la integridad y estabilidad de la red.

### - HTTP/REST para Comunicación Externa (Cliente - Servidor)

Para la comunicación entre el cliente y el servidor, utilizaremos el protocolo HTTP por ser un estándar ampliamente adoptado y compatible con la mayoría de los clientes. La separación de tareas relacionadas con CHORD y tareas específicas de los nodos entre gRPC y HTTP asegura que cada protocolo se utilice según sus fortalezas, optimizando la eficiencia general de la comunicación del sistema.

## 4. Coordinación

La coordinación dentro del sistema es crucial para asegurar que los nodos operen eficientemente y sin conflictos. Emplearemos un mecanismo de bloqueo distribuido para gestionar la **sincronización de acciones** y el **acceso exclusivo a recursos**. El sistema de bloqueo distribuido controlará el acceso a colas, evitando que dos nodos scrapper lean el mismo mensaje al mismo tiempo. Para ello, los mensajes se vuelven inaccesibles durante cierto intervalo de tiempo para el resto de scrappers, evitando el procesamiento duplicado de una misma solicitud. Si la solicitud de scrapper culminó exitosamente, el nodo scrapper envía un mensaje de ACK a la cola para eliminar definitivamente el request. Por otra parte, el proceso de coordinación de cada ciente se mantendrá en espera hasta que la solicitud sea atendida para enviar una respuesta.

## 5. Nombrado y Localización

Identificar, ubicar y localizar claramente los componentes del sistema de scrapper distribuido es esencial para su coordinación y gestión eficiente.

**- Colas**

- **Nombres y Etiquetas**: Asignar nombres y etiquetas únicas a cada cola.
- **Metadatos**: Añadir metadatos descriptivos como estado actual, cantidad de solicitudes, etc.

**- Scrappers**

- **Nombres de Nodo**: Usar nombres de nodo únicos.
- **Metadatos**: Añadir metadatos como estado actual, si se encuentra en proceso o no de una solicitud, etc.
- **Direcciones IP**: Registrar las direcciones IP o los nombres de host de cada nodo scrapper.

**- Almacenes**

- **Identificadores de Nodo del Anillo CHORD**: En un anillo CHORD, cada nodo de almacenamiento tiene un identificador único generado por el algoritmo de hashing consistente.
- **Direcciones IP/Nombres de Host**: Cada nodo de almacenamiento debe tener una dirección IP o nombre de host registrado.
- **Etiquetas y Categorías**: Clasificar los nodos según los tipos de datos que almacenan, como links, archivos o texto html.

**- Coordinador**

- **Servicios**: Identificar cada servicio que el coordinador ofrece.
- **EndPoints API**: Registrar los endpoints API del coordinador para que los scrappers y otros componentes puedan comunicarse con él.
- **UUIDs y Nombres de Servicio**: Asignar identificadores únicos a cada instancia del coordinador.
- **Direcciones IP/Nombres de Host**: Cada instancia debe tener una dirección IP o nombre de host único registrado en un servicio de descubrimiento.

**- Monitoreador**

- **Etiquetas de Métrica**: Clasifica las métricas monitoreadas con etiquetas.
- **UUIDs y Direcciones IP/Nombres de Host**: Identifica y asigna UUIDs y direcciones IP/nombres de host a cada instancia monitoreadora.

## 6. Consistencia y Replicación

- Distribución de los datos: El sistema empleará una estrategia de replicación para los nodos de almacenamiento, replicando datos para asegurar redundancia y tolerancia a fallos.
- Replicación: Cada dato se replica en el nodo responsable y en sus dos sucesores inmediatos en el anillo.
- Confiabilidad de las réplicas de los datos tras una actualización: Cuando se escribe o actualiza un dato, las réplicas deben ser actualizadas simultáneamente en los sucesores. Siempre que el nodo responsable presente fallas, uno de los sucesores puede ofrecer acceso al dato, manteniendo su disponibilidad hasta que el sistema recupere el nodo fallido. Esto requiere monitoreo y actualizaciones constantes para asegurar que los datos y sus réplicas permanezcan sincronizados.

## 7. Tolerancia a fallas

- Tolerancia a Fallos de Colas: La tolerancia a fallos de las colas se logra a través del sistema de bloqueo distribuido y de la replicación. Como cada solicitud se encuentra almacenada en 3 colas diferentes, se garantiza que si se pierden dos colas, los mensajes aun se encuentren almacenados en alguna otra.

- Tolerancia a Fallos de Scrappers: Si un nodo scrapper falla antes de completar su tarea, el request eventualmente se volverá visible nuevamente en la cola y podrá ser procesado por otro scrapper. Este enfoque asegura que los mensajes no se pierdan debido a fallos de nodos scrappers.

- Tolerancia a Fallos de Almacenes: La tolerancia a fallos de los almacenes se logra a través de la replicación de datos. Cada dato se replica en los dos sucesores inmediatos en el anillo del nodo de almacenamiento responsable de los datos, asegurando que al menos otros dos nodos tengan una copia. Esta estrategia de replicación asegura que los datos permanezcan disponibles incluso si dos nodos de almacenamiento adyacentes en el anillo fallan.

- Tolerancia a Fallos de los Nodos de Monitoreo y Coordinación: Existirán tres instancias diferentes de cada uno de estos nodos. Se mantendrá activa simultánemante solo una instancia, que será la principal, y las dos secundarias se quedarán en espera en caso de que exista algún fallo en la instancia principal. En ese caso, una de las dos instancias secundarias toma el control y pasa a ser la principal.