Distributed Systems

Spring Semester 2020

Lecture 16: Bayou (Eventual Consistency)

John Liagouris
liagos@bu.edu

Eventual Consistency

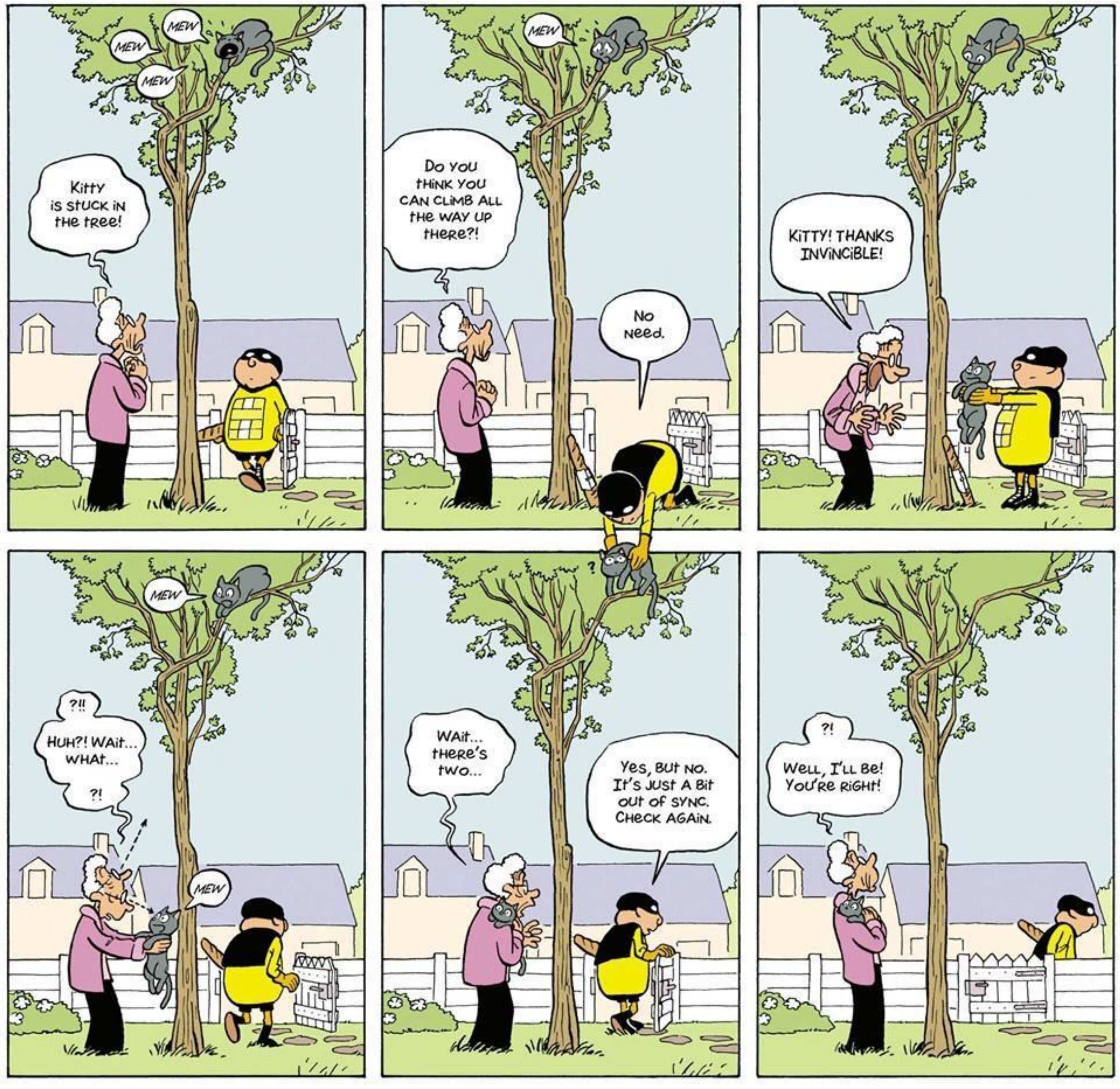
Eventual Consistency — used a lot today — smart phone syncing, Dropbox, Cloud App Data Stores (Amazon Dynamo - Lecture 25)

- The Good
  - Fast Local reads and writes — done as soon as local op occurs — really fast …
  - Disconnected operation is implicitly supported (on the plane, network outage, on the mountain top)
  - Ad-hoc synchronization

Eventual Consistency

- The BAD
  - Replicas diverge — CONFLICTS
    - git just can't fix a conflict to the same line of code
    - Eventual consistency conflicts are going to happen — hard to automatically resolve them
  - BAYOU — sophisticated scheme for dealing with conflicts

Eventual Consistency



BAYOU

- Research focusing on the "future"
  - Mobile computing — PDAs with intermittent wireless connectivity
  - No ubiquitous cellular data services
  - How to keep devices useful despite not having connectivity
- Unlike prior systems targets the development of first class Eventual Consistency applications

Bayou

1. Collection of hosts
2. Clients and servers distributed on hosts
3. Servers only do pair-wise communications to eventually reach consistency — anti-entropy

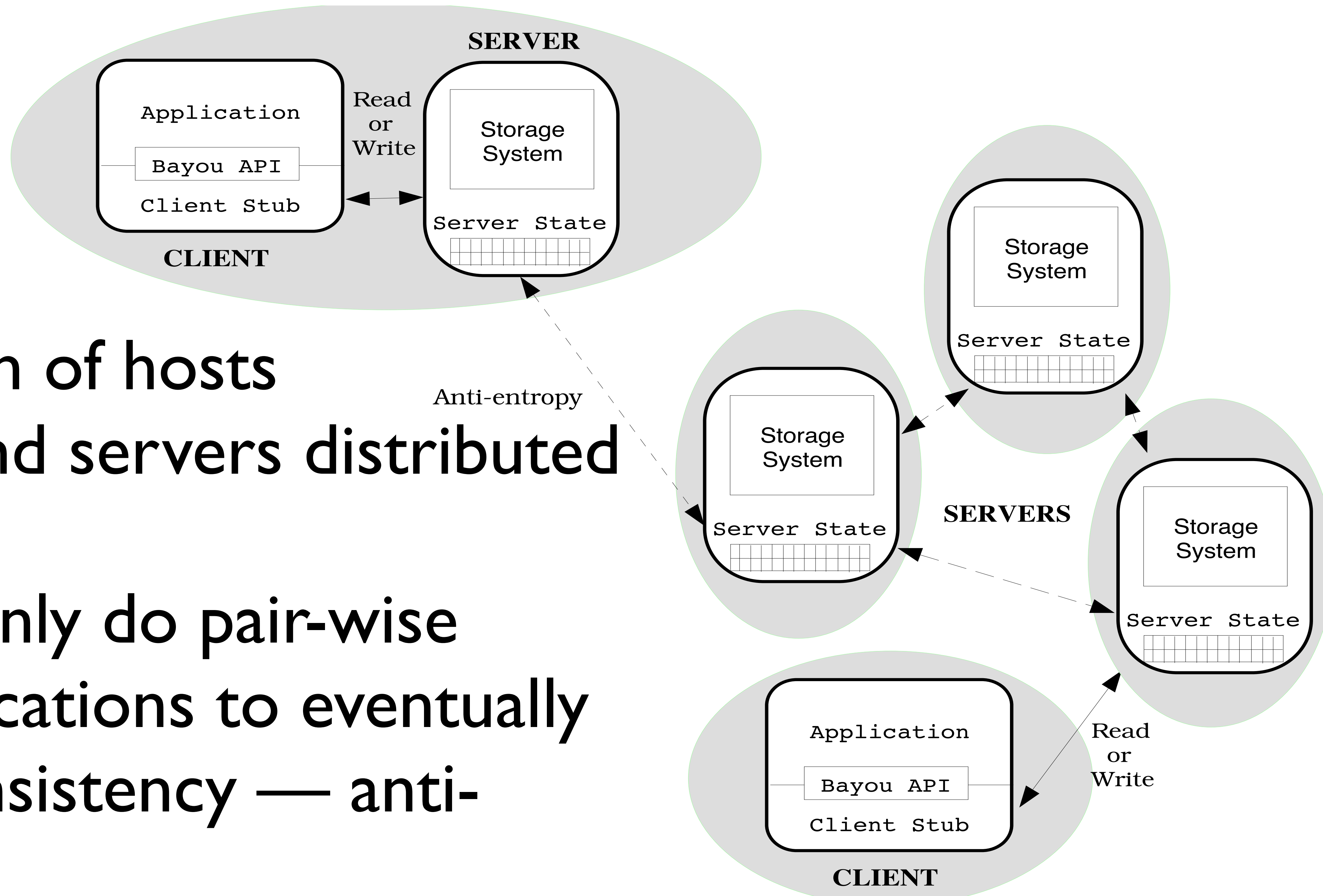


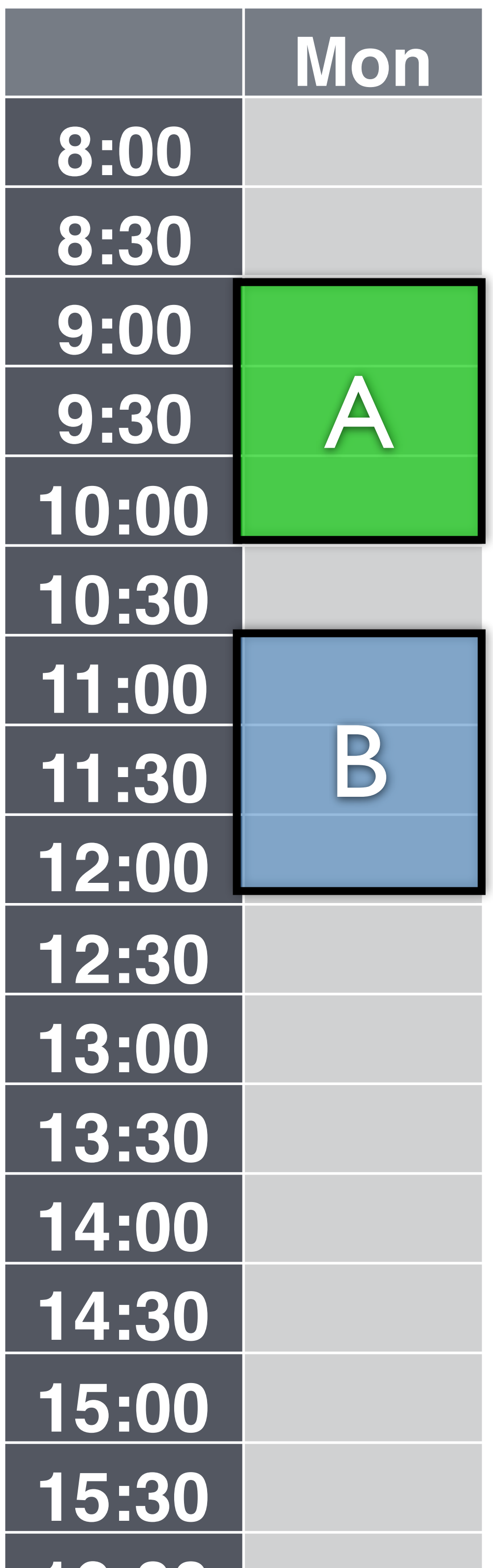
Figure 1. Bayou System Model

MEETING ROOM APP

	Mon
8:00	
8:30	
9:00	
9:30	
10:00	
10:30	
11:00	
11:30	
12:00	
12:30	
13:00	
13:30	
14:00	
14:30	
15:00	
15:30	

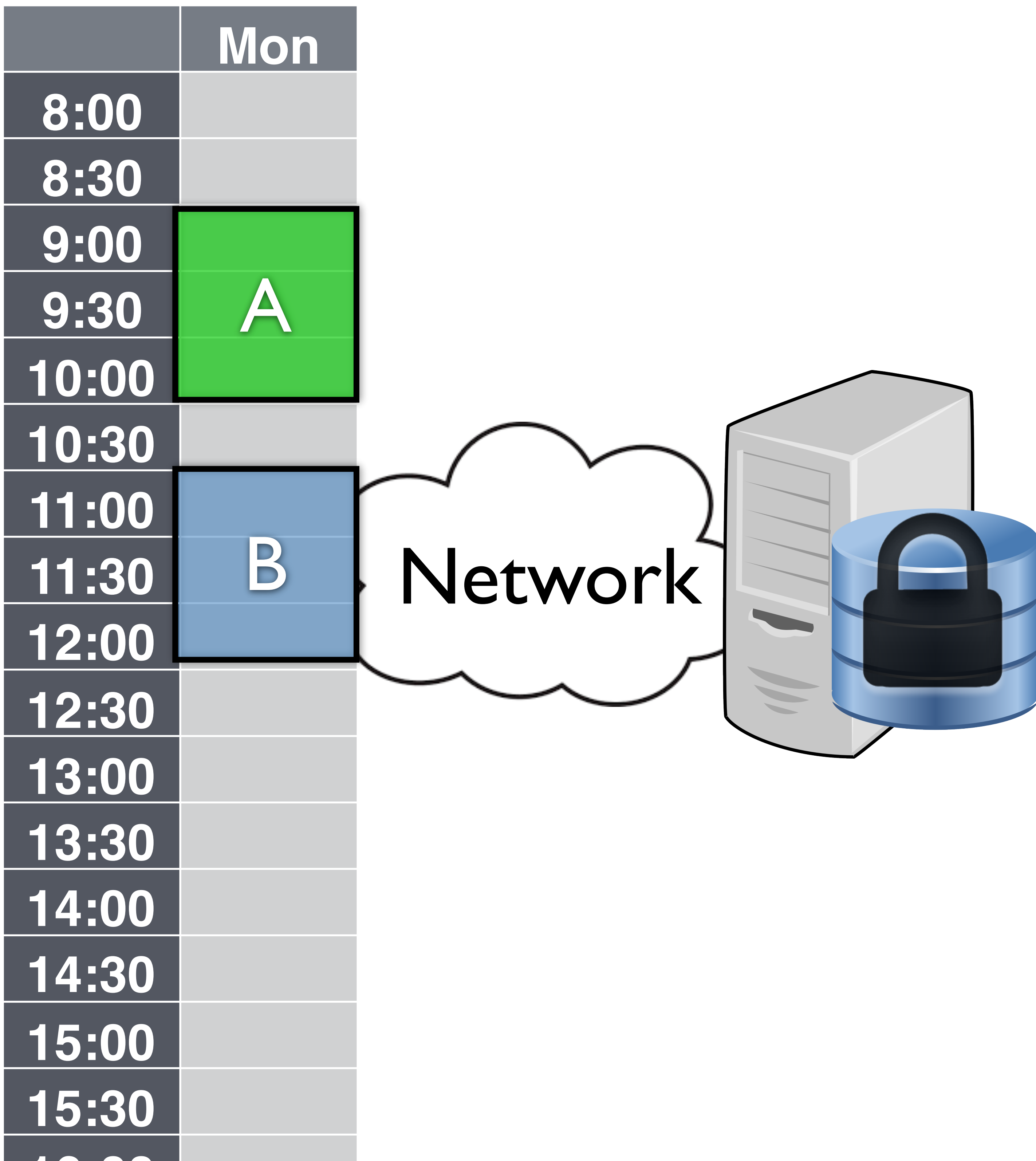
- GUI based room scheduling app

MEETING ROOM APP



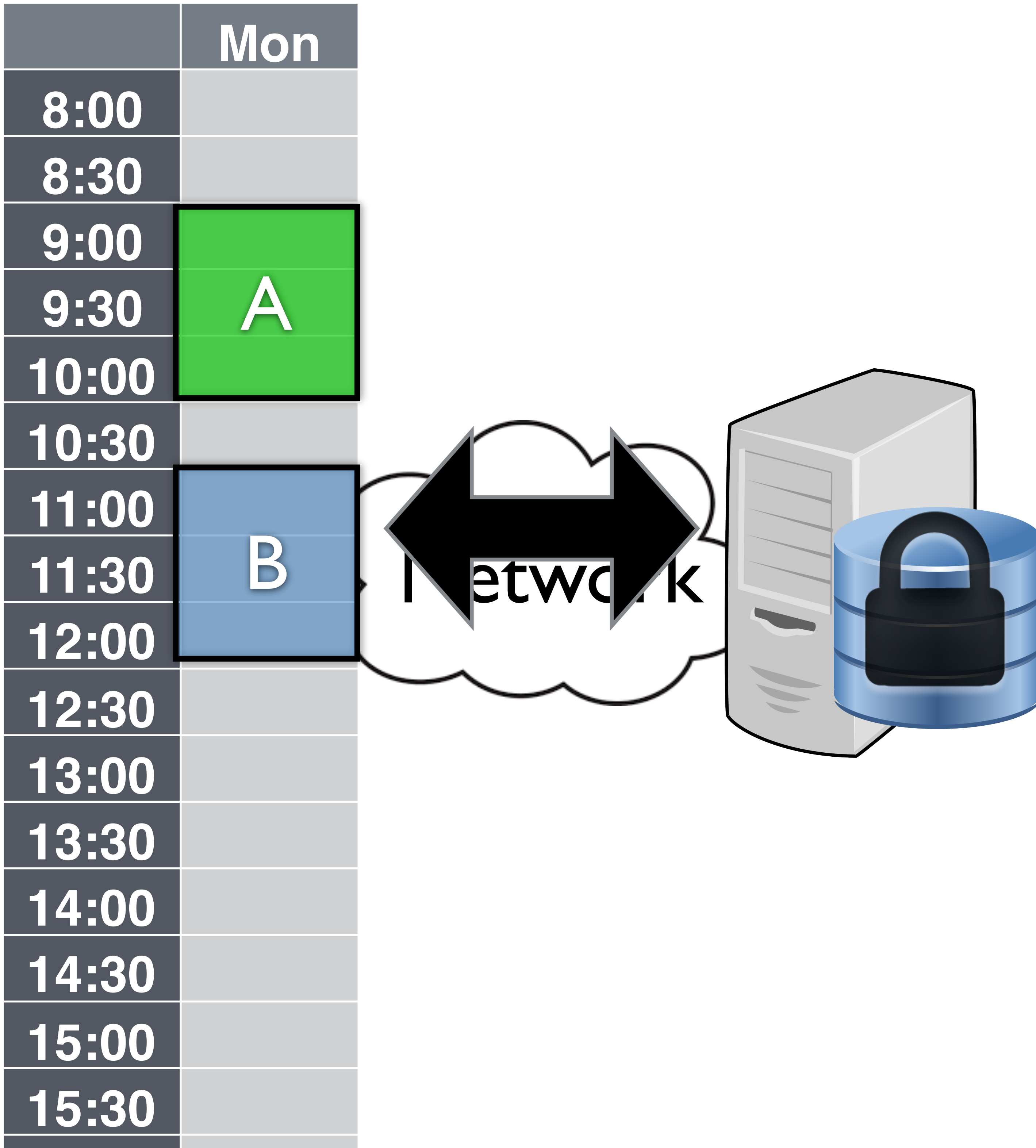
- GUI based room scheduling app
- Users book meetings by visually inspecting and reserving time slots

MEETING ROOM APP



- GUI based room scheduling app
- Users book meetings by visually inspecting and reserving time slots
- Obvious approach talk to a central booking server

MEETING ROOM APP

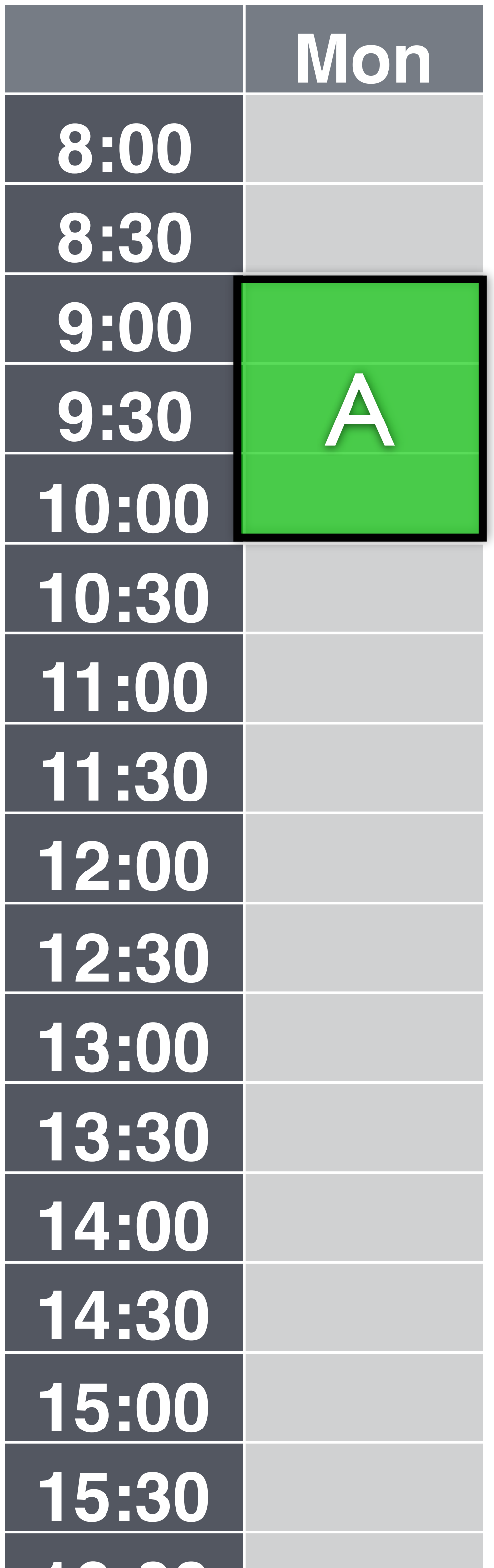


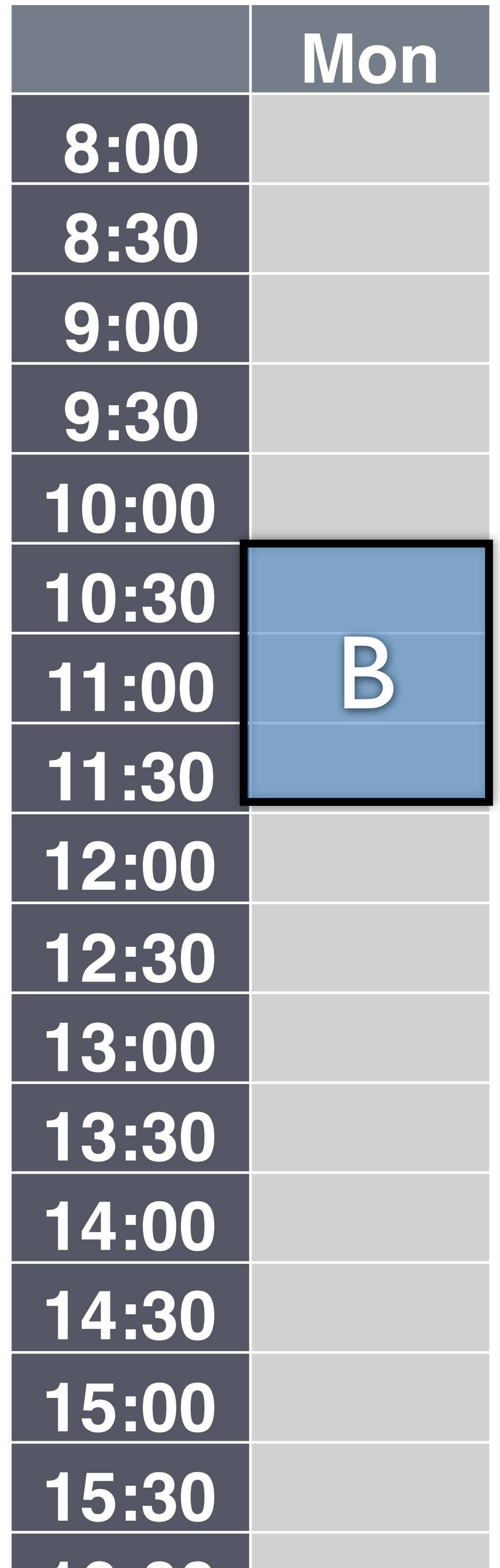
- GUI based room scheduling app
- Users book meetings by visually inspecting and reserving time slots
- Obvious approach talk to a single central booking server
- all operations require network communications
- Server is single point of authority

BUT

- Could they come up with a decentralized scheme?
- No need for connectivity
- No need to server to be up
- My local devices could be consistent by themselves
- Reusable infrastructure for eventual consistency apps

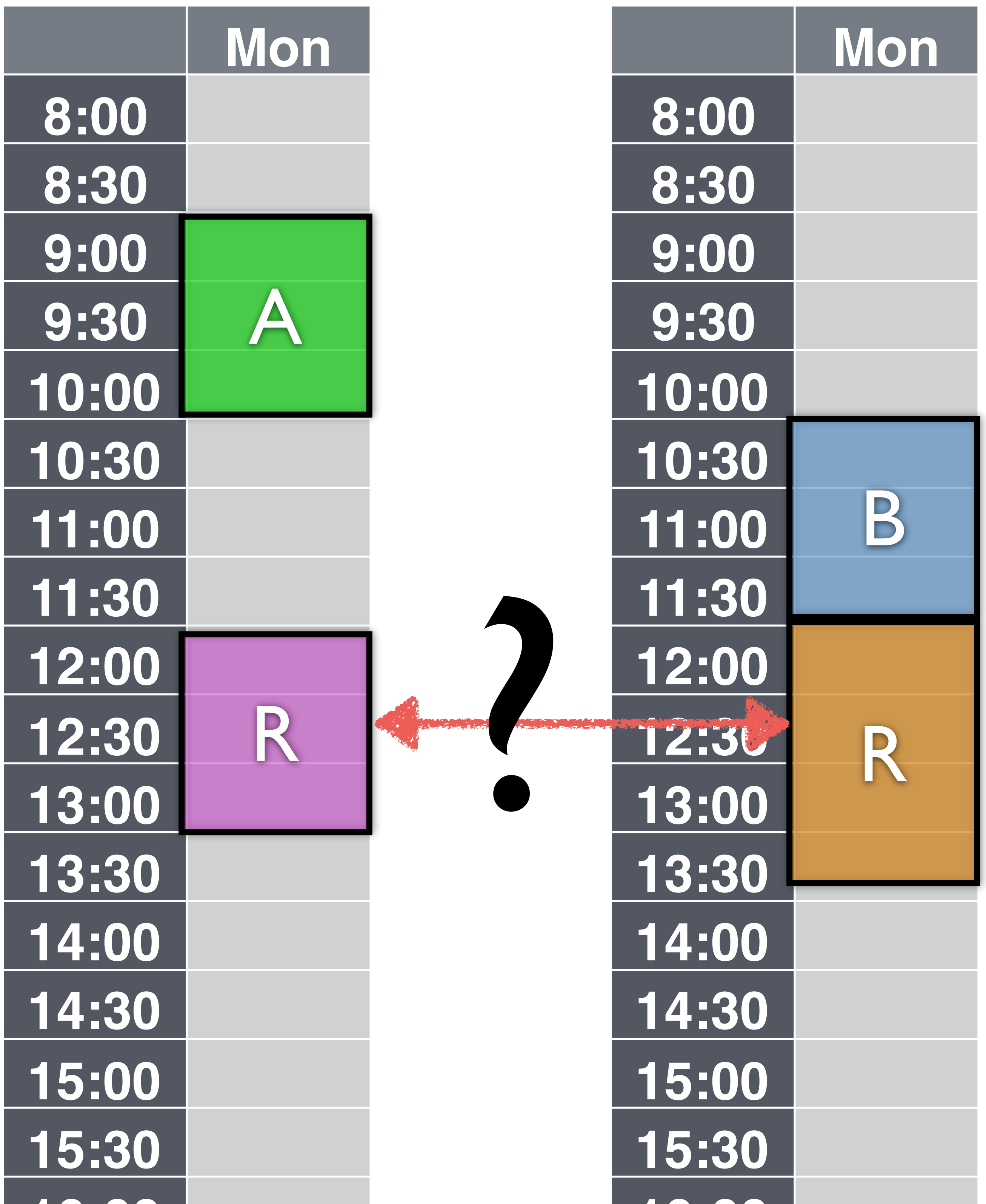
Straw man 1: DB Merging





- App can be treated as a simple DB
- Allow changes to occur
- when connected sync and merge DB's
- sort of works if no conflicting entries

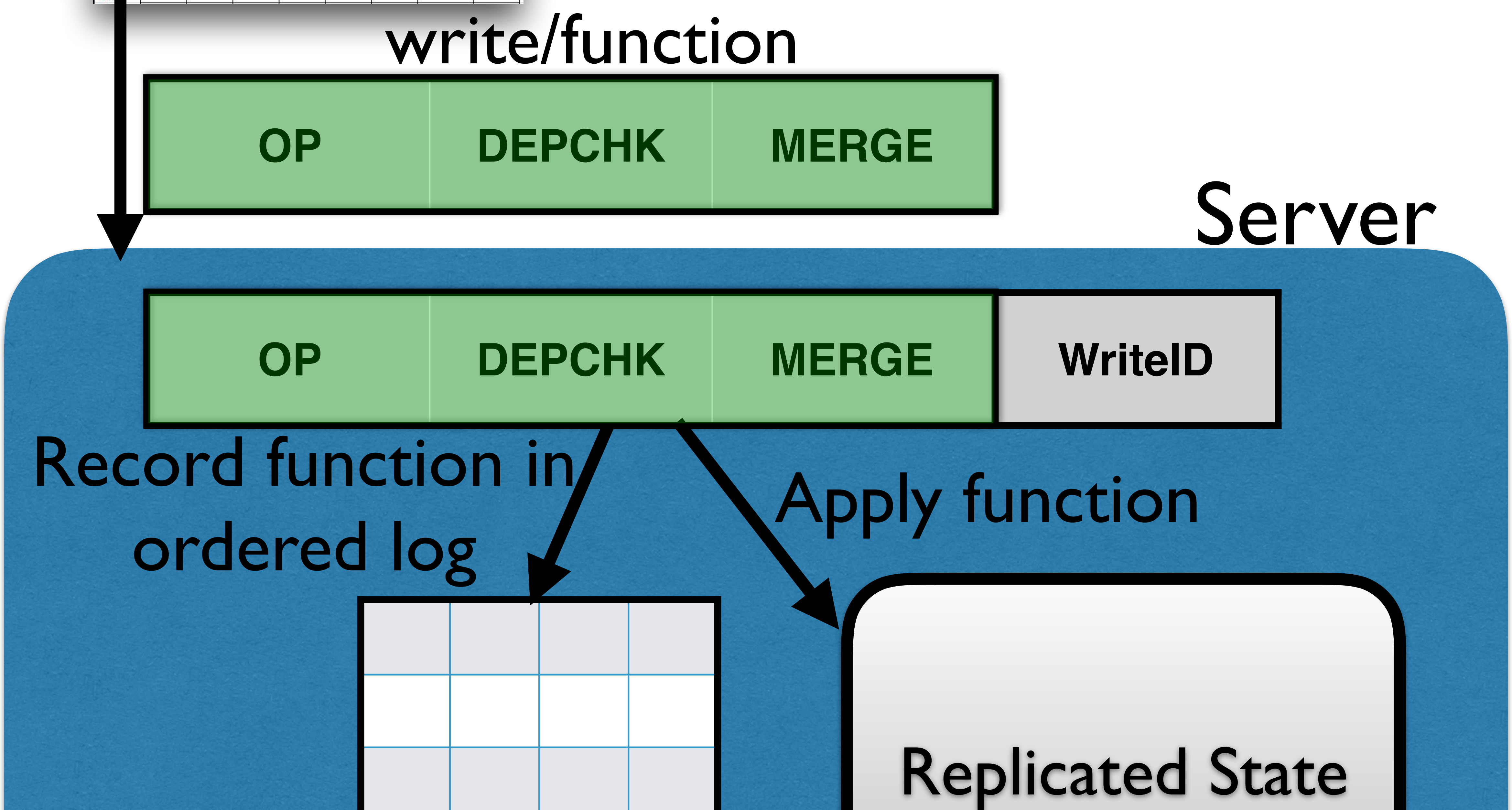
Straw man 1: DB Merging



- Best we could do is keep both copies of DB
- Flag conflict in DB
- We have lost app semantics and user intent
- All we have are two 'final' version of DB

BAYOU

- DB is second Class
- First class is ordered log of "functions"



Writes are functions on DB STATE

```
Bayou_Write(
    update = {insert, Meetings, 12/18/95, 1:30pm, 60min, “Budget Meeting”},
    dependency_check = {
        query = “SELECT key FROM Meetings WHERE day = 12/18/95
            AND start < 2:30pm AND end > 1:30pm”,
        expected_result = EMPTY},
    mergeproc = {
        alternates = {{12/18/95, 3:00pm}, {12/19/95, 9:30am}};
        newupdate = {};
        FOREACH a IN alternates {
            # check if there would be a conflict
            IF (NOT EMPTY (
                SELECT key FROM Meetings WHERE day = a.date
                 AND start < a.time + 60min AND end > a.time))
                    CONTINUE;
            # no conflict, can schedule meeting at that time
            newupdate = {insert, Meetings, a.date, a.time, 60min, “Budget Meeting”};
            BREAK;
        }
        IF (newupdate = {})   # no alternate is acceptable
            newupdate = {insert, ErrorLog, 12/18/95, 1:30pm, 60min, “Budget Meeting”};
        RETURN newupdate;}
)
```

functions not values

- User’s don’t write these functions
- App construct functions based on client side UI
- BAYOU carries around functions and runs them as needed to update/re-updated DB as servers settle on log order
- View’s DB as resulting state of a sequence of function invocations — DB can always be recreated if needed from functions
- Sync exchanges functions not value of DB

functions not values

- Functions much richer than just a value — can express application and update specific conflict resolution behavior
  - Meet a 9 if room free else 10, else 11else 8
  - Functions can do reconciliation without user

Not Done Yet

- PDA A: Staff meeting at 9 or 10
- PDA B: Hiring meeting 9 or 10
- PDA X sync's with A and then B
  - Staff meeting @ 9, Hiring meeting @ 10
- PDA Y syncs with B and then A
  - Hiring meeting @ 9, Staff meeting @ 10

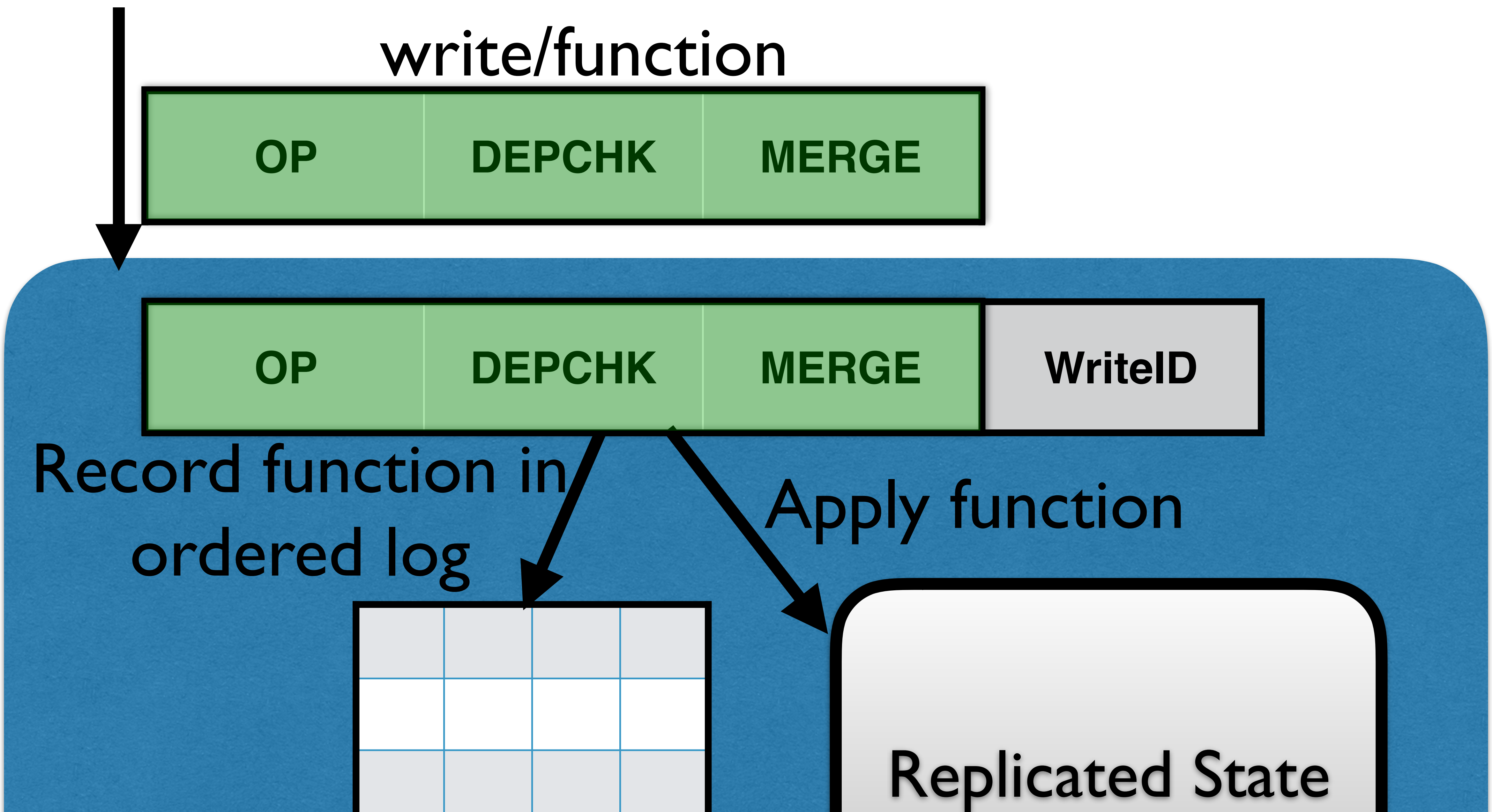
Will not eventually become consistent as it stands :-(

ORDER THE LOGS

- Order log updates on each node
- Sync ensures logs are the same on both nodes (eg insert entires into a consistent order)
- DB state from applying functions in log order
- same log = same order = same DB

ORDER THE LOGS

- WriteID — a globally ordered unique value that can be generated locally without communication



WriteID: <time T, node ID>

a < b if a.T < b.T or
(a.T == b.T a.ID < b.ID)

Example

- On A: <10,A>: staff meeting @ 9 or 10
- On B: <20,B>: hiring meeting @ 9 or 10

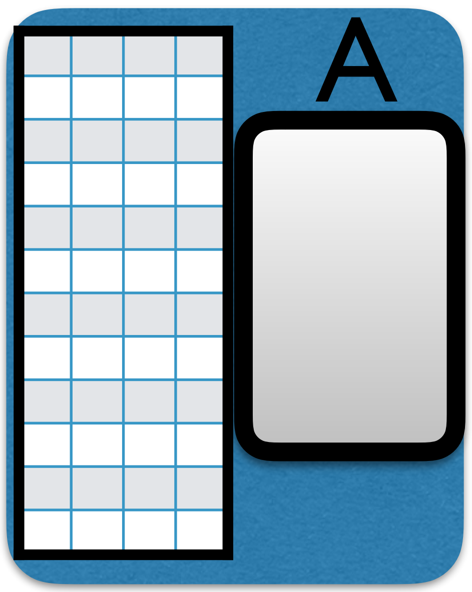
What is the eventual correct outcome?

Example

- On A: <10,A>: staff meeting @ 9 or 10
- On B: <20,B>: hiring meeting @ 9 or 10

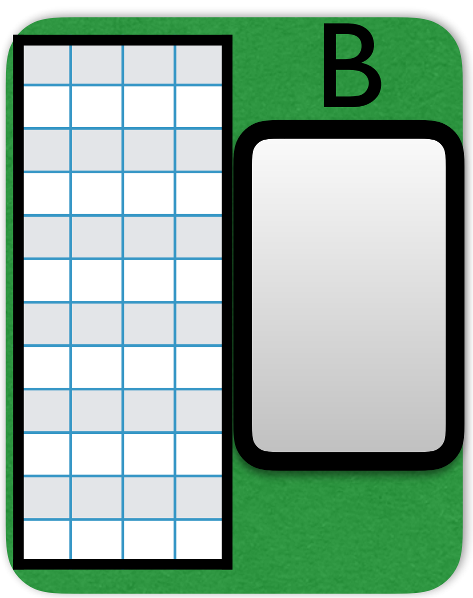
What is the eventual correct outcome?
Time Stamp order:
Staff Mtg @ 9
Hiring Mtg @ 10

<10,A>: staff @ 10 or 11



Staff @ 10

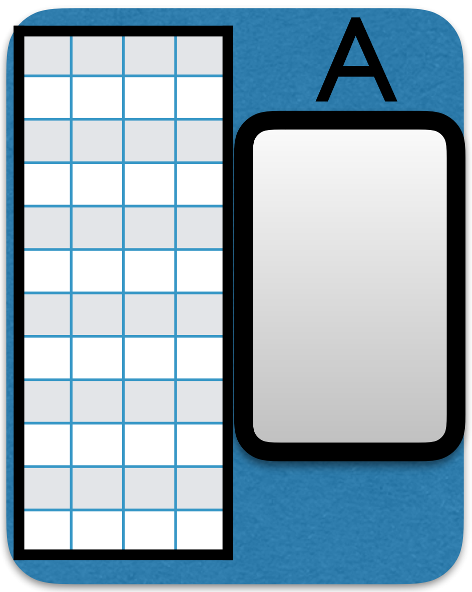
<20,B>: hiring @ 10 or 11



Hiring @ 10

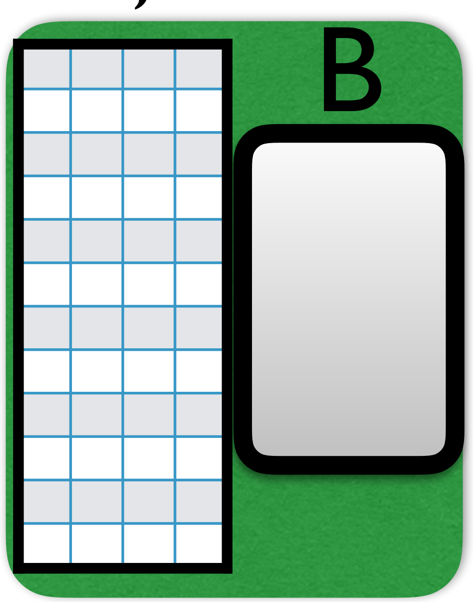
- View on Nodes A and B prior to Sync

<10,A>: staff @ 10 or 11
<20,B>: hiring @ 10 or 11



Staff @ 10

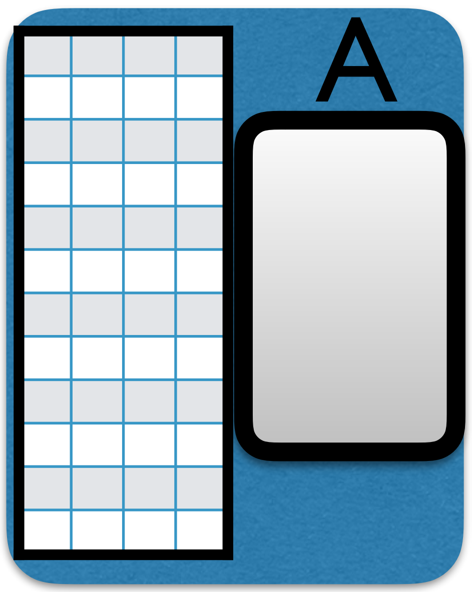
<10,A>: staff @ 10 or 11
<20,B>: hiring @ 10 or 11



Hiring @ 10

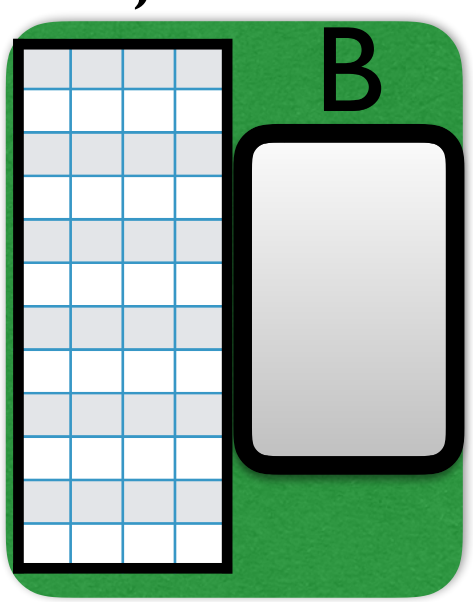
- After Sync log on both A and B have same functions in same order
- A can simply run <20,B> but… B can't run <10,A> as it has already run <20,B> too soon :-(

<10,A>: staff @ 10 or 11
<20,B>: hiring @ 10 or 11



Staff @ 10

<10,A>: staff @ 10 or 11
<20,B>: hiring @ 10 or 11



Hiring @ 11

- But log is the TRUTH
  - B needs to roll-back and re-run log
  - We will optimize this but worst case can go all the way back

What will the user see in the UI?

What will the user see in the UI?
Assumption is app will have a tentative view and things might change

- Will updates be consistent with wall clock time?
- Will updates be consistent with causality?