Introduction to Apache Kafka And Real-Time ETL

for DBAs and others who are interested in new ways of working with relational databases

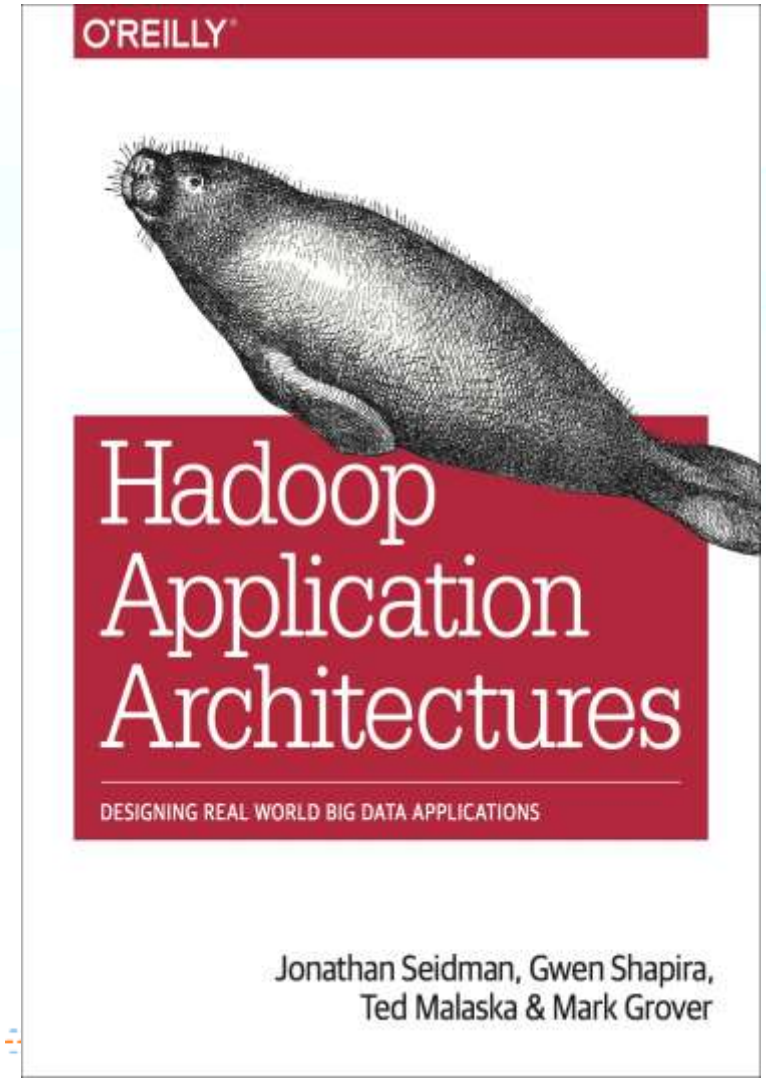
About Myself

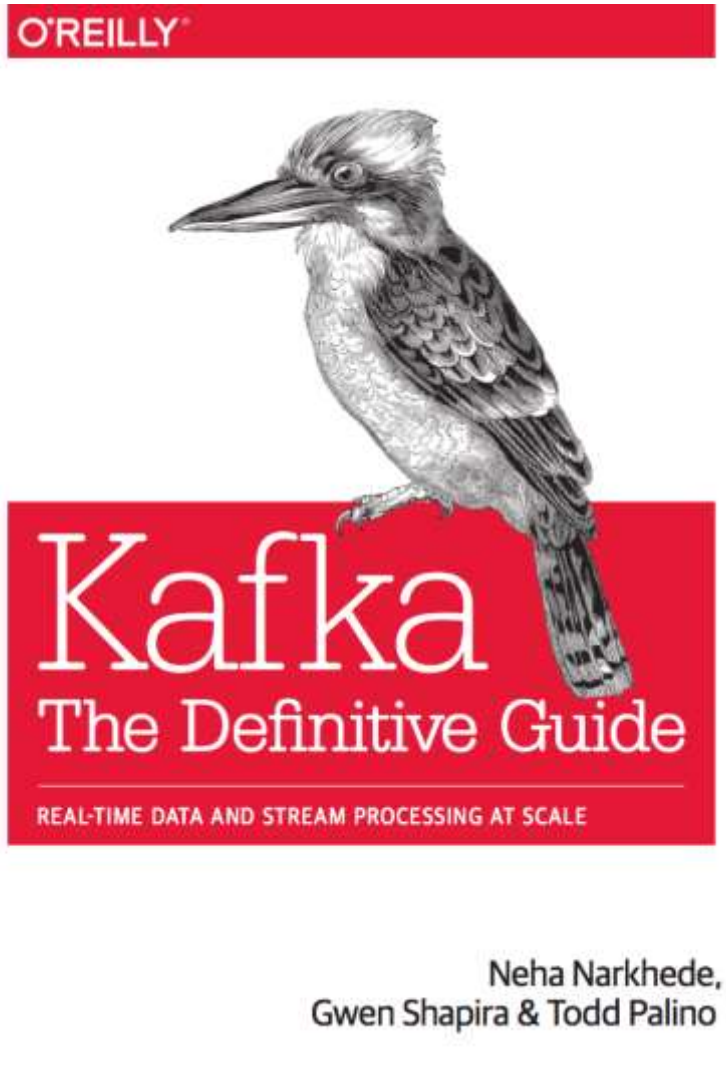
- Gwen Shapira – System Architect @Confluent
- Committer @ Apache Kafka, Apache Sqoop
- Author of “Hadoop Application Architectures”, “Kafka – The Definitive Guide”

- Previously:
  - Software Engineer @ Cloudera
  - Oracle ACE Director
  - Senior Consultants @ Pythian
  - DBA @ Mercury Interactive

- Find me:
  - gwen@confluent.io
  - @gwenshap

There's a Book on That!





An Optical Illusion

Apache Kafka is

publish-subscribe messaging

rethought as a

distributed commit log.

turned into

a stream data platform

confluent

We'll talk about:

- Write-ahead Logs
- So What is Kafka?
- Awesome use-case for Kafka
- Data streams and real-time ETL
- Where can you learn more

Write-Ahead Logging (WAL)

a standard method for ensuring data integrity… changes to data files … must be written only after those changes have been logged… in the event of a crash we will be able to recover the database using the log.

Important Point

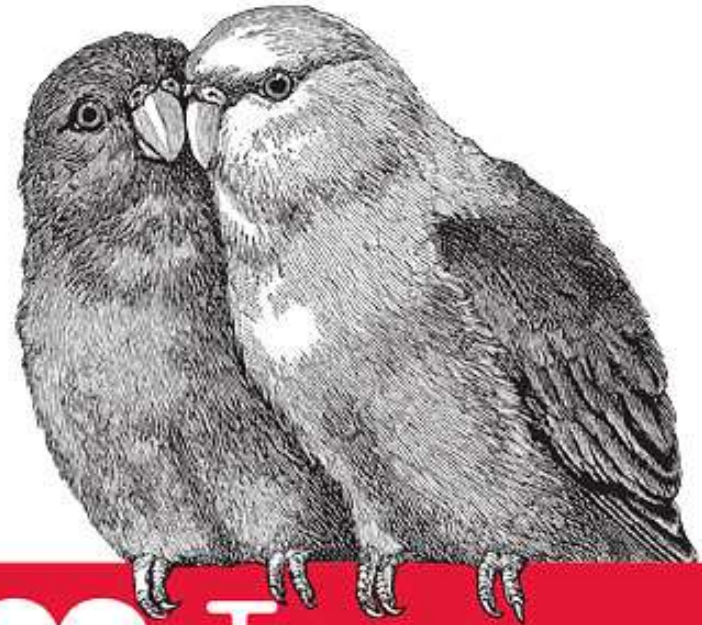
The write-ahead log is the only reliable source of information about current state of the database.

WAL is used for

- Recover consistent state of a database
- Replicate the database (Streaming Replication, Hot Standby)

If you look far enough into archived logs – you can reconstruct the entire database.

O'REILLY



I ♥ Logs

EVENT DATA, STREAM PROCESSING, AND DATA INTEGRATION

Jay Kreps

confluent

That’s nice, but what is Kafka?

Kafka provides a fast, distributed, highly scalable, highly available, publish-subscribe messaging system.

Based on the tried and true log structure.

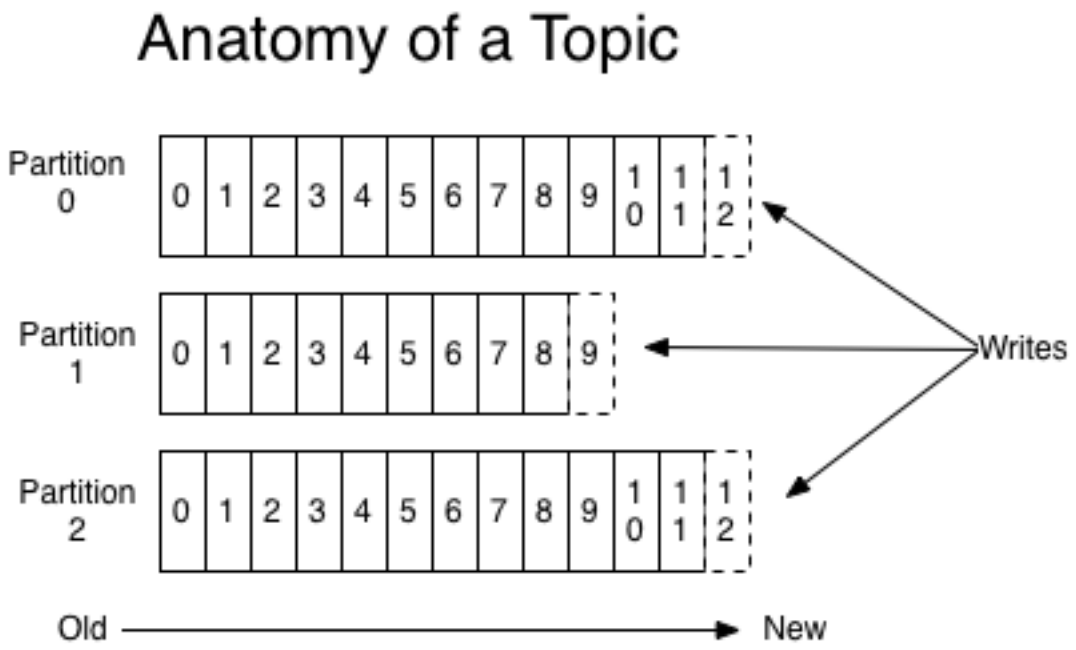
In turn this solves part of a much harder problem:

Communication and integration between components of large software systems

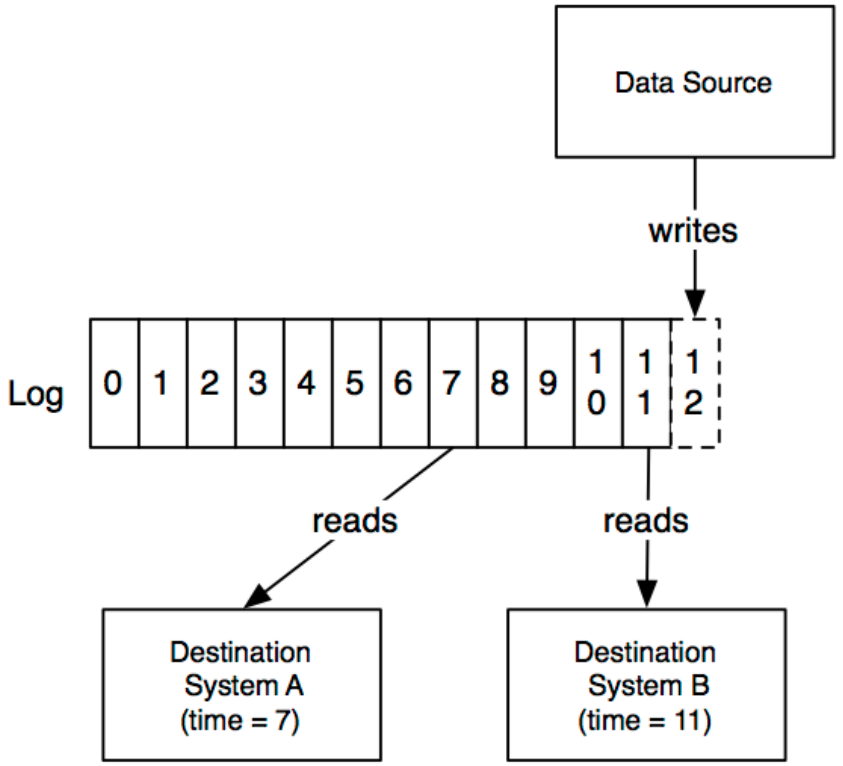
The Basics

- Messages are organized into **topics**
- **Producers** push messages
- **Consumers** pull messages
- Kafka runs in a cluster. Nodes are called **brokers**

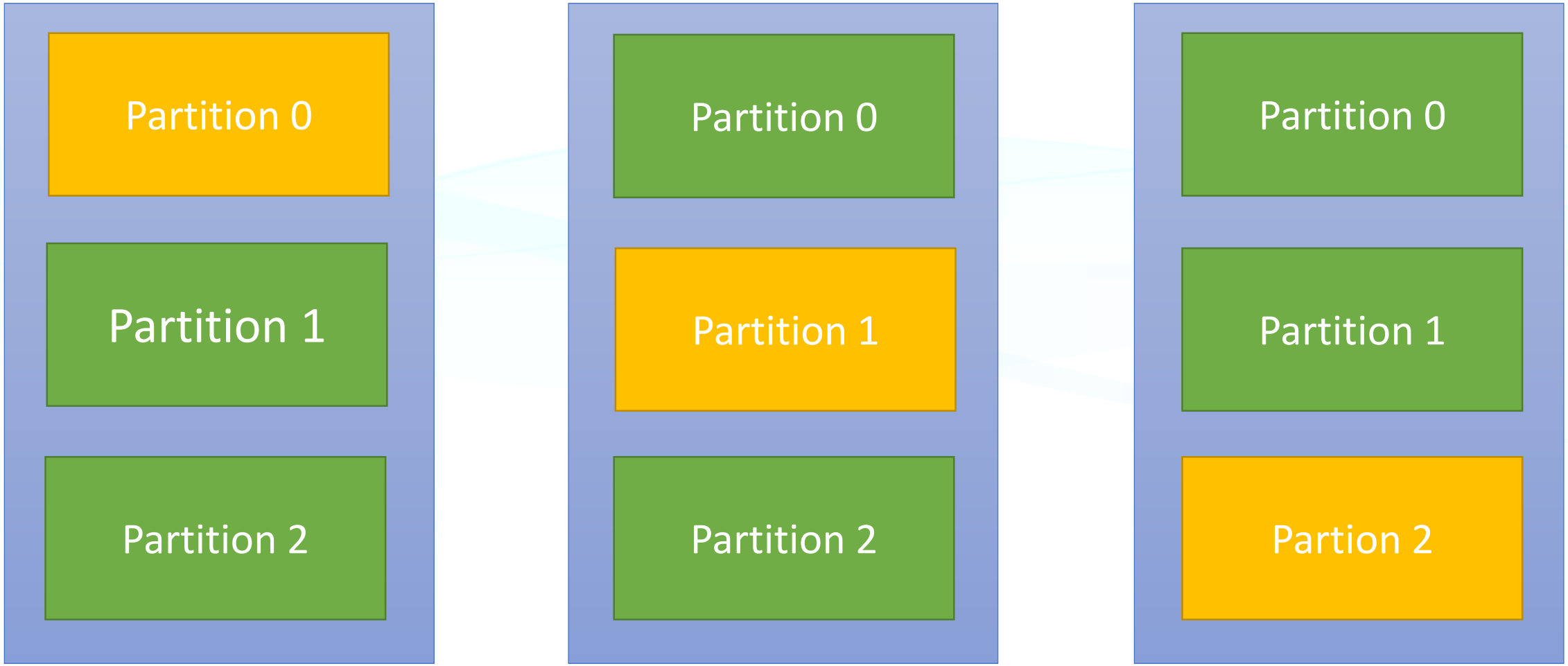
Topics, Partitions and Logs



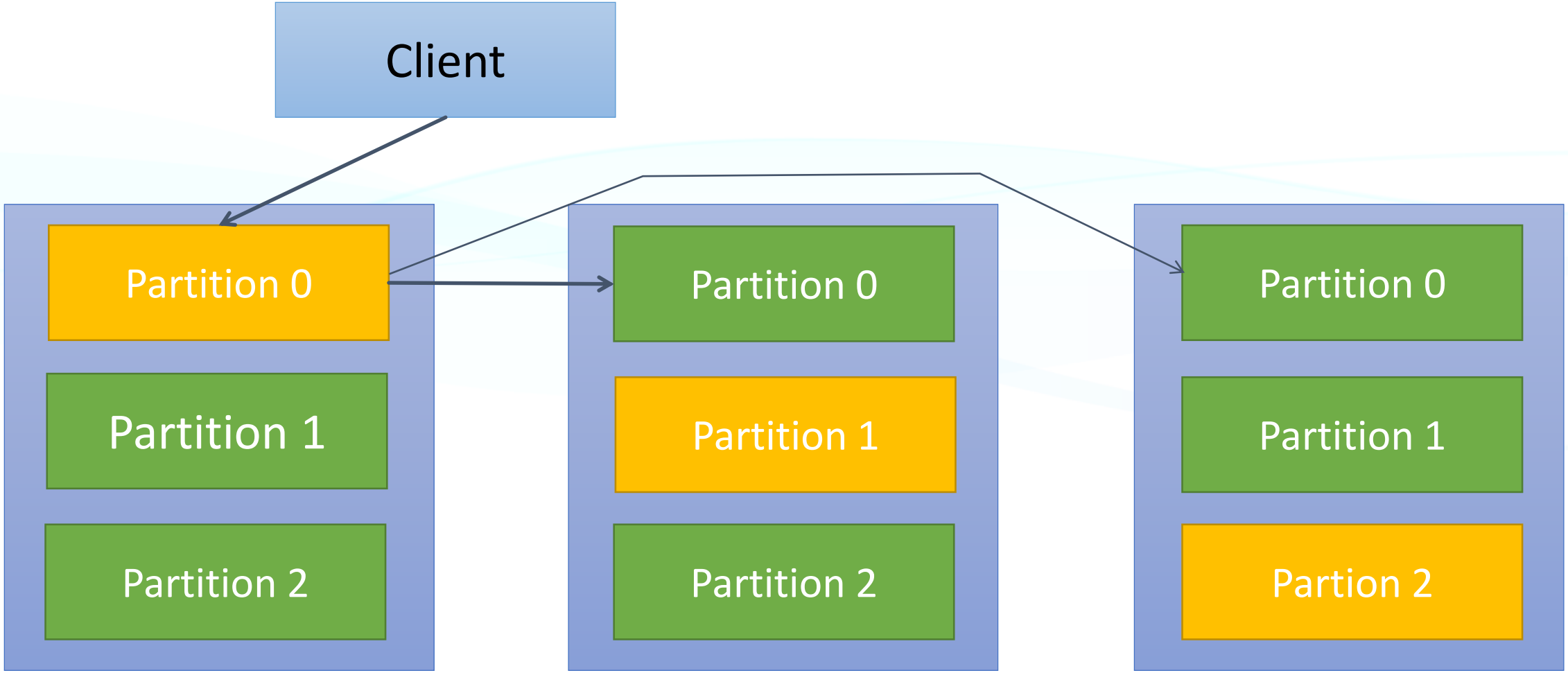
Each partition is a log



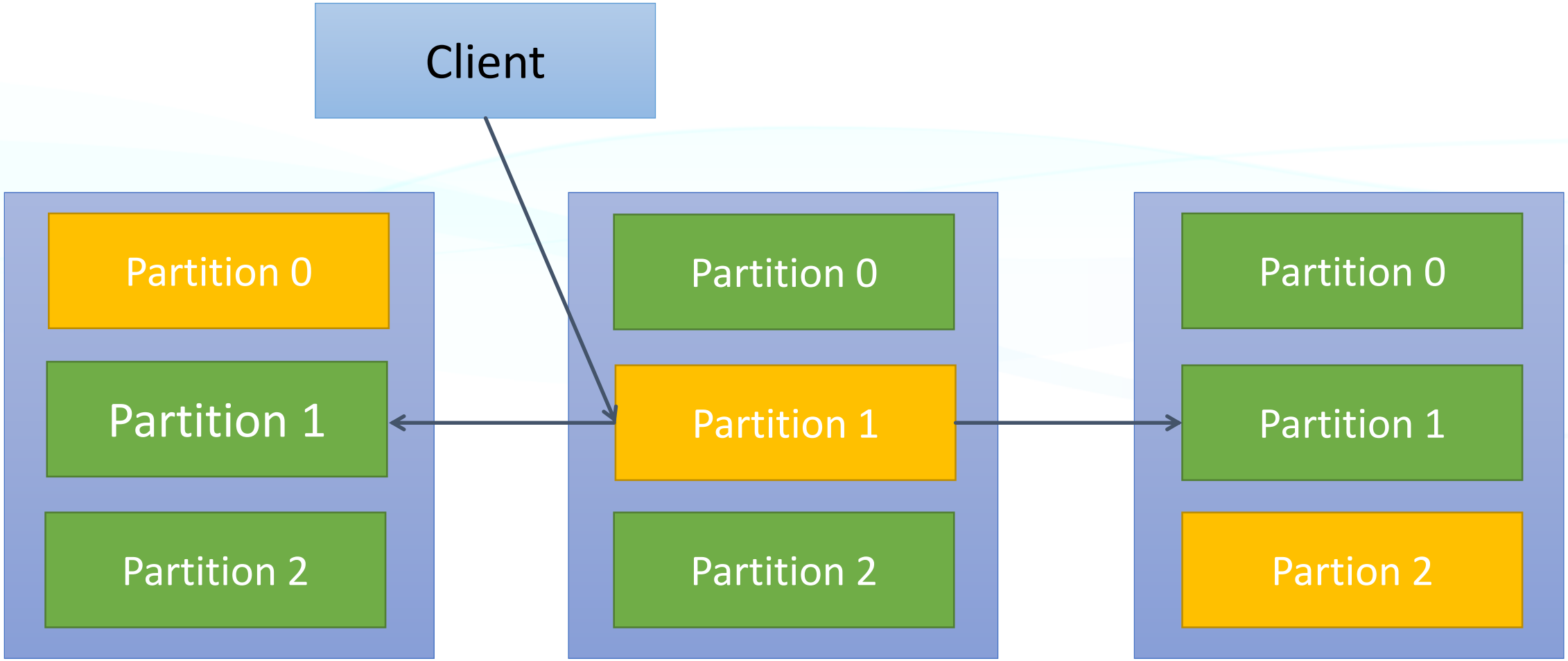
Each Broker has many partitions



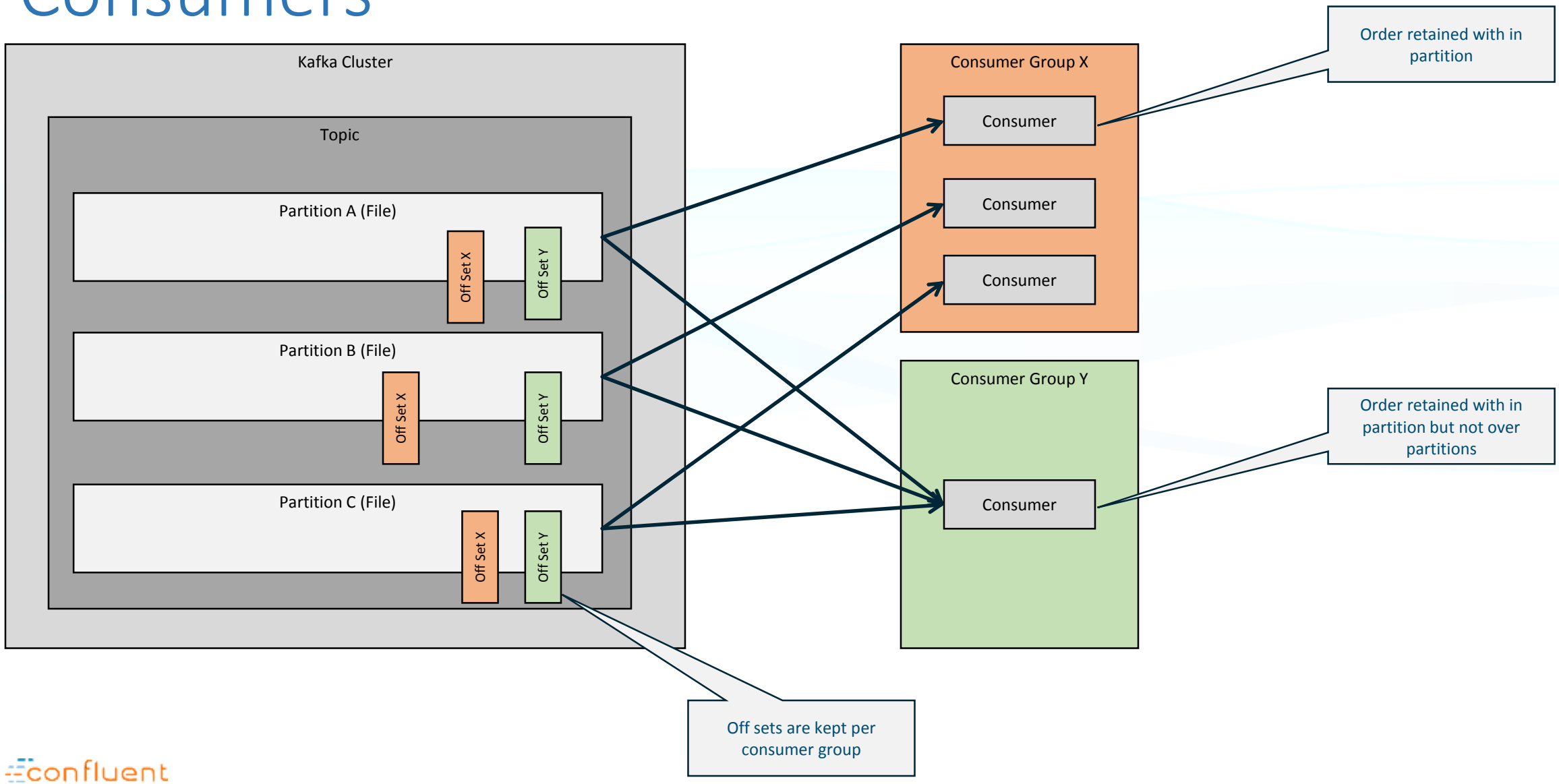
Producers load balance between partitions



Producers load balance between partitions



Consumers


Kafka Cluster
Topic
Partition A (File)
Off Set X
Off Set Y
Partition B (File)
Off Set X
Off Set Y
Partition C (File)
Off Set X
Off Set Y
Consumer Group X
Consumer
Consumer
Consumer
Consumer Group Y
Consumer
Order retained with in partition
Order retained with in partition but not over partitions
Off sets are kept per consumer group

Kafka “Magic” – Why is it so fast?

- 250M Events per sec on one node at 3ms latency
- Scales to any number of consumers
- Stores data for set amount of time –
  Without tracking who read what data
- Replicates – but no need to sync to disk
- Zero-copy writes from memory / disk to network

How do people use Kafka?

- As a message bus
- As a buffer for replication systems
- As reliable feed for event processing
- As a buffer for event processing
- **Decouple apps from databases**

But really,
how do they use Kafka?

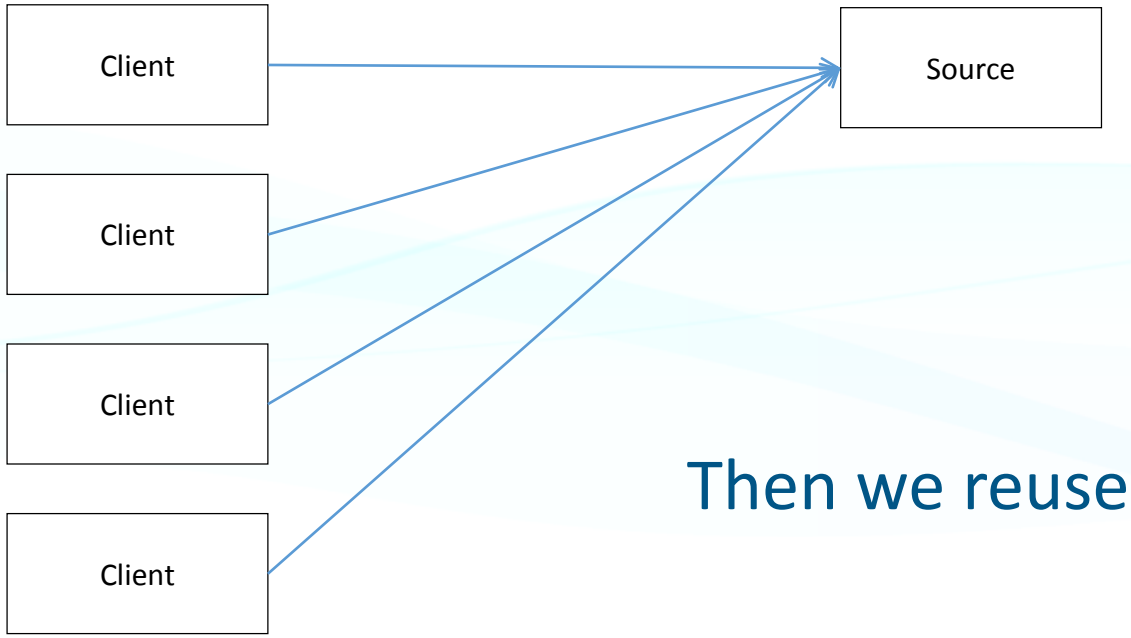
Raise your hand if this sounds familiar

"My next project was to get a working Hadoop setup…

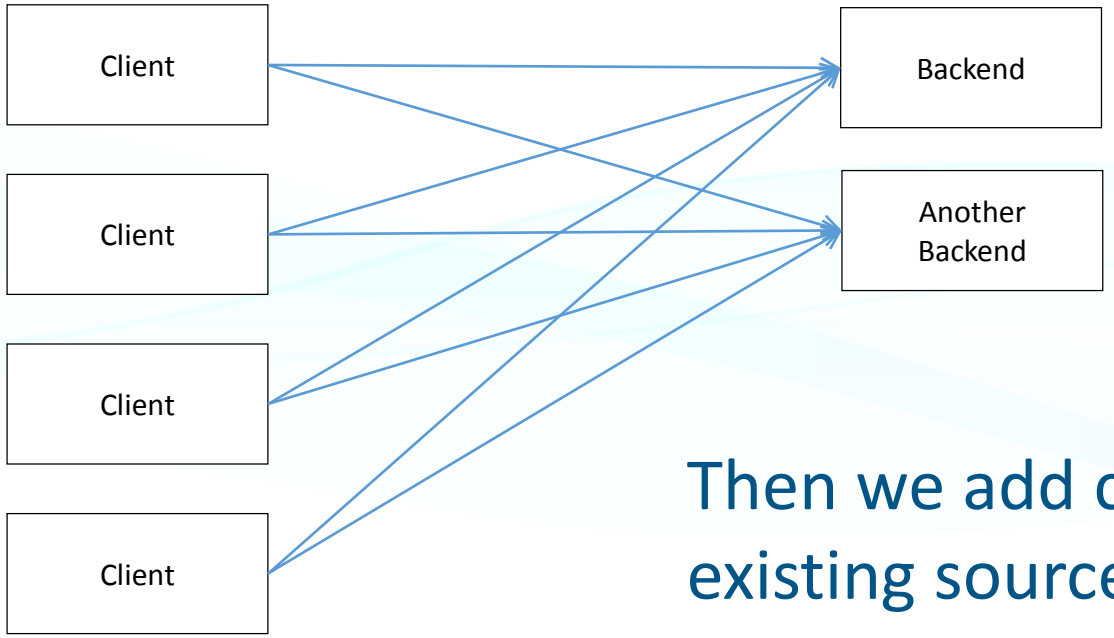
Having little experience in this area, we naturally budgeted a few weeks for getting data in and out, and the rest of our time for implementing fancy algorithms."

--Jay Kreps, Kafka PMC

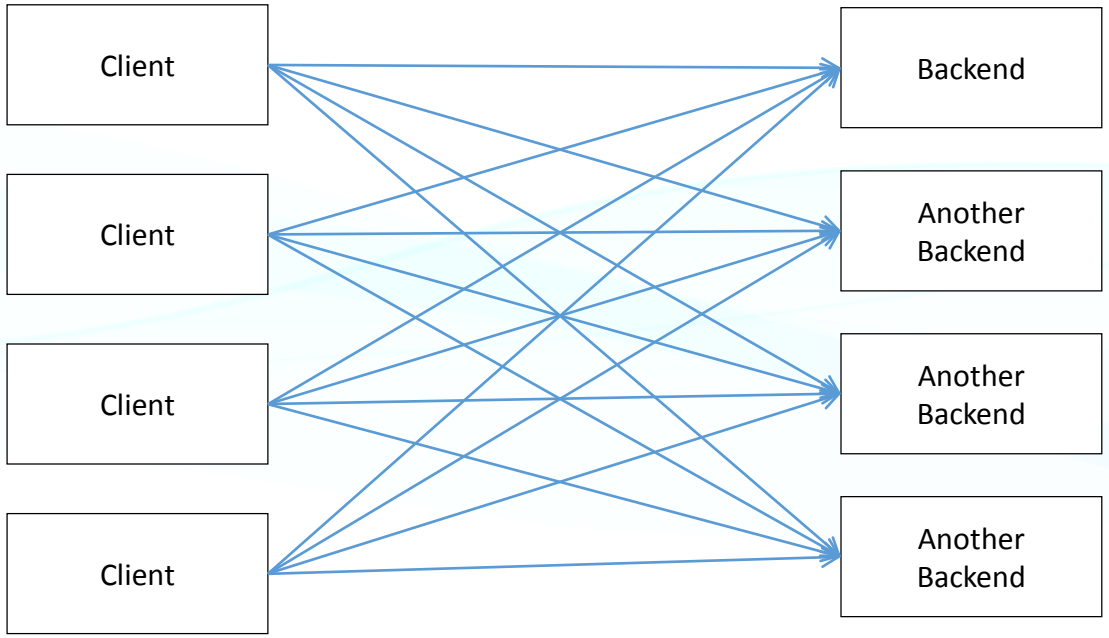
Data Pipelines Start like this.



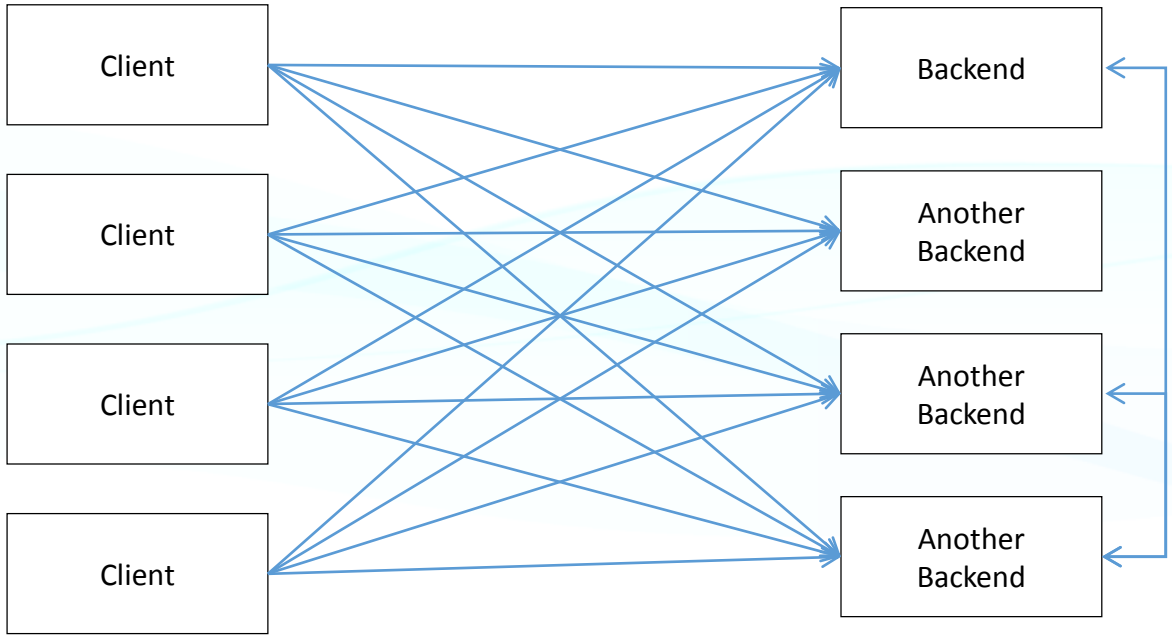
Then we reuse them



Then we add consumers to the existing sources



Then it starts to look like this



With maybe some of this

Queues decouple systems:

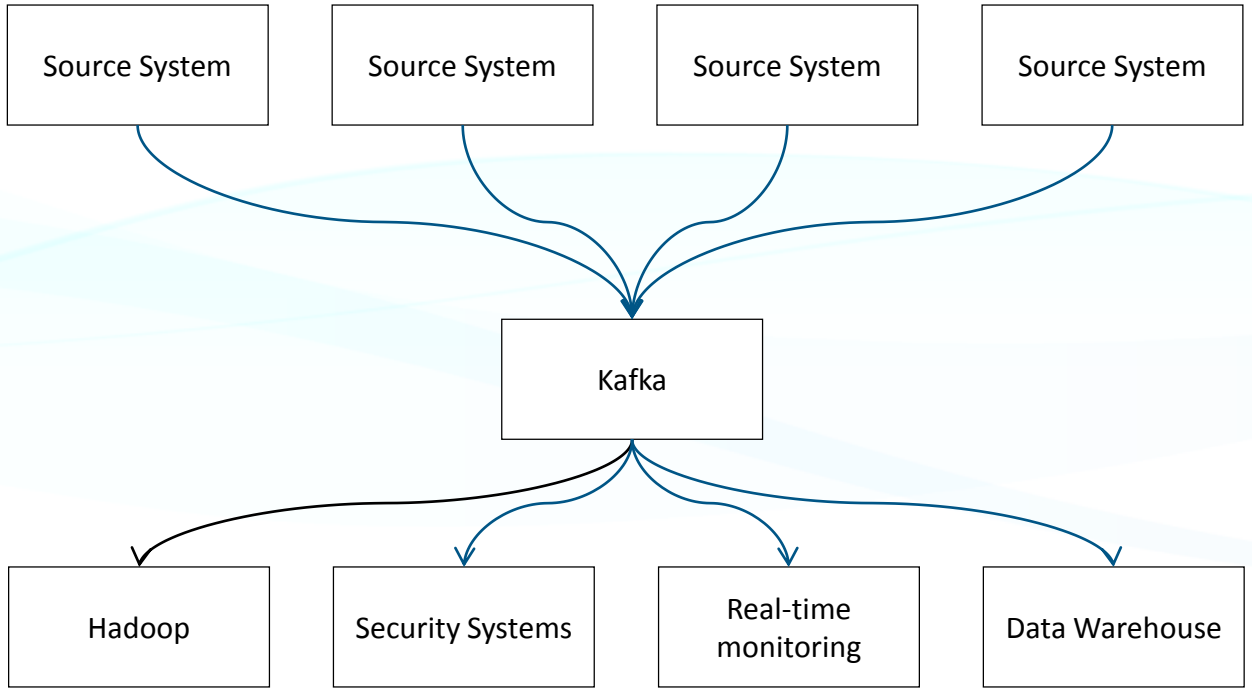
Adding new systems doesn’t require changing Existing systems

This is where we are trying to get

Producers

Brokers

Consumers



Kafka decouples Data Pipelines

Important notes:

- Producers and Consumers dont need to know about each other
- Performance issues on Consumers dont impact Producers
- Consumers are protected from herds of Producers
- Lots of flexibility in handling load
- Messages are available for anyone –
  lots of new use cases, monitoring, audit, troubleshooting

http://www.slideshare.net/gwenshap/queues-pools-caches

My Favorite Use Cases

- Shops consume inventory updates
- Clicking around an online shop? Your clicks go to Kafka and recommendations come back.
- Flagging credit card transactions as fraudulent
- Flagging game interactions as abuse

- Least favorite: Surge pricing in Uber
- Huge list of users at kafka.apache.org

Got it!

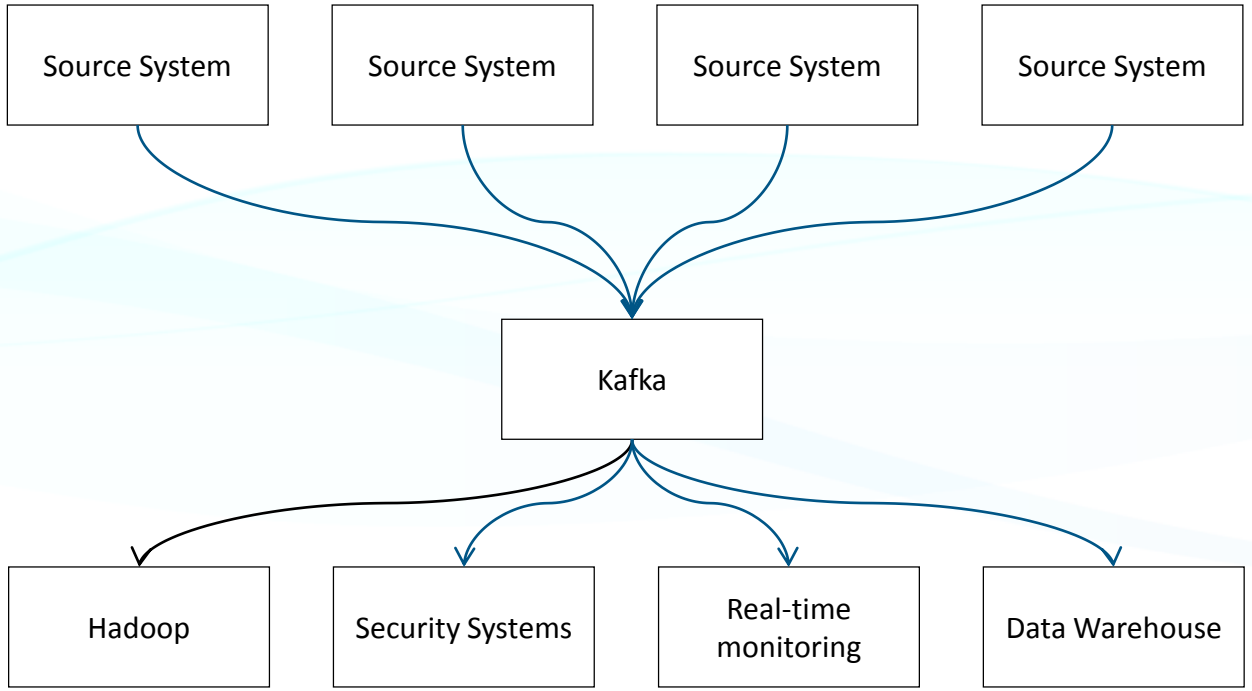
But what about real-time ETL?

Remember This?

Producers

Brokers

Consumers



Kafka is smack in middle of all Data Pipelines

If data flies into Kafka in real time

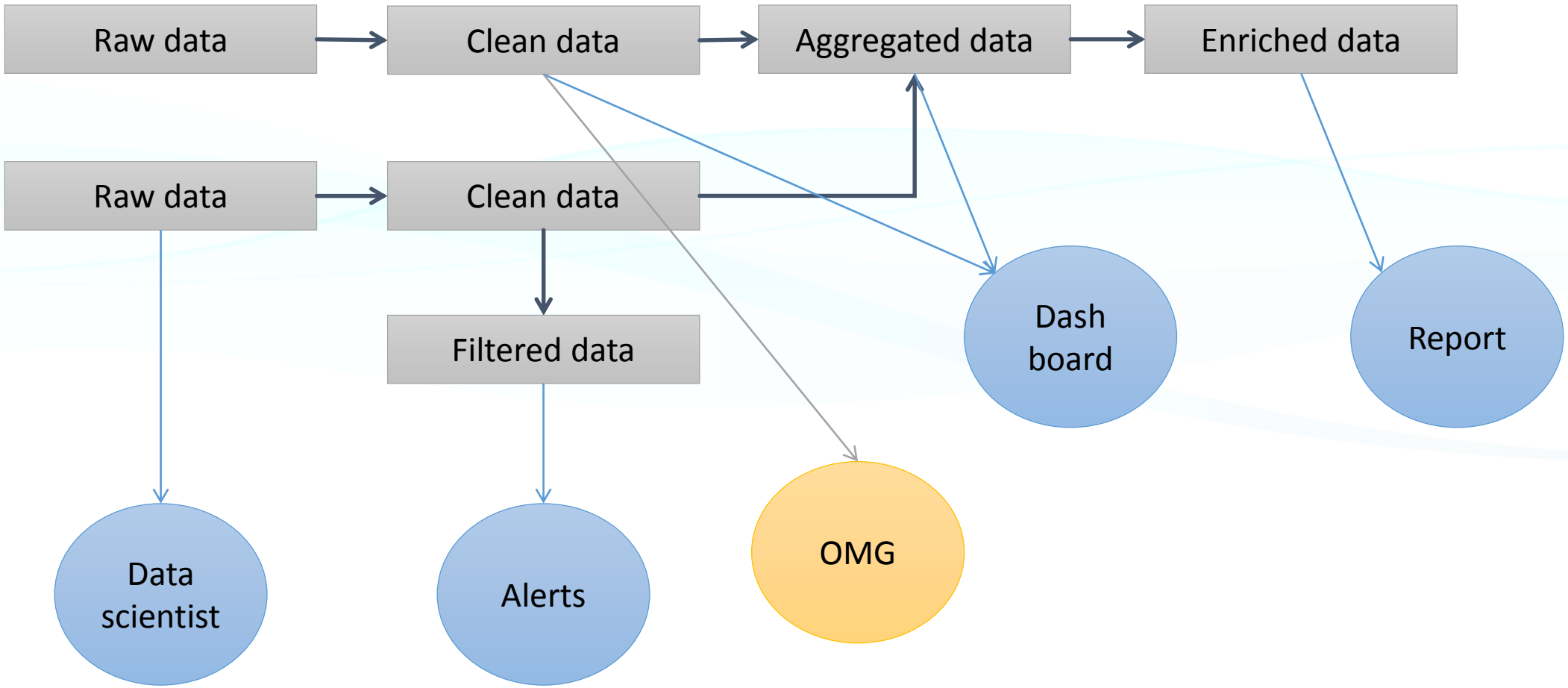
Why wait 24h before pulling it into a DWH?

	Batch	Mini-Batch	Micro-Batch	Real-Time
Description	Data is loaded in full or incrementally using a off-peak window.	Data is loaded incrementally using intra-day loads.	Source changes are captured and accumulated to be loaded in intervals.	Source changes are captured and immediately applied to the DW.
Latency	Daily or higher	Hourly or higher	15min & higher	sub-second
Capture	Filter Query	Filter Query	CDC	CDC
Intialization	Pull	Pull	Push, then Pull	Push
Target Load	High Impact	Low Impact, load frequency is tuneable		
Source Load	High Impact	Queries at peak times necessary	Some to none depending on CDC technique	

Why Kafka makes real-time ETL better?

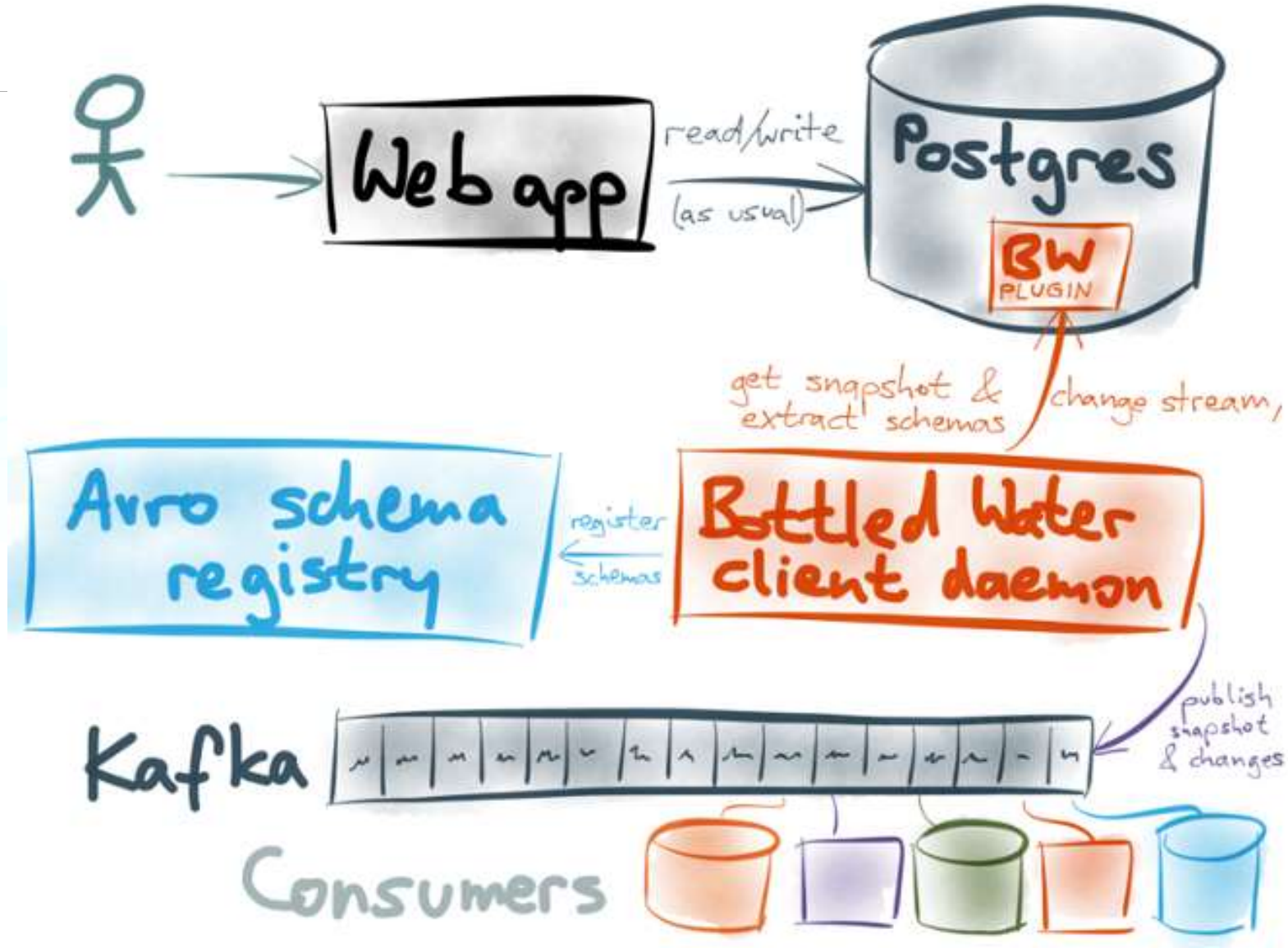
- Can integrate with any data source
  - RDBMS, NoSQL, Applications, web applications, logs
- Consumers can be real-time
  But they do not have to
- Reading and writing to/from Kafka is cheap
  - So this is a great place to store intermediate state
- You can fix mistakes by rereading some of the data again
  - Same data in same order
- Adding more pipelines / aggregations has no impact on source systems = low risk

It is all valuable data

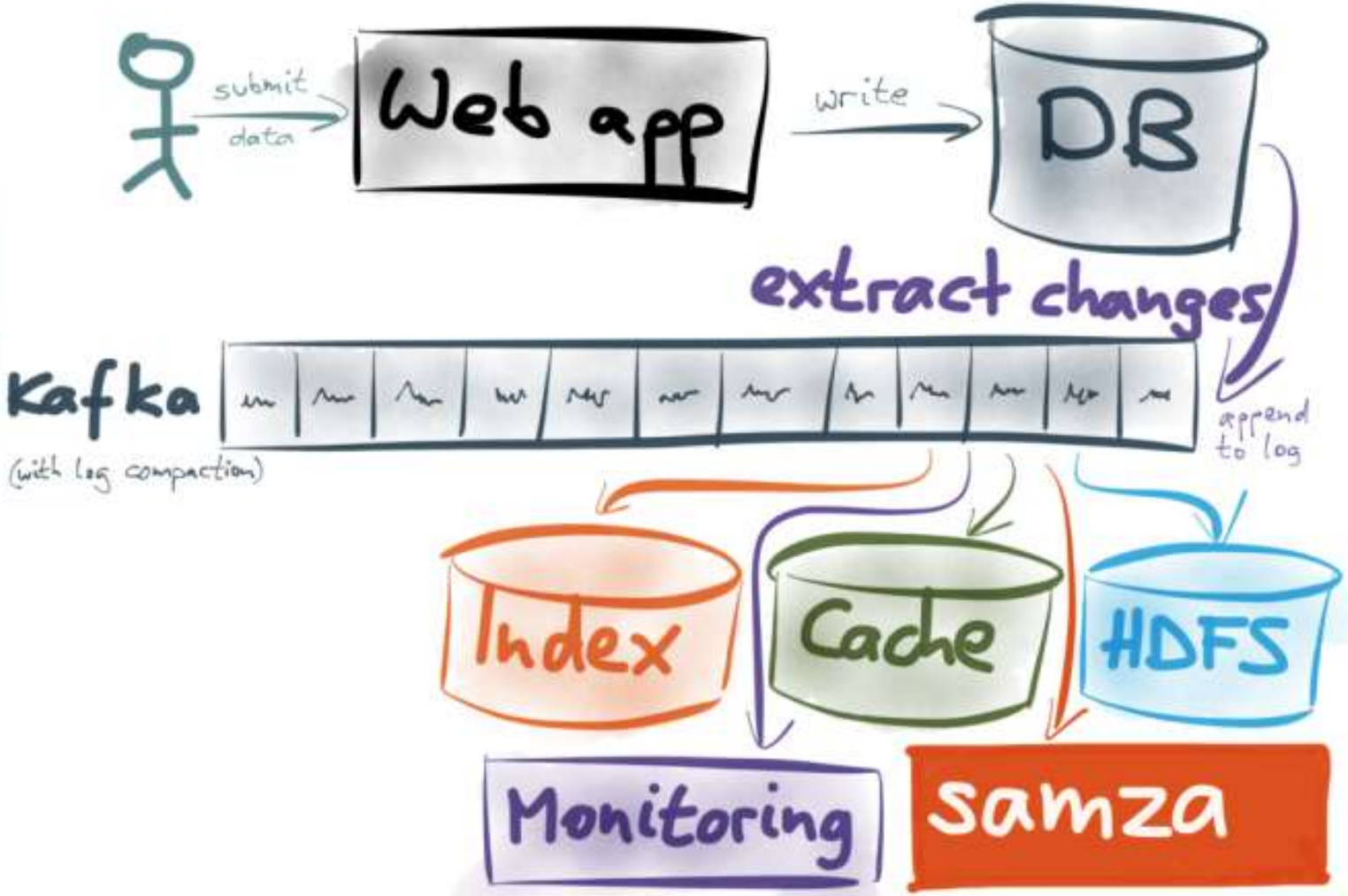


OK, but how does my data get into Kafka

- Producers
- Log4J
- Rest Proxy
- BottledWater
- KafkaConnect and its connectors ecosystem
- Other ecosystem


Web app
read/write
(as usual)
Postgres
BW
PLUGIN
get snapshot &
extract schemas
change stream,
Avro schema
registry
register
schemas
Bottled Water
client daemon
publish
snapshot
& changes
Kafka
Consumers

USING CHANGE CAPTURE



But wait, how do we process the data?

- However you want:
  - You just consume data, modify it, and produce it back
- Built into Kafka:
  - Kprocessor
  - Kstream
- Popular choices:
  - Storm
  - SparkStreaming

One more thing...

Schema is a MUST HAVE for data integration

Need More Kafka?

- https://kafka.apache.org/documentation.html
- My video tutorial: http://shop.oreilly.com/product/0636920038603.do
- http://www.michael-noll.com/blog/2014/08/18/apache-kafka-training-deck-and-tutorial/
- Our website: http://confluent.io
- Oracle guide to real-time ETL: http://www.oracle.com/technetwork/middleware/data-integrator/overview/best-practices-for-realtime-data-wa-132882.pdf