NSQ

a realtime distributed messaging platform

https://github.com/bitly/nsq

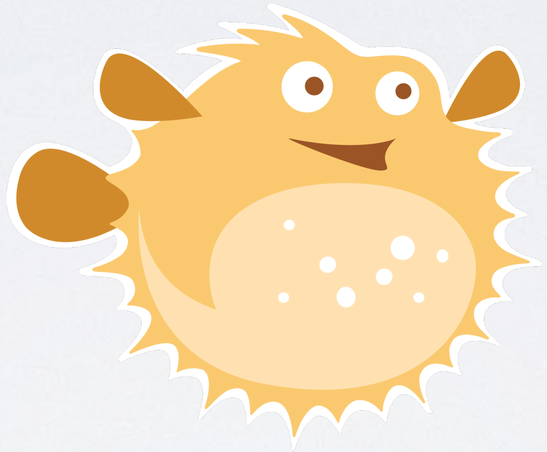
@imsnakes

April 24th 2014 - GopherCon

Matt Reiferson - CTO at Torando Labs

patterns

HI, I'M MATT



HI, I'M MATT

NAMING SUCKS QUEUE

WHAT EVEN IS **NSQ**?

NOT STABLE QUEUE

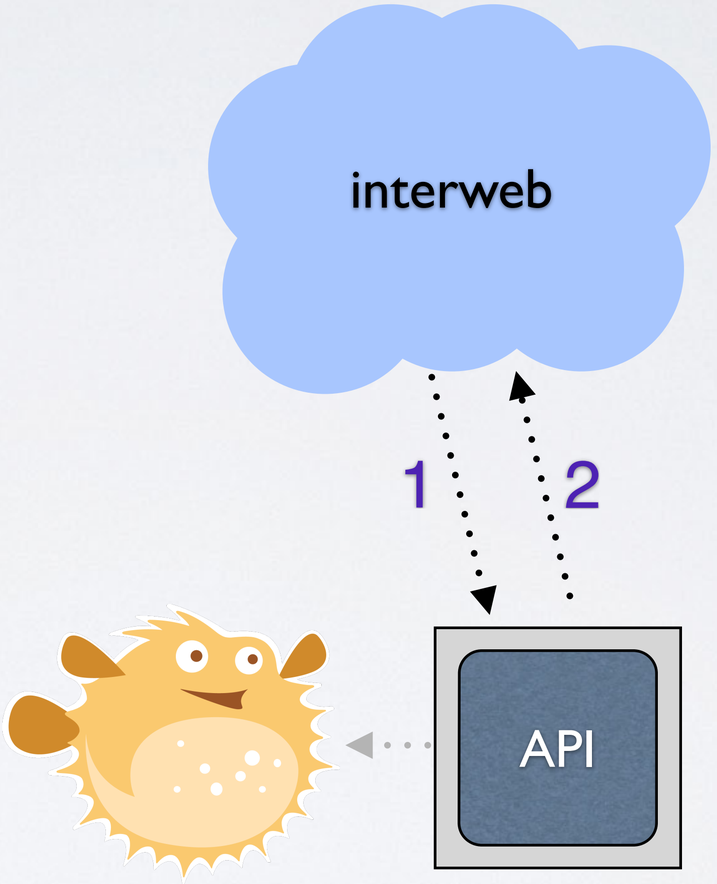
NEW SIMPLE QUEUE

Environmental
Friendly

SPRAY SOME **NSQ** ON IT

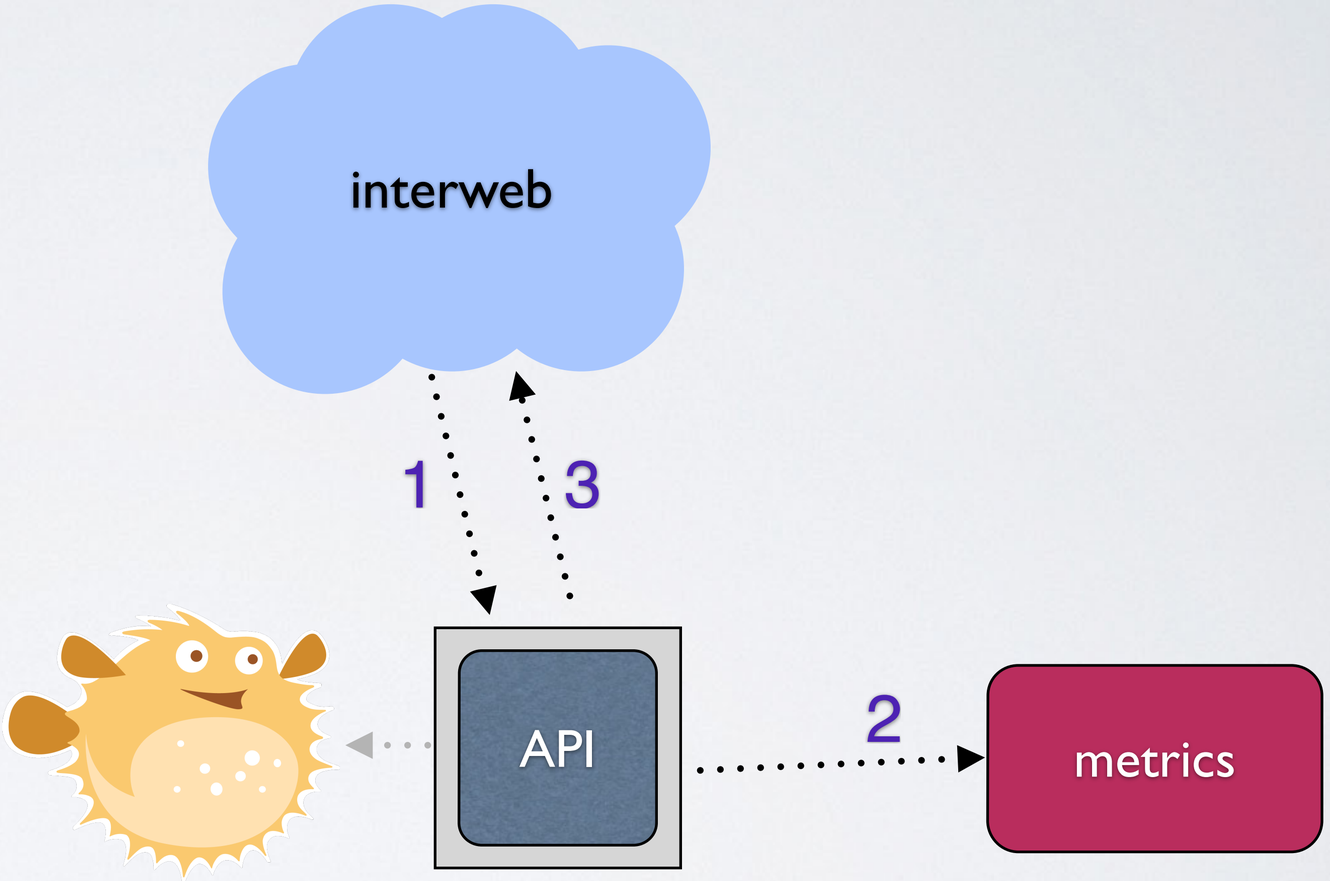
BIG THINGS HAVE SMALL BEGINNINGS

- boss says: "**I want metrics!**"
- **single** host
- **synchronous** writes

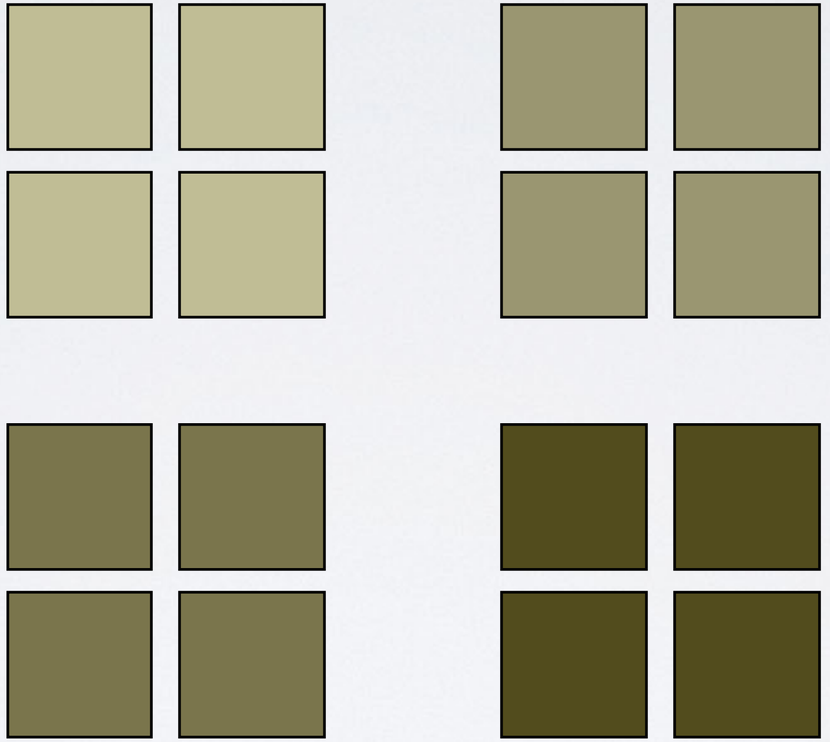


BIG THINGS HAVE SMALL BEGINNINGS

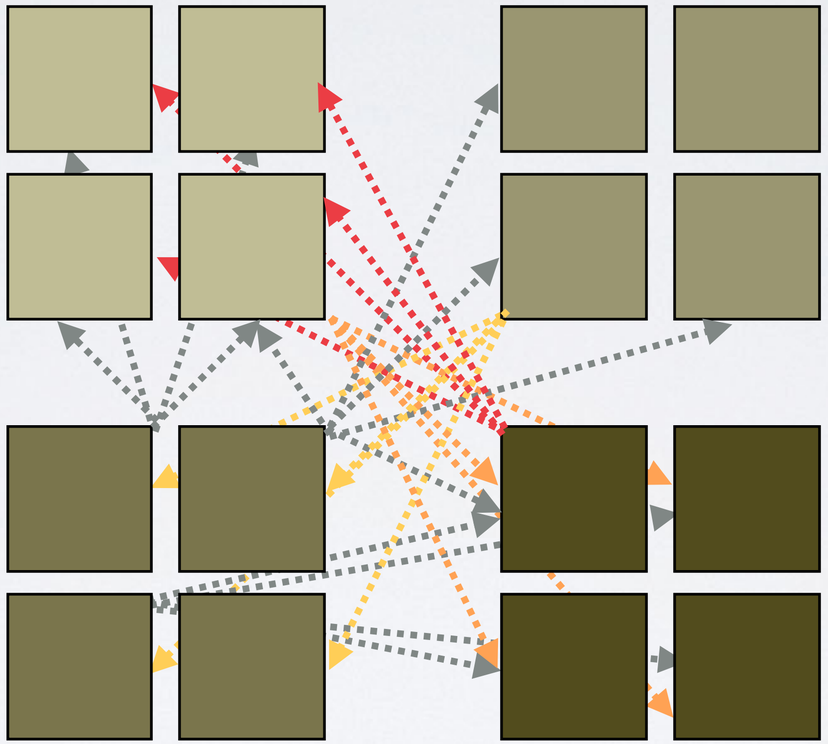
- boss says: "**I want metrics!**"
- **single** host
- **synchronous** writes



. . . AND THEN THEY EAT YOU



. . . AND THEN THEY EAT YOU

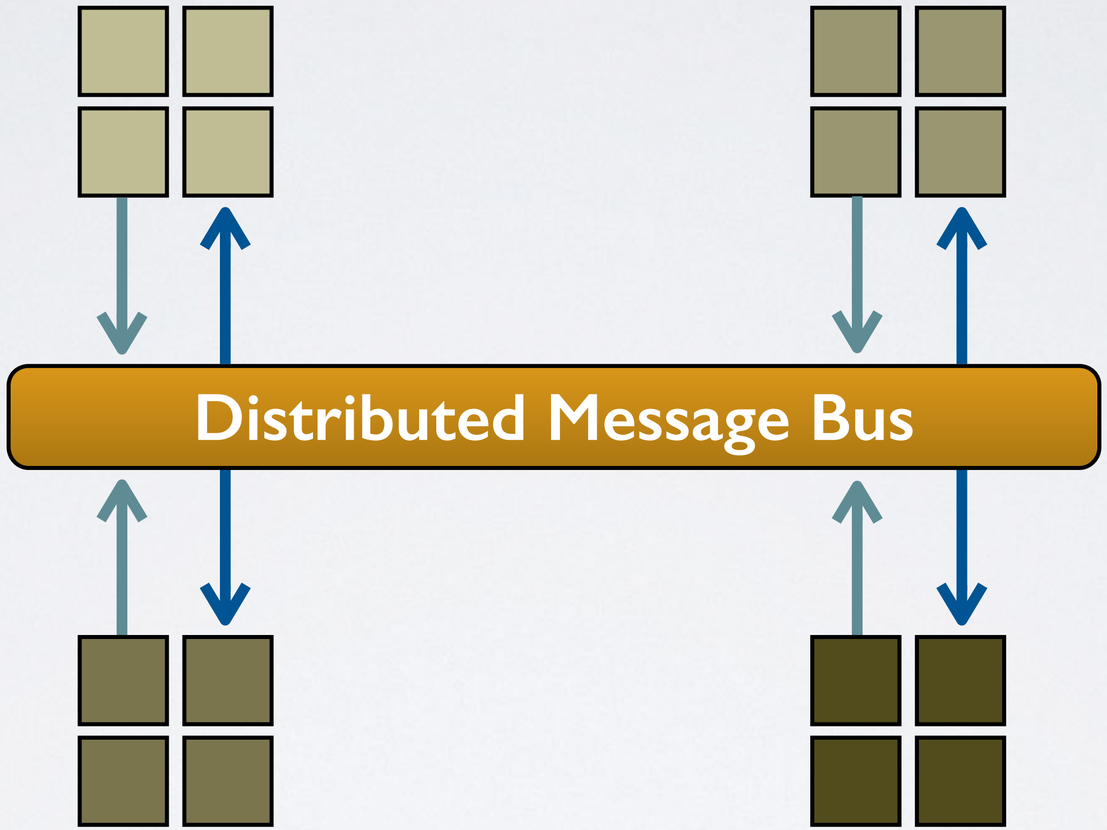


nightmare

BUT THERE'S HOPE

JUST USE

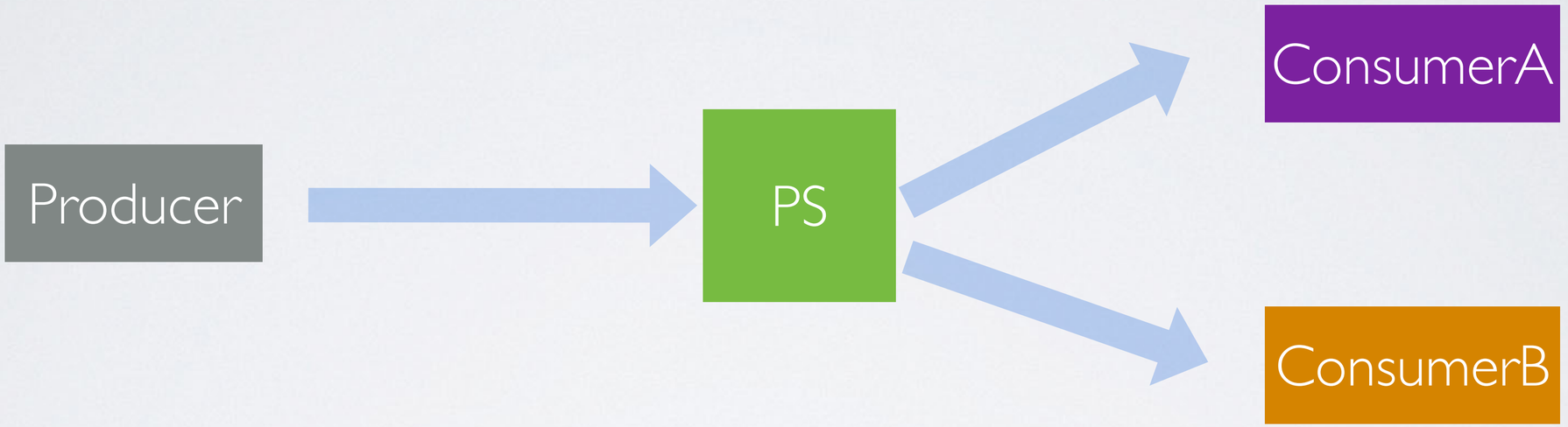
THE GOAL



provide a unifying distributed system to receive
and disseminate event data

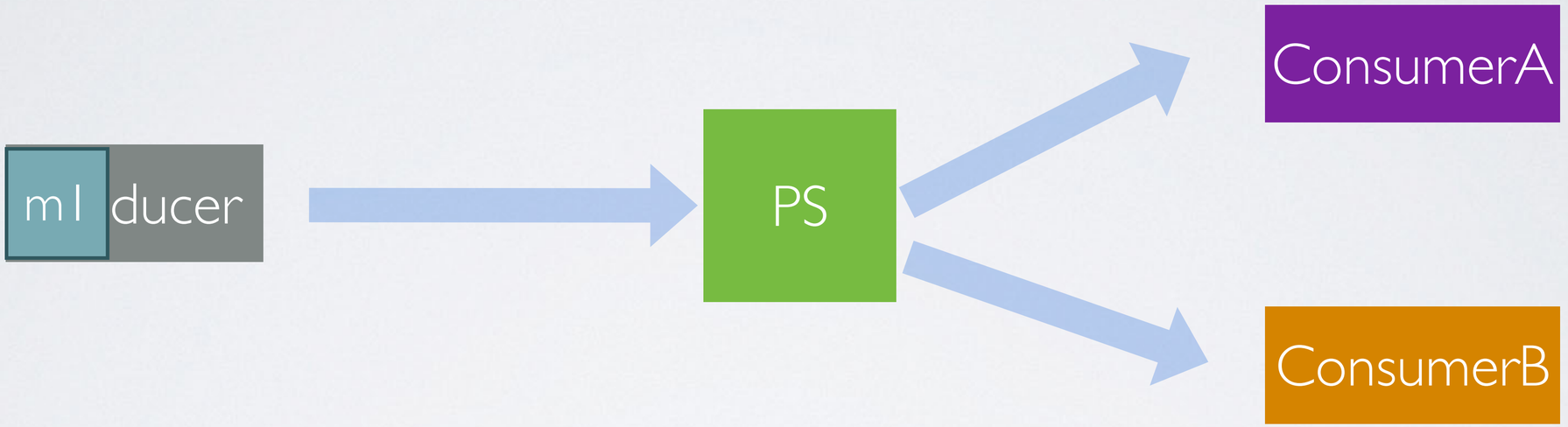
MESSAGING PATTERNS

BROADCAST



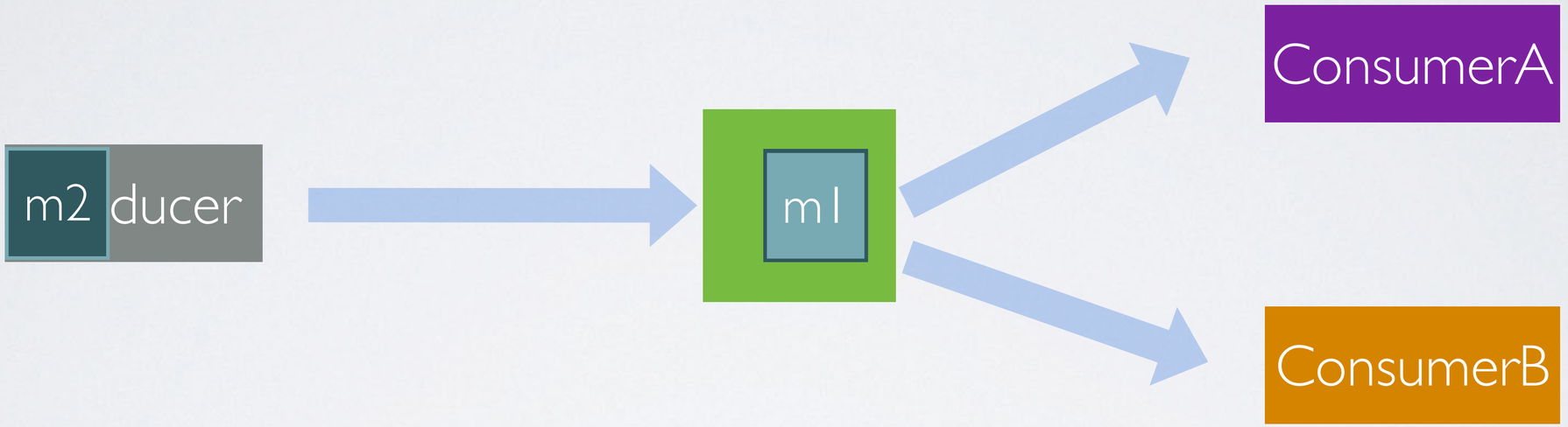
de-couple

BROADCAST



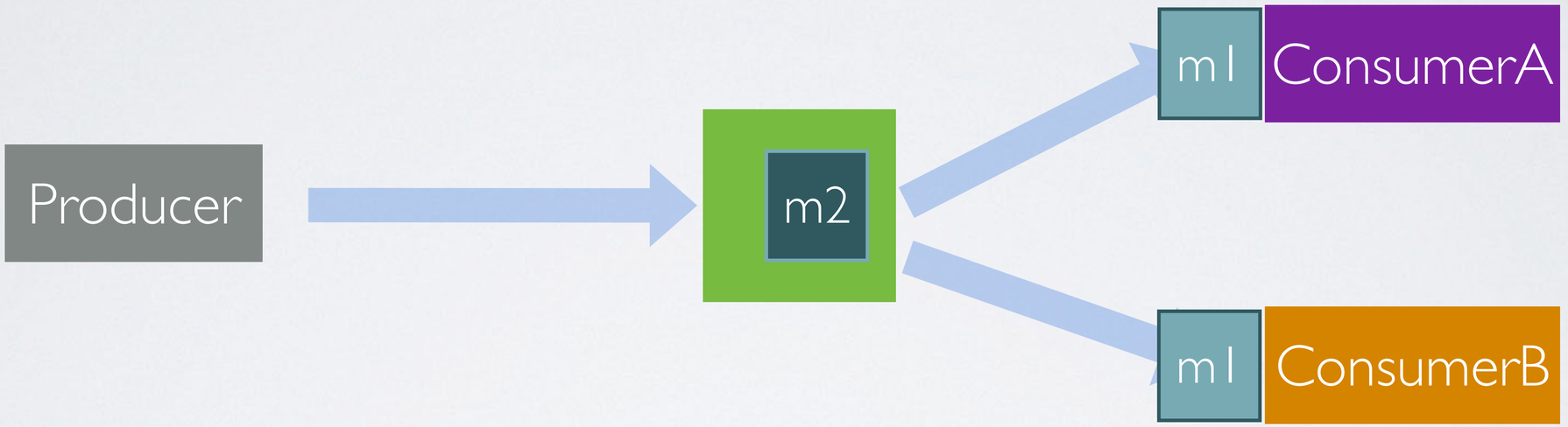
de-couple

BROADCAST



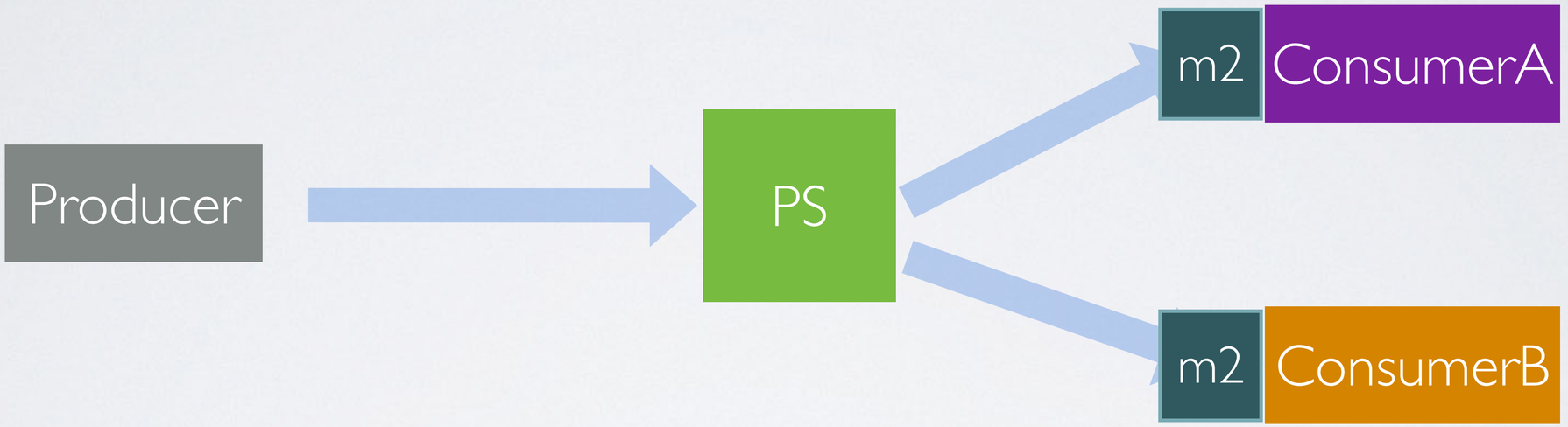
de-couple

BROADCAST



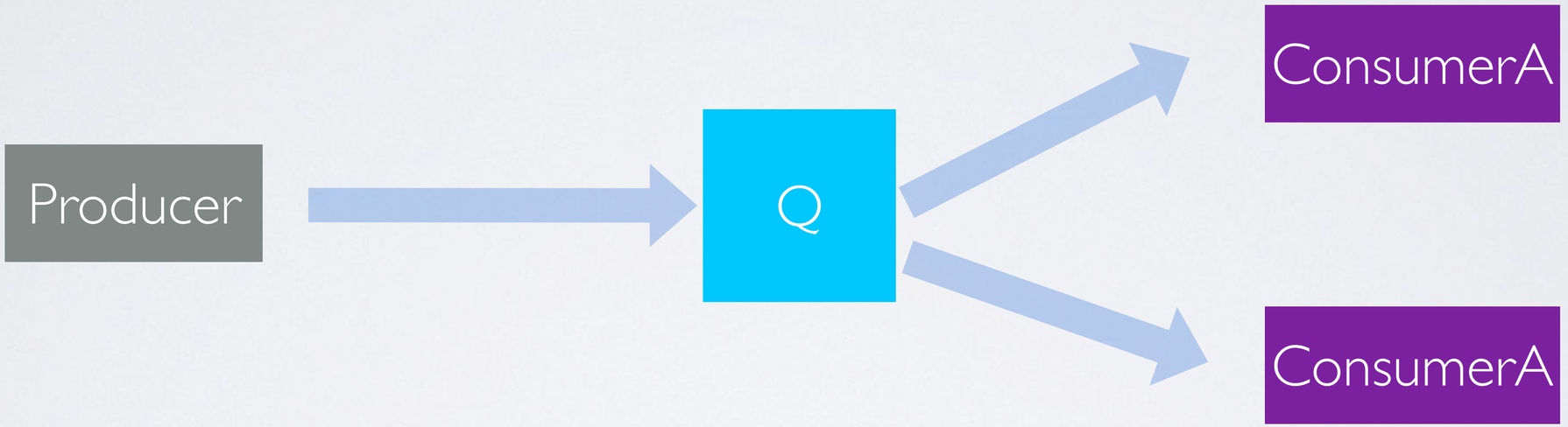
de-couple

BROADCAST



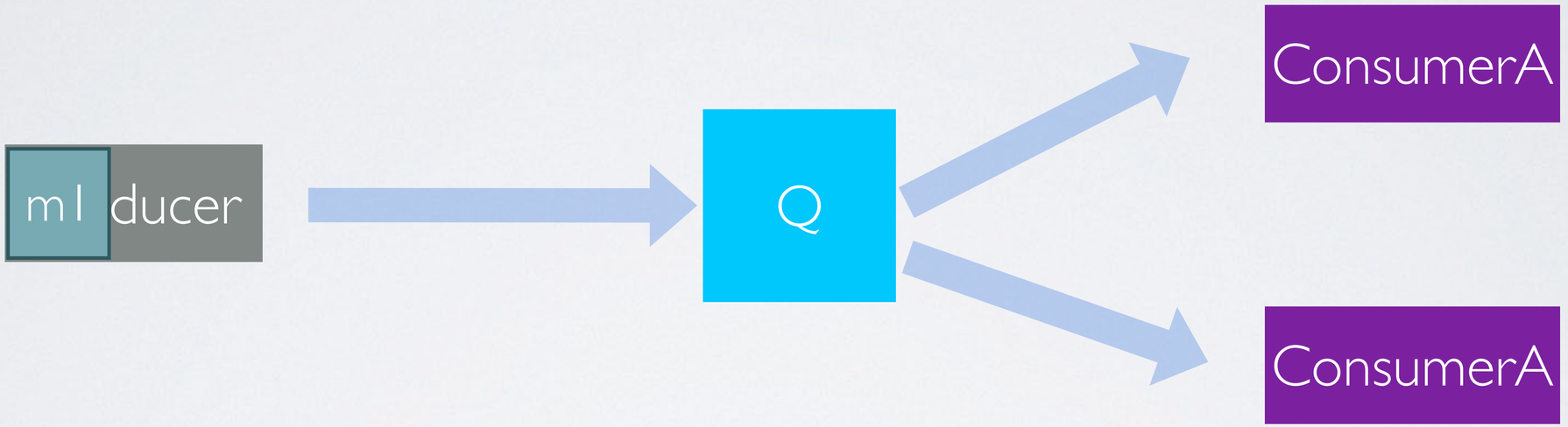
de-couple

DISTRIBUTION



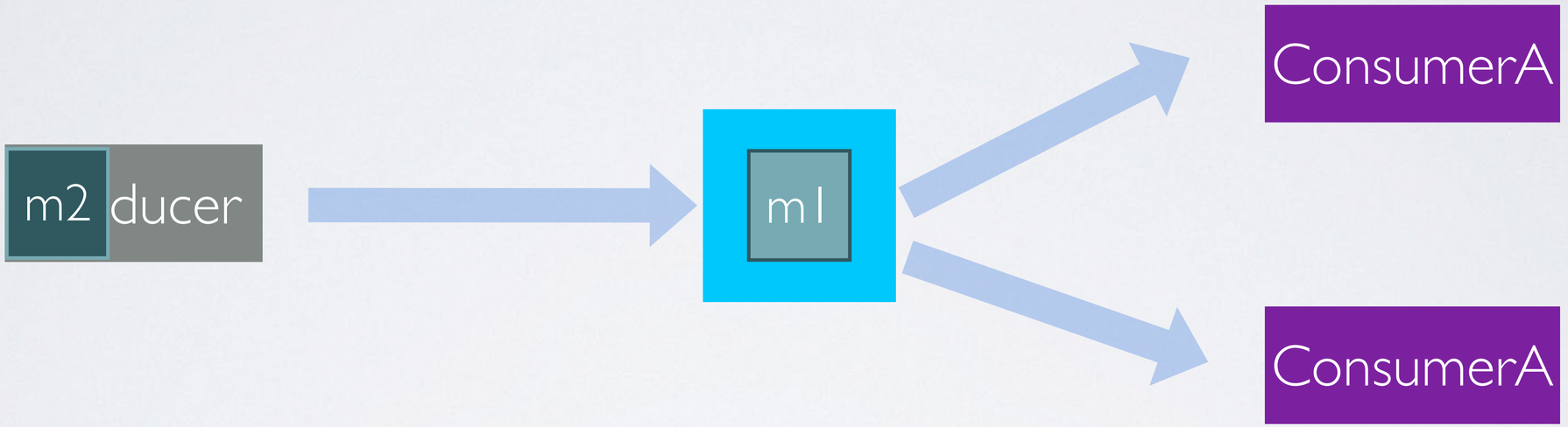
horizontal scalability

DISTRIBUTION



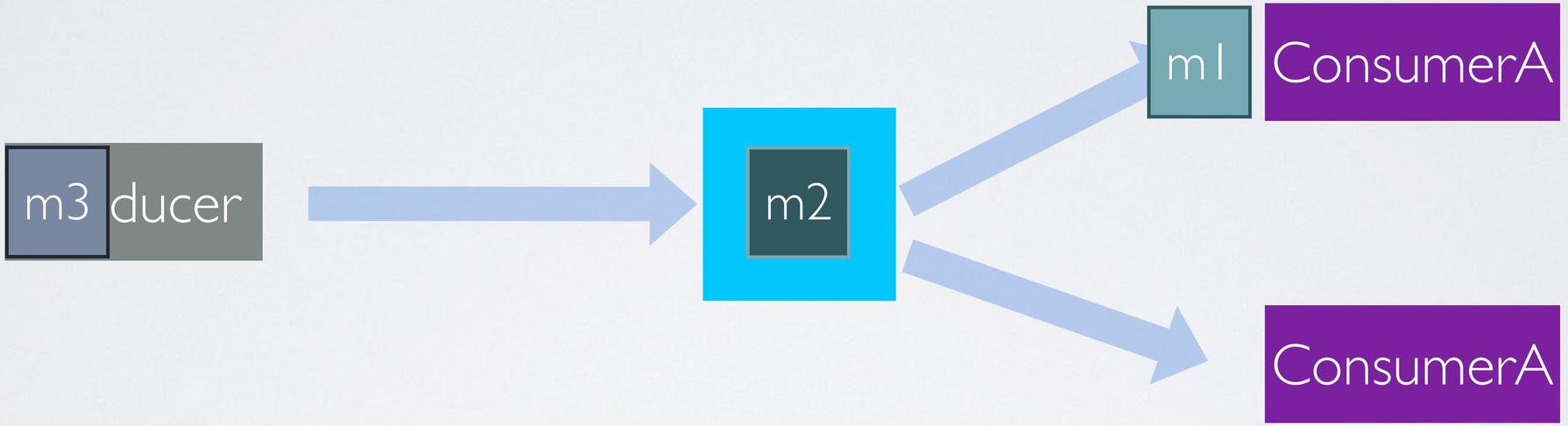
horizontal scalability

DISTRIBUTION



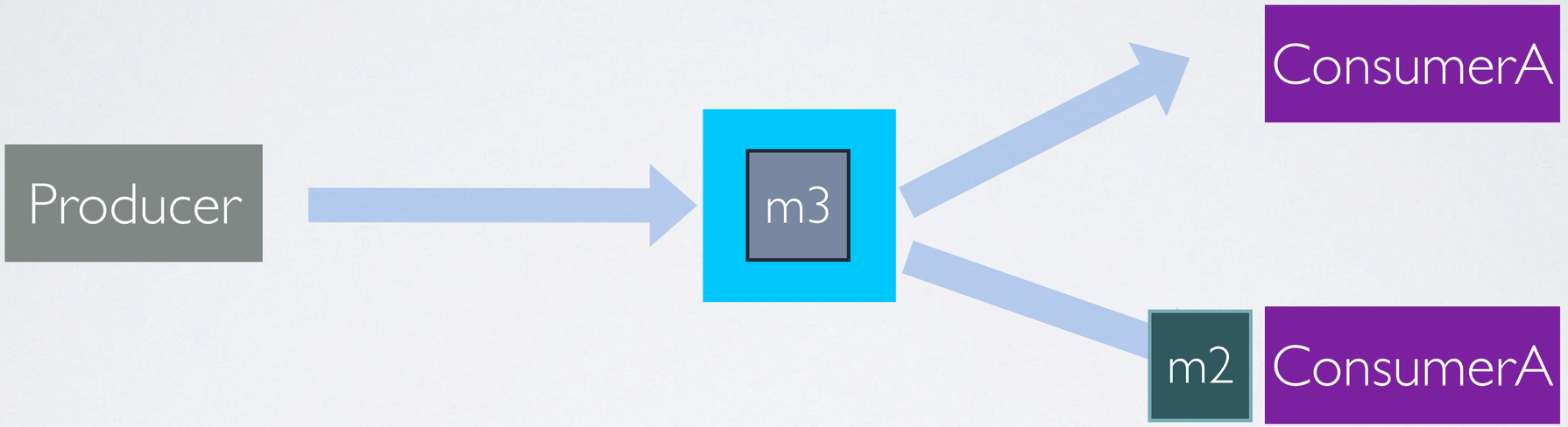
horizontal scalability

DISTRIBUTION



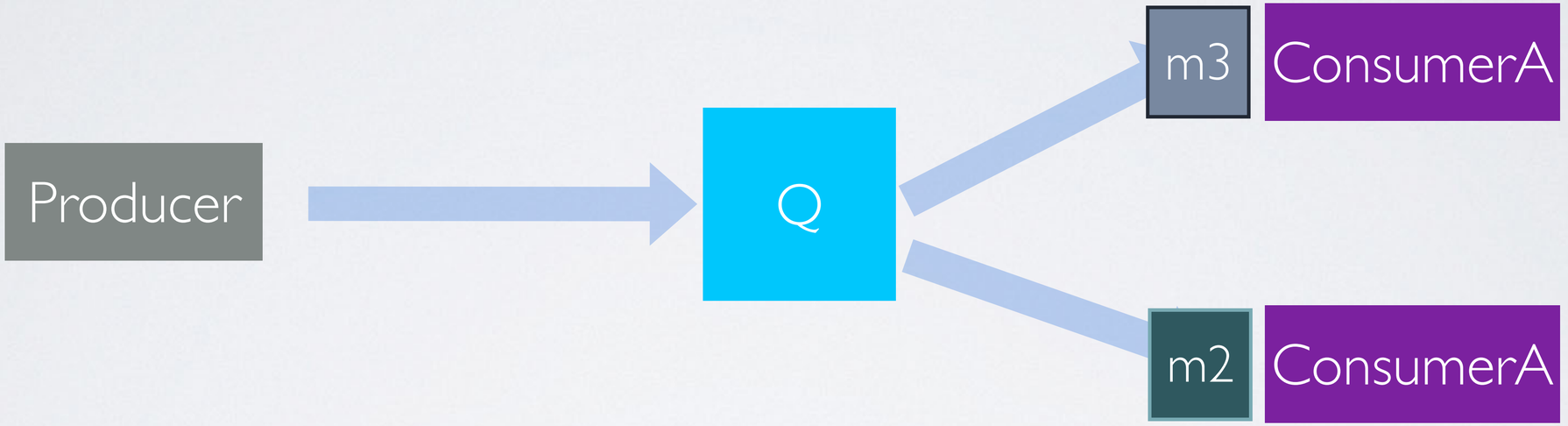
horizontal scalability

DISTRIBUTION



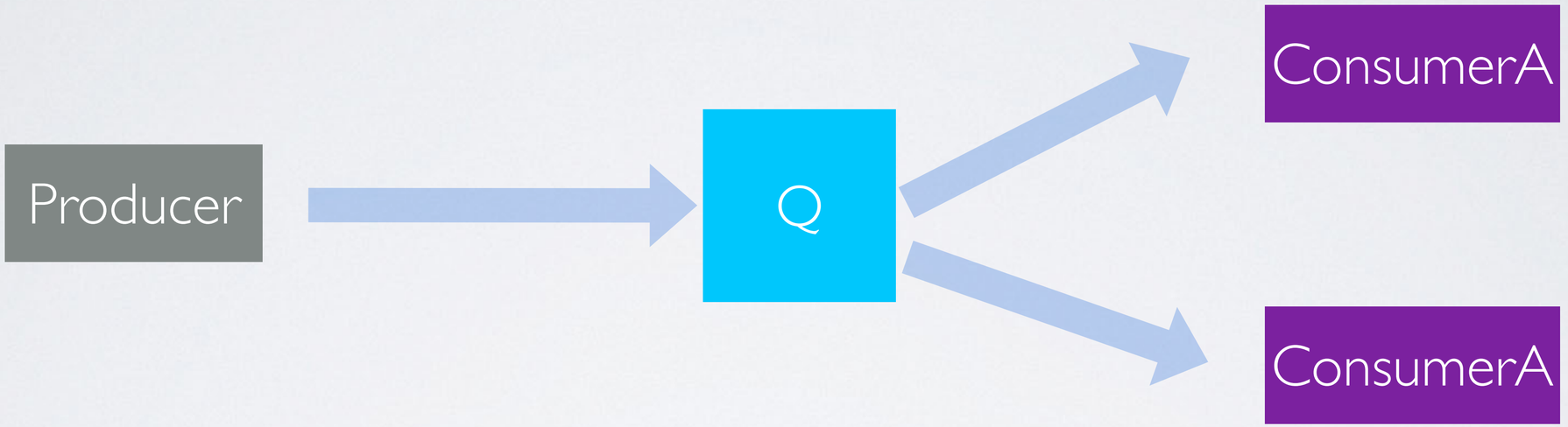
horizontal scalability

DISTRIBUTION



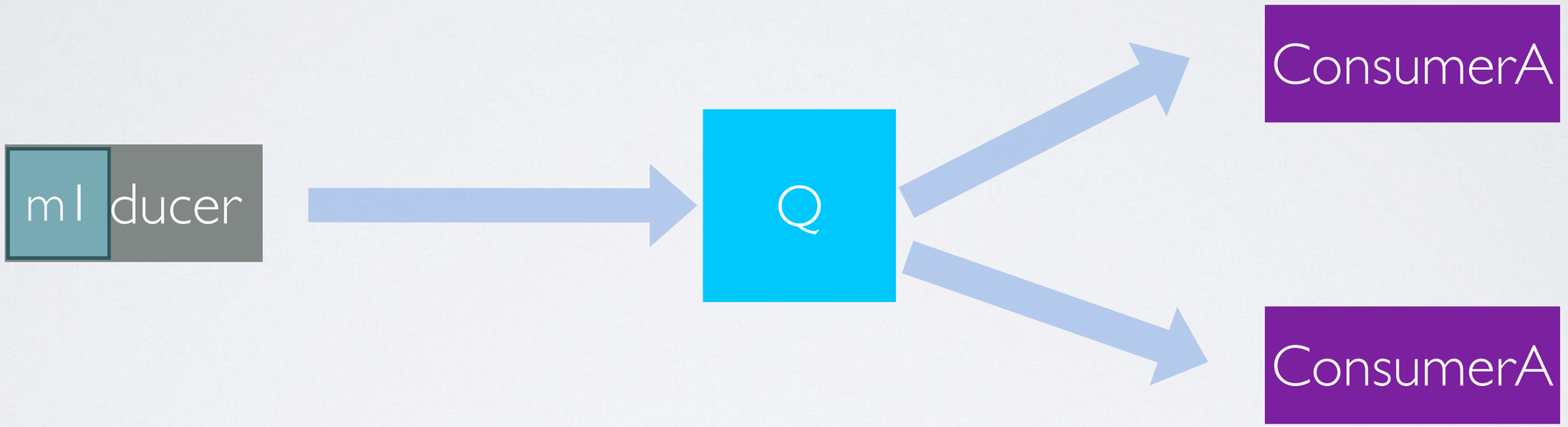
horizontal scalability

FAILURE



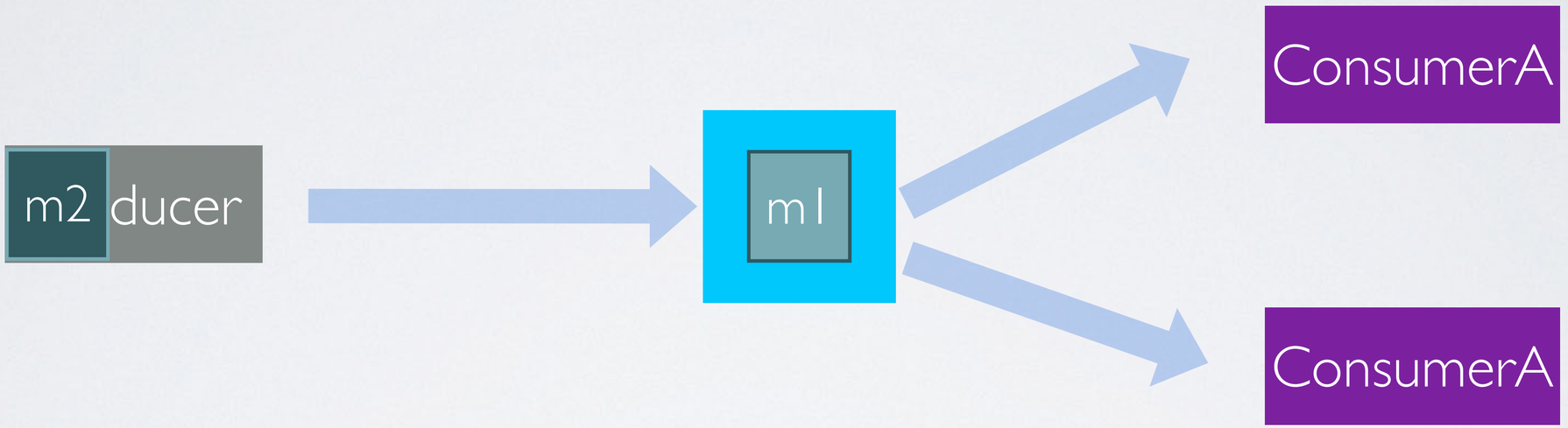
fault tolerance

FAILURE



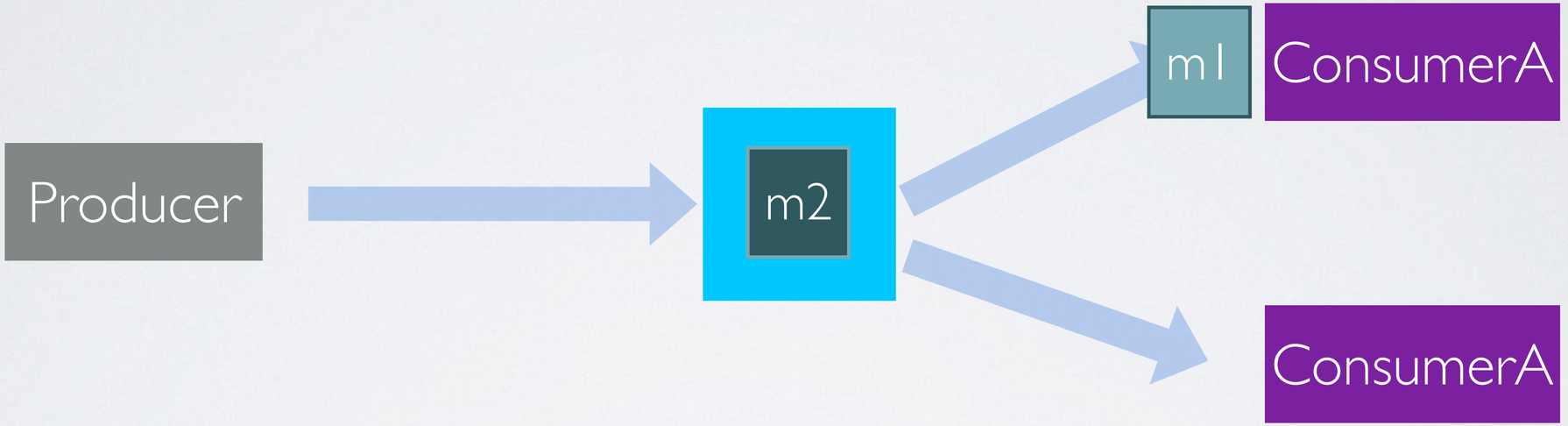
fault tolerance

FAILURE



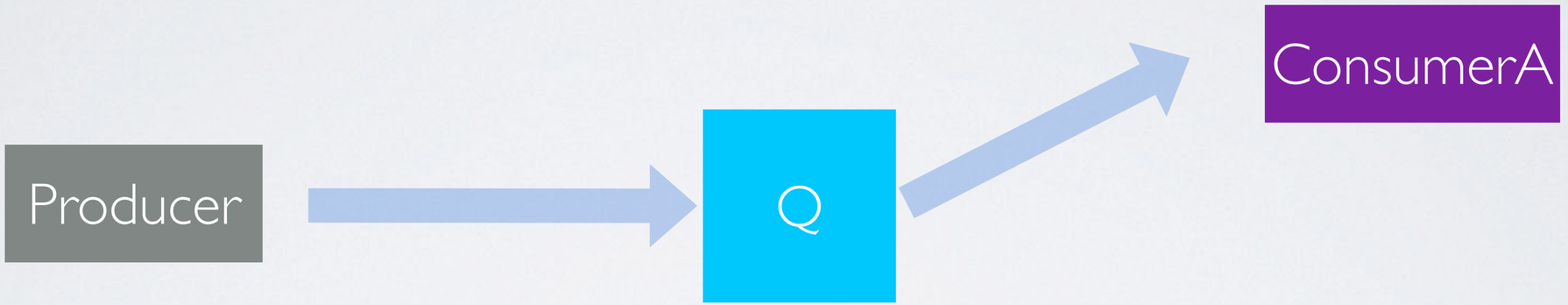
fault tolerance

FAILURE



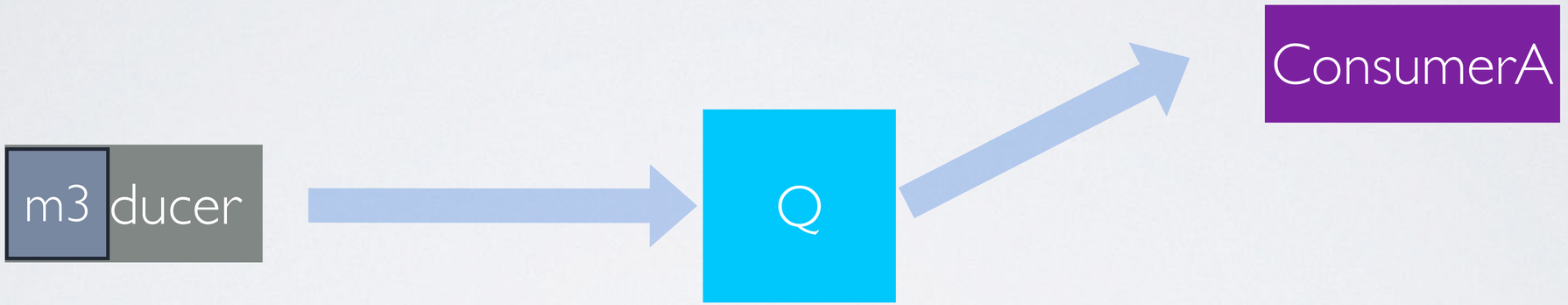
fault tolerance

FAILURE



fault tolerance

FAILURE

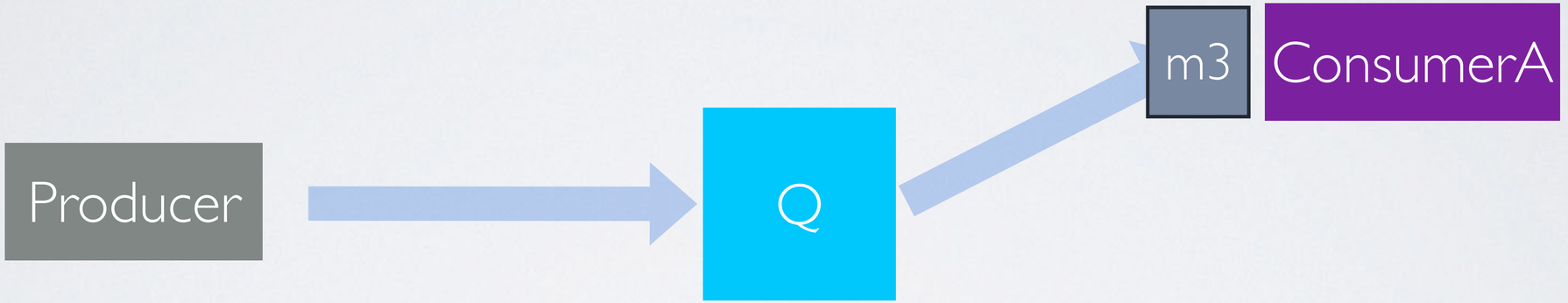


fault tolerance

FAILURE

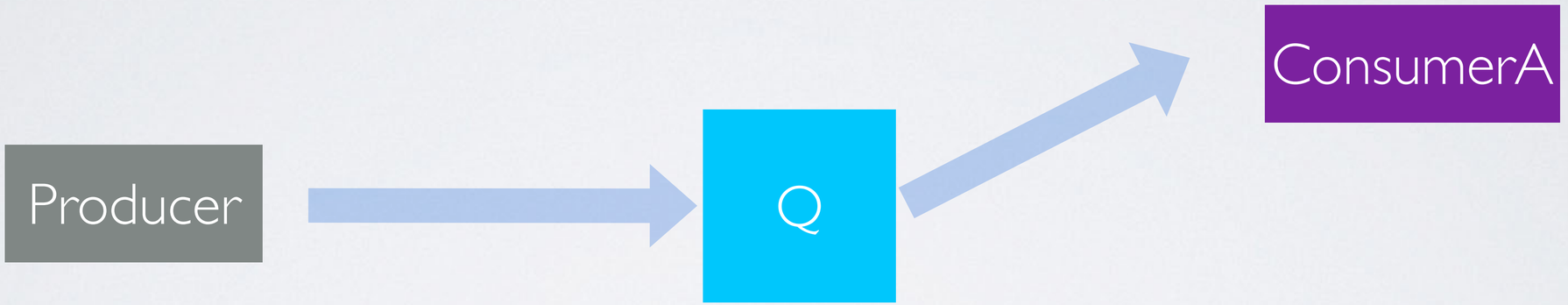
fault tolerance

FAILURE



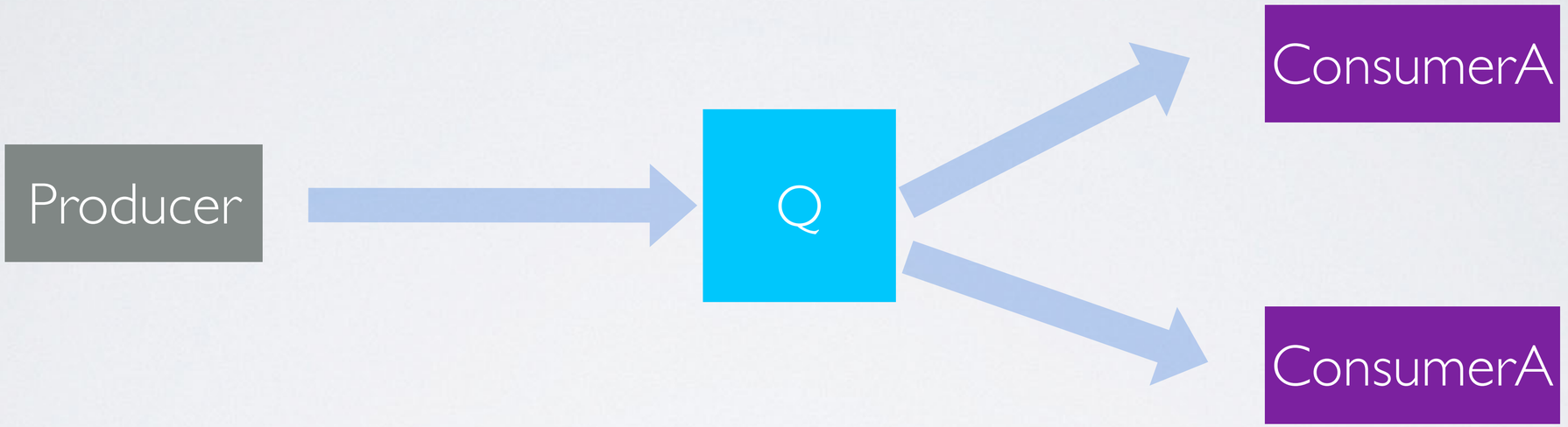
fault tolerance

FAILURE



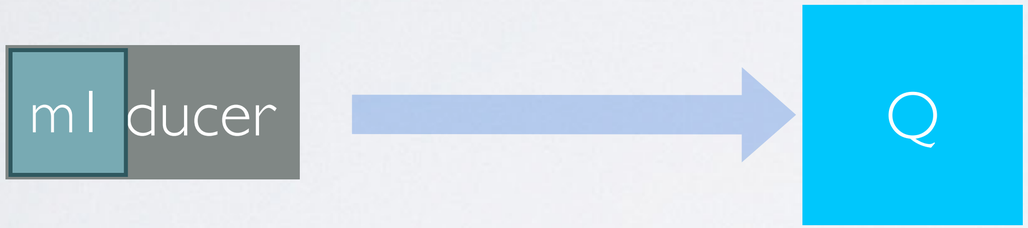
fault tolerance

EVEN MORE FAILURE



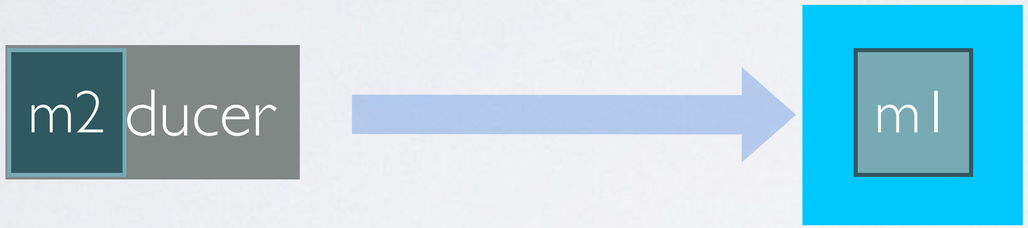
welp

EVEN MORE FAILURE



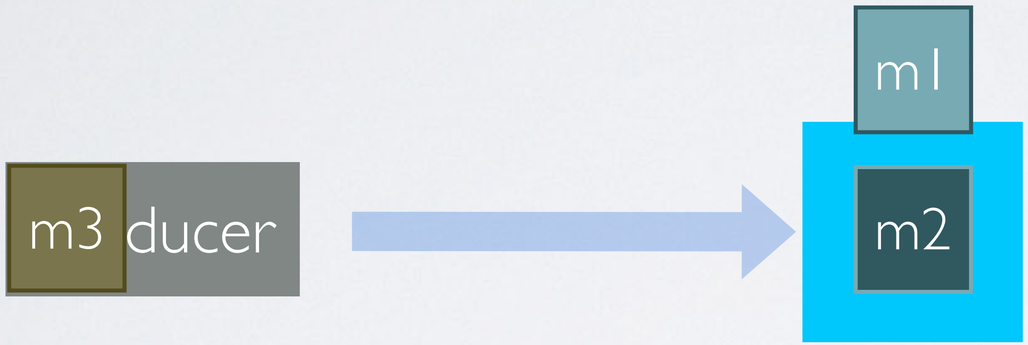
welp

EVEN MORE FAILURE



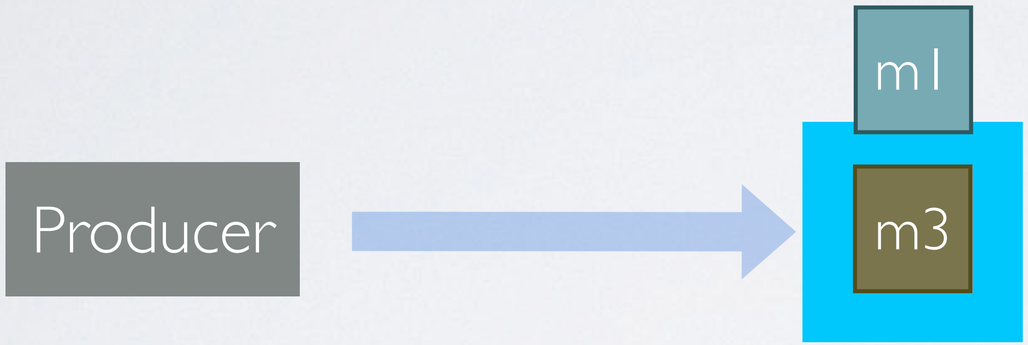
welp

EVEN MORE FAILURE



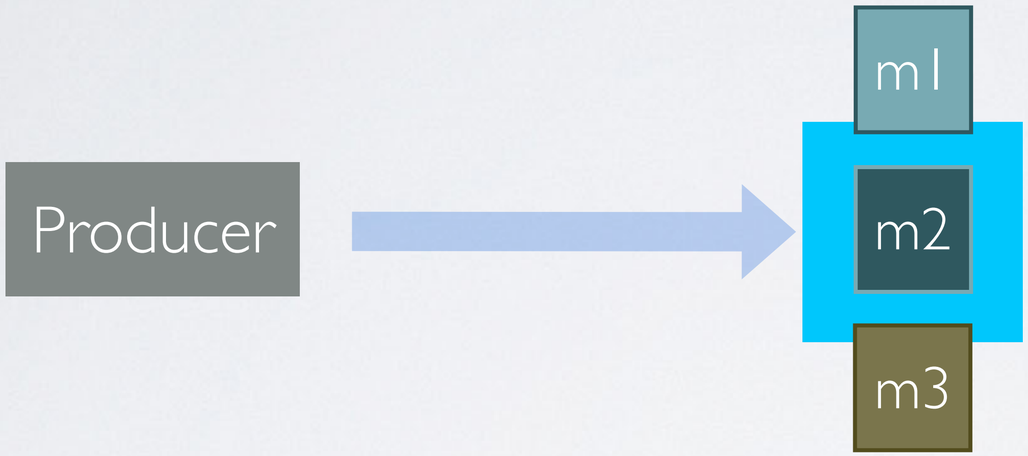
welp

EVEN MORE FAILURE



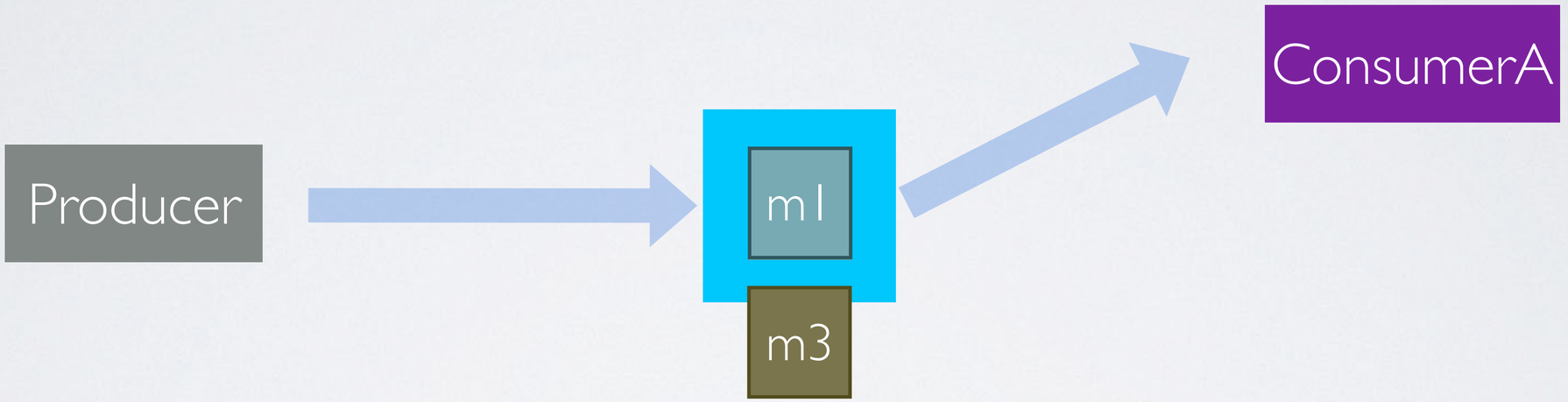
welp

EVEN MORE FAILURE



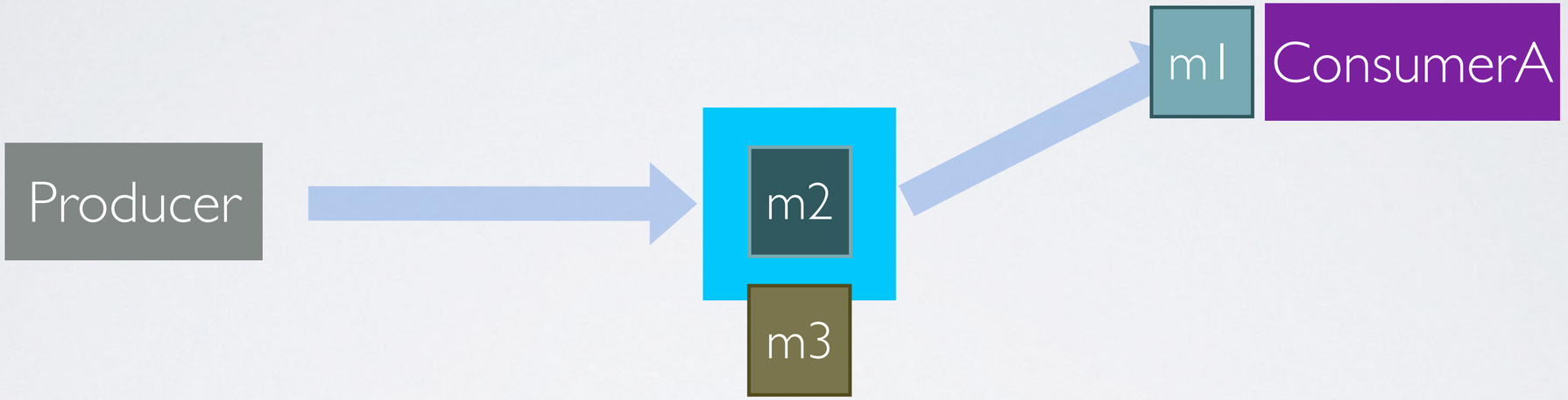
welp

EVEN MORE FAILURE



welp

EVEN MORE FAILURE

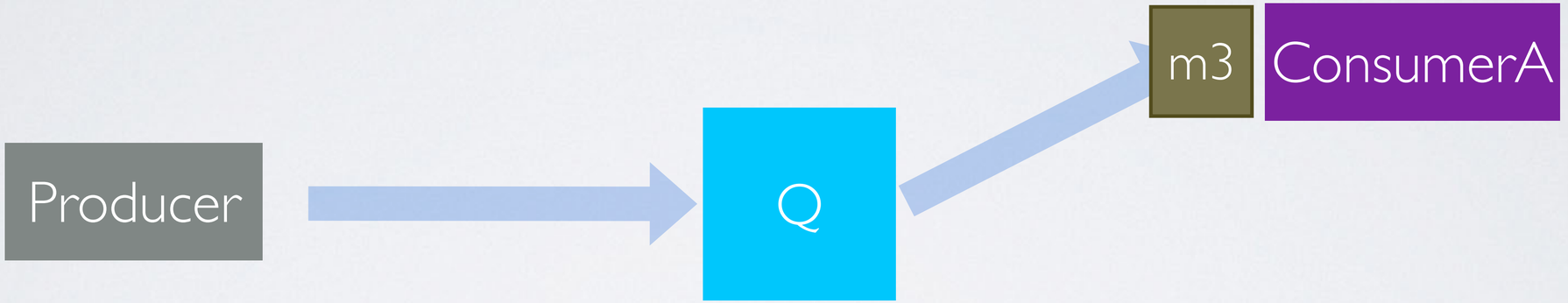


welp

EVEN MORE FAILURE

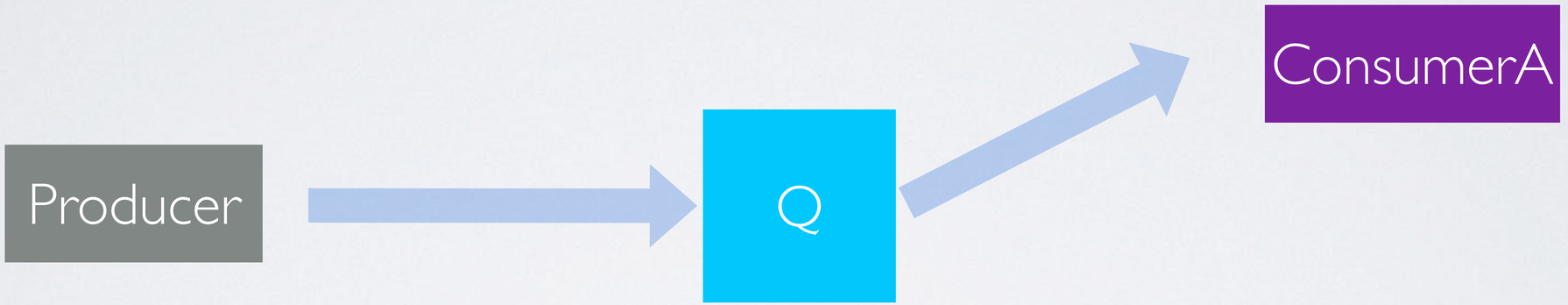
welp

EVEN MORE FAILURE



welp

EVEN MORE FAILURE



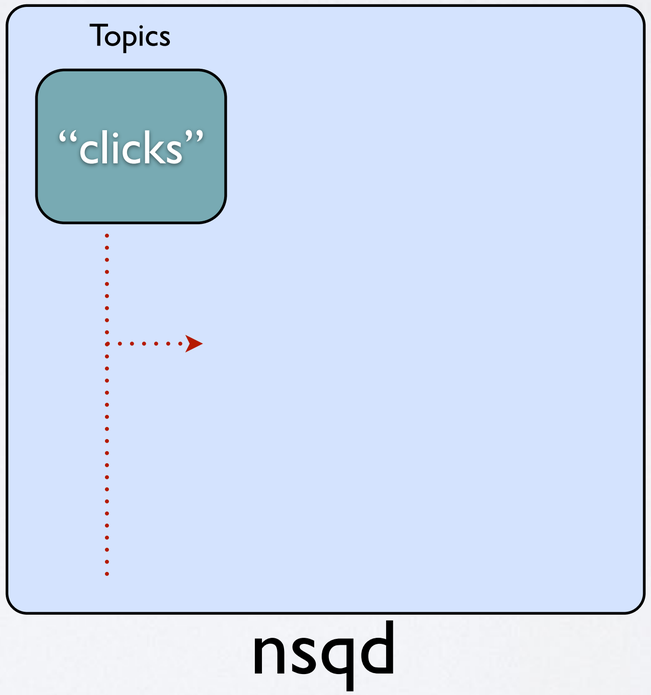
welp

NSQD

TOPICS AND CHANNELS

combine pubsub, distribution, and queueing

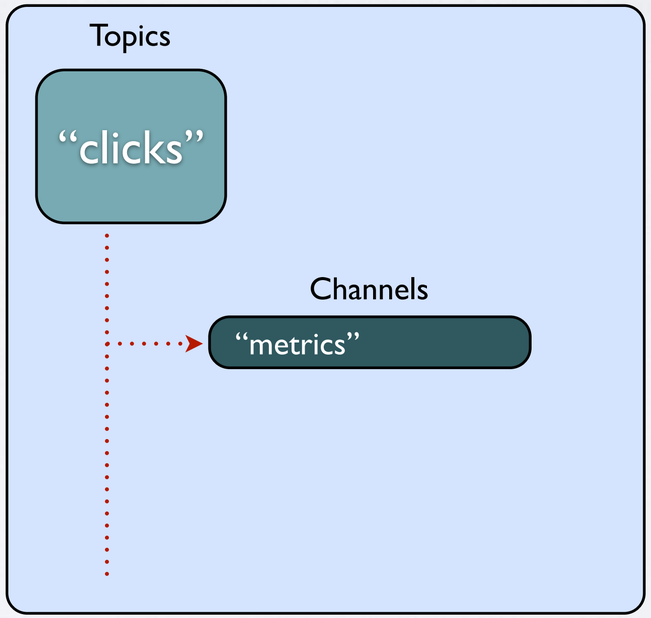
- a **topic** is a distinct stream of messages
- a **topic** has one or more **channels**
- topics and channels are created at **runtime**
- messages are **pushed** to consumers



TOPICS AND CHANNELS

combine pubsub, distribution, and queueing

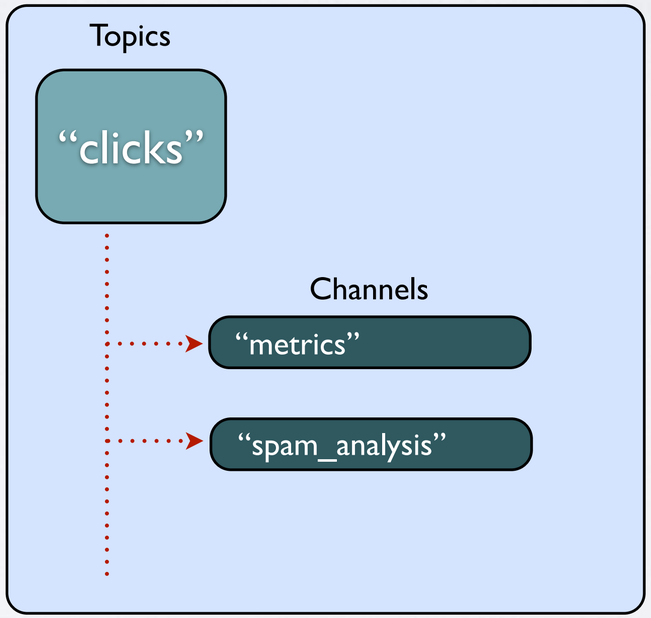
- a **topic** is a distinct stream of messages
- a **topic** has one or more **channels**
- topics and channels are created at **runtime**
- messages are **pushed** to consumers



TOPICS AND CHANNELS

combine pubsub, distribution, and queueing

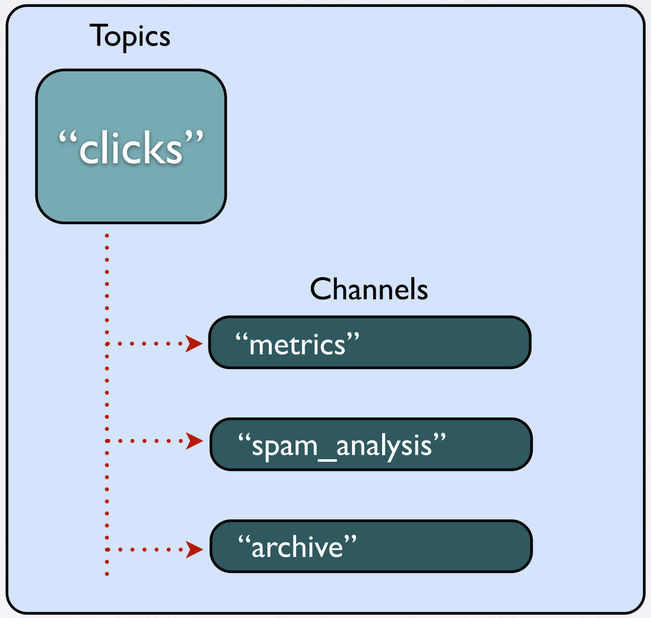
- a **topic** is a distinct stream of messages
- a **topic** has one or more **channels**
- topics and channels are created at **runtime**
- messages are **pushed** to consumers



TOPICS AND CHANNELS

combine pubsub, distribution, and queueing

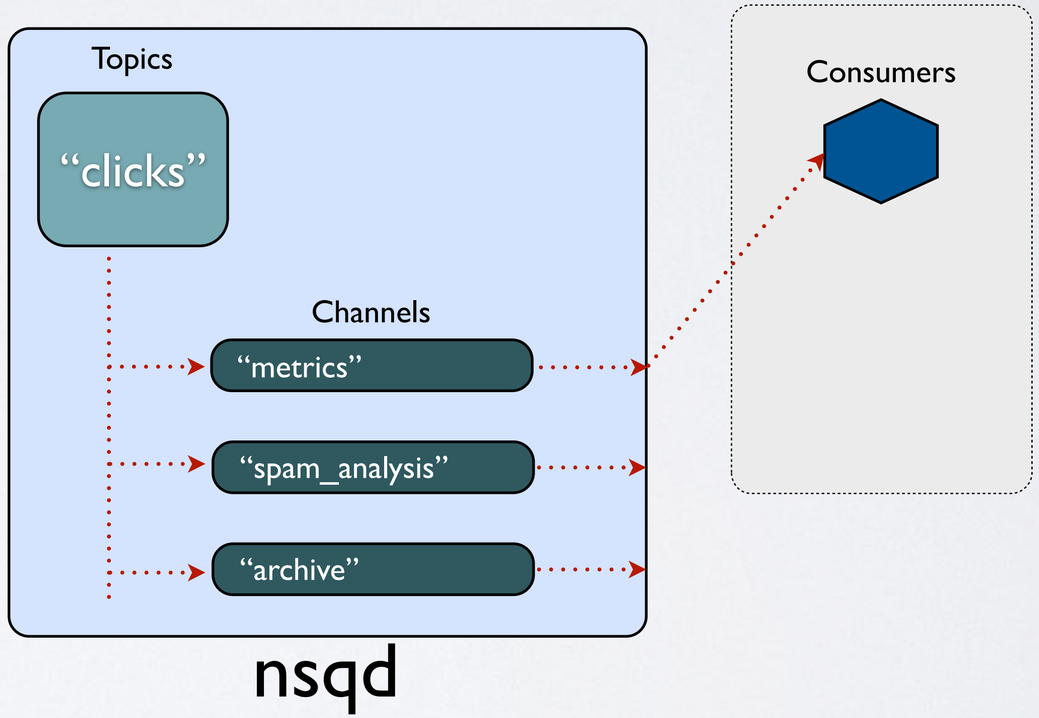
- a **topic** is a distinct stream of messages
- a **topic** has one or more **channels**
- topics and channels are created at **runtime**
- messages are **pushed** to consumers



TOPICS AND CHANNELS

combine pubsub, distribution, and queueing

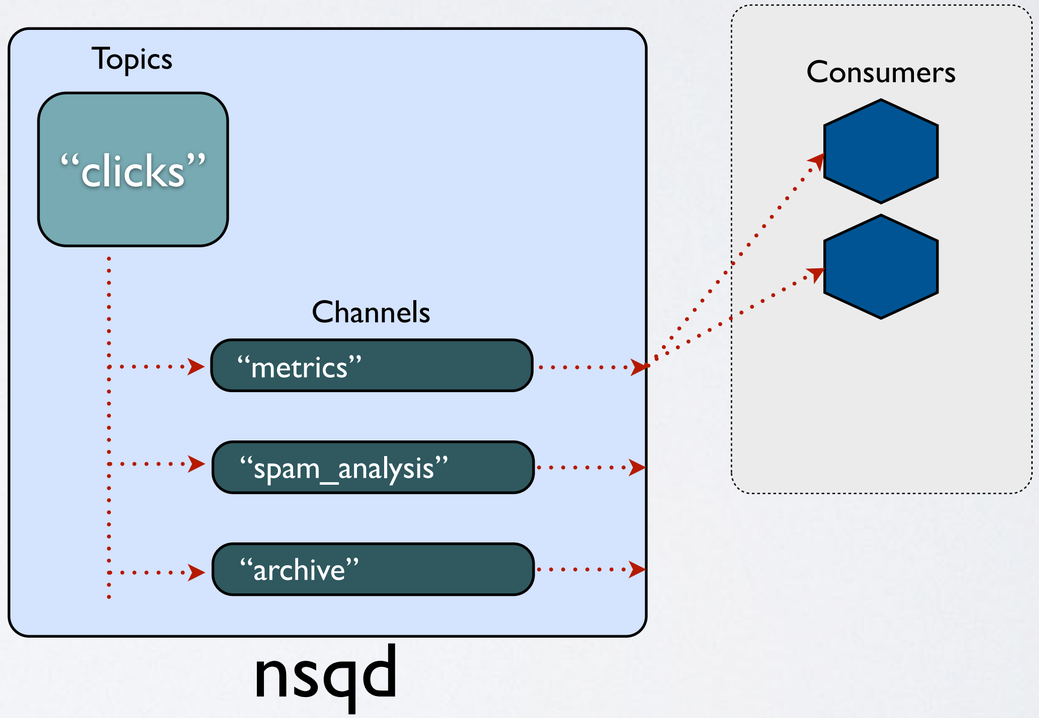
- a **topic** is a distinct stream of messages
- a **topic** has one or more **channels**
- topics and channels are created at **runtime**
- messages are **pushed** to consumers



TOPICS AND CHANNELS

combine pubsub, distribution, and queueing

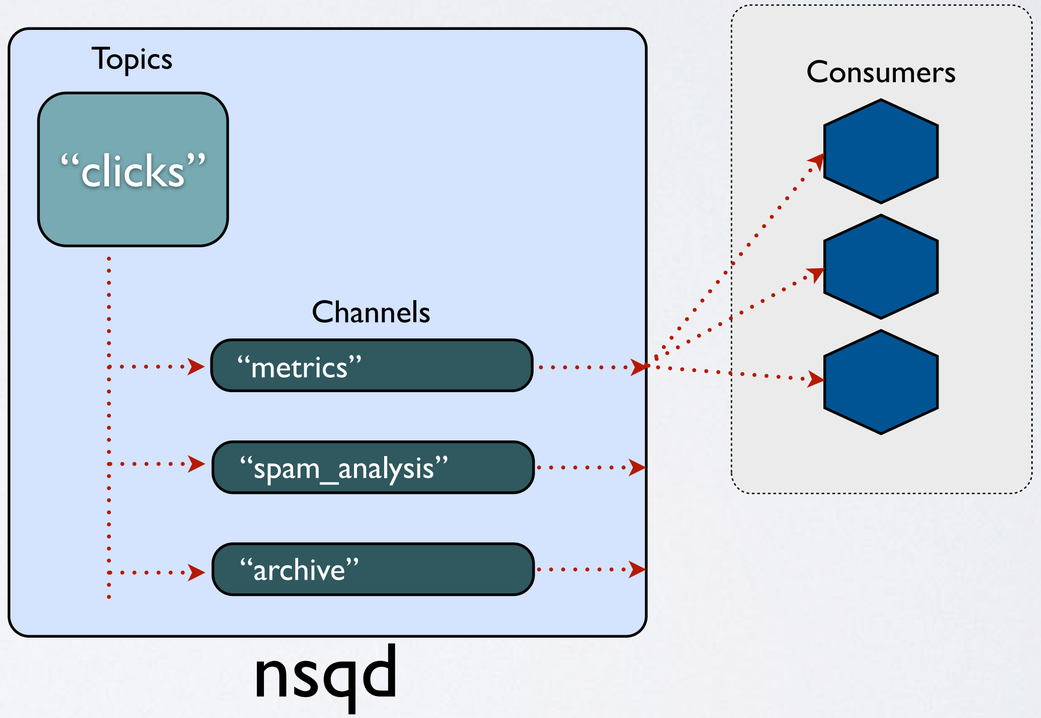
- a **topic** is a distinct stream of messages
- a **topic** has one or more **channels**
- topics and channels are created at **runtime**
- messages are **pushed** to consumers



TOPICS AND CHANNELS

combine pubsub, distribution, and queueing

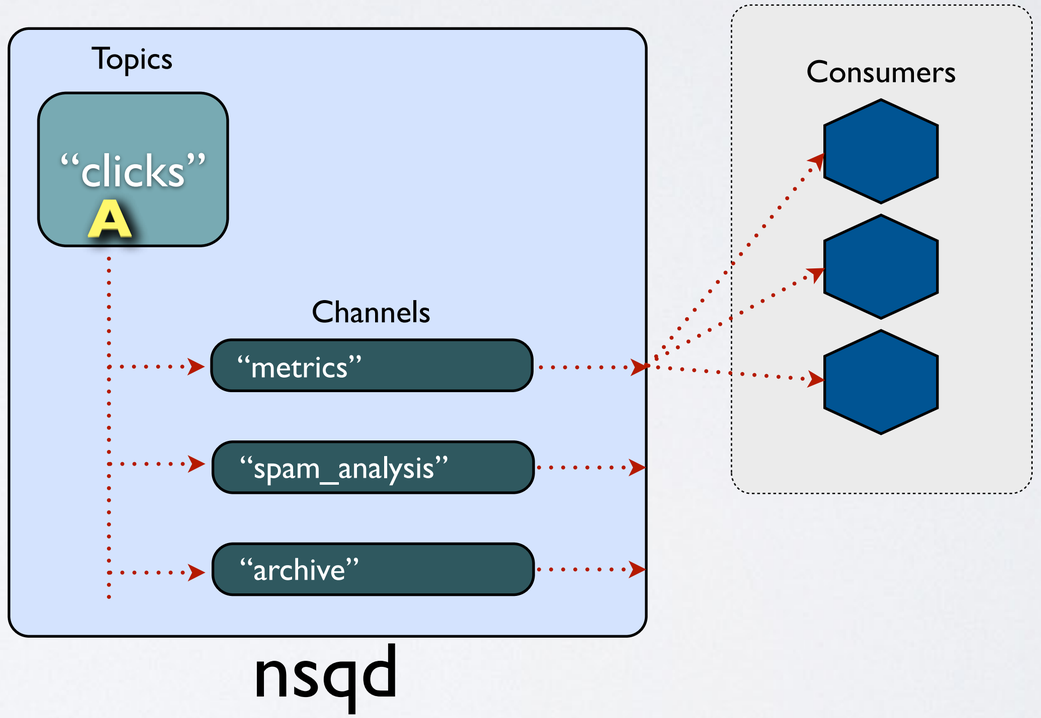
- a **topic** is a distinct stream of messages
- a **topic** has one or more **channels**
- topics and channels are created at **runtime**
- messages are **pushed** to consumers



TOPICS AND CHANNELS

combine pubsub, distribution, and queueing

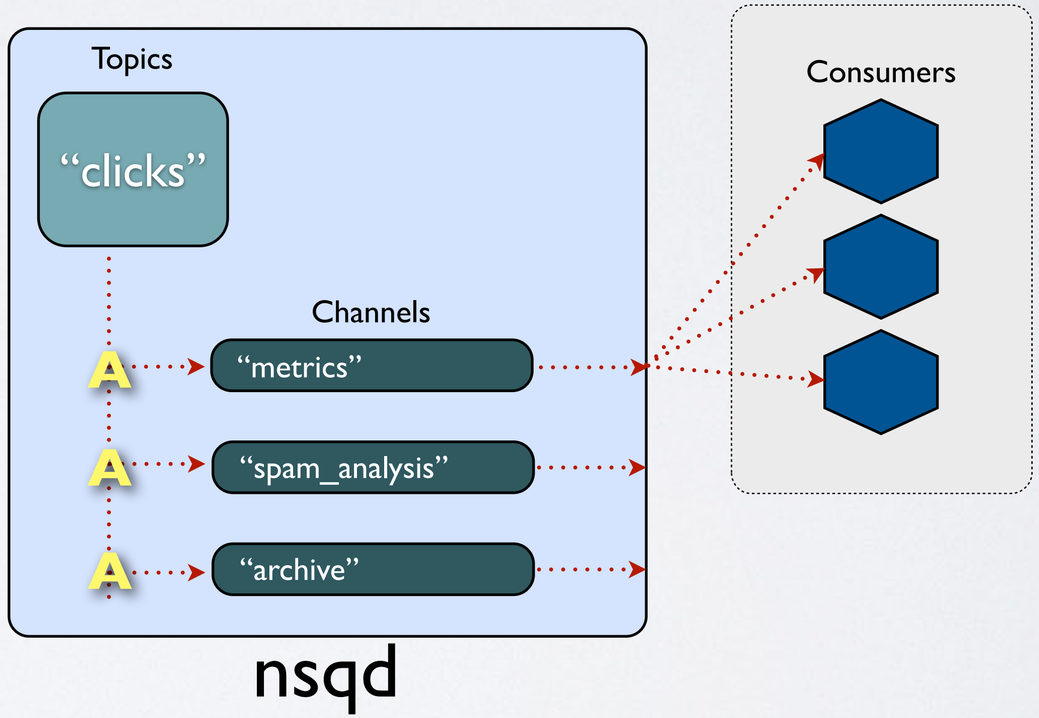
- a **topic** is a distinct stream of messages
- a **topic** has one or more **channels**
- topics and channels are created at **runtime**
- messages are **pushed** to consumers



TOPICS AND CHANNELS

combine pubsub, distribution, and queueing

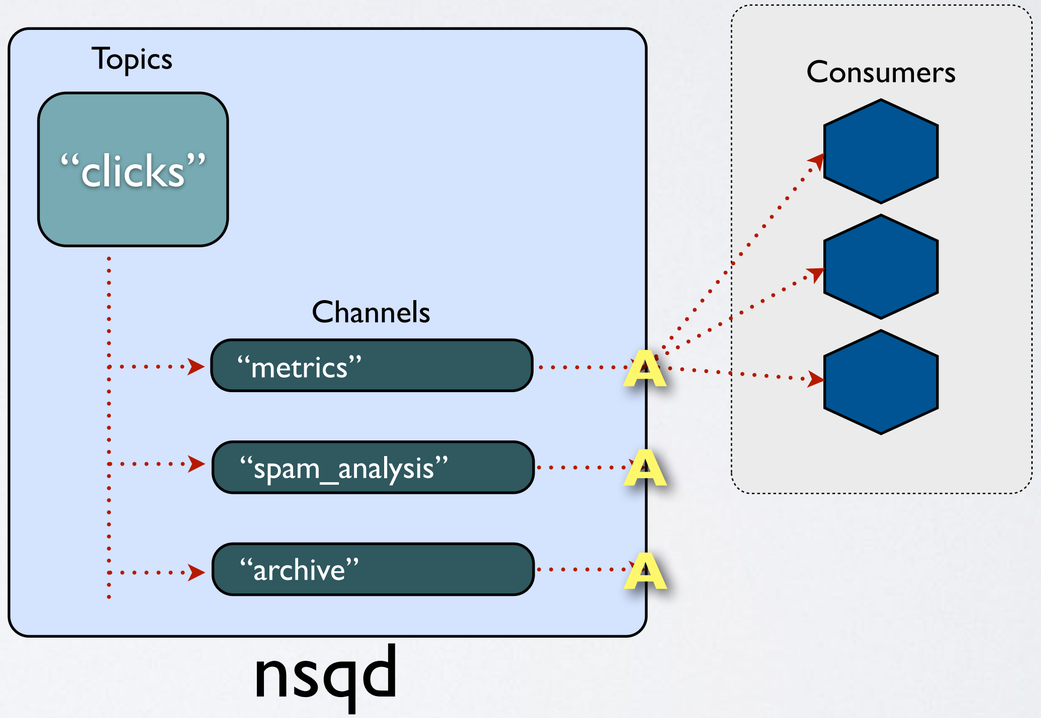
- a **topic** is a distinct stream of messages
- a **topic** has one or more **channels**
- topics and channels are created at **runtime**
- messages are **pushed** to consumers



TOPICS AND CHANNELS

combine pubsub, distribution, and queueing

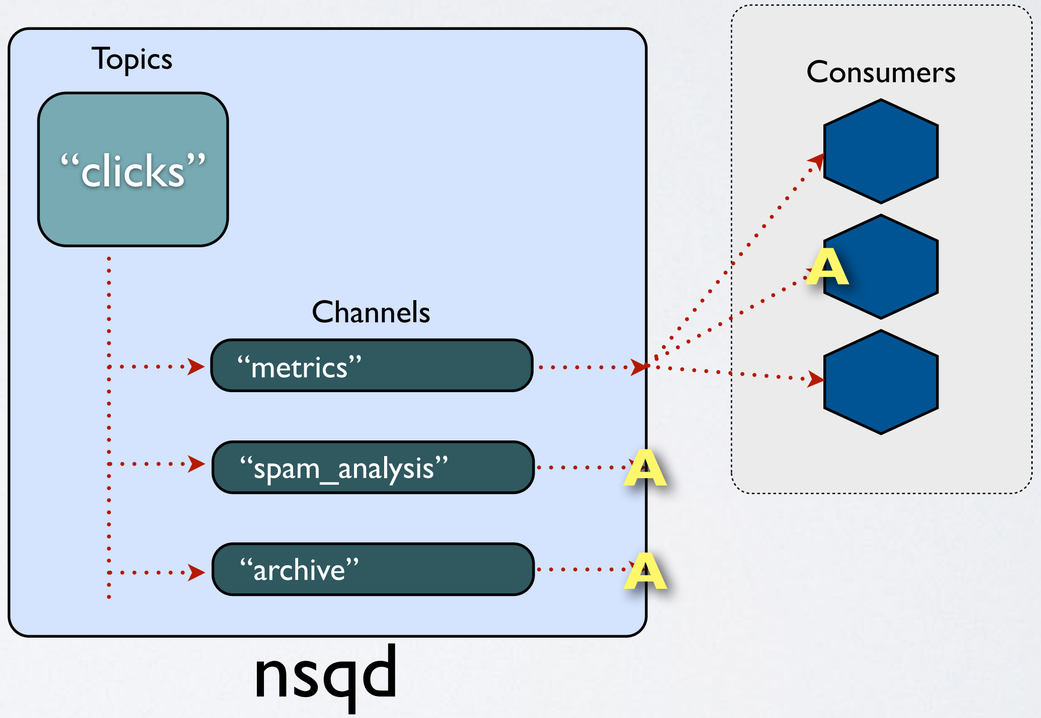
- a **topic** is a distinct stream of messages
- a **topic** has one or more **channels**
- topics and channels are created at **runtime**
- messages are **pushed** to consumers



TOPICS AND CHANNELS

combine pubsub, distribution, and queueing

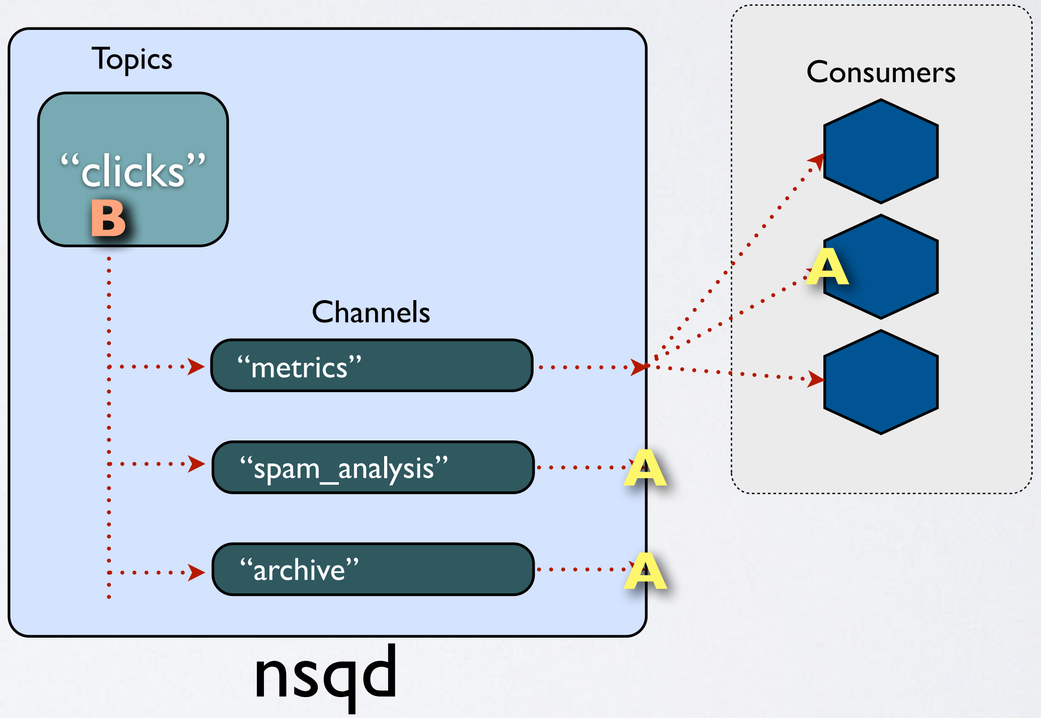
- a **topic** is a distinct stream of messages
- a **topic** has one or more **channels**
- topics and channels are created at **runtime**
- messages are **pushed** to consumers



TOPICS AND CHANNELS

combine pubsub, distribution, and queueing

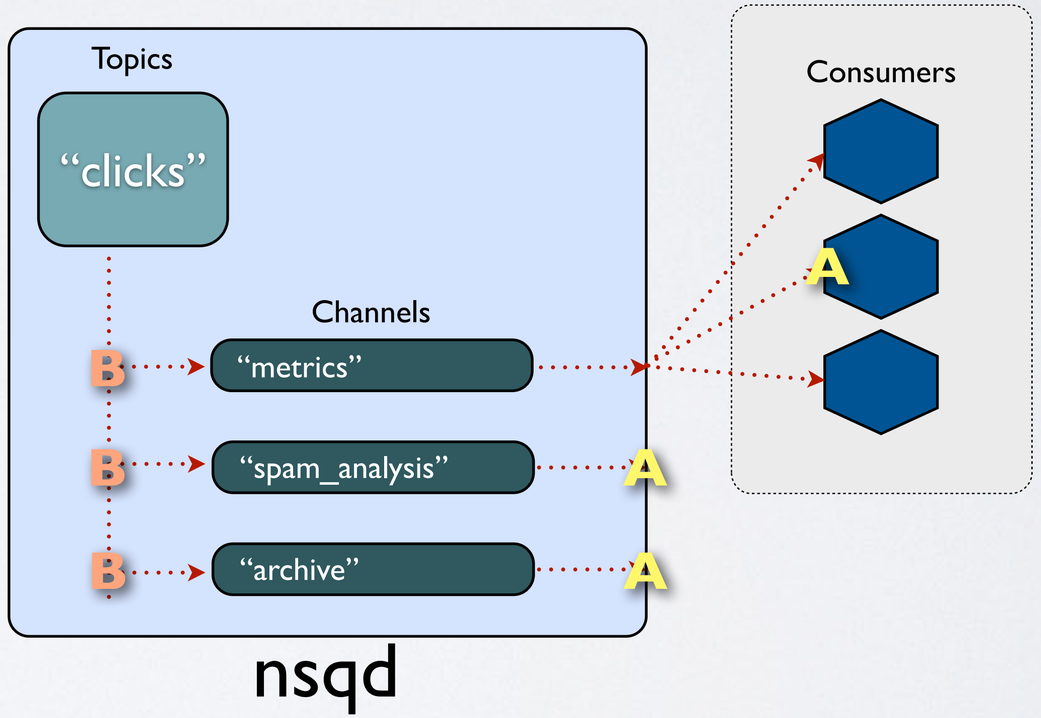
- a **topic** is a distinct stream of messages
- a **topic** has one or more **channels**
- topics and channels are created at **runtime**
- messages are **pushed** to consumers



TOPICS AND CHANNELS

combine pubsub, distribution, and queueing

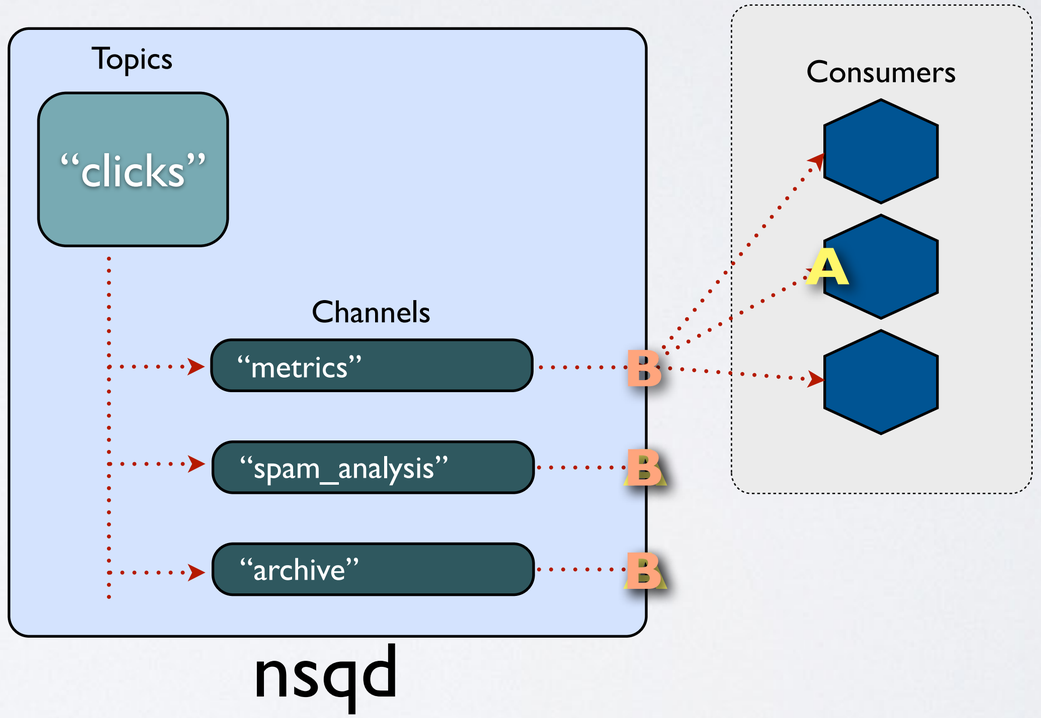
- a **topic** is a distinct stream of messages
- a **topic** has one or more **channels**
- topics and channels are created at **runtime**
- messages are **pushed** to consumers



TOPICS AND CHANNELS

combine pubsub, distribution, and queueing

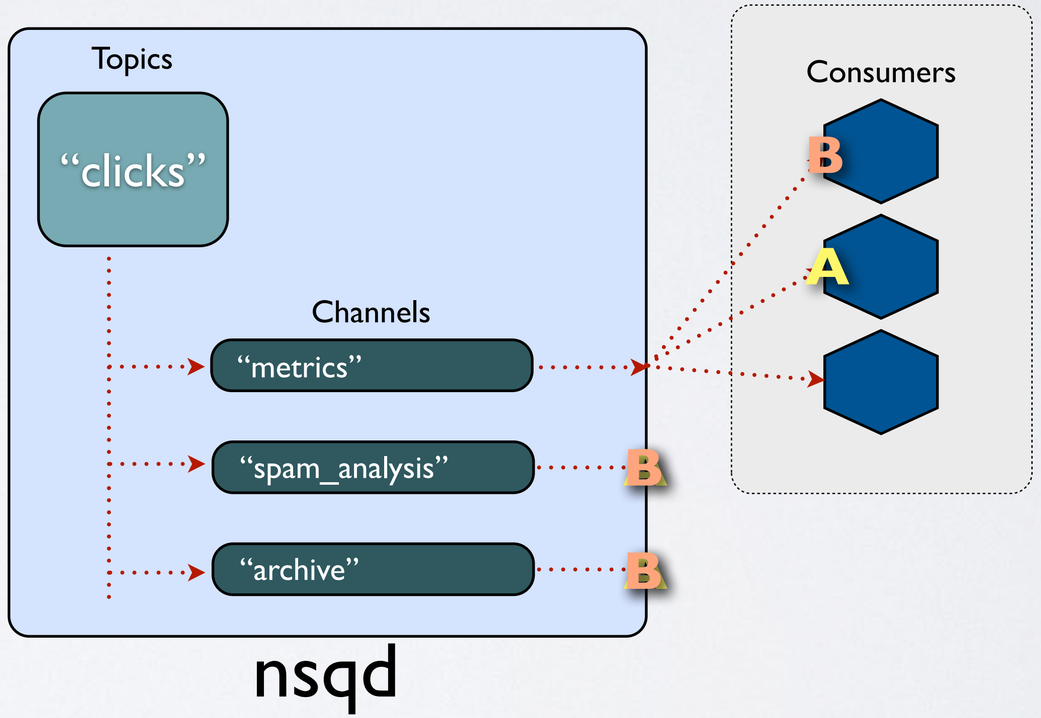
- a **topic** is a distinct stream of messages
- a **topic** has one or more **channels**
- topics and channels are created at **runtime**
- messages are **pushed** to consumers



TOPICS AND CHANNELS

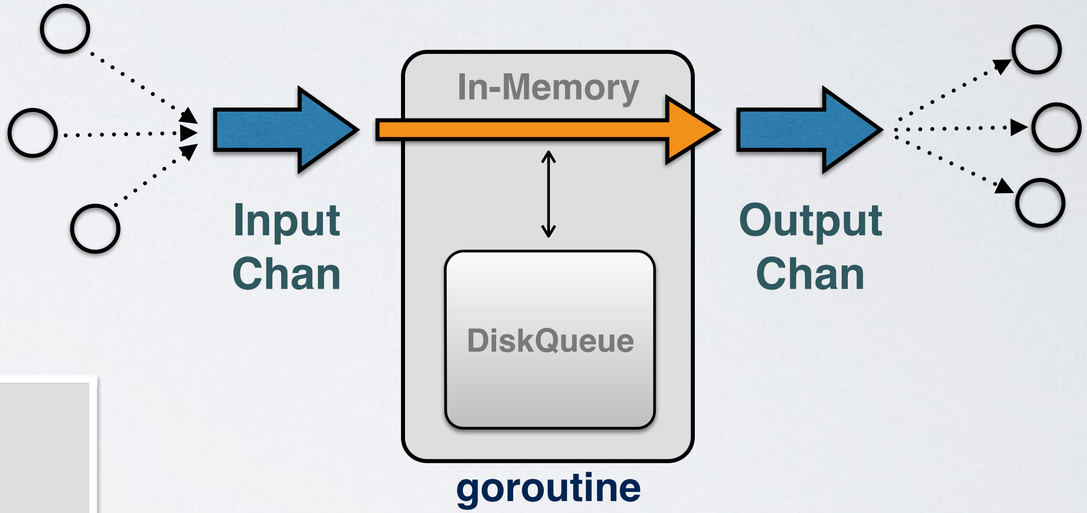
combine pubsub, distribution, and queueing

- a **topic** is a distinct stream of messages
- a **topic** has one or more **channels**
- topics and channels are created at **runtime**
- messages are **pushed** to consumers



UNDER THE HOOD

- **topics** and **channels** are *independent*
- configurable high water mark (disk persistence)
- "bounded" memory footprint

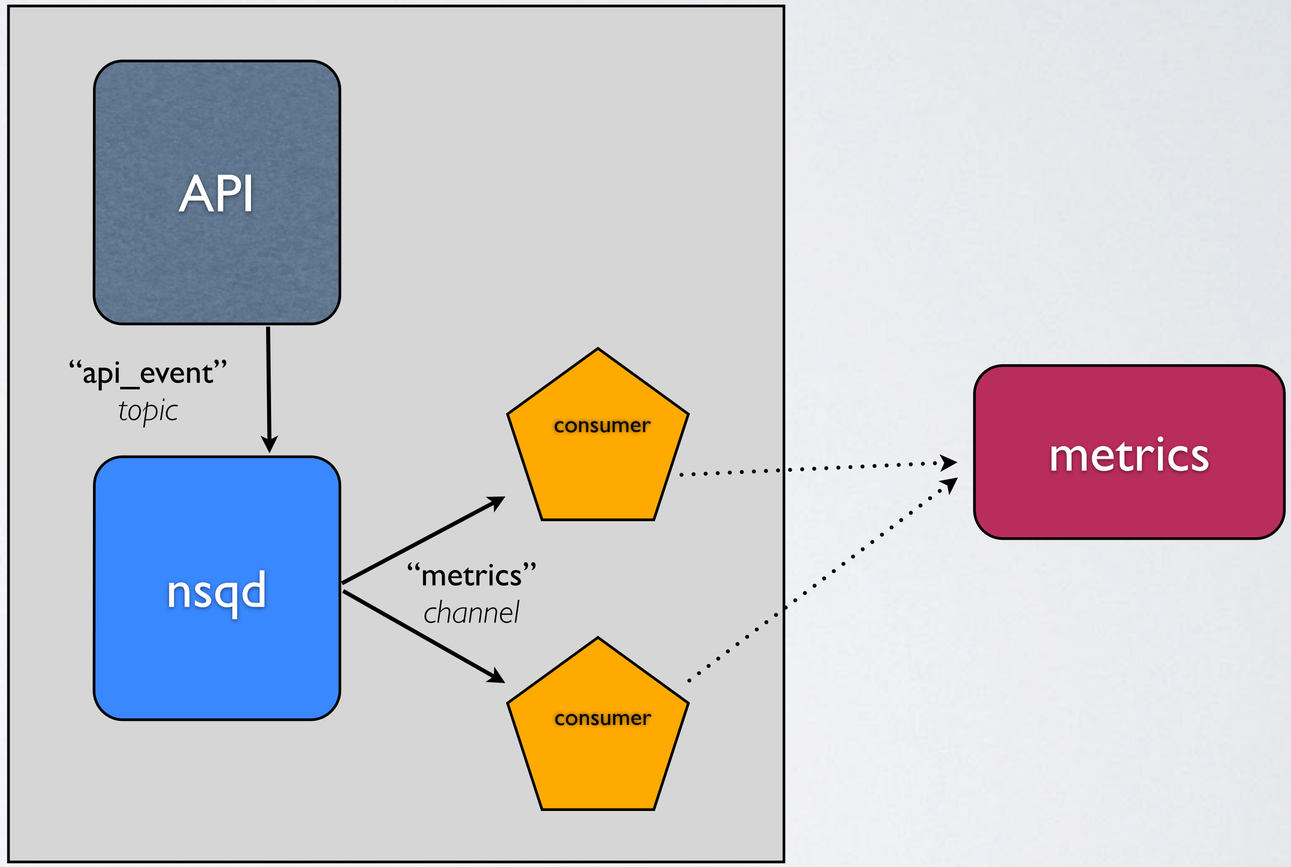



```
for msg := range c.incomingMsgChan {
    select {
    case c.memoryMsgChan <- msg:
    default:
        err := WriteMessageToBackend(&msgBuf, msg, c)
        if err != nil {
            // log whatever
        }
    }
}
```

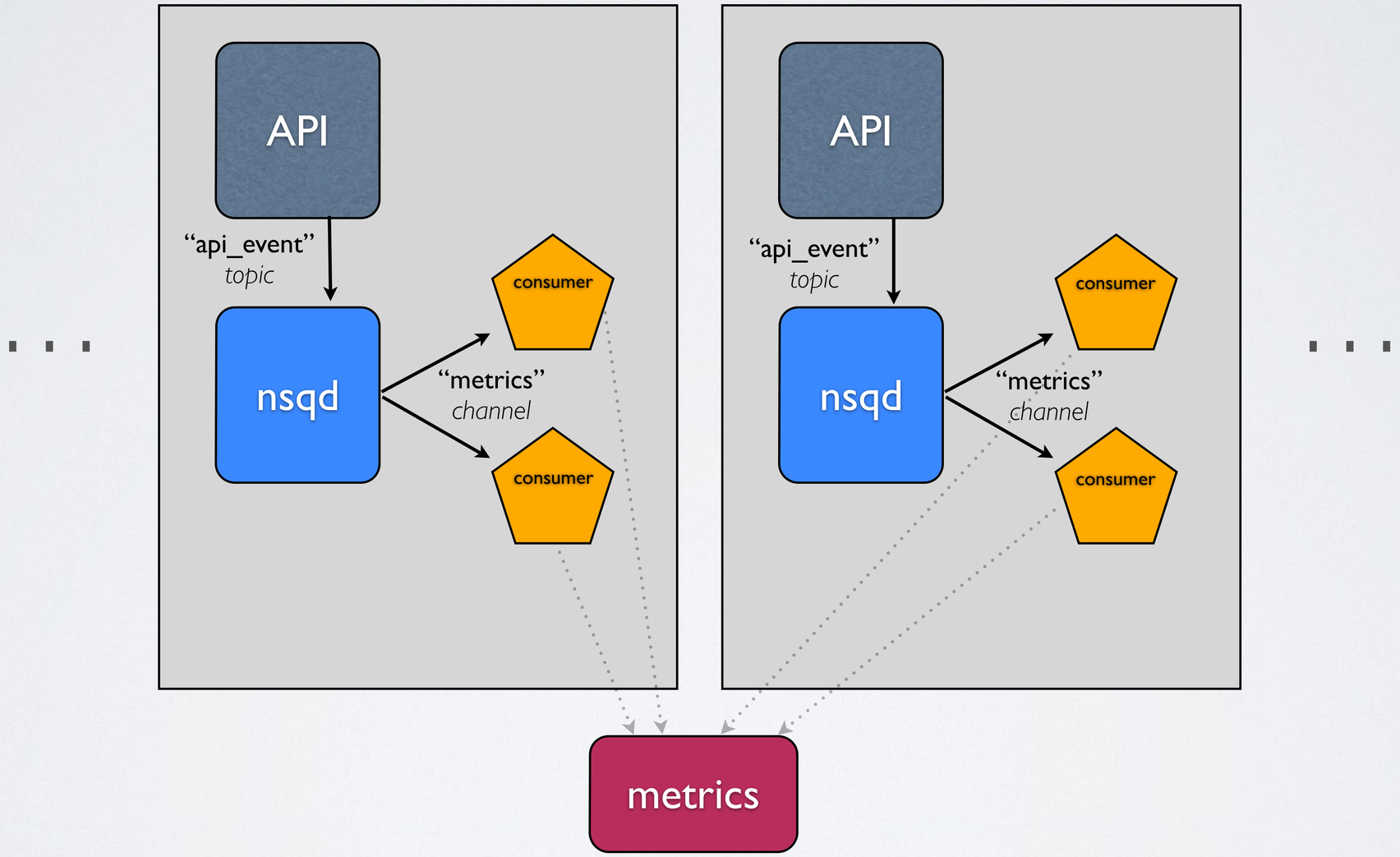
THERE AND BACK AGAIN

DE-COUPLE

- PUB locally to **nsqd** via HTTP
- perform work **async**
- co-locate everything (**silo**)



SCALE HORIZONTALLY



A MESSAGE QUEUE

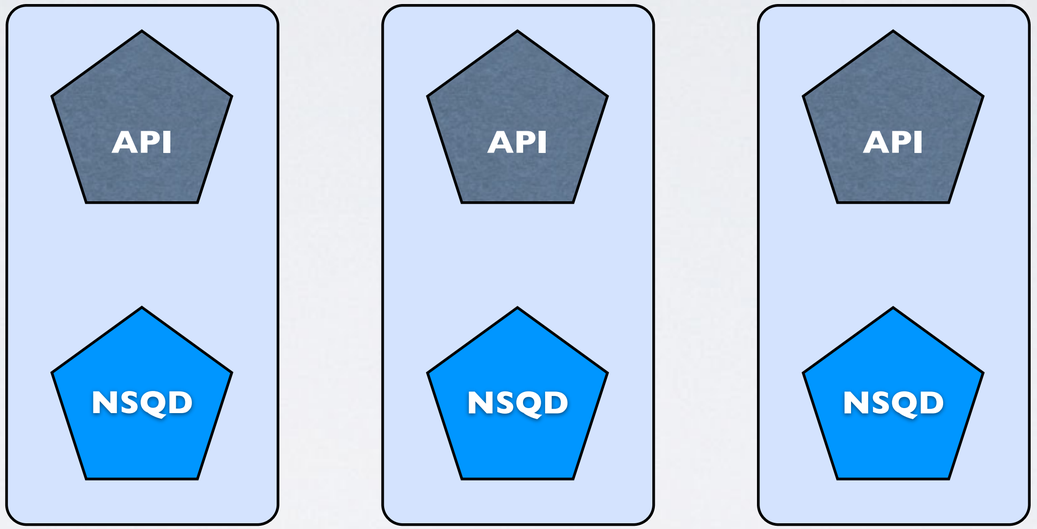
IS **BORING**

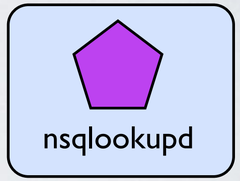
(IN ISOLATION)

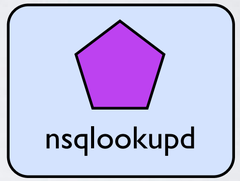
NSQLOOKUPD

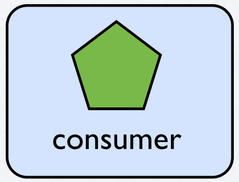
TYPICAL NSQ CLUSTER

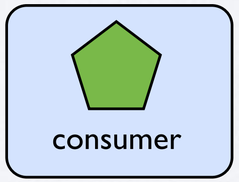
- enable *distributed* and *decentralized* topologies
- no centralized broker
- **nsqlookupd** instances are *independent* (**no coordinatation**)





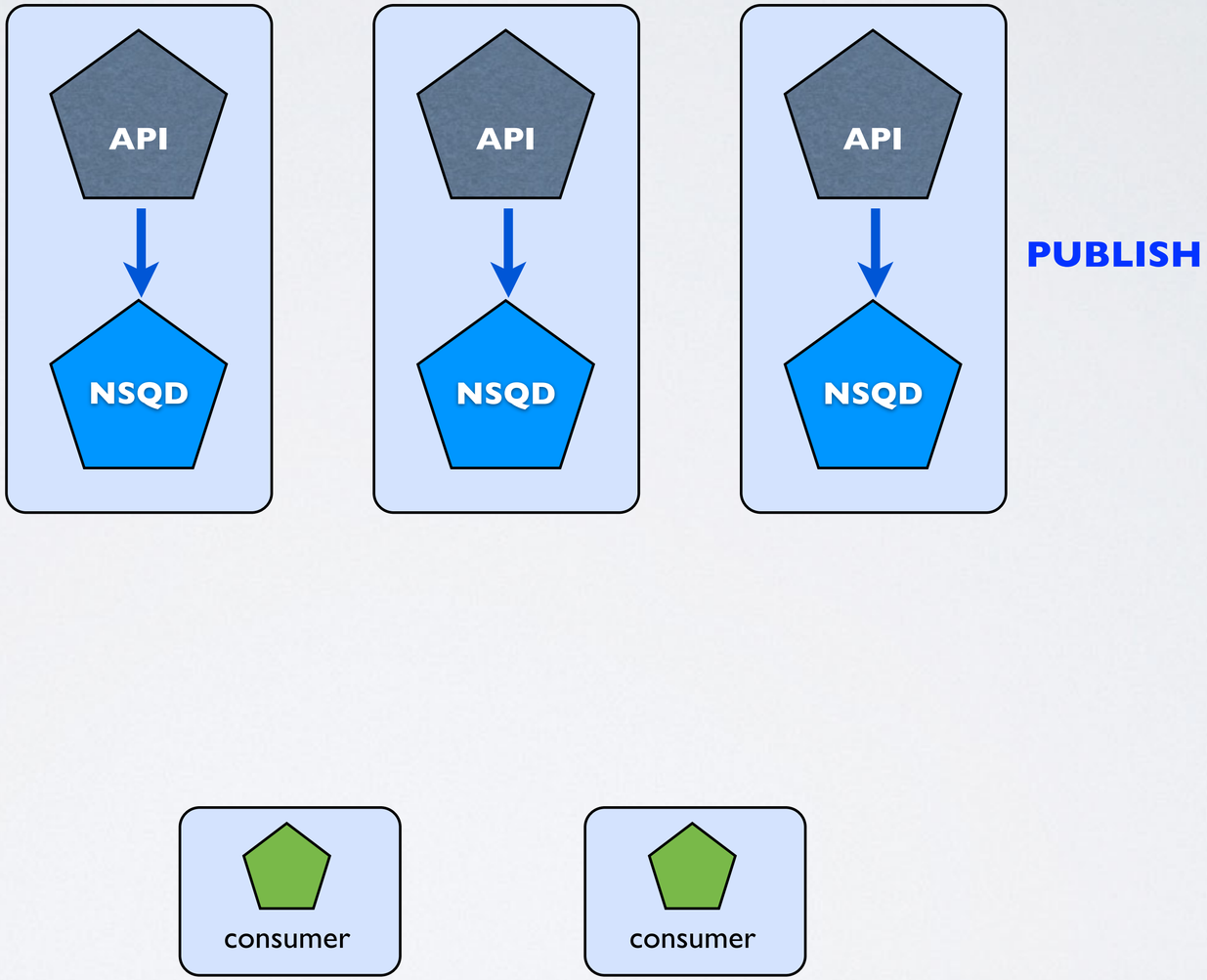






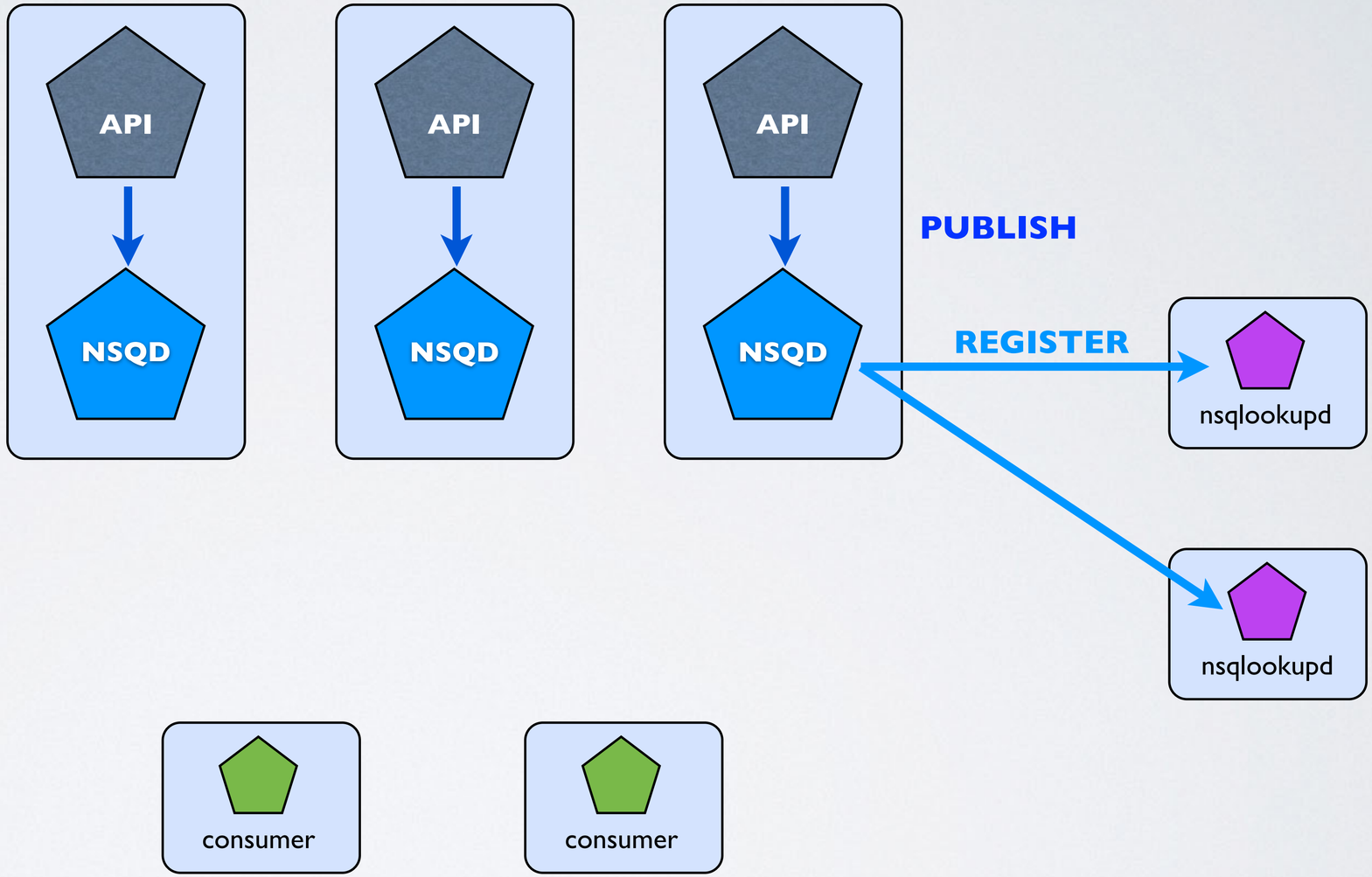
TYPICAL NSQ CLUSTER

- enable *distributed* and *decentralized* topologies
- no centralized broker
- **nsqlookupd** instances are *independent* (**no coordinatation**)



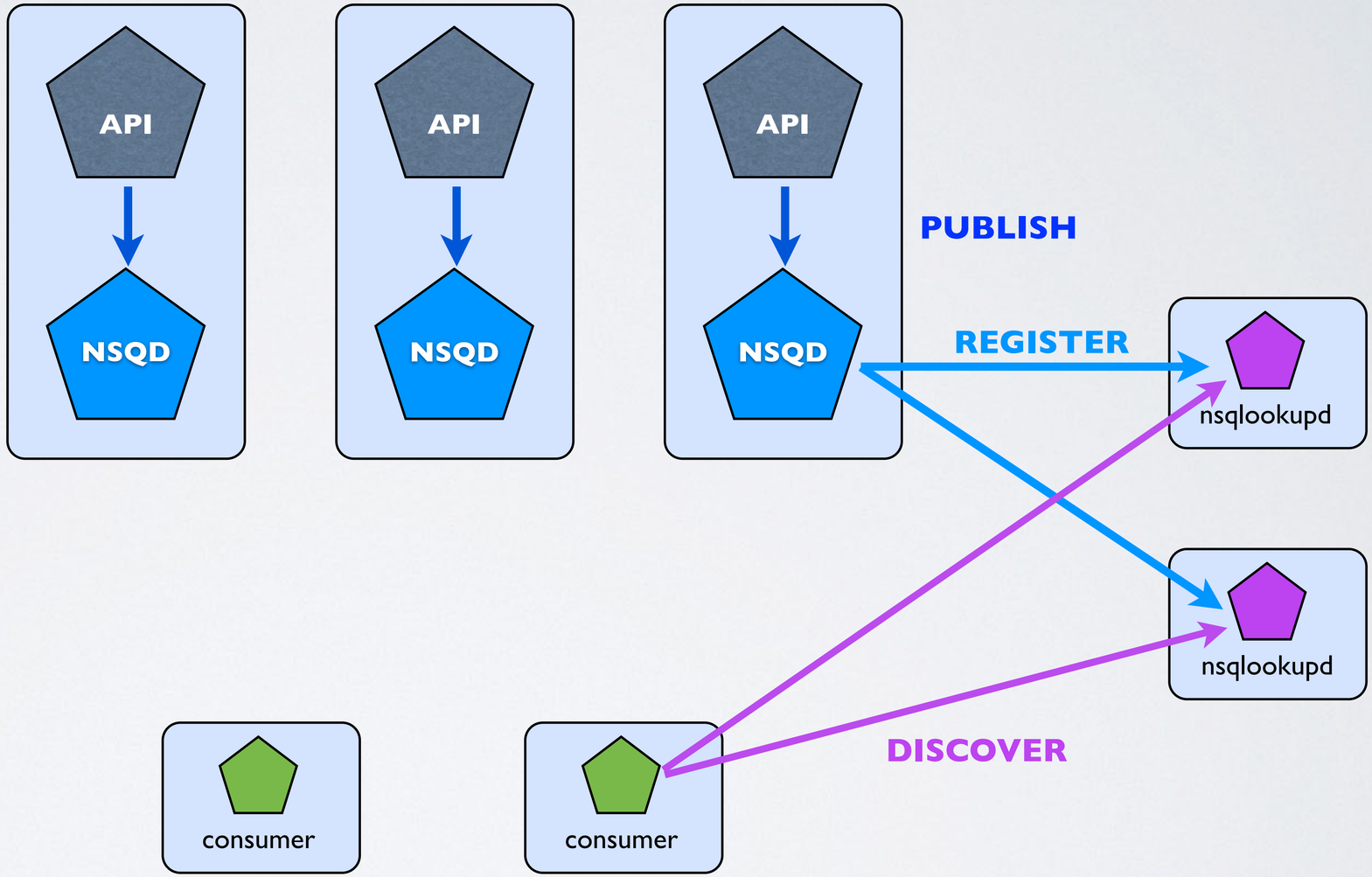
TYPICAL NSQ CLUSTER

- enable *distributed* and *decentralized* topologies
- no centralized broker
- **nsqlookupd** instances are *independent* (**no coordinatation**)



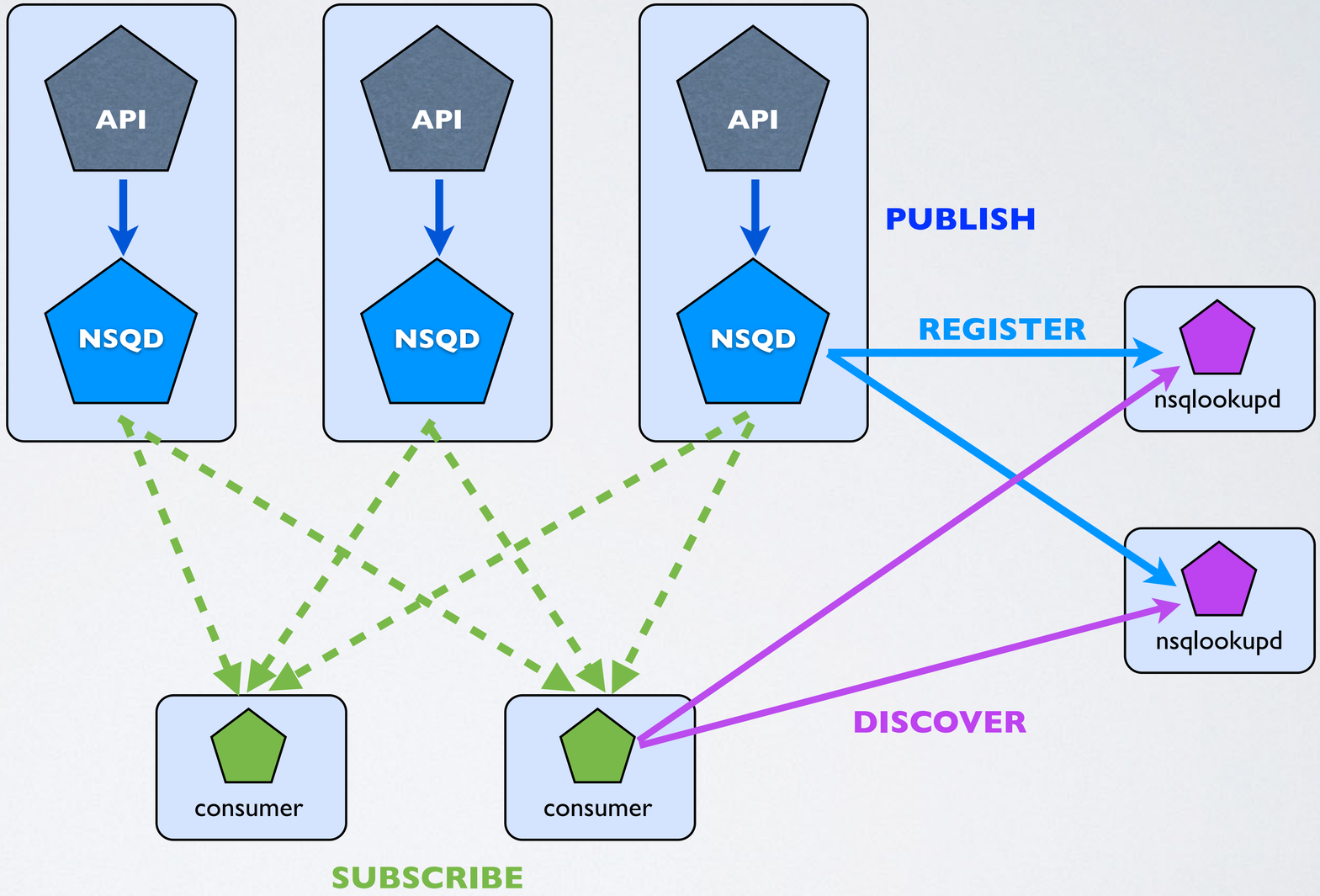
TYPICAL NSQ CLUSTER

- enable *distributed* and *decentralized* topologies
- no centralized broker
- **nsqlookupd** instances are *independent* (**no coordinatation**)



TYPICAL NSQ CLUSTER

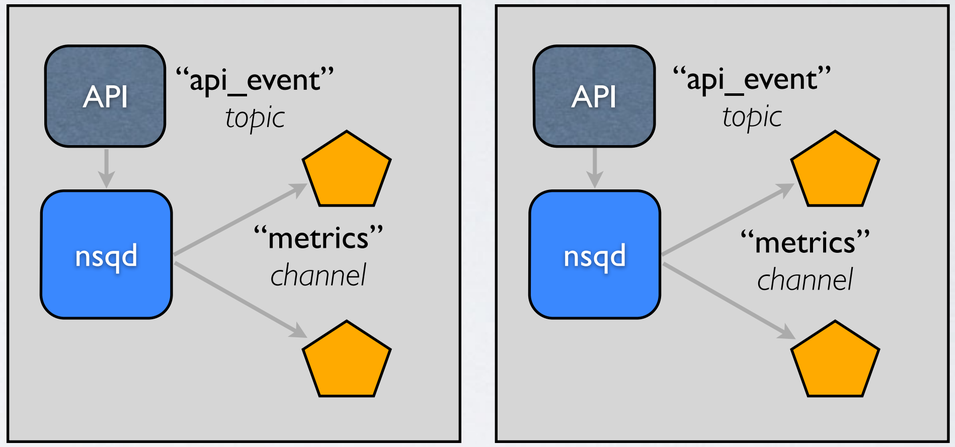
- enable *distributed* and *decentralized* topologies
- no centralized broker
- **nsqlookupd** instances are *independent* (**no coordinatation**)



… AND AGAIN

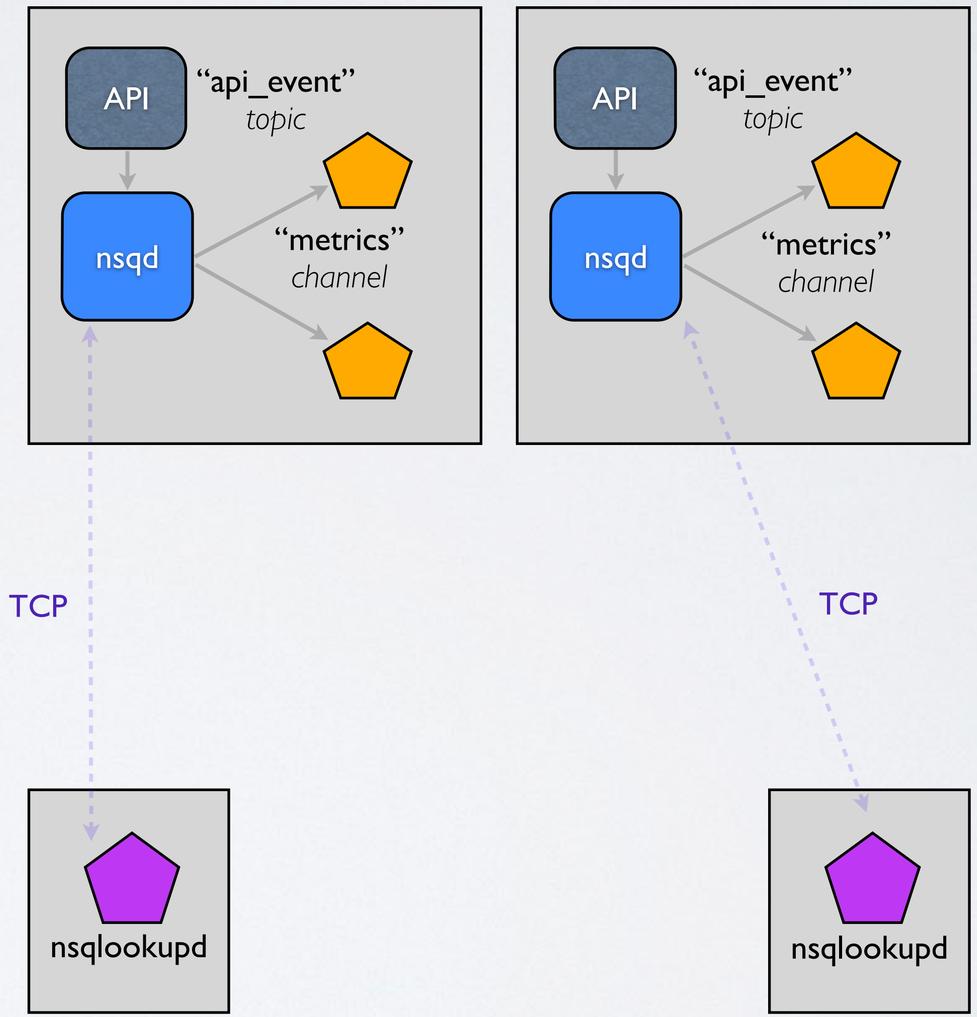
DISCOVERABILITY

- introduce **nsqlookupd**
- producers and consumers come and go
- other services can discover and subscribe to this topic



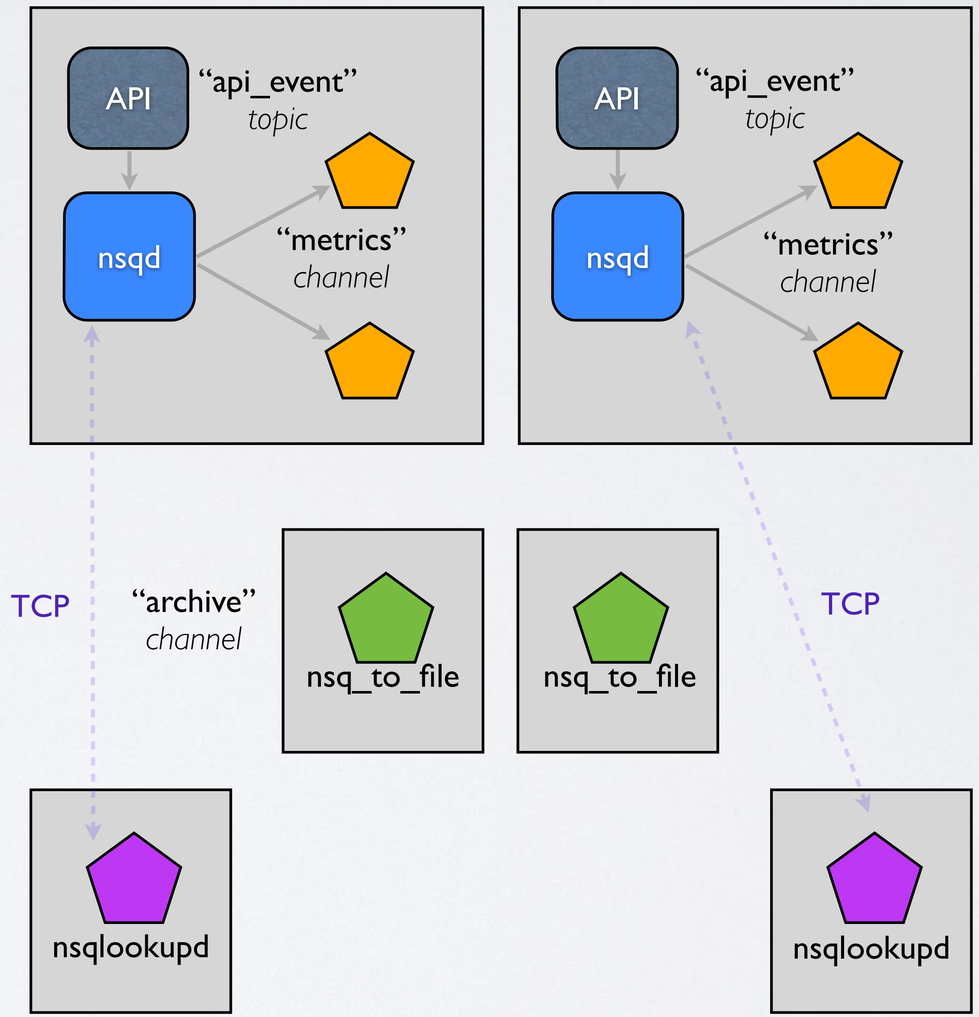
DISCOVERABILITY

- introduce **nsqlookupd**
- producers and consumers come and go
- other services can discover and subscribe to this topic



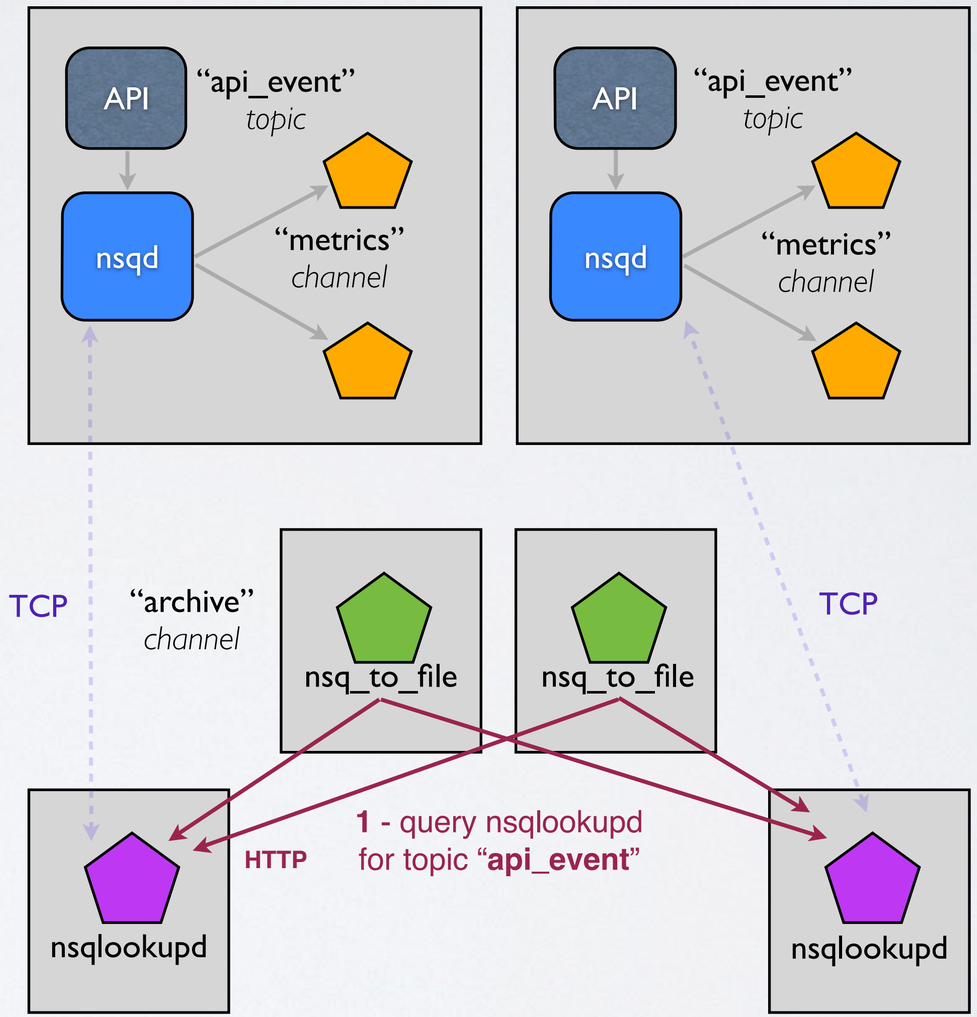
DISCOVERABILITY

- introduce **nsqlookupd**
- producers and consumers come and go
- other services can discover and subscribe to this topic



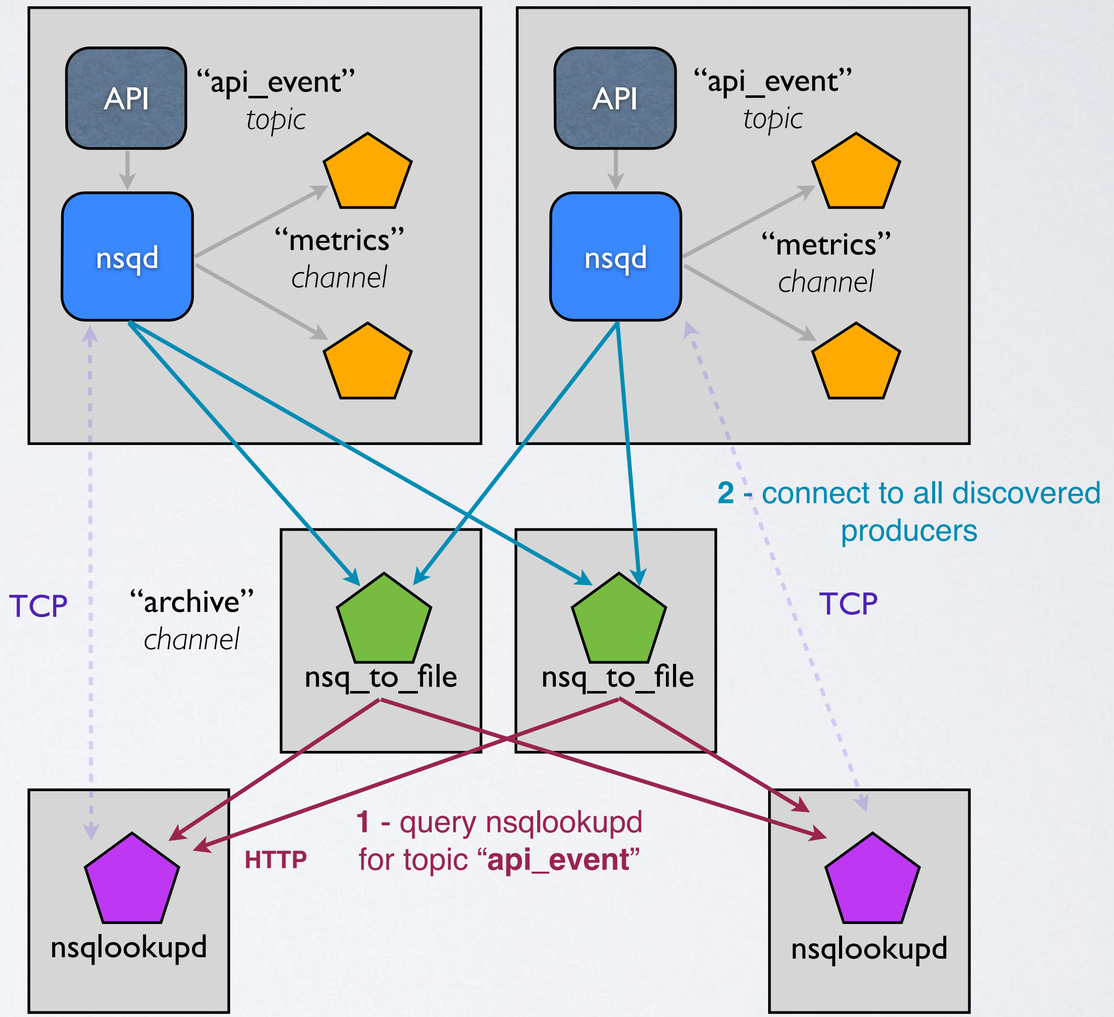
DISCOVERABILITY

- introduce **nsqlookupd**
- producers and consumers come and go
- other services can discover and subscribe to this topic

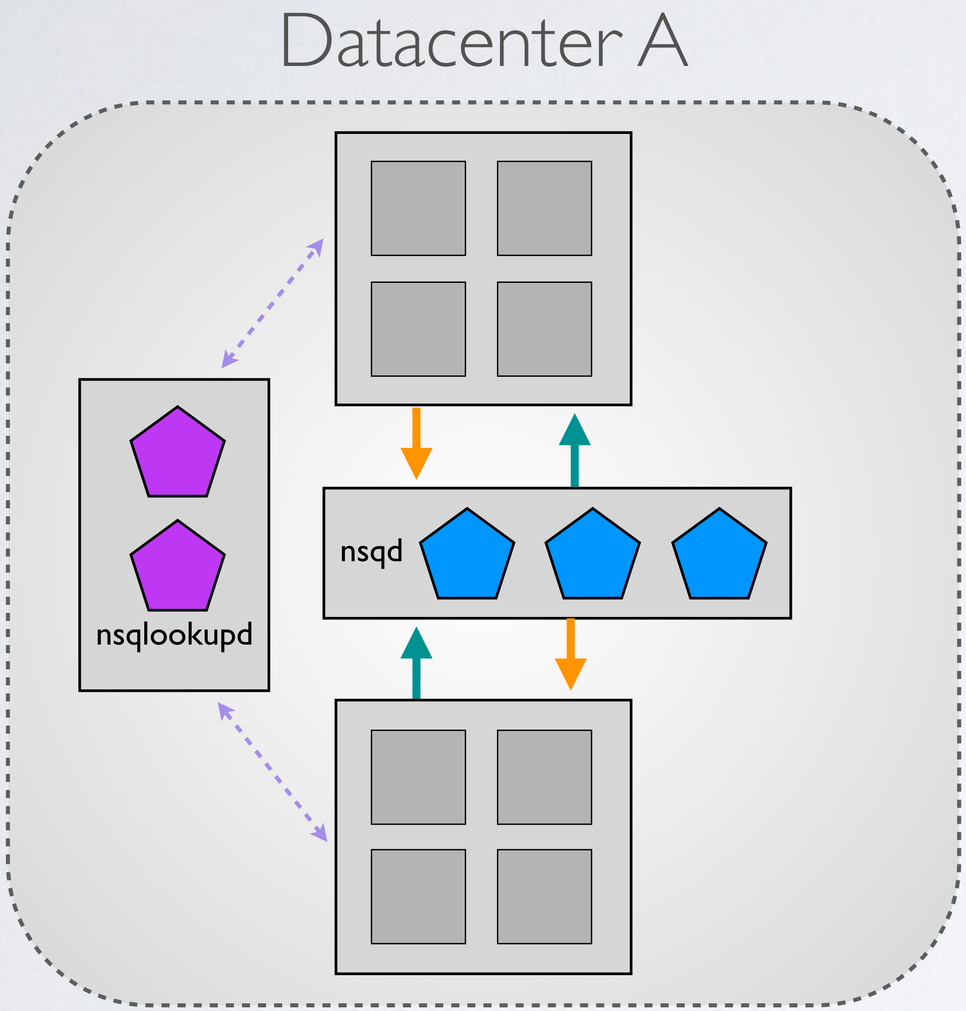


DISCOVERABILITY

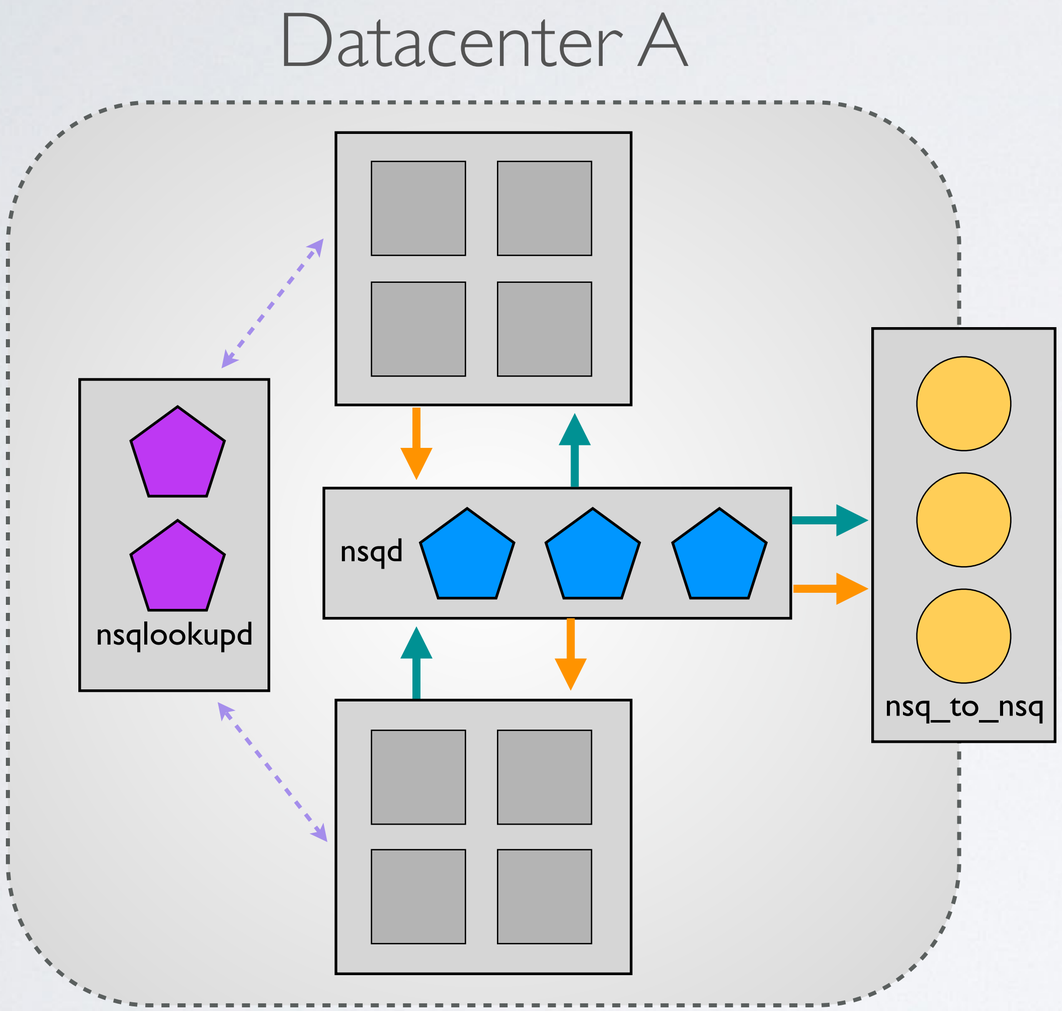
- introduce **nsqlookupd**
- producers and consumers come and go
- other services can discover and subscribe to this topic



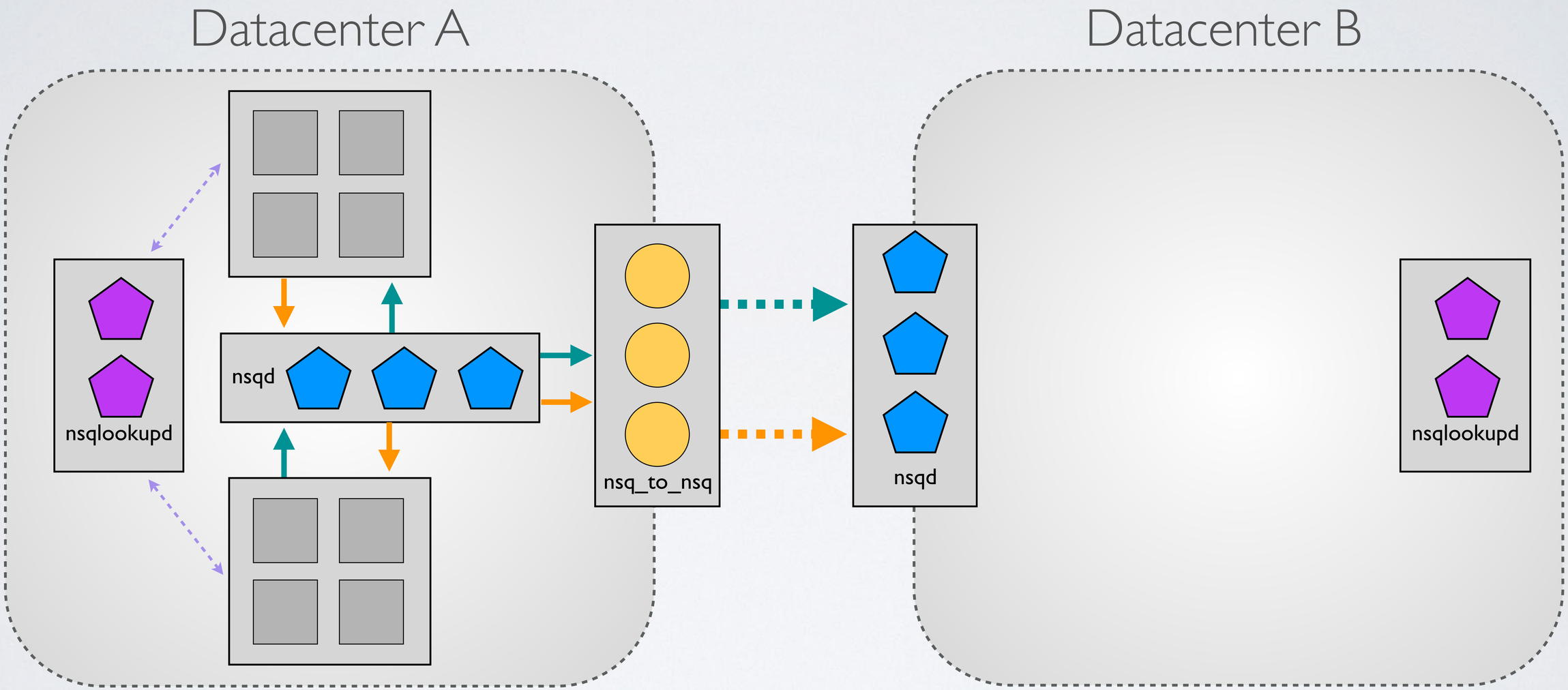
TO INFINITY AND BEYOND



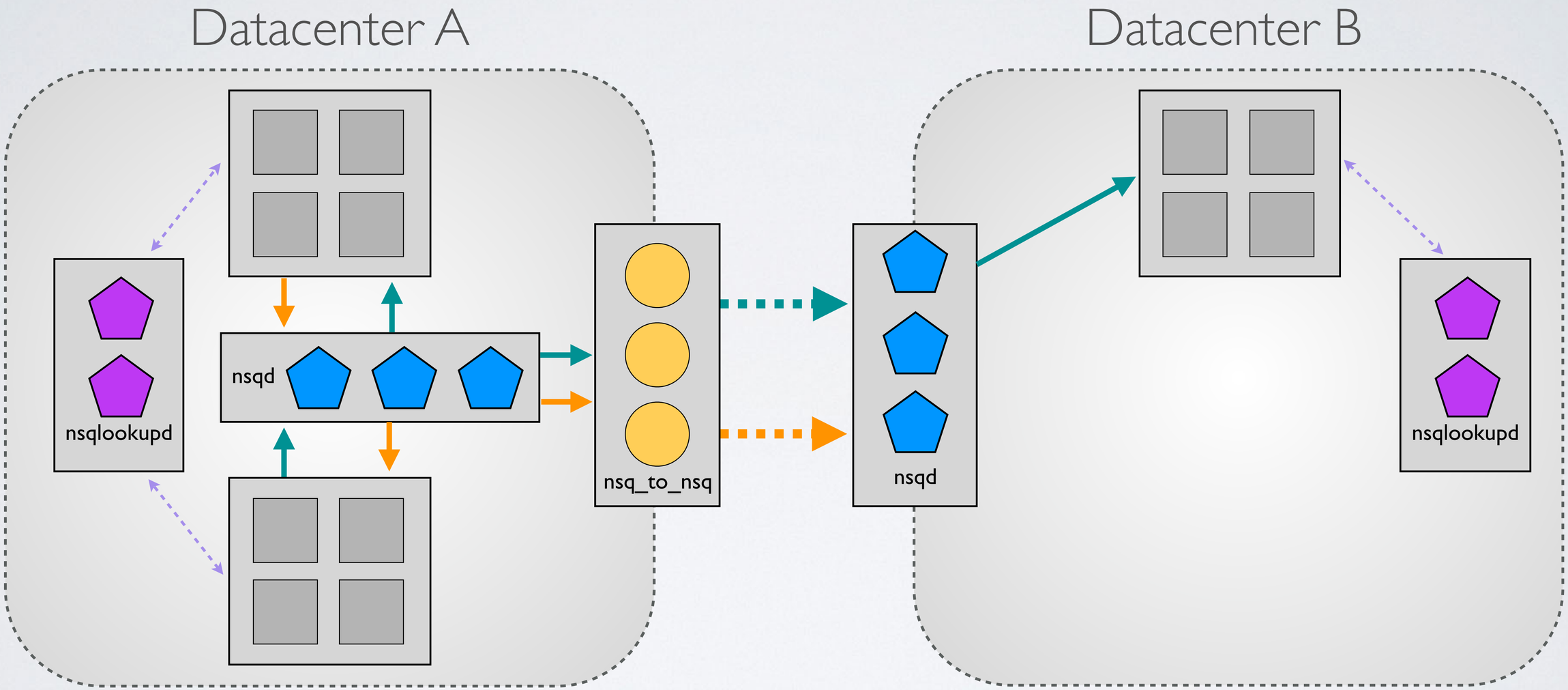
TO INFINITY AND BEYOND



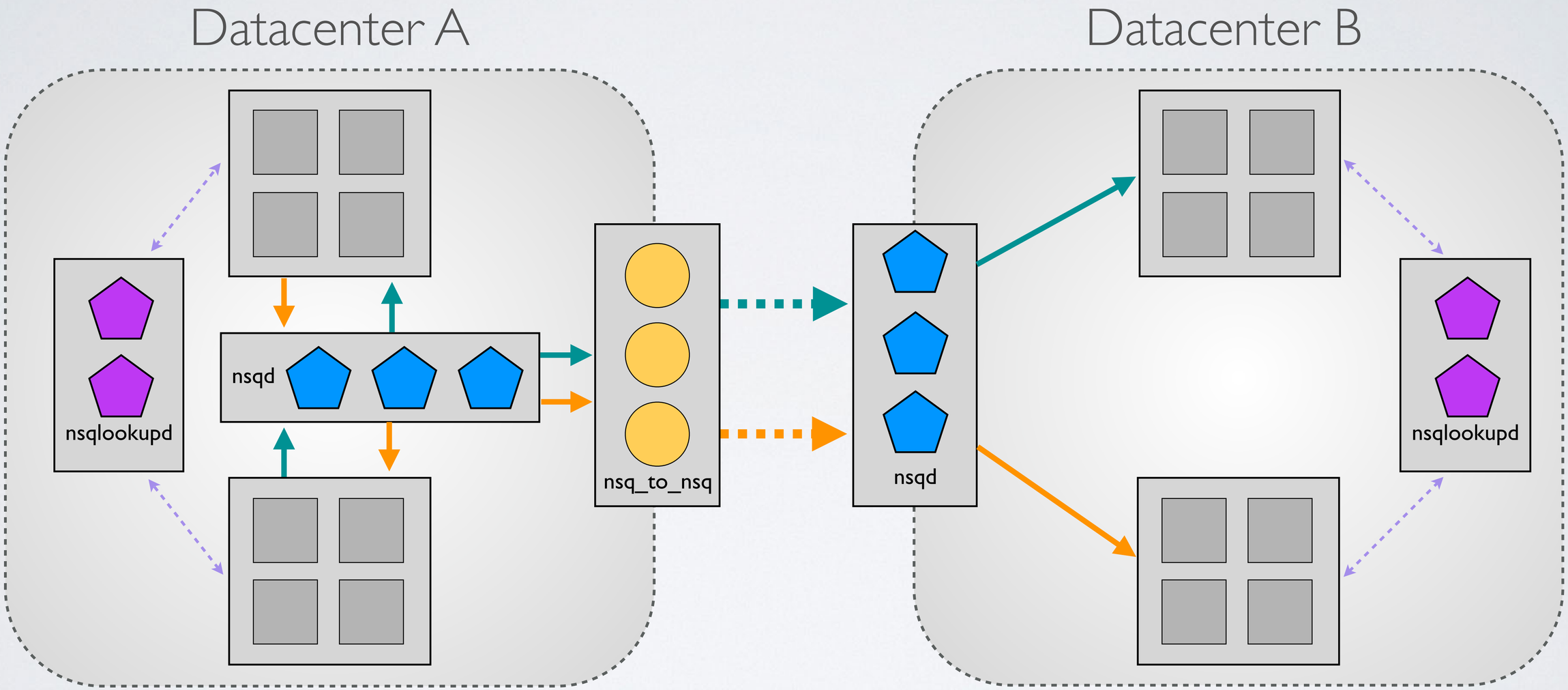
TO INFINITY AND BEYOND



TO INFINITY AND BEYOND



TO INFINITY AND BEYOND



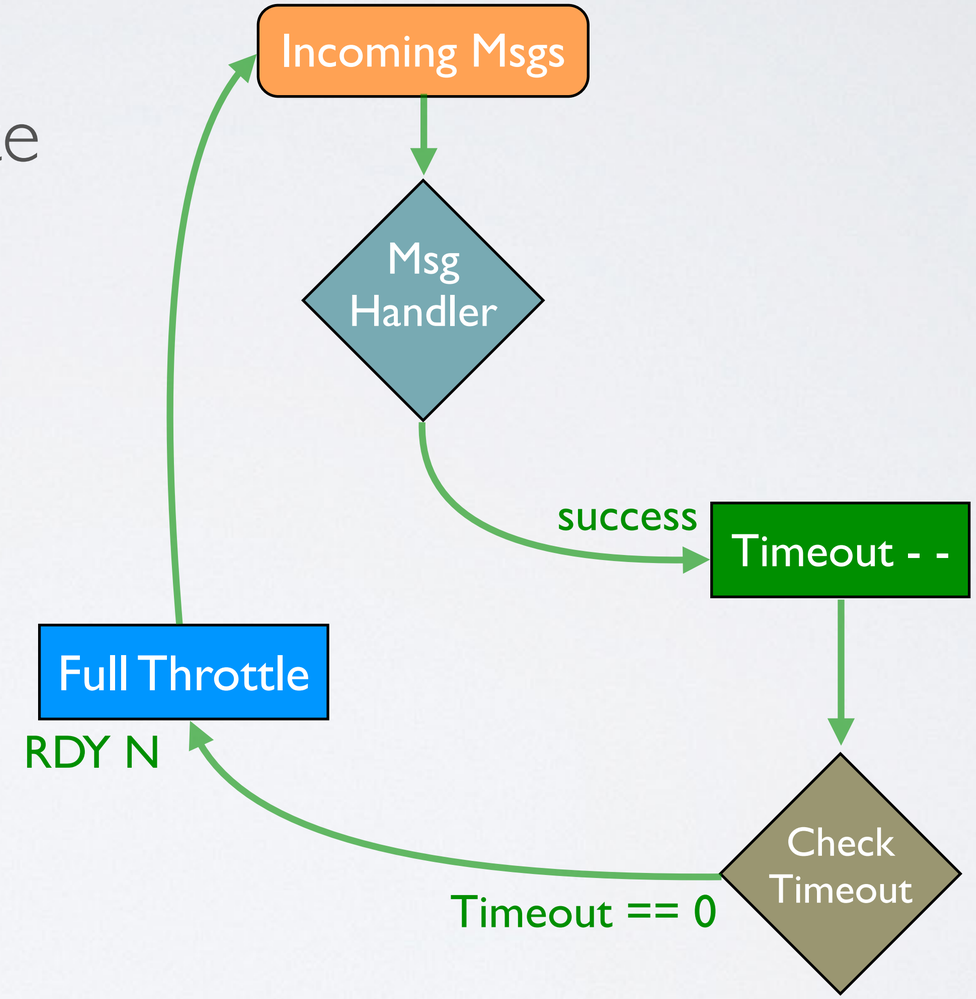
OPS

GUARANTEES

- messages are delivered *at least* once
- messages are *not* durable (by default)
- messages received are *un-ordered*
- consumers *eventually* find all topic producers

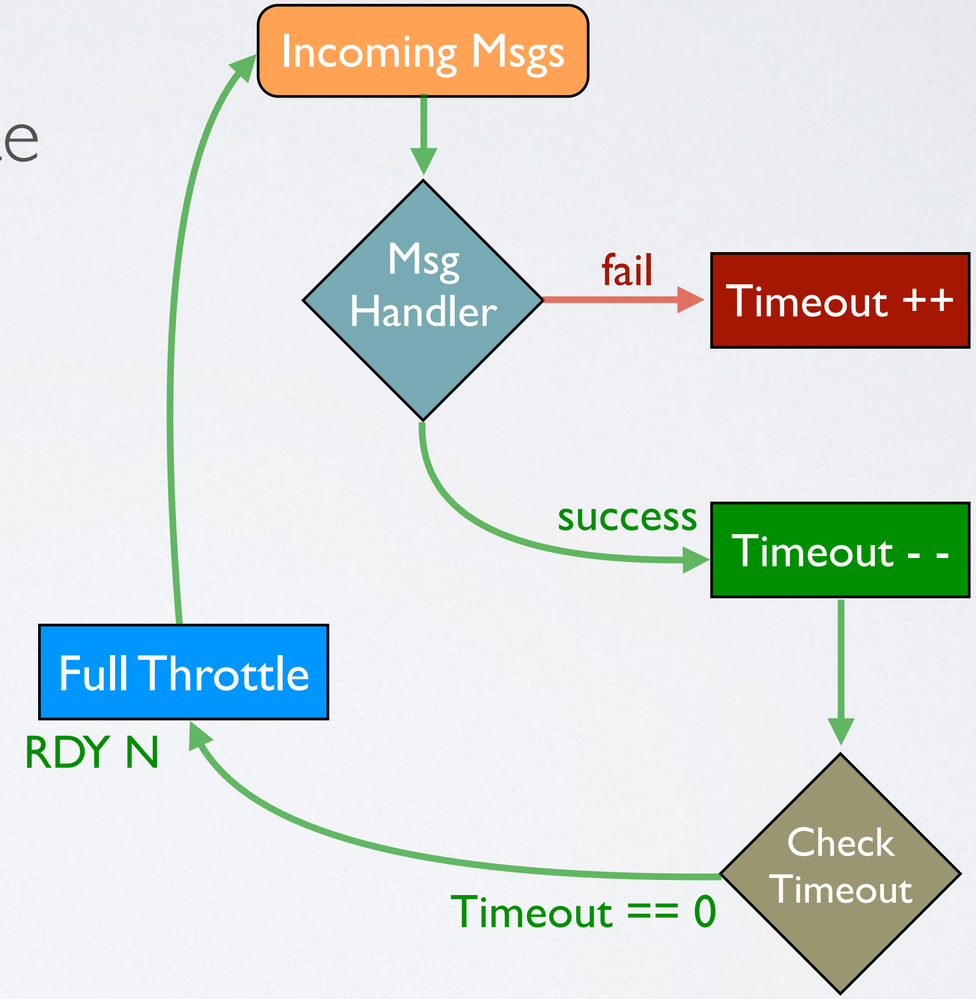
BACK PRESSSSSSSSSSSURE

- clients control flow - **RDY** state
  - concurrency knob
- processing fails?
  - slow down rate
- bad message?
  - limit attempts
  - delay re-processing



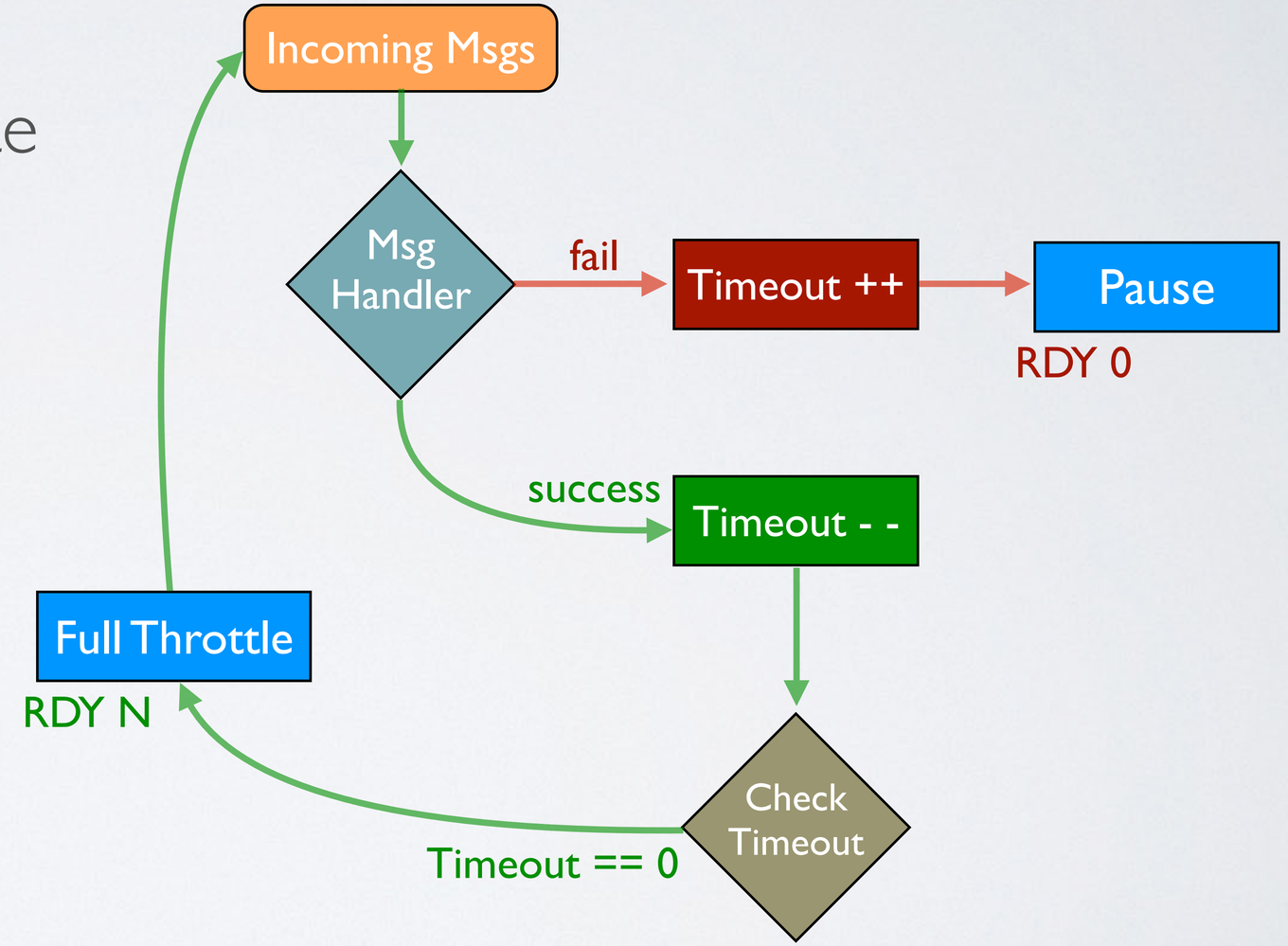
BACK PRESSSSSSSSSSSURE

- clients control flow - **RDY** state
  - concurrency knob
- processing fails?
  - slow down rate
- bad message?
  - limit attempts
  - delay re-processing



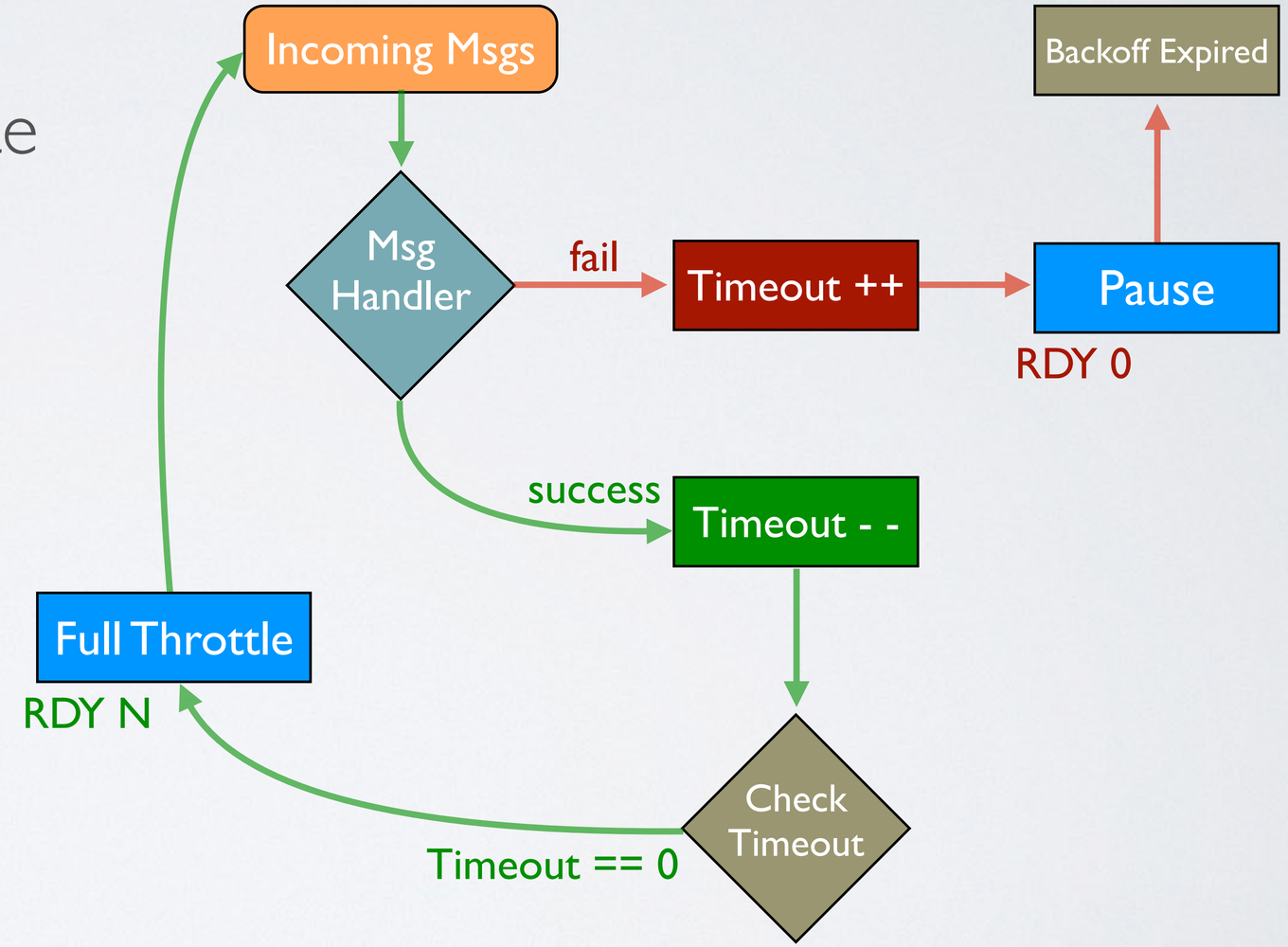
BACK PRESSSSSSSSSSSURE

- clients control flow - **RDY** state
  - concurrency knob
- processing fails?
  - slow down rate
- bad message?
  - limit attempts
  - delay re-processing



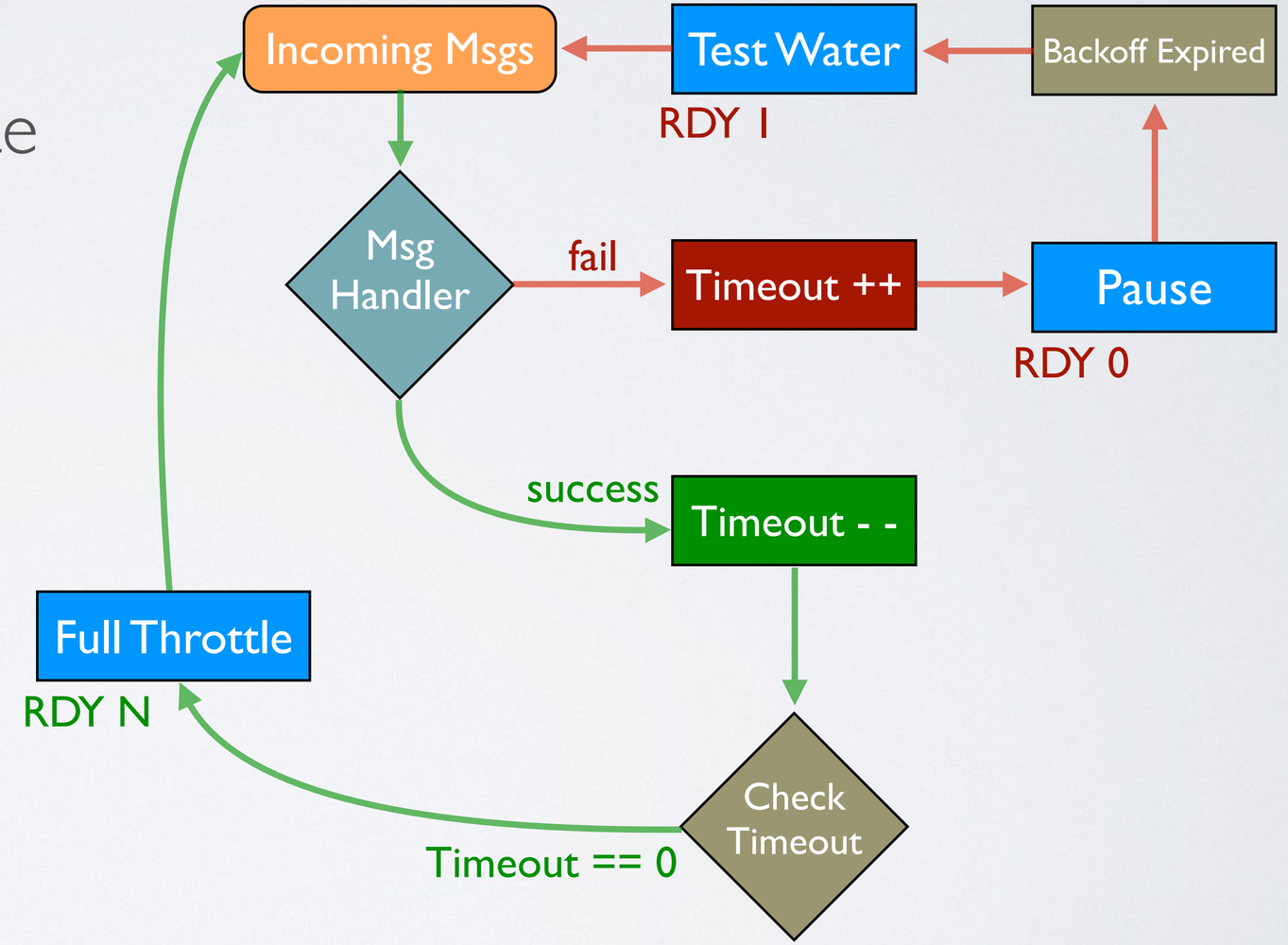
BACK PRESSSSSSSSSSSURE

- clients control flow - **RDY** state
  - concurrency knob
- processing fails?
  - slow down rate
- bad message?
  - limit attempts
  - delay re-processing



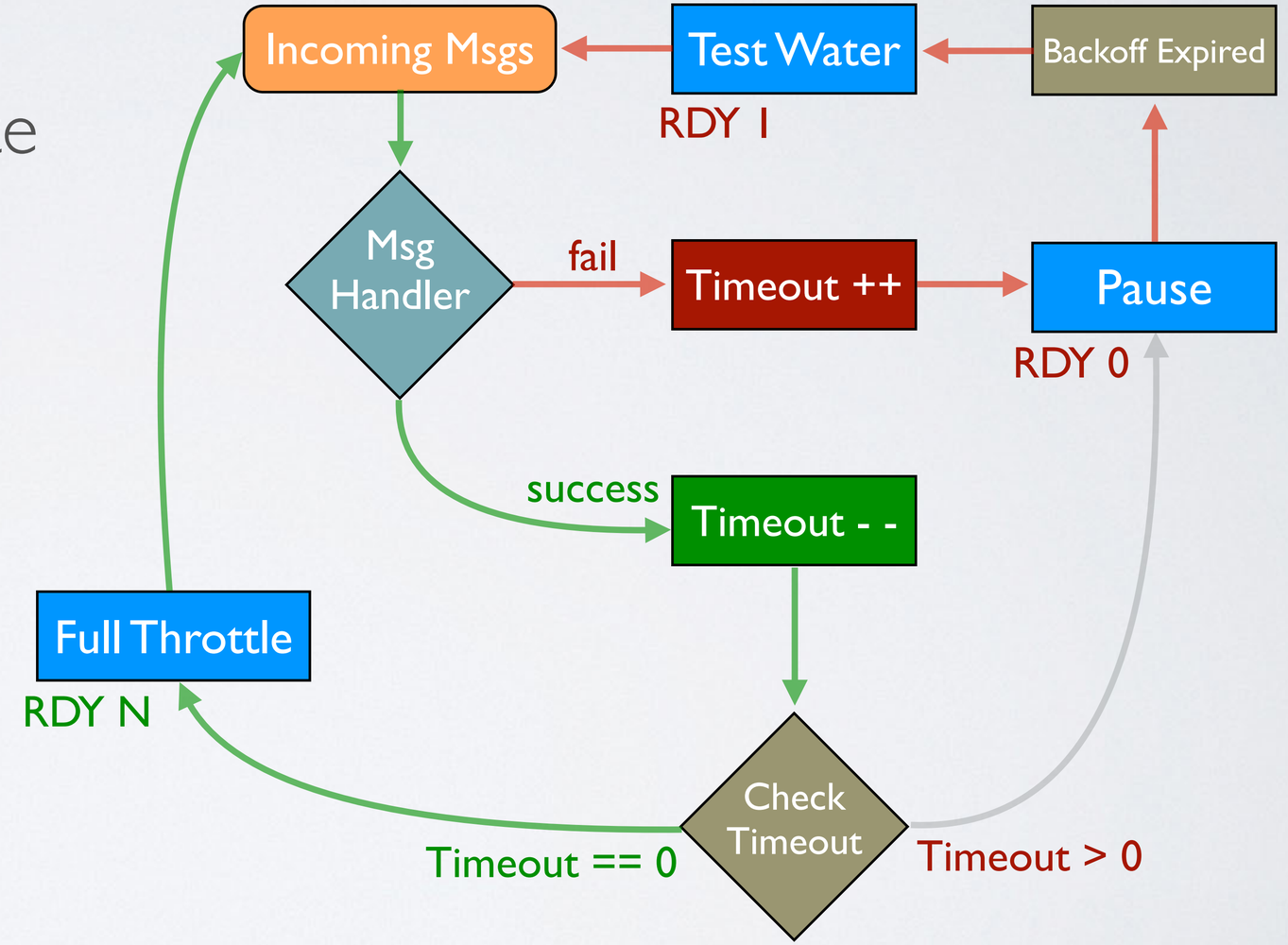
BACK PRESSSSSSSSSSSURE

- clients control flow - **RDY** state
  - concurrency knob
- processing fails?
  - slow down rate
- bad message?
  - limit attempts
  - delay re-processing

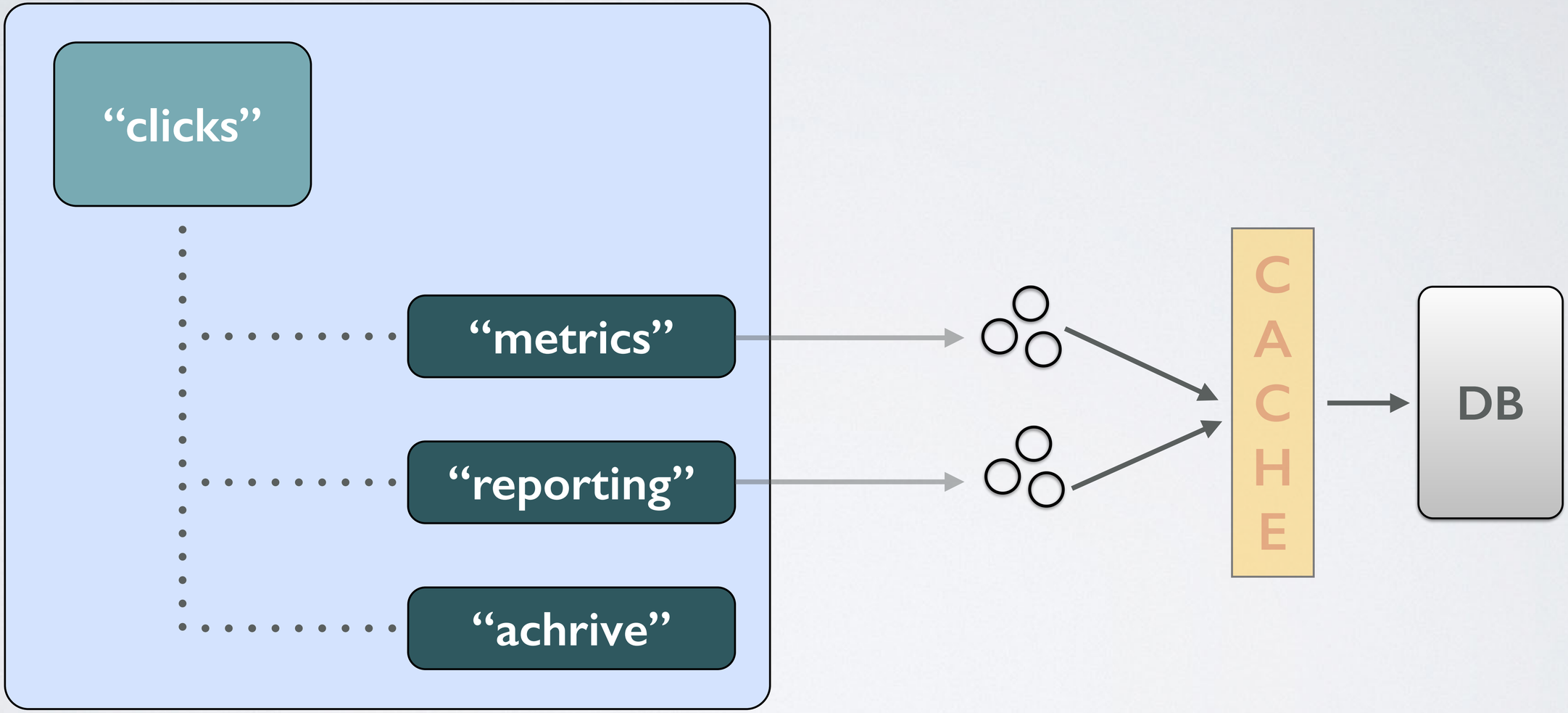


BACK PRESSSSSSSSSSSURE

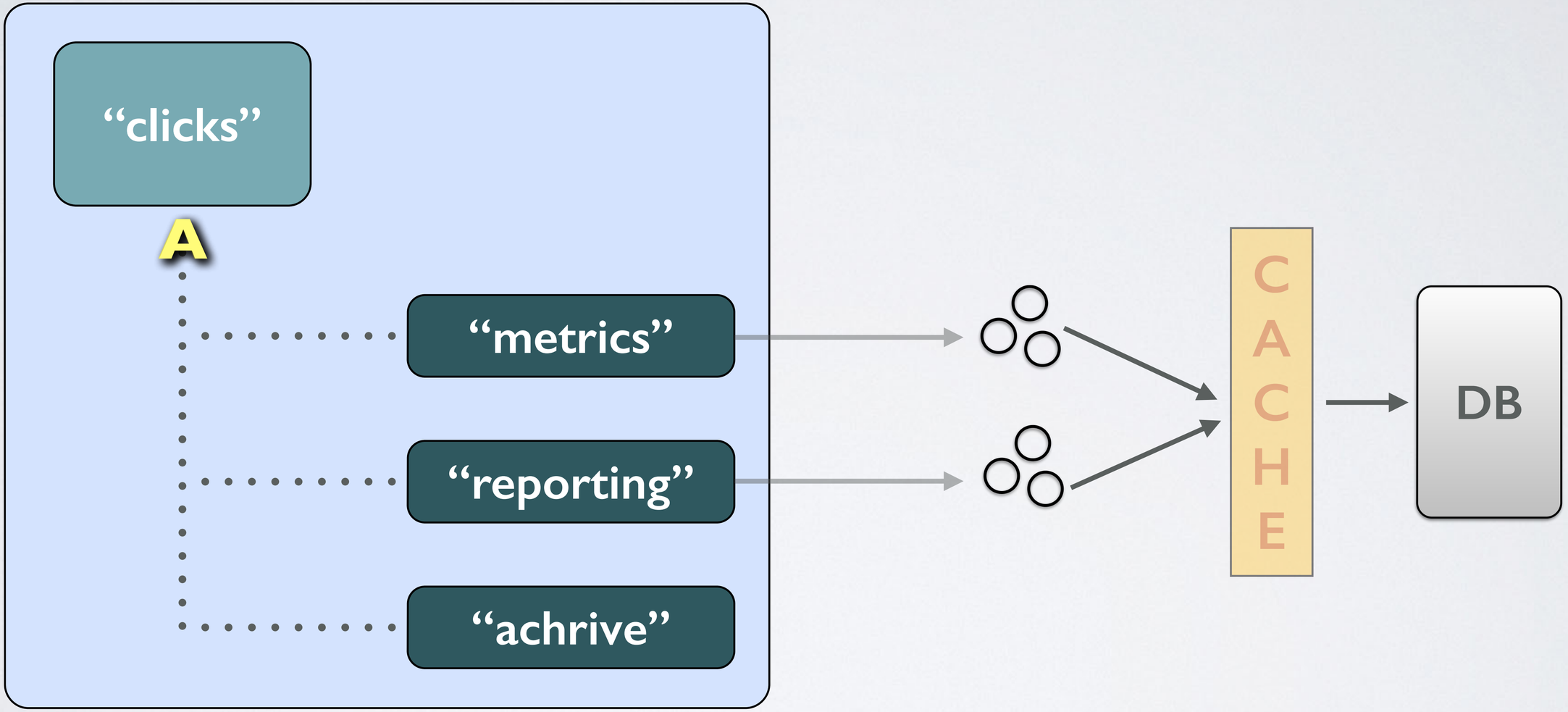
- clients control flow - **RDY** state
  - concurrency knob
- processing fails?
  - slow down rate
- bad message?
  - limit attempts
  - delay re-processing



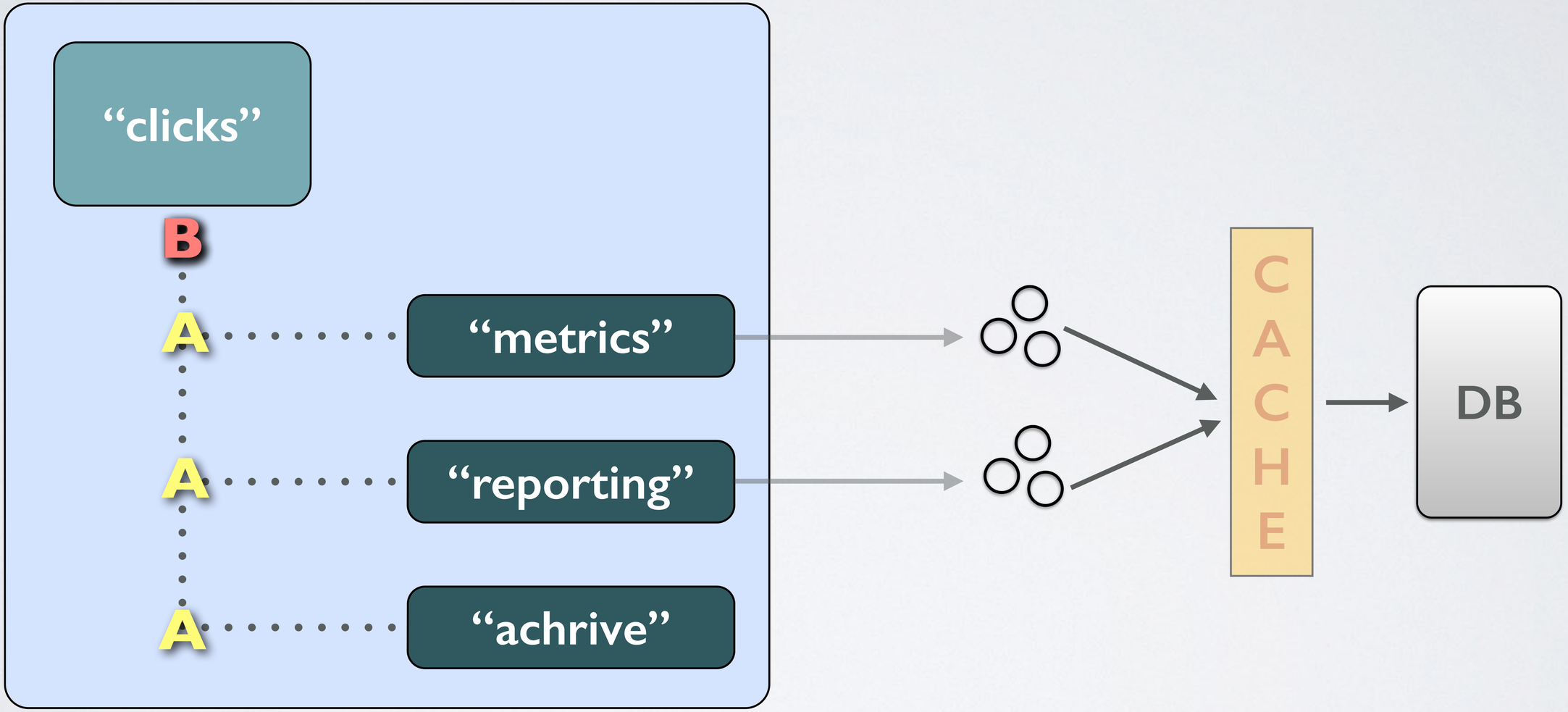
TOPIC & CHANNEL PAUSING



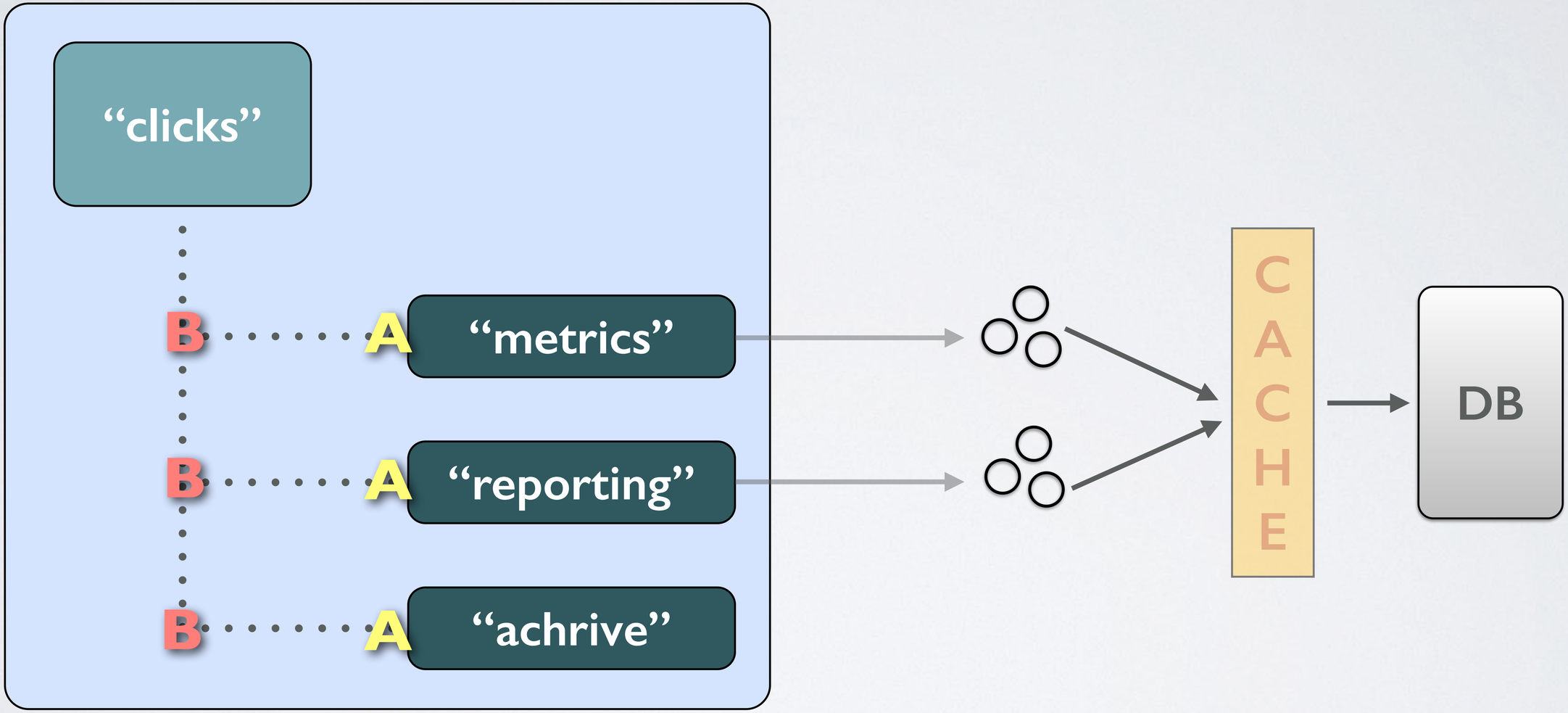
TOPIC & CHANNEL PAUSING



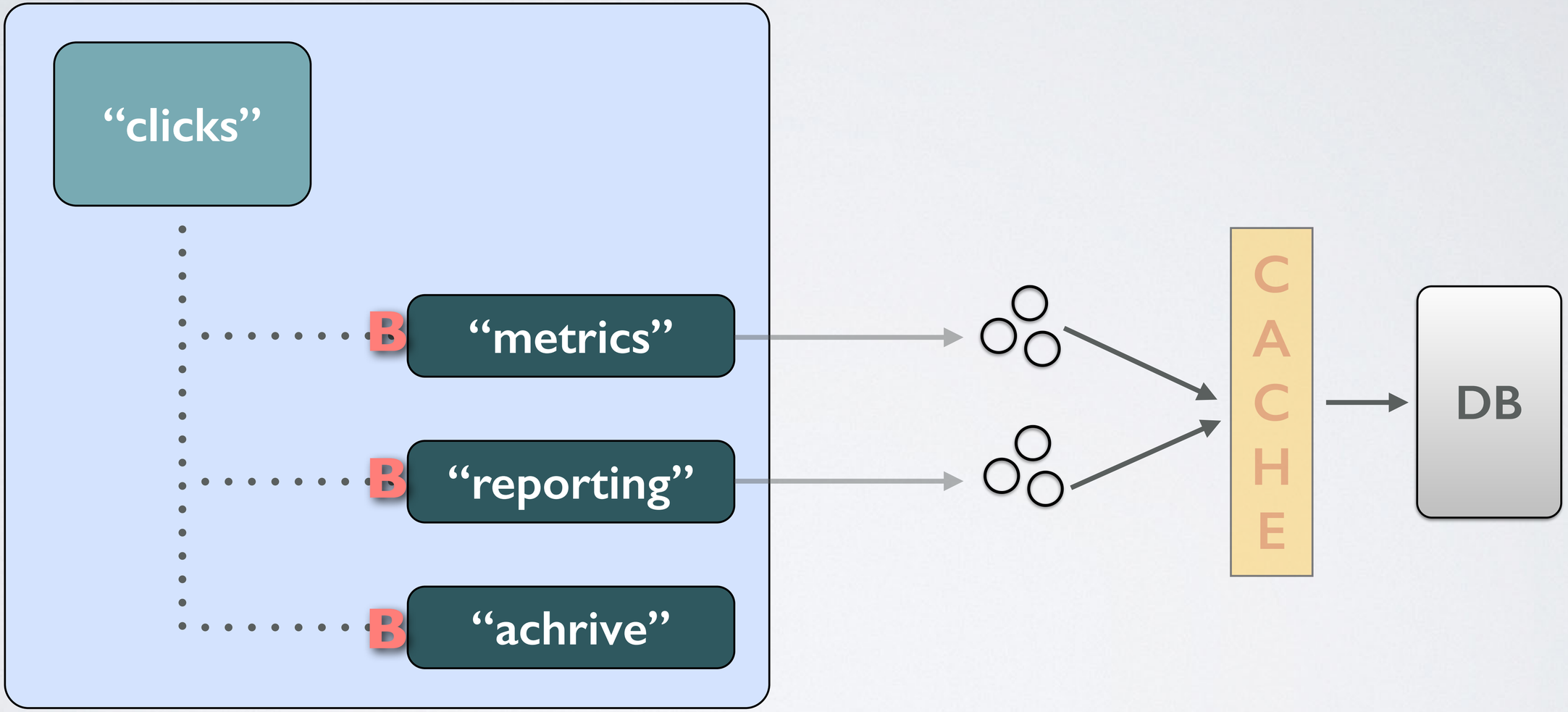
TOPIC & CHANNEL PAUSING



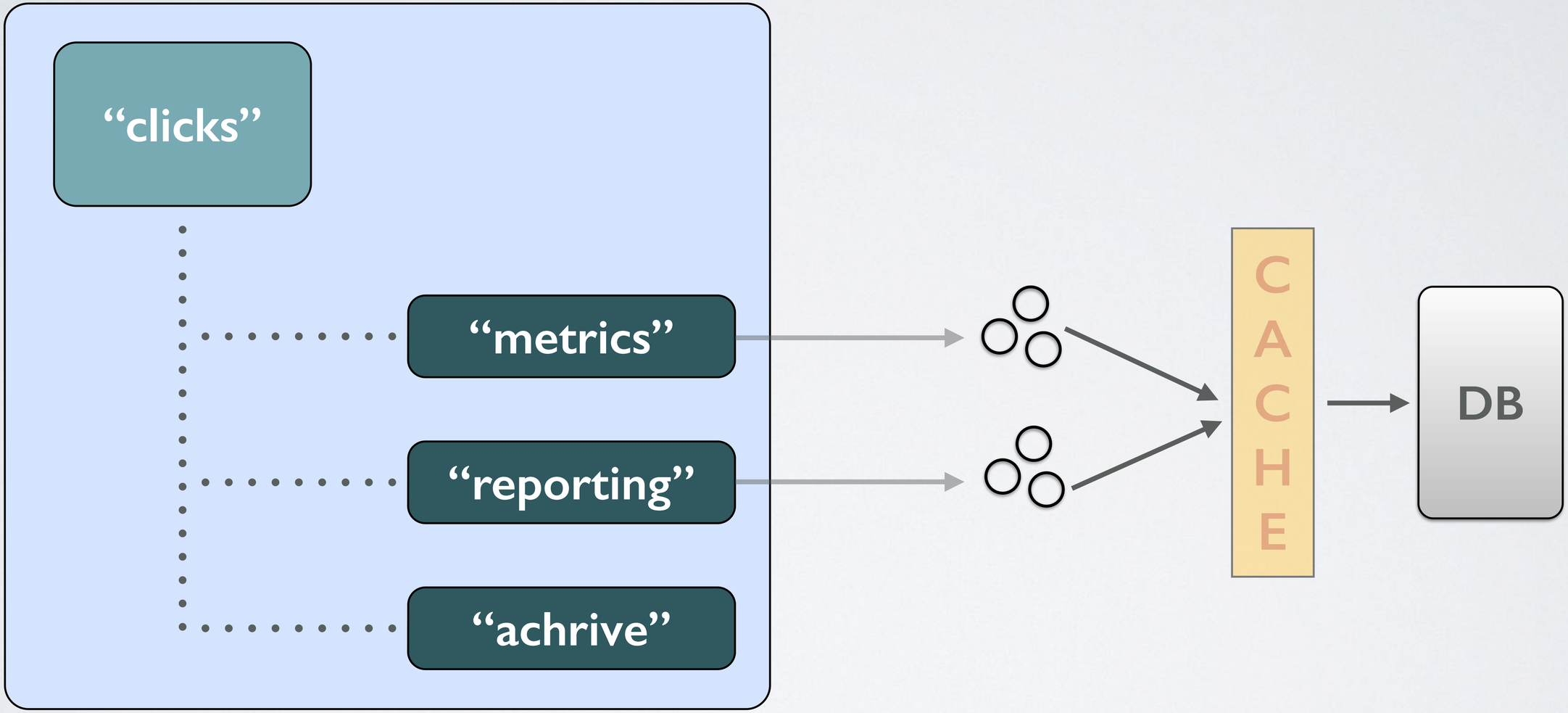
TOPIC & CHANNEL PAUSING



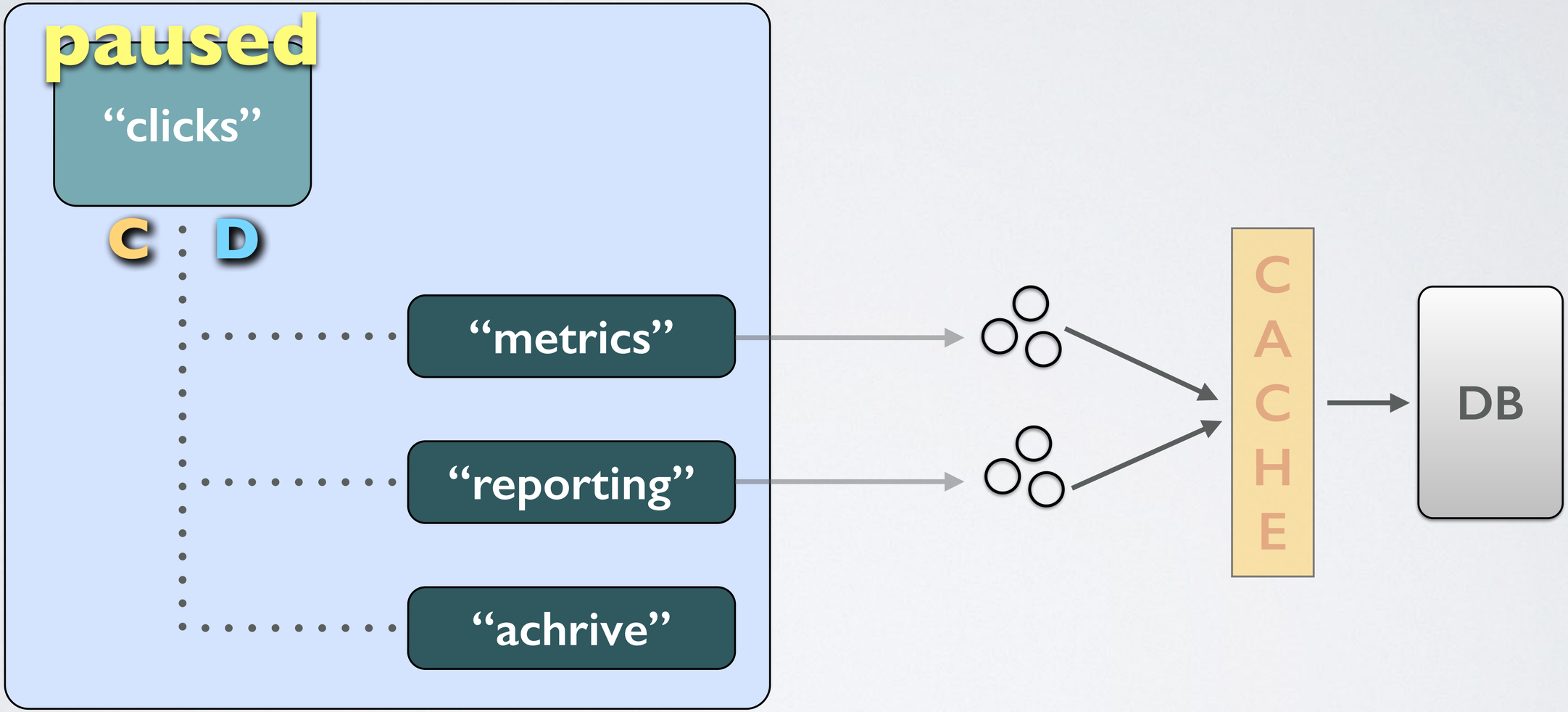
TOPIC & CHANNEL PAUSING



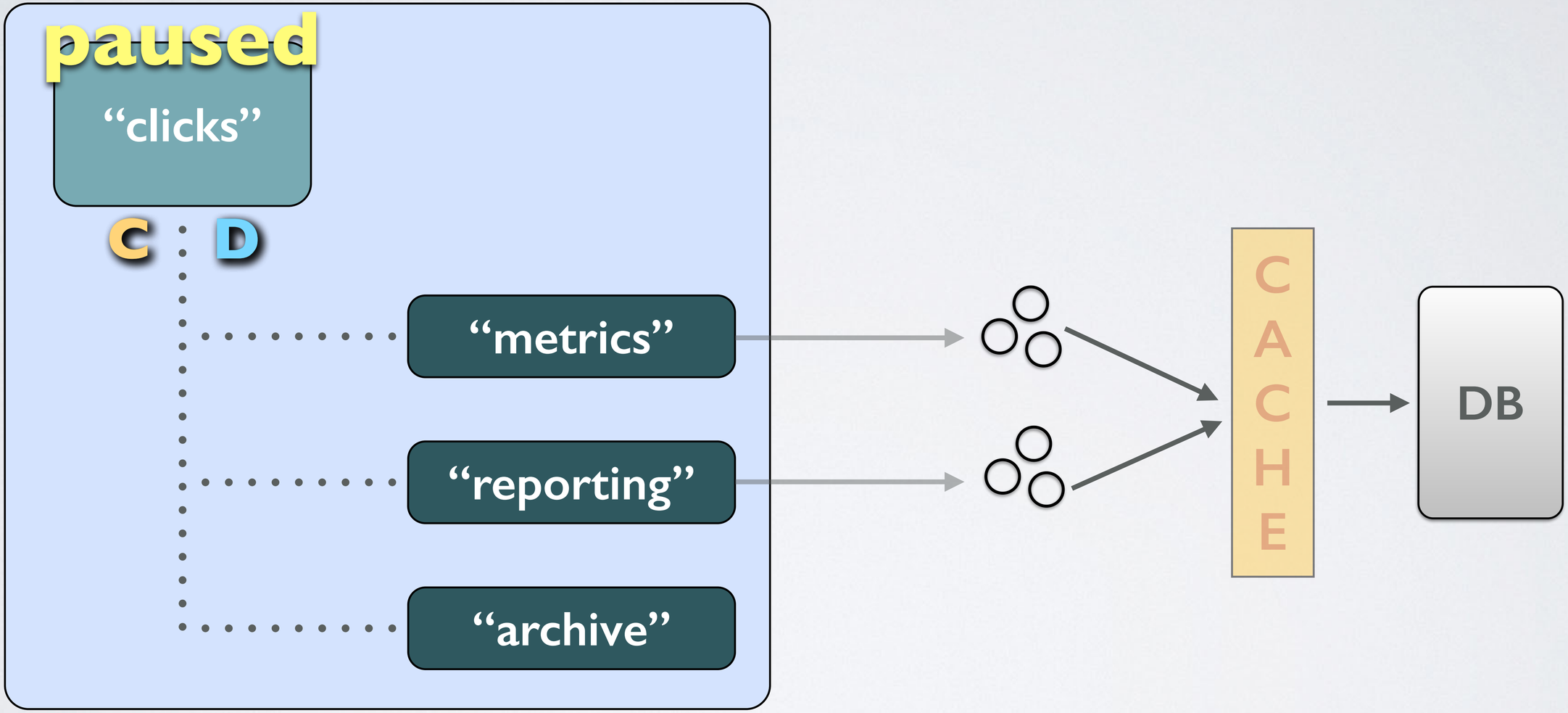
TOPIC & CHANNEL PAUSING



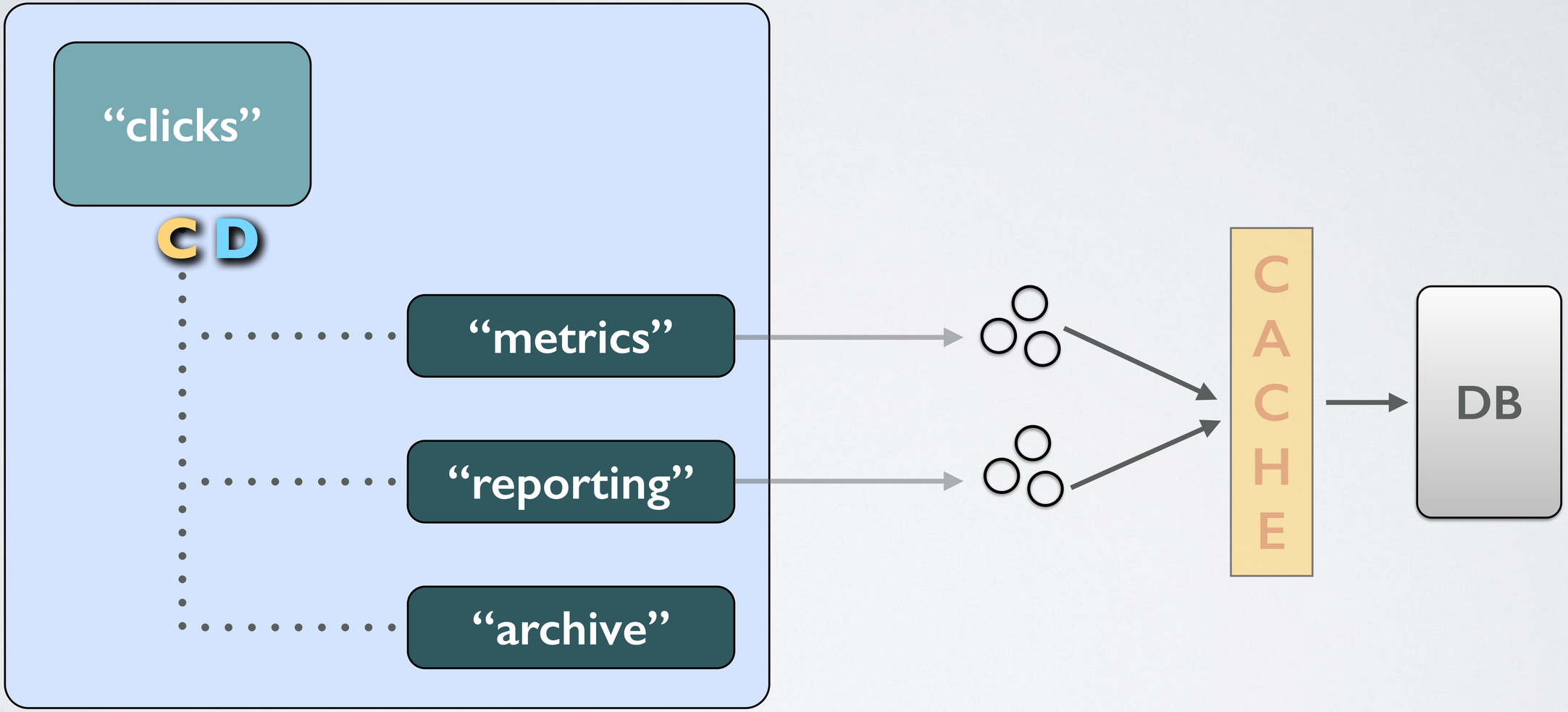
TOPIC & CHANNEL PAUSING



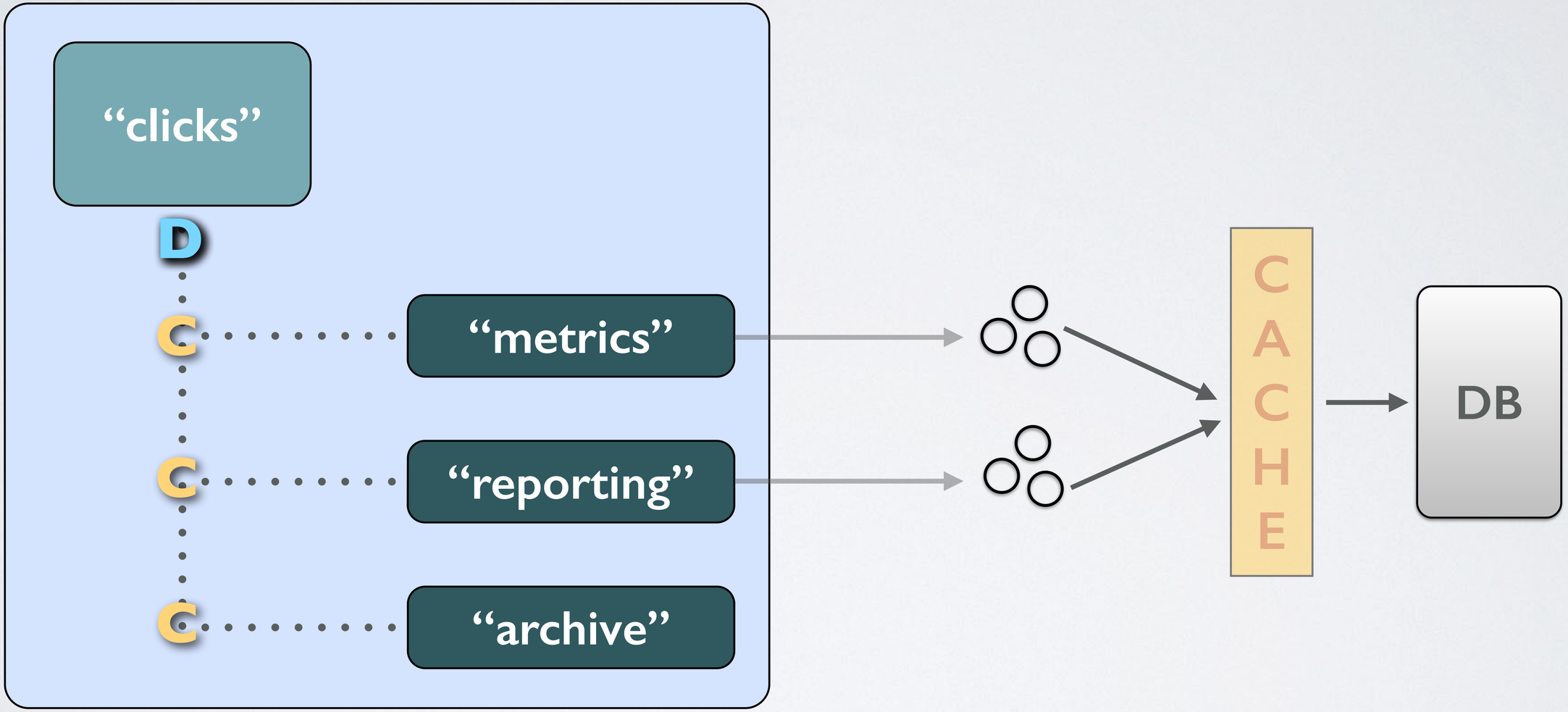
TOPIC & CHANNEL PAUSING



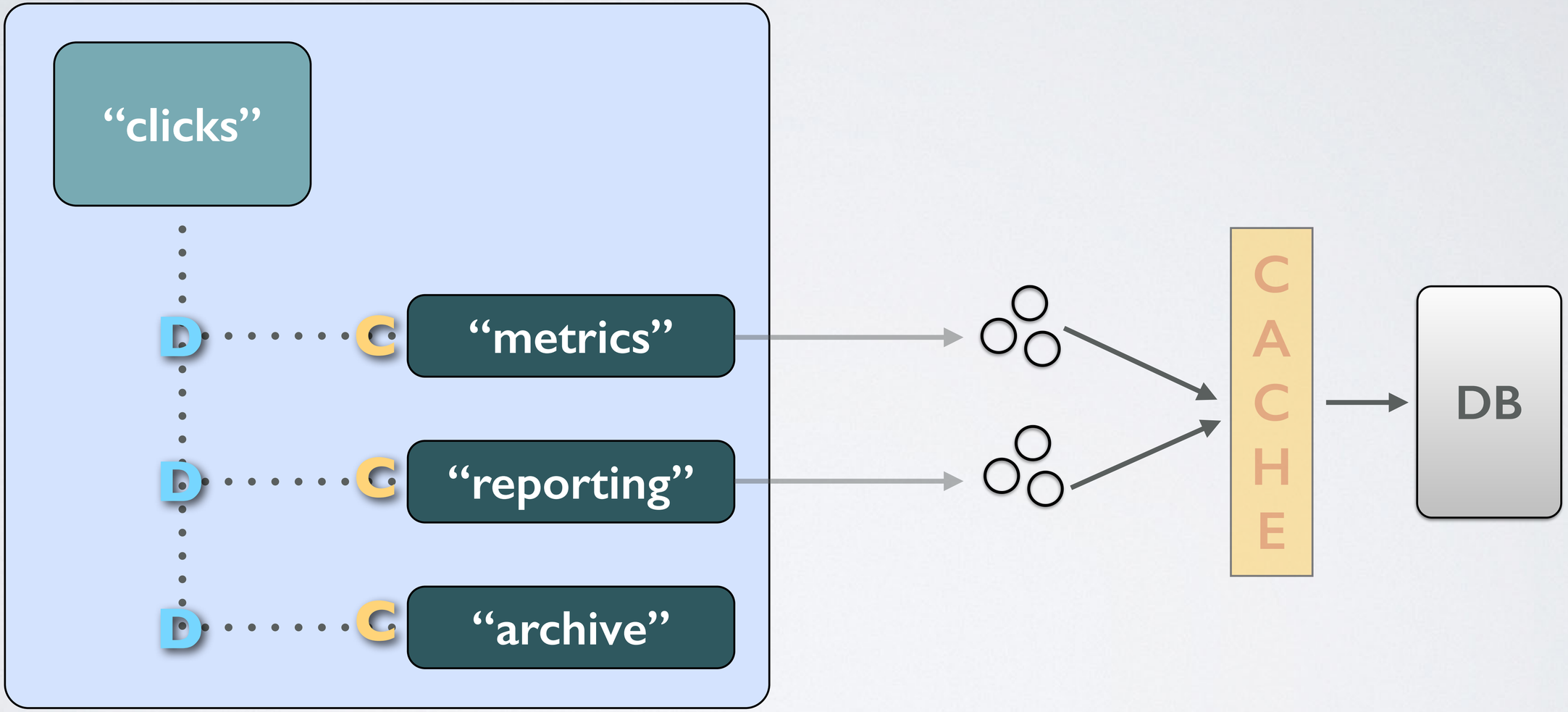
TOPIC & CHANNEL PAUSING



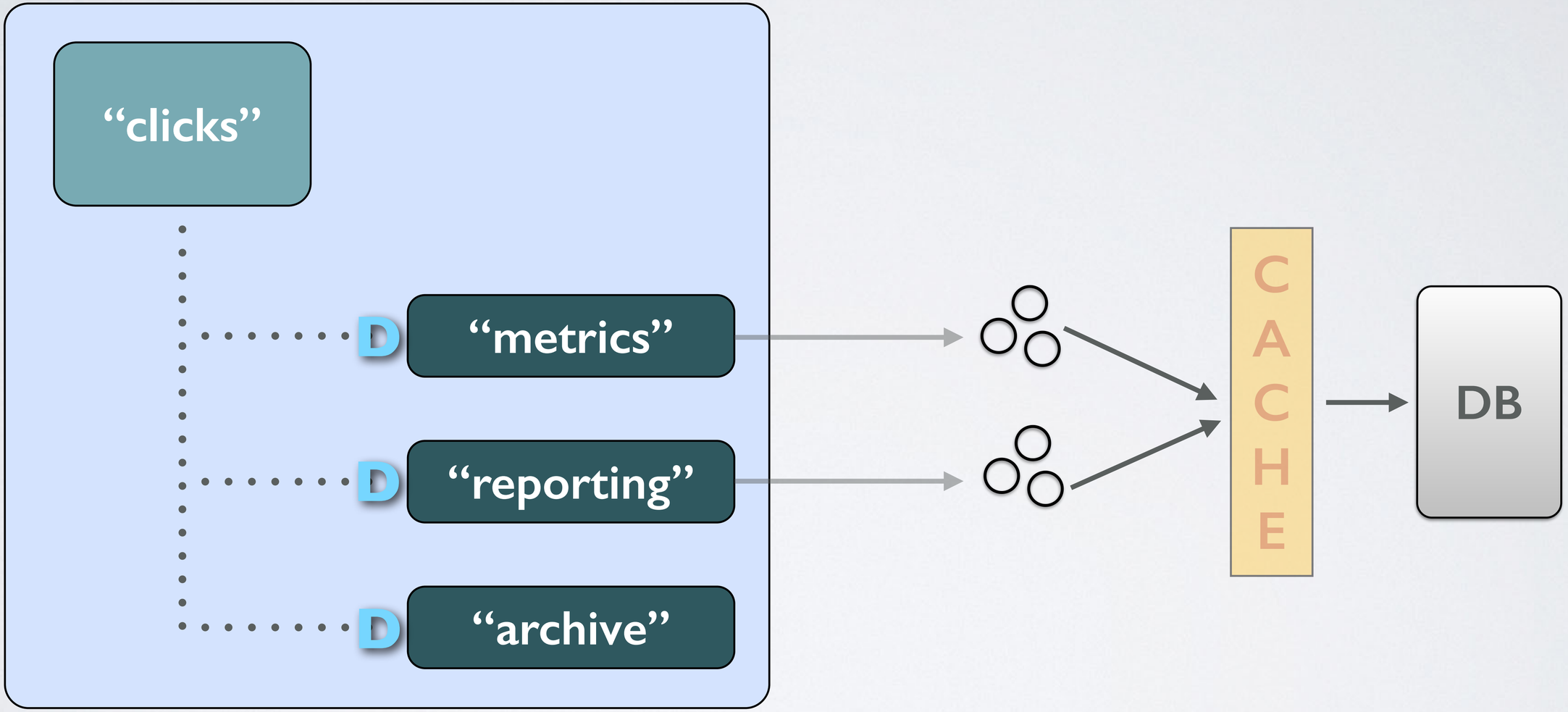
TOPIC & CHANNEL PAUSING



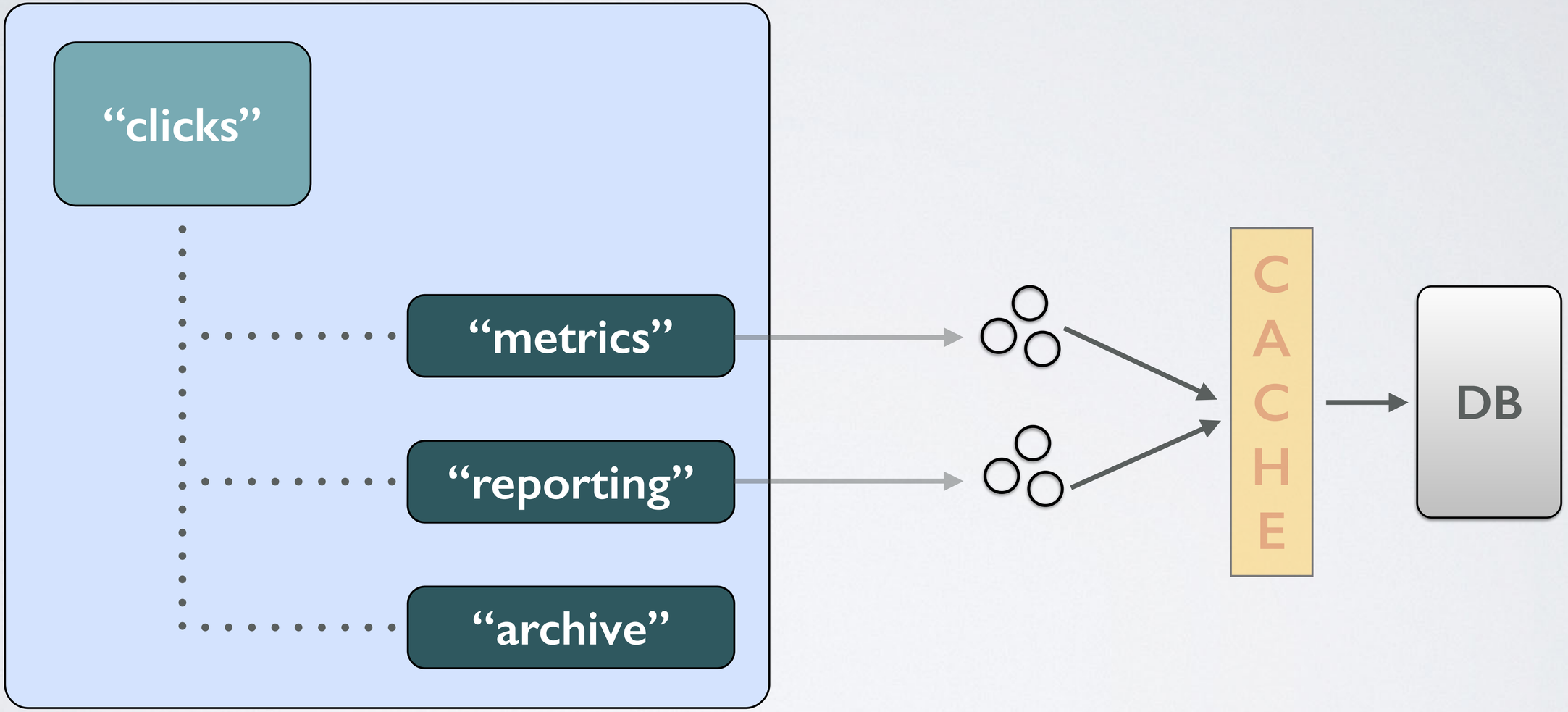
TOPIC & CHANNEL PAUSING



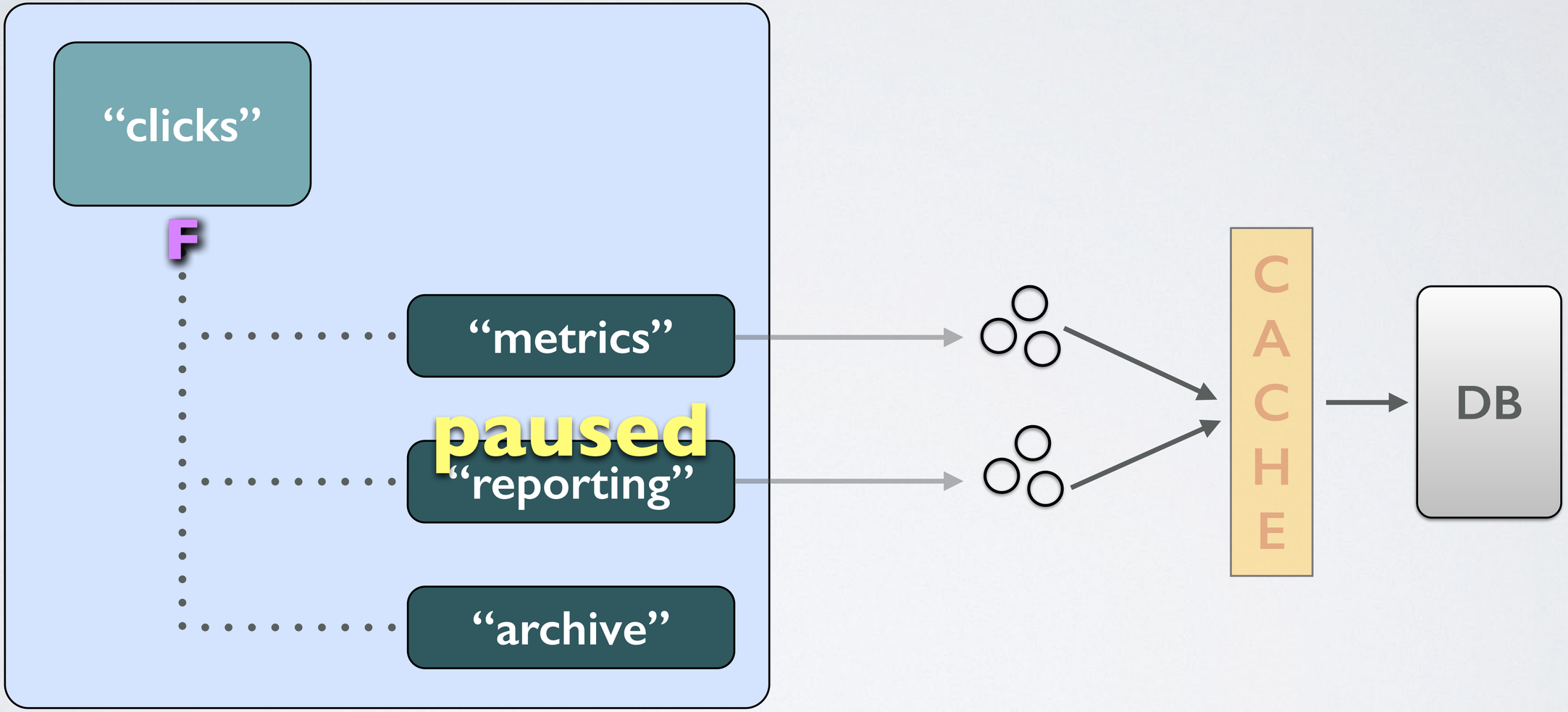
TOPIC & CHANNEL PAUSING



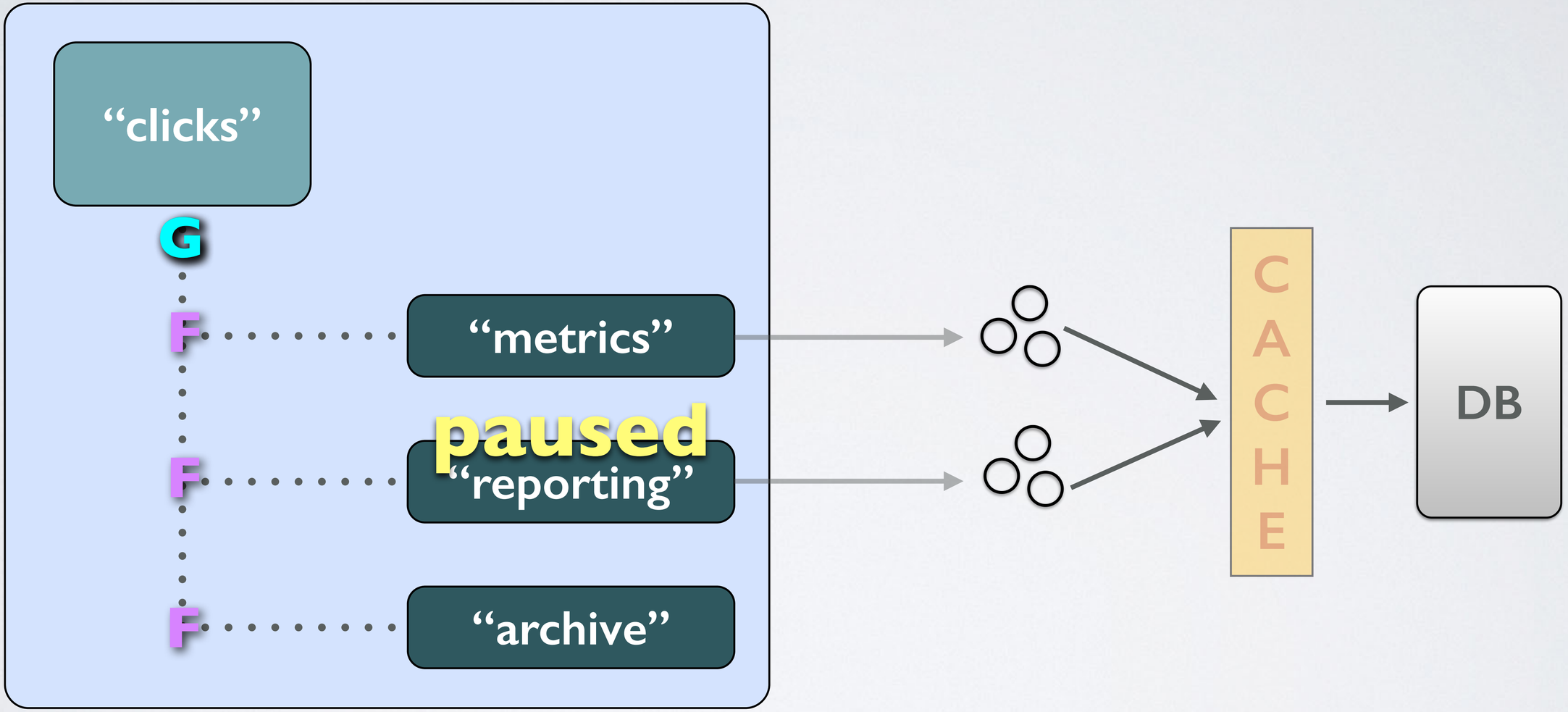
TOPIC & CHANNEL PAUSING



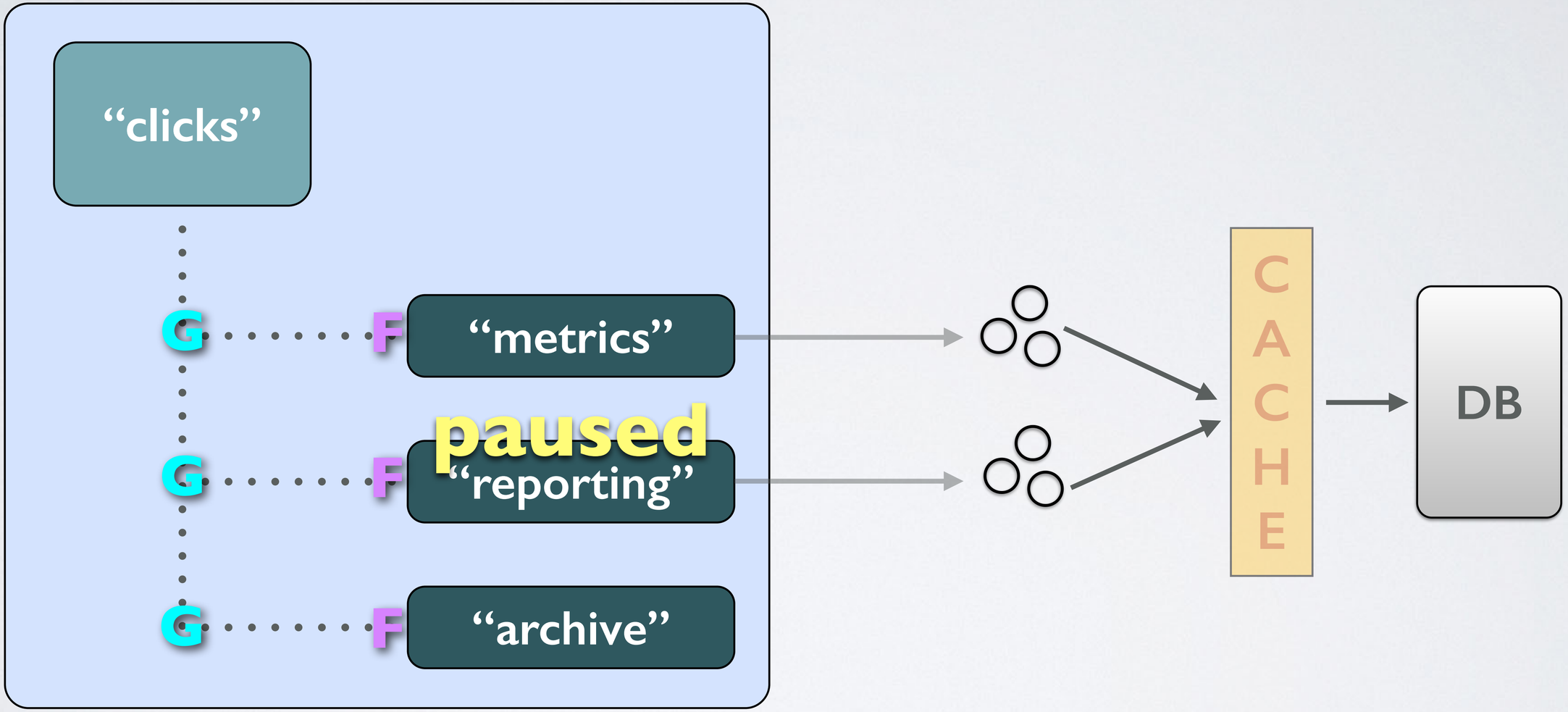
TOPIC & CHANNEL PAUSING



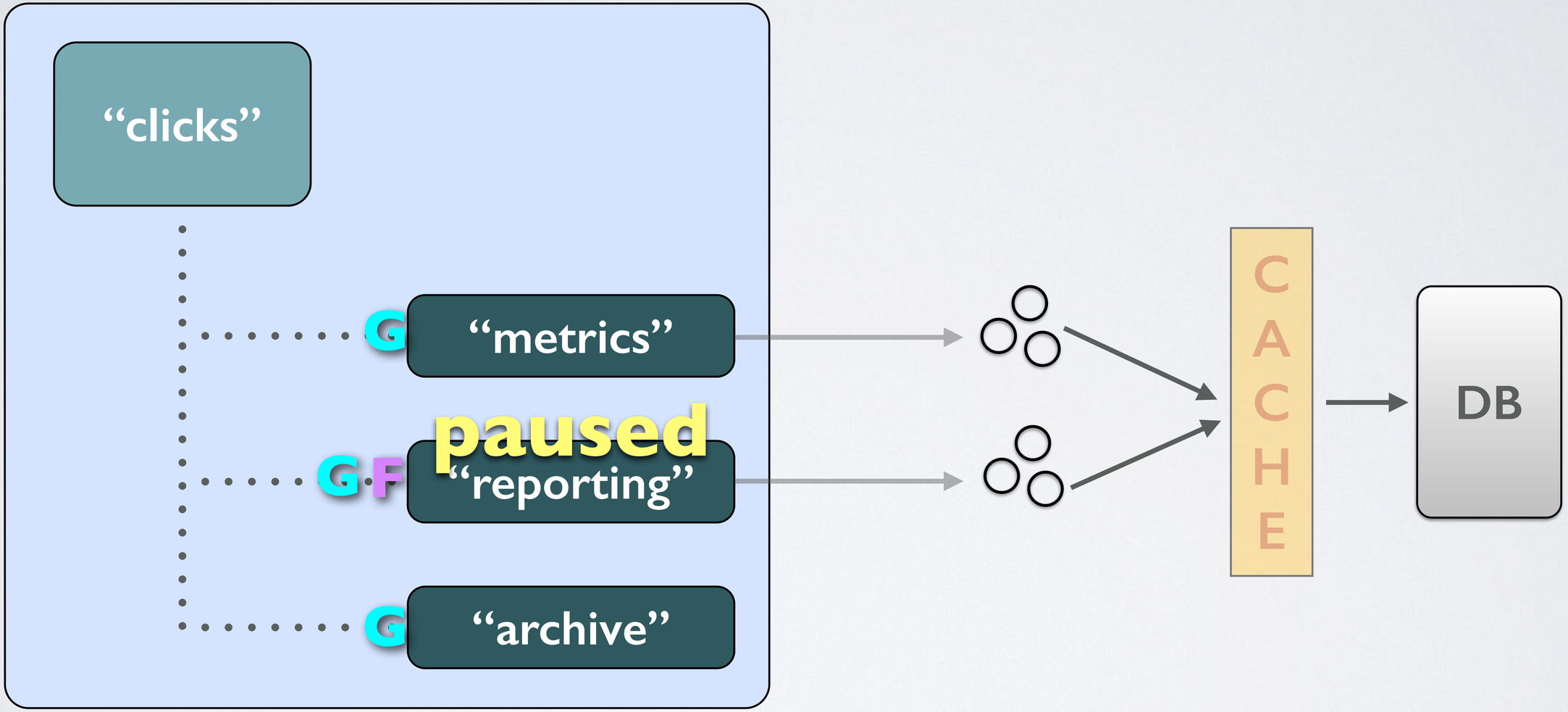
TOPIC & CHANNEL PAUSING



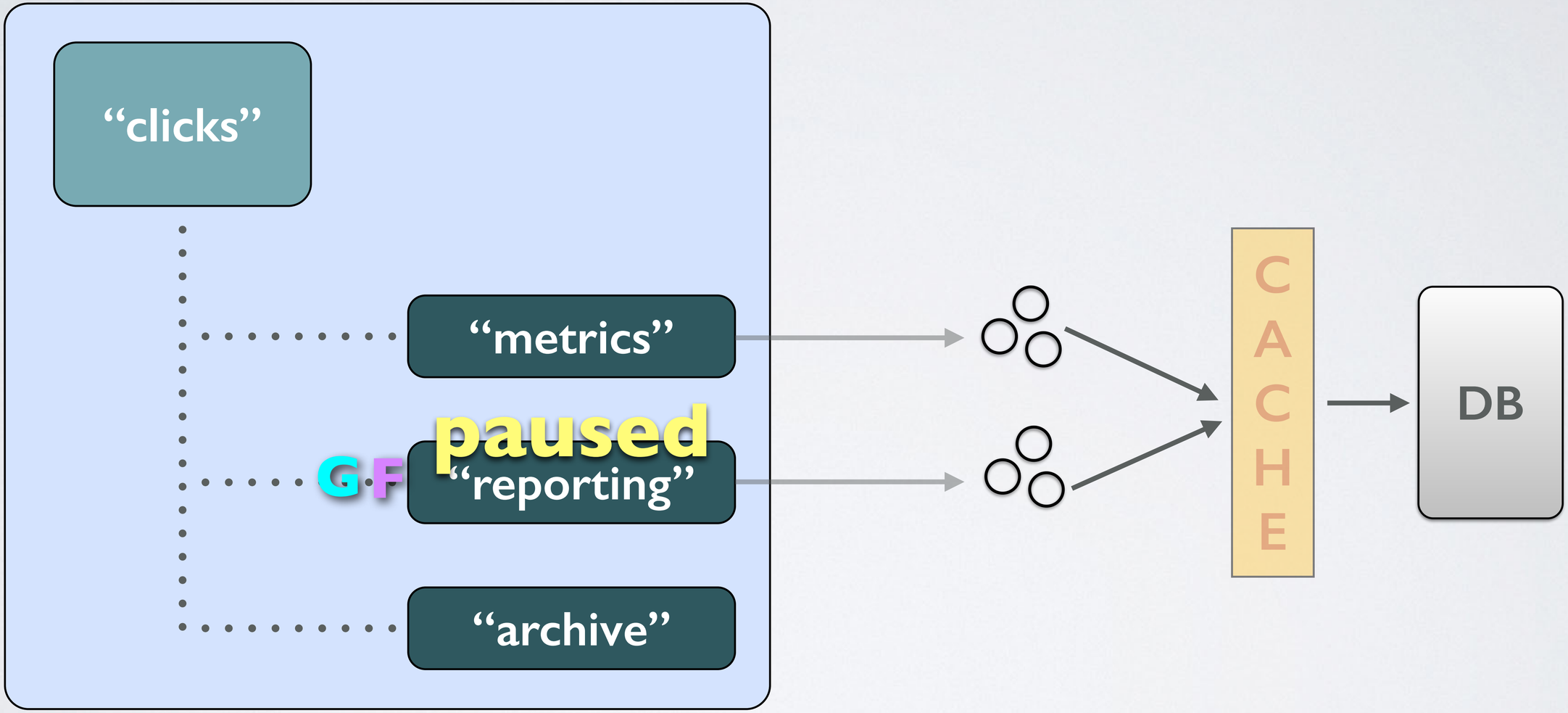
TOPIC & CHANNEL PAUSING



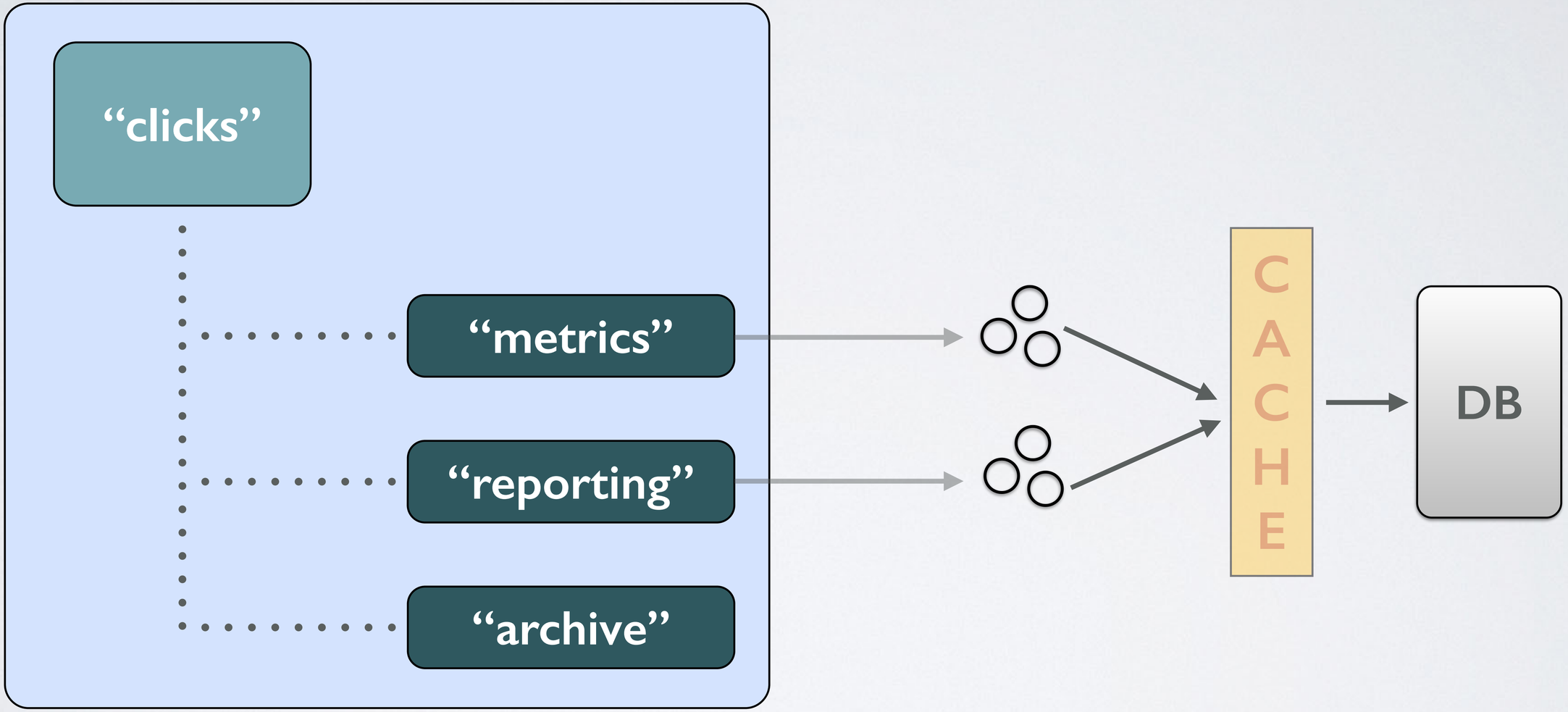
TOPIC & CHANNEL PAUSING



TOPIC & CHANNEL PAUSING

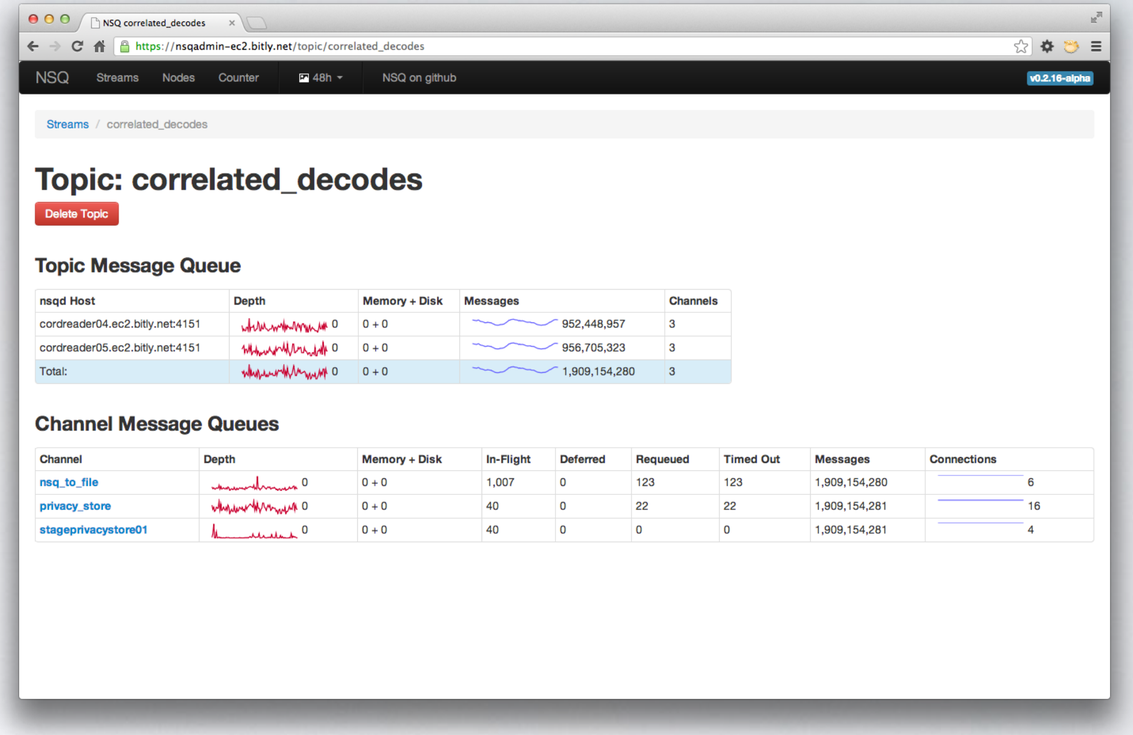


TOPIC & CHANNEL PAUSING



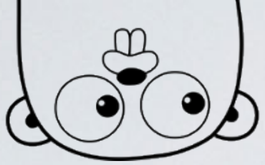
ONE MORE THING

- **#ephemeral** channels
- channel **sampling**
- TLS / Snappy
- telemetry over statsd / HTTP



SUMMARY

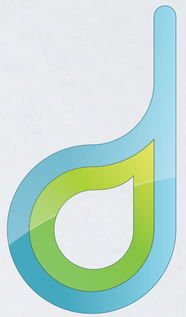
- github.com/bitly/nsq
- 10,300 lines of Go
- 19 client libraries, 11 languages
- over 2 years in production



SUMMARY

- github.com/bitly/nsq
- 10,300 lines of Go
- 19 client libraries, 11 languages
- over 2 years in production

IN PRODUCTION



heavy water

Thanks!

@imsnakes

https://github.com/bitly/nsq

shoutout to ***@jehiah*** *(co-author of NSQ)*