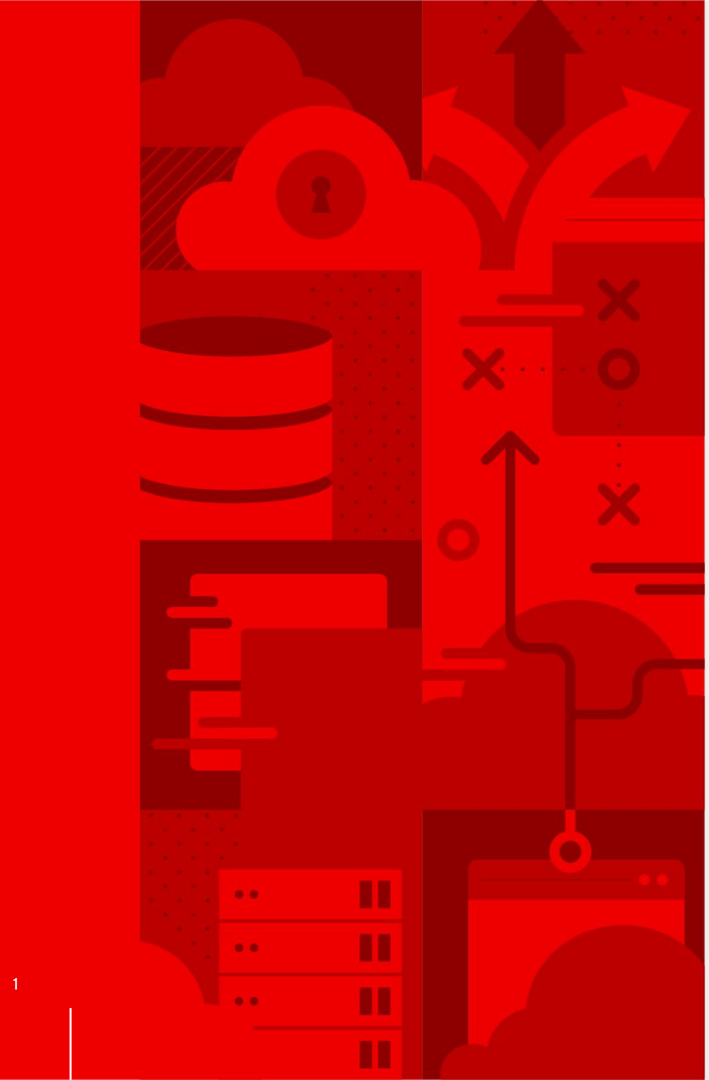


CQRS and Event Source with Kafka and Eclipse Vert.x

MIcroservices Data Patterns

Don Schenck
digital evangelist

Code is easy…

Unless it's RPG III

Code is easy…

…Data is **HARD**

Data Everywhere:

Data Everywhere: RDBMS

Data Everywhere:
RDBMS
NoSQL

Data Everywhere:
RDBMS
NoSQL
Flat files

Data Everywhere:
RDBMS
NoSQL
Flat files (including spreadsheets!)

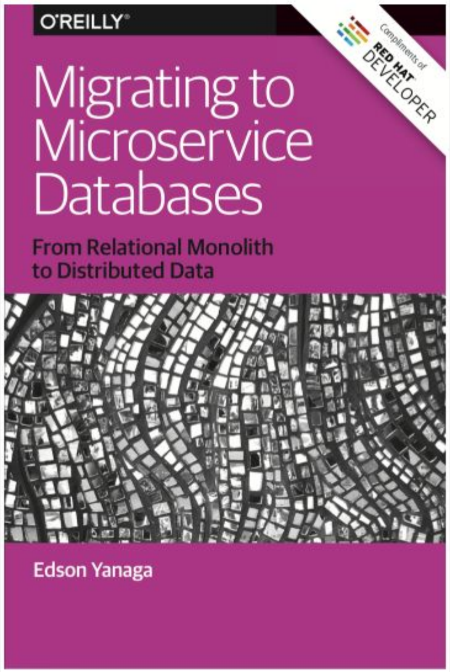
Ugh. Make it stop.

Centralized App + Centralized Data = "Everything's fine"

But then …

iTunes Store
Yelp
Instagram
Phoster
Facebook
Twitter

Remember: The “S” in IoT stands for “Security” https://developers.redhat.com/books/migrating-microservice-databases-relational-monolith-distributed-data/


O'REILLY
Compliments of RED HAT DEVELOPER
Migrating to Microservice Databases
From Relational Monolith to Distributed Data
Edson Yanaga

Split data from RDBMS

Apps change; data tends to stick around

You always have data.

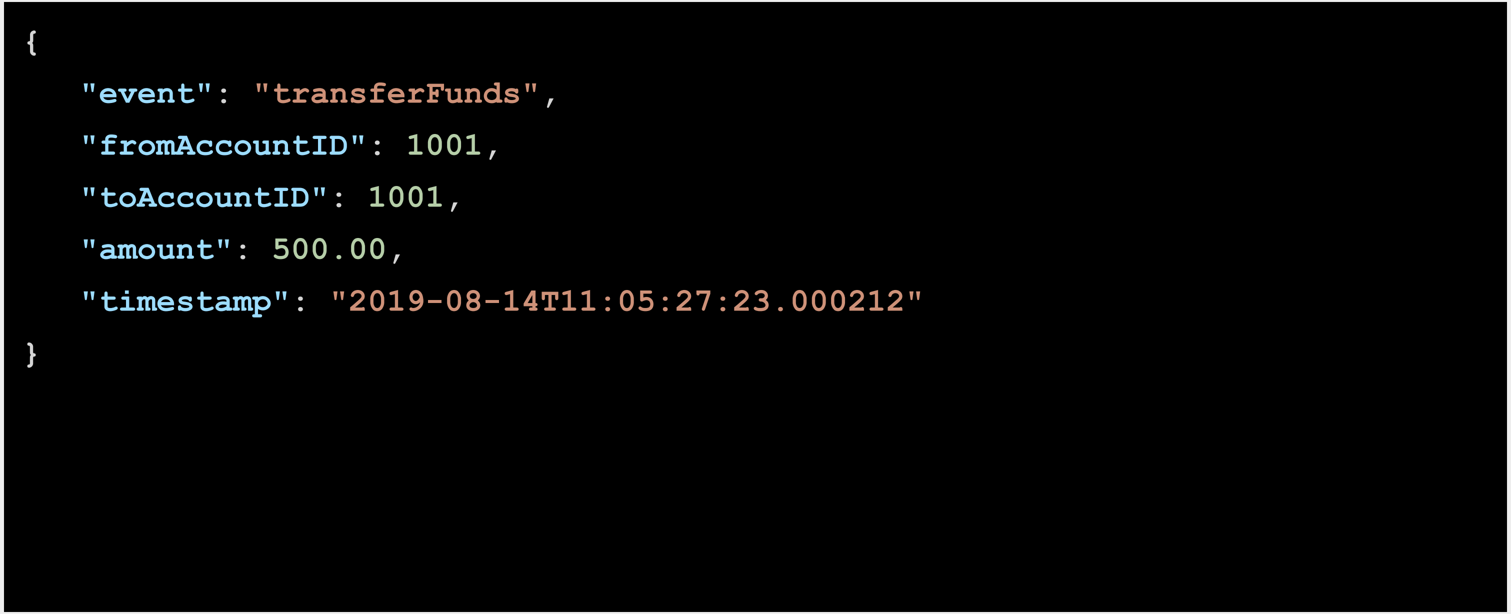
A decade (or so) ago:
ORM - Hibernate
POJOs and XML

Then ... Event Sourcing came along

Think “Events” instead of “Data Structure”.

Account

ID	customerID	balance
1001	990	1000
1002	991	0
1003	991	-500
1004	992	300



```
{
    "event": "transferFunds",
    "fromAccountID": 1001,
    "toAccountID": 1001,
    "amount": 500.00,
    "timestamp": "2019-08-14T11:05:27:23.000212"
}
```

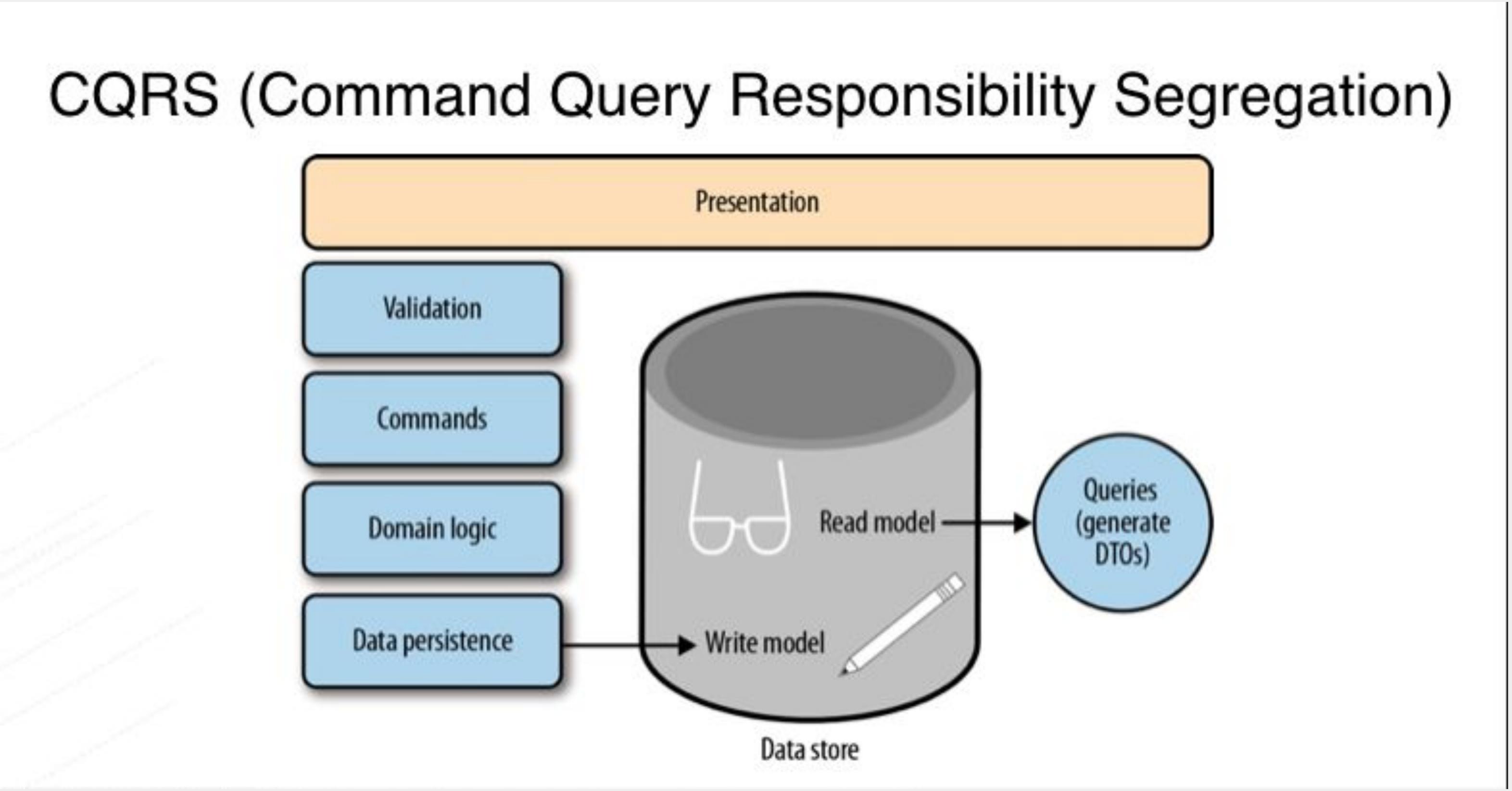
An Event is based on the real world, not IT-think.

CQS
Command-Query Separation

“Asking a question should not change the answer” -- Bertrand Meyer

CQRS - Command Query Responsibility Segregation

A simple, basic example

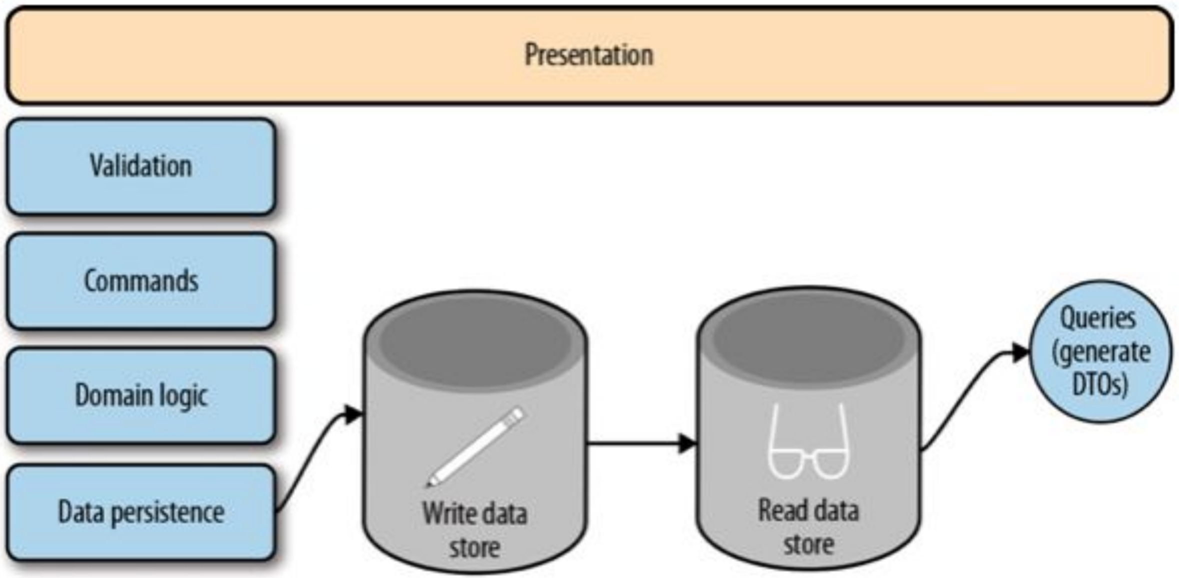


CQRS (Command Query Responsibility Segregation)

This is simple?

But distributed systems bring complexity…

CQRS with separate data stores



Examples:

- Read one
- Read a list
- Search...

CQRS & Event Sourcing

Why CQRS?

Performance

...and...

- Distribution
- Availability
- Integration
- Analytics

WHEN CQRS?

Single Source of Truth: The WRITE data store

Next: Create your READ data stores.

Add Events to update data stores.

More about Event Sourcing:

It’s WRITE-ONLY

You get audit trails/history built in

Well … that's nice

Events are immutable

The Event Store fires events that are independent of the origin

So convenient. I guess it's a convenience store yuk yuk

Events...

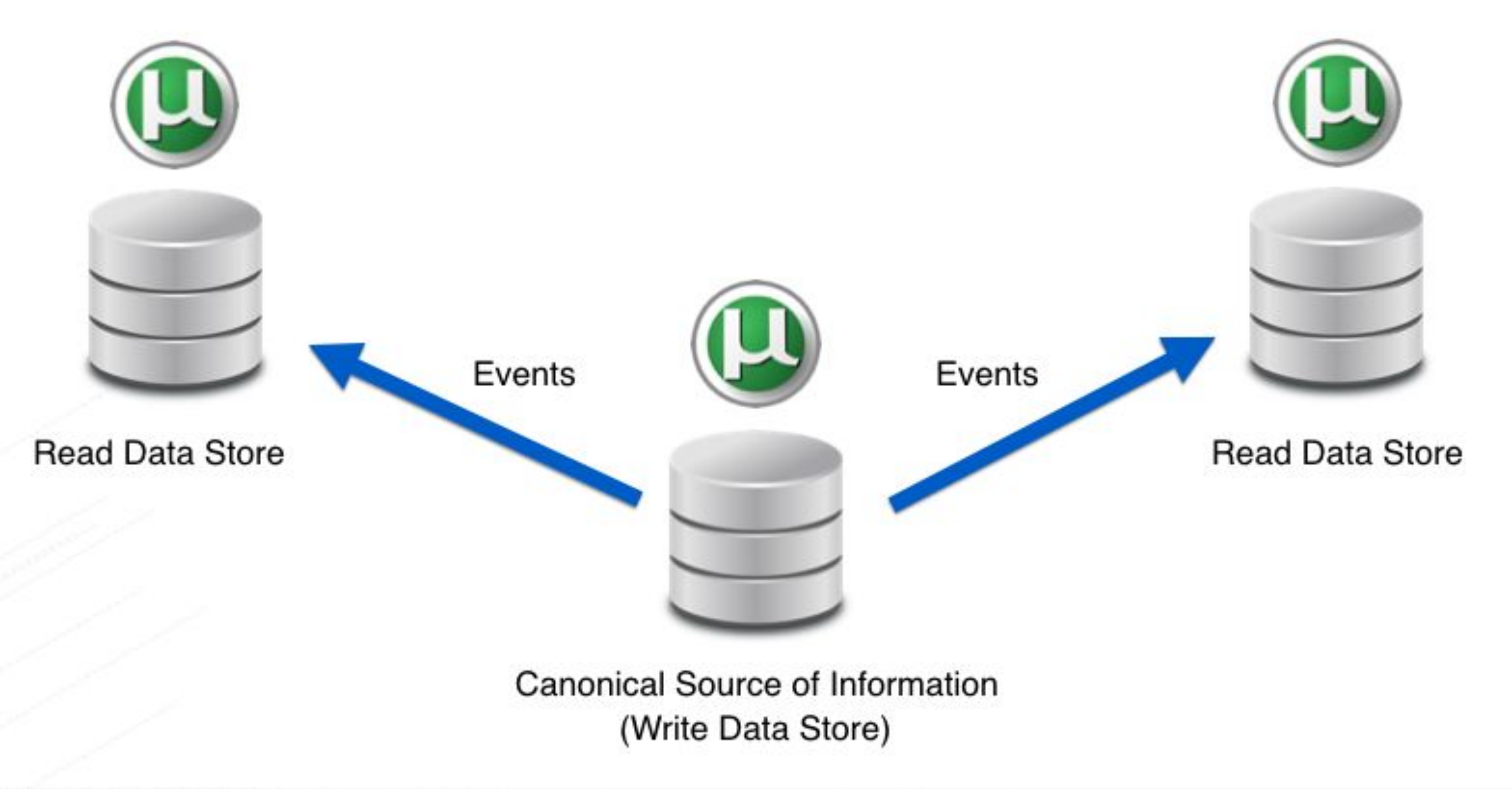
Maintain a Materialized View

Integrate external systems

Are not reliant on a database schema

Can materialize the current state (This can be a batch job!)

Events follow the “Fire-and-Forget” model of operation.



It was time for another pretty picture

Somethings that affect your decision regarding updating READ stores:

Latency

Size

Staleness

Ownership

Security

Nature and depth of information

So how do we find the current state?

Cheat Mode = ON (It’s okay to bookmark data)

Up up down down left right left right B A

CQRS considerations:

CQRS considerations:

- Complexity

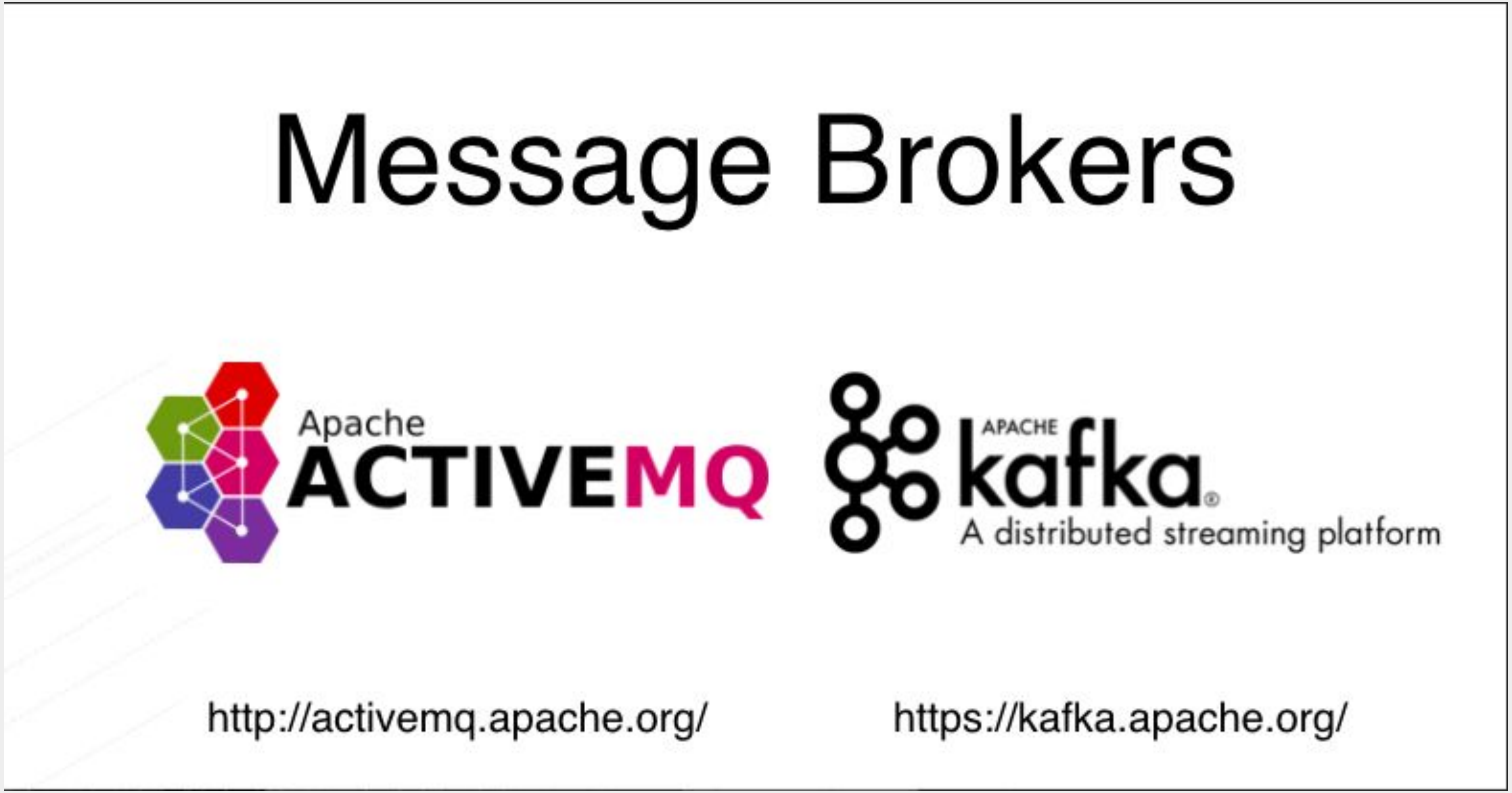
CQRS considerations:

- Complexity
- Consistency

CQRS considerations:

- Complexity
- Consistency
- Communication

Distributing events using a Message Broker



Message Brokers

Apache
ACTIVEMQ
APACHE
kafka
A distributed streaming platform

http://activemq.apache.org/

https://kafka.apache.org/

Kafka
Streaming platform (with ordered delivery)

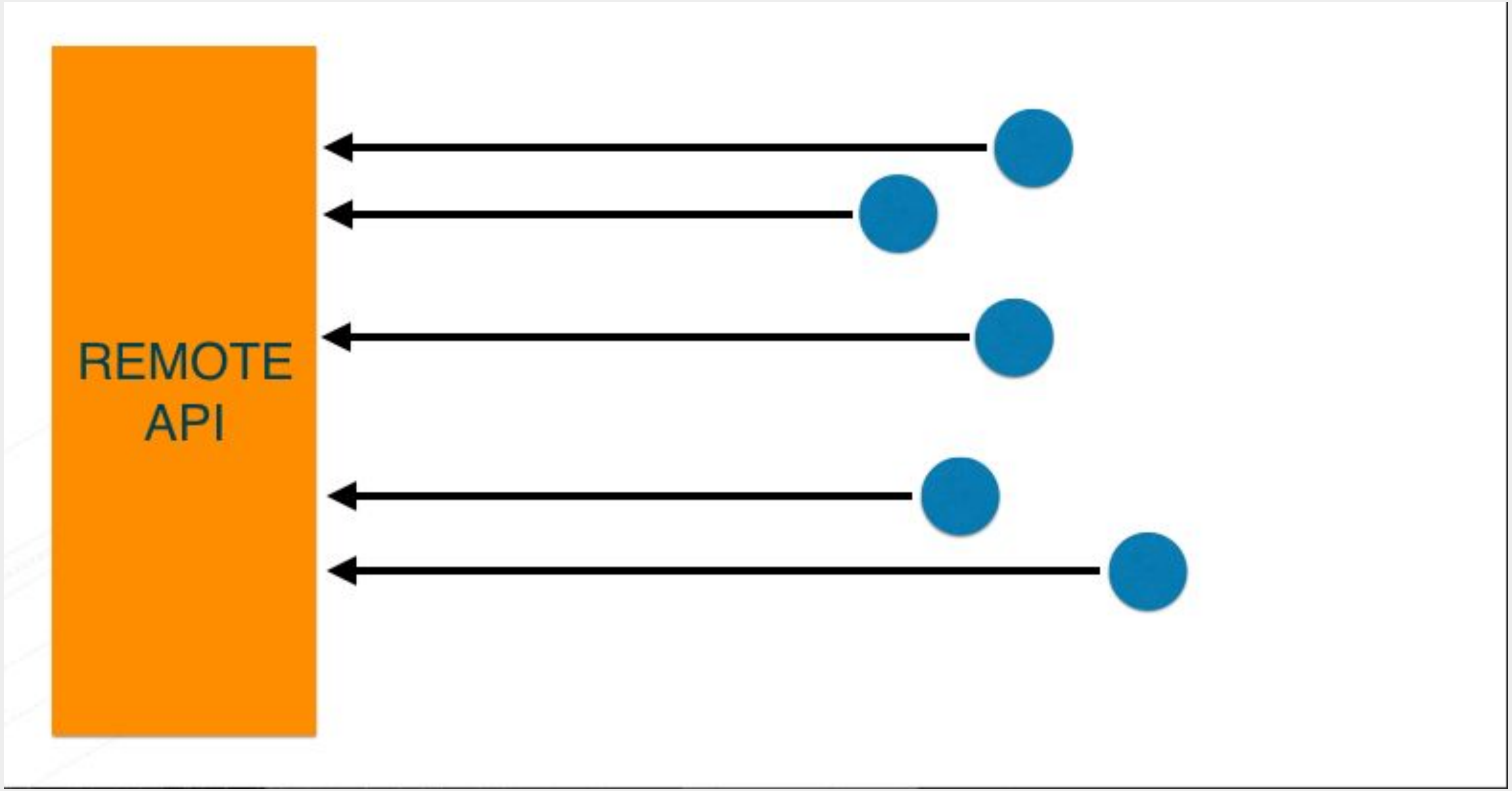
Kafka uses a Publish/Subscribe model with Topics

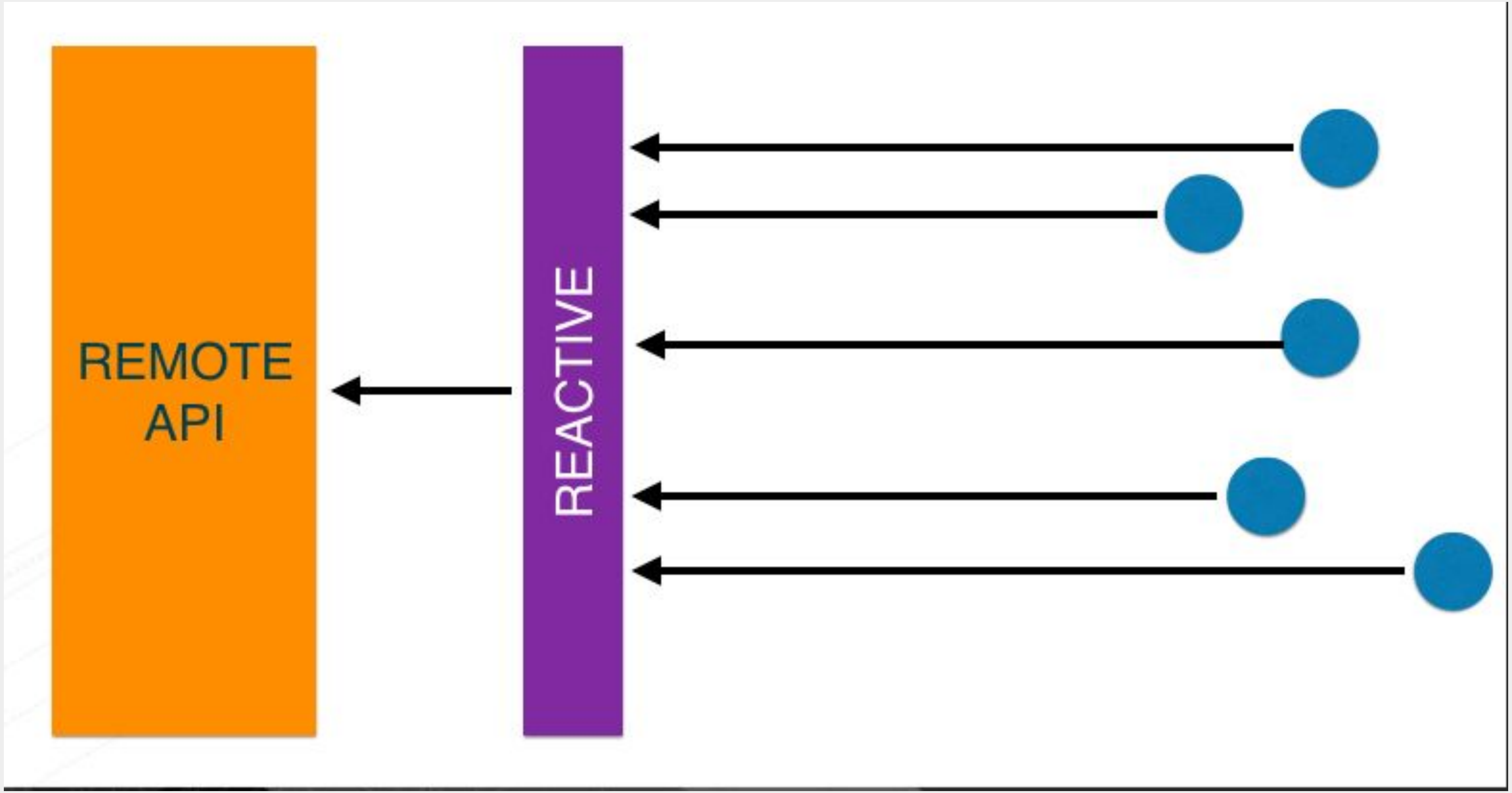
Vert.x
Reactive platform

Reactive Platform

VERT.X

http://vertx.io/


REMOTE
API


REMOTE API
REACTIVE

github.com/reactica/rhte-demo

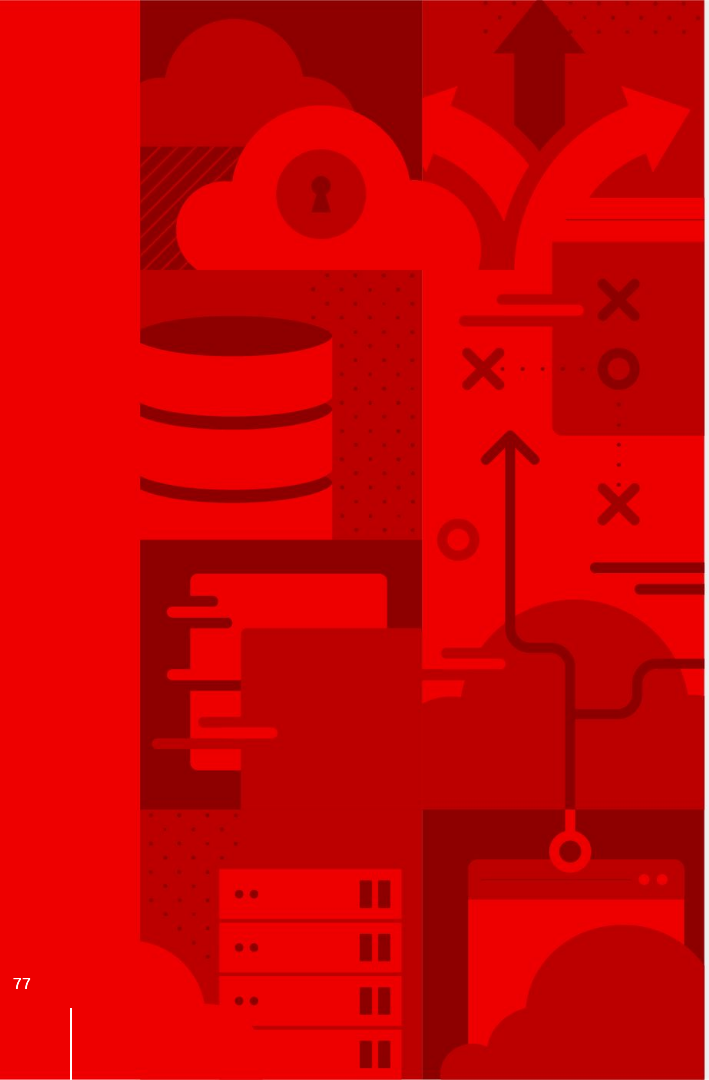
github.com/vert-x3/vertx-examples

developers.redhat.com

@DonSchenck

So many URLs, so little time. What's another browser tab open between friends?

This slide intentionally left blank



Thank you

Red Hat is the world’s leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.

in linkedin.com/company/red-hat

f facebook.com/redhatinc

youtube.com/user/RedHatVideos

twitter.com/RedHat

Red Hat | Red Hat Developer