YugaByteDB

+

Jepsen

Kyle Kingsbury

@aphyr

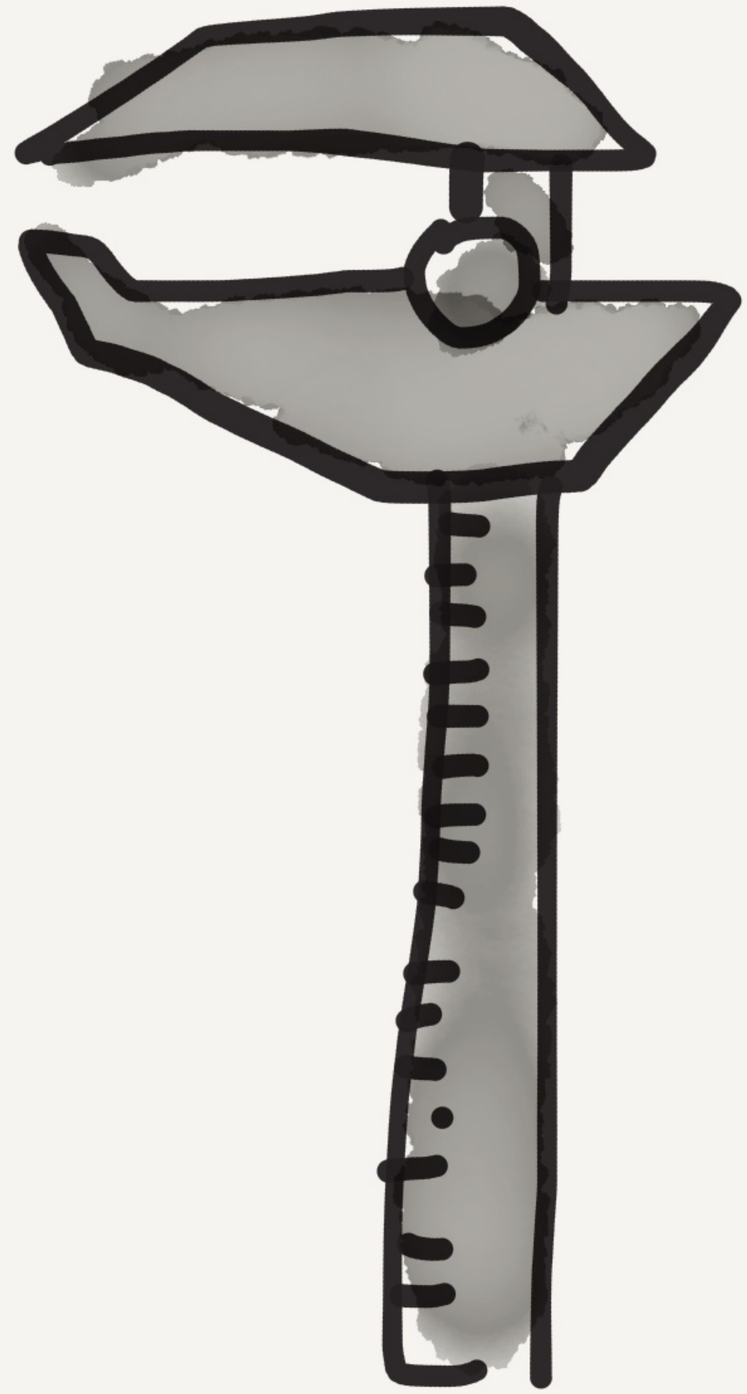
I break databases!

1. How we tested

2. What we found

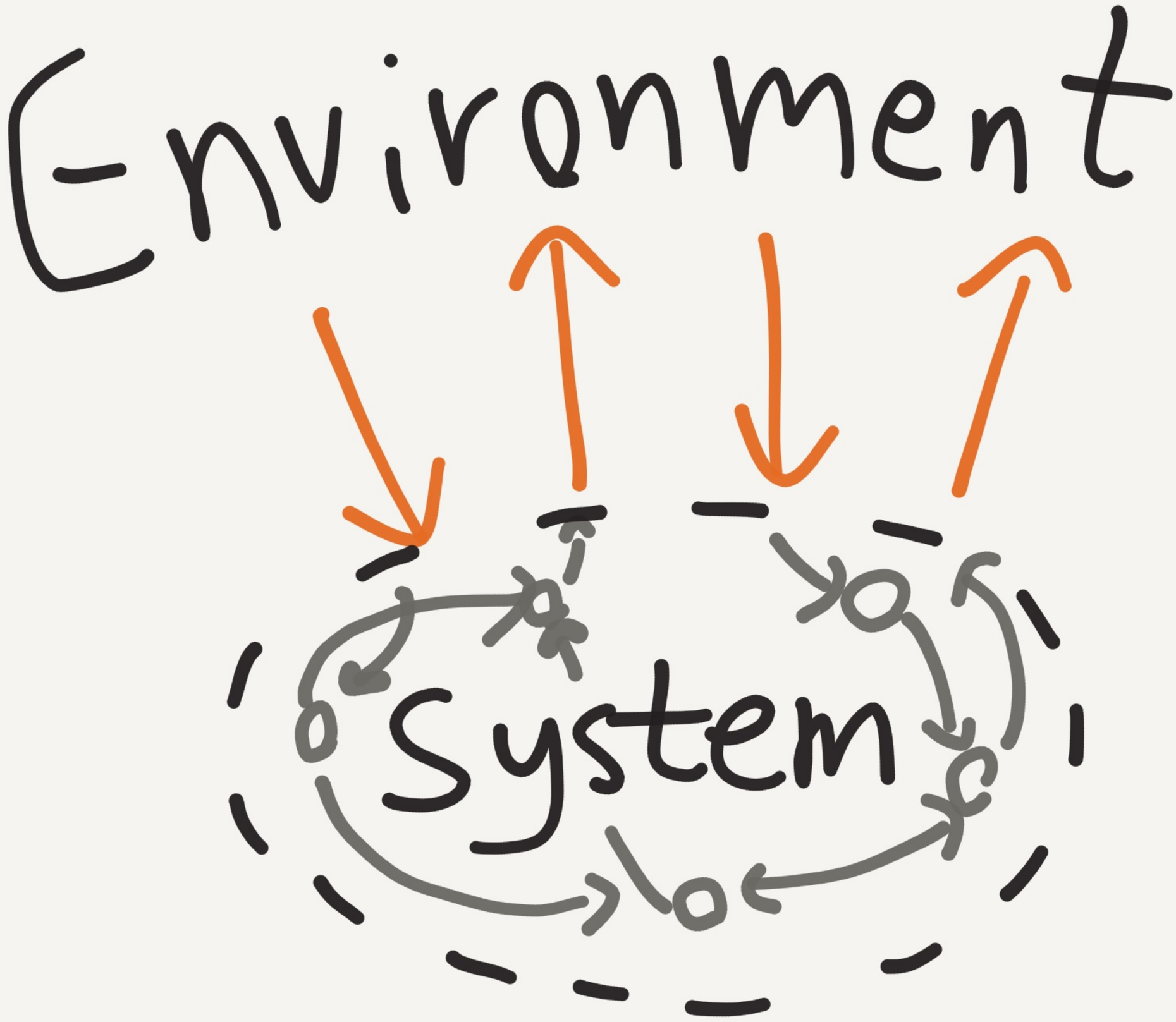
3. Where Yugabyte is going

Measure your Systems

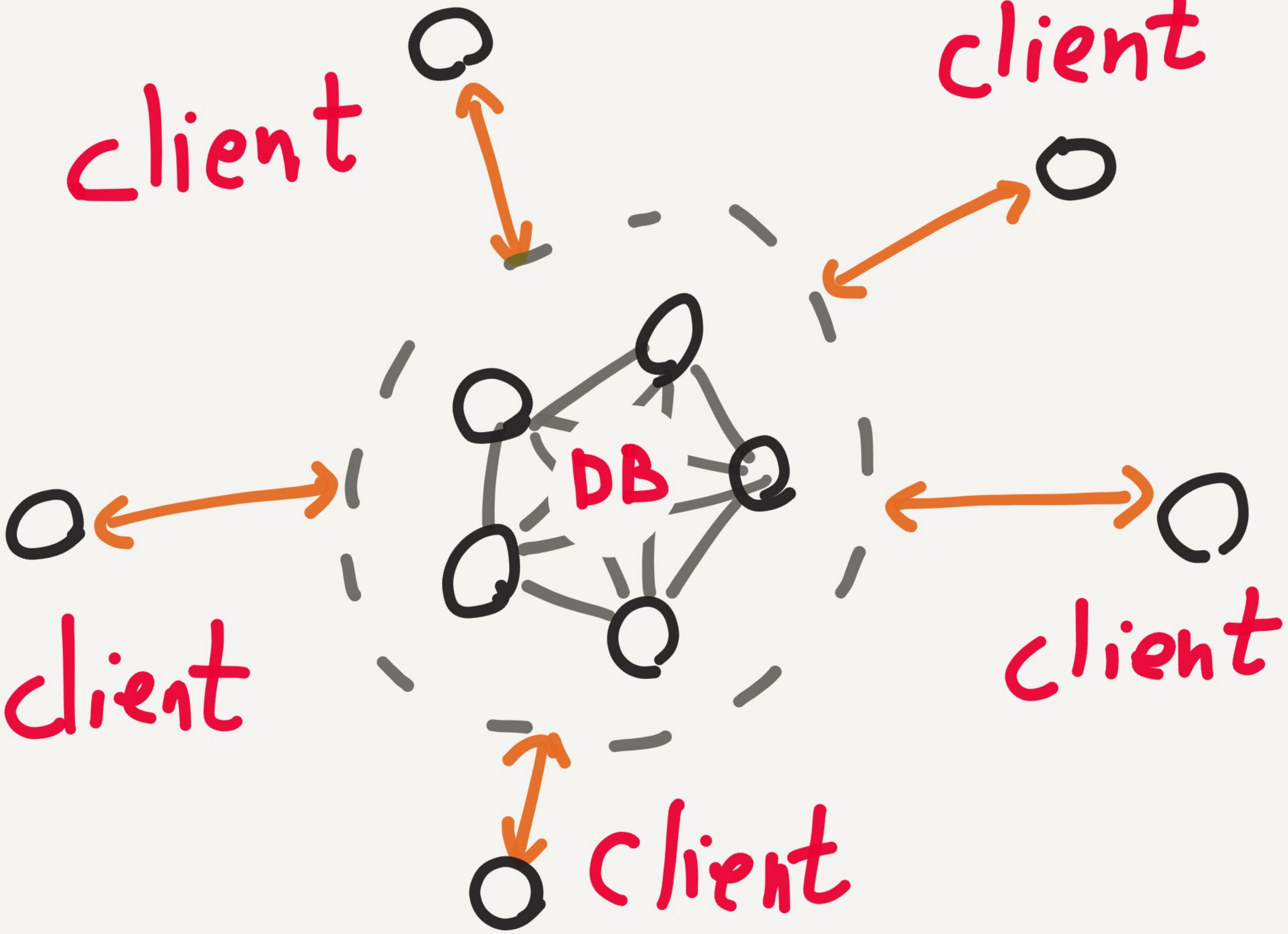


Jepsen

github.com/aphyr/jepsen


Environment
System

-INVARIANTS-

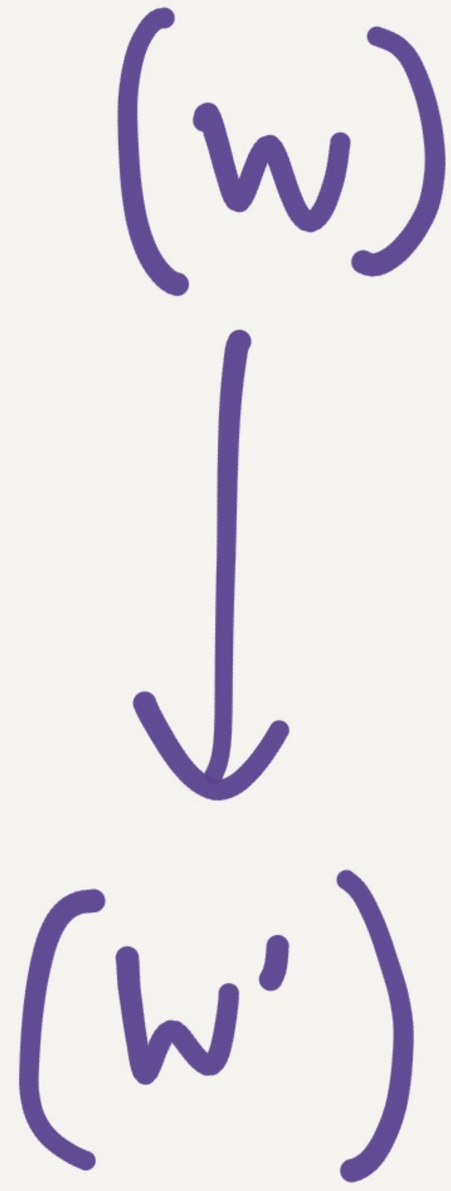

client
client
DB
client
client
client

client: w—w' r—r' w—w'

DB:

client: w —— w' w —— - - -

Clients Generate random operations (w) and apply them to the system



invoke ok

invoke fail

invoke ? ? info ? ? ?

invoke ok

invoke fail

invoke ok

invoke info

invoke ok

invoke fail

invoke ok

invoke info

invoke ok

invoke fail

invoke ok

invoke info

invoke ok

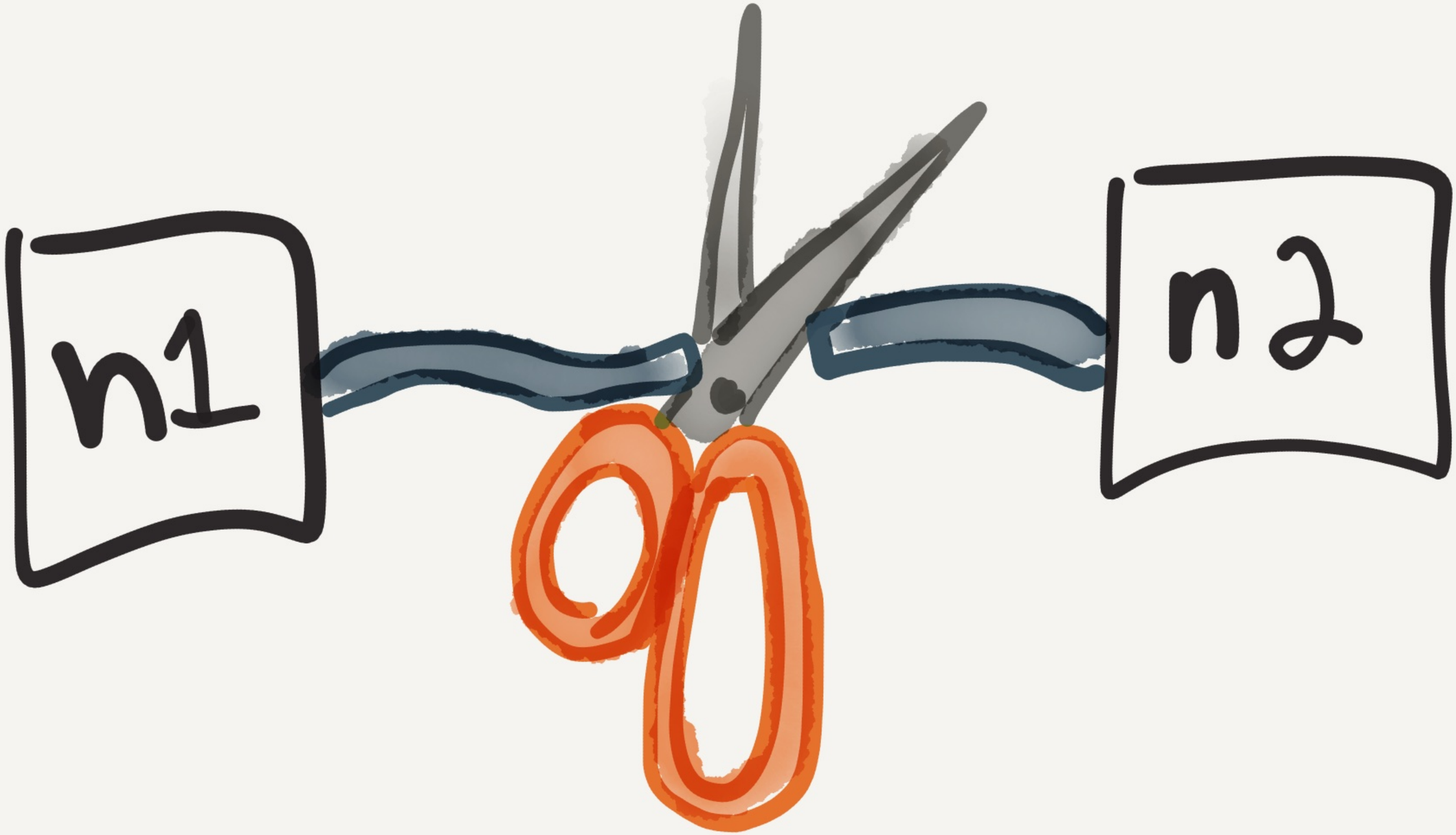
invoke fail

invoke ok

invoke ok

invoke info

1. Generate ops
2. Record history
3. Verify history is consistent w/ model



Partitions!

YugaByte

1.1.9

CQL/SQL

Ready now

In beta

"Consistency"

"Availability"

"Performance"

"Consistency"

"Availability"

"Performance"

"Transactional NoSQL"

"Consistency"

"Availability"

"Performance"

"Transactional NoSQL"

"Full spectrum of ACID Compliance"

"Consistency"

"Availability"

"Performance"

"Transactional NoSQL"

"Full spectrum of ACID Compliance"

"0 data loss"

"Fully compatible with PostgresQL"

"Fully compatible with PostgresQL"

"Distributed ACID"

"Fully compatible with PostgresQL"

"Distributed ACID"

"Strongly consistent replication"

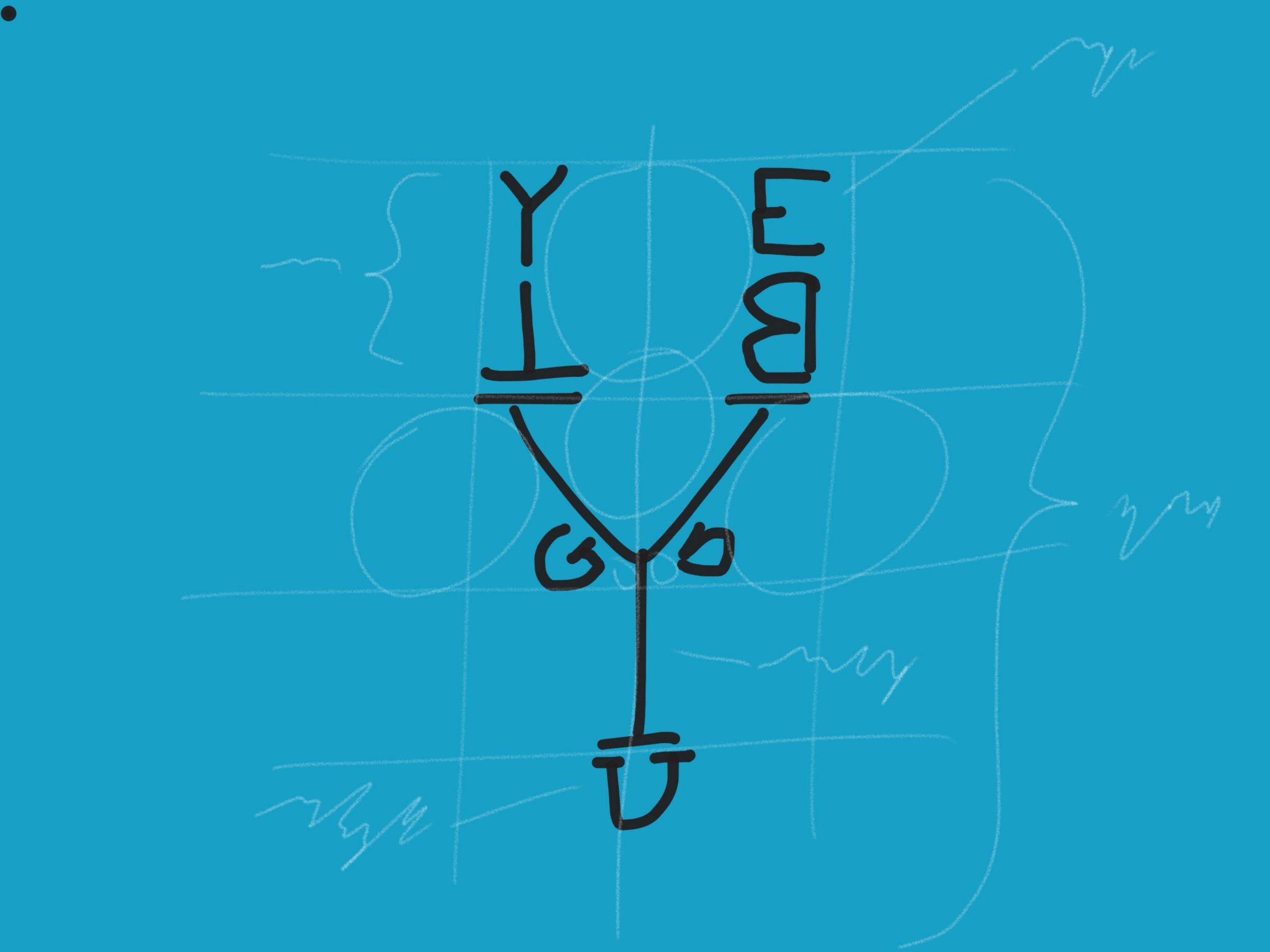
"Fully compatible with PostgresQL"

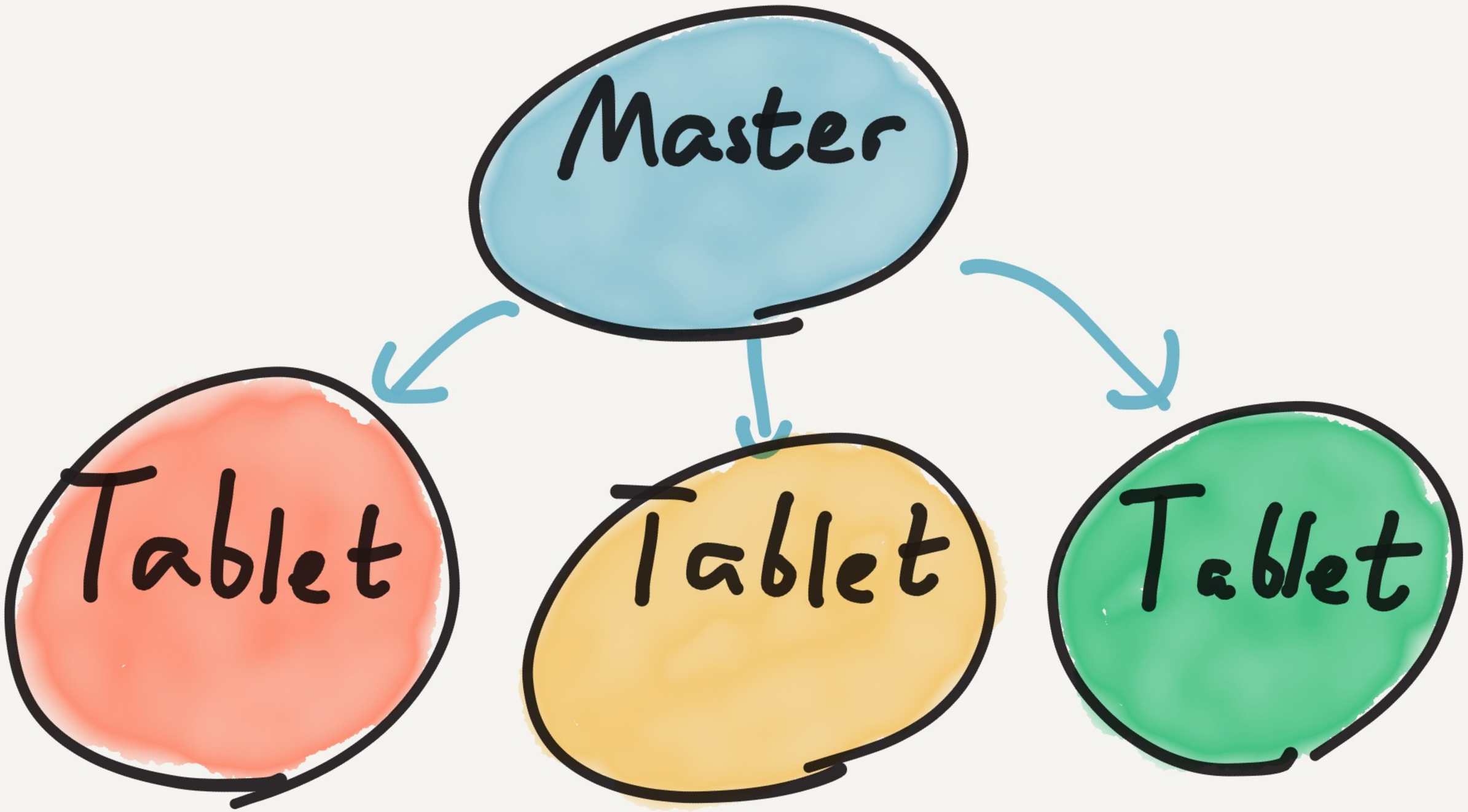
"Distributed ACID"

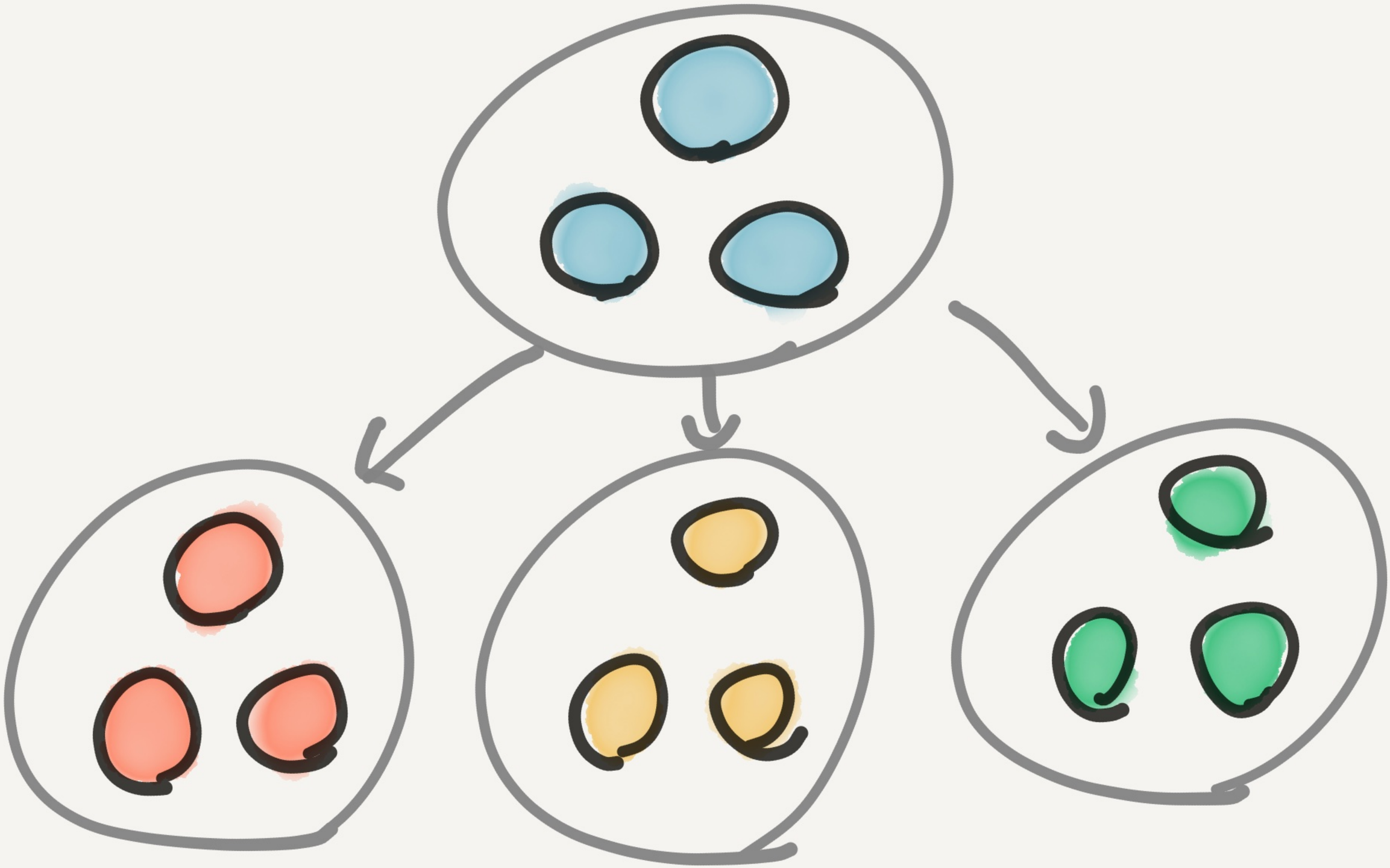
"Strongly consistent replication"

"Jepsen test suite verified"

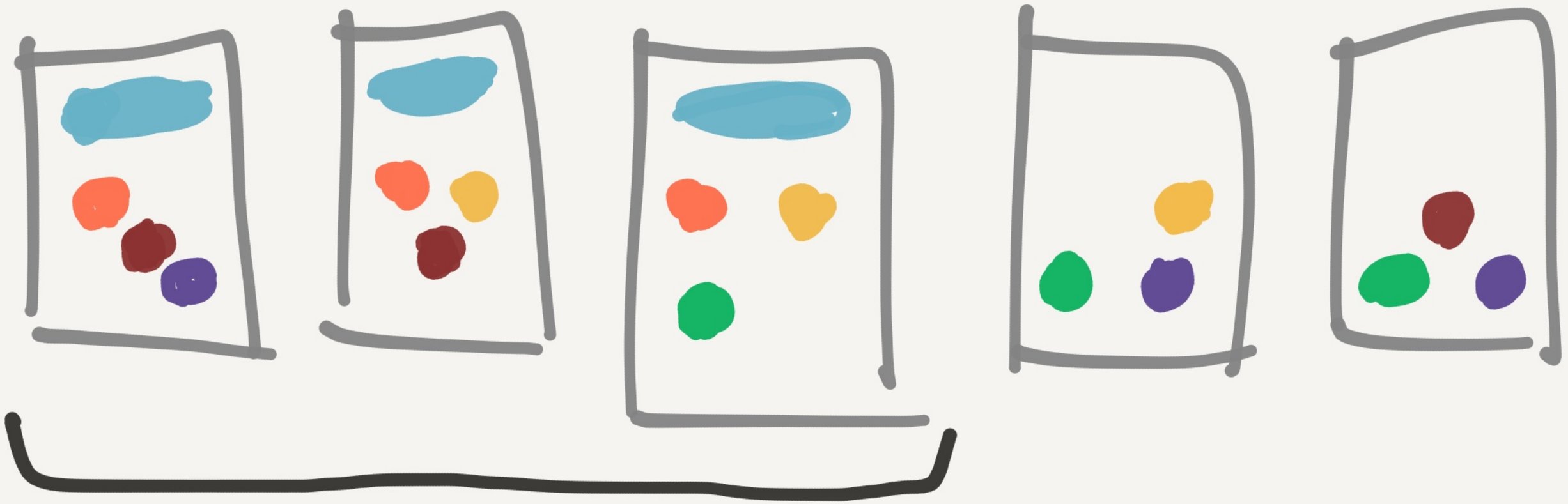
- Snapshot Isolated CQL
- Serializable $ SQL WIP
- Linearizable per-key
- Linearizable indices
- Requires good clocks




Master
Tablet
Tablet
Tablet



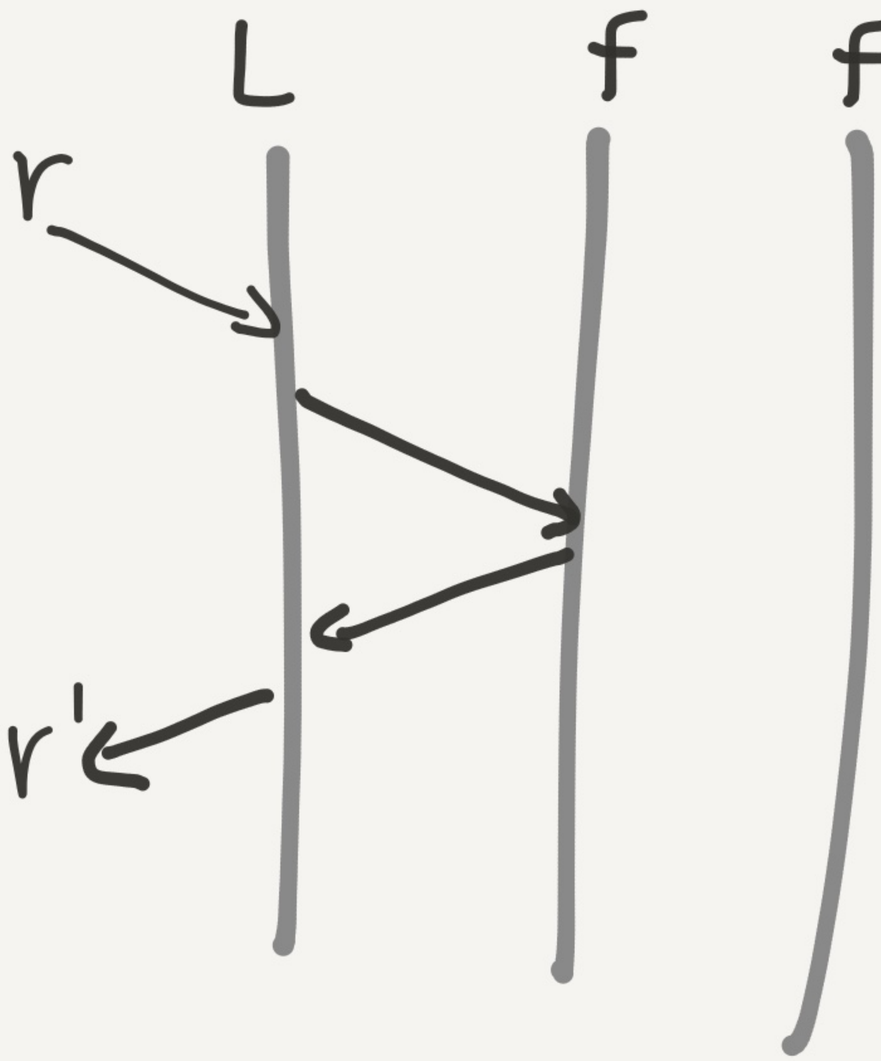
Q=Raft



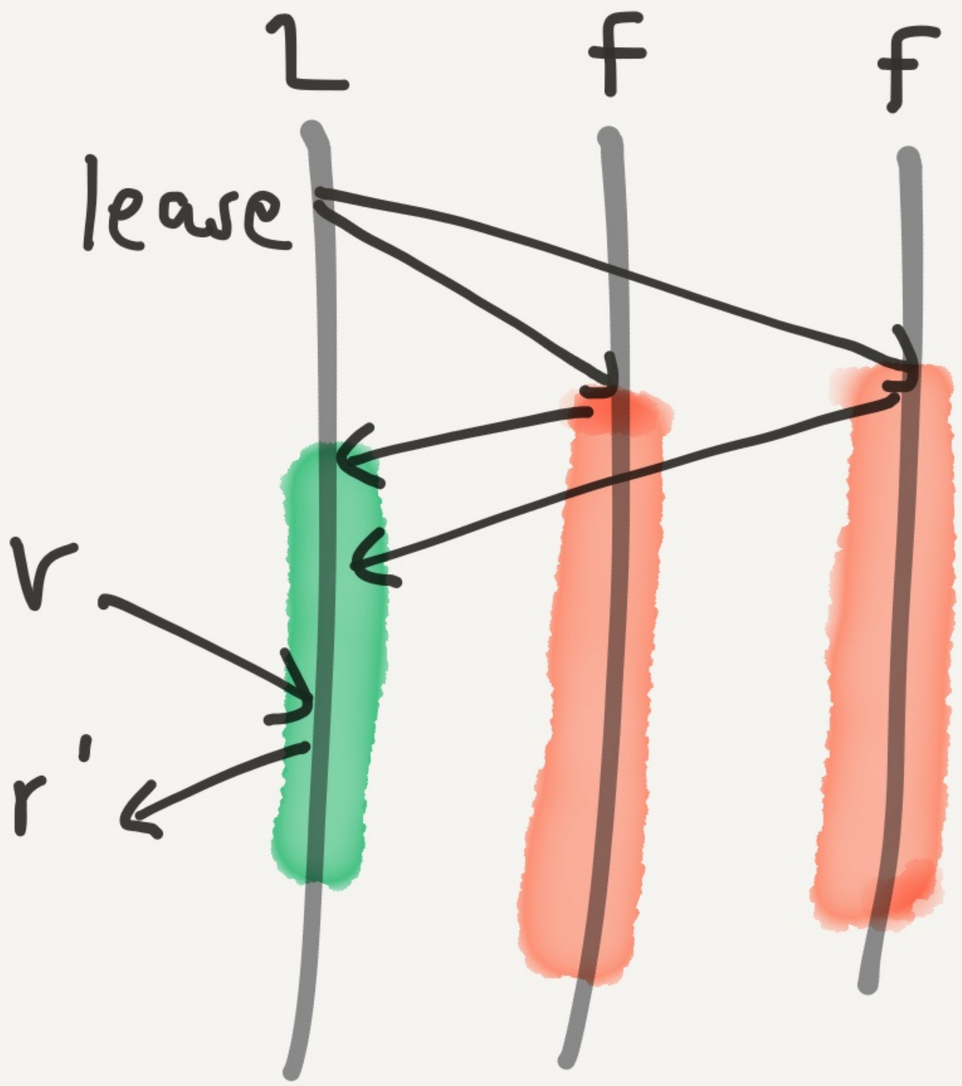
Masters

Tablet Servers

Leader Leases



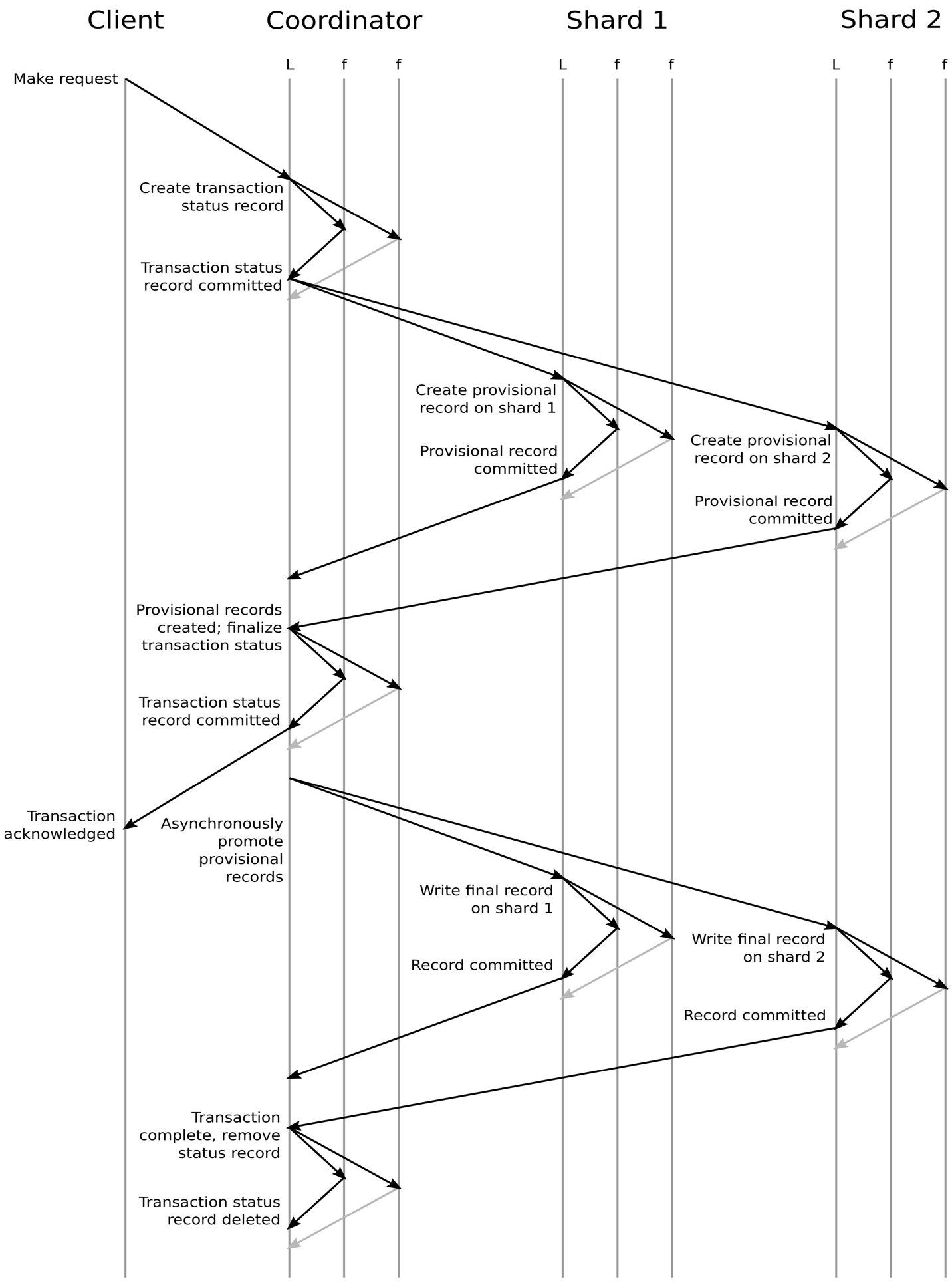
Raft



Raft w/ Leases

Single-shard txns go through Raft...

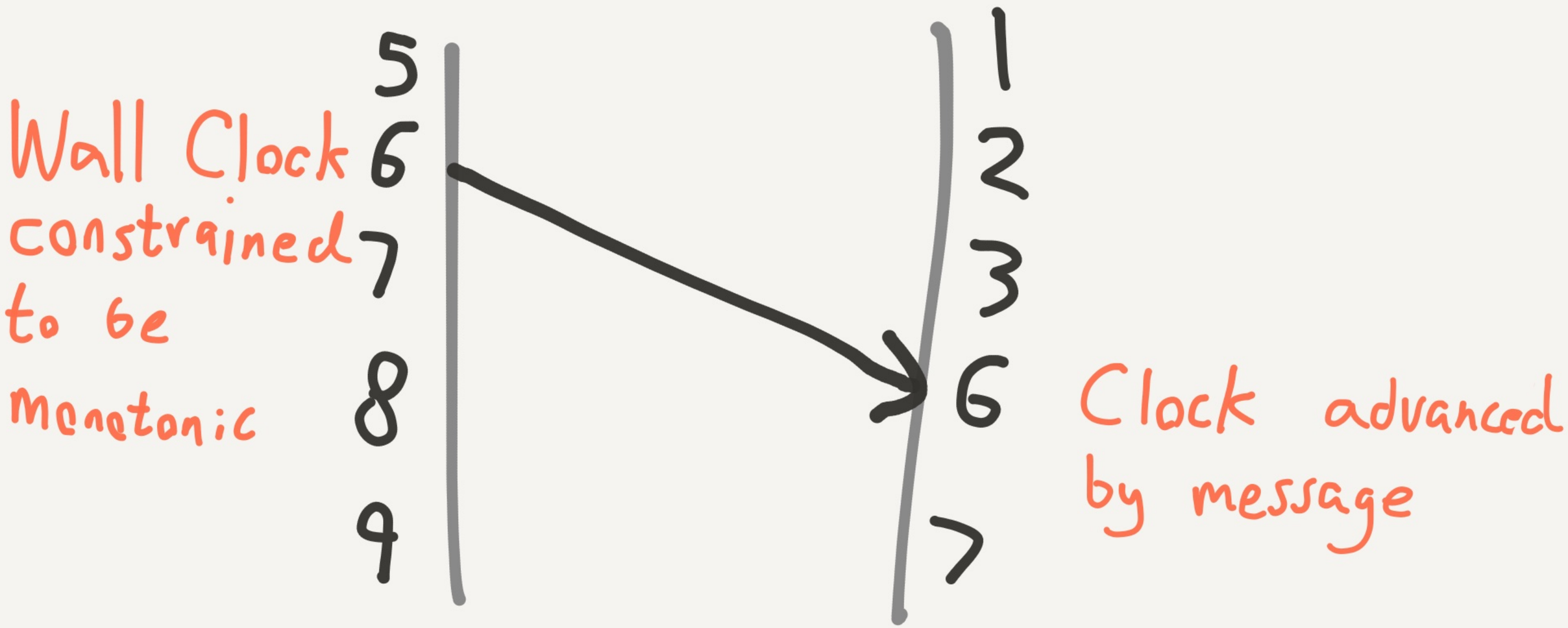
Multi-shard txns?

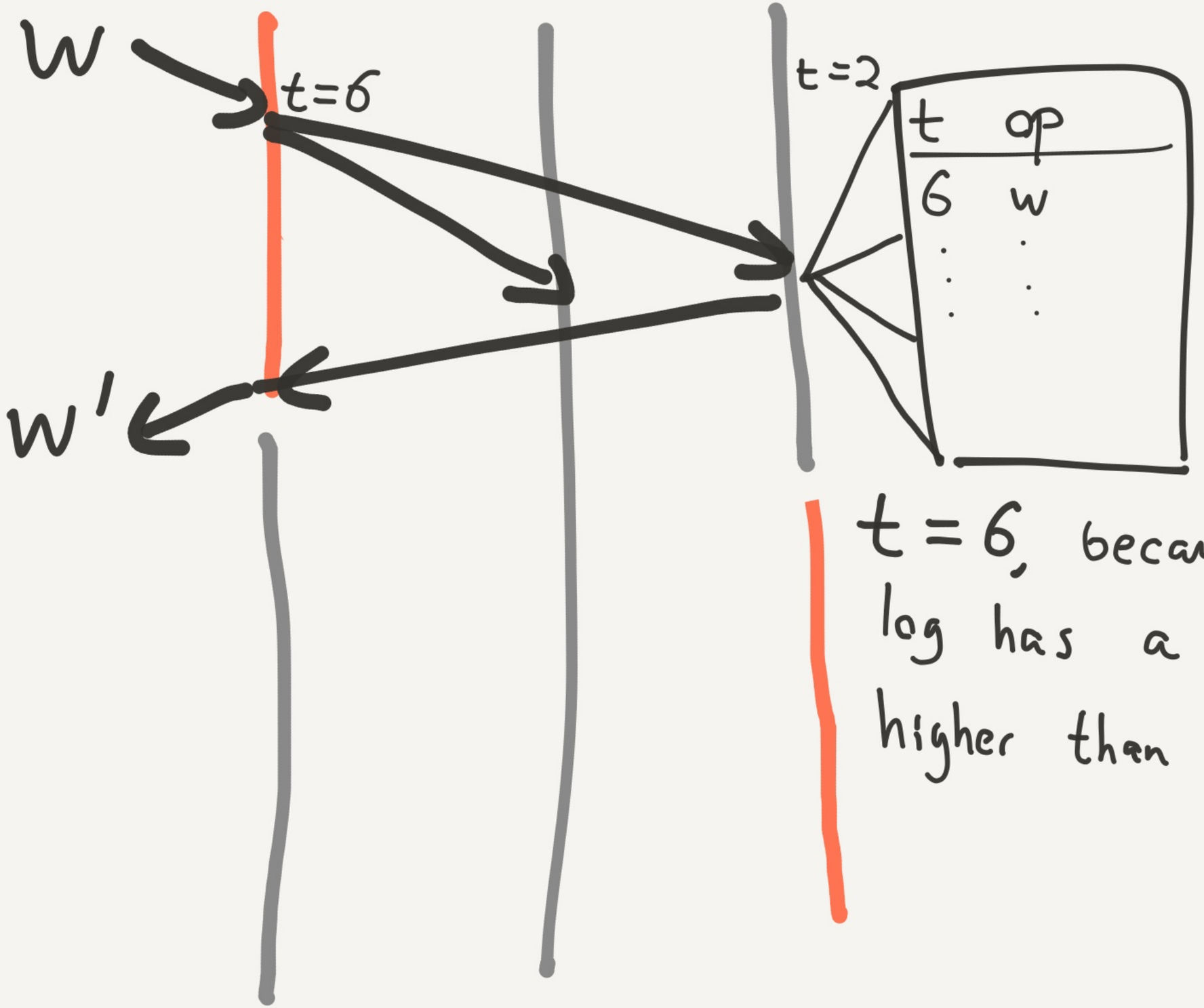

Client
Coordinator
Shard 1
Shard 2
L
f
f
L
f
f
L
f
f
Make request
Create transaction status record
Transaction status record committed
Create provisional record on shard 1
Provisional record committed
Create provisional record on shard 2
Provisional record committed
Provisional records created; finalize transaction status
Transaction status record committed
Transaction acknowledged
Asynchronously promote provisional records
Write final record on shard 1
Record committed
Write final record on shard 2
Record committed
Transaction complete, remove status record
Transaction status record deleted

Updates: 3 cross-DC round trips

Reads: 1 cross-DC round trip

Hybrid Logical Clocks





$t = 6$, because our log has a time higher than our clock!

YugaByte wrote their own tests!

- [ ] Counter
- [ ] Set
- [ ] Long Fork
- [ ] Bank
- [ ] Single-key linearizability
- [ ] Multi-key linearizability

"transactions"

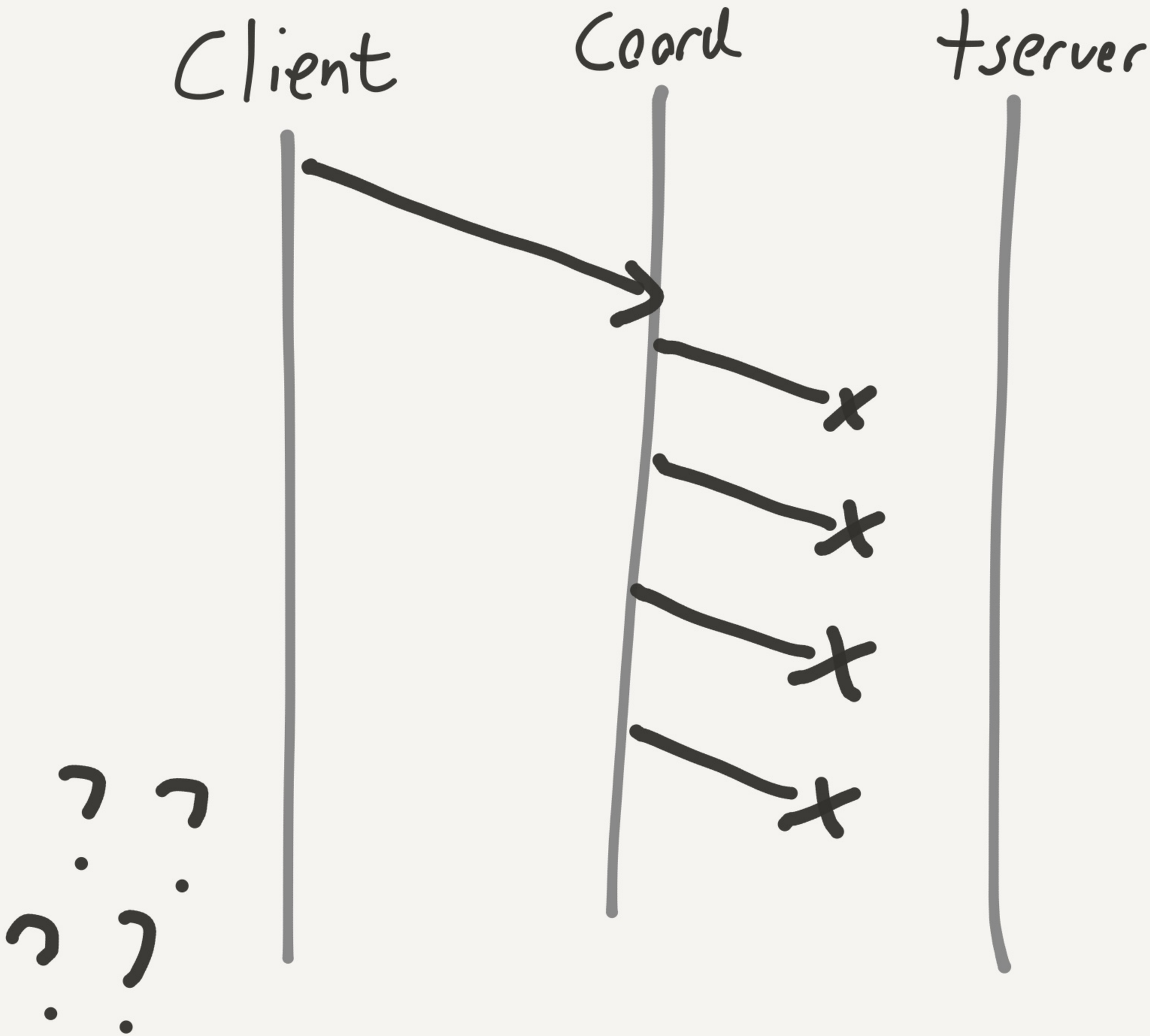
1. Multi key single read query

2. Batch update

- Network Partitions
- Clock skew
- Node crashes
- Node restarts

#822

CQL requests never time out


Client
Coord
tserver

Fixed in 1.1.13

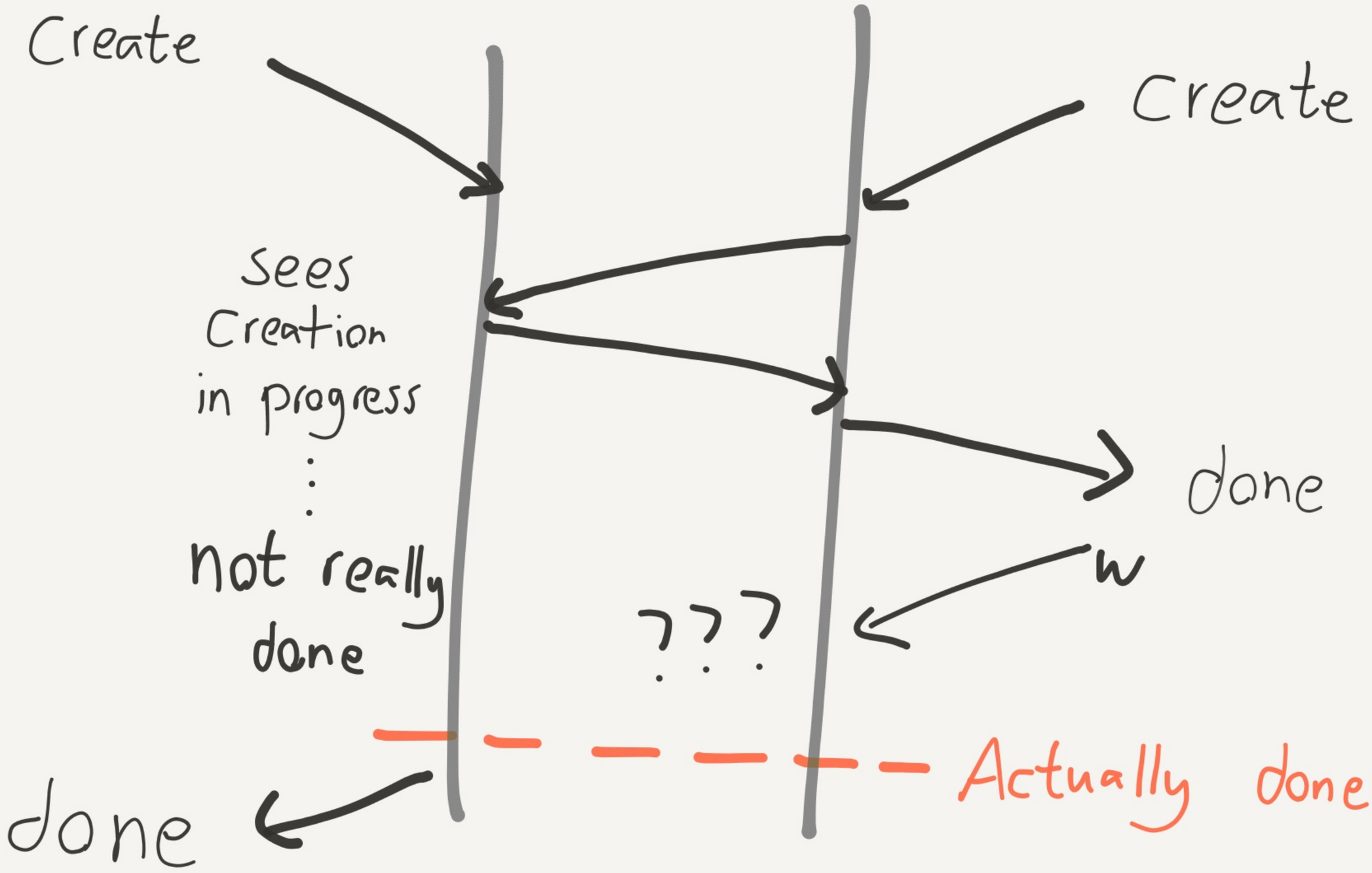
#823 Repeated log messages

Leader Master has changed, re-trying
Leader Master has changed, re-trying
Leader Master has changed, re-trying
Leader Master has changed, re-trying
Leader Master has changed, re-trying
Leader Master has changed, re-trying
Leader Master has changed, re-trying

~4,000/sec

Fixed in 1.1.13

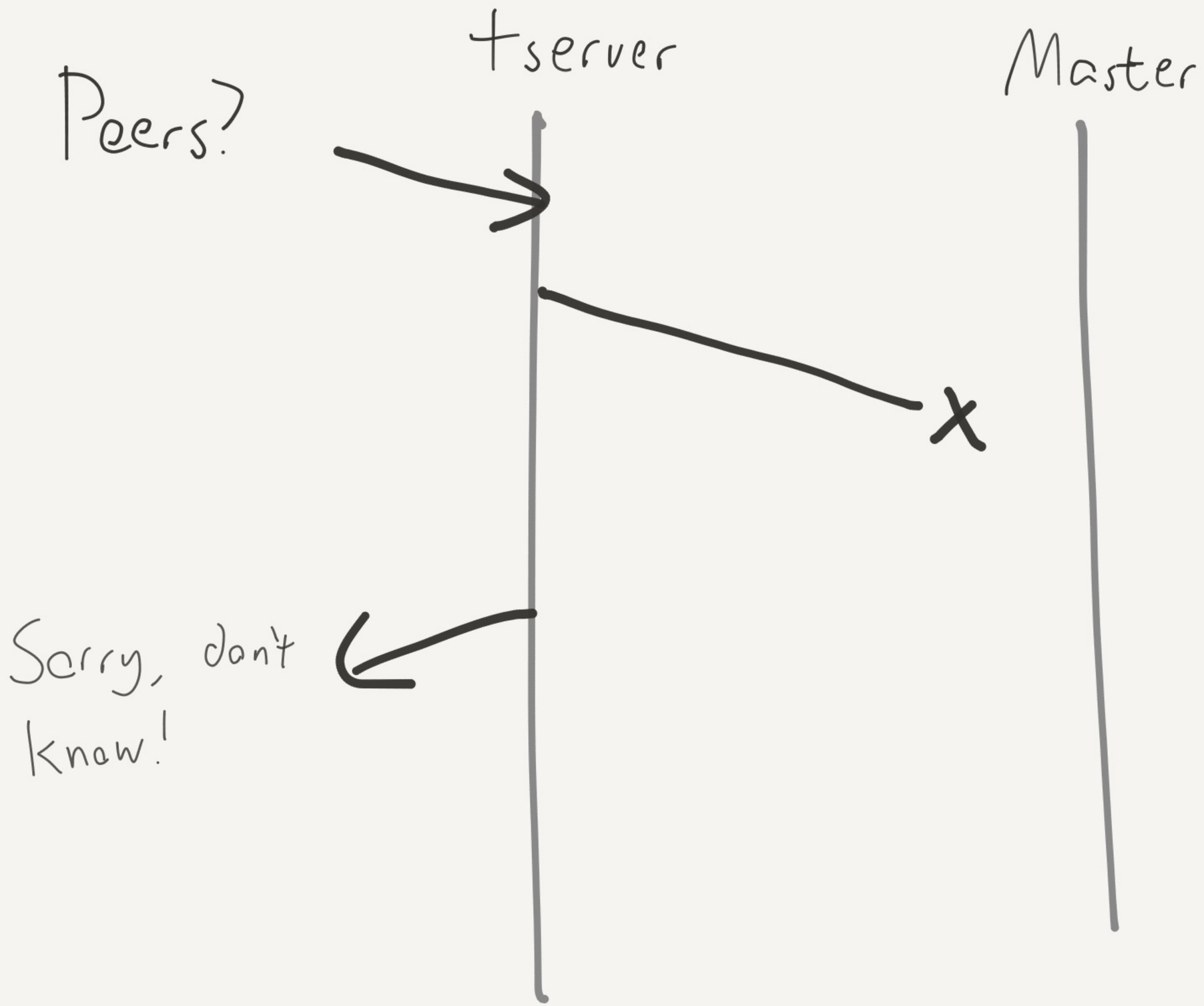
#798 Race condition in CREATE TABLE


Create
Create
Sees
Creation
in progress
⋮
not really
done
???
done
done
Actually done

Fixed in 1.1.15

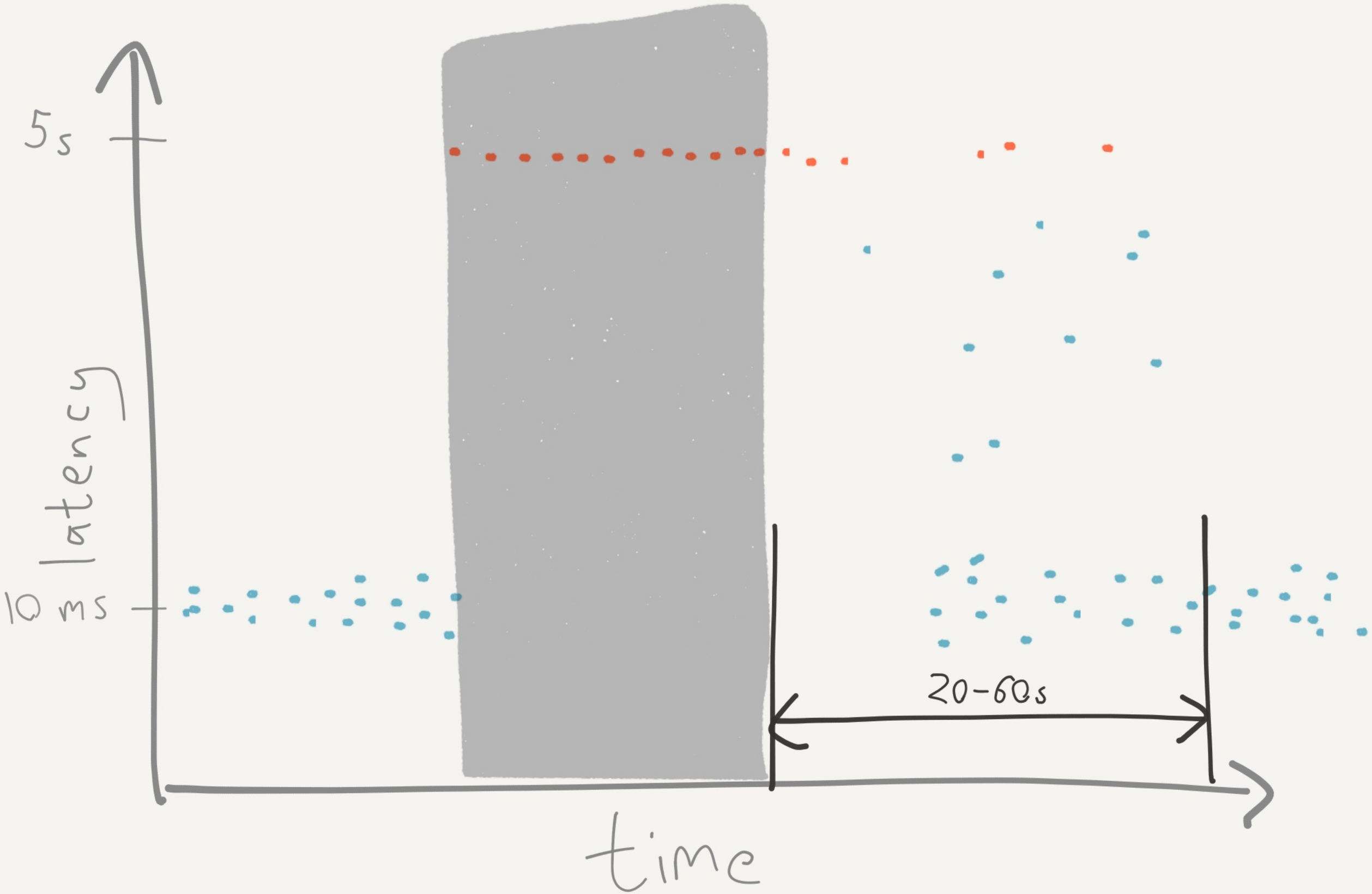
#864

Can't connect to partitioned nodes

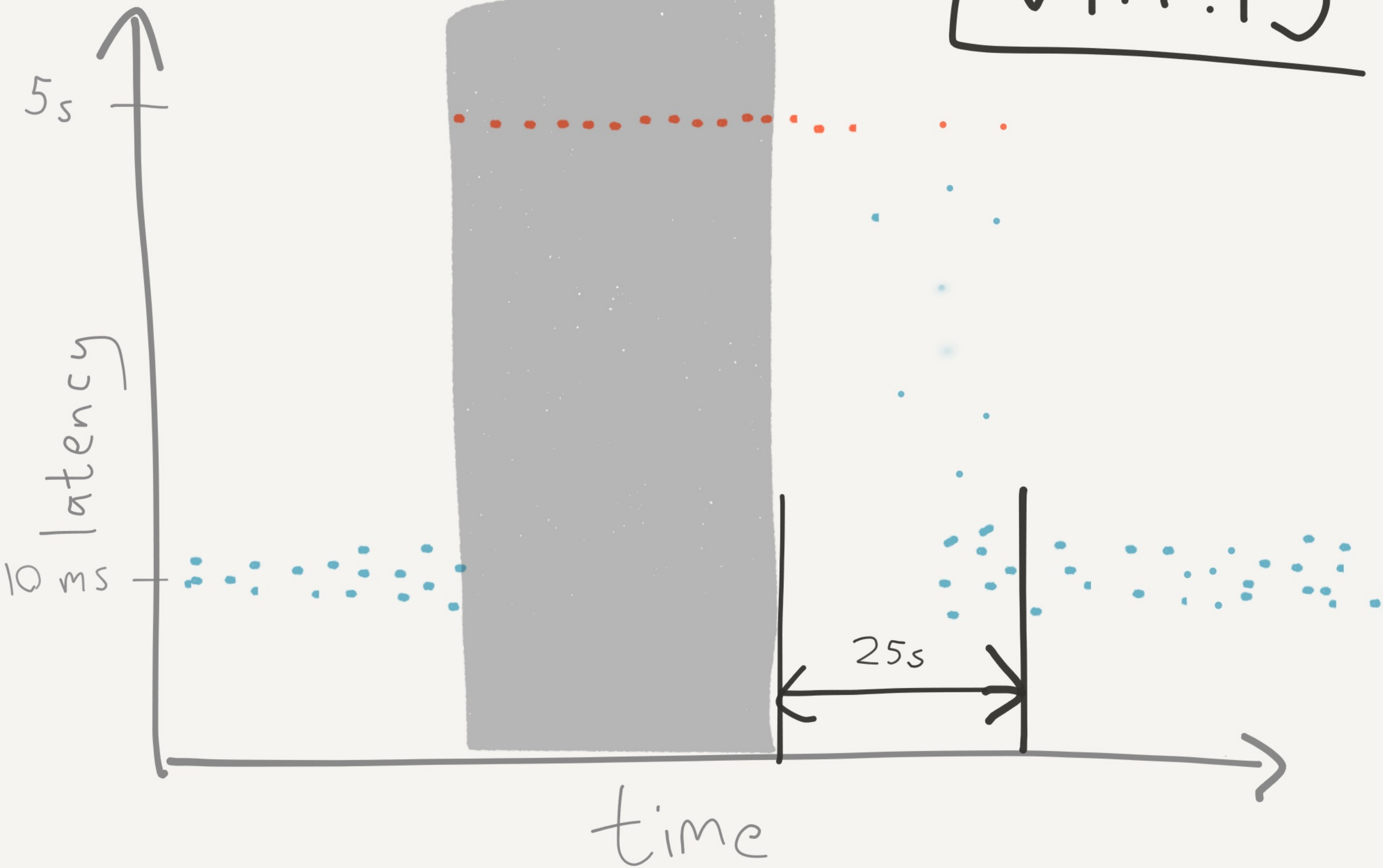

Peers?
tserver
Master
X
Sorry, don't know!

Still open

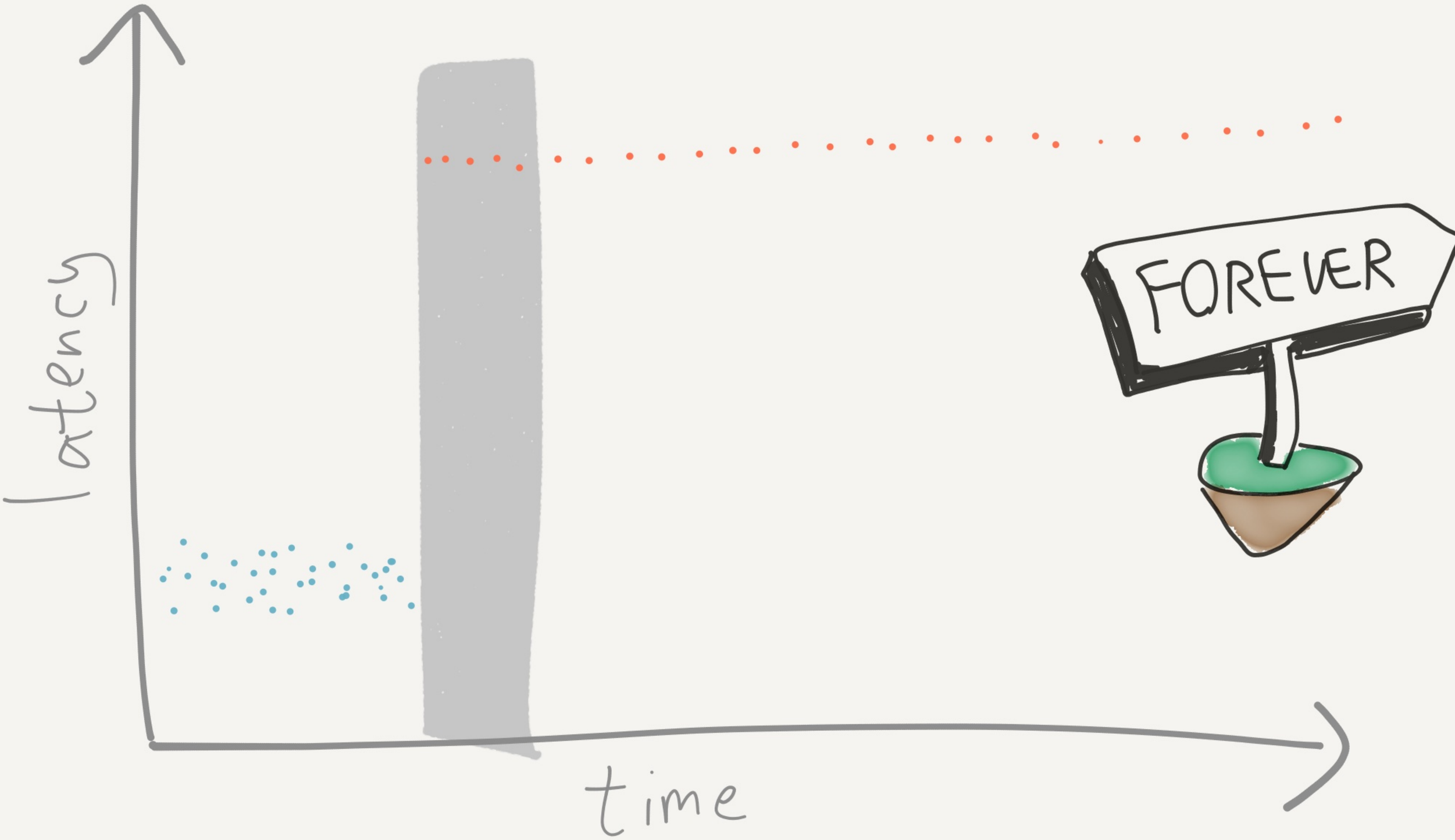
#821 Slow partition recovery


5 s
latency
10 ms
20-60s
time

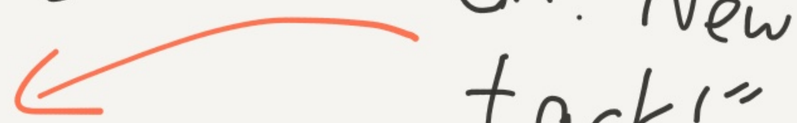
v1.1.15



#886 Master stalls



- [ ] Become leader
- [ ] Abort all running tasks
- [ ] Wait for all tasks



.
.
.
.
.

Fixed in 1.1.15

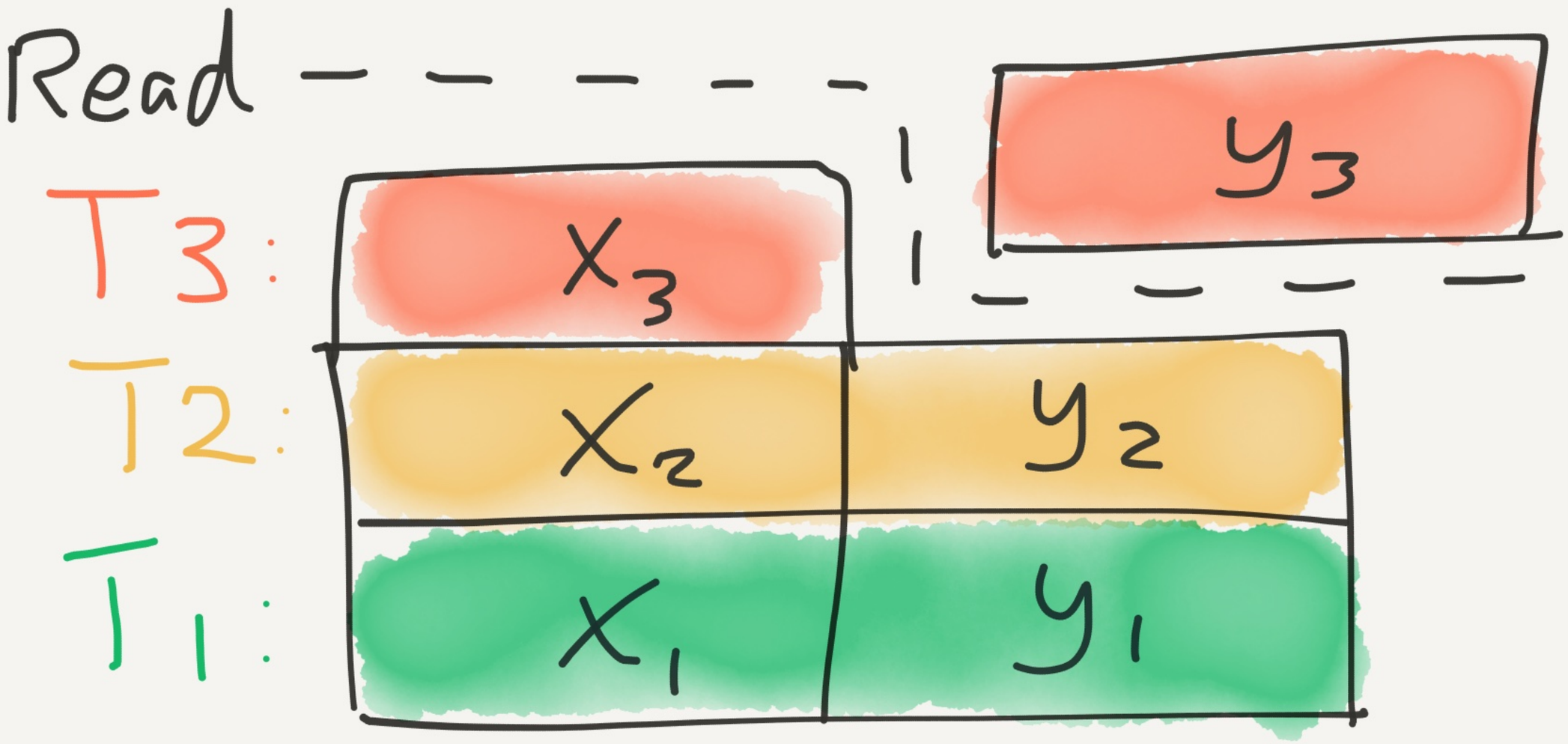
#862 Memory leaks

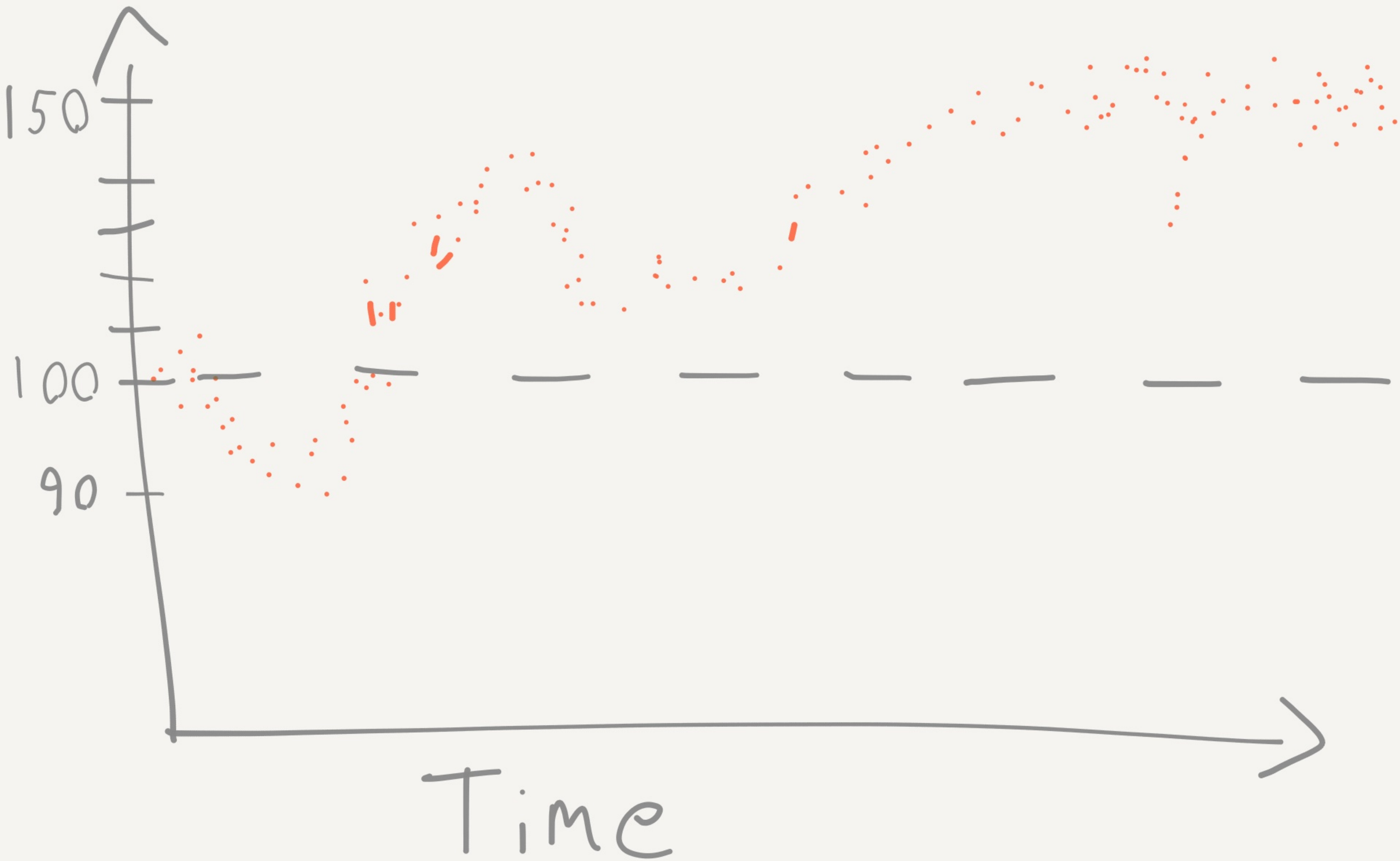
libbacktrace

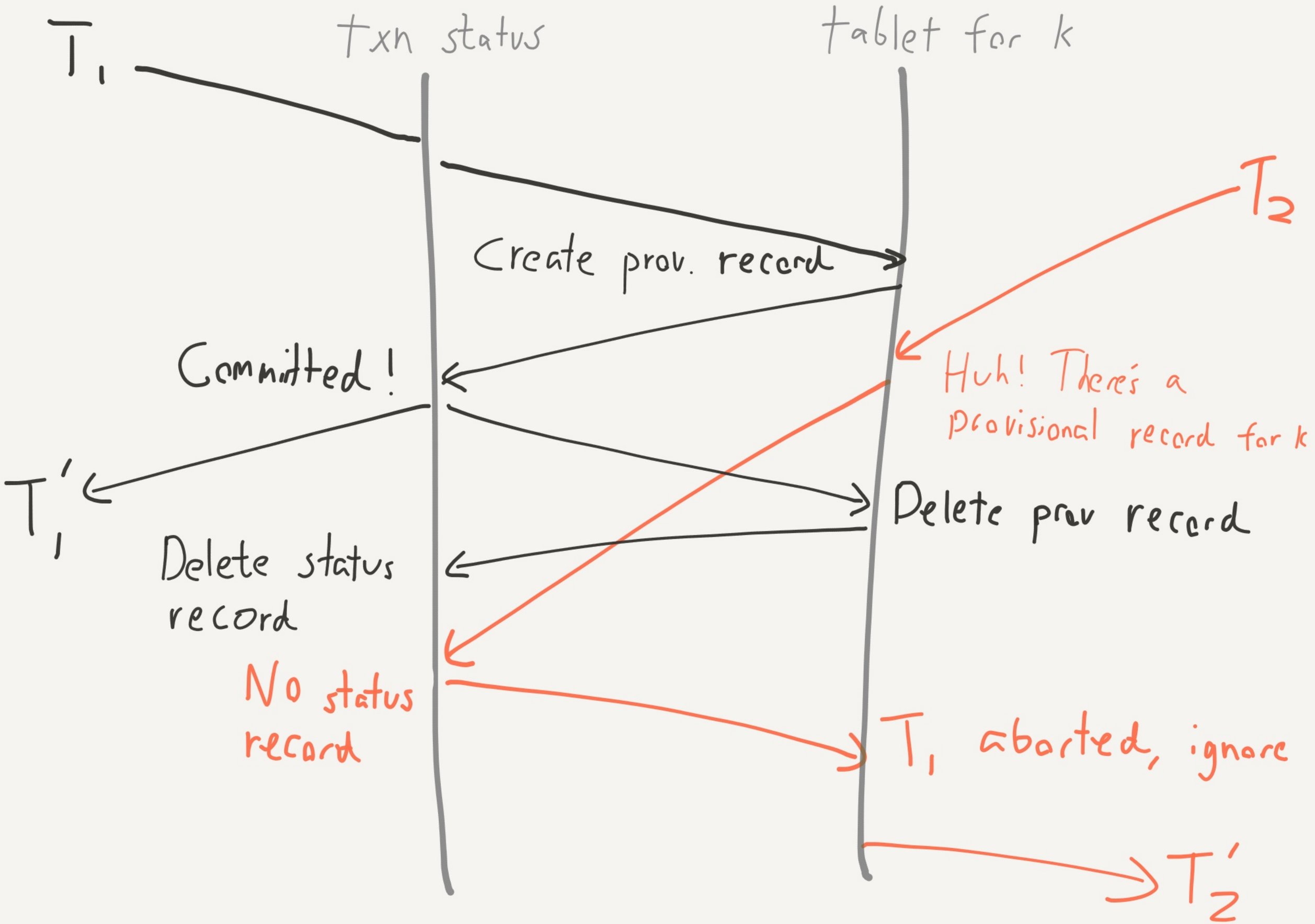
glog

Fixed in 1.1.15

#894 Read skew

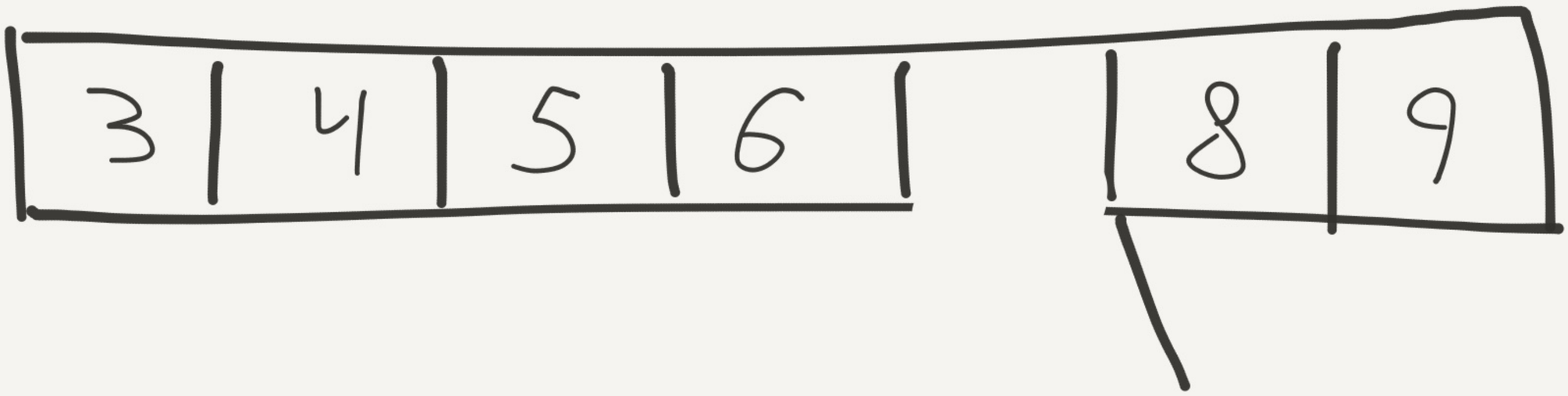



150
100
90
Time


T1
txn status
tablet for k
Create prov. record
T2
Committed!
Huh! There's a provisional record for k
T1'
Delete prov record
Delete status record
No status record
T1 aborted, ignore
T2'

Fixed in 1.1.10!

#824 Lost inserts



Partition ... Partition

↳ Membership change

Invalid election

Non-voting members counted towards majority for commit!

→ Fixed in 1.1.13

#975 More read skew

Requires clock skew

LockBatch failed to

transfer a status

field when moved

Fixed in 1.20!

Where does this leave us?

- [x] Counter
- [x] Set
- [x] Long Fork
- [x] Bank
- [x] Single-key linearizability
- [x] Multi-key linearizability

good clocks

- [x] Counter
- [ ] Set
- [x] Long fork
- [x] Bank
- [x] Single-key linearizability
- [ ] Multi-key linearizability

clock skew

key 1 must be 3

2:

(r 2 1)(r 1 2)

k1 is 2?

8:

(w 1 3)

time ⟶

- Definitely stale reads

- We suspect there should be multi-key txn anomalies

Two types of Clock

1. Raft Leader Lease
(CLOCK_MONOTONIC)

2. Txn timestamps
(CLOCK_REALTIME)

If CLOCK_MONOTONIC fails...

- Stale single-key reads

If CLOCK_REALTIME fails...

- Stale multi-key reads

- Read skew?

- ???

--max_clock_skew_usec

→ 50ms

--disable_clock_sync_error

→ 100s

You MUST measure skew!

Up Next

- SQL tests
- Generalized transactions
- Performance & tuning
- Formal model?

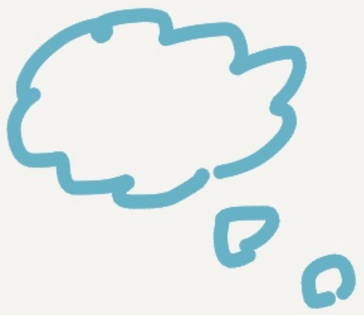
Thanks

- YungByte

- Jessie Frazelle

- Kit Patella

- Peter Bailis



Thanks

- Peter Alvaro

- Diego Ongaro

https://jepsen.io