THE ONE ABOUT KAFKA

LILY MARA

OneSignal

Refactoring to Rust

Lily Mara
Joel Holmes

MEAP

MANNING

ONCE UPON A TIME

ONCE UPON A TIME

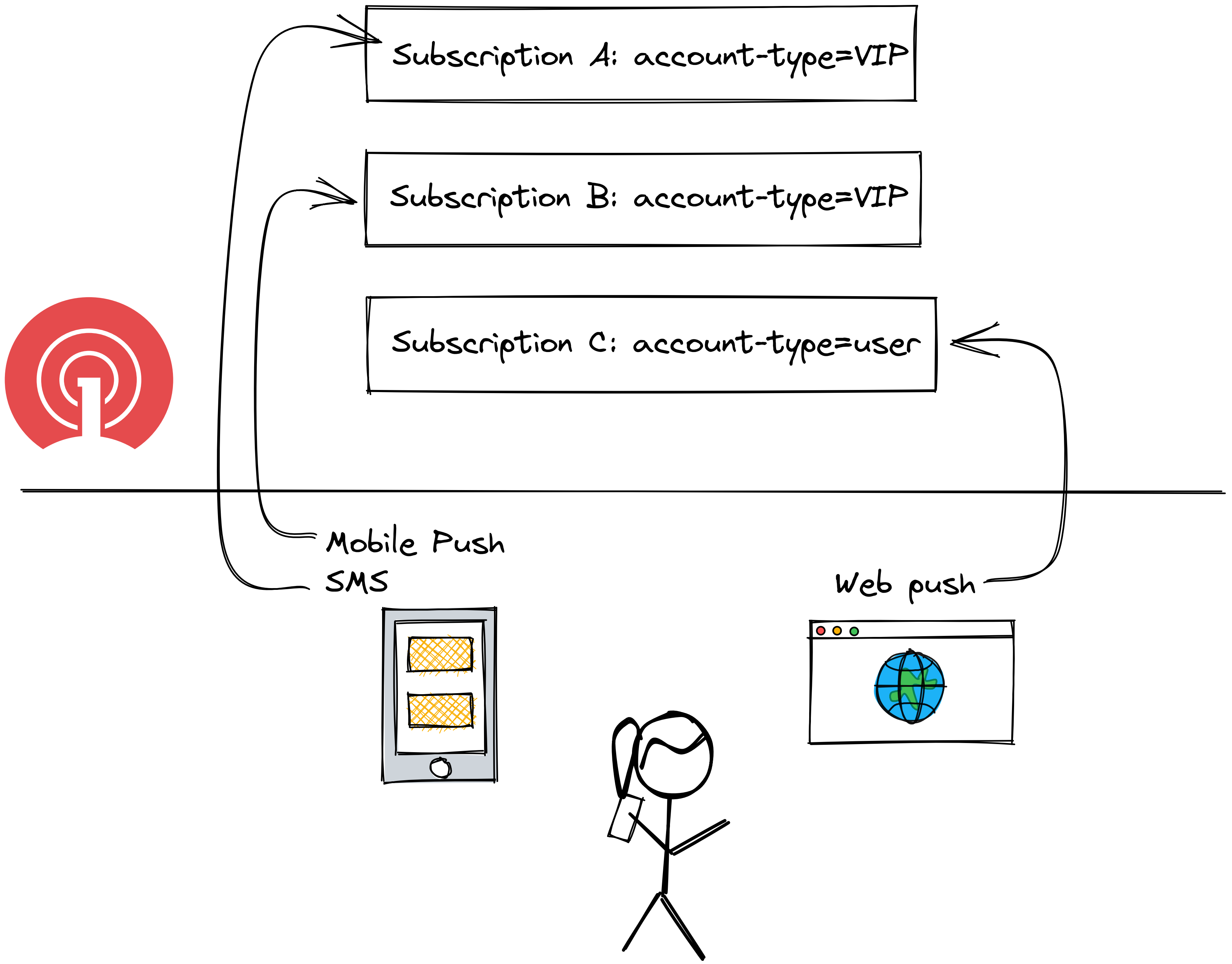
- 8B notifications/day

ONCE UPON A TIME

- 8B notifications/day
- 10 backend engineers

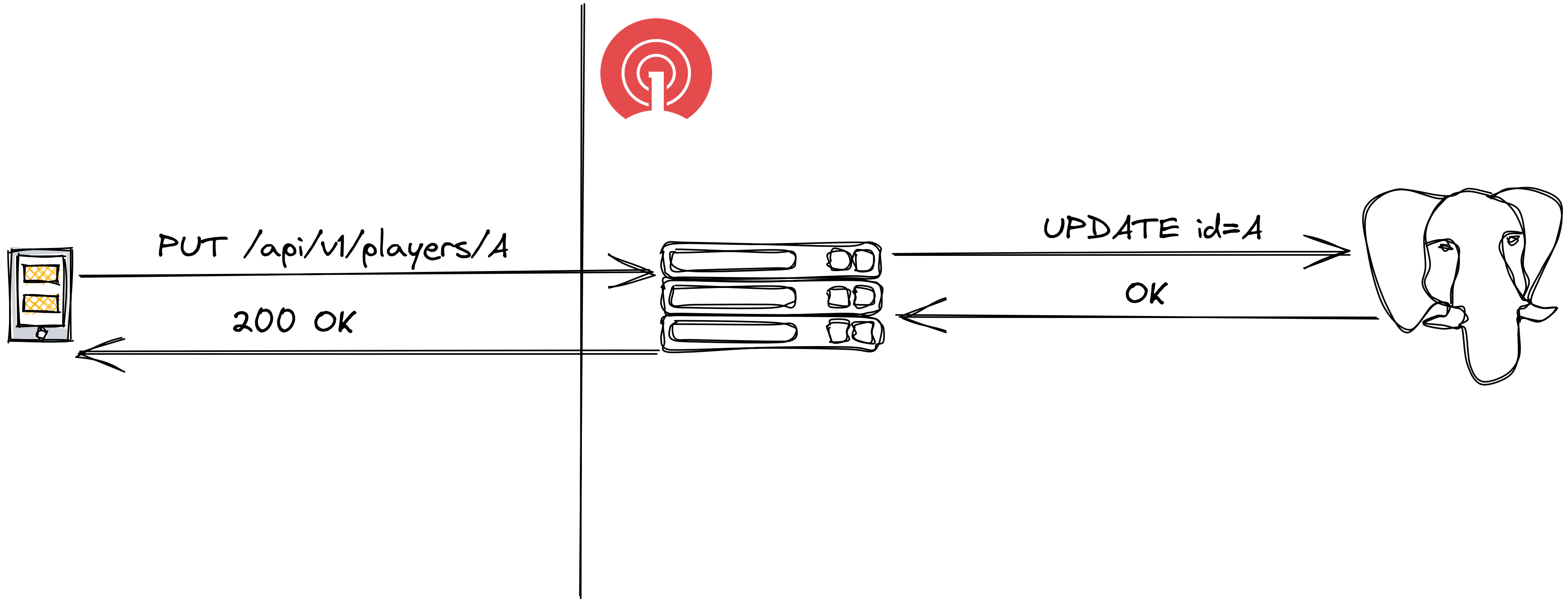
ONCE UPON A TIME

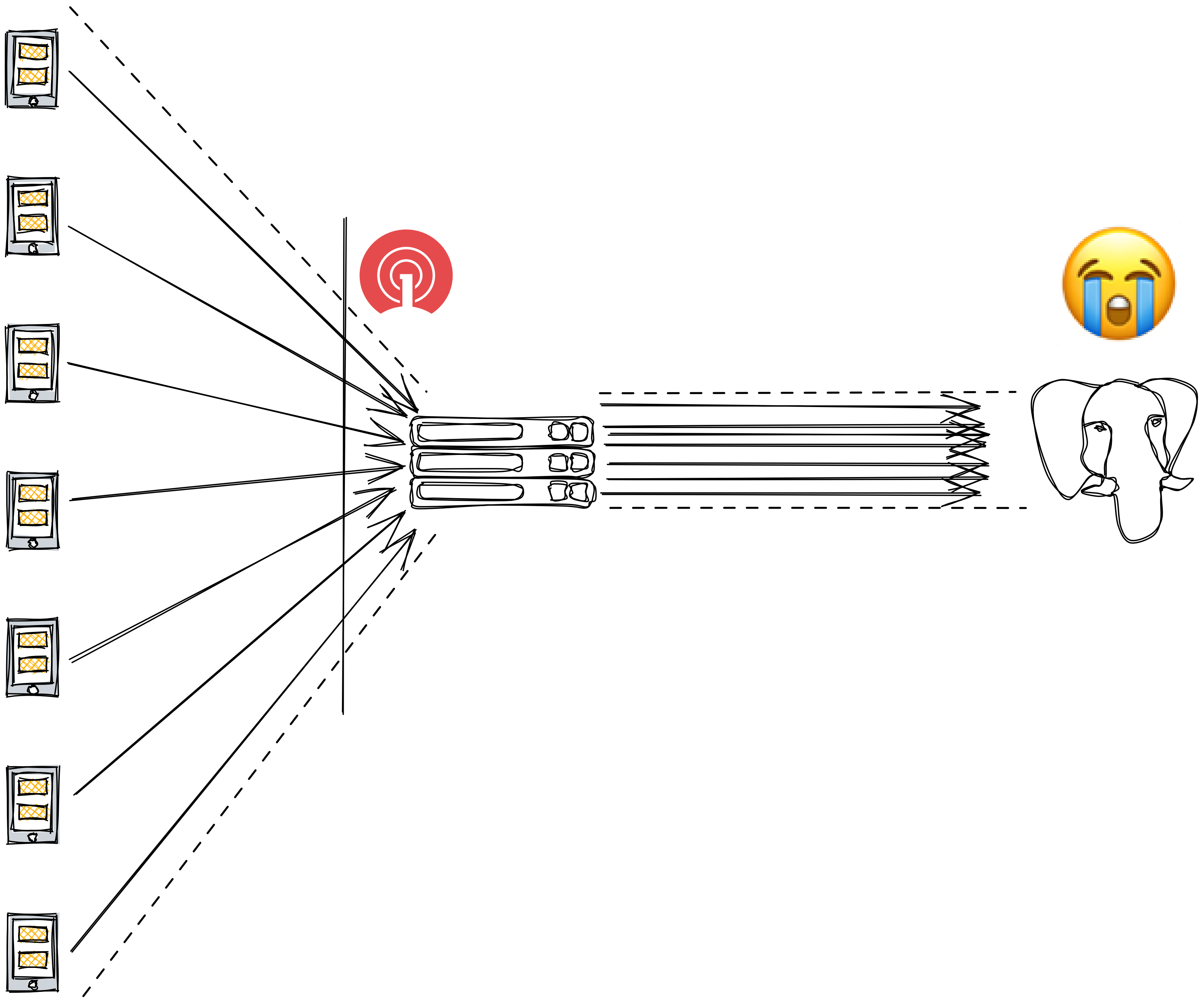
- 8B notifications/day
- 10 backend engineers
- Make simplifying assumptions


Subscription A: account-type=VIP
Subscription B: account-type=VIP
Subscription C: account-type=user
Mobile Push
SMS
Web push

API

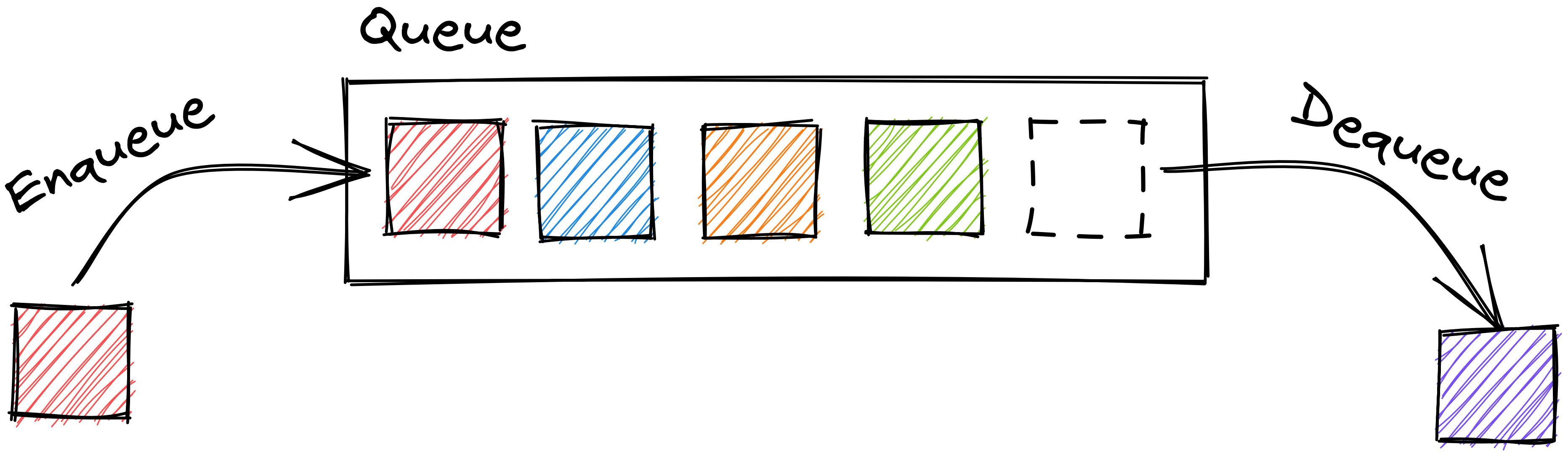
```
PUT https://onesignal.com/api/v1/players/{SUBSCRIPTION_ID}
{
     "app_id": "{APP_ID}",
     "tags": {
          "first_name": "Jon",
          "last_name": "Smith",
     }
}
```


PUT /api/v1/players/A
200 OK
UPDATE id=A
OK

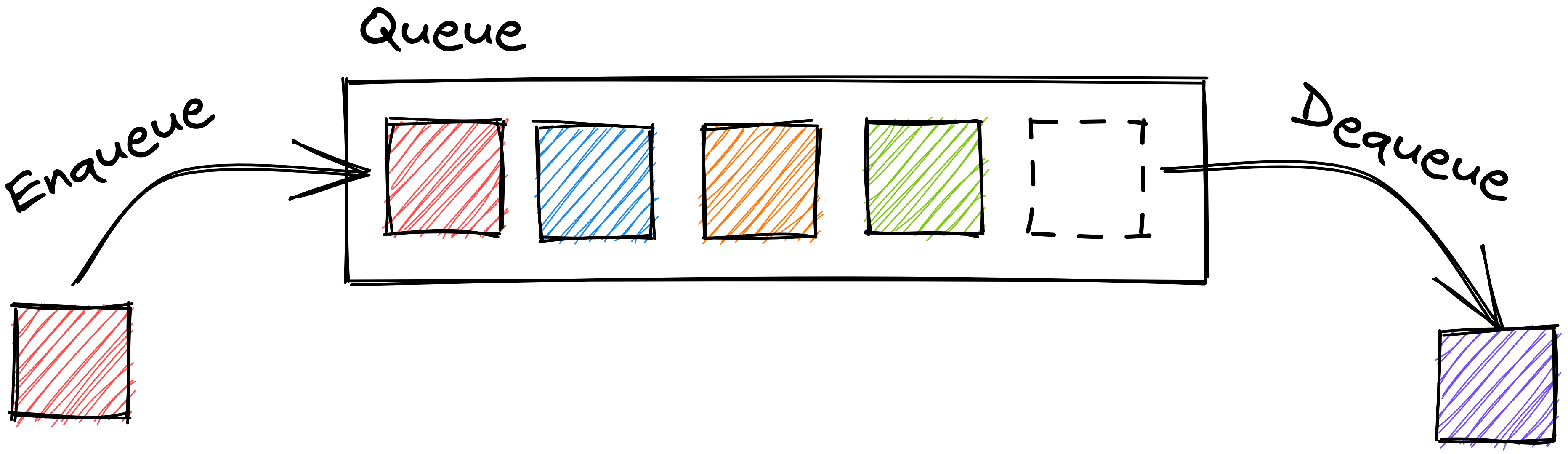


QUEUE

QUEUE



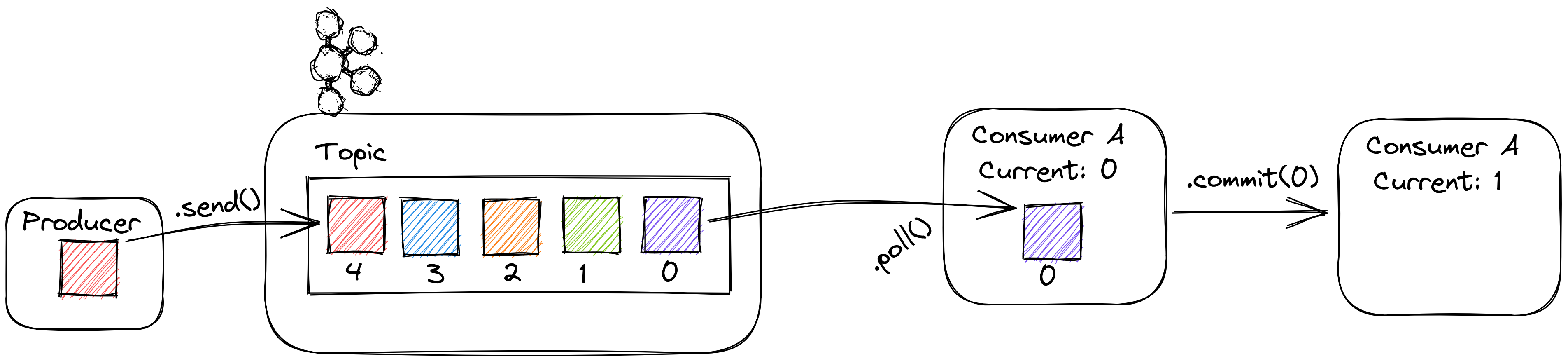
QUEUE



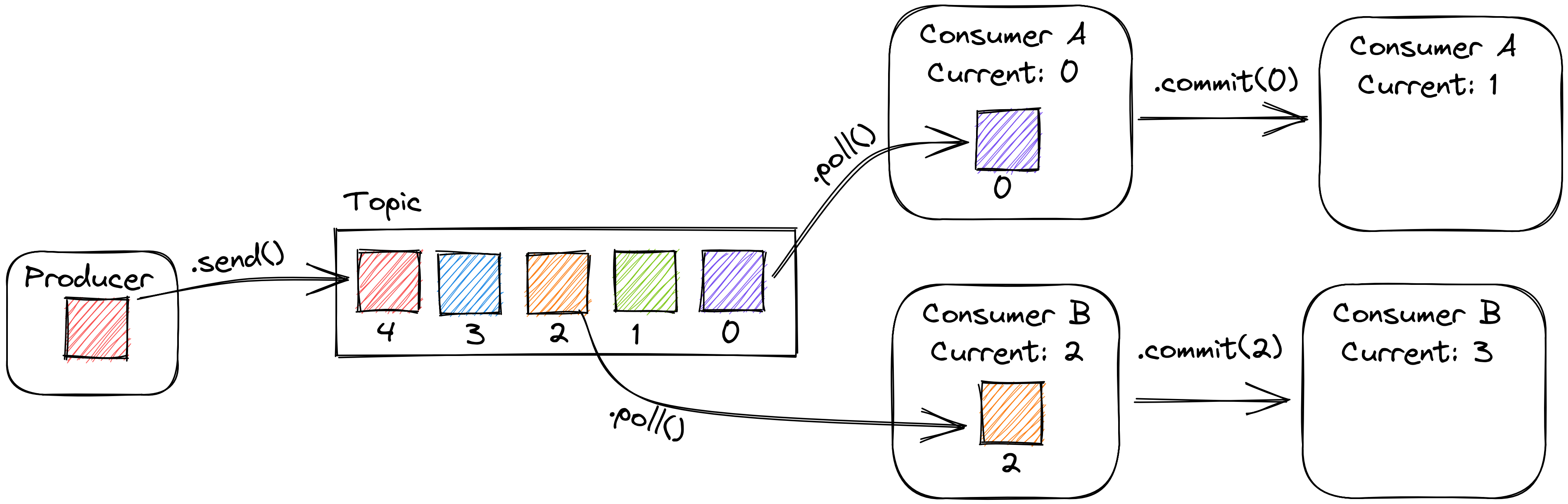
Metric: Lag=4

kafka

KAFKA PIPELINE

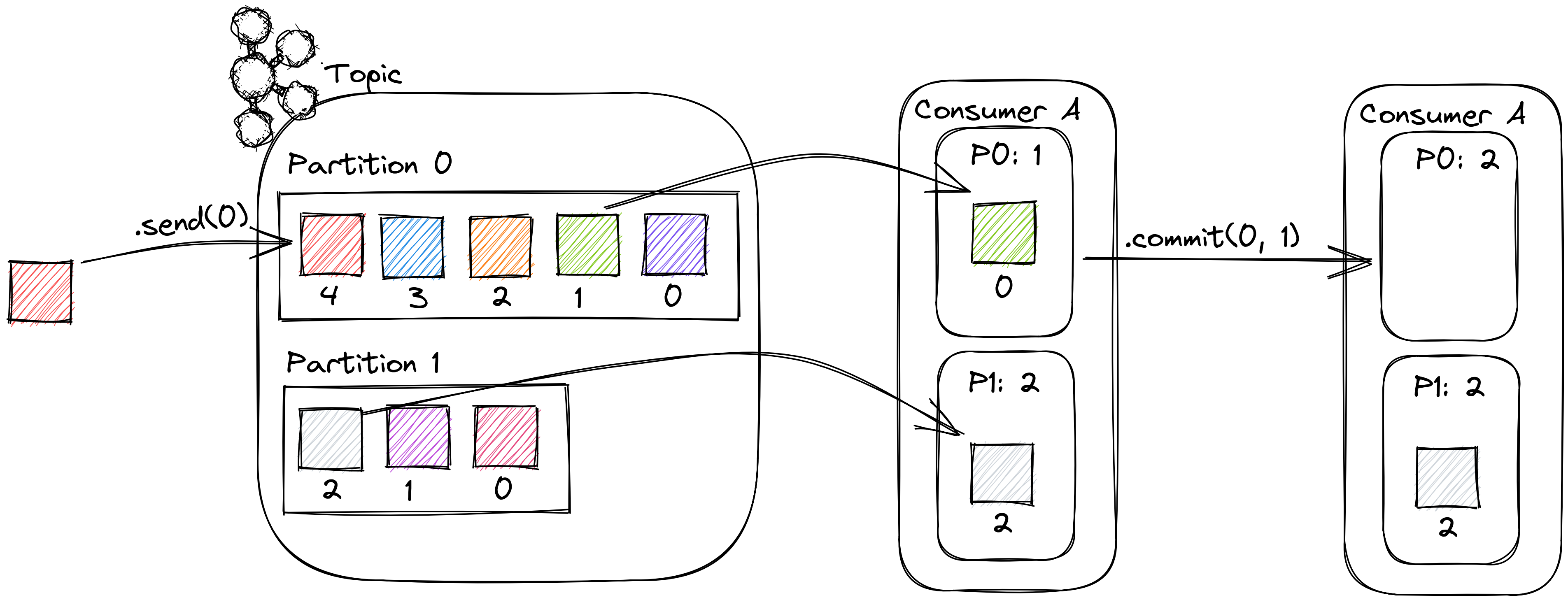


CONSUMER

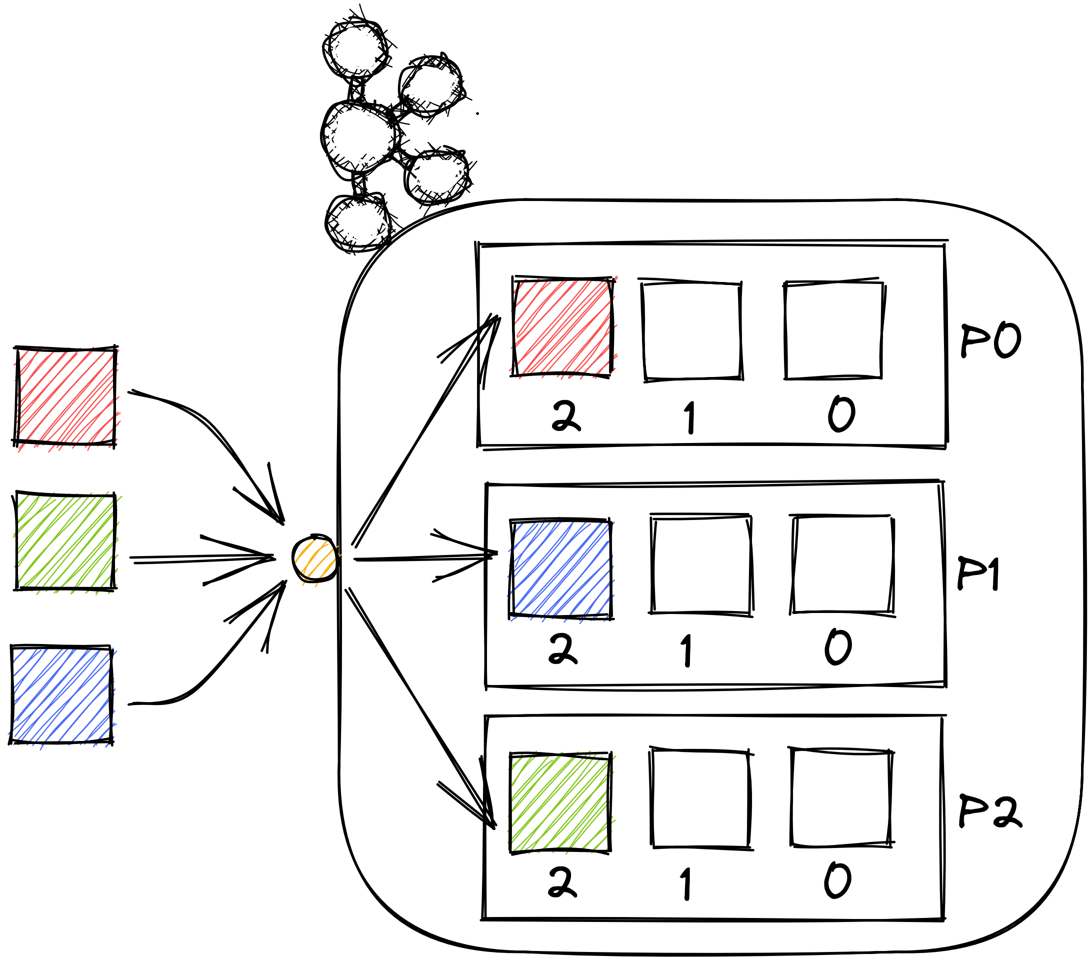


Producer
.send()
Topic
4
3
2
1
0
.poll()
.poll()
Consumer A
Current: 0
0
.commit(0)
Consumer A
Current: 1
Consumer B
Current: 2
2
.commit(2)
Consumer B
Current: 3

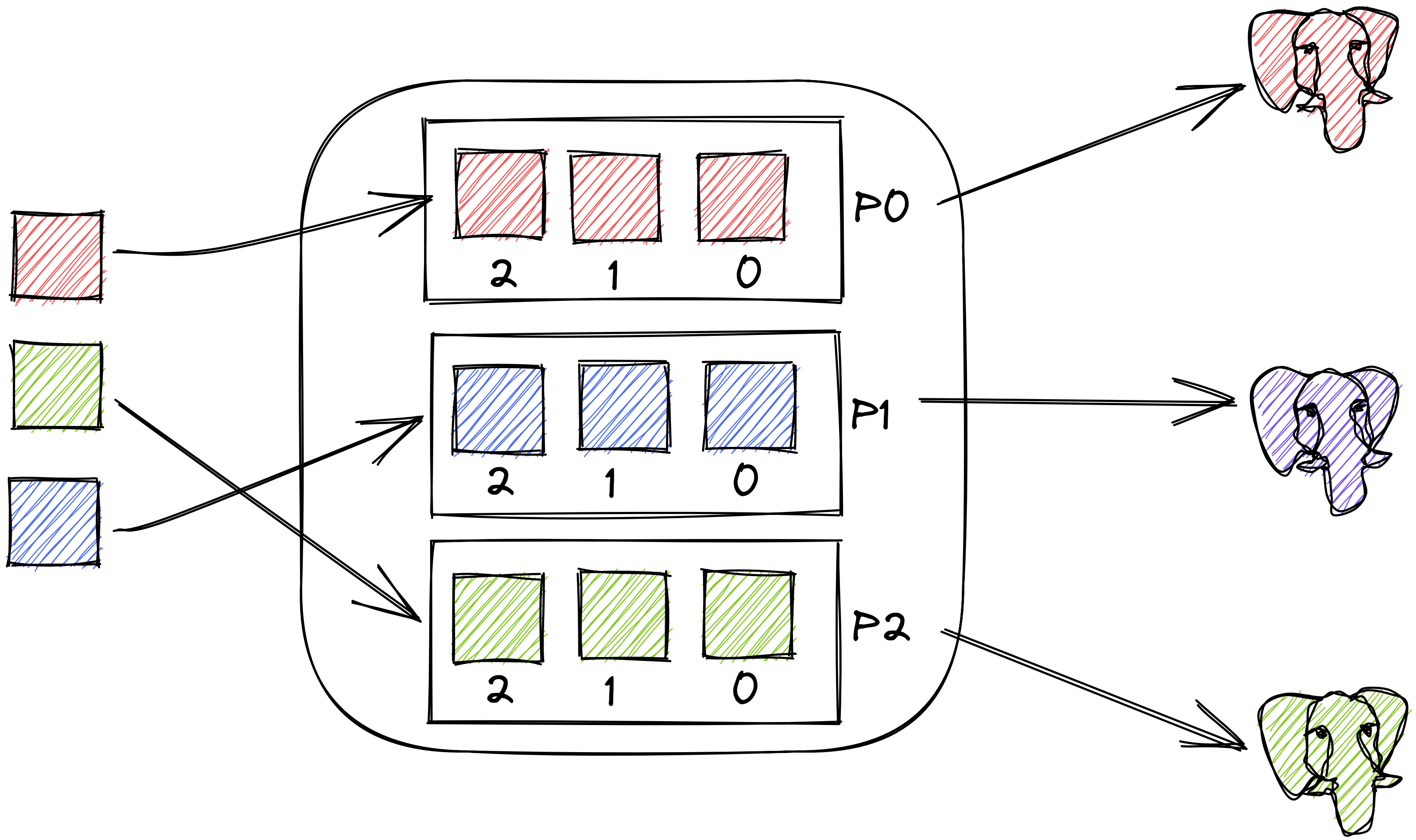
PARTITION


Topic
.send(0)
Partition 0
4
3
2
1
0
Partition 1
2
1
0
Consumer A
P0: 1
0
P1: 2
2
.commit(0, 1)
Consumer A
P0: 2
P1: 2
2

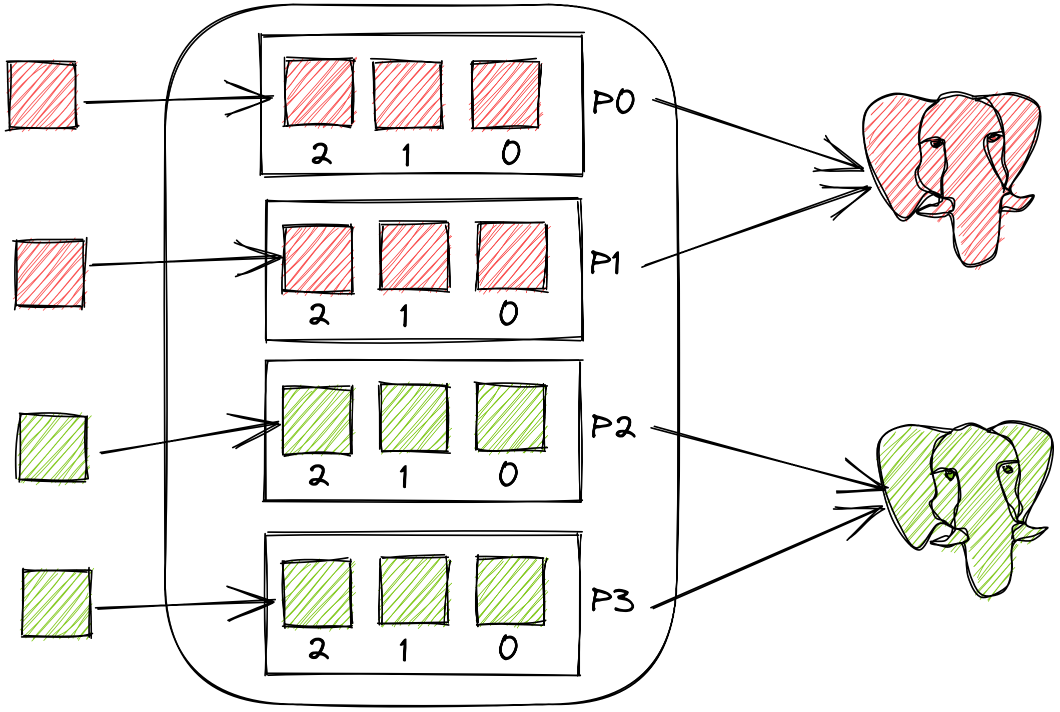
ROUND-ROBIN



EXPLICIT



CONCURRENT WRITES



ISSUES

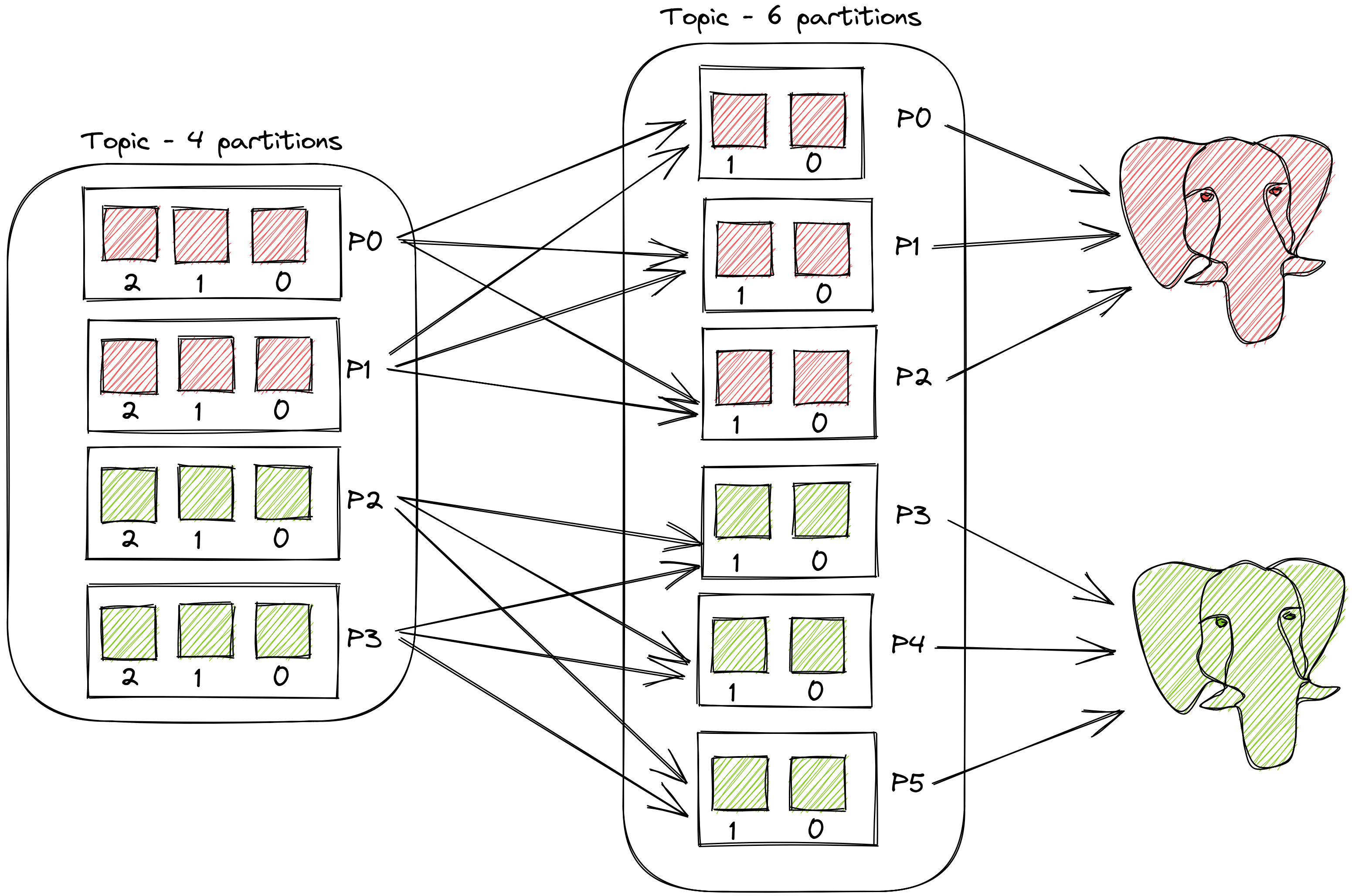
ISSUES

- Inflexible

ISSUES

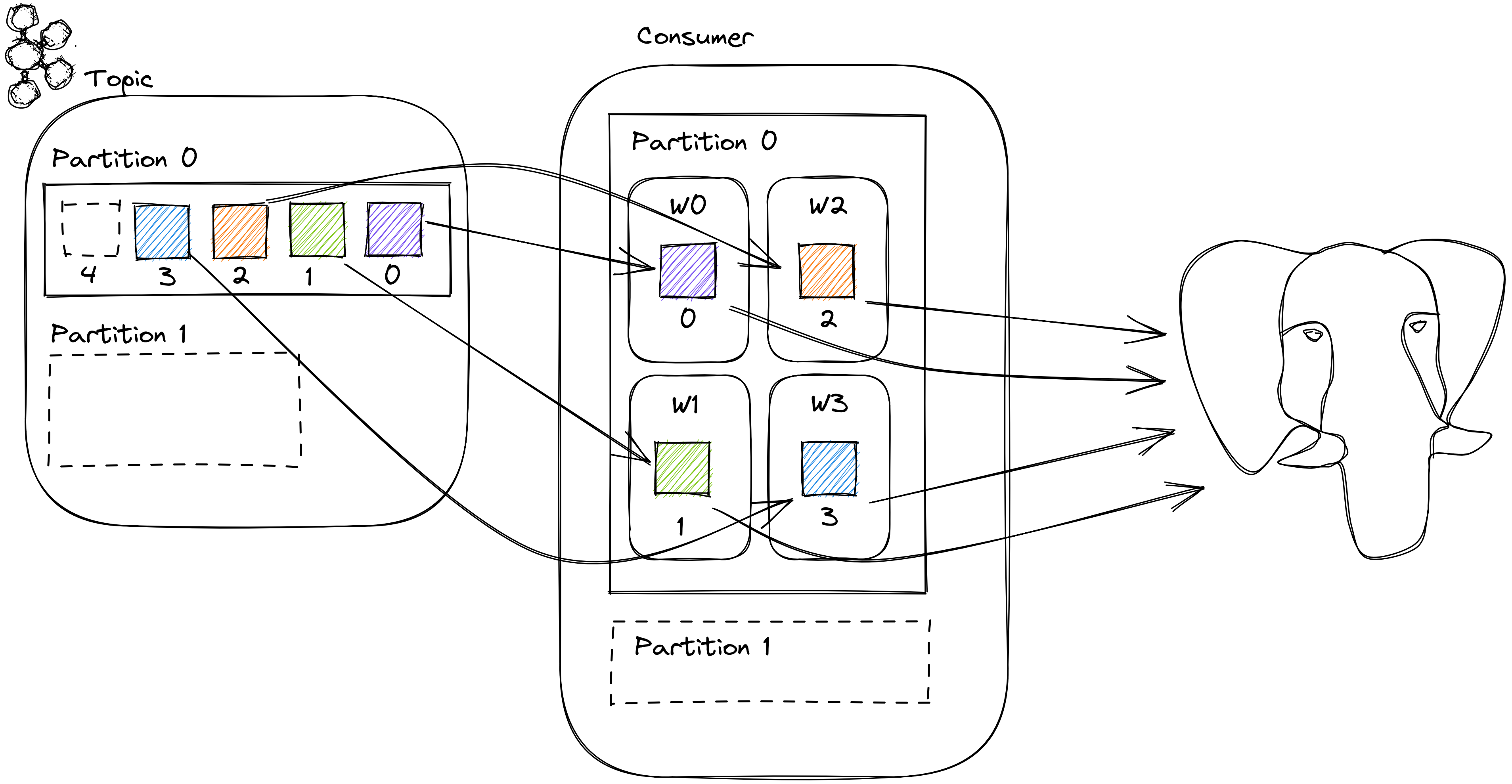
- Inflexible
- Kafka repartitioning

REPARTITIONING

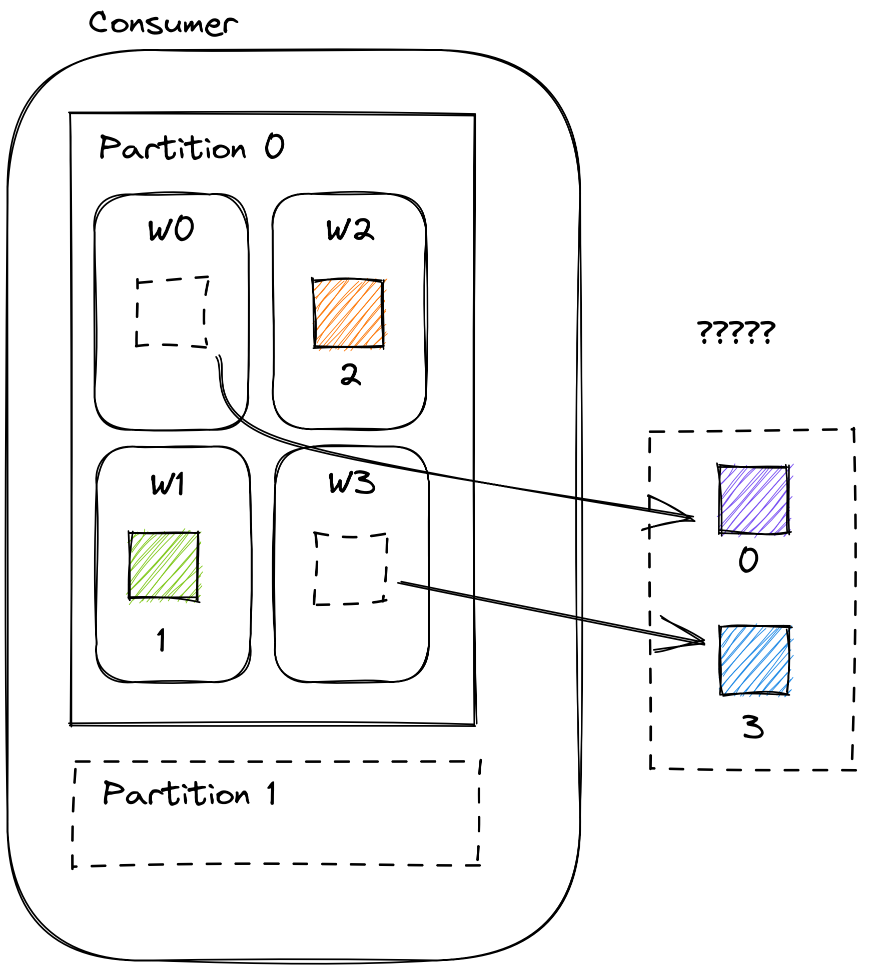


SUBPARTITION PROCESSING

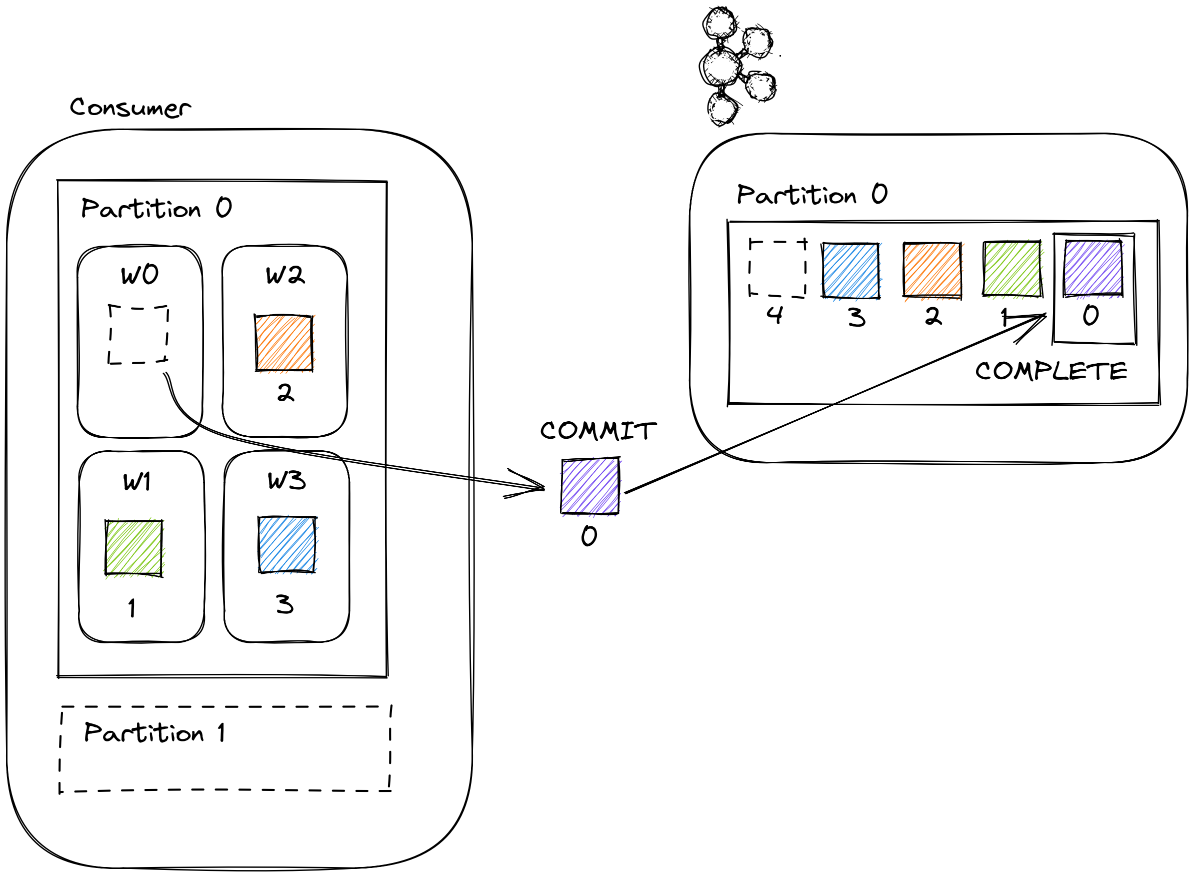
SUBPARTITION PROCESSING



ISSUE

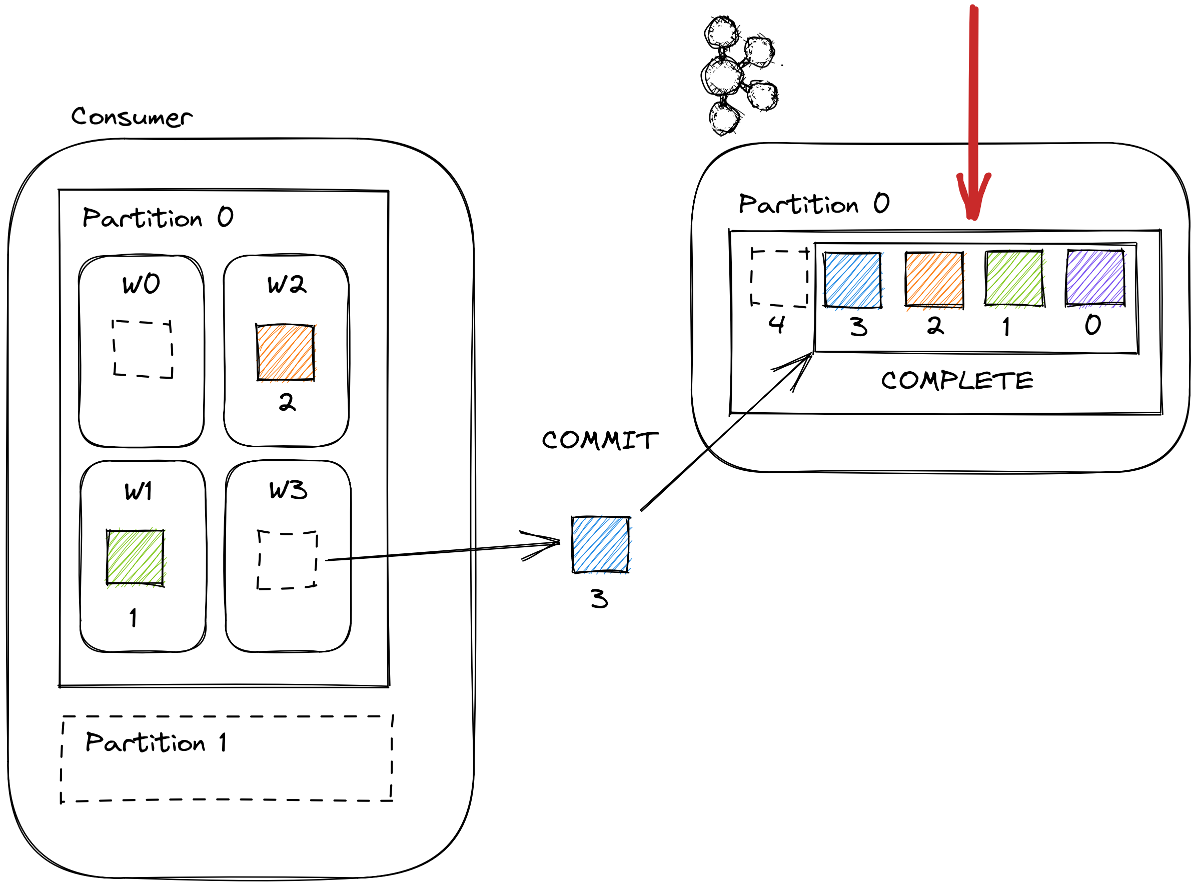


COMMIT(0)



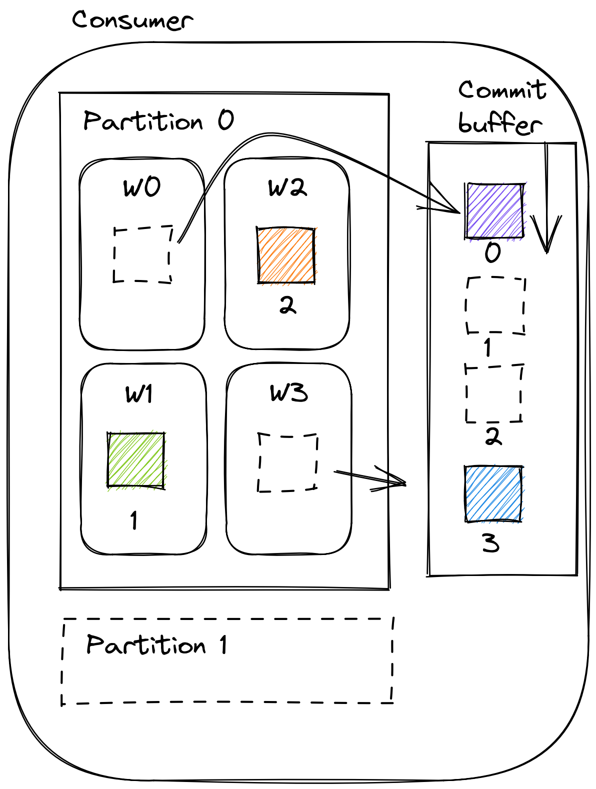
Consumer
Partition 0
W0
W2
2
W1
1
W3
3
Partition 1
COMMIT
0
Partition 0
4
3
2
1
0
COMPLETE

COMMIT(3)

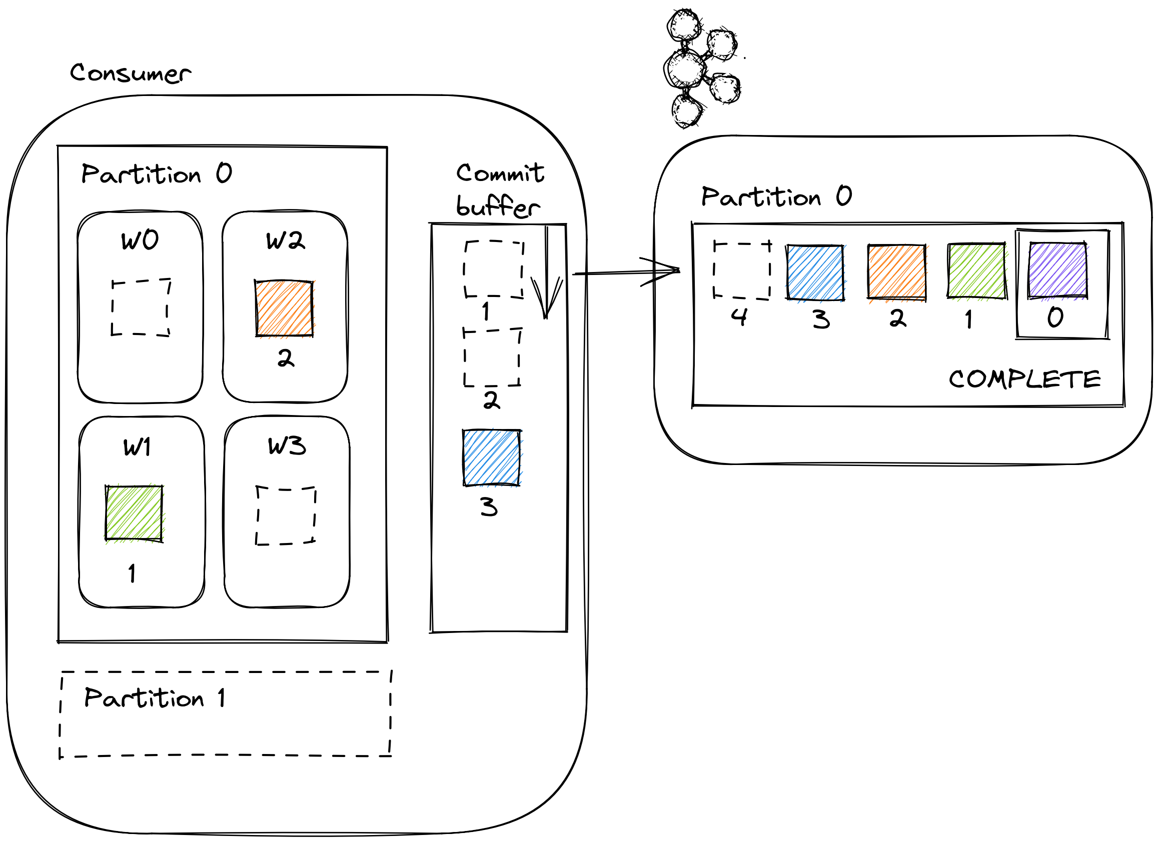


Consumer
Partition 0
W0
W2
2
W1
W3
1
Partition 1
COMMIT
3
Partition 0
4
3
2
1
0
COMPLETE

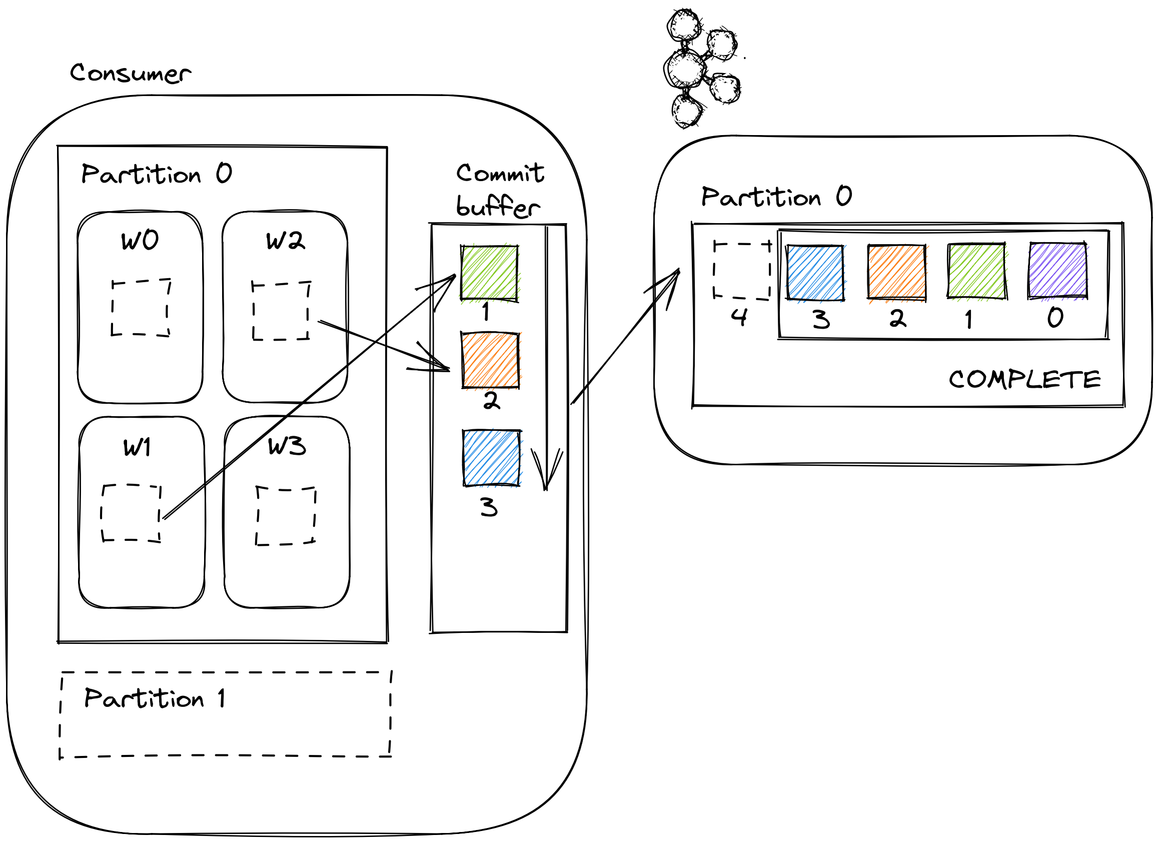
SOLUTION


Consumer
Partition 0
W0
W2
2
W1
1
W3
Commit buffer
0
1
2
3
Partition 1

COMMIT(0)


Consumer
Partition 0
W0
W2
2
W1
1
W3
Commit buffer
1
2
3
Partition 1
Partition 0
4
3
2
1
0
COMPLETE

COMMIT(3)


Consumer
Partition 0
W0
W2
W1
W3
Commit buffer
1
2
3
Partition 1
Partition 0
4
3
2
1
0
COMPLETE

CONCESSION

CONCESSION

- at-least-once delivery

CONCESSION

- at-least-once delivery
- messages will be replayed

CONCESSION

- at-least-once delivery
- messages will be replayed
- design around this

REVIEW

REVIEW

- kafka topic

REVIEW

- kafka topic
- contains partitions - queues

REVIEW

- kafka topic
- contains partitions - queues
- message has incrementing offset

REVIEW

- kafka topic
- contains partitions - queues
- message has incrementing offset
- producers enqueue

REVIEW

- kafka topic
- contains partitions - queues
- message has incrementing offset
- producers enqueue
- consumers dequeue

REVIEW

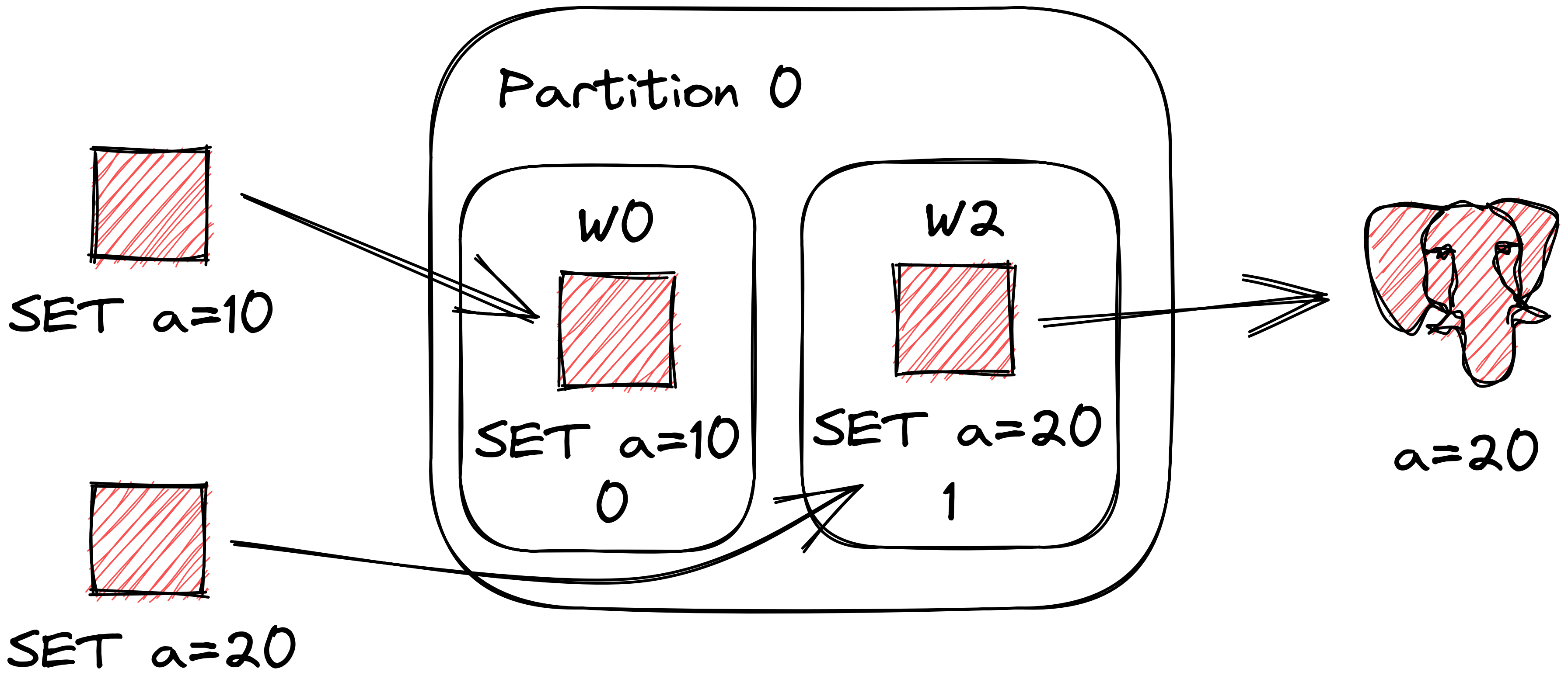
- kafka topic
- contains partitions - queues
- message has incrementing offset
- producers enqueue
- consumers dequeue
- consumer concurrency via partitioning and subpartitioning

REVIEW

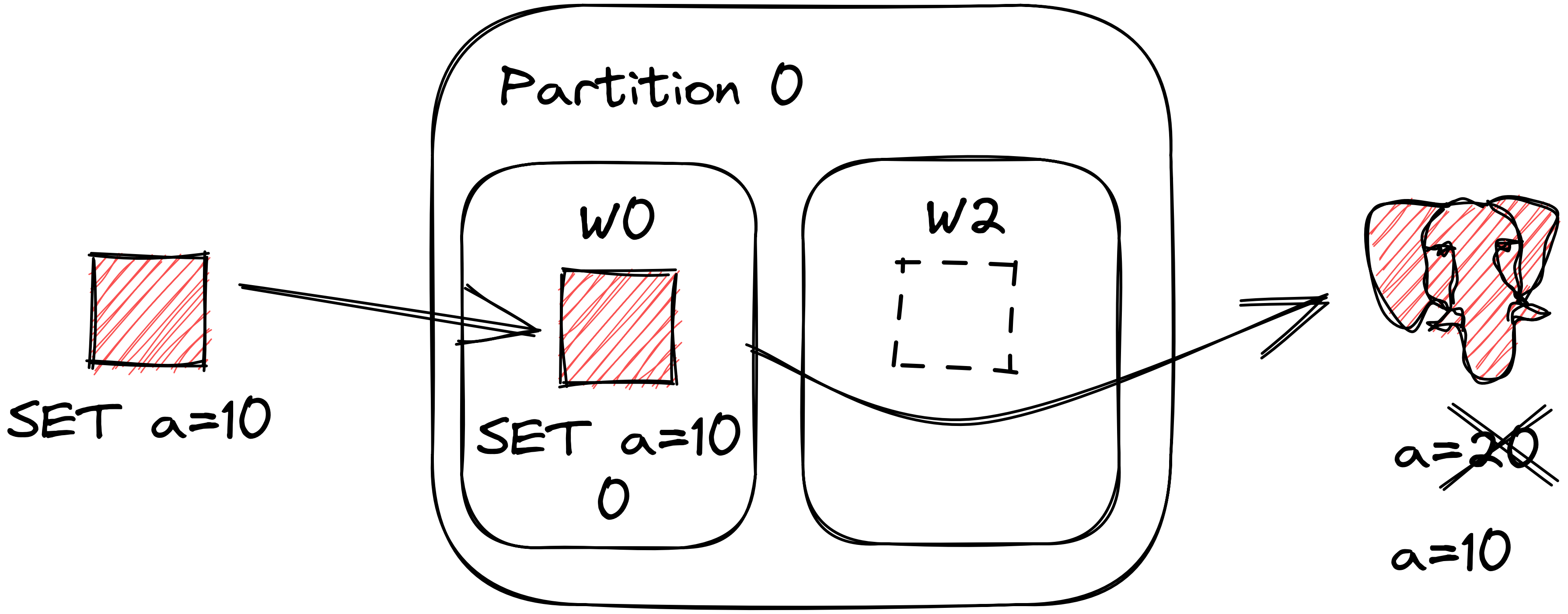
- kafka topic
- contains partitions - queues
- message has incrementing offset
- producers enqueue
- consumers dequeue
- consumer concurrency via partitioning and subpartitioning
- consumer performing PG writes

POSTGRES WRITES

CONCURRENCY



CONCURRENCY



GOALS

GOALS

- maximize concurrency

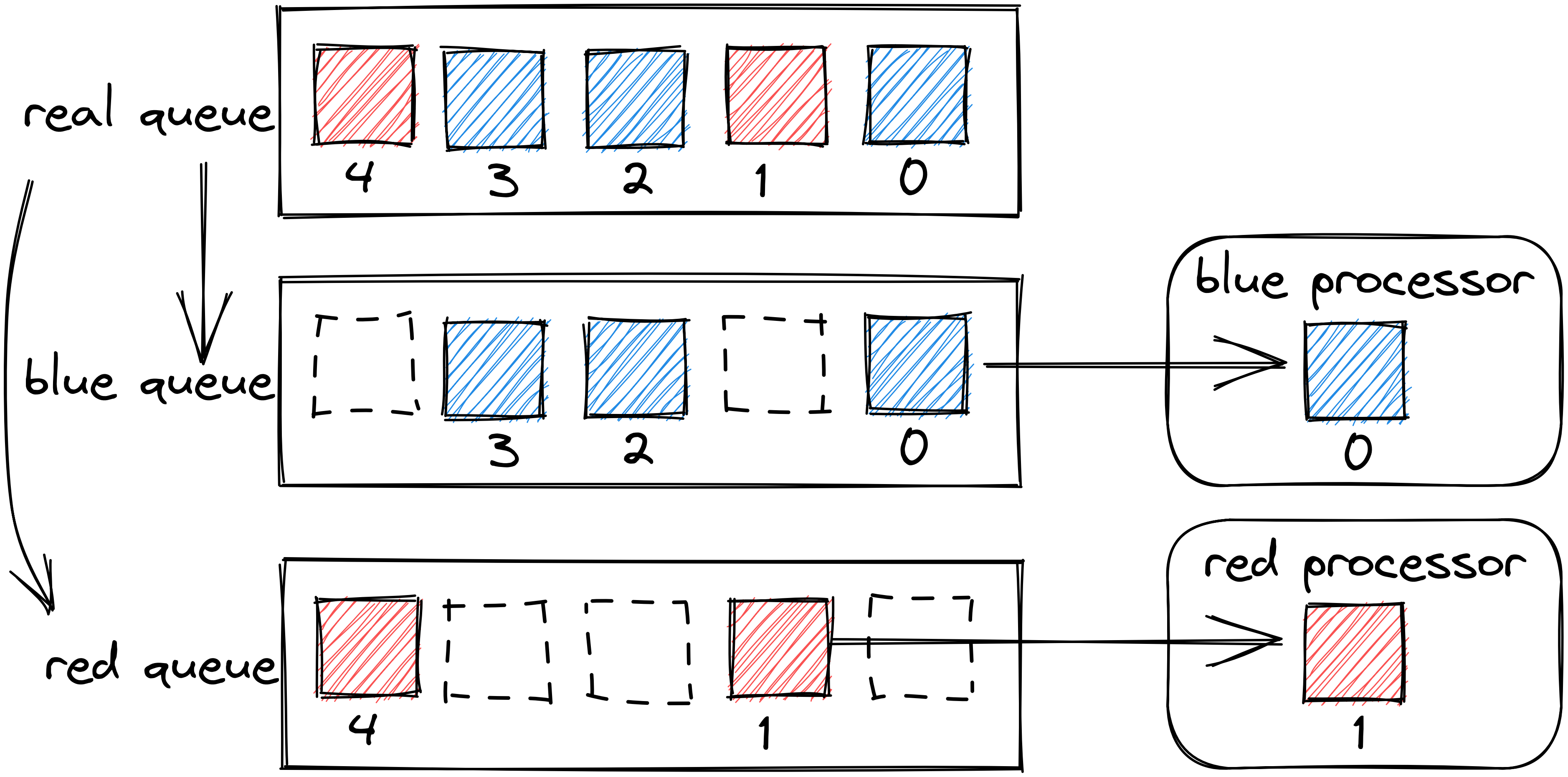
GOALS

- maximize concurrency
- minimize contention

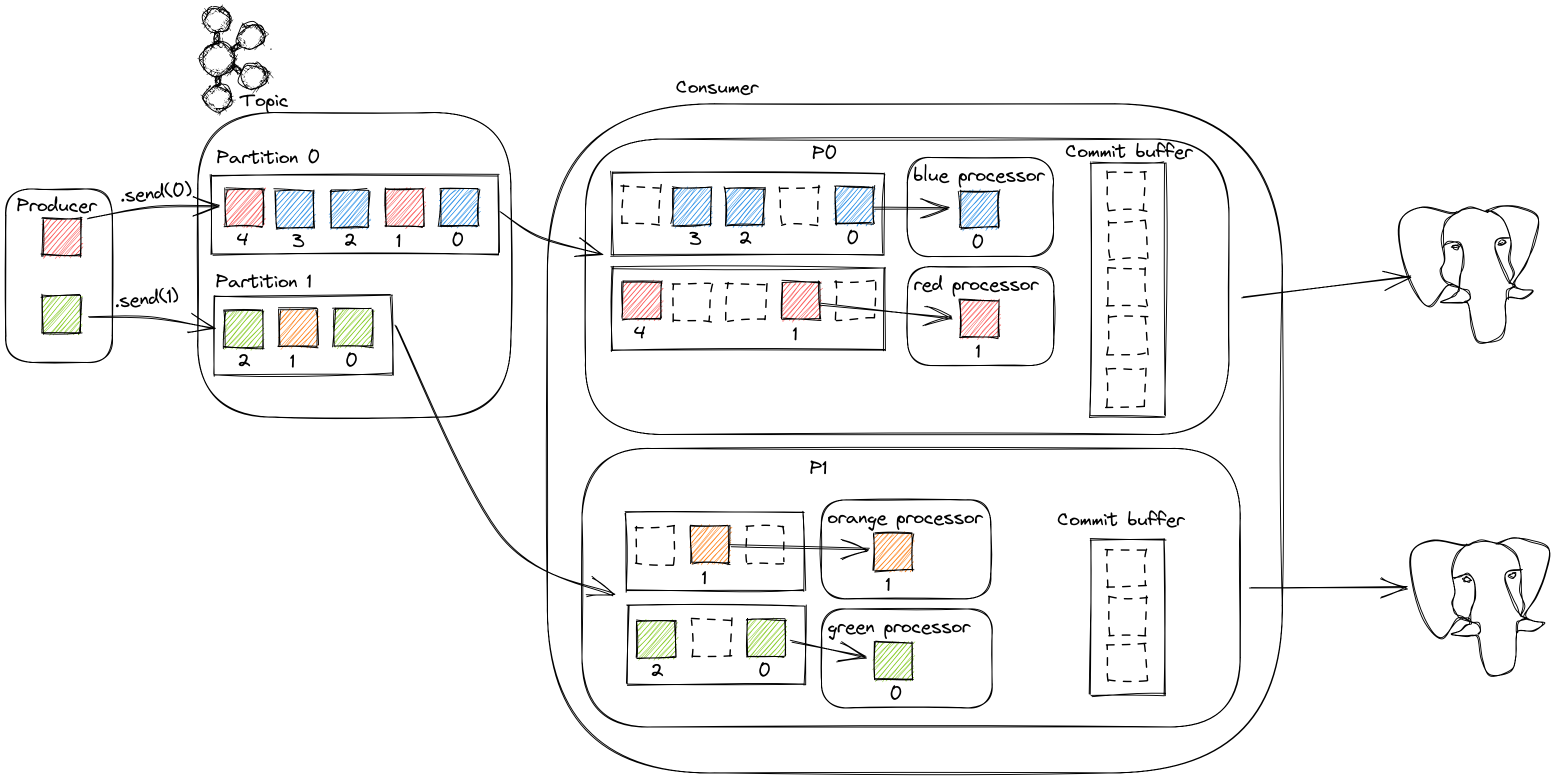
GOALS

- maximize concurrency
- minimize contention
- no concurrent updates to single row

SUBPARTITION QUEUES



ALL TOGETHER



ANOTHER ISSUE

ANOTHER ISSUE

- In-memory queuing

ANOTHER ISSUE

- In-memory queuing
- Memory overloads

ANOTHER ISSUE

- In-memory queuing
- Memory overloads
- Cap on messages in memory

SUDDENLY

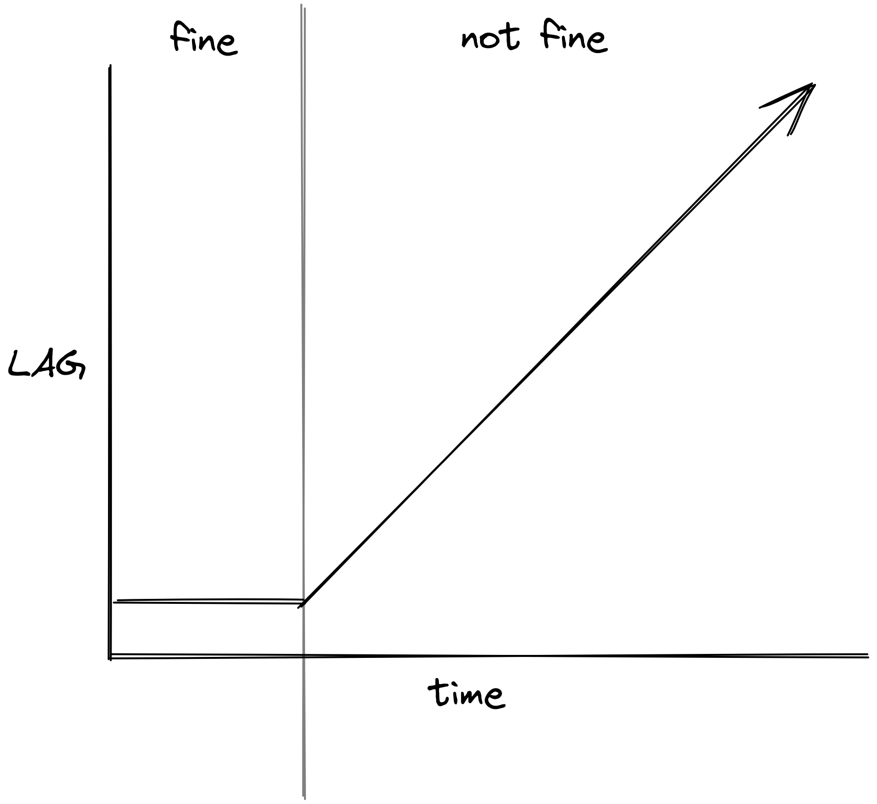
everything was fine

SUDDENLY

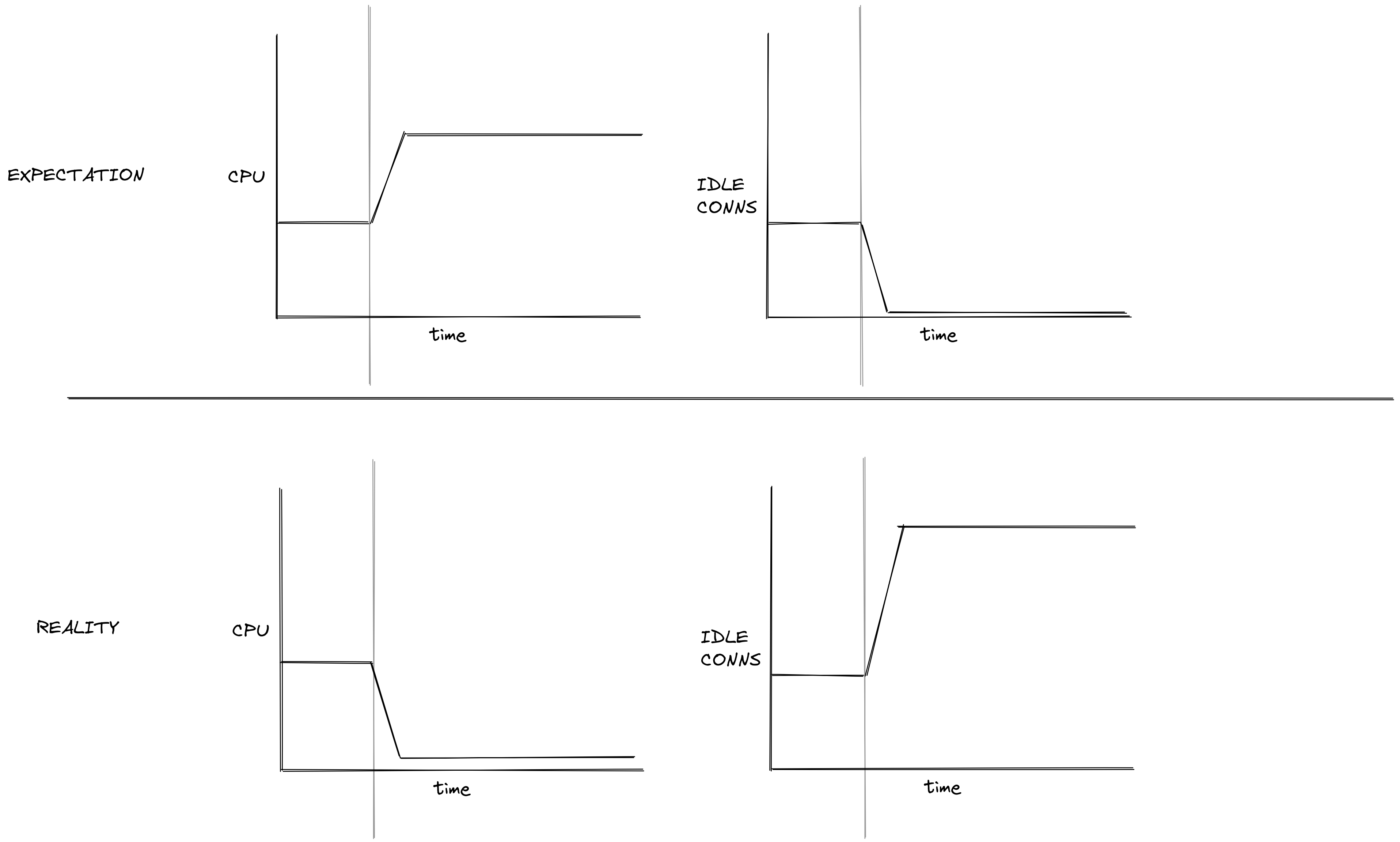
everything was fine

until it wasn't

🚨🚨🚨


fine
not fine
LAG
time

??



EXPECTATION
CPU
time
IDLE CONNS
time
REALITY
CPU
time
IDLE CONNS
time

OBSERVABILITY

OBSERVABILITY

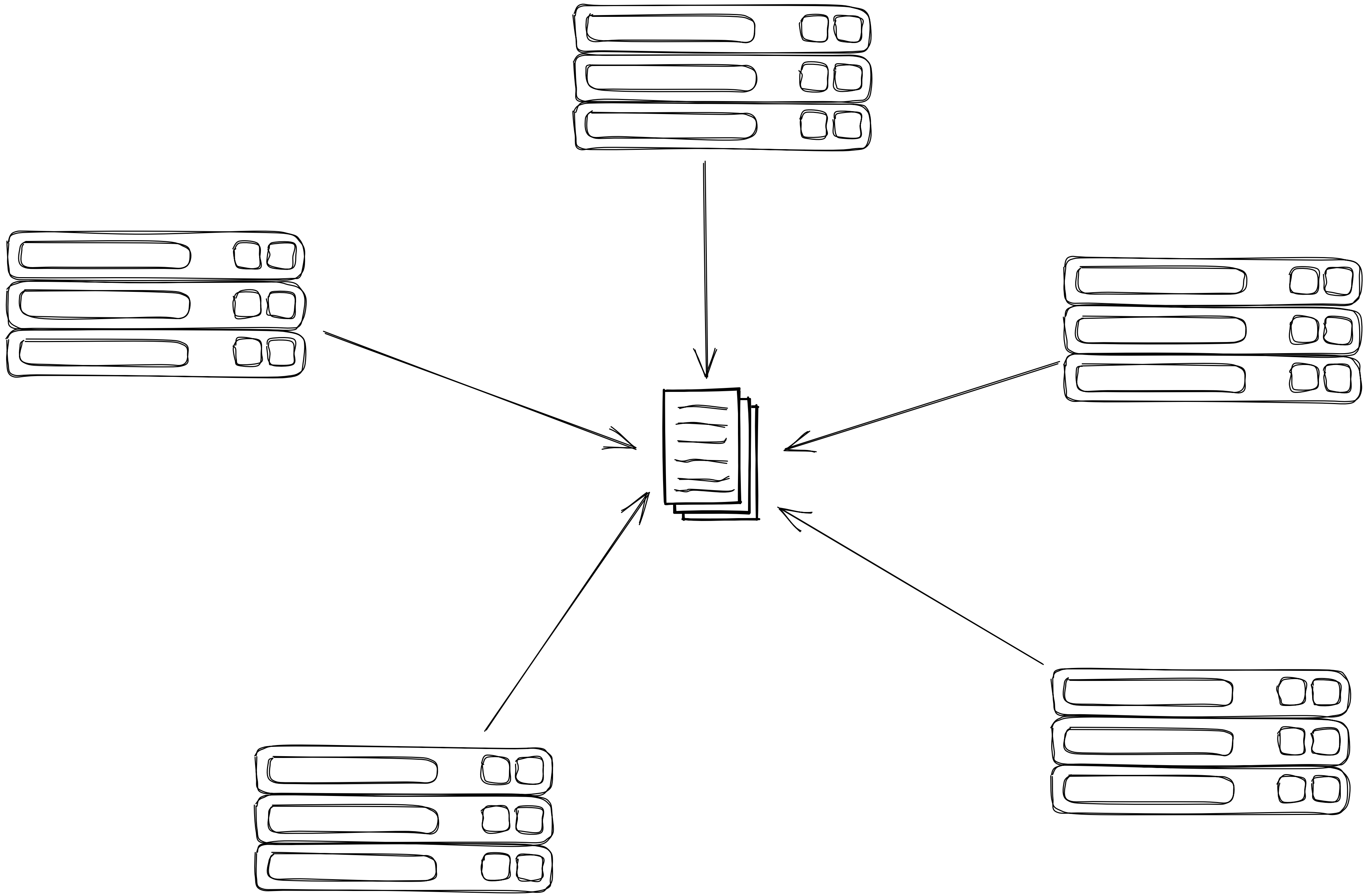
- Only metrics

OBSERVABILITY

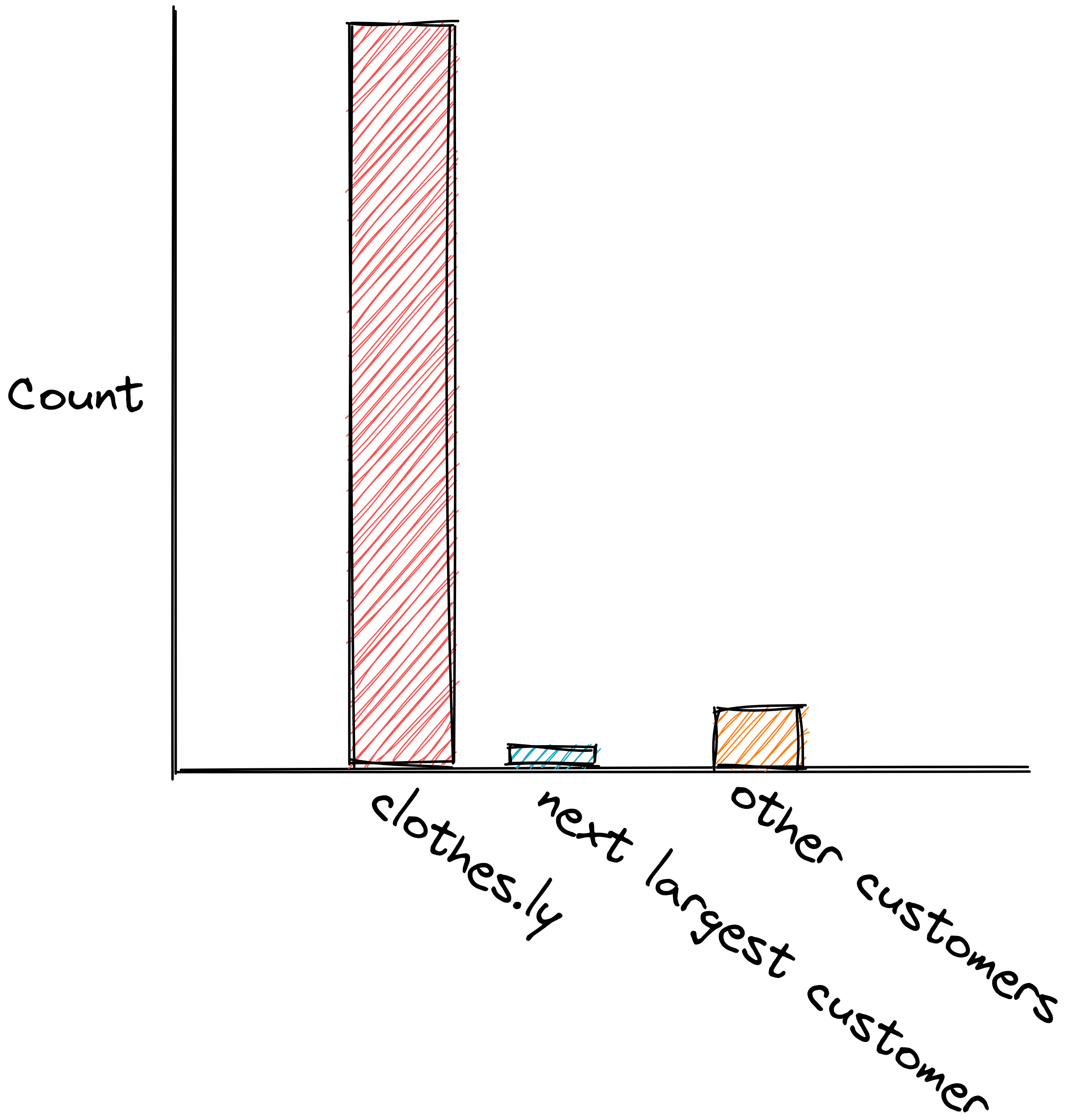
- Only metrics
- Unstructured logs on boxes

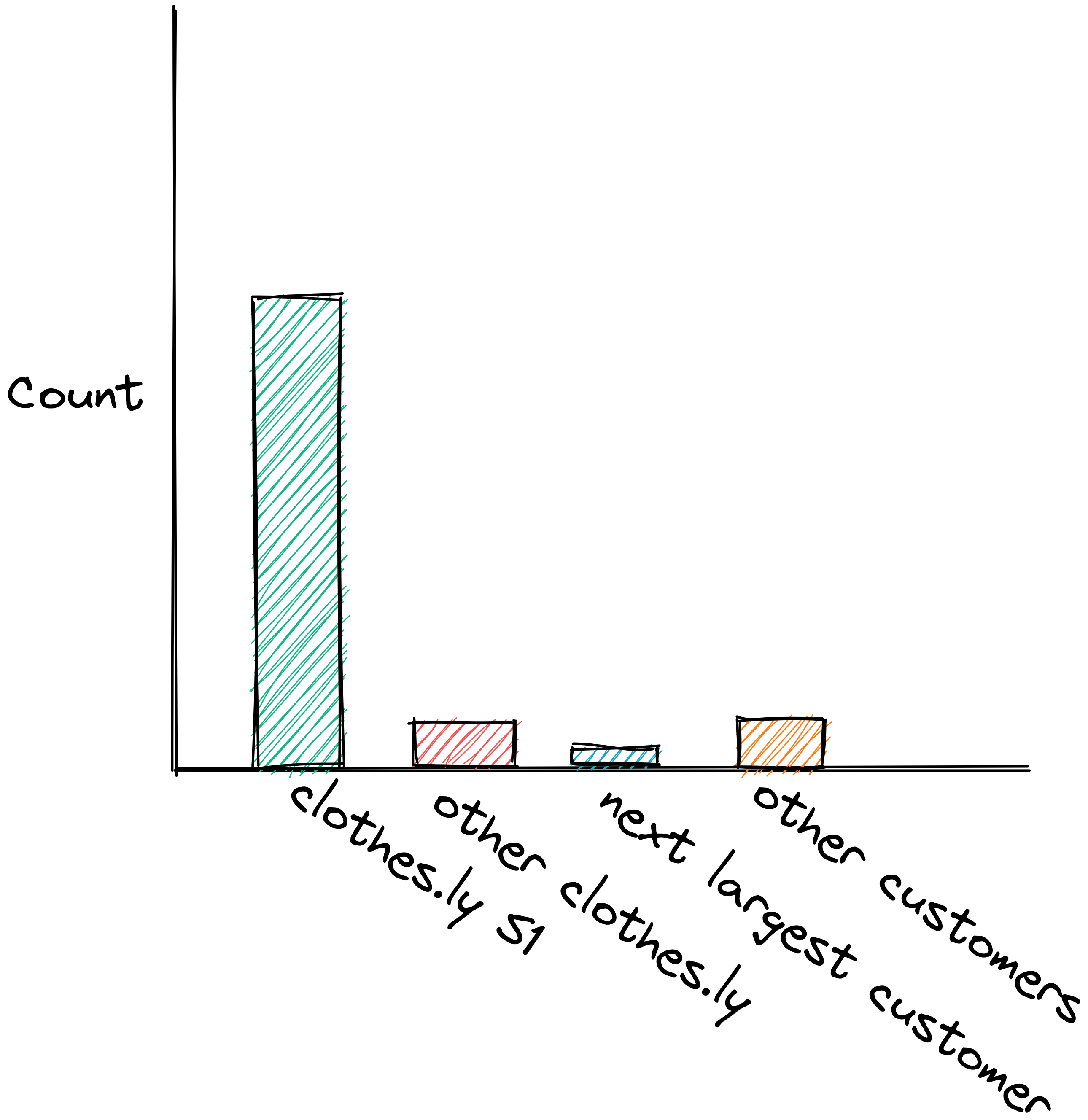
OBSERVABILITY

- Only metrics
- Unstructured logs on boxes
- I insisted on getting centralized logging



```
{
  "app_id": "9...",
  "subscription_id": "6...",
  "sql": "UPDATE ... WHERE id=6...",
  "hostname": "consumer-01"
}
```


Count
clothes.ly
next largest customer
other customers


Count
clothes.ly S1
other clothes.ly
next largest customer
other customers

WHAT?

WHAT?

- Tons of individual updates

WHAT?

- Tons of individual updates
- Incompatible updates

WHAT?

- Tons of individual updates
- Incompatible updates
- Location moving all over

👀

location	color	level	device type	identifier
Chicago	Red	VIP	email	admin@clothes.ly
NYC	Blue	User	email	admin@clothes.ly
Tokyo	Pink	Anon	email	admin@clothes.ly

ONESIGNAL

ONESIGNAL

- More than just push

ONESIGNAL

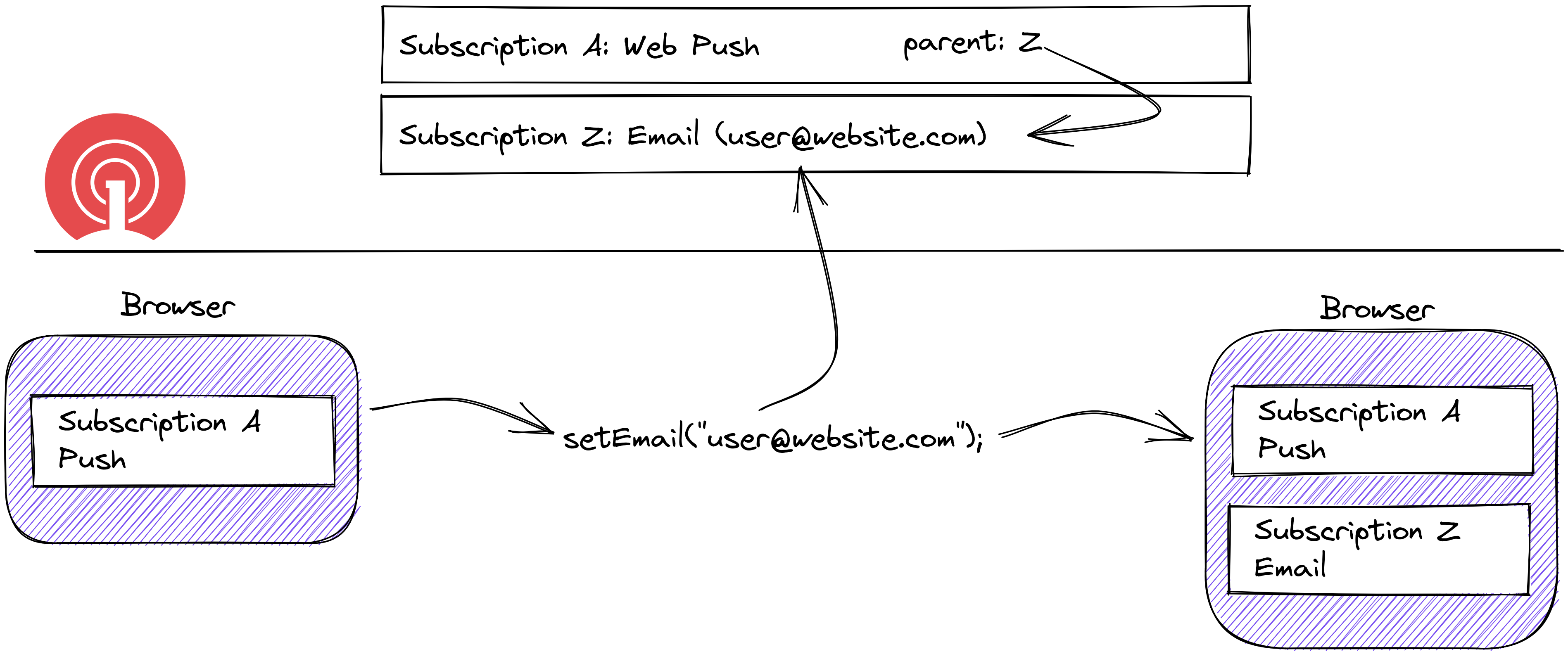
- More than just push
- Omnichannel messaging

ONESIGNAL

- More than just push
- Omnichannel messaging
- Push, email, sms, in-app

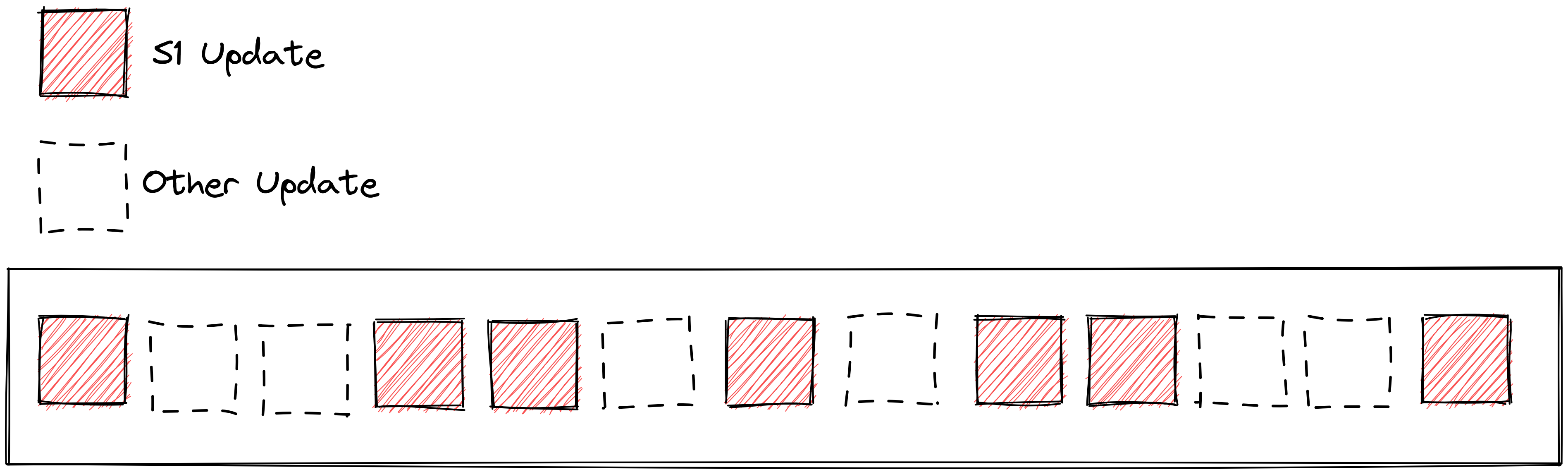
`setEmail`

setEmail

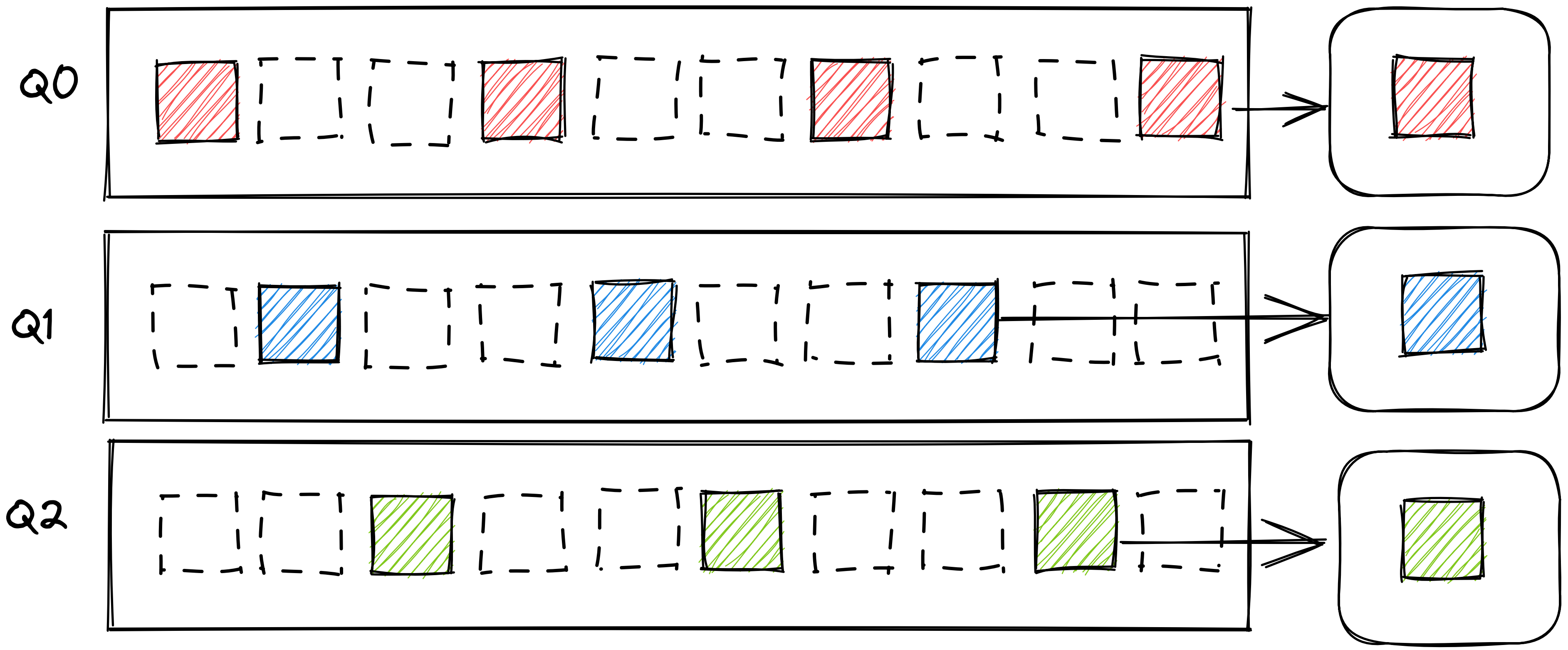

Subscription A: Web Push
parent: Z
Subscription Z: Email (user@website.com)
Browser
Subscription A
Push
setEmail("user@website.com");
Browser
Subscription A
Push
Subscription Z
Email

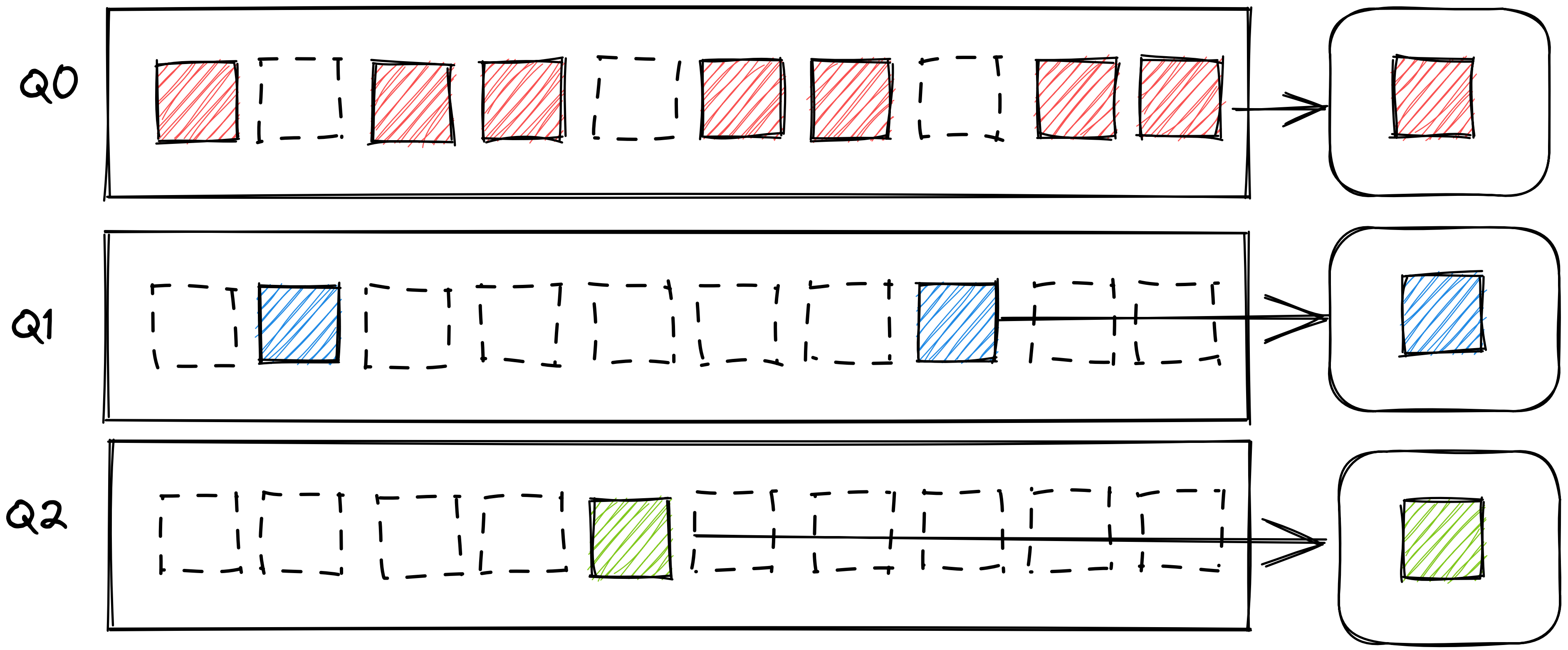
```
COUNT(*) ...
5,000,000

COUNT(*) ... WHERE parent_player_id=S1
4,800,000
```


S1 Update
Other Update

WHY IS THAT A PROBLEM?


Q0
Q1
Q2


Q0
Q1
Q2

OK BUT IN REALITY
IT WAS WORSE

WHAT DID WE DO?

WHAT DID WE DO?

- Skip the updates

WHAT DID WE DO?

- Skip the updates
- Fix message limiting

WHAT DID WE DO?

- Skip the updates
- Fix message limiting
- Limit subscription linking

WHAT DID WE LEARN?

WHAT DID WE LEARN?

- Shift API write workloads to async workers

WHAT DID WE LEARN?

- Shift API write workloads to async workers
- Benefits of subpartition queueing

WHAT DID WE LEARN?

- Shift API write workloads to async workers
- Benefits of subpartition queueing
- Struggles of subpartition queuing

WHAT DID WE LEARN?

- Shift API write workloads to async workers
- Benefits of subpartition queueing
- Struggles of subpartition queuing
- Centralized observability

WHAT DID WE LEARN?

- Shift API write workloads to async workers
- Benefits of subpartition queueing
- Struggles of subpartition queuing
- Centralized observability
- Customers are so creative