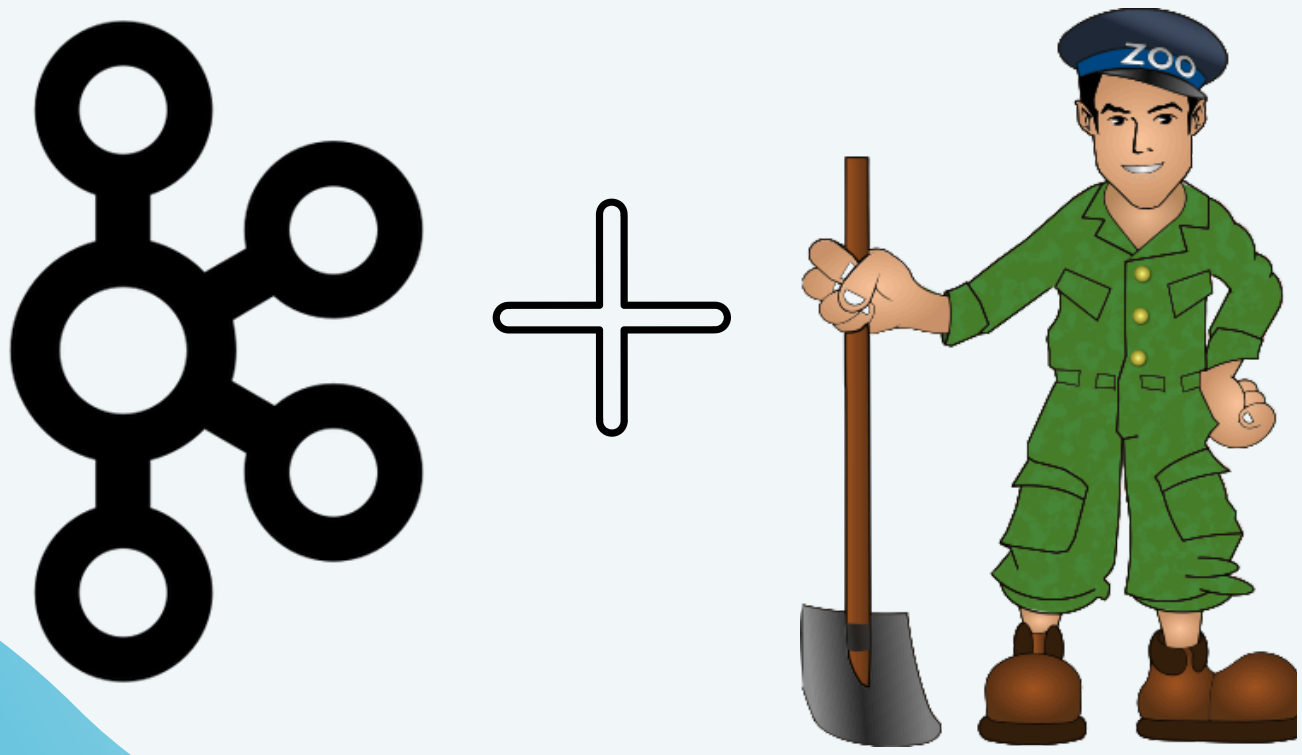
KAFKA + ZOOKEEPER

De 0 a expert@



PABLO DEL ÁLAMO

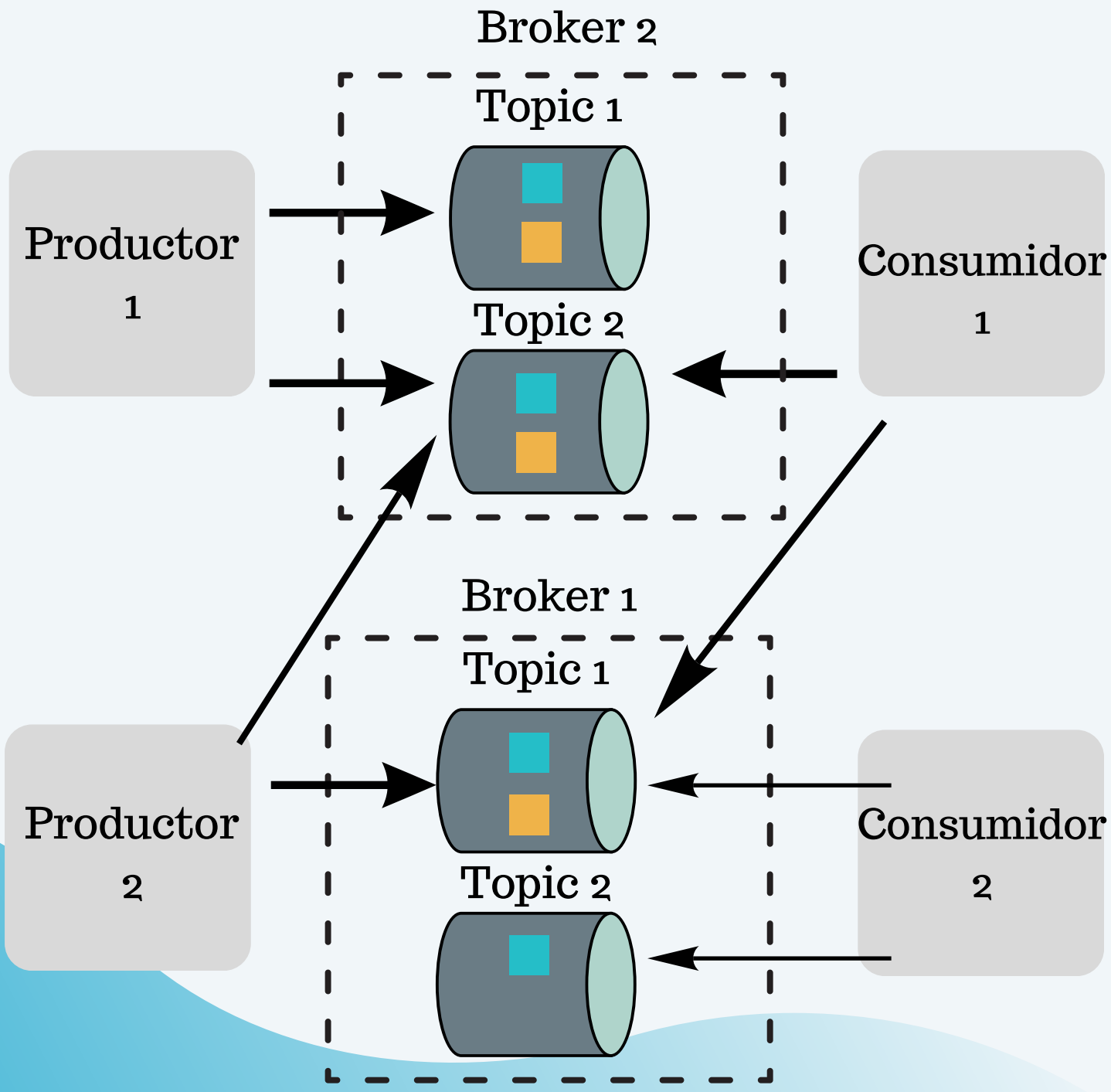
¿Qué es Kafka?

Kafka es una plataforma de mensajería distribuida que permite manejar flujos de datos en tiempo real entre diferentes sistemas.

🔑 Punto clave: Kafka distribuye grandes volúmenes de datos en tiempo real a través de "topics" (temas).

PABLO DEL ÁLAMO

¿Qué es Kafka?



PABLO DEL ÁLAMO

¿Qué es Kafka?

> Básicamente permite que ciertos productores manden mensajes a unos topics.
>
> Los consumidores se suscriben a los topics que quieran, y consumen los datos de manera asíncrona.

PABLO DEL ÁLAMO

Historia de Kafka

> Kafka fue desarrollado originalmente por LinkedIn para manejar los datos de sus usuarios en tiempo real.
>
> Más tarde, lo liberaron como un proyecto de código abierto bajo Apache.

PABLO DEL ÁLAMO

¿Por qué Kafka es tan popular?

- Velocidad: Maneja millones de mensajes por segundo.
- Escalabilidad: Kafka crece con tus necesidades.
- Resiliencia: Los datos están siempre disponibles, incluso si falla un servidor.

PABLO DEL ÁLAMO

Casos de uso de Kafka

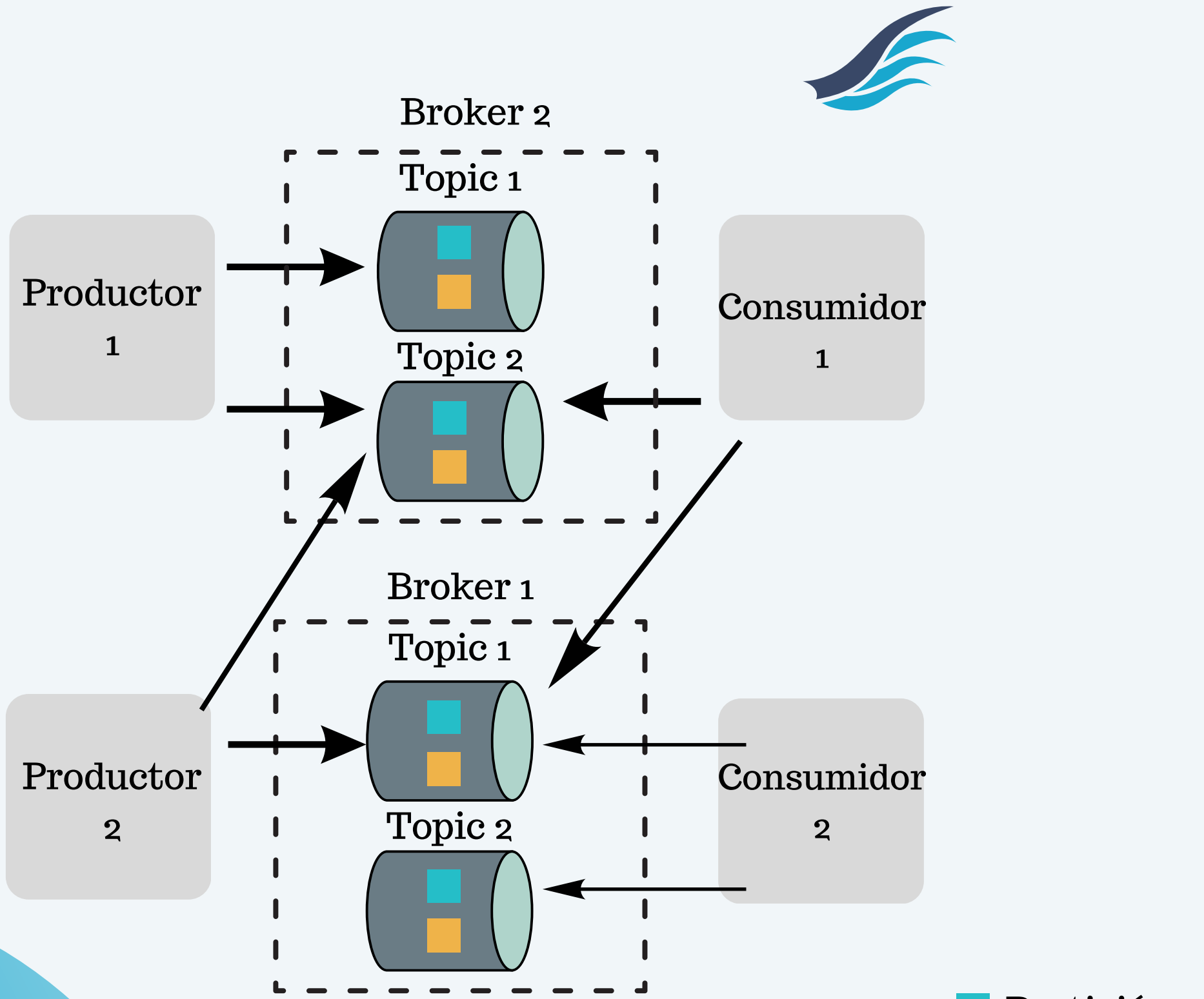
- Procesamiento de eventos en tiempo real (como transacciones bancarias)
- Seguimiento de actividades de usuarios en grandes plataformas como Netflix o LinkedIn
- Monitoreo de infraestructuras y sistemas

PABLO DEL ÁLAMO

Componentes Clave de Kafka

- Producer: Envía mensajes a Kafka.
- Consumer: Recibe los mensajes.
- Broker: Servidores que gestionan los mensajes
- Topics: Los temas que agrupan mensajes.
- Particiones: Una partición es una subdivisión de un topic que permite el procesamiento paralelo.

PABLO DEL ÁLAMO



PABLO DEL ÁLAMO

¿Qué son los Topics y Particiones?

- Topic: Es un canal o categoría donde se publican los mensajes.

- Particiones: Dividen los topics para gestionar los datos de manera paralela.

PABLO DEL ÁLAMO

Ejemplo Práctico de Topics y Particiones

Imagina un sistema de compras online:

- Topic: "Pedidos"
- Particiones: 1 partición para pedidos de ropa, 1 para electrónica, 1 para comestibles.

PABLO DEL ÁLAMO

¿Qué Son los Producers y Consumers en Kafka?

- Producers: Aplicaciones que envían datos (mensajes) a Kafka.

- Consumers: Aplicaciones que reciben esos datos.

Ejemplo: Una app móvil puede ser un producer, enviando información sobre el uso de la app. Un servidor puede ser un consumer, analizando esos datos.

PABLO DEL ÁLAMO

Brokers de Kafka: El Almacén de Datos

Un broker en Kafka es un servidor dentro de un clúster de Kafka que se encarga de almacenar, gestionar y distribuir los mensajes.

Los brokers reciben mensajes de los producers (productores de datos) y los guardan en particiones asociadas a un topic.

Además, sirven estos mensajes a los consumers (consumidores de datos) cuando los solicitan.

PABLO DEL ÁLAMO

¿Qué es Zookeeper?

Zookeeper es un sistema que actúa como el coordinador de Kafka.

Garantiza que los brokers de Kafka funcionen en armonía y mantengan la consistencia de los datos.

PABLO DEL ÁLAMO

¿Por qué Kafka necesita Zookeeper?

- Coordinación: Zookeeper organiza los brokers de Kafka y asegura que estén sincronizados.

- Gestión de Particiones: Decide qué broker maneja cada partición.

- Tolerancia a fallos: Si un broker falla, Zookeeper reasigna sus responsabilidades.

PABLO DEL ÁLAMO

El Papel de Zookeeper en Kafka

Zookeeper monitorea el estado de los brokers, coordina la replicación de particiones y asegura que los datos se procesen sin problemas.

Sin Zookeeper, Kafka no puede funcionar correctamente.

PABLO DEL ÁLAMO

¿Cómo Funciona la Replicación en Kafka?

Kafka replica los datos para asegurar disponibilidad.

Cada topic y partición tiene una réplica en otro broker.

Ejemplo: Si un broker cae, la réplica en otro broker se activa automáticamente.

PABLO DEL ÁLAMO

Coordinación de Réplicas por Zookeeper

Zookeeper se asegura de que las réplicas estén siempre sincronizadas.

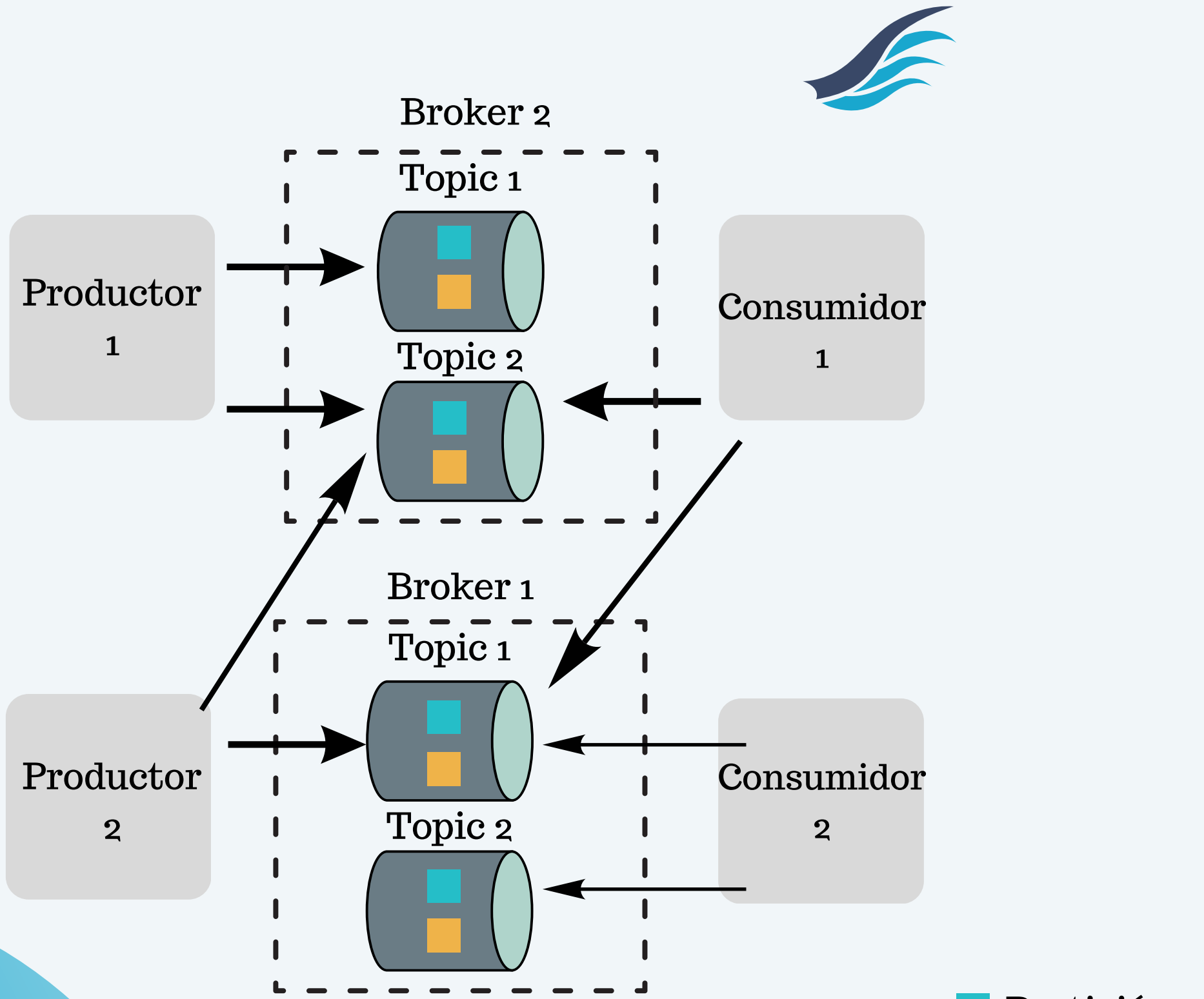
Si un broker principal falla, Zookeeper elige una réplica para que tome su lugar.

PABLO DEL ÁLAMO

¿Cómo un Producer Envía Mensajes a Kafka?

- El producer crea un mensaje.
- Kafka lo distribuye en particiones.
- El mensaje se almacena en un broker hasta que un consumer lo lea.

PABLO DEL ÁLAMO



PABLO DEL ÁLAMO

¿Cómo un Consumer Lee los Mensajes?

El consumer se suscribe a un topic y lee los mensajes de las particiones asignadas.

Kafka asegura que los mensajes sean procesados sin perder ninguno.

PABLO DEL ÁLAMO

Cómo Zookeeper Asigna Particiones a los Consumers

Zookeeper coordina qué partición será procesada por cada consumer, asegurando que no se dupliquen esfuerzos.

PABLO DEL ÁLAMO

Garantías de Entrega en Kafka

Kafka asegura que los mensajes se entreguen al menos una vez a los consumers.

Los parámetros como acks (confirmaciones de entrega) y retention policy aseguran la integridad del mensaje.

PABLO DEL ÁLAMO

Kafka Manager y Herramientas de Monitoreo

Para monitorear Kafka, puedes usar:

- Kafka Manager
- Confluent Control Center
- Prometheus + Grafana

PABLO DEL ÁLAMO

Kafka Streams: Procesamiento de Datos en Tiempo Real

Kafka Streams permite el procesamiento de los datos directamente dentro de Kafka.

Puedes realizar operaciones como filtros, agregaciones y uniones sin necesidad de mover los datos a otro sistema.

PABLO DEL ÁLAMO

Buenas Prácticas para Usar Kafka

- Configura la retención de datos según el caso de uso.
- Usa particiones para dividir la carga.
- Monitorea el rendimiento y la latencia de los brokers.

PABLO DEL ÁLAMO

Kafka en el Mundo Real

Kafka es utilizado por empresas como Netflix, Uber y LinkedIn para manejar sus datos en tiempo real.

Su capacidad para procesar grandes volúmenes de datos lo hace ideal para sistemas distribuidos.

PABLO DEL ÁLAMO

Ejemplo completo

Imagina que trabajas en una plataforma de redes sociales como Twitter. Todos los días, millones de usuarios publican mensajes (tweets) en tiempo real.

Estos mensajes necesitan ser procesados y almacenados rápidamente para que otros usuarios puedan verlos.

Kafka entra en juego para manejar este flujo masivo de datos. Aquí es donde los brokers y particiones ayudan a escalar la solución.

PABLO DEL ÁLAMO

Topic: TweetsPublicados

En este caso, tienes un topic llamado TweetsPublicados que almacena todos los mensajes que publican los usuarios. Este topic no está dividido por categoría de mensaje, sino que todos los brokers manejan el mismo tipo de mensajes (los tweets).

Kafka divide este topic en varias particiones. Cada partición contiene una parte de los tweets publicados. Esto se hace para distribuir la carga entre los brokers y permitir un procesamiento más rápido.

PABLO DEL ÁLAMO

Por ejemplo, Kafka podría dividir los mensajes así:

- Partición 1: Contiene los tweets publicados entre las 12:00 PM y 12:01 PM.
- Partición 2: Contiene los tweets publicados entre las 12:01 PM y 12:02 PM.
- Partición 3: Contiene los tweets publicados entre las 12:02 PM y 12:03 PM.

PABLO DEL ÁLAMO

> Tienes tres brokers que manejan las particiones del topic TweetsPublicados.
>
> Aquí, los brokers no manejan diferentes tipos de datos, sino que cada uno almacena las particiones y distribuye la carga.

PABLO DEL ÁLAMO

- Broker 1: Almacena la Partición 1 y tiene una réplica de la Partición 2.

- Broker 2: Almacena la Partición 2 y tiene una réplica de la Partición 3.

- Broker 3: Almacena la Partición 3 y tiene una réplica de la Partición 1.

PABLO DEL ÁLAMO

En este caso, todos los brokers manejan el mismo tipo de datos (los tweets).

Lo que Kafka hace es distribuir las particiones entre los brokers para que:

Escalabilidad Horizontal: Los brokers puedan procesar los datos en paralelo. En vez de que un solo broker procese millones de tweets por minuto, varios brokers manejan porciones de ese tráfico, acelerando el procesamiento.

Disponibilidad: Si el Broker 1 falla, la Partición 1 todavía está disponible en el Broker 3 gracias a la replicación. Así, los usuarios no experimentan pérdida de datos ni de servicio.

PABLO DEL ÁLAMO

Cada vez que un usuario publica un tweet:

- El Producer (la aplicación de Twitter) envía el tweet a Kafka.

- Kafka distribuye ese tweet en una de las particiones del topic TweetsPublicados. Por ejemplo, si el tweet fue publicado a las 12:01 PM, se almacenará en la Partición 2.

- El Broker 2, que maneja la Partición 2, almacena el tweet y replica el mensaje en el Broker 1 para asegurar disponibilidad.

PABLO DEL ÁLAMO

> Si un Consumer (como la API de Twitter que muestra tweets a los usuarios) quiere leer los tweets publicados, Kafka le asigna las particiones de manera equilibrada para que pueda procesar y entregar los mensajes rápidamente.

PABLO DEL ÁLAMO

Ejemplo GitHub

A continuación te dejo un ejemplo de un Kafka simple con un Broker + 1 productor + 1 consumidor + 1 topic con una sola partición, para que puedas verlo en funcionamiento y practicar con ello ajustando cosas (el enlace a github también lo tienes en la parte superior de mi perfil):

https://github.com/pdelalamo/kafka-java-example

PABLO DEL ÁLAMO

¿Te ha resultado útil?

- Comparte esta guía con tu equipo o amigos desarrolladores.
- Guárdala para tenerla siempre a mano.
- ¡Dale un like o comenta si tienes preguntas!

PABLO DEL ÁLAMO