Informe

1. Arquitectura: Problemas en el diseño del sistema

Organización del sistema distribuido El sistema se organiza bajo la arquitectura Chord, que utiliza tablas de enrutamiento distribuidas para garantizar un acceso eficiente a los datos. Cada nodo posee un identificador único generado mediante un hash, lo que permite asignar datos de manera equitativa entre los nodos. Esta disposición asegura que las consultas puedan resolverse en un número logarítmico de pasos en relación con el tamaño del sistema, lo que favorece la escalabilidad.

Roles del sistema El sistema define dos roles principales:

1. **Clientes**:
   - Los clientes suben archivos calculando un identificador basado en un hash del contenido del archivo.
   - Realizan búsquedas de archivos mediante consultas en el sistema, que se resuelven explorando los nodos en busca de coincidencias en las tablas de metadatos.
   - Descargan archivos desde el nodo correspondiente responsable de almacenarlos.
2. **Servidores**:
   - Almacenan archivos junto con sus metadatos y gestionan la pertenencia de los nodos al anillo.
   - Coordinan la replicación y garantizan la continuidad de los servicios mediante tareas de mantenimiento de la estructura del anillo.

Distribución de servicios en redes Docker Se implementan redes Docker separadas para clientes y servidores. Los clientes se comunican a través de un router centralizado, mientras que los servidores conforman una red independiente donde mantienen el anillo lógico de Chord.

---

2. Procesos: Cantidad y organización de programas

Tipos de procesos dentro del sistema

- **Cliente**: Ejecuta solicitudes como subir, buscar y descargar archivos. Cada solicitud se gestiona mediante hilos específicos que manejan la comunicación, el procesamiento de datos y la respuesta.
- **Servidor**: Gestiona tareas de almacenamiento, replicación, búsqueda y comunicación interna entre nodos.

Organización de los procesos Los procesos están organizados en grupos funcionales:

1. Hilo principal: Coordina la inicialización y supervisión del nodo.

2. Hilos para solicitudes de clientes: Manejan tareas específicas de interacción con los clientes.
3. Hilos para comunicación entre nodos: Gestionan el intercambio de información y el mantenimiento del anillo.
4. Hilos de mantenimiento: Ejecutan funciones periódicas como estabilización y actualización de las tablas de enrutamiento.

Patrón de diseño utilizado El sistema utiliza multithreading para manejar múltiples solicitudes simultáneamente, lo que permite un procesamiento concurrente sin bloquear operaciones críticas. Este enfoque, combinado con patrones asíncronos, garantiza una alta capacidad de respuesta.

---

3. Comunicación: Envío de información en la red

Tipo de comunicación La comunicación en el sistema se basa en un enfoque de colas de mensajes utilizando ZeroMQ, empleando patrones como Request-Reply y Publish-Subscribe para garantizar eficiencia y flexibilidad.

Comunicación cliente-servidor Los clientes envían solicitudes utilizando un patrón Request-Reply. Este diseño asegura que cada mensaje de solicitud tenga una respuesta directa del servidor correspondiente, minimizando la latencia y garantizando una interacción clara.

Comunicación servidor-servidor Entre nodos, la comunicación se basa en la transferencia de mensajes para tareas como la verificación de datos, la actualización de tablas de enrutamiento y la coordinación de replicación. El patrón Publish-Subscribe facilita las actualizaciones globales necesarias para mantener la consistencia del anillo.

Comunicación entre procesos Dentro de los nodos, los procesos utilizan mecanismos sincronizados como locks para gestionar el acceso concurrente a recursos compartidos, previniendo condiciones de carrera y manteniendo la integridad del sistema.

---

4. Coordinación: Sincronización y toma de decisiones

Sincronización de acciones El diseño del sistema asegura que las operaciones críticas estén sincronizadas a nivel local mediante estructuras de control como semáforos y locks. Esto permite que las tareas se realicen sin conflictos, incluso en un entorno distribuido.

Acceso exclusivo a recursos Para evitar conflictos en la modificación de datos compartidos, el sistema utiliza bloqueos exclusivos que garantizan que un único proceso tenga acceso a los recursos en un momento dado.

Toma de decisiones distribuidas Las decisiones clave, como la reasignación de datos y el manejo de fallos, se realizan de forma autónoma por los nodos individuales, siguiendo algoritmos distribuidos predefinidos que garantizan consistencia y eficiencia.

---

5. Nombrado y localización: Identificación y acceso a recursos

Identificación de los datos y servicios Los archivos se identifican mediante un hash único derivado de su contenido, mientras que los nodos del sistema se identifican por un hash basado en su dirección IP.

Ubicación de los datos y servicios Los datos se asignan al nodo cuyo identificador sea el sucesor inmediato del hash del archivo, lo que asegura una distribución uniforme y predecible.

Localización de los datos y servicios El protocolo Chord permite localizar datos mediante consultas distribuidas, utilizando tablas de enrutamiento que reducen significativamente la cantidad de saltos necesarios para encontrar un recurso.

---

6. Consistencia y replicación: Manejo de copias de datos

Distribución de los datos El sistema distribuye los datos de manera uniforme entre los nodos, utilizando funciones de hash que minimizan colisiones y aseguran una carga balanceada.

Replicación y cantidad de réplicas Cada archivo tiene varias réplicas almacenadas en los sucesores inmediatos al nodo principal responsable. Este esquema proporciona alta disponibilidad y resistencia frente a fallos.

Confiabilidad de las réplicas Se emplean mecanismos de monitorización continua, como heartbeats, para verificar el estado de los nodos y garantizar que las réplicas se mantengan actualizadas y accesibles en caso de fallos.

---

7. Tolerancia a fallas: Respuesta a errores

Respuesta a errores Cuando un nodo falla, sus sucesores inmediatos asumen temporalmente sus responsabilidades, garantizando la continuidad del servicio.

Nivel de tolerancia a fallos esperado La capacidad de recuperación del sistema depende del número de réplicas configuradas y de la velocidad de detección de fallos. Un mayor número de réplicas incrementa la disponibilidad a cambio de un mayor uso de recursos.

Fallos parciales

- **Nodos caídos temporalmente**: Se marcan como inactivos hasta su reintegración, momento en el cual actualizan sus datos para sincronizarse con el estado actual del sistema.
- **Nodos nuevos**: Al unirse, reciben las réplicas y datos relevantes de acuerdo con su posición en el anillo, garantizando una integración eficiente.