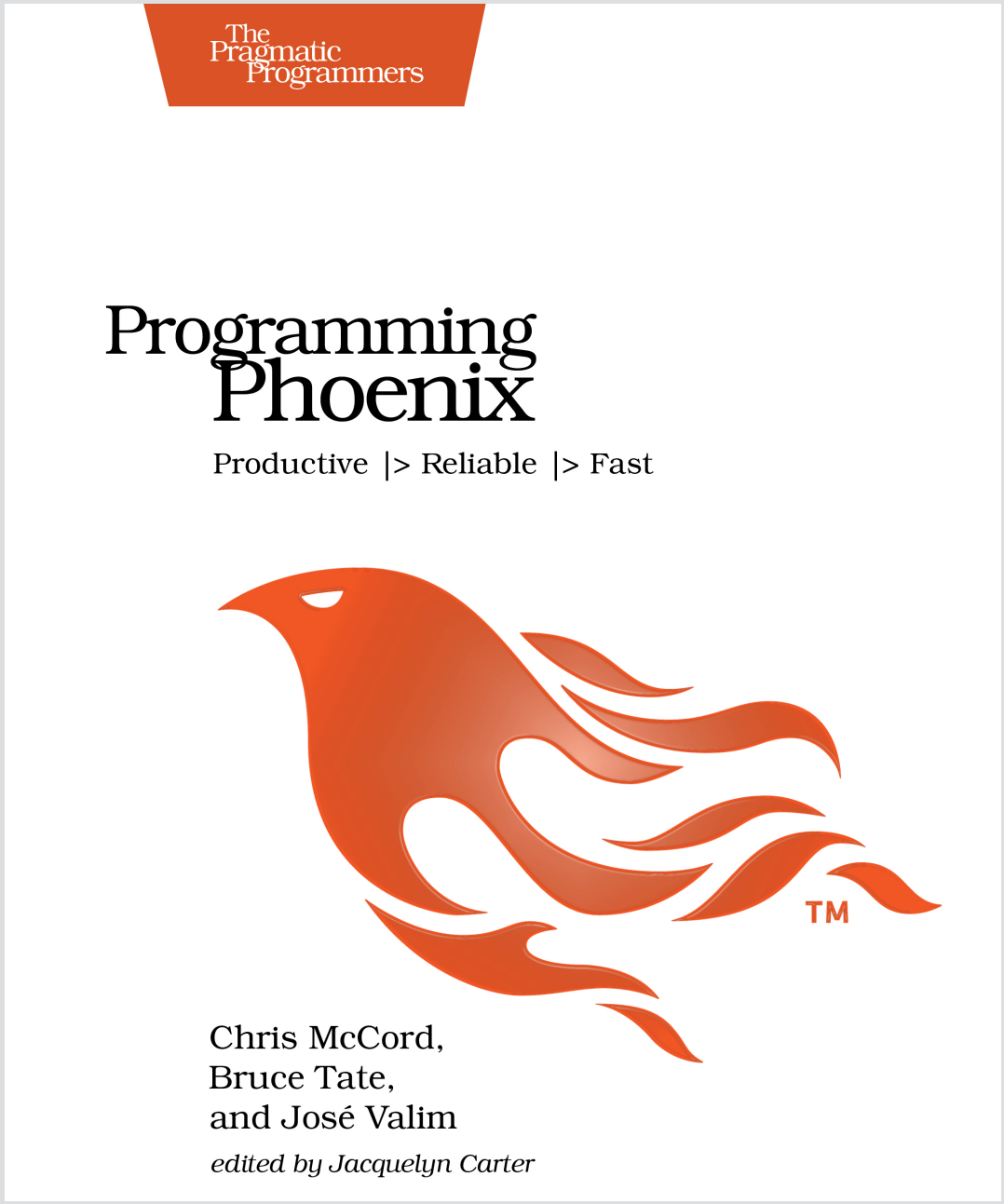
Phoenix
Framework

1.2 and Beyond

@chris_mccord



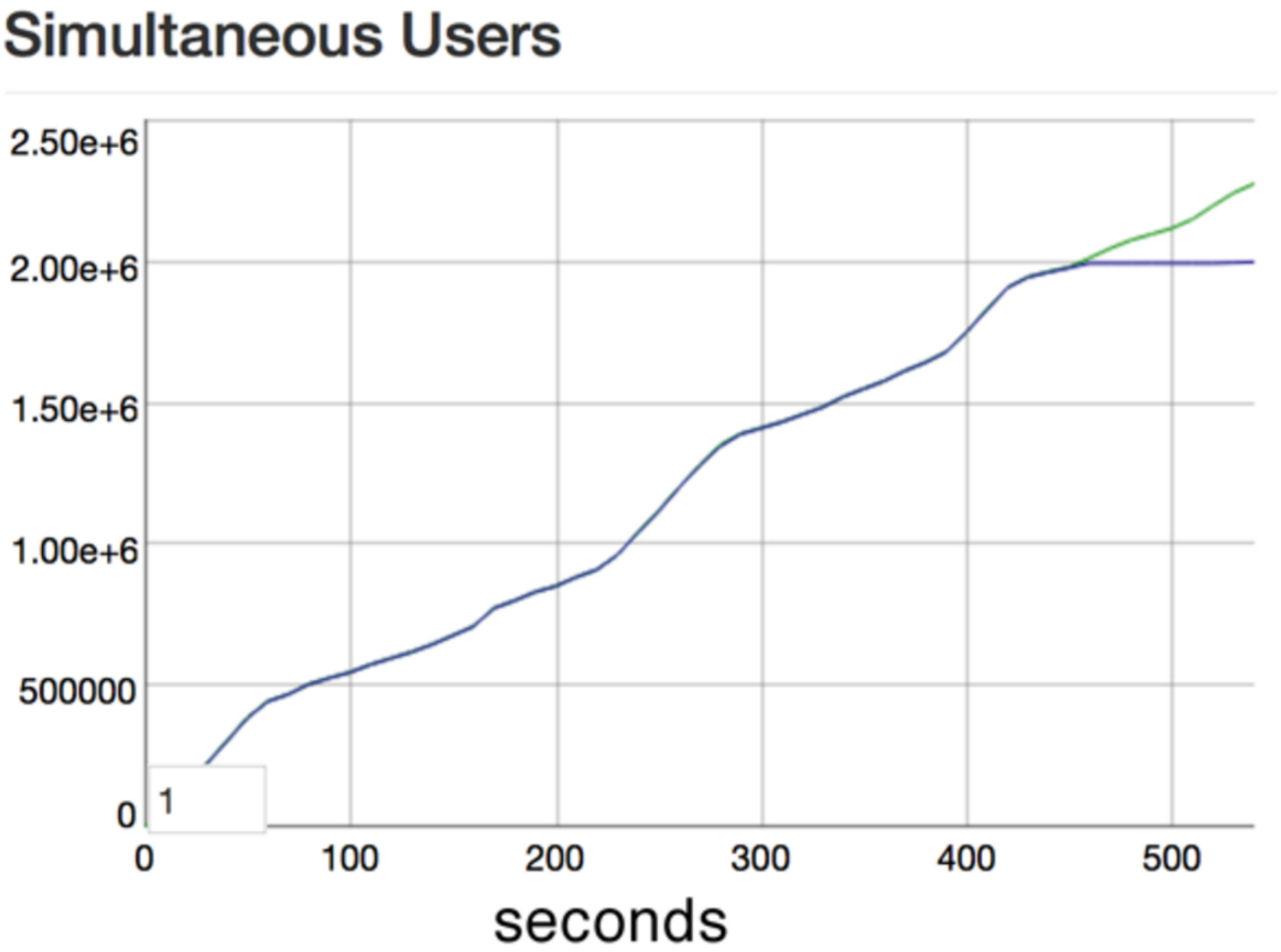
25% discount code

ElixirConfEU2016_phoenix

pragprog.com

bit.ly/phoenix_book

2M channel clients on one server



Growth

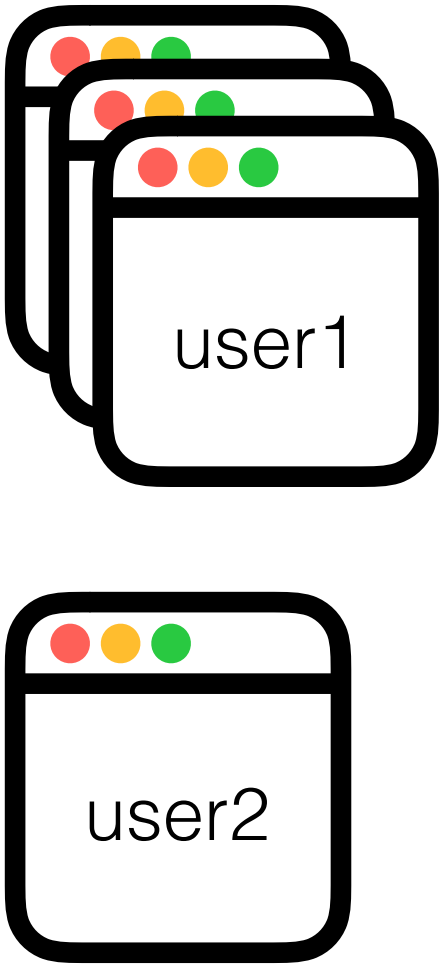
- 4k downloads/day from hex
- Focused books in the works
  - Phoenix with gaming
  - Phoenix with embedded
- Job opportunities

Phoenix 1.2rc

- Available today
- lib/ now reloaded in dev
- Extracted Phoenix.PubSub
- Phoenix.Presence

Phoenix.Presence

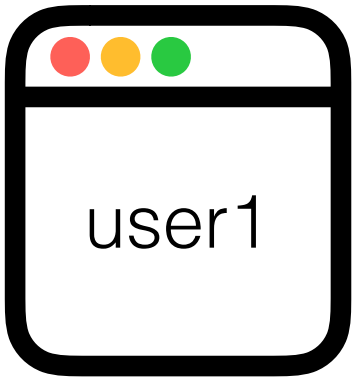
A look at the problem



Presence.list()

- user1 (3)
- user2

A look at the problem

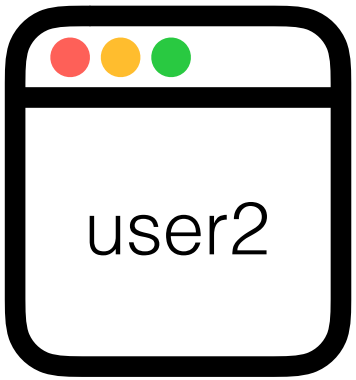


`Presence.list()`

- user1

node1

node2



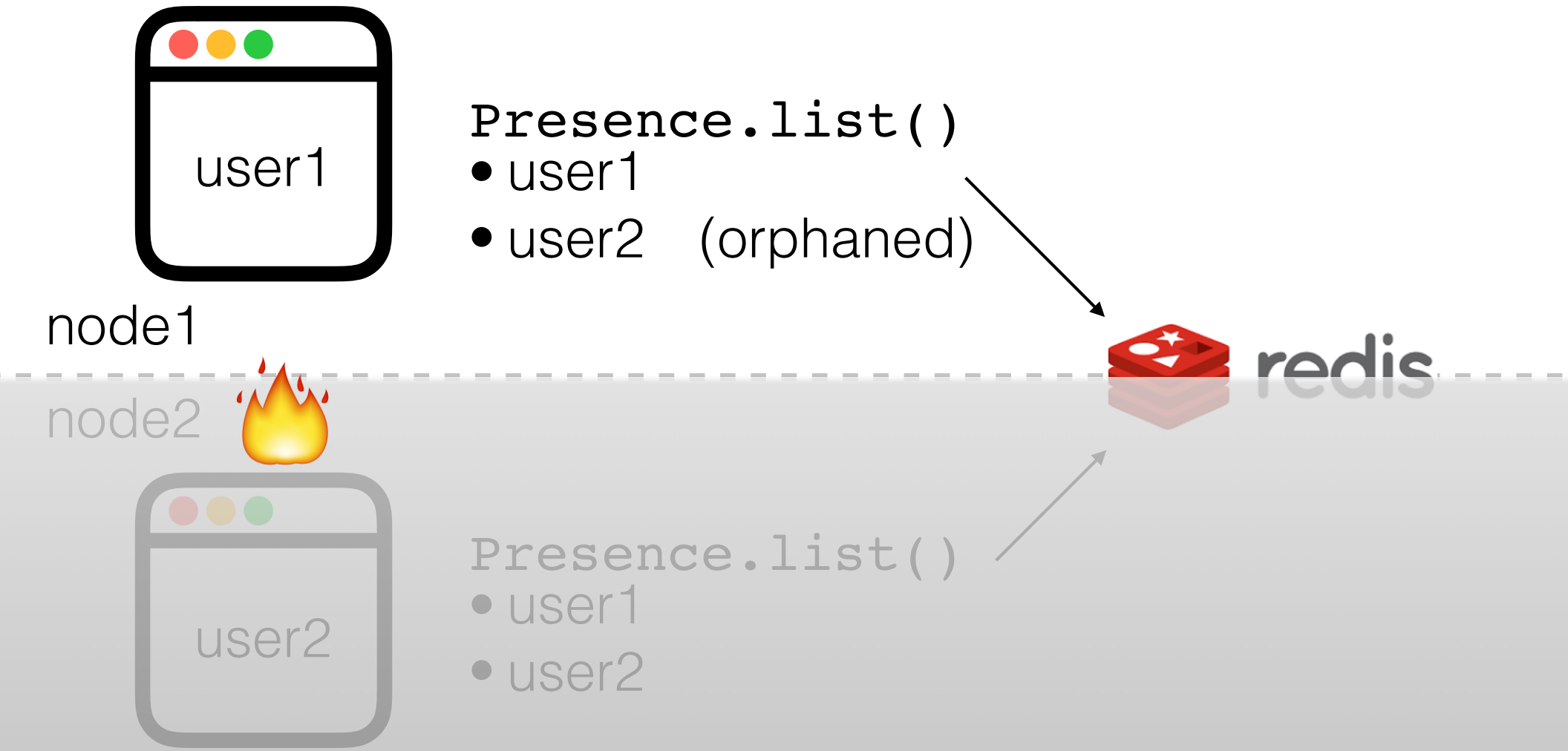
`Presence.list()`

- user2

Solution?:

Presence backed by shared database!

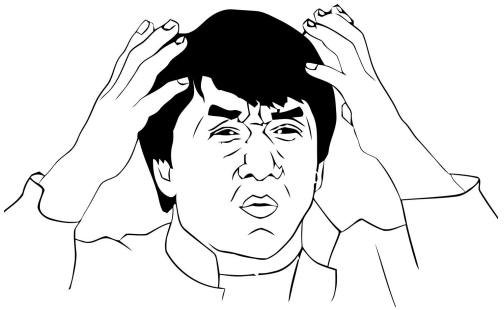
A look at the problem

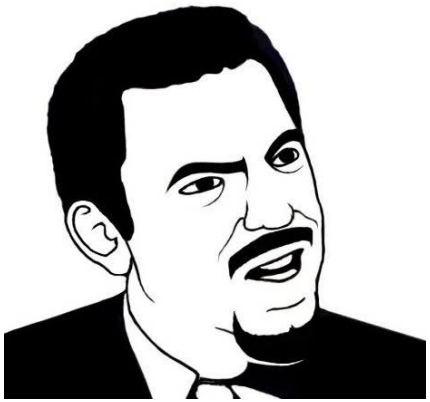


Solution: CRDT
conflict-free, replicated data type

- Strong eventual consistency
- Replicate presence join and leave events across the cluster without merge conflicts
- Conflicts are *mathematically impossible*
- Supports replication without remote synchronization

"In this way, for each replica there is a monotonic sequence of states, defined under the lattice partial order, where each subsequent state subsumes the previous state when joined elsewhere."



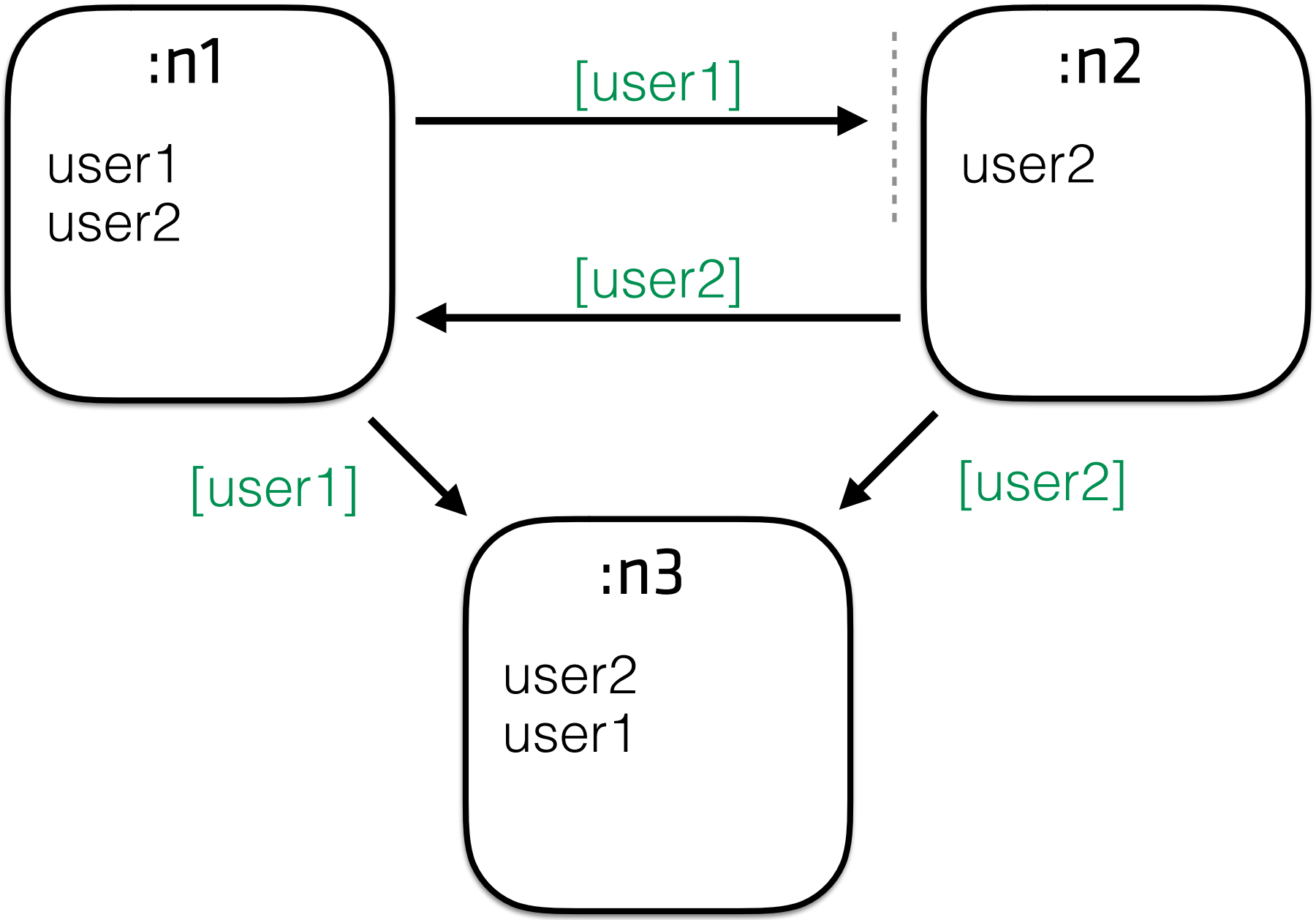


"Proof. *Trivial*, given the associativity, commutativity, and idempotence of the join operation in any join-semilattice."

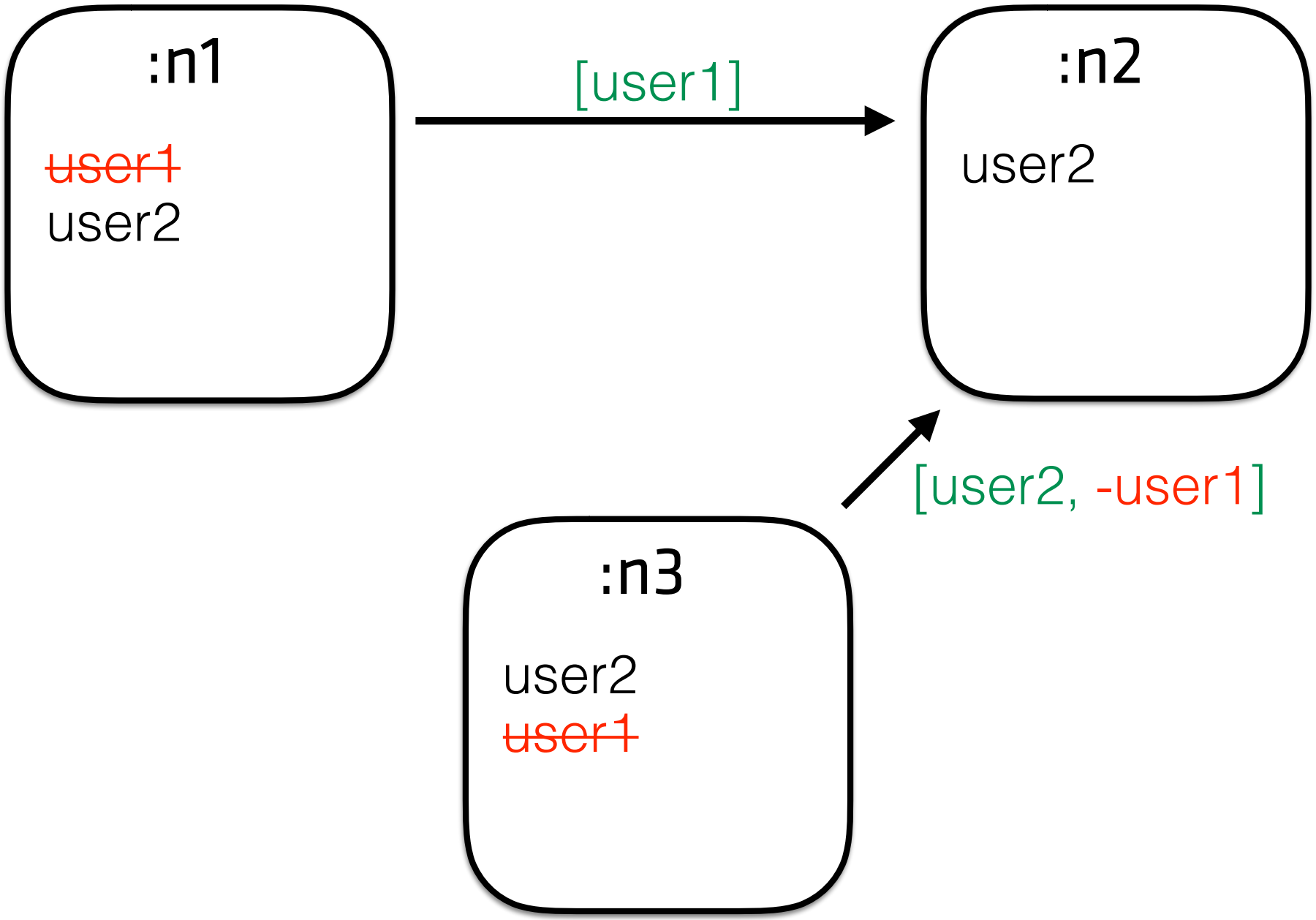
http://gsd.di.uminho.pt/members/cbm/ps/delta-crdt-draft16may2014.pdf

“Nah, you can do it”

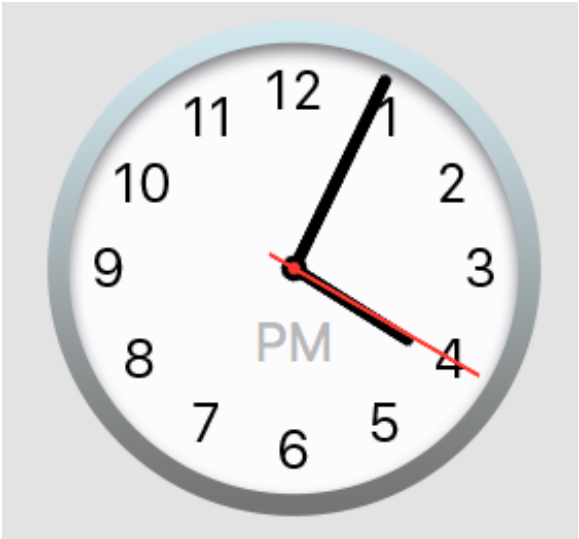
CRDT (ORSWOT)

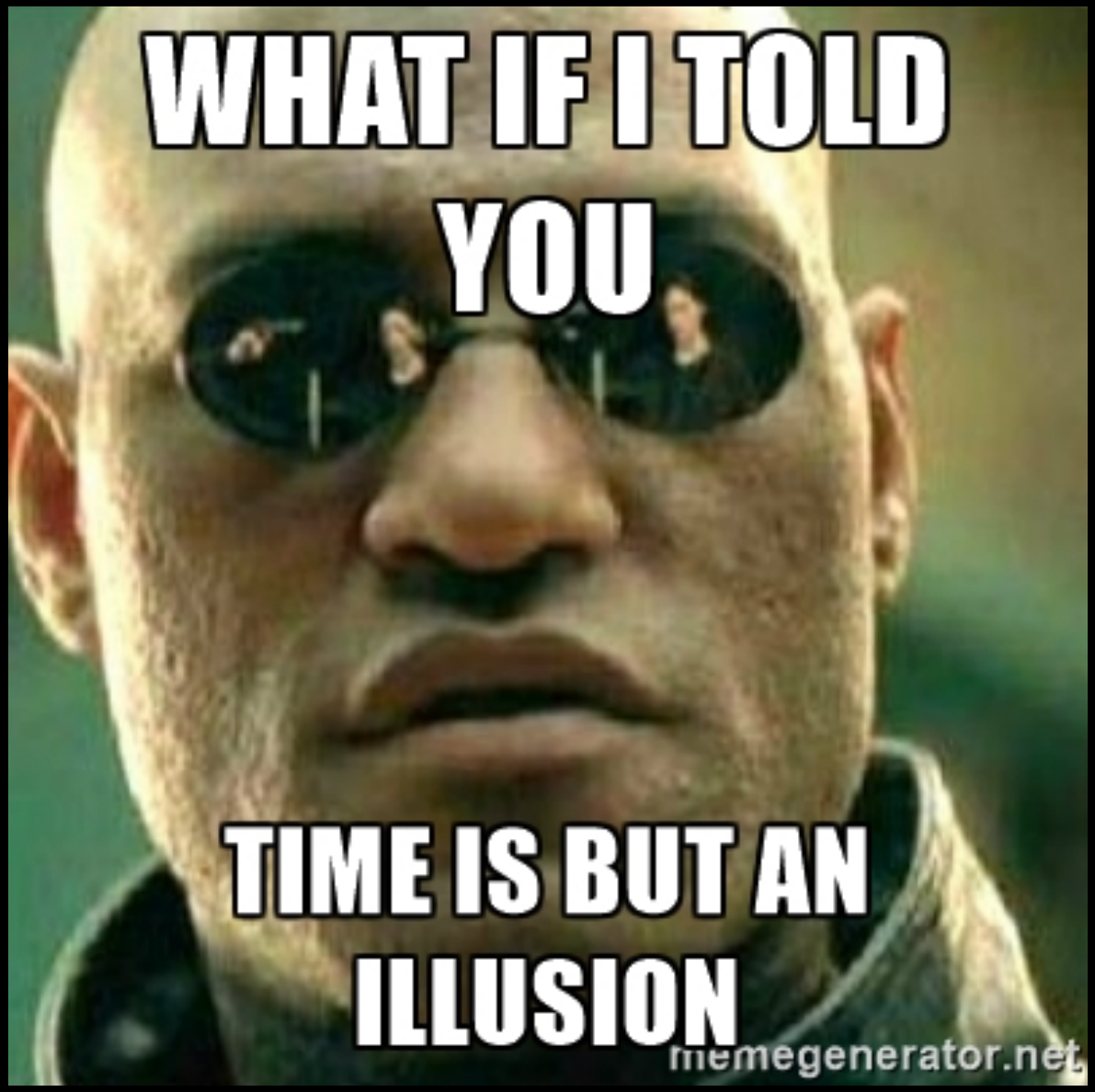

:n1
user1
user2
[user1]
:n2
user2
[user2]
[user1]
[user2]
:n3
user2
user1

Does node2 add or discard user1?



Solution: Timestamps?



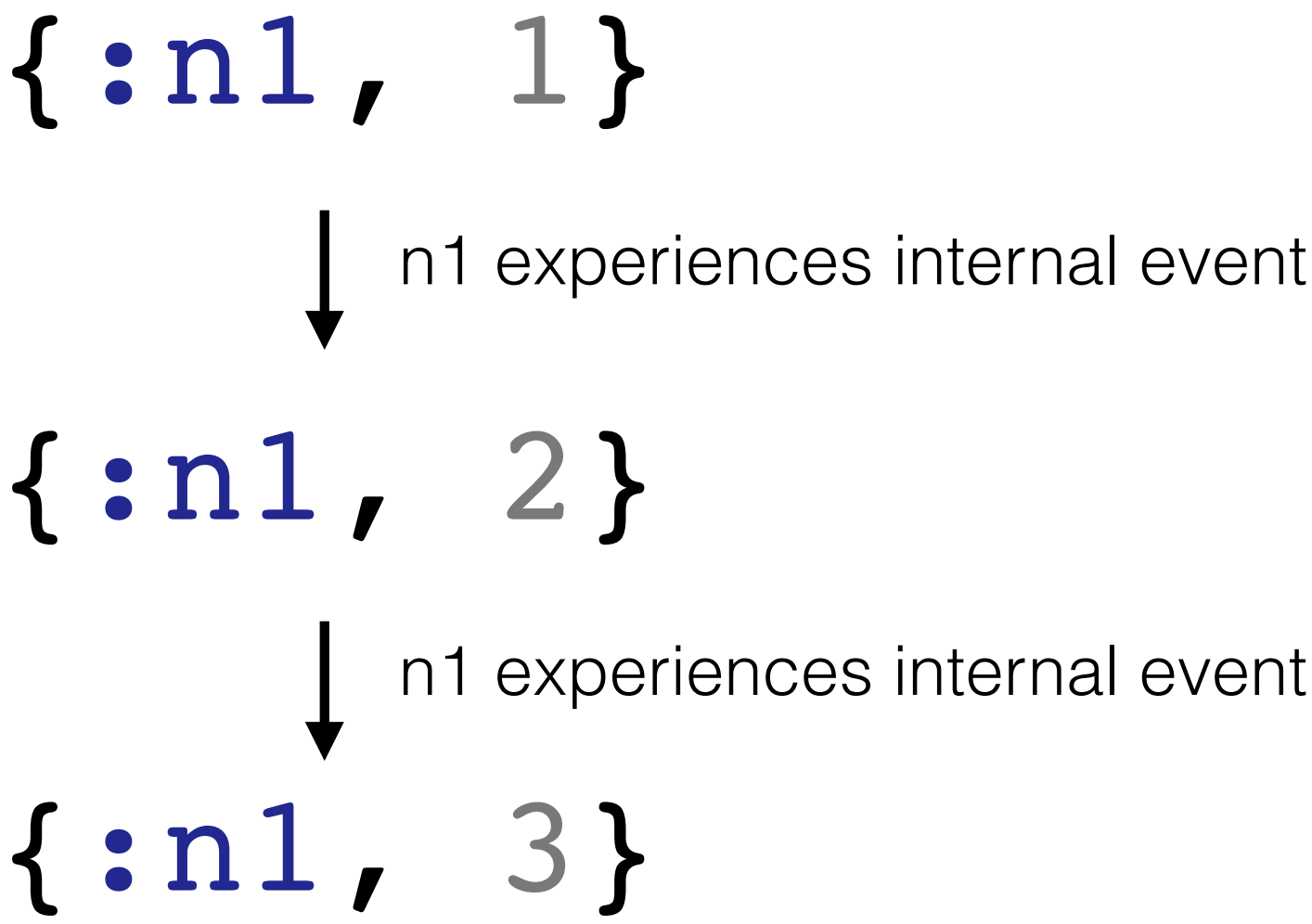

WHAT IF I TOLD YOU
TIME IS BUT AN ILLUSION
memegenerator.net

Alexander Songe
asonge

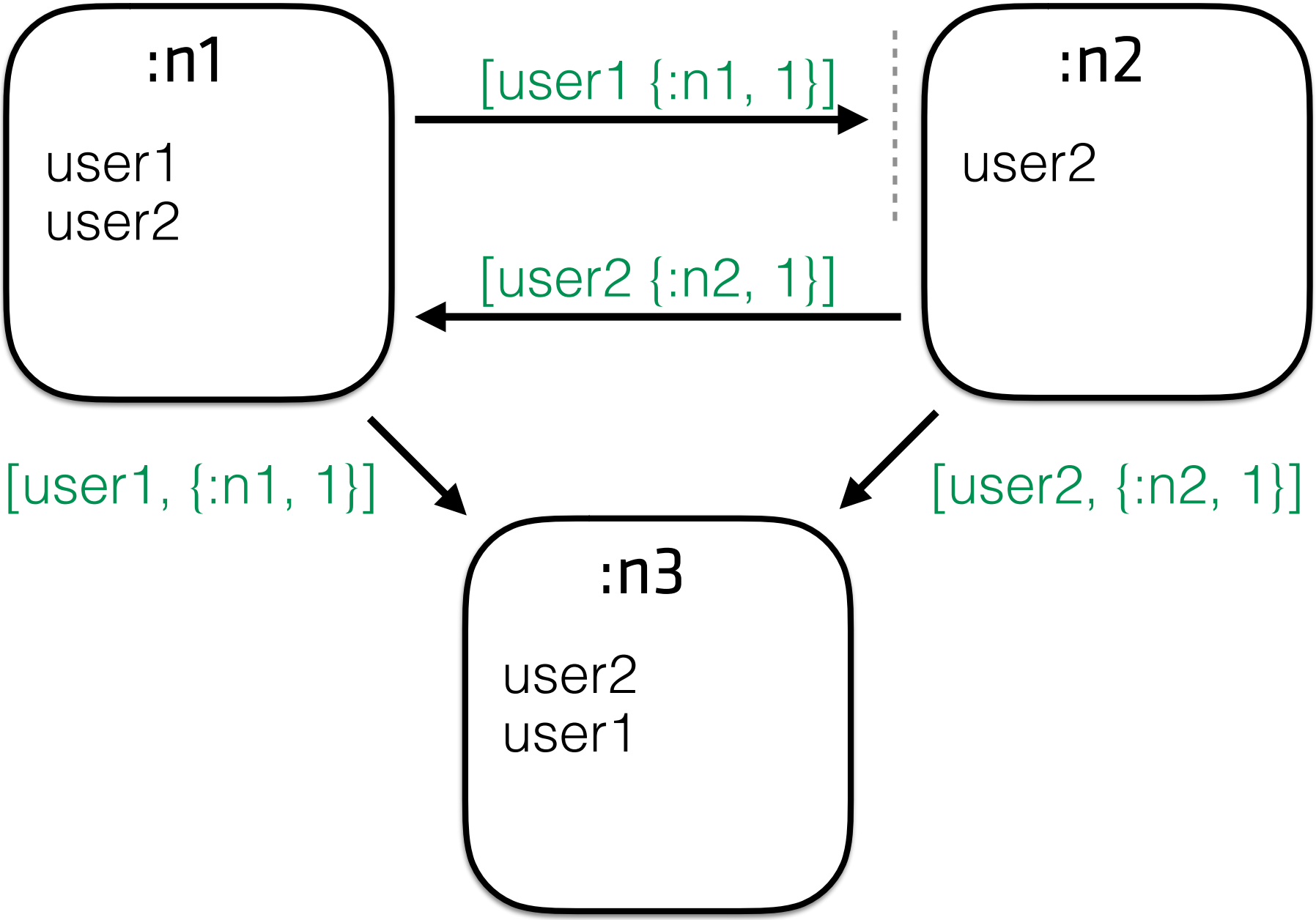
“The past will always come back to bite you”

How do you model a system without time?

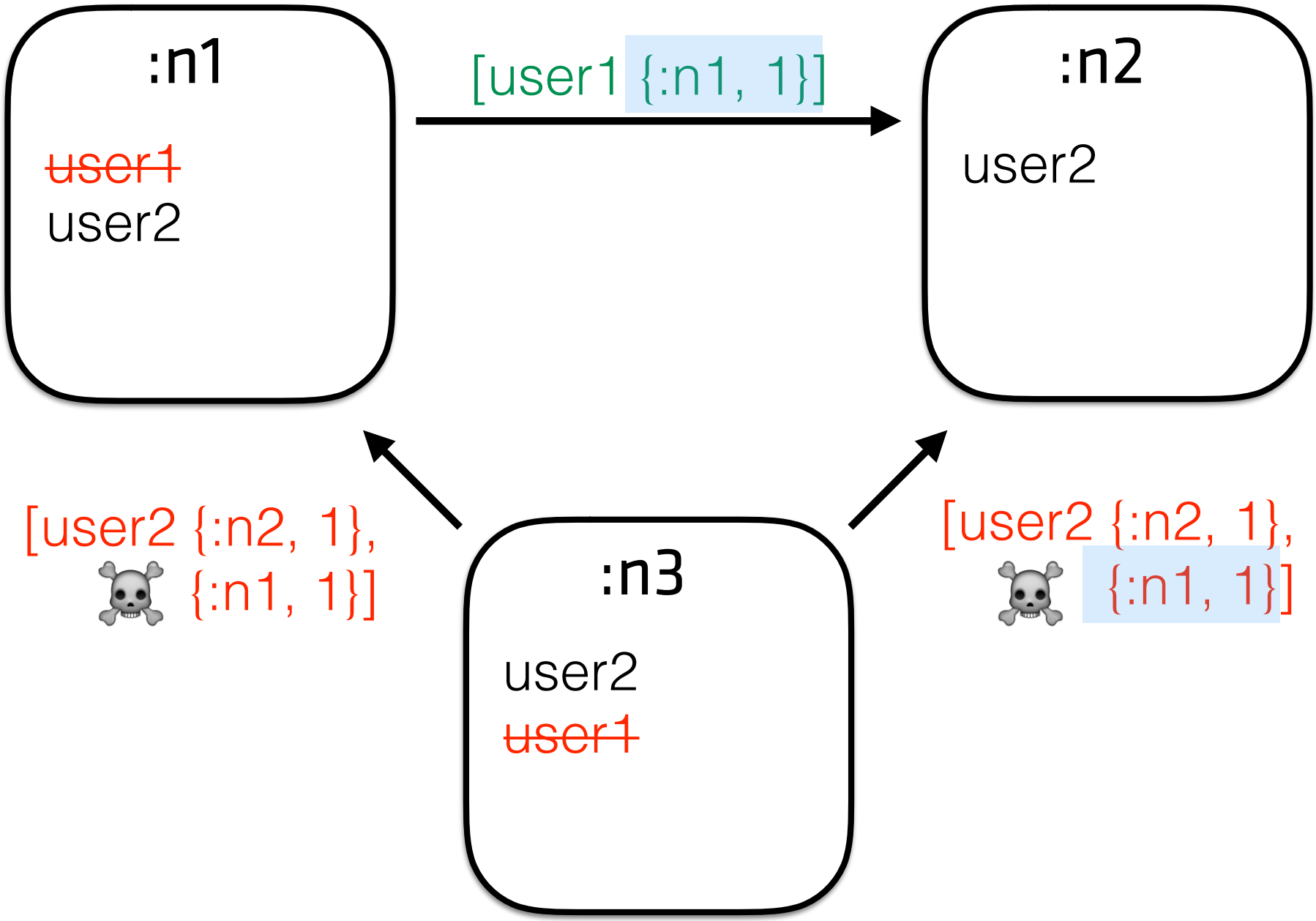
Vector Clocks



Vector Clocks

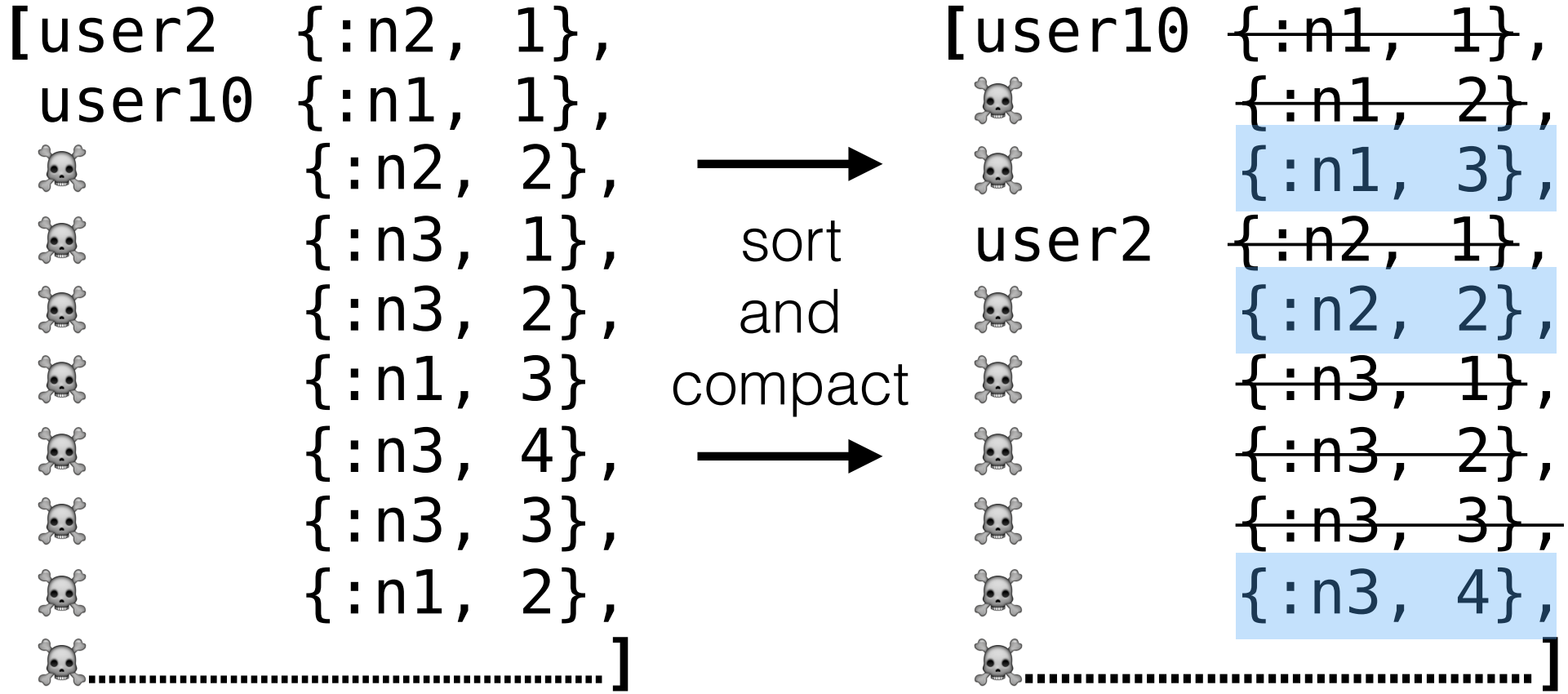


Vector Clocks and Tombstones



ORWSOT

Observed-removed set **without tombstones**



Causal context `{:n1 >= 3},{:n2 >= 2},{:n3, >= 4}`

ORWSOT

Observed-removed set without tombstones

Causal context `{:n1 >= 3},{:n2 >= 2},{:n3, >= 4}`

`[user10 {:n1, 1}, user2 {:n2, 1}]`

Server API

```
defmodule RoomChannel do
  def join("rooms:" <> room_id, _, socket) do
    send self(), :after_join
    {:ok, socket}
  end

  def handle_info(:after_join, sock) do
    id = sock.assigns.user_id
    Presence.track(sock, id,%{status: "avail"})
    push sock, "presences", Presence.list(sock)
    {:noreply, sock}
  end
end
```

Client API

```
import {Socket, Presence} from "phoenix"
let socket = new Socket("/socket")
let room = socket.channel("rooms:" + id)
let presences = {}

room.on("presences", state => {
  Presence.syncState(presences, state)
})
room.on("presence_diff", diff => {
  Presence.syncDiff(presences, diff)
})
console.log("users", Presence.list(presences)})
```

Demo

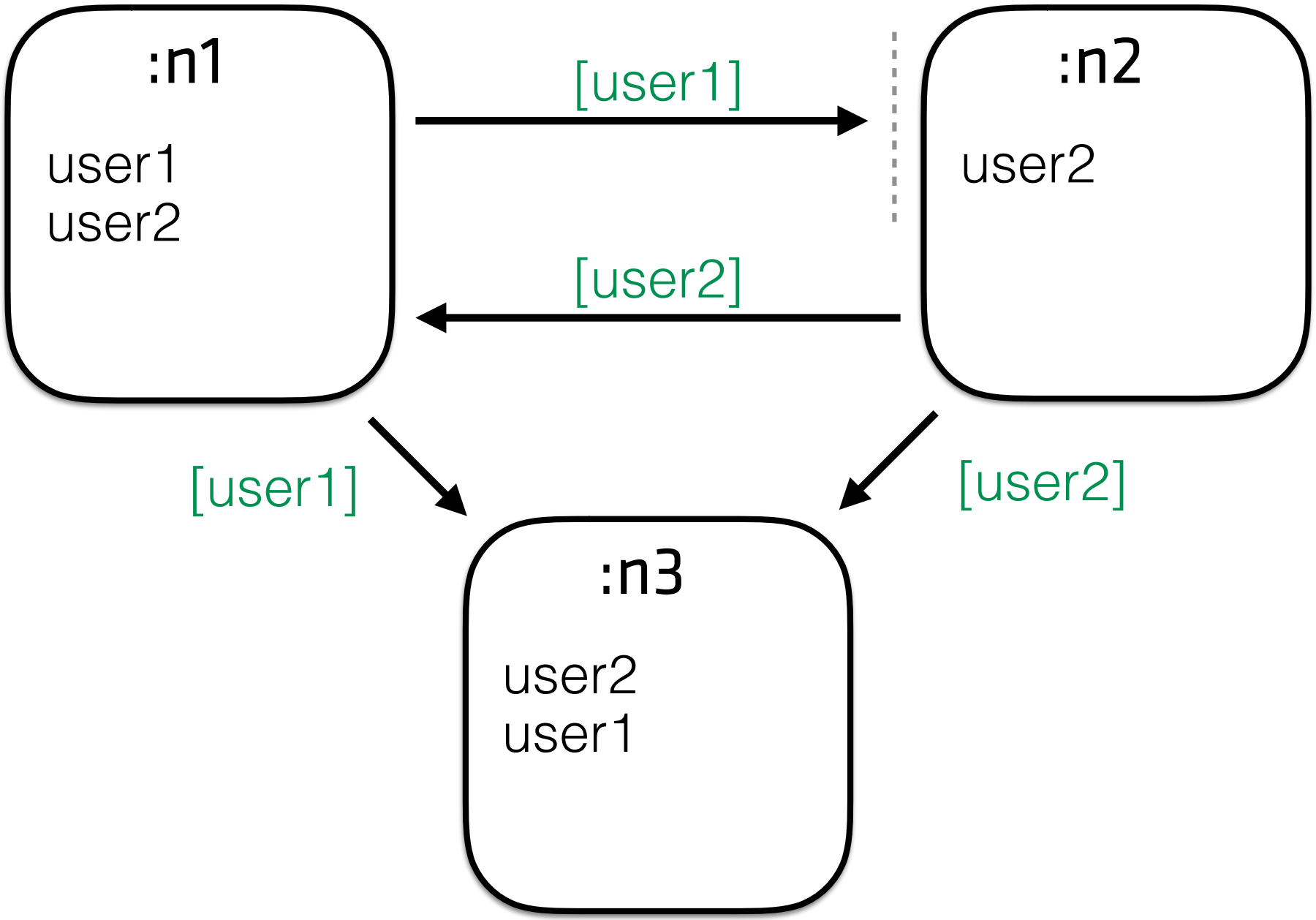
Phoenix vNext

Service Discovery

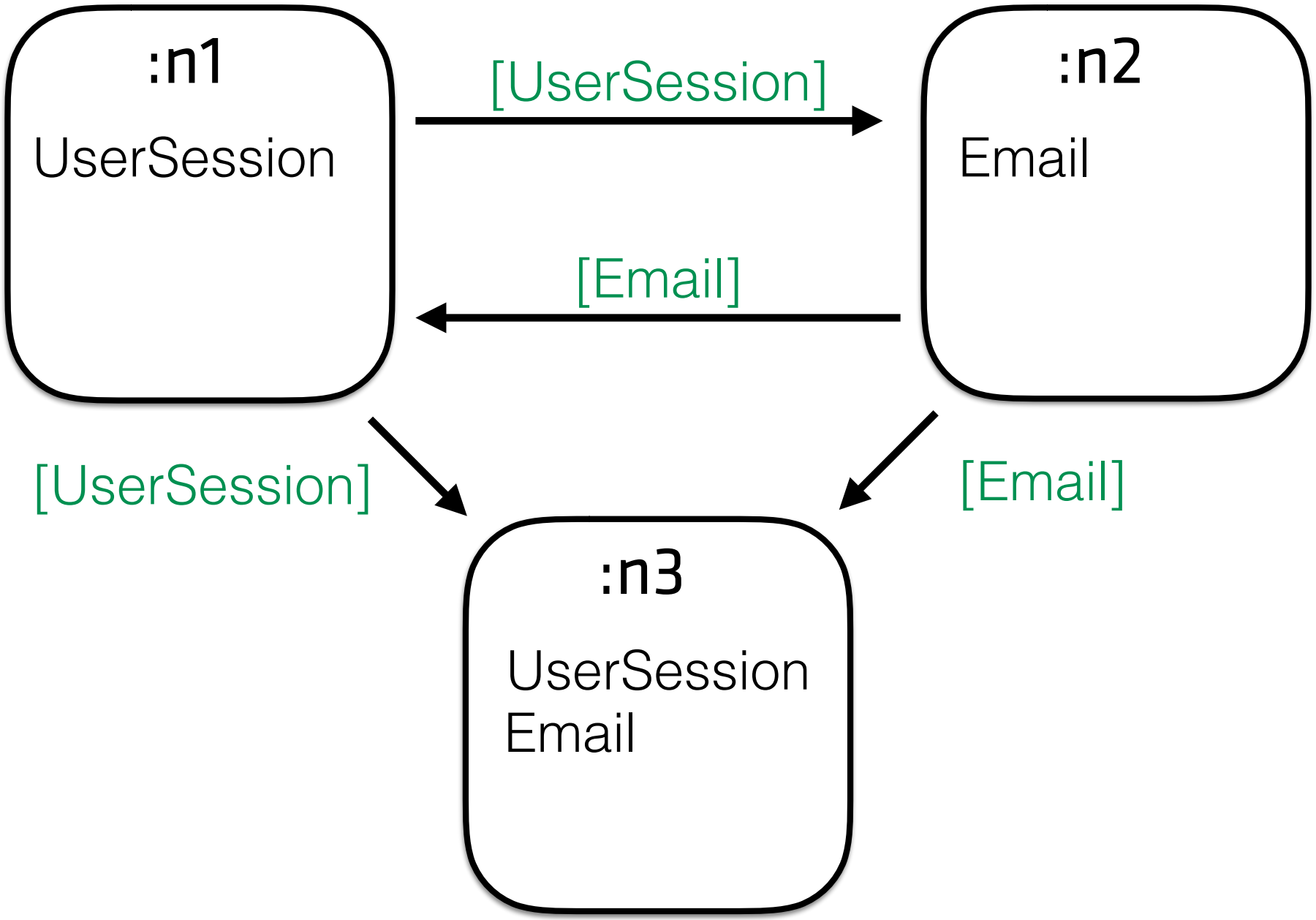
Building on top of Presence

- Distributed Service Discovery
- Process Placement

CRDT (ORSWOT)


:n1
user1
user2
[user1]
:n2
user2
[user2]
[user1]
[user2]
:n3
user2
user1

Service Discovery



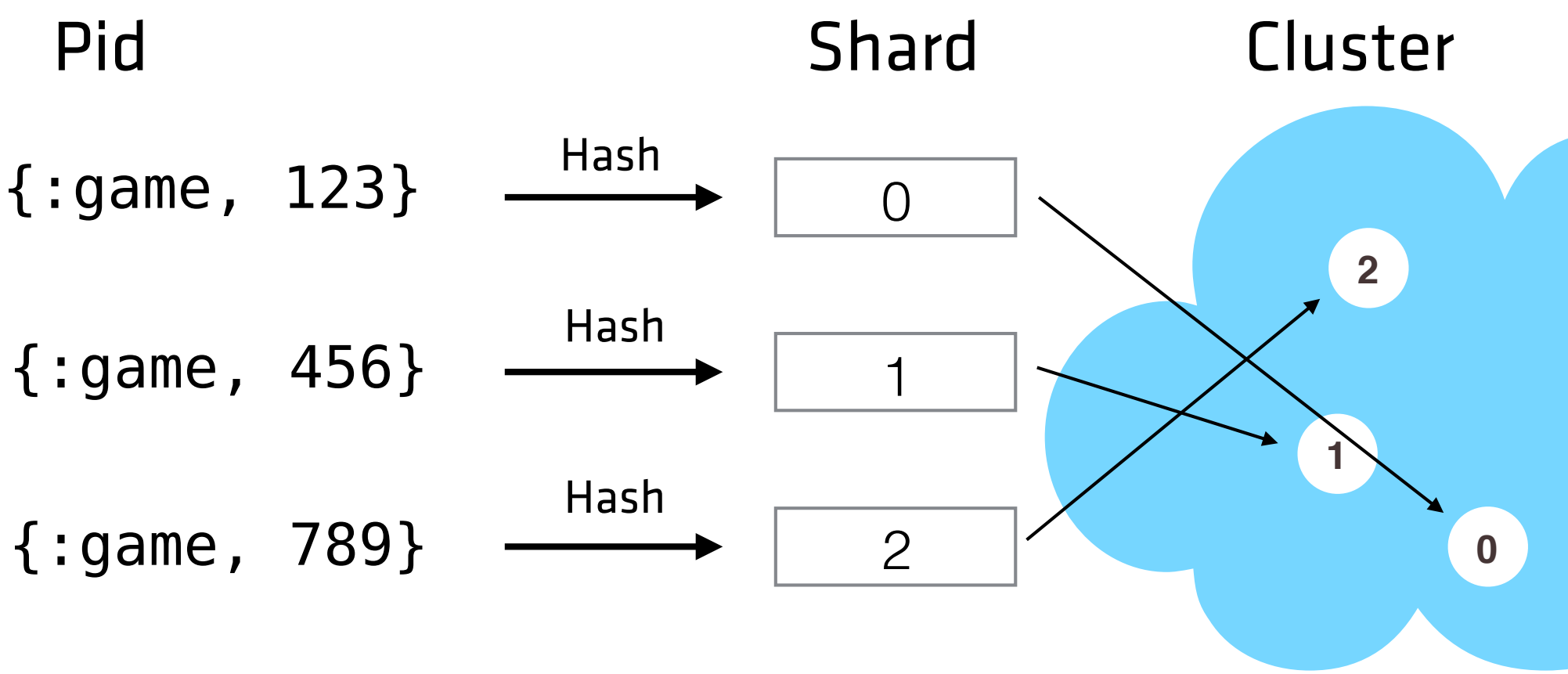
Phoenix.Presence (Tracker)

An accidental process group

```
Presence.track("rooms:11", "user1", %{
  name: "Chris"
})
:ok

Presence.list("rooms:11")
%{"user1" => [%{name: “Chris"}], ...}
```

```
Presence.list("rooms:123")
=> %{"user1" => %{name: "Chris"}, ...}

Presence.list("crawler")
=> %{"crawler" => [%{pid: pid, jobs: 99},
                   %{pid: pid, jobs: 45}]}


Service.list(:crawler)
=> [%{pid: pid, jobs: 99},
    {pid: pid, jobs: 45}]
```

Service Routing Strategies



```elixir
# Public API
Game.spawn_new_session(user.id)
=> {:ok, #PID<0.70.0>}


# Internal - Routing by shard
Service.call(Game, {:spawn, uid})


# Internal - custom client load balancing
Service.call(Game, {:spawn, gid}, fn s1, s2 ->
  s1.work_factor < s2.work_factor
end)
```

Other Areas of Interest

- First-class umbrella integration
  - mix phoenix.new app —umbrella
- More extensible transports
- Better generators that focus on isolation
- Client/Server DataSync

DOCKYARD

www.dockyard.com