FoundationDB

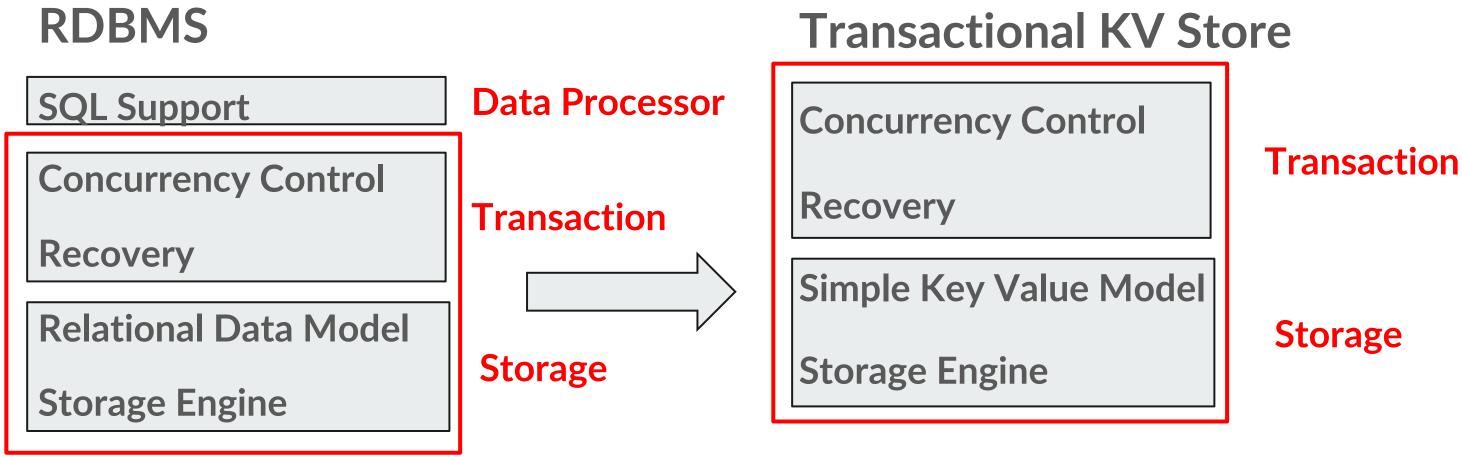
A Distributed Unbundled Transactional Key Value Store

Yuhan Wang

Agenda

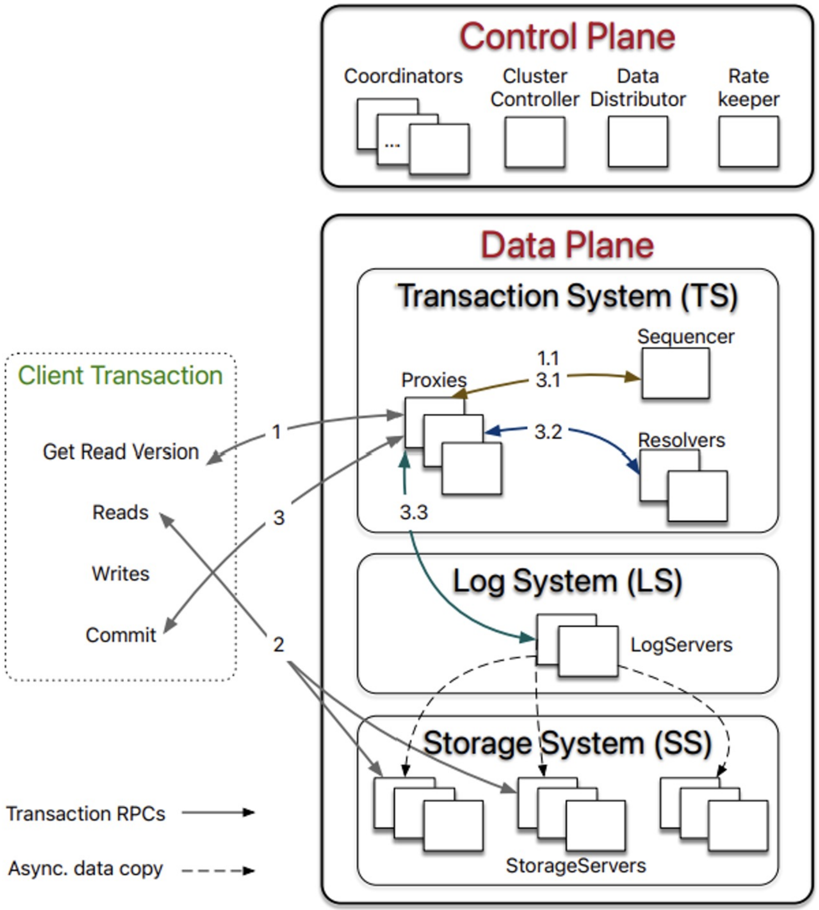
- Background
- Design
- Contributions
- Limitations

Background: why a transactional key value store?



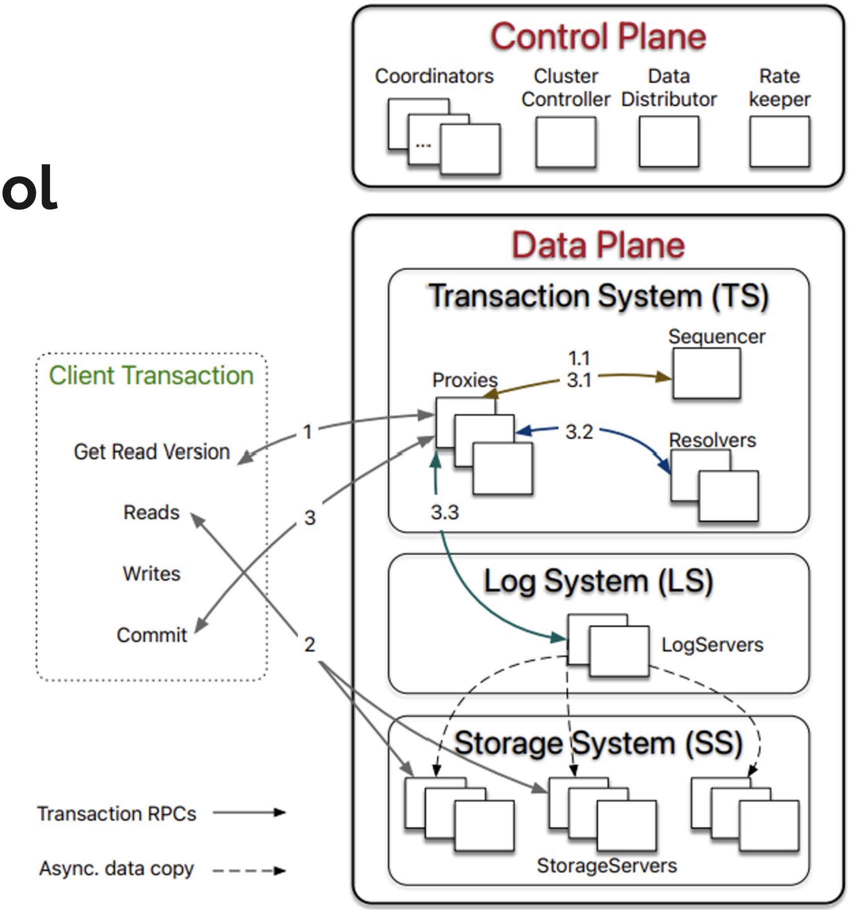
Design: Architecture

- Coordinators: Paxos
- Transaction System: Stateless
- Log System: Stateful
- Storage System: Stateful

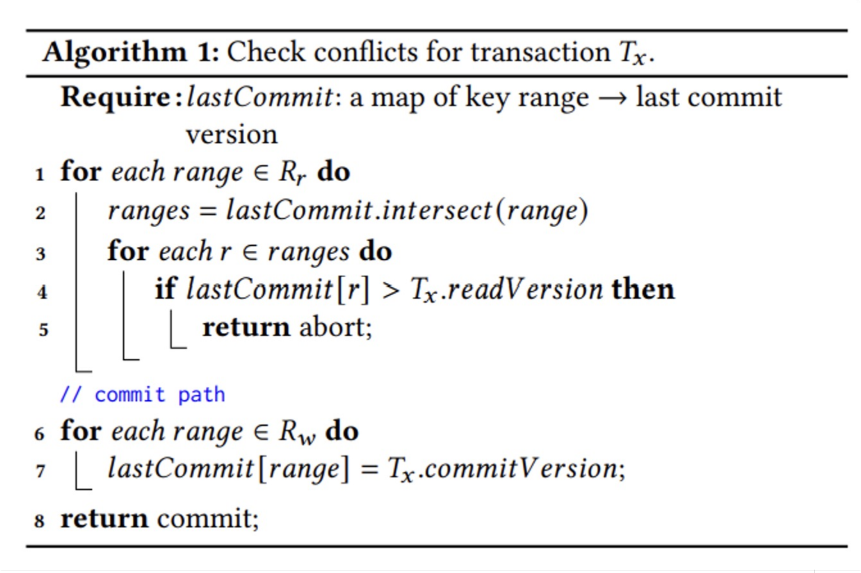


Design: Concurrency Control

- Strict Serializability
- Backward Validation OCC
- MVCC
- Commit Version：
    - prev version
    - current version

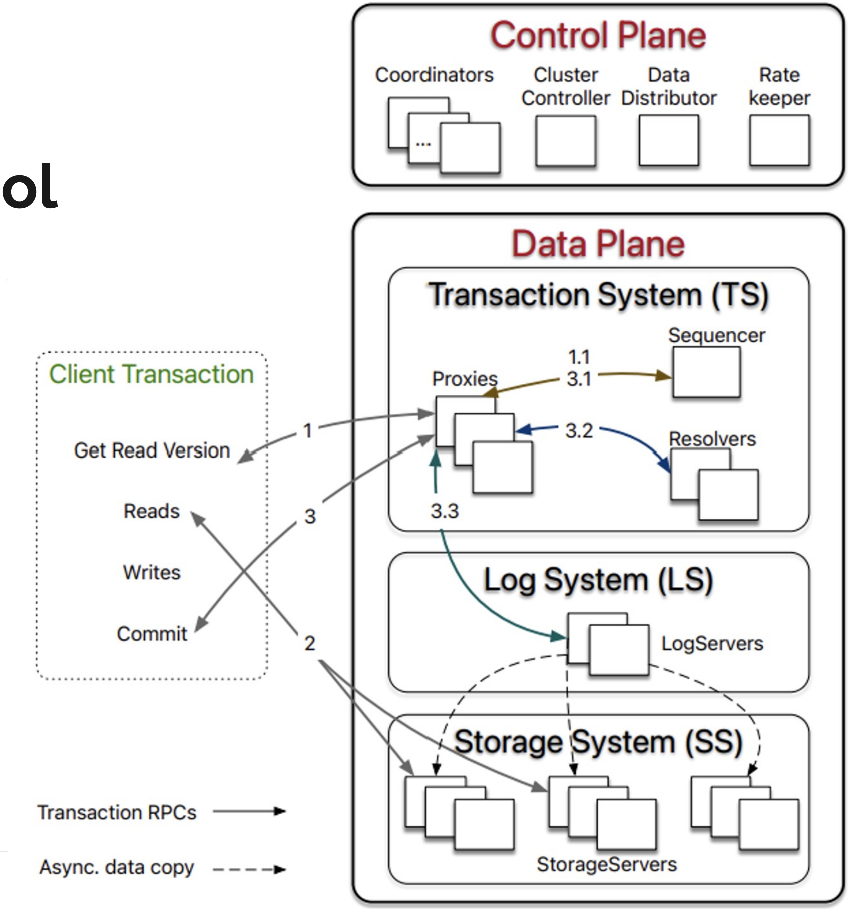


Design: Concurrency Control

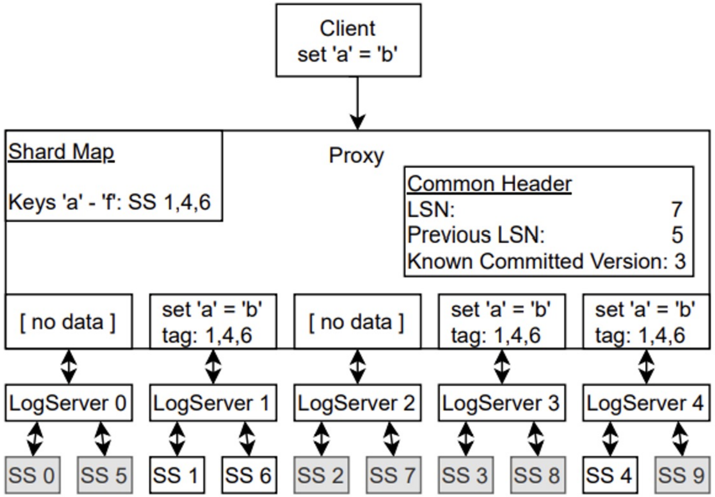


Algorithm 1: Check conflicts for transaction T_x.

```
Require: lastCommit: a map of key range → last commit
         version
for each range ∈ $R_r$ do
    ranges = lastCommit.intersect(range)
    for each r ∈ ranges do
        if lastCommit[r] > $T_x$.readVersion then
            return abort;
// commit path
for each range ∈ $R_w$ do
    lastCommit[range] = $T_x$.commitVersion;
return commit;
```



Design: Log System



Client
set 'a' = 'b'

Proxy

Shard Map

Keys 'a' - 'f': SS 1,4,6

Common Header
LSN: 7
Previous LSN: 5
Known Committed Version: 3

[no data]

set 'a' = 'b'
tag: 1,4,6

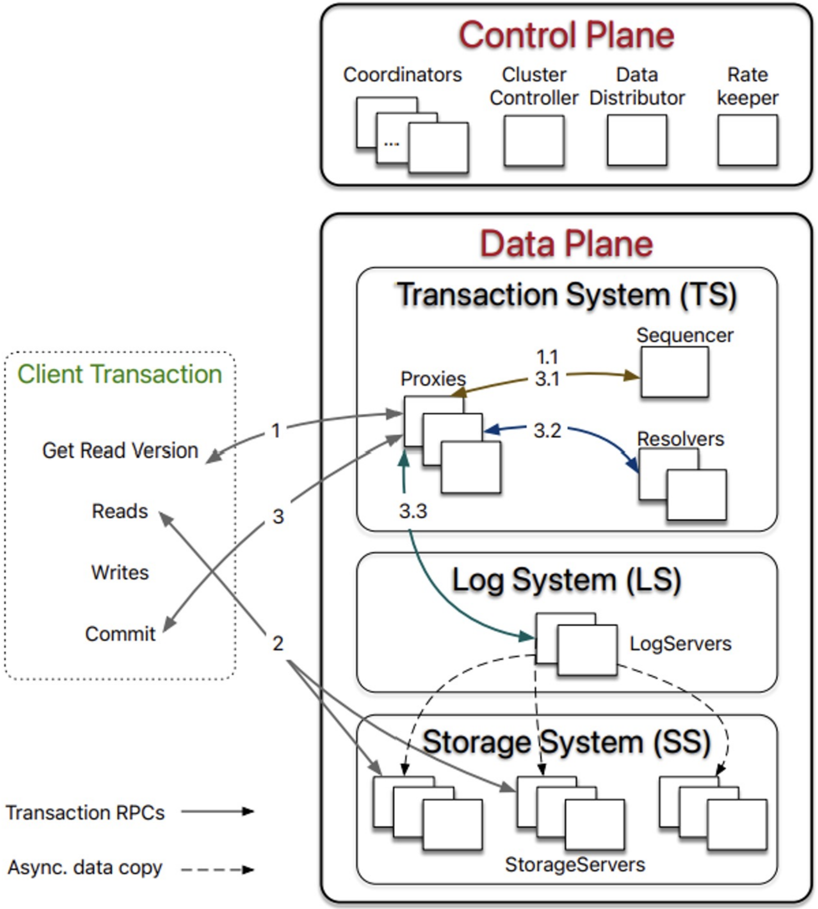
[no data]

set 'a' = 'b'
tag: 1,4,6

set 'a' = 'b'
tag: 1,4,6

LogServer 0
LogServer 1
LogServer 2
LogServer 3
LogServer 4

SS 0
SS 5
SS 1
SS 6
SS 2
SS 7
SS 3
SS 8
SS 4
SS 9



Control Plane

Coordinators

Cluster Controller

Data Distributor

Rate keeper

Data Plane

Transaction System (TS)

Sequencer

Proxies

1.1
3.1

3.2

Resolvers

3.3

Client Transaction

Get Read Version

Reads

Writes

Commit

1

3

2

Log System (LS)

LogServers

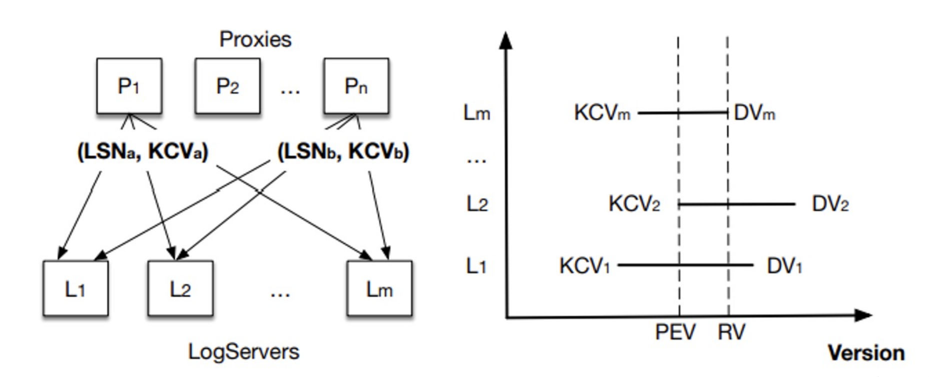
Storage System (SS)

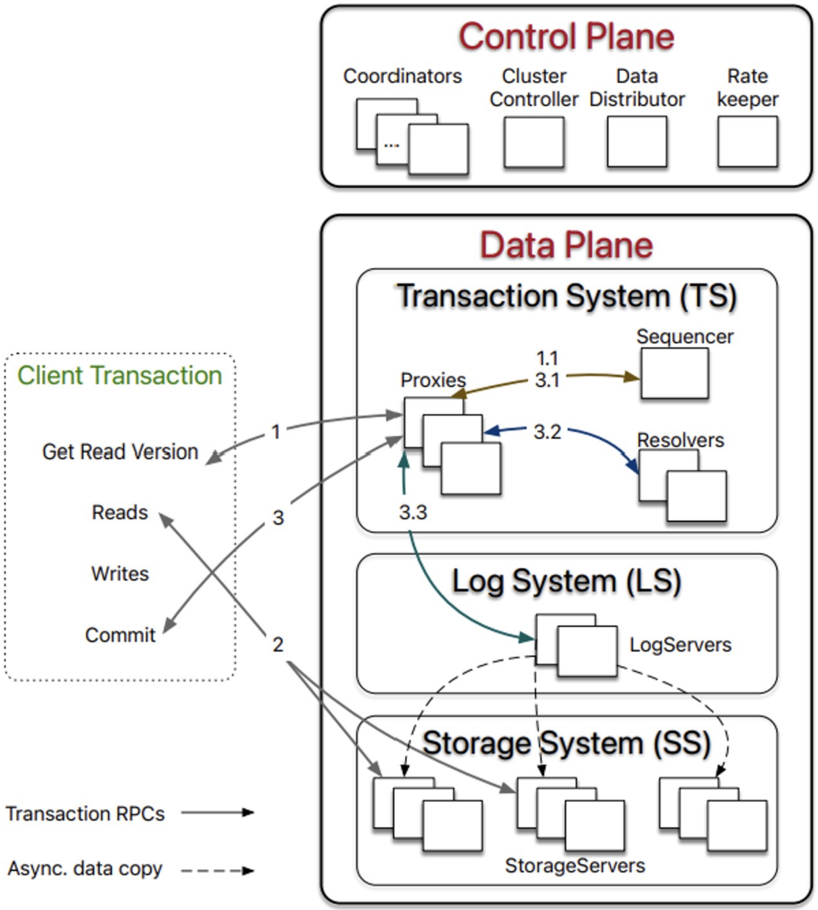
StorageServers

Transaction RPCs

Async. data copy

Design: Recovery





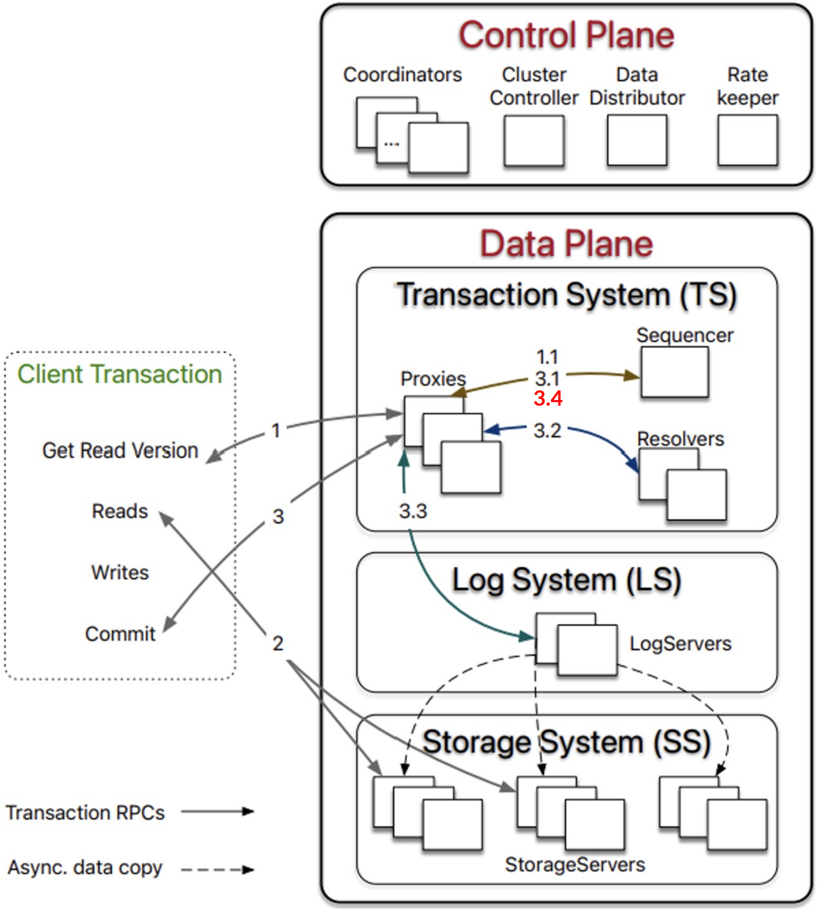
Design: Tricky Problem

Assign versions in Sequencer?

- Strict Serializability
- SS Aggressive Fetching

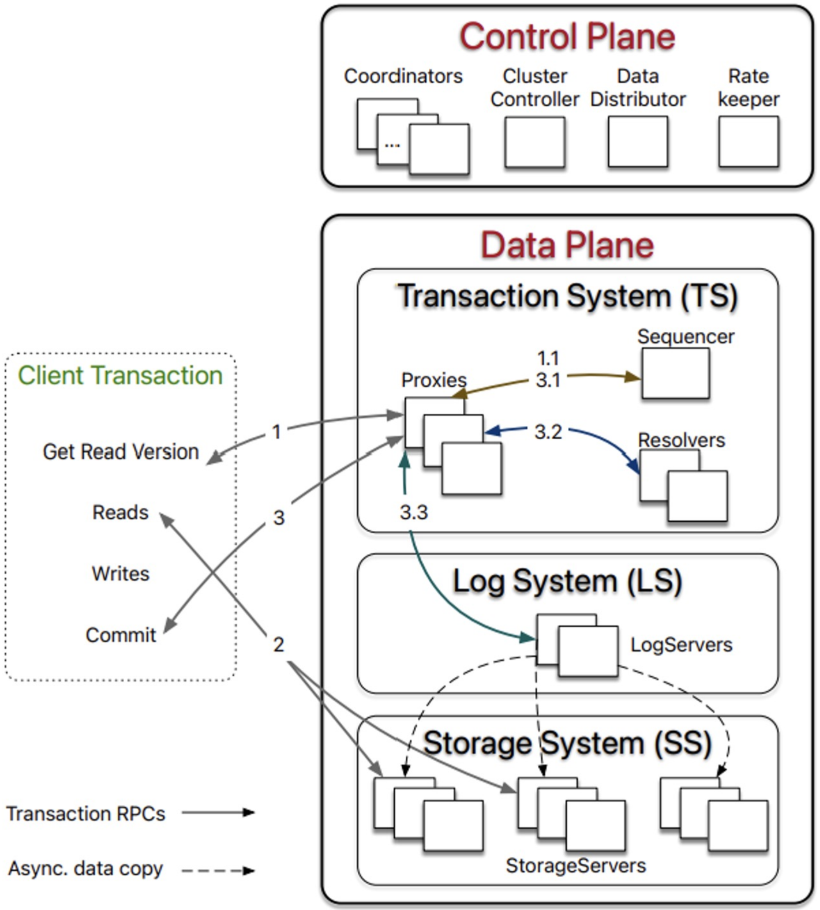
Solution:

- Increasing Commit Versions
- Lasted Commited Version (as Read Version)



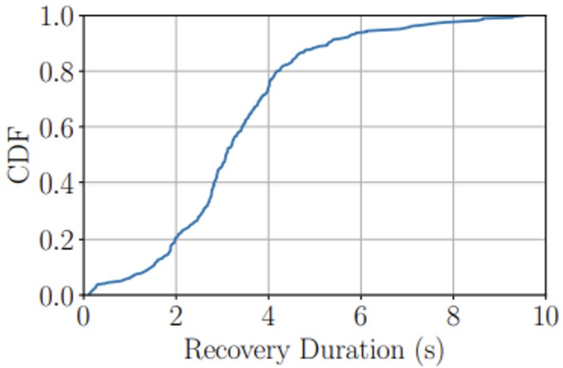
Contributions

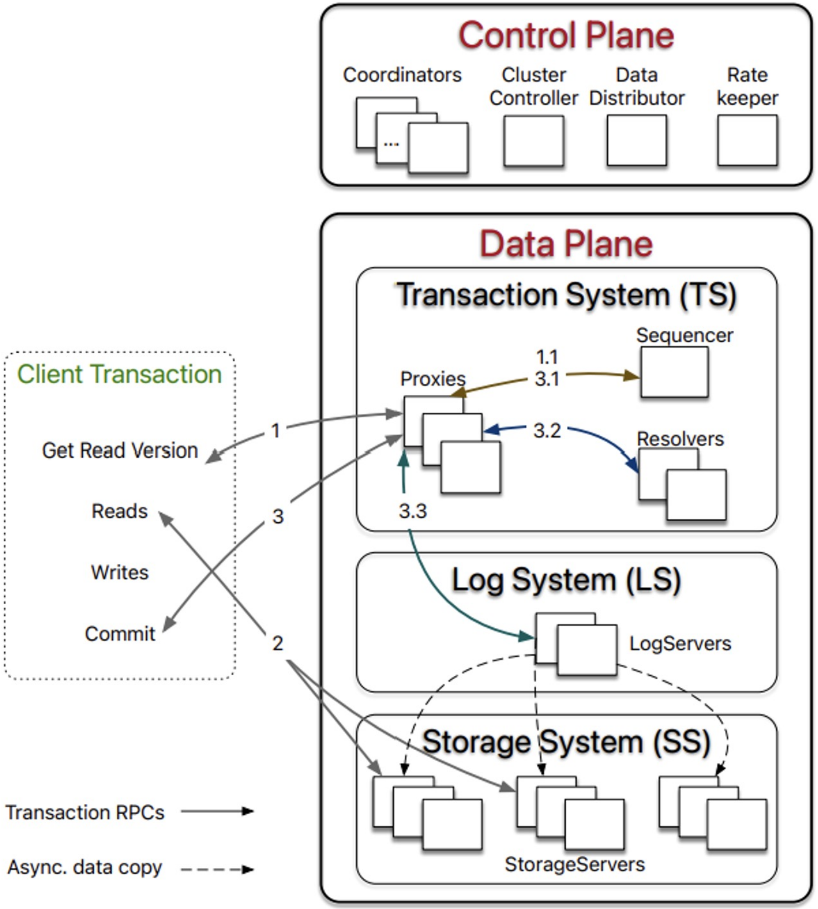
- OCC vs Two Phase Locking
- Fast Recovery vs Always On
- Optimize For Happy Case



Limitations

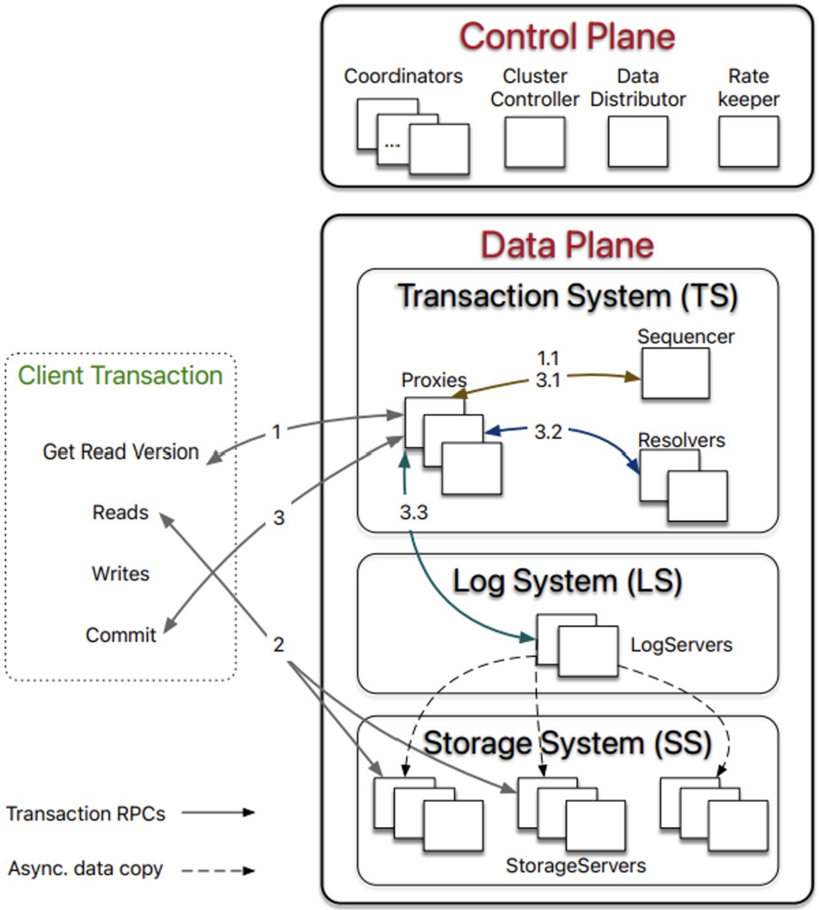
- Transaction Size (OCC)
- 5s MVCC Window
- No RW Transactions in Recovery





Thanks!

And Questions?


Control Plane
Coordinators
Cluster Controller
Data Distributor
Rate keeper
....
Data Plane
Transaction System (TS)
Sequencer
1.1
3.1
Proxies
3.2
Resolvers
3.3
Client Transaction
Get Read Version
Reads
Writes
Commit
1
3
2
Log System (LS)
LogServers
Storage System (SS)
StorageServers
Transaction RPCs
Async. data copy