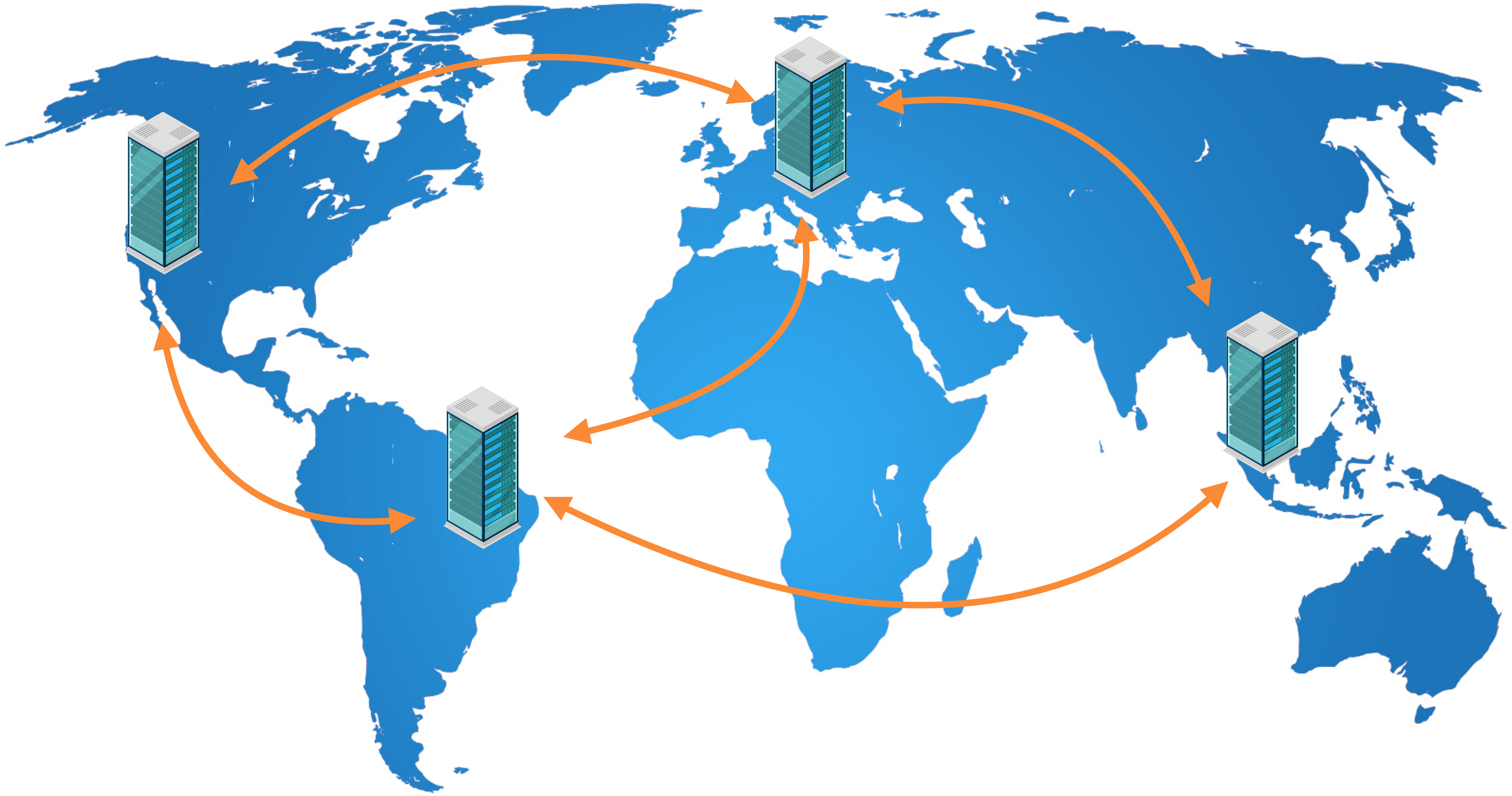
CRDTs in Production

Dmitry Martyanov, Software Engineer @ PayPal

QCon, 2018

Geo-Distributed Datastore



Context

- More than 200 countries
- Regulatory requirements
- State Machine of Compliance Status
- Modified by multiple Actors

Shared Mutable State

Shared Mutable State

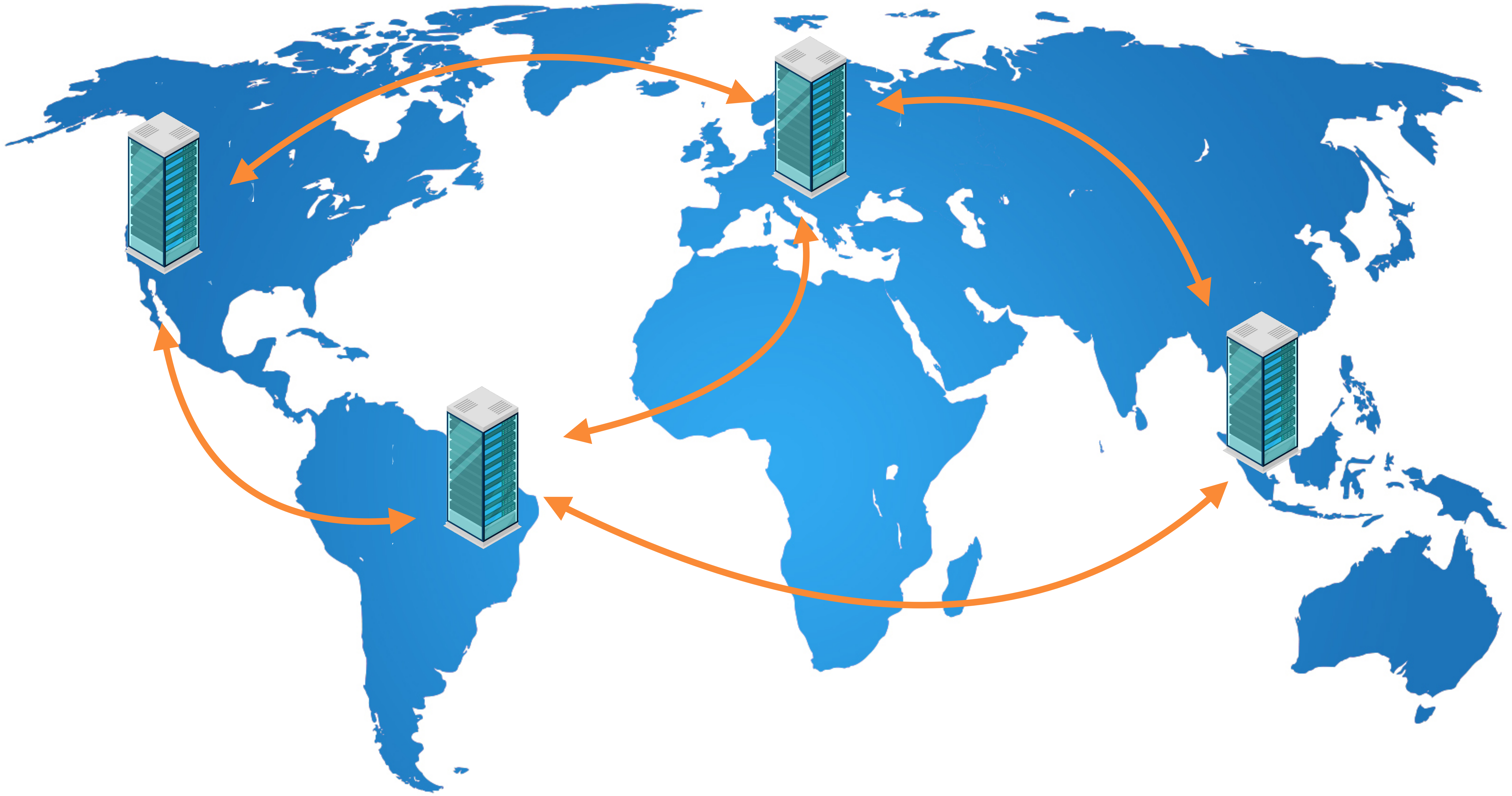
Mutex

Shared Mutable State

Mutex

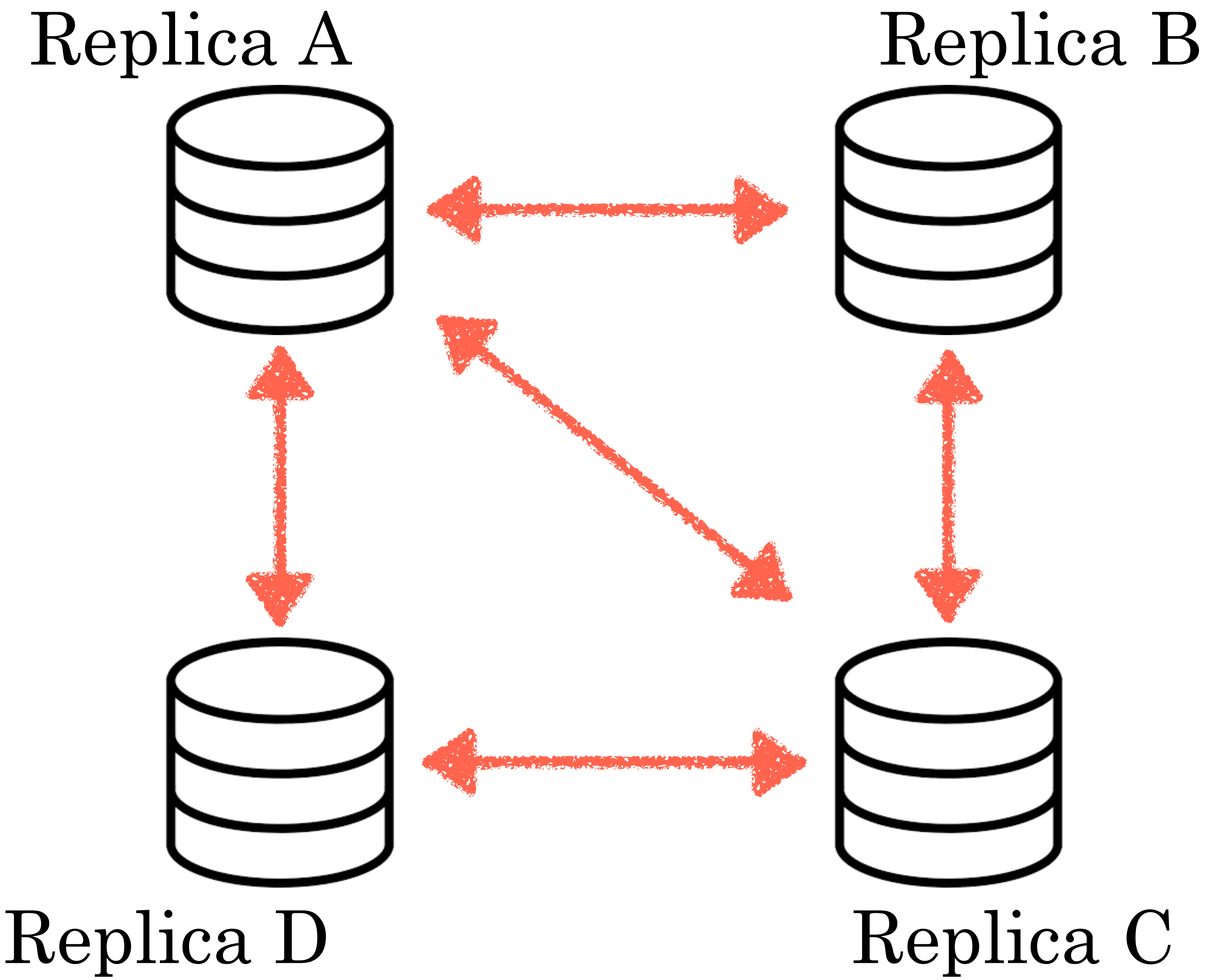
Transactions

Geo-Distributed Datastore

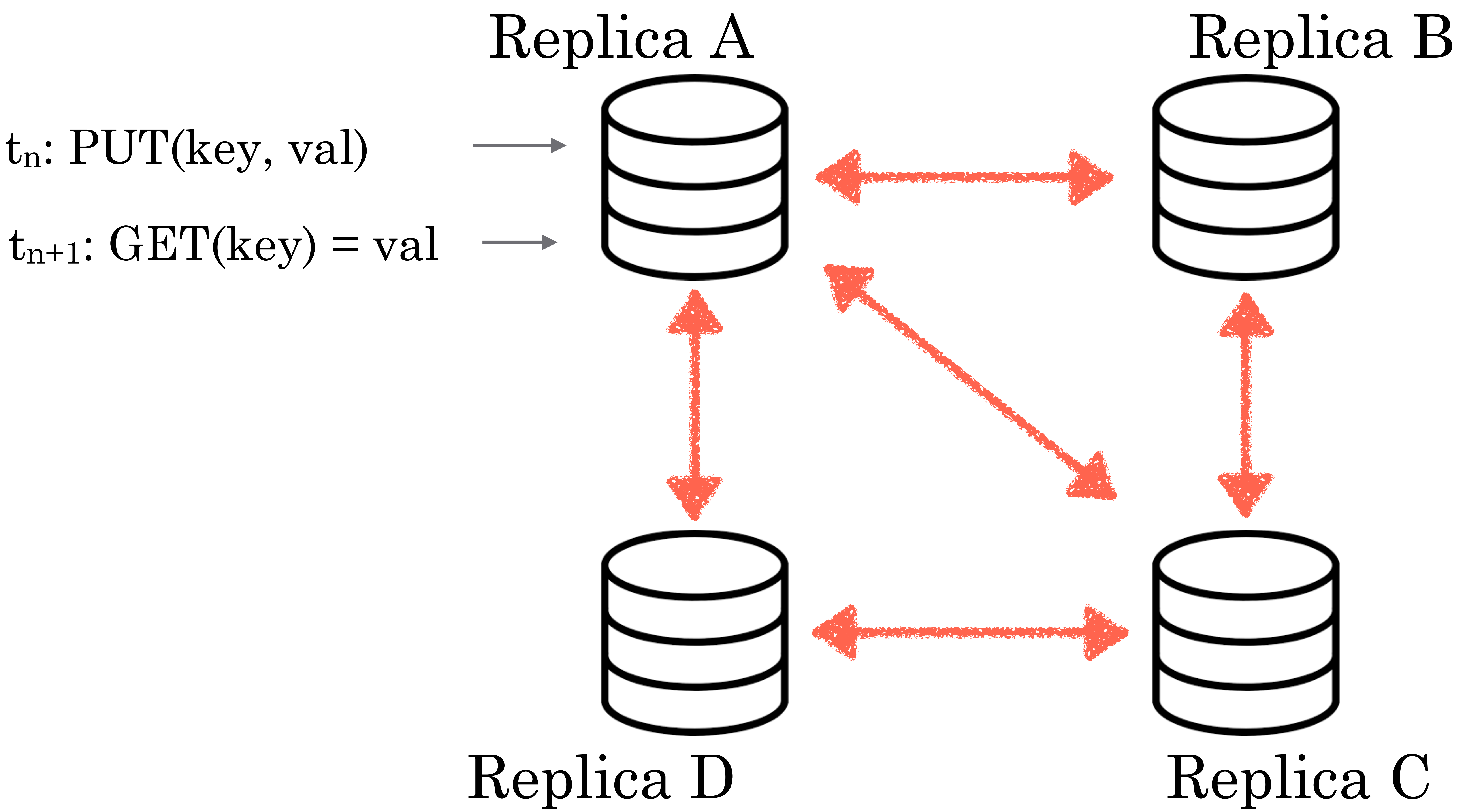


Eventual Consistency

Distributed System



Distributed System



t_n: PUT(key, val)

t_{n+1}: GET(key) = val

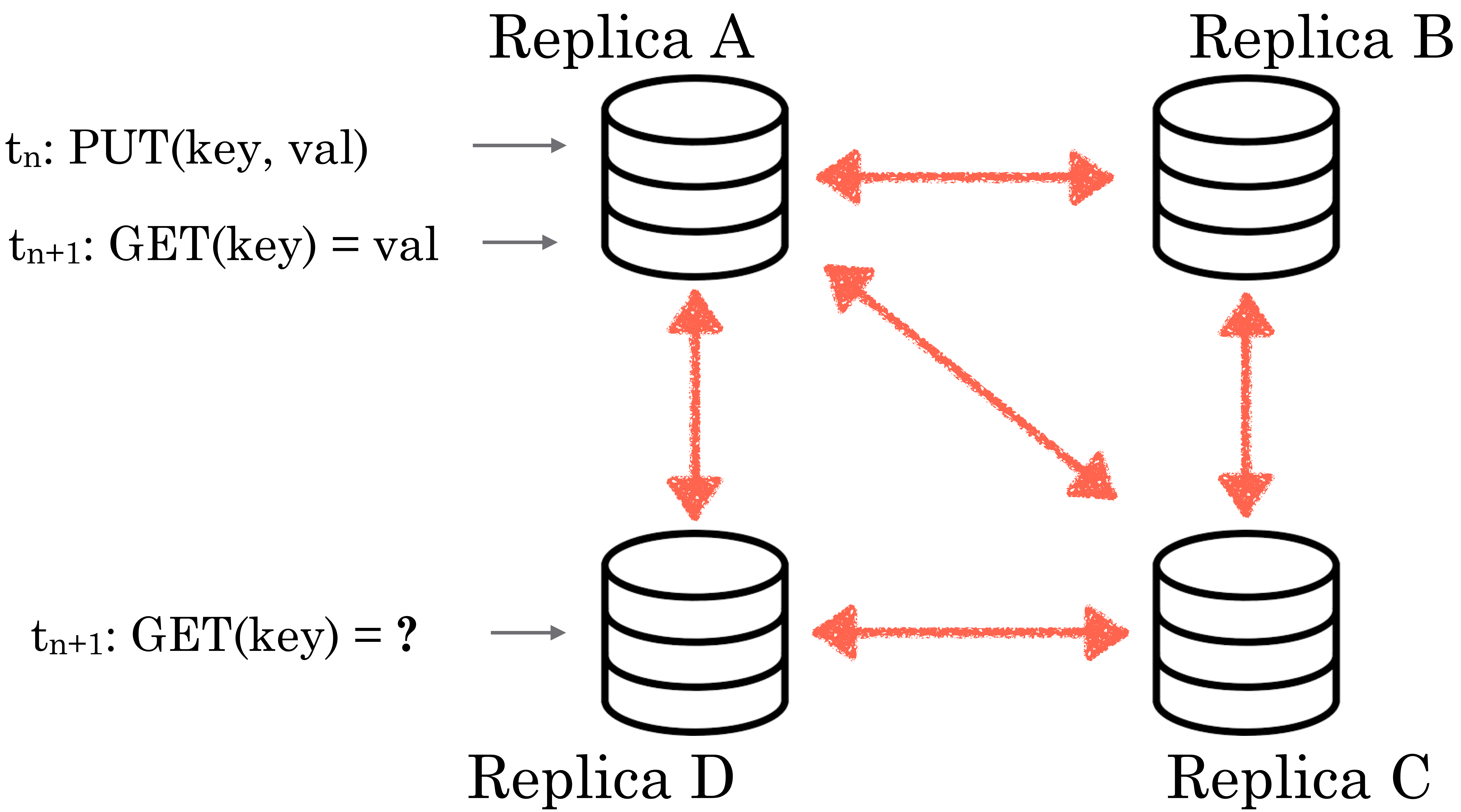
Replica A

Replica B

Replica D

Replica C

Distributed System



Replica A

Replica B

t_n: PUT(key, val)

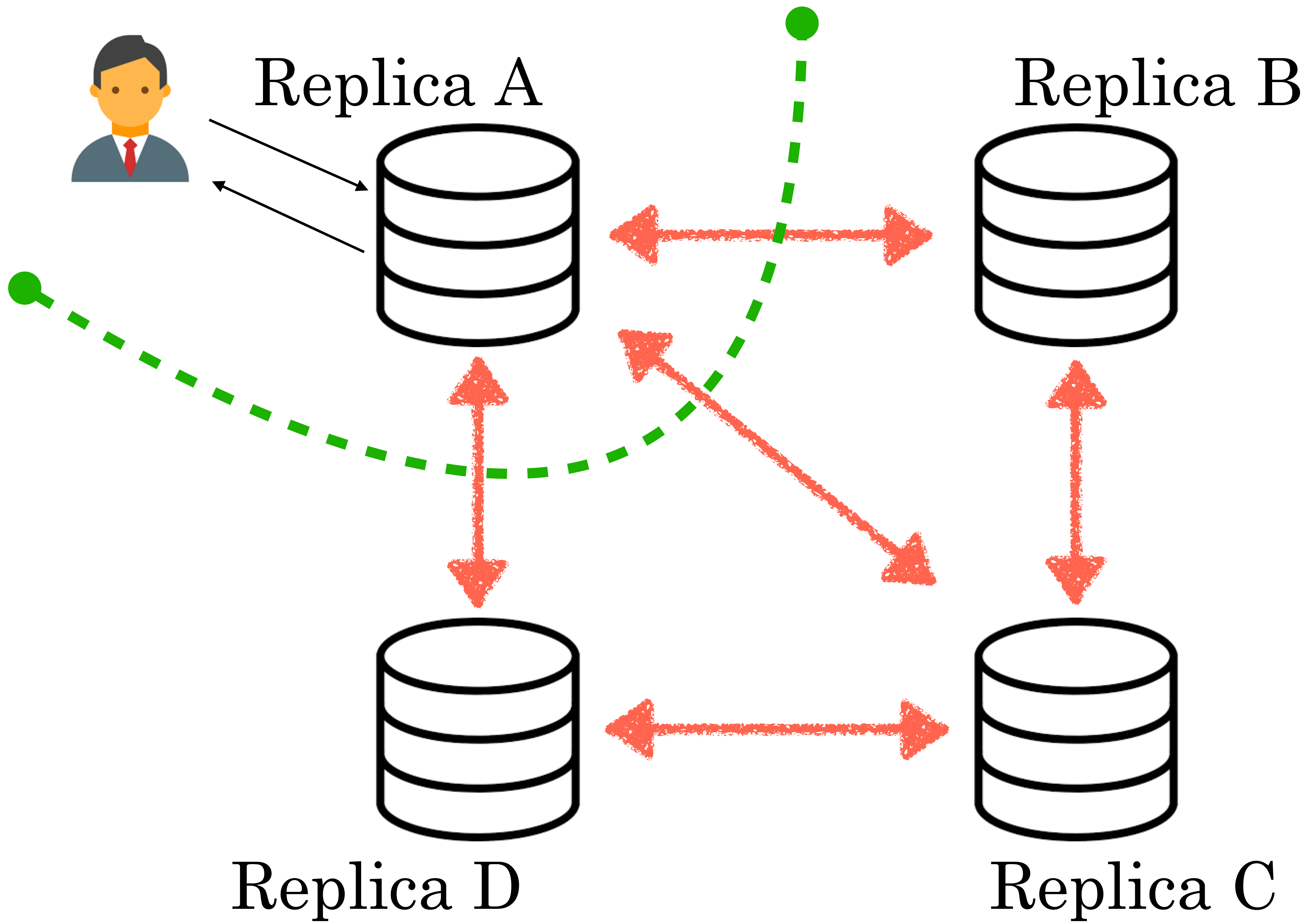
t_{n+1}: GET(key) = val

t_{n+1}: GET(key) = ?

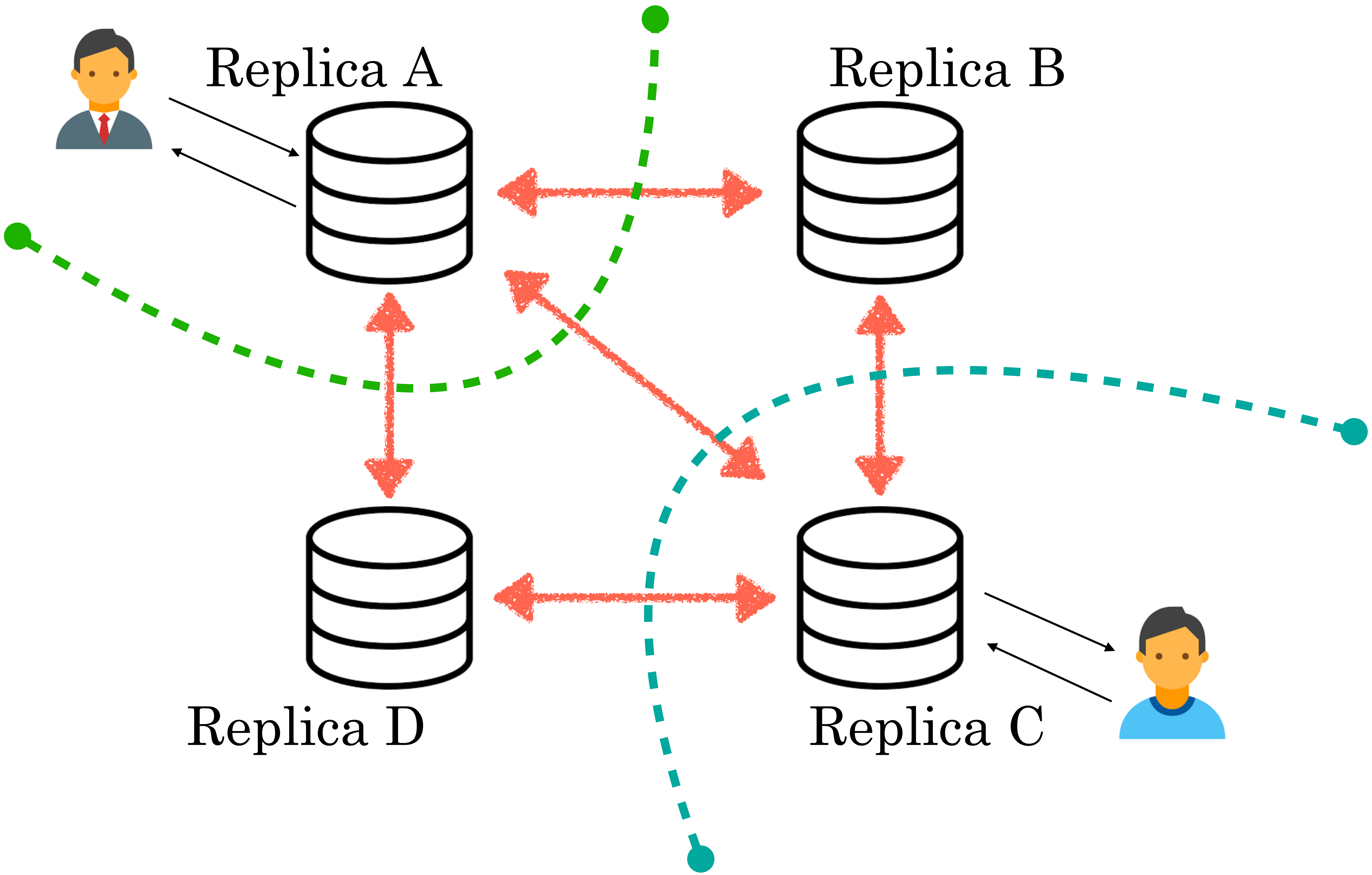
Replica D

Replica C

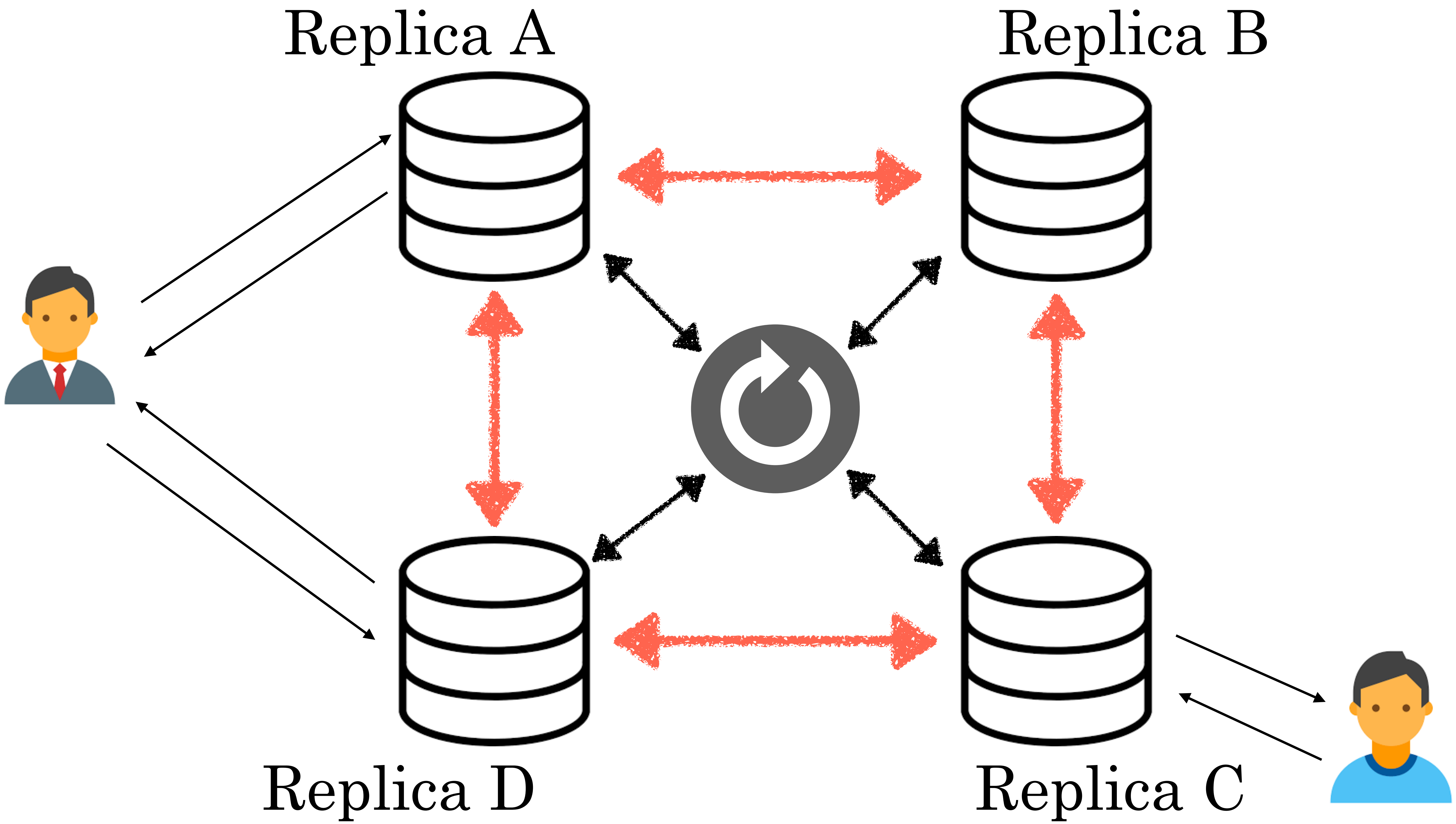
Affinity Based Approaches



Affinity Based Approaches

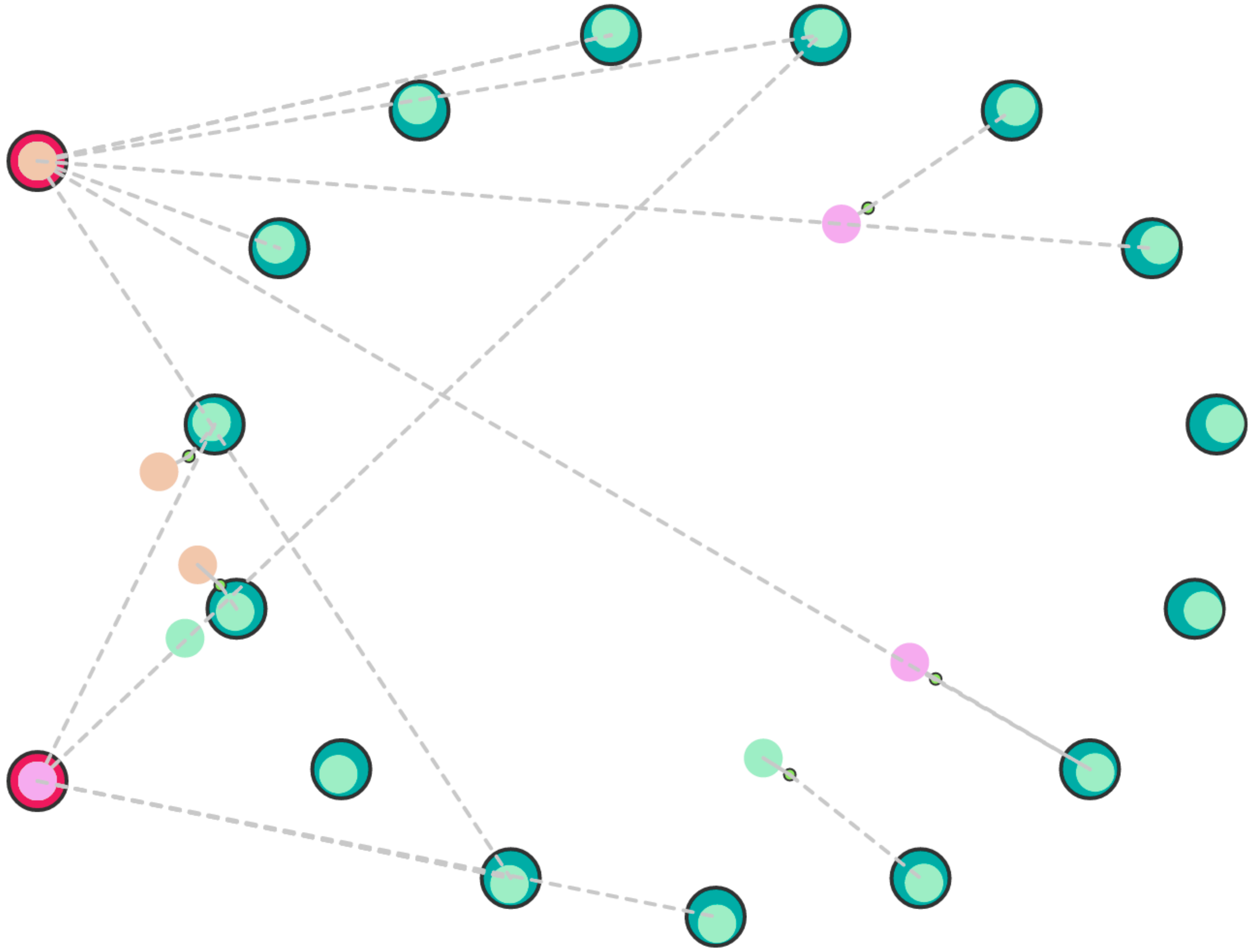


Coordinator Based Approaches

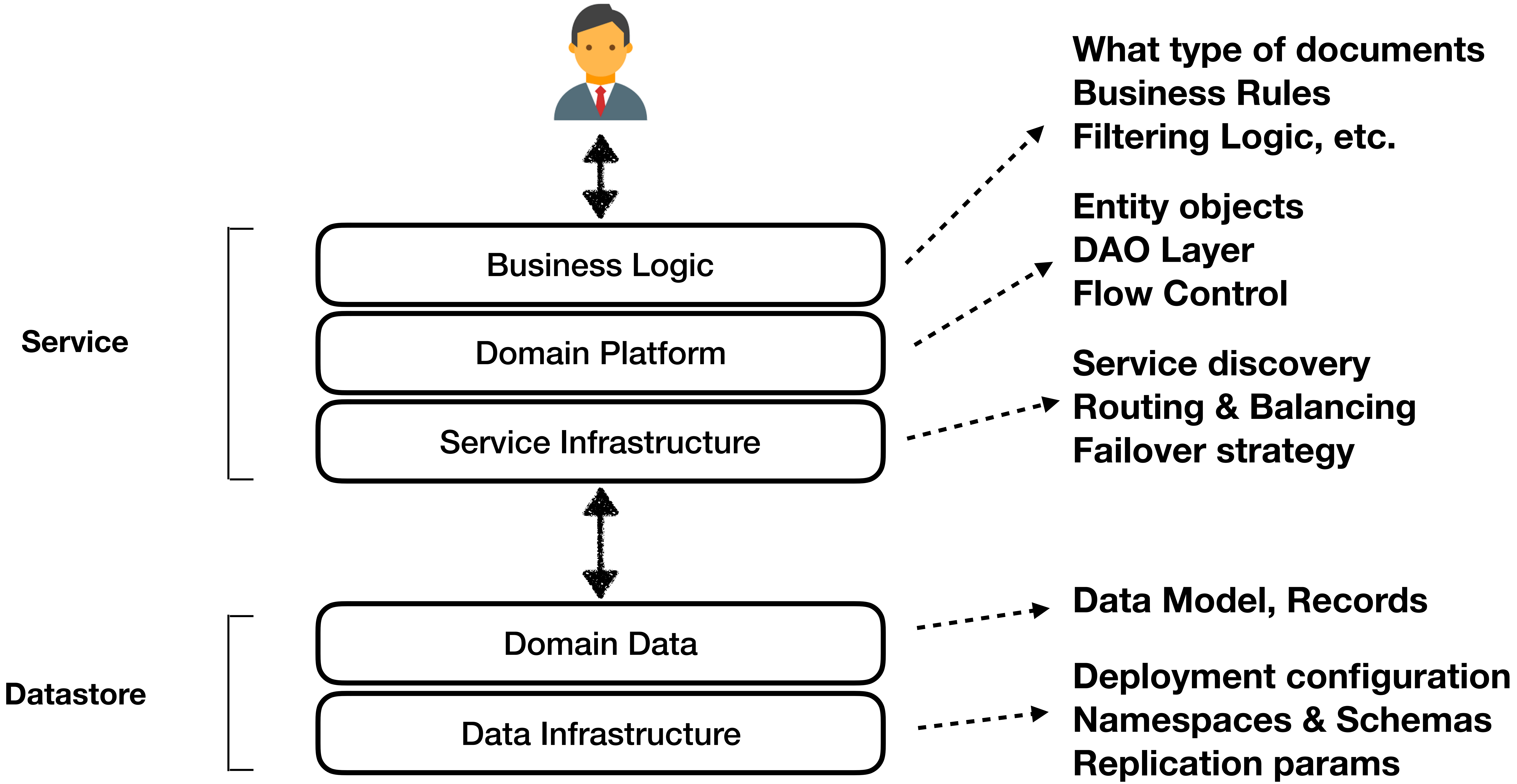


Consensus Based Approaches

Paxos, Raft, etc.



Service Stack



Service

Business Logic

Domain Platform

Service Infrastructure

Datastore

Domain Data

Data Infrastructure

What type of documents
Business Rules
Filtering Logic, etc.

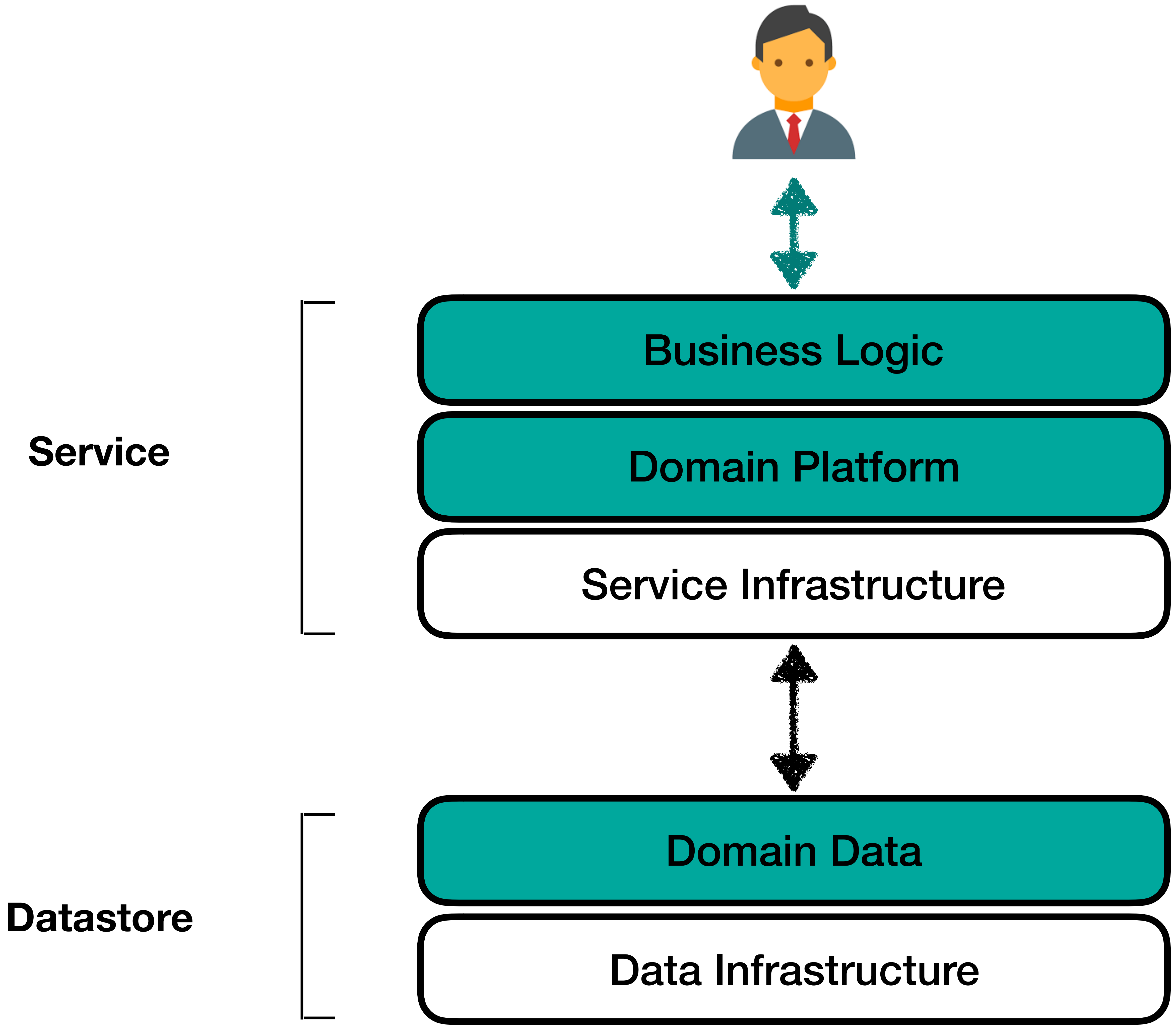
Entity objects
DAO Layer
Flow Control

Service discovery
Routing & Balancing
Failover strategy

Data Model, Records

Deployment configuration
Namespaces & Schemas
Replication params

Service Stack



Service

Business Logic

Domain Platform

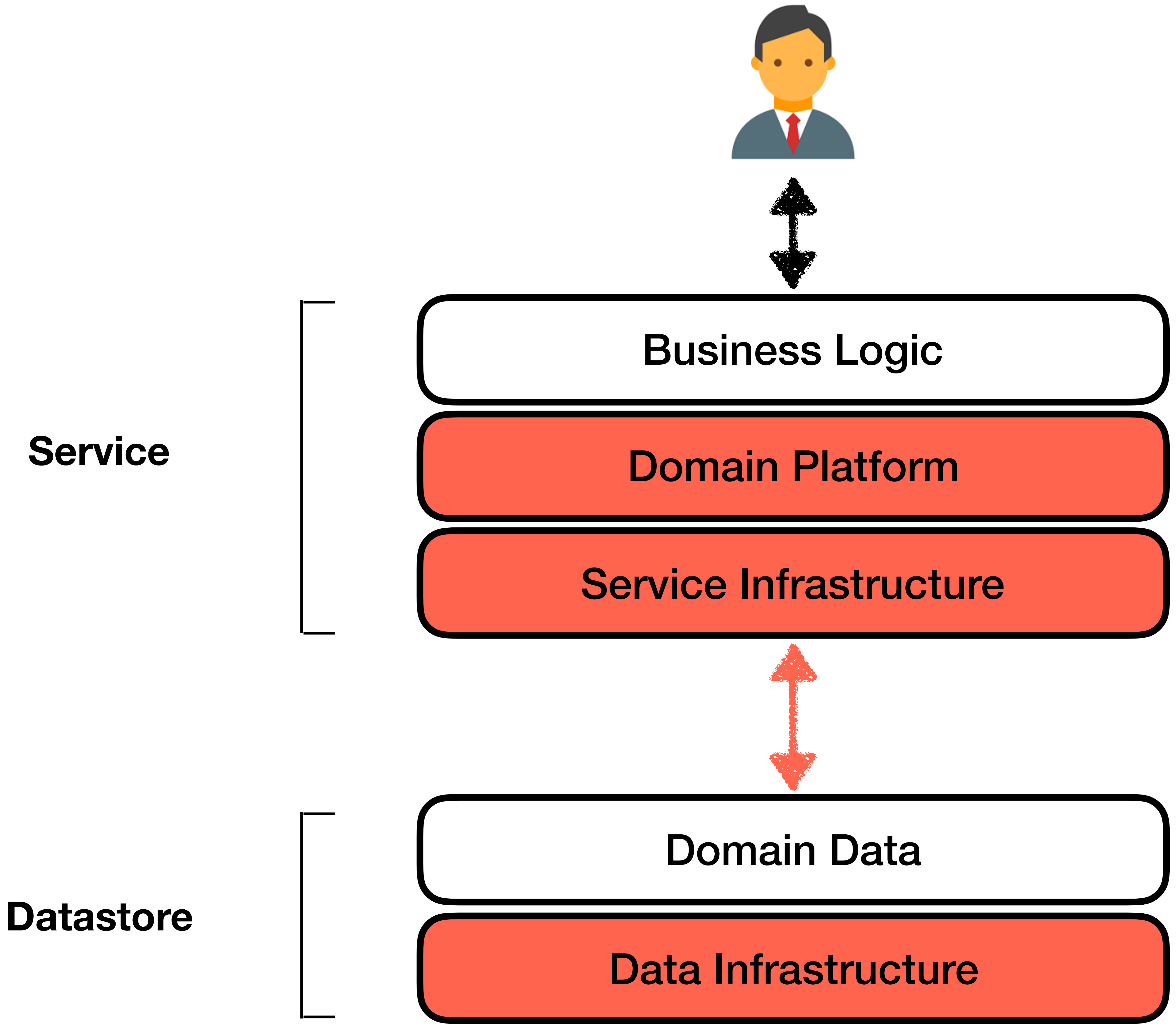
Service Infrastructure

Datastore

Domain Data

Data Infrastructure

Service Stack



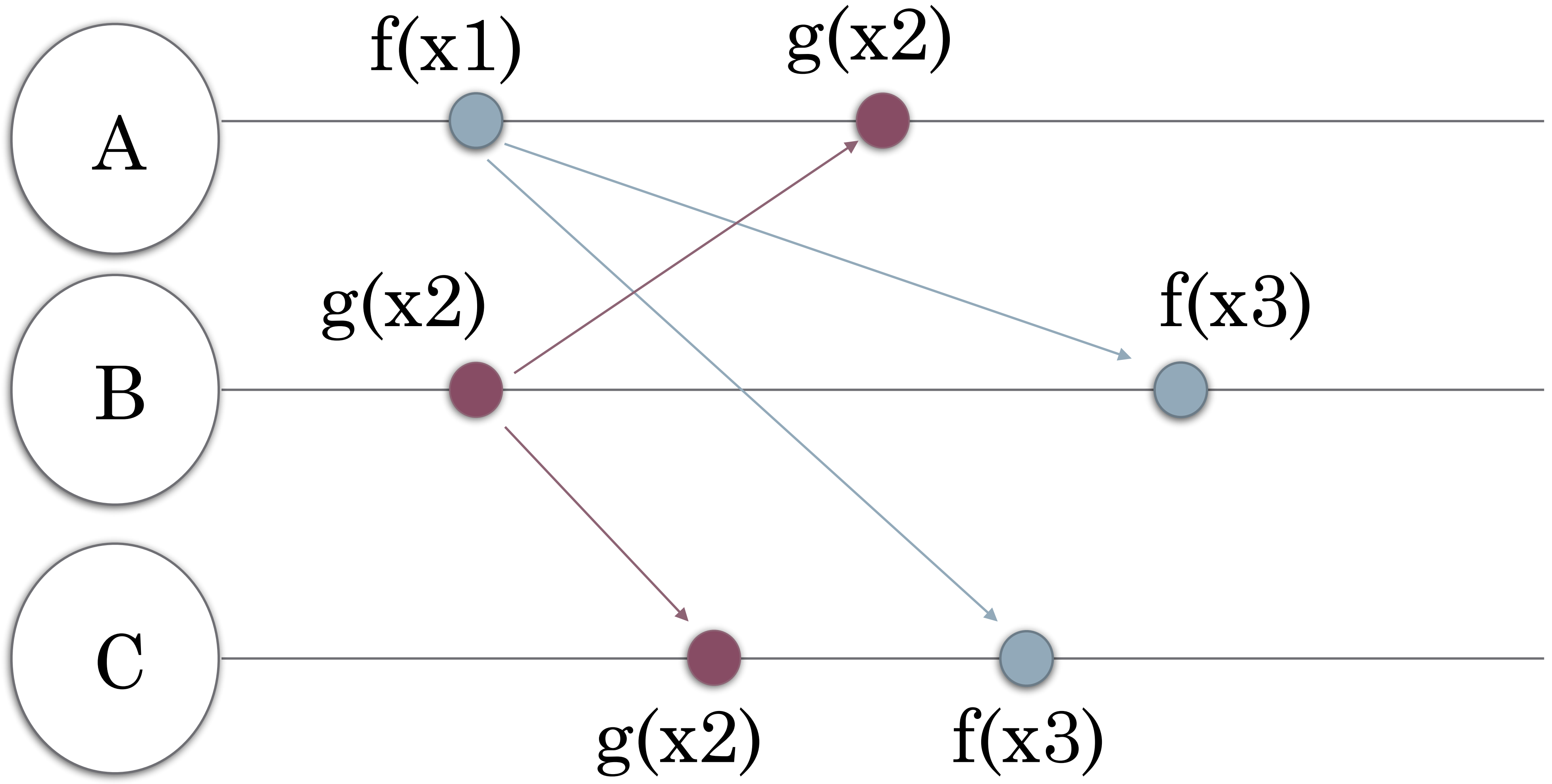
Conflict-free Replicated Data Types

CRDTs

commutative

Requirements:

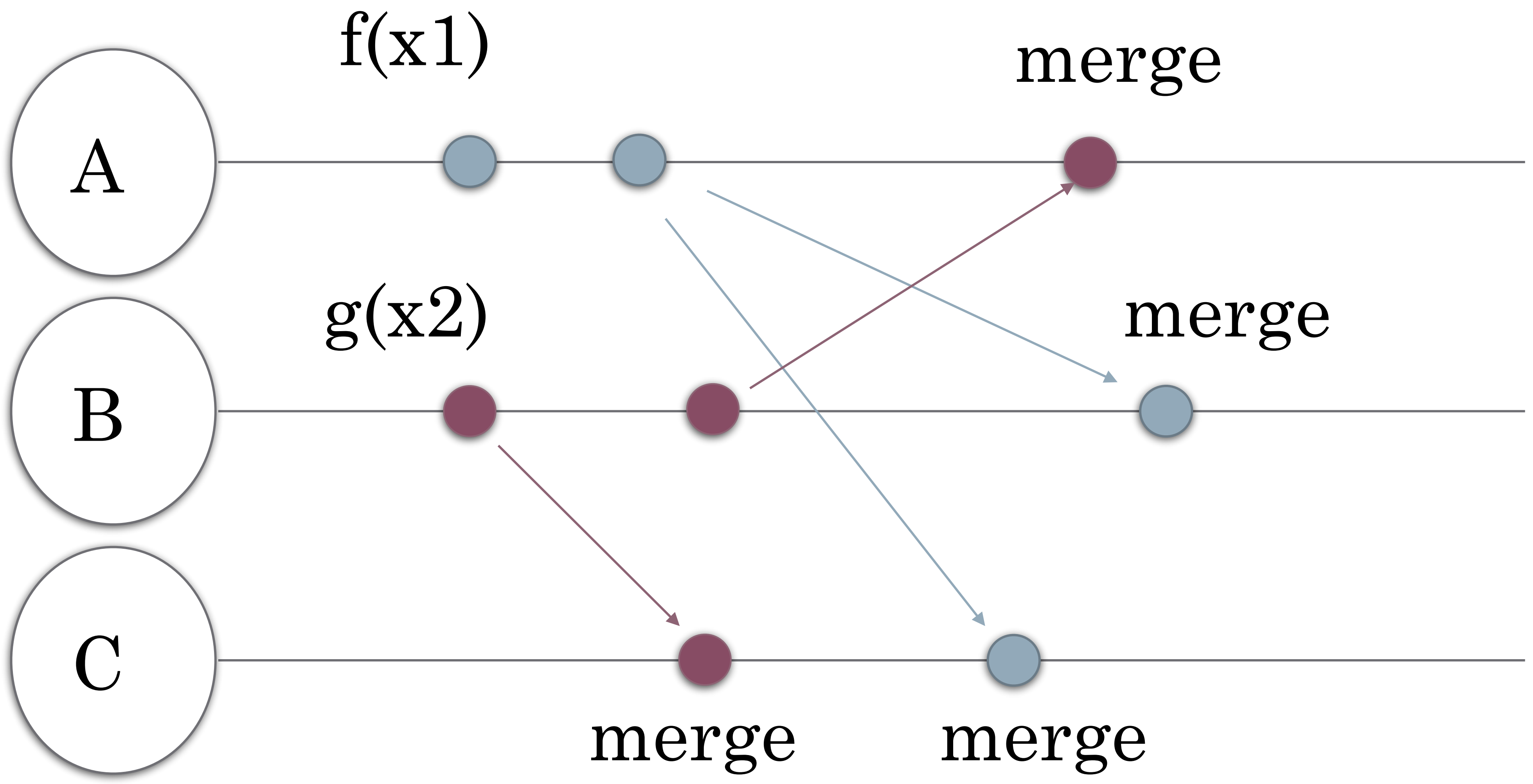
- + Commutativity
- + Associativity
- + Exactly once delivery
- - Idempotence



convergent

Requirements:

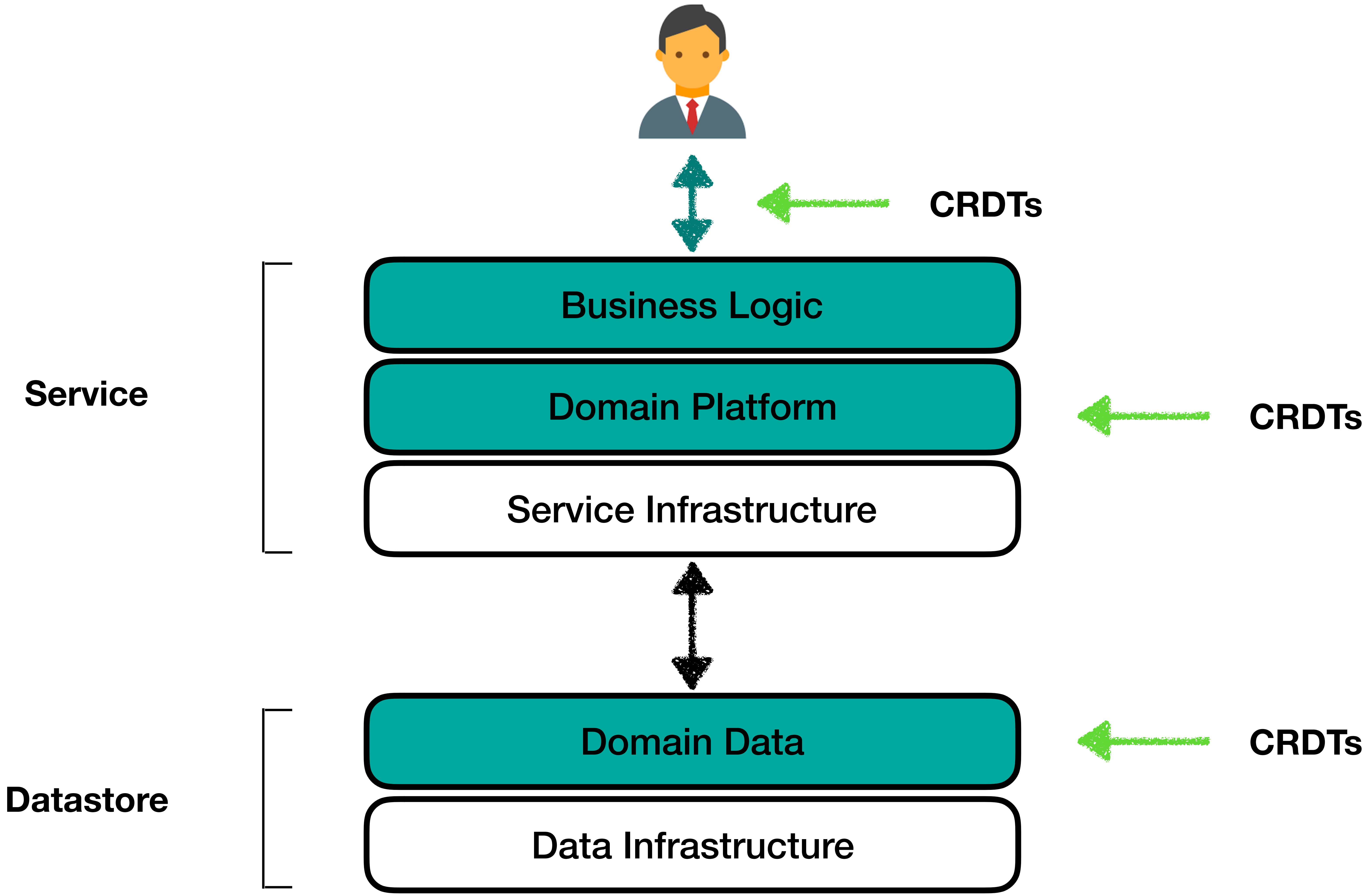
- + Commutativity
- + Associativity
- + Idempotence
- - Exactly once delivery



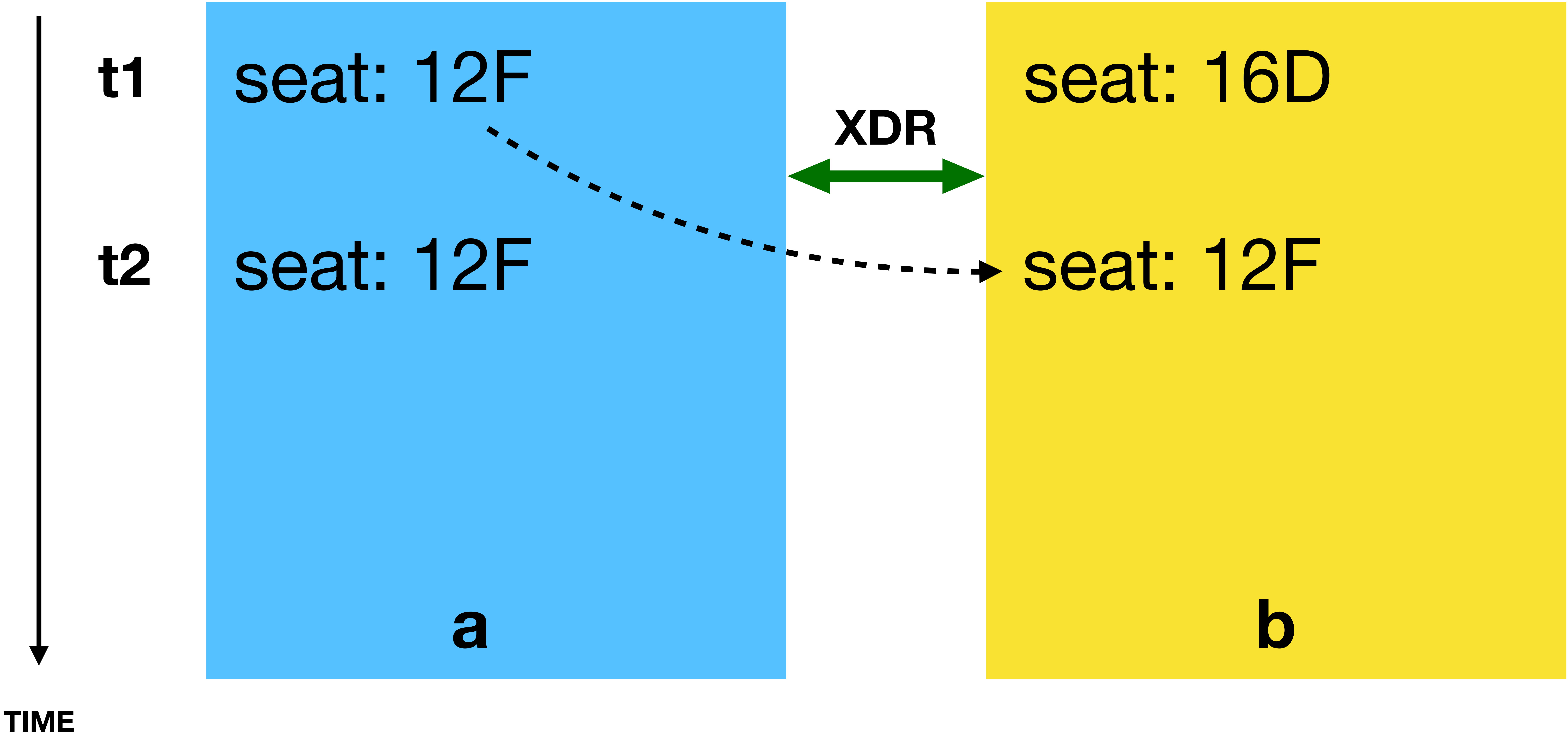
Convergent CRDTs

- $M(a, b) = M(b, a)$
- $M(M(a, b), c) = M(a, M(b, c))$
- $M(a, b) = M(M(a, b), b) = M(M(M(a, b), b), b)$

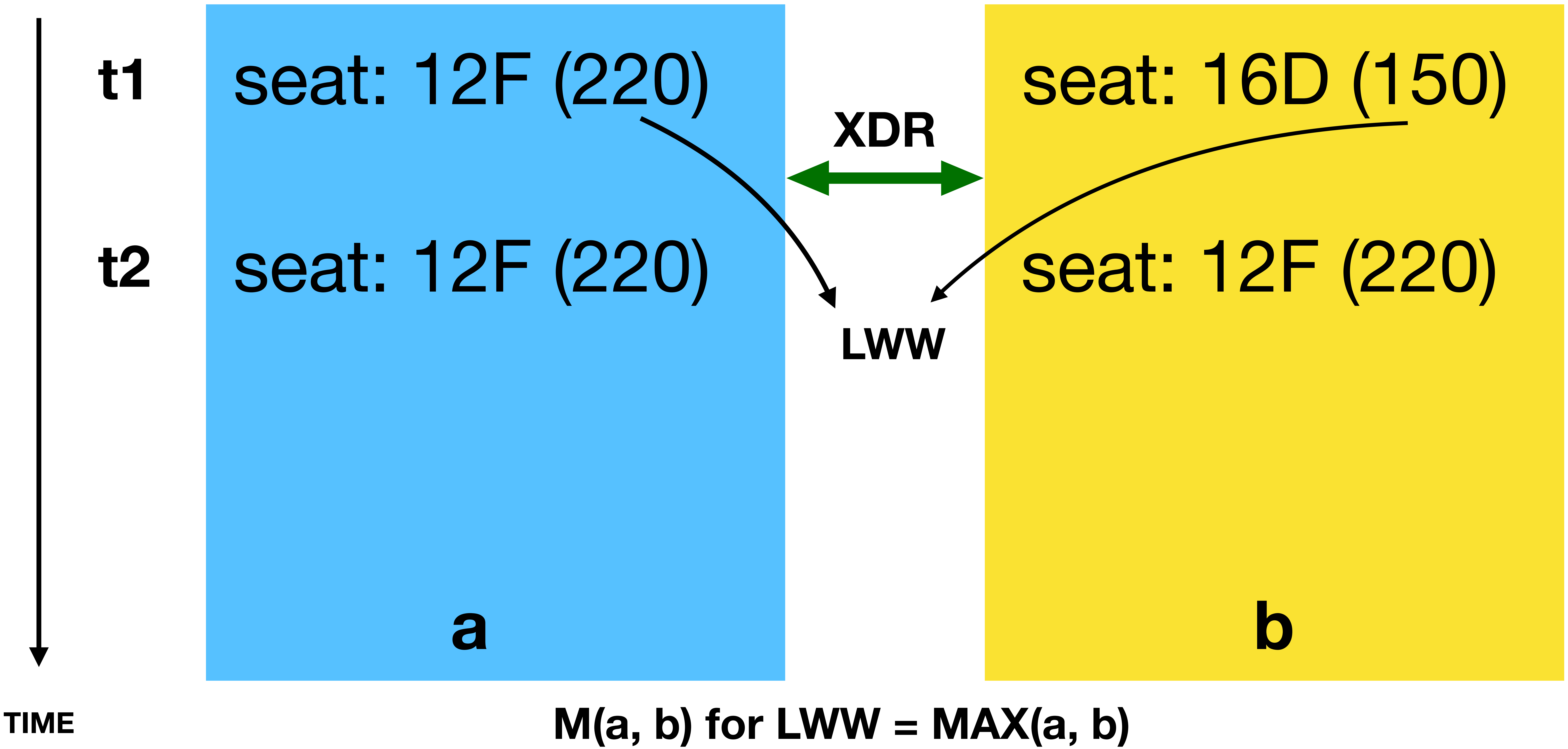
Impacted Components for CRDTs



Online Flight Check-in System



Online Flight Check-in System



t1

t2

TIME

seat: 12F (220)

seat: 12F (220)

a

XDR

LWW

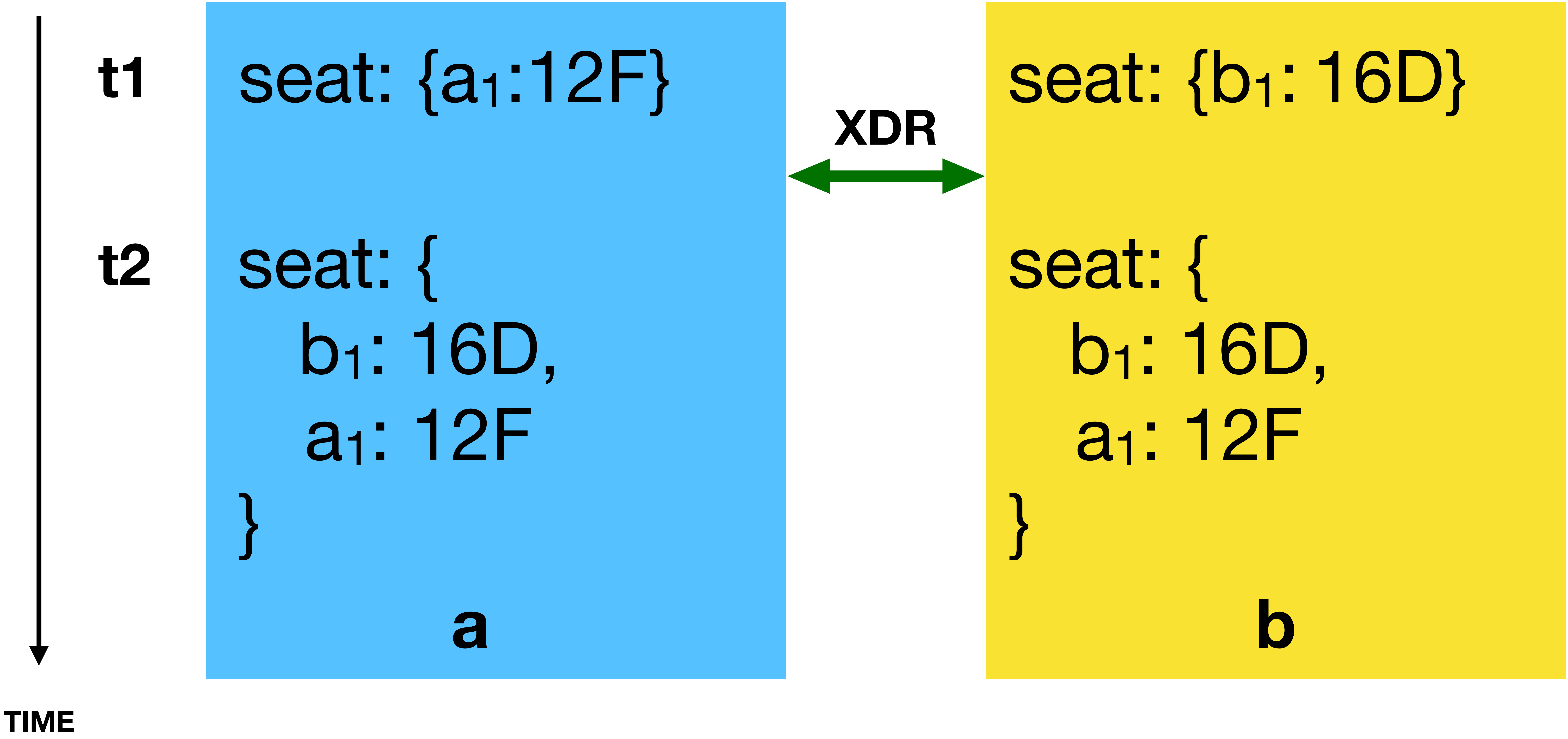
seat: 16D (150)

seat: 12F (220)

b

M(a, b) for LWW = MAX(a, b)

Online Flight Check-in System



t1

t2

TIME

seat: {a_1:12F}

seat: {
b_1: 16D,
a_1: 12F
}

a

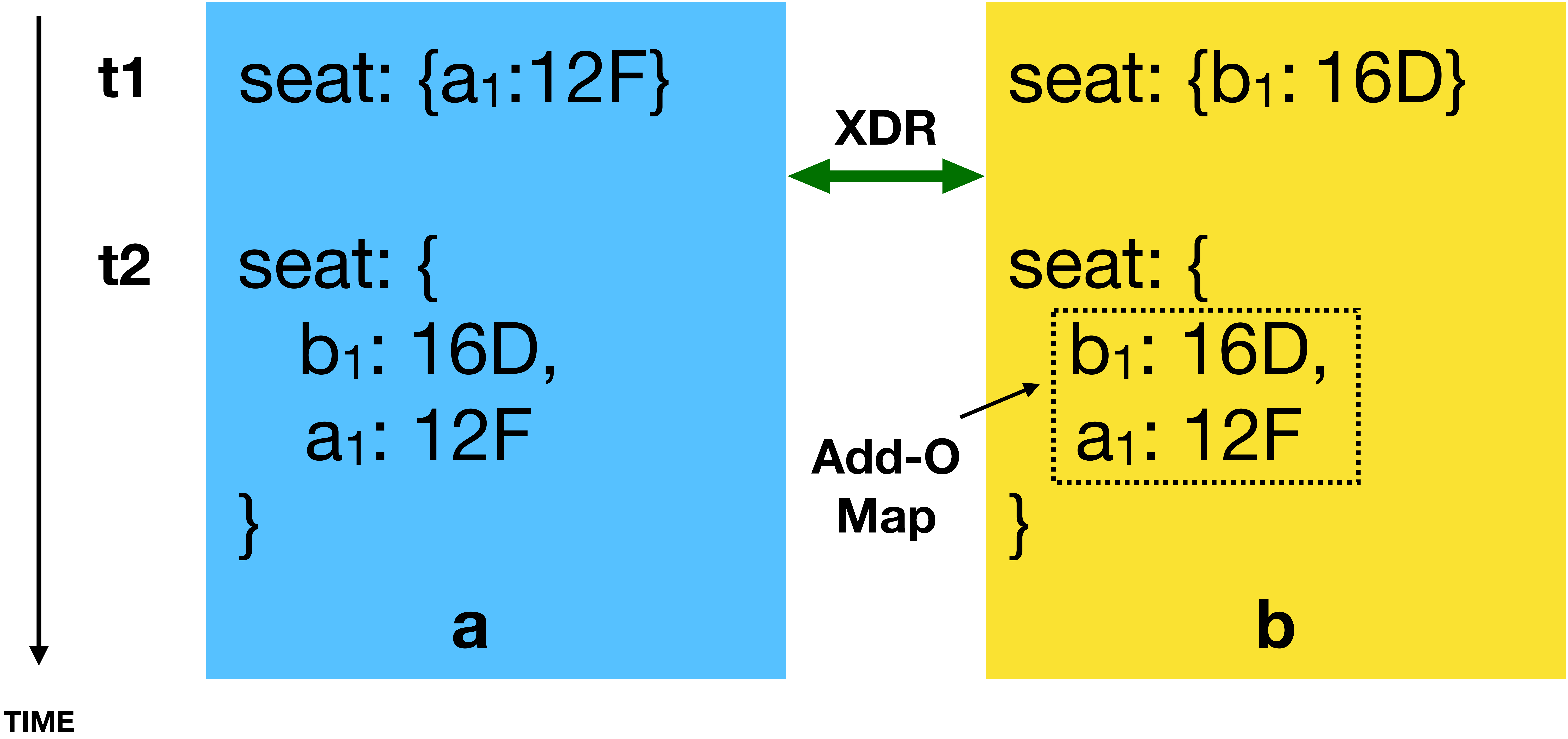
XDR

seat: {b_1: 16D}

seat: {
b_1: 16D,
a_1: 12F
}

b

Online Flight Check-in System



t1

seat: {a_1:12F}

XDR

seat: {b_1: 16D}

t2

seat: {
b_1: 16D,
a_1: 12F
}

Add-O Map

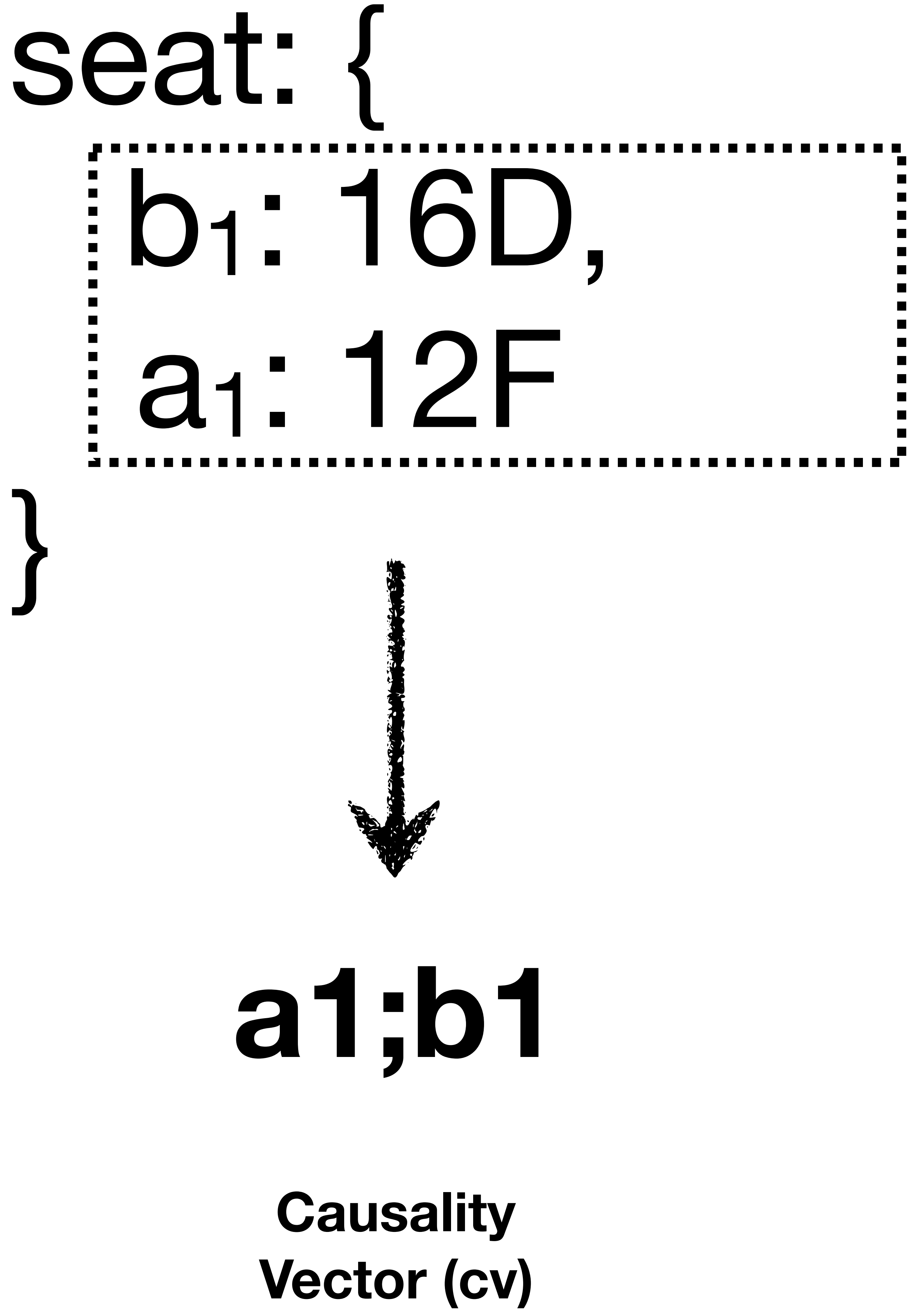
seat: {
b_1: 16D,
a_1: 12F
}

a

b

TIME

Causality



seat: {

b_1: 16D,

a_1: 12F

}

a1;b1

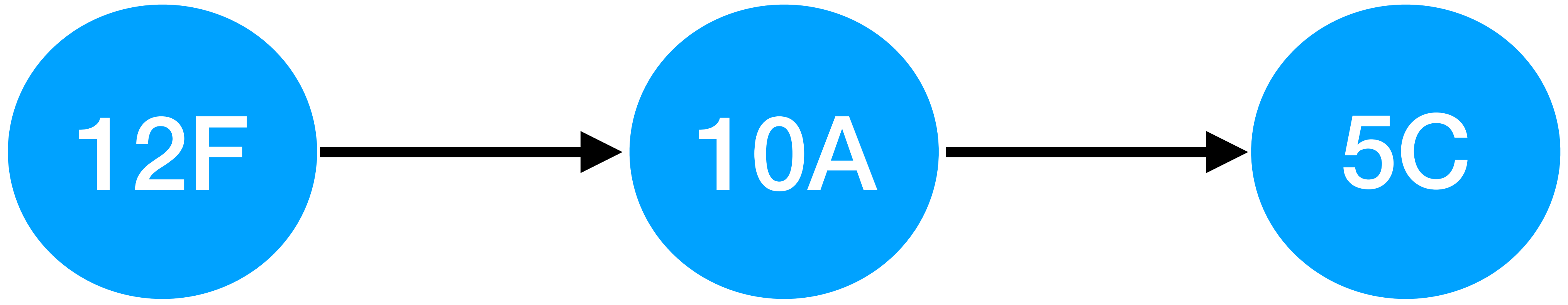
Causality
Vector (cv)

Causality

seat: {
b_1: 16D,
a_1: 12F
}

a1;b1

Causality Vector (cv)



12F is causal to 10A - we can drop 12F

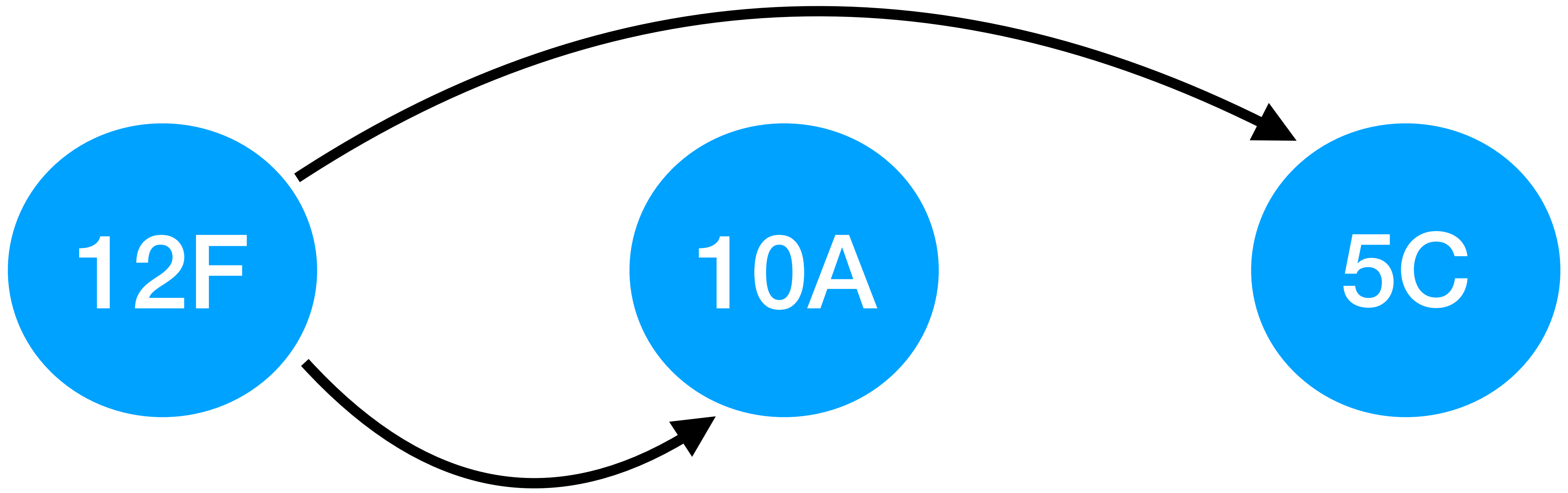
10A is causal to 5C - we can drop 10A

Causality

seat: {
  b_1: 16D,
  a_1: 12F
}

a1;b1

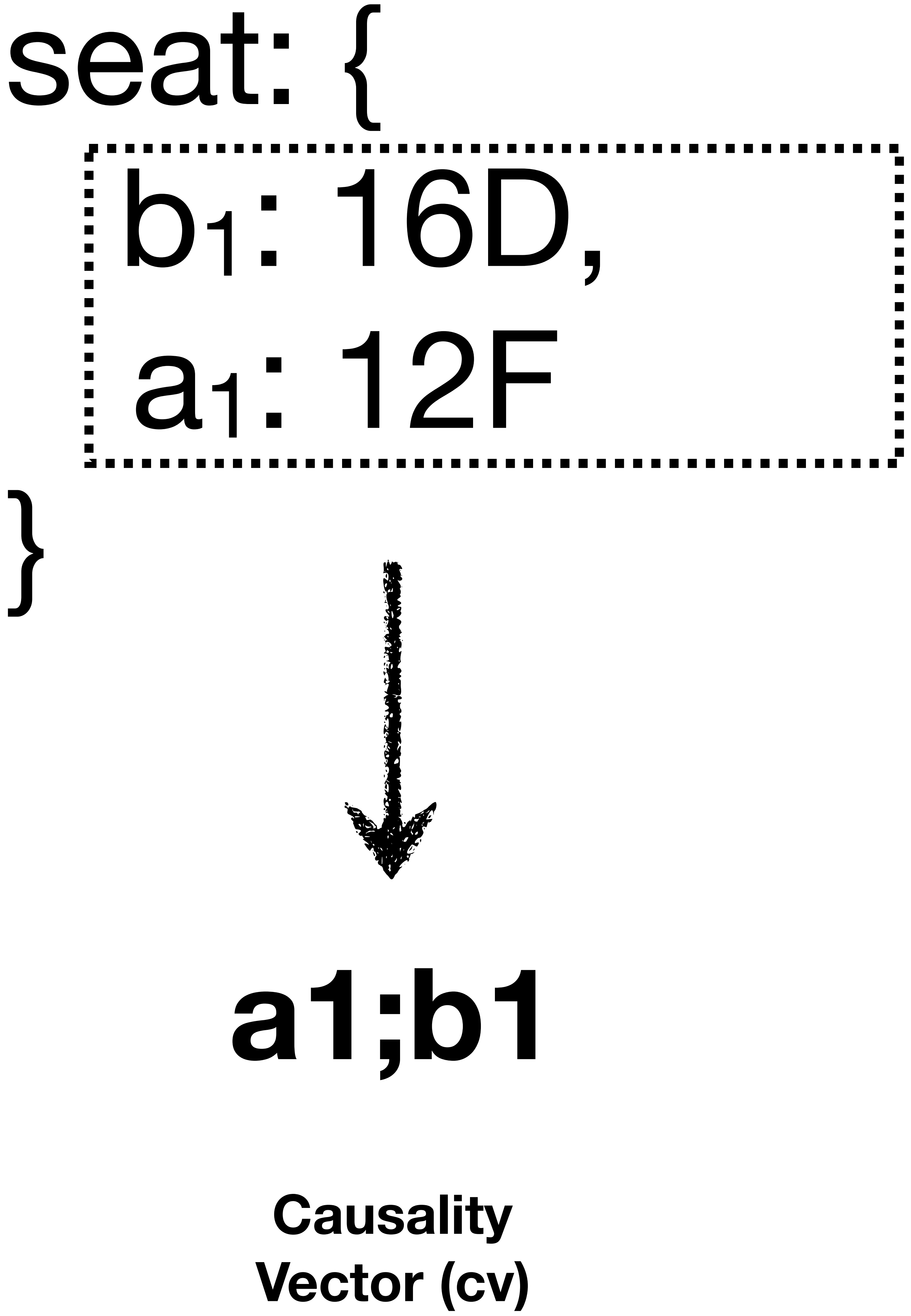
Causality Vector (cv)



12F is causal to 10A - we can drop 12F

10A is **NOT** causal to 5C - we can **NOT** drop 10A

Causality



seat: {

b_1: 16D,

a_1: 12F

}

a1;b1

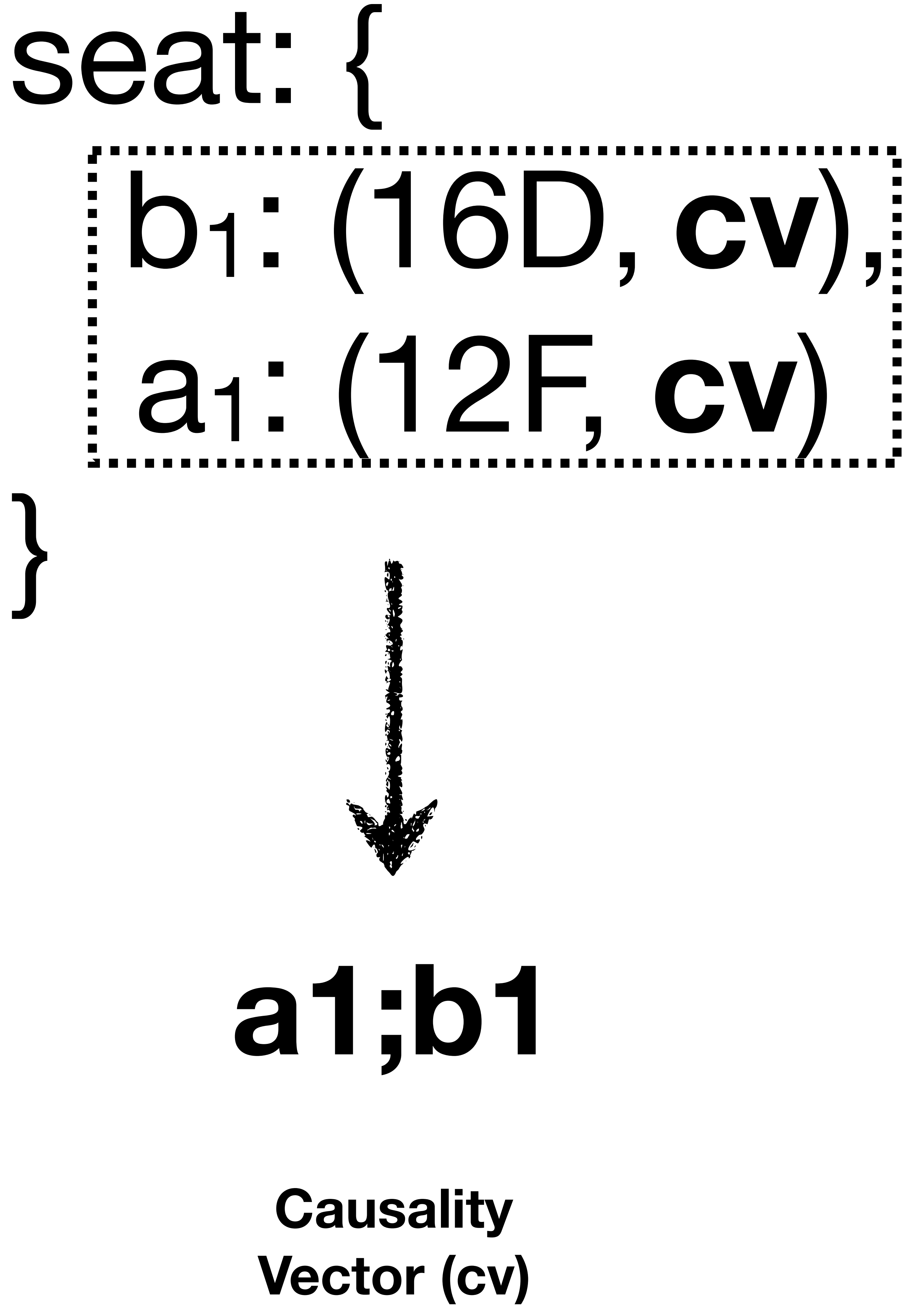
Causality Vector (cv)

Client Operations:

GET(key): value => GET(key): (value**, cv**)

PUT(key, value) => PUT(key, value**, cv**)

Causality



seat: {

b_1: (16D, **cv**),

a_1: (12F, **cv**)

}

a1;b1

Causality Vector (cv)

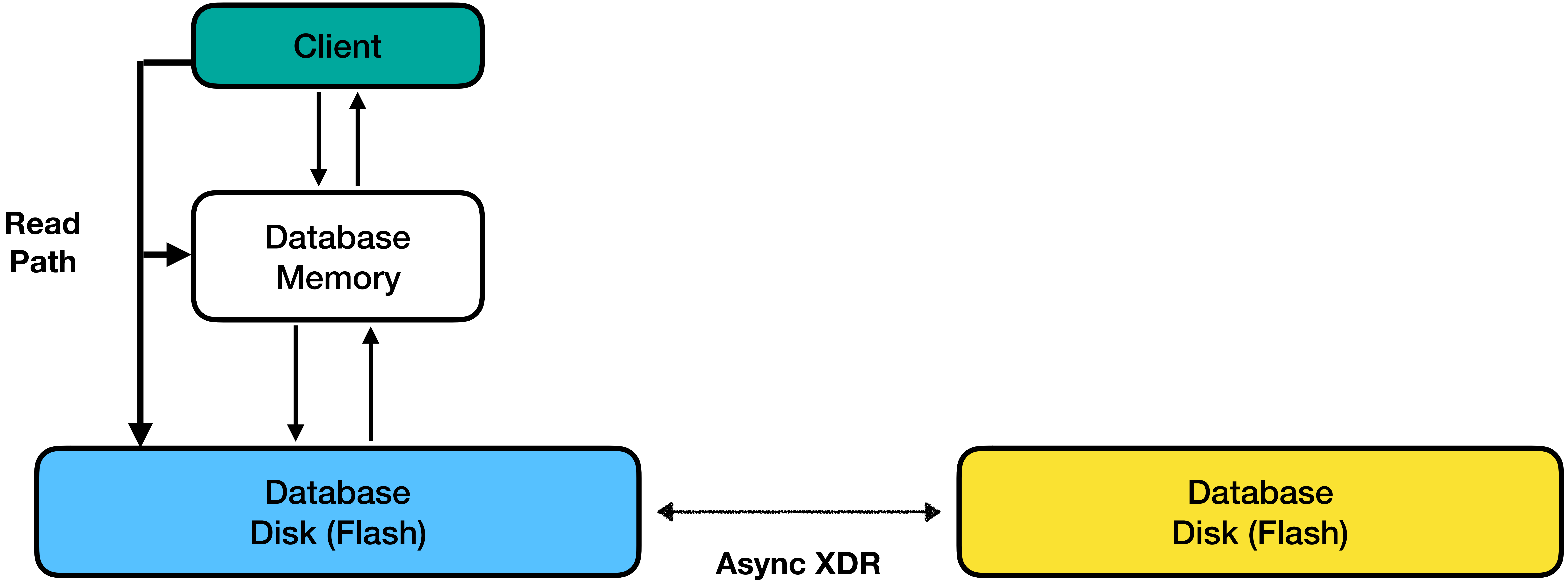
Client Operations:

GET(key): value => GET(key): (value**, cv**)

PUT(key, value) => PUT(key, value**, cv**)

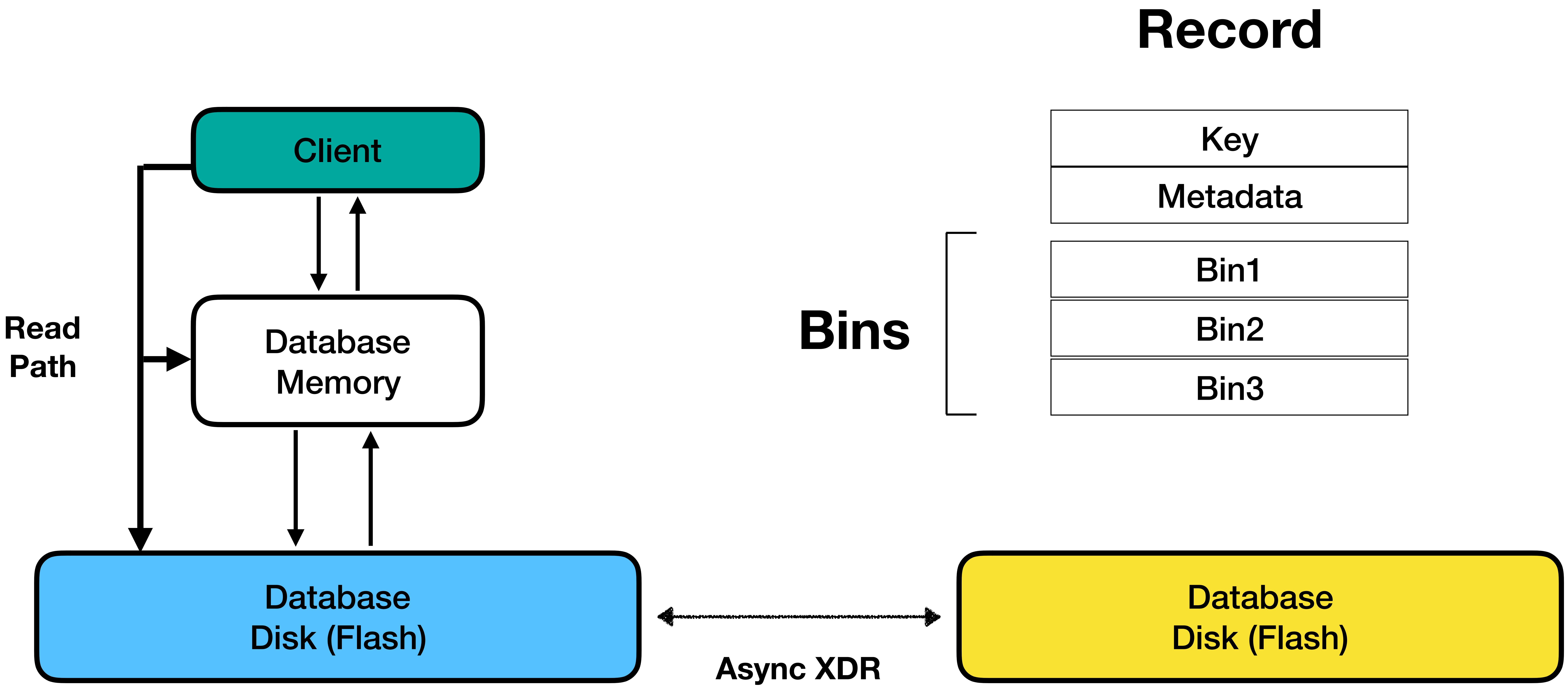
Aerospike Datastore

AEROSPIKE



Aerospike Datastore

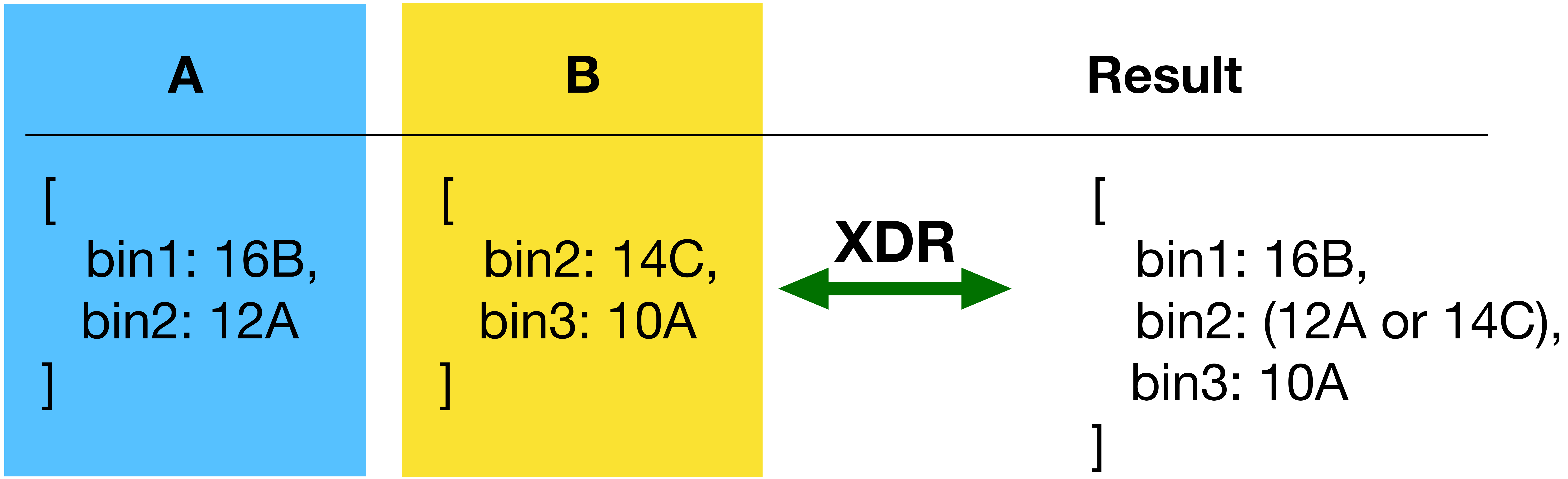
AEROSPIKE



Aerospike Datastore

1 User-Defined Functions

2



A

```
[
   bin1: 16B,
   bin2: 12A
]
```

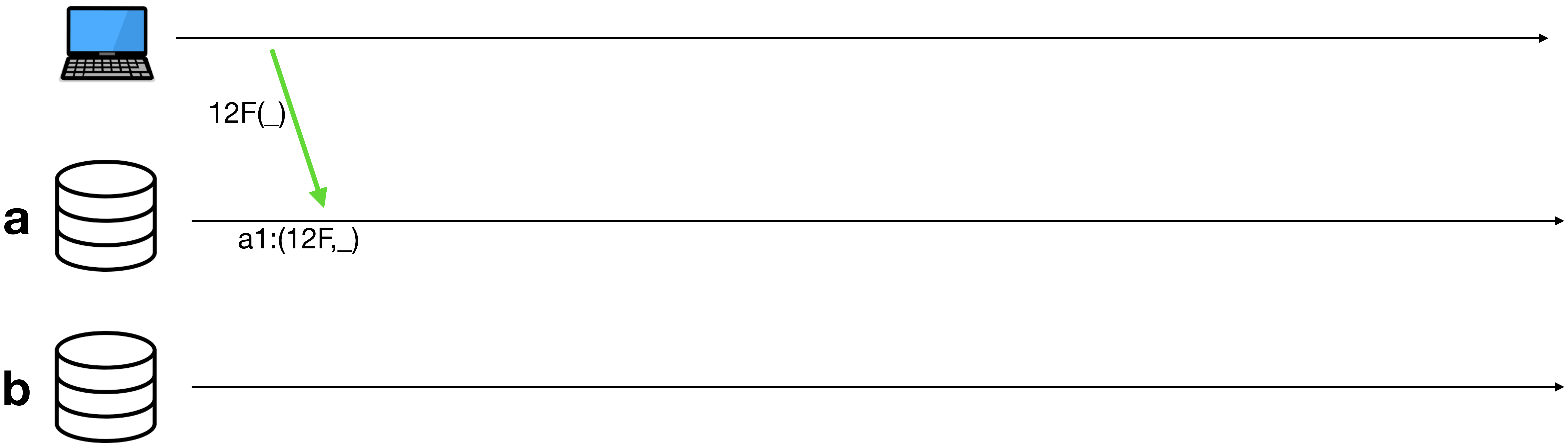
B

```
[
   bin2: 14C,
   bin3: 10A
]
```

XDR

Result

```
[
   bin1: 16B,
   bin2: (12A or 14C),
   bin3: 10A
]
```

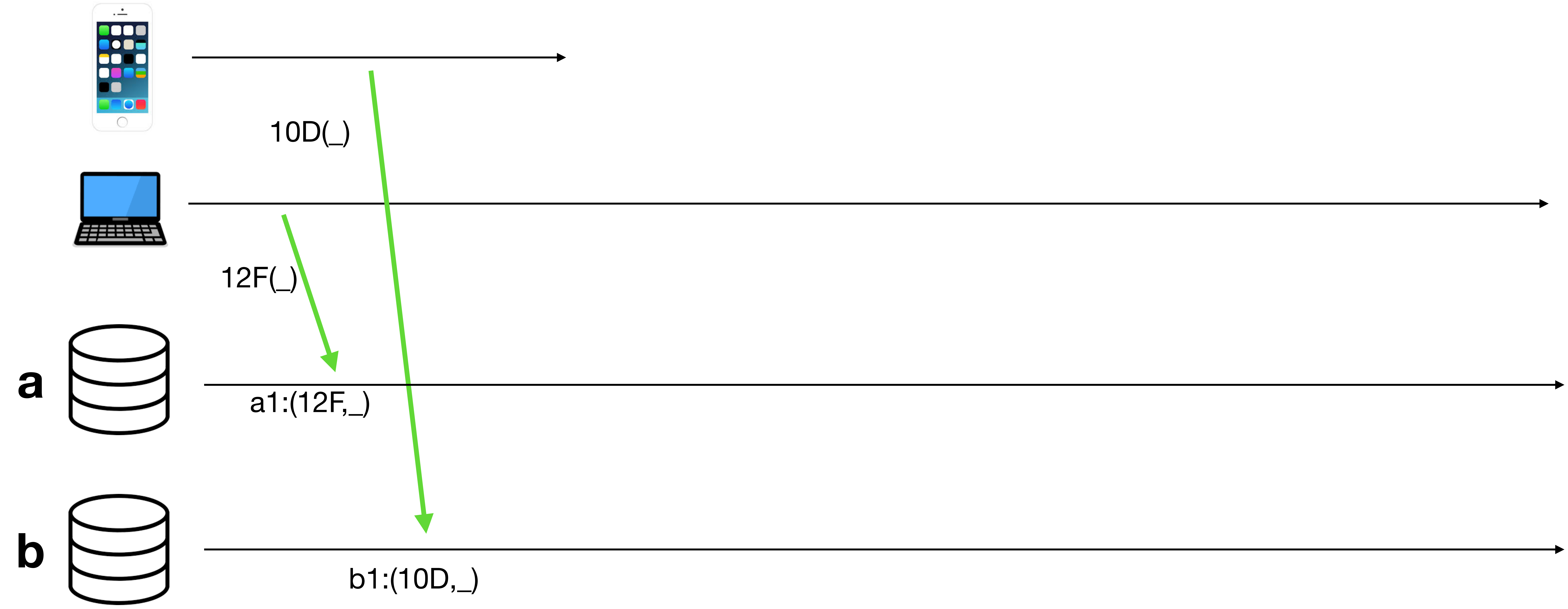


Bins	1
a1	(12F, _)

a

Bins	1

b

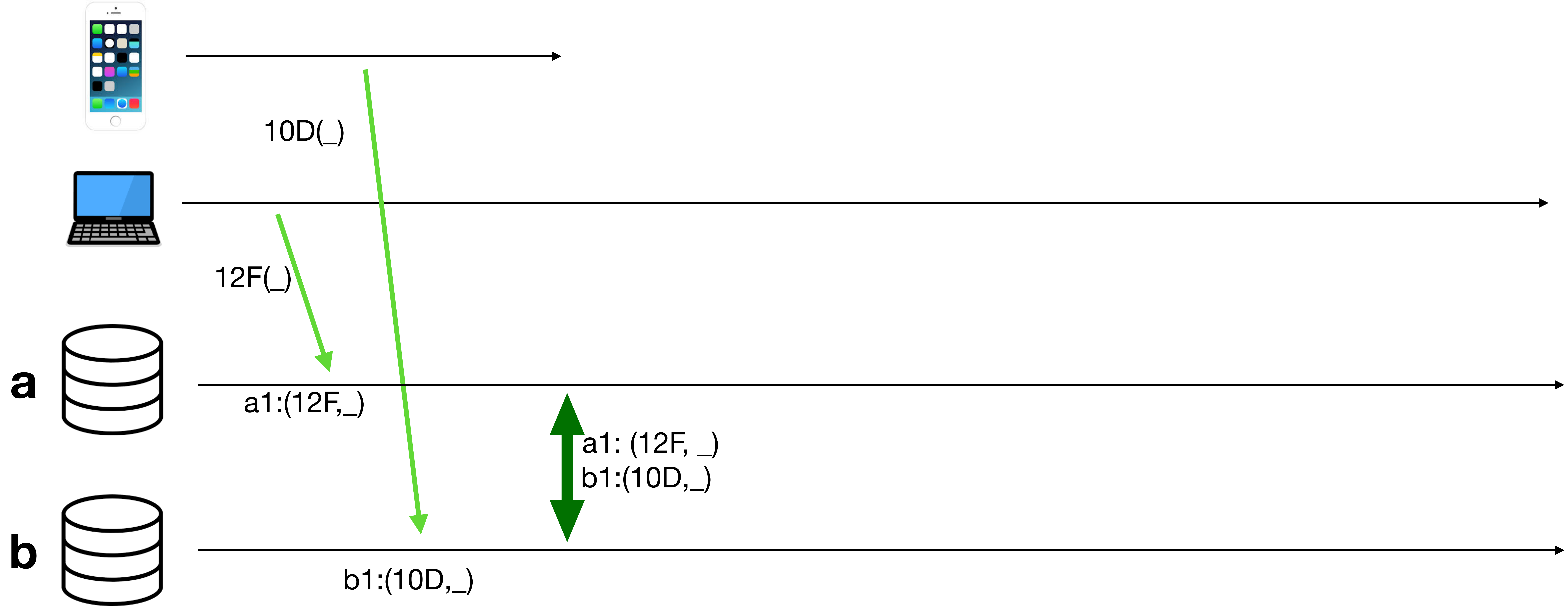


Bins	1	2
a1	(12F, _)	(12F, _)

a

Bins	1	2
b1		(10D, _)

b

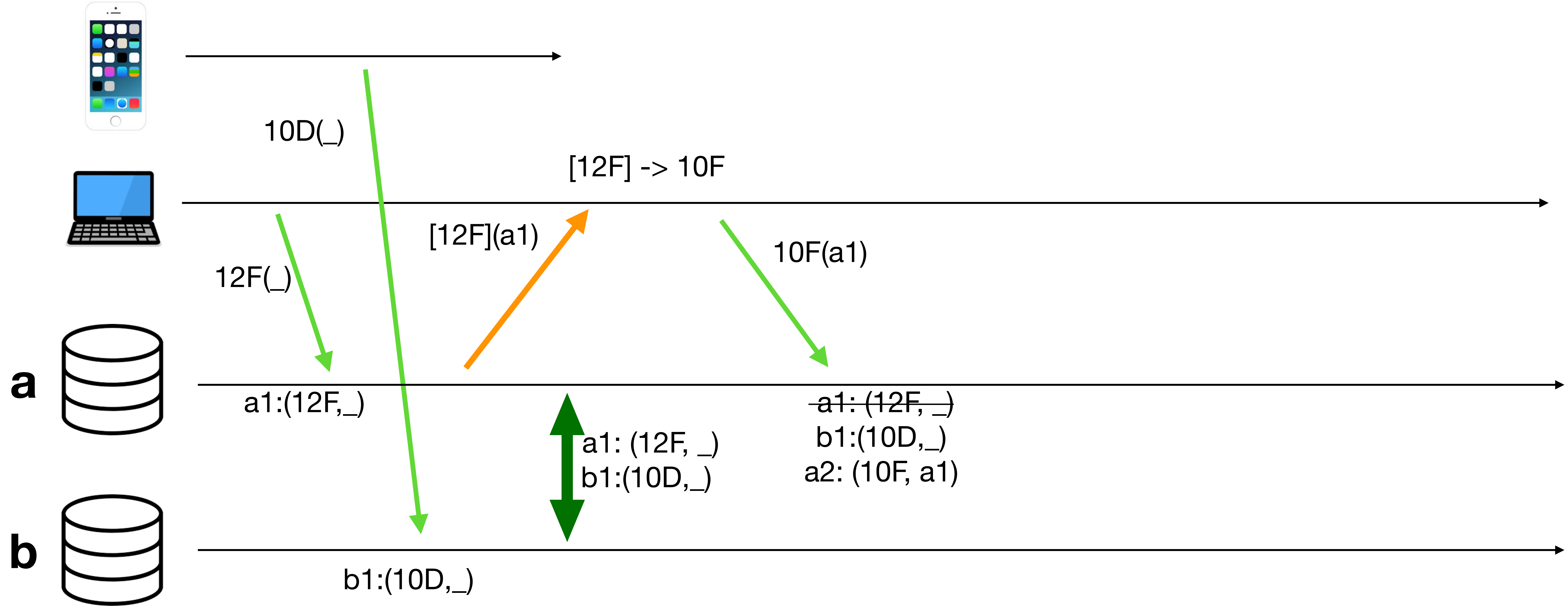


Bins	1	2	3
a1	(12F, _)	(12F, _)	(12F, _)
b1			(10D, _)

a

Bins	1	2	3
a1			(12F, _)
b1		(10D, _)	(10D, _)

b

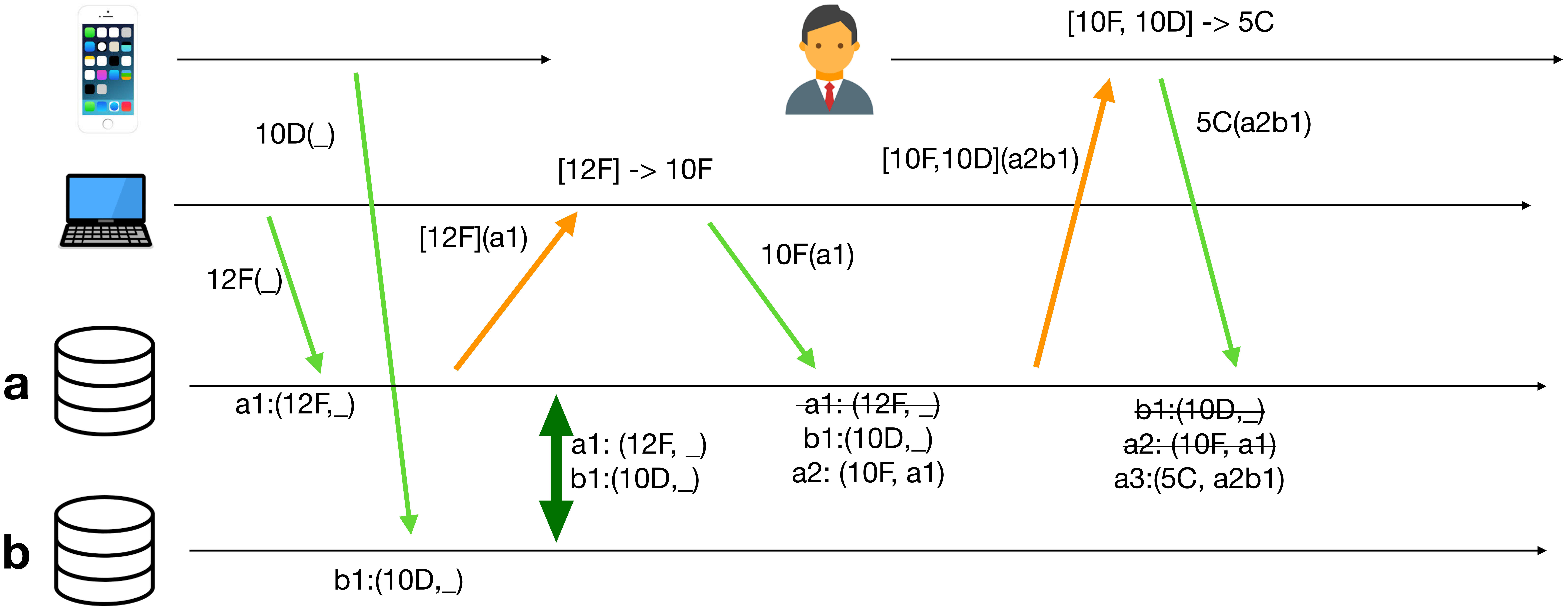


Bins	1	2	3	4
a1	(12F, _)	(12F, _)	(12F, _)	(12F, _)
b1			(10D, _)	(10D, _)
a2				(10F, a1)

a

Bins	1	2	3	4
a1			(12F, _)	(12F, _)
b1		(10D, _)	(10D, _)	(10D, _)

b

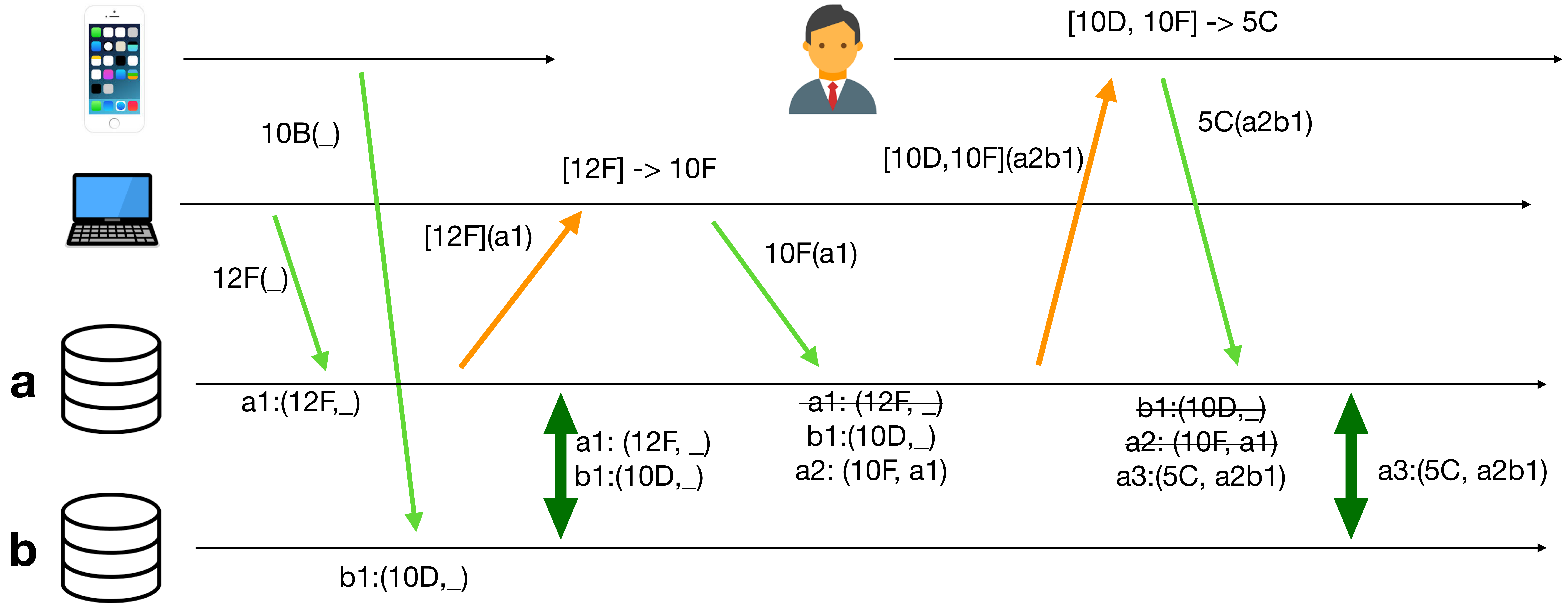


Bins	1	2	3	4	5
a1	(12F, _)	(12F, _)	(12F, _)	(12F, _)	(12F, _)
b1			(10D, _)	(10D, _)	(10D, _)
a2				(10F, a1)	(10F, a1)
a3					(5C, a2b1)

a

Bins	1	2	3	4	5
a1			(12F, _)	(12F, _)	(12F, _)
b1		(10D, _)	(10D, _)	(10D, _)	(10D, _)

b



[10D, 10F] -> 5C

10B(_)

5C(a2b1)

[10D,10F](a2b1)

[12F] -> 10F

[12F](a1)

10F(a1)

12F(_)

a

a1:(12F,_)

a1: (12F, _)
b1:(10D,_)

~~a1: (12F, _)~~
b1:(10D,_)
a2: (10F, a1)

~~b1:(10D,_)~~
~~a2: (10F, a1)~~
a3:(5C, a2b1)

a3:(5C, a2b1)

b

b1:(10D,_)

Bins	1	2	3	4	5	6
a1	(12F, _)	(12F, _)	(12F, _)	(12F, _)	(12F, _)	(12F, _)
b1			(10D, _)	(10D, _)	(10D, _)	(10D, _)
a2				(10F, a1)	(10F, a1)	(10F, a1)
a3					(5C, a2b1)	(5C, a2b1)

a

Bins	1	2	3	4	5	6
a1			(12F, _)	(12F, _)	(12F, _)	(12F, _)
b1		(10D, _)	(10D, _)	(10D, _)	(10D, _)	(10D, _)
a3						(5C, a2b1)

b

Learnings

- CRDTs allowed us to achieve convergent **predictable** state of our data

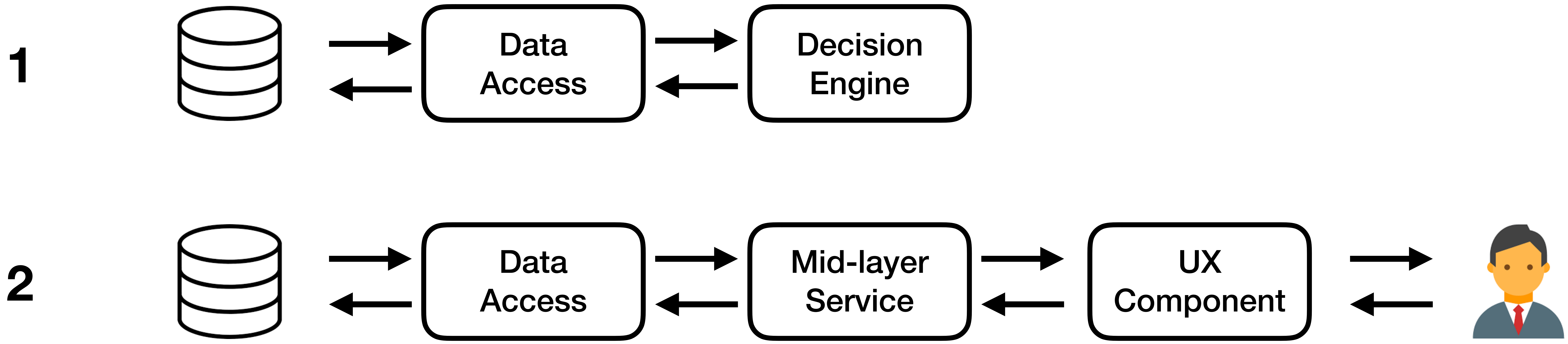
Learnings

- CRDTs allowed us to achieve convergent **predictable** state of our data
- Education about right trade-off between **Consistency** and **Correctness**

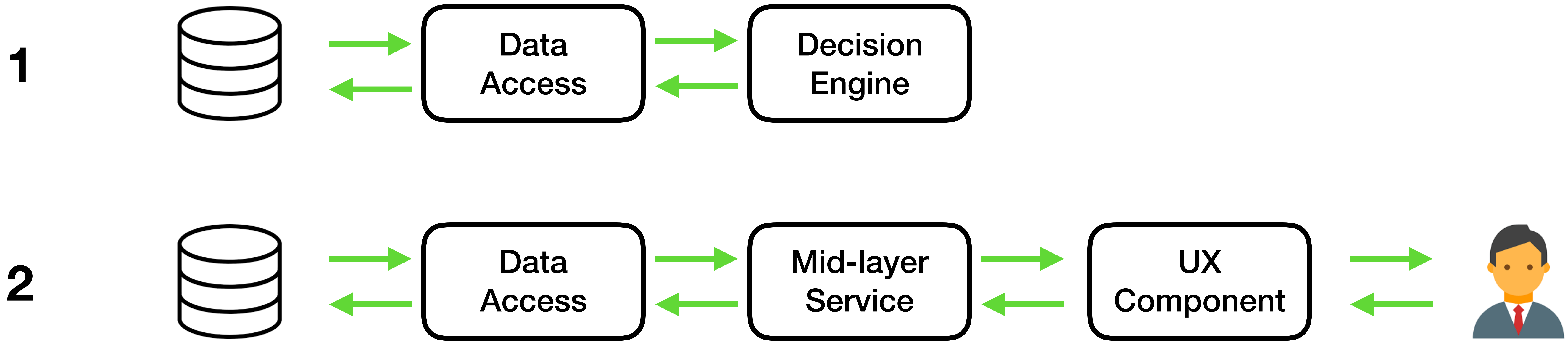
Learnings

- CRDTs allowed us to achieve convergent **predictable** state of our data
- Education about right trade-off between **Consistency** and **Correctness**
- Do not **underestimate** concurrent data access

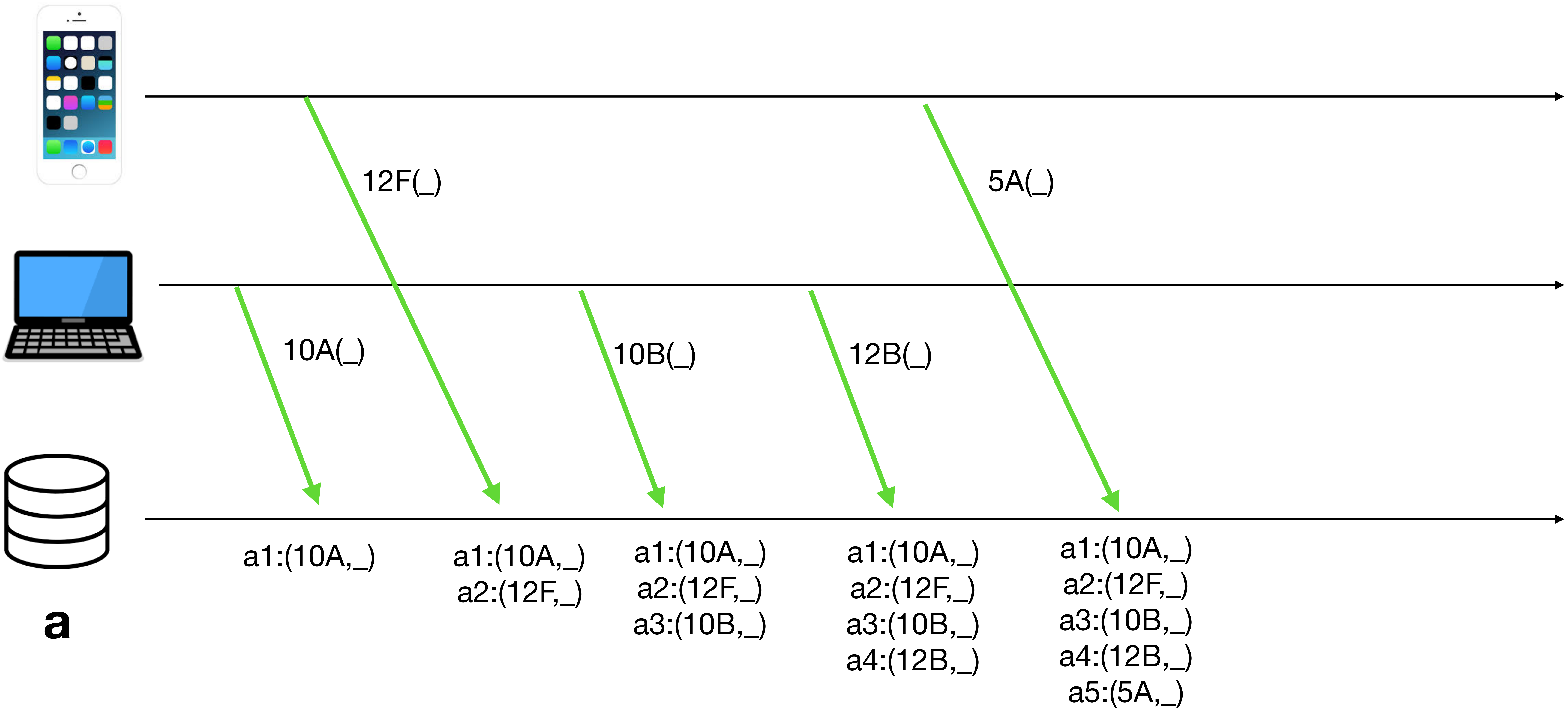
Caveat #1: CV Propagation



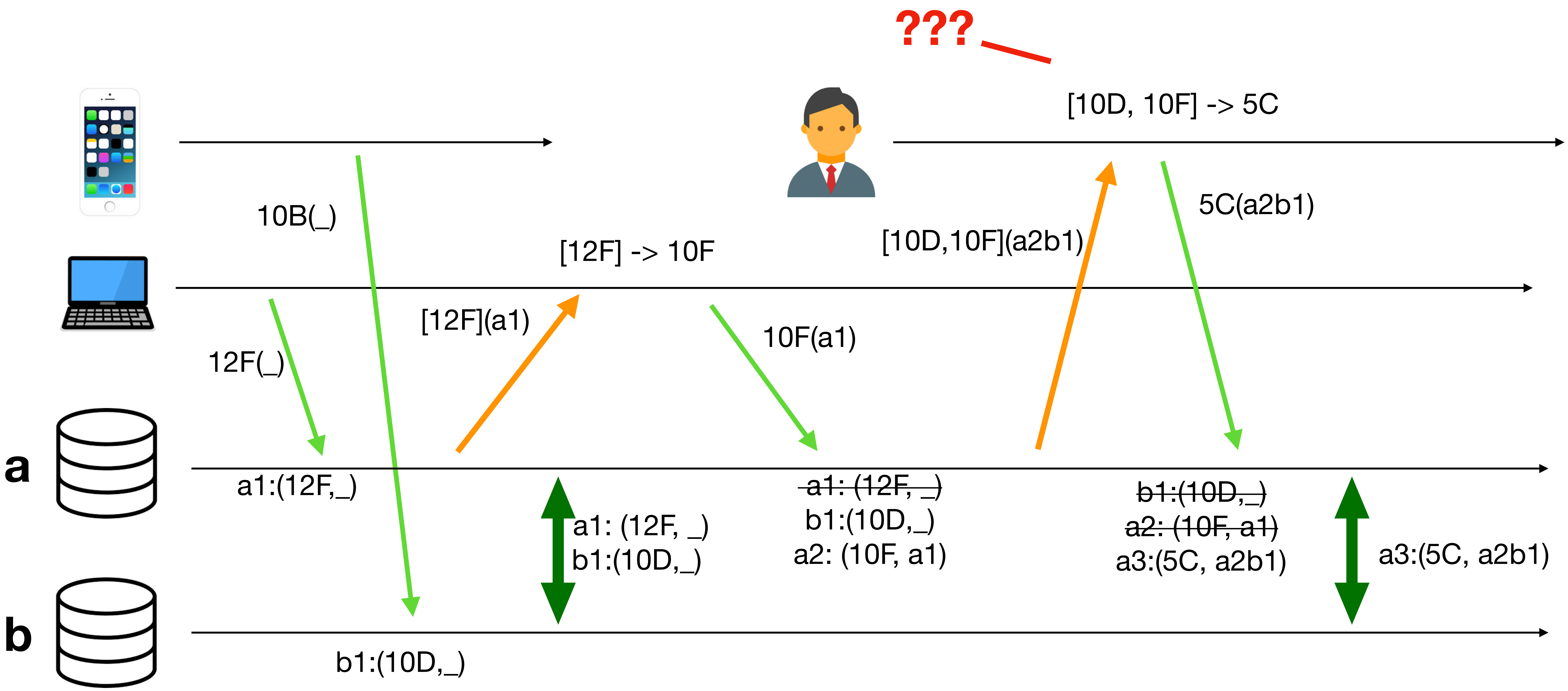
Caveat #1: CV Propagation



Caveat #2: Siblings Explosion



Caveat #3: Wait, Siblings ?



Thanks!