



SISTEMAS DISTRIBUTIVOS

# PROTOCOLOS DE ACTUALIZACIÓN DE RÉPLICAS

## PROTOCOS DE ACTUALIZACIÓN DE RÉPLICAS


En un sistema distribuido que emplea replicación, no basta con tener copias de los datos; es fundamental asegurar que las actualizaciones se propaguen correctamente entre todas las réplicas para preservar la coherencia y la funcionalidad del sistema. Este proceso se rige por los llamados protocolos de actualización de réplicas, los cuales determinan cómo, cuándo y quién aplica los cambios, definiendo así las reglas del juego para mantener el sistema sincronizado, disponible y confiable (Robledo Sosa, 2002).

Estos protocolos son esenciales para evitar inconsistencias, resol

ver conflictos, y permitir que las aplicaciones distribuidas operen correctamente aun en presencia de concurrencia, latencia de red o fallos parciales.

### Actualización primaria (Primary-Backup)

Uno de los esquemas más comunes es el modelo Primary-Backup, también conocido como actualización centralizada. En este protocolo, una única réplica es designada como nodo primario (o líder), y todas las actualizaciones deben pasar por él. Una vez que la operación se ejecuta exitosamente en el primario, se propaga los cambios a las réplicas secundarias (Arboledas Brihuega, 2015).

 **Ejemplo.** Un sistema de base de datos con una instancia primaria en un centro de datos principal (por ejemplo, Frankfurt) y varias réplicas en otras regiones. Las escrituras se realizan solo en Frankfurt y luego se sincronizan con los demás nodos.

#### Ventajas:

- Simplifica la coordinación y evita conflictos.
- Alto control sobre el orden de las operaciones.

#### Desventajas:

- El primario puede convertirse en un cuello de botella.
- Si el nodo primario falla, se requiere un proceso de reelección.

### Protocolos basados en quorum

Este enfoque, inspirado en técnicas de votación, requiere que un subconjunto mínimo de réplicas llamado quorum participe en cada operación. Se establecen dos valores:

- $Q_{\text{r}}$ : número de réplicas necesarias para una lectura.
- $Q_{\text{w}}$ : número de réplicas necesarias para una escritura.

La condición clave es que:

$$Q_{\text{r}} + Q_{\text{w}} > N$$

donde  $N$  es el número total de réplicas. Esto garantiza que al menos una réplica común participe en ambas operaciones.

#### Ejemplo. En un sistema con 5 réplicas:

- $Q_{w} = 3$  para escribir (la mayoría debe aceptar).
- $Q_{r} = 3$  para leer (también mayoría).

Así, siempre habrá al menos una réplica con información actualizada.

Ventajas:

- Mayor tolerancia a fallos.
- Equilibrio entre consistencia y disponibilidad.

Desventajas:

- Lecturas y escrituras pueden ser más lentas.
- Complejidad en el manejo de quorum dinámico.

#### Protocolos de difusión (Gossip protocols)

Inspirados en la propagación de rumores, los protocolos de tipo gossip, permiten que cada réplica informe a otras sobre las actualizaciones que ha recibido, de manera aleatoria o periódica. Con el tiempo, toda la red de réplicas converge al mismo estado (Arboledas Brihuega, 2015).

Este enfoque es especialmente útil en sistemas muy grandes o geográficamente dispersos, donde no es práctico contactar a todas las réplicas simultáneamente.

- 📌 **Ejemplo.** En un sistema P2P de distribución de archivos, cuando un nodo recibe una nueva versión de un archivo, comunica esa novedad a sus vecinos, quienes a su vez replican el mensaje.

Ventajas:

- Alta escalabilidad.
- Tolerancia natural a fallos y particiones.

Desventajas:

- La convergencia puede tomar tiempo.
- No garantiza consistencia inmediata.

## Elección del protocolo adecuado

La elección de un protocolo de actualización depende de múltiples factores:

**Tabla 1.** Factores

Criterio	Primary-Backup	Quorum	Gossip
Consistencia.	Alta.	Configurable.	Eventual.
Rendimiento.	Alto (si primario fuerte).	Medio-alto.	Alto en entornos amplios.
Escalabilidad.	Limitada.	Media.	Muy alta.
Tolerancia a fallos.	Media.	Alta.	Alta.
Complejidad de implementación.	Baja.	Media-alta.	Media.