EUC 2016

BARREL

https://barrel-db.org

https://www.icloud.com/keynote/0ntpkGjFB5xxjO5-6J3S1LeSg#talk_EUC_2016

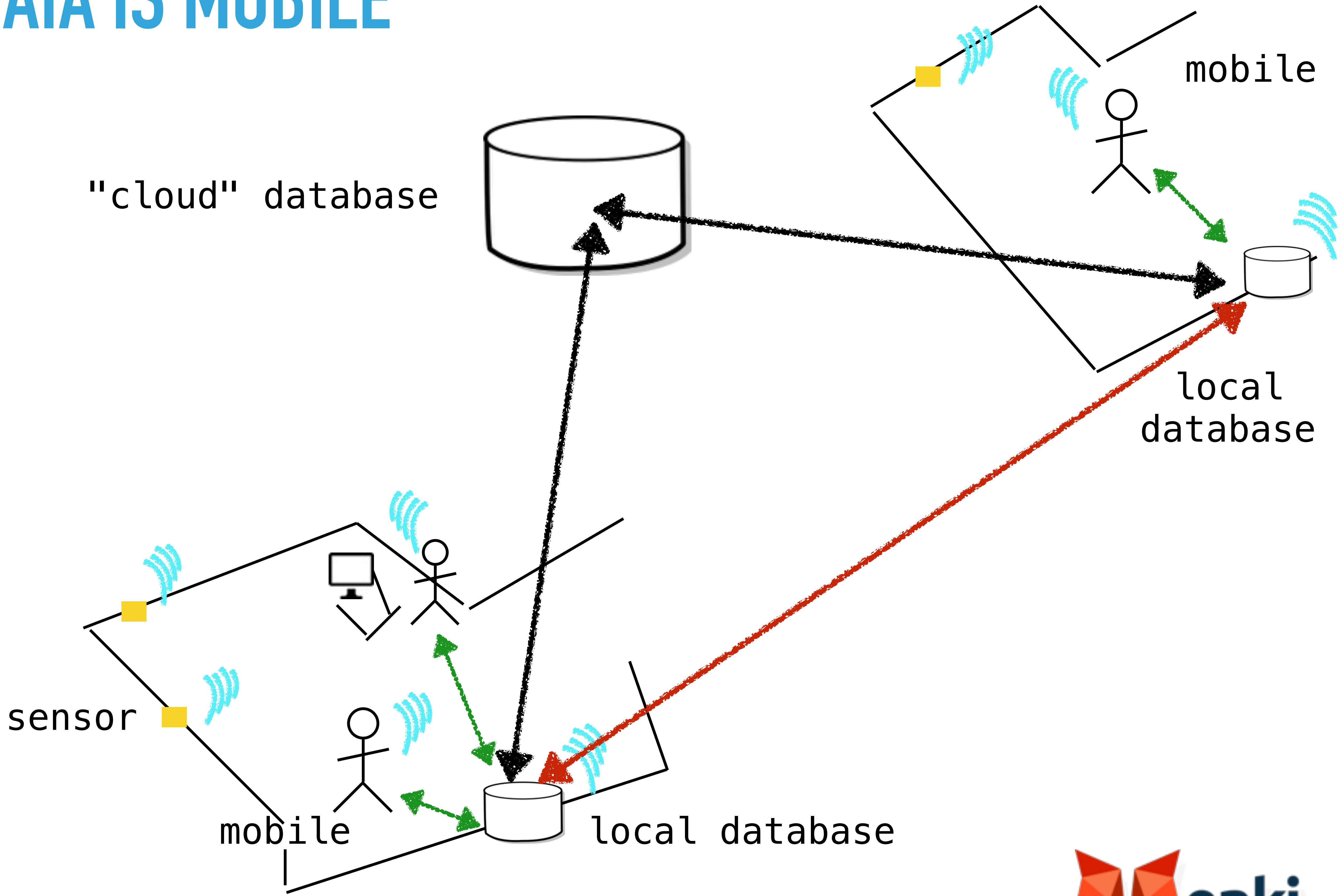
TODAY PLAN

1. Barrel?
2. Building an Erlang database in Erlang and actually fully basing it on the actor model
3. Pluggable backends & plugin: how to have plugins, nifs/dirty nifs
4. Brand new code: very alpha release.

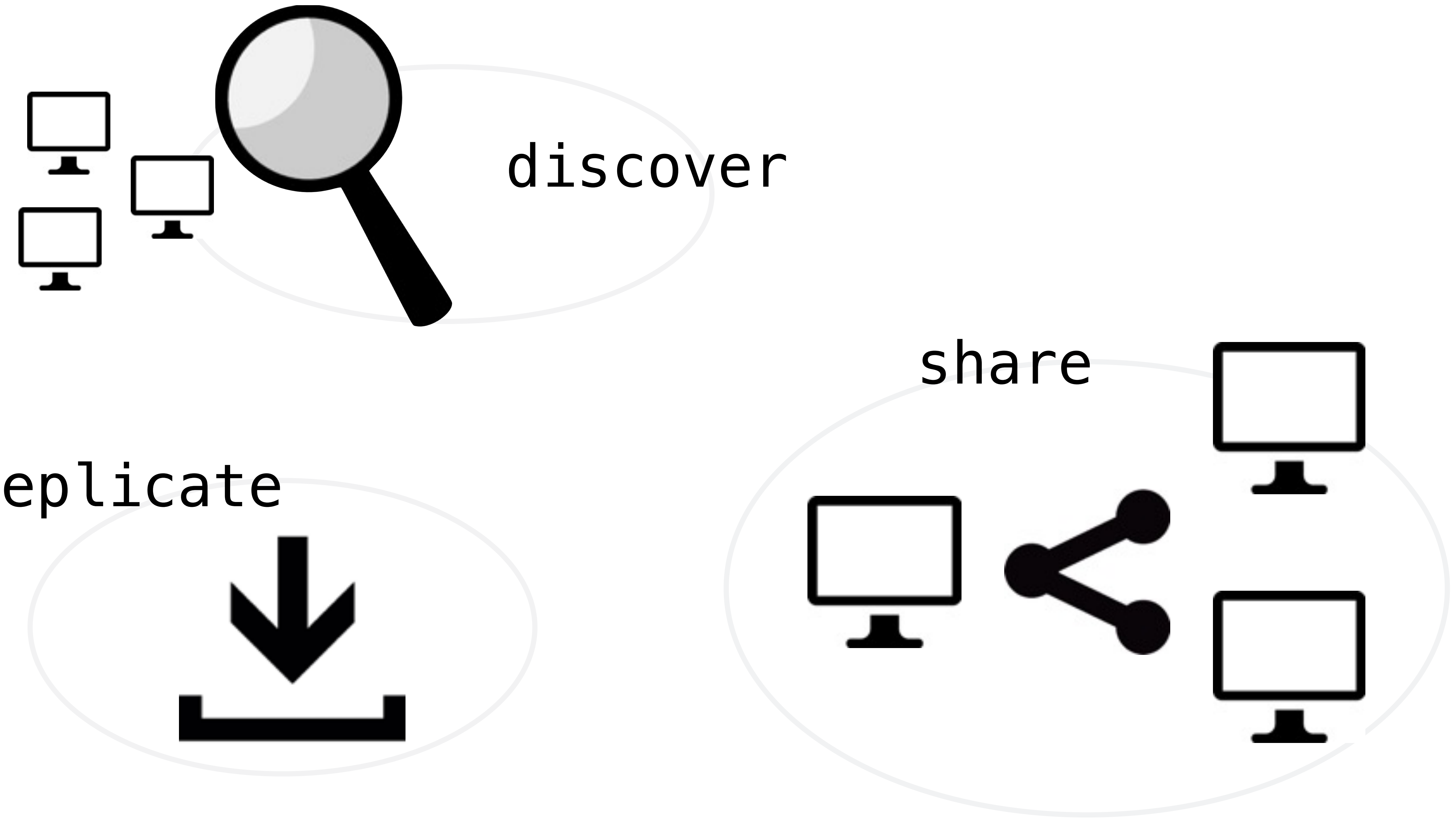
VISION AND CONCEPT

- **A decentralized data platform**
- Easily **coordinate multiple data sources** coming from devices, peoples or services around the world through a P2P platform

DATA IS MOBILE



PEER TO PEER (P2P)



enki

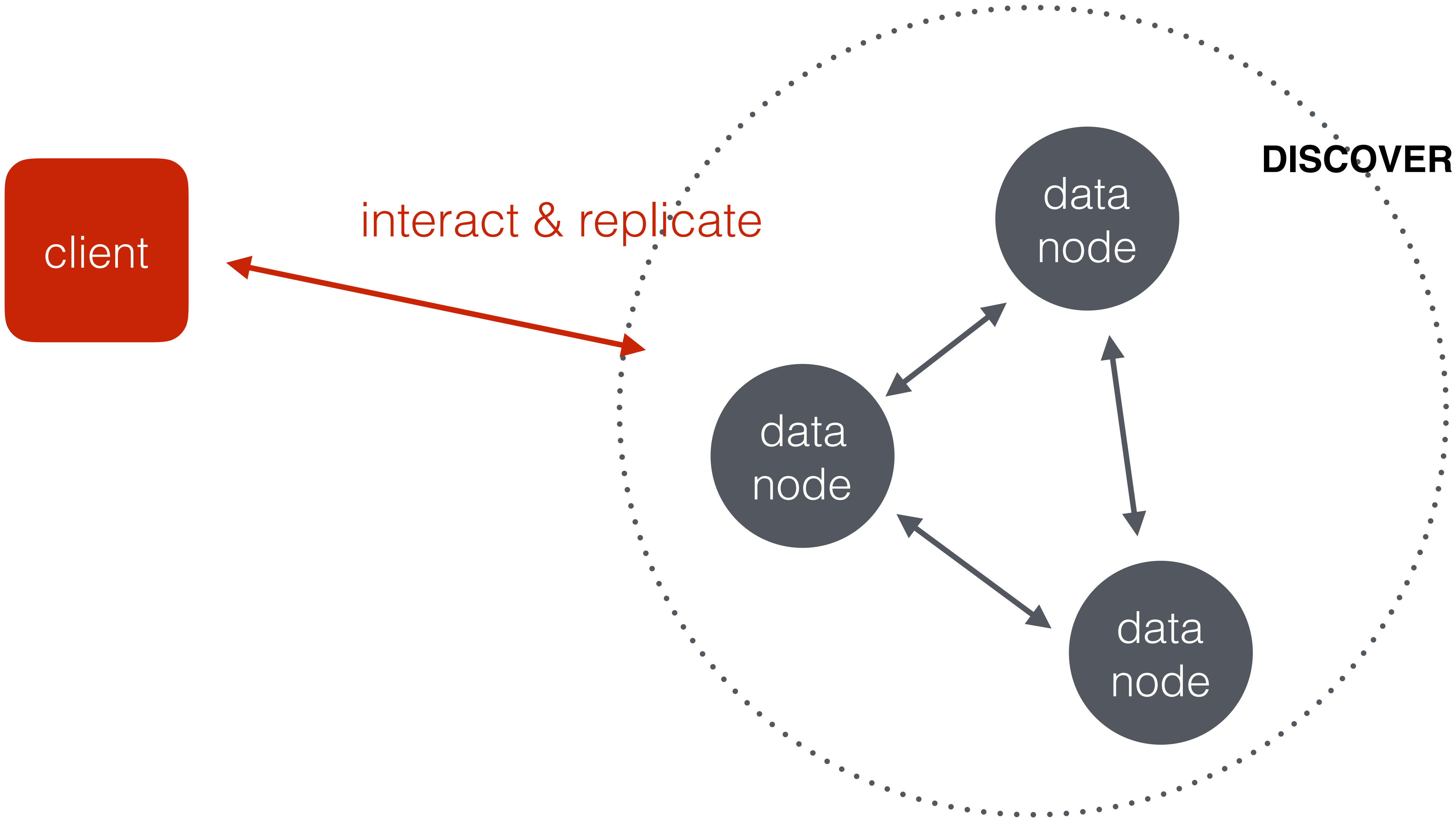
- Local first
- Put/Match the data next to you
- Query Locally
- Replicate a view of the data you need

WHAT IS BARREL

- a database that focus on the replication
- Organize the data to be replicated between machines/peopl
- a document database with attachments
- replication between any nodes in both way
- HTTP 1.1/2 API

WHAT IS BARREL

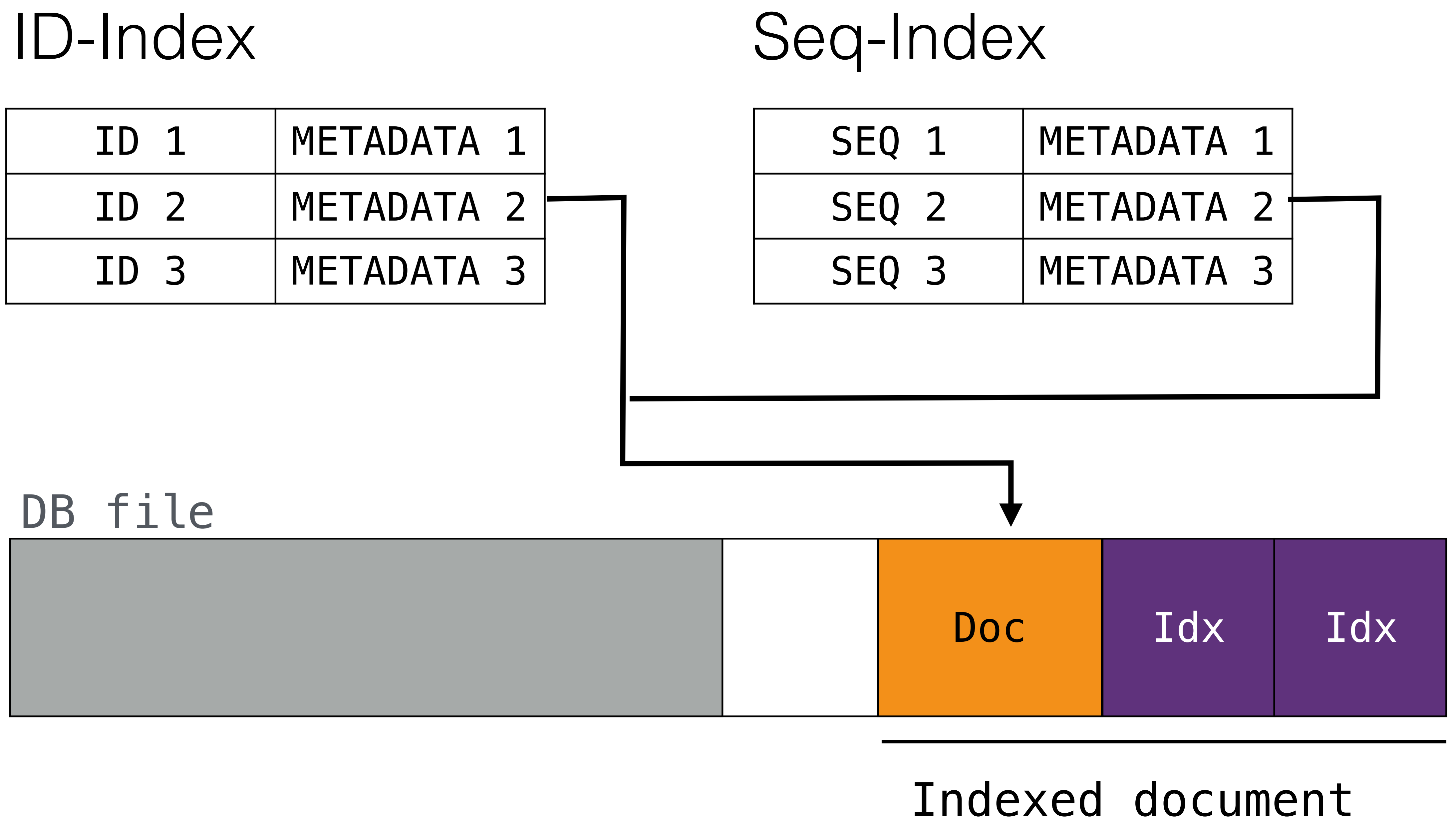
enki


client
interact & replicate
DISCOVER
data node
data node
data node

P2P

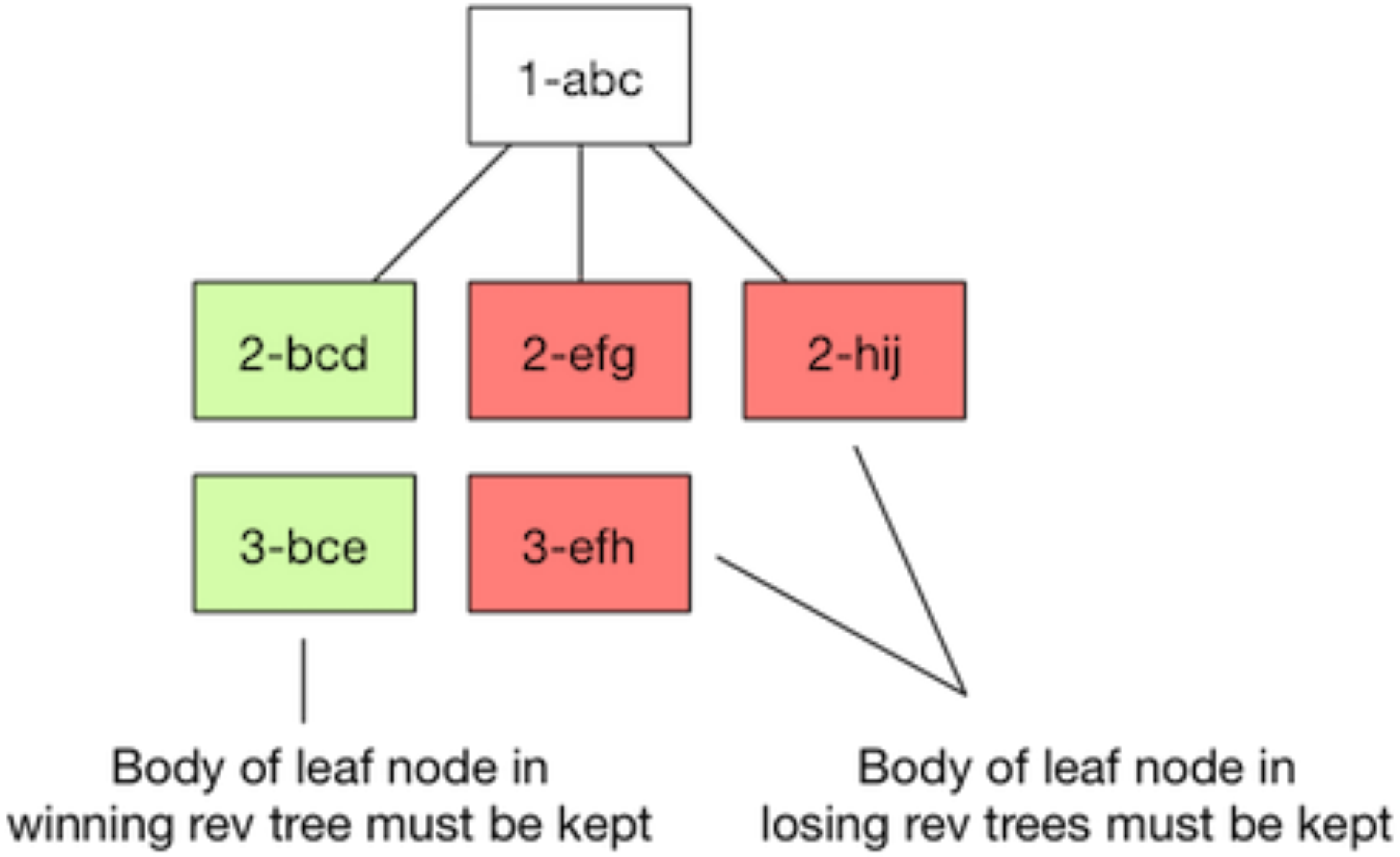
- Over HTTP
- Replication is the core
- Each nodes can replicate each others (PUSH/PULL), can happen simultaneously
- Chained replication
- AUTO-DISCOVERY

P2P



DOCUMENT STORAGE

REVISION TREE


1-abc
2-bcd
2-efg
2-hij
3-bce
3-efh
Body of leaf node in winning rev tree must be kept
Body of leaf node in losing rev trees must be kept

- keep the history of a document
- Leafs contains the conflict
- normal case: A document is stored with its parent revision
- during replication: A document fetched or pushed with its history
- deleted revision are kept

WRAP-UP: REVISION TREE

- Compatible with POUCHDB (& Couchbase Lite)
- Can be embedded in Elixir & Erlang applications (soon)
- GSOC 2016: https://github.com/barrel-db/rebar3_elixir_compile (easily embed elixir app in Erlang)

ONE MORE THING

enki

- selectively get updates from a database
- Get view change /<db>/_changes?filter=_view&view=<dname>/<viewname>.

VIEW CHANGE ?

BUILD <🏠/>
IN ERLANG

enki

- concurrent writers/concurrent readers?
- can’t rely on the usual suspects: no (system) atomic lock no STM)
- we have “actors” (kind of) , at least Erlang processes & OTP

CHALLENGES

enki

- isolated, only message passing
- A process is a bottleneck
- ETS?

CHALLENGES

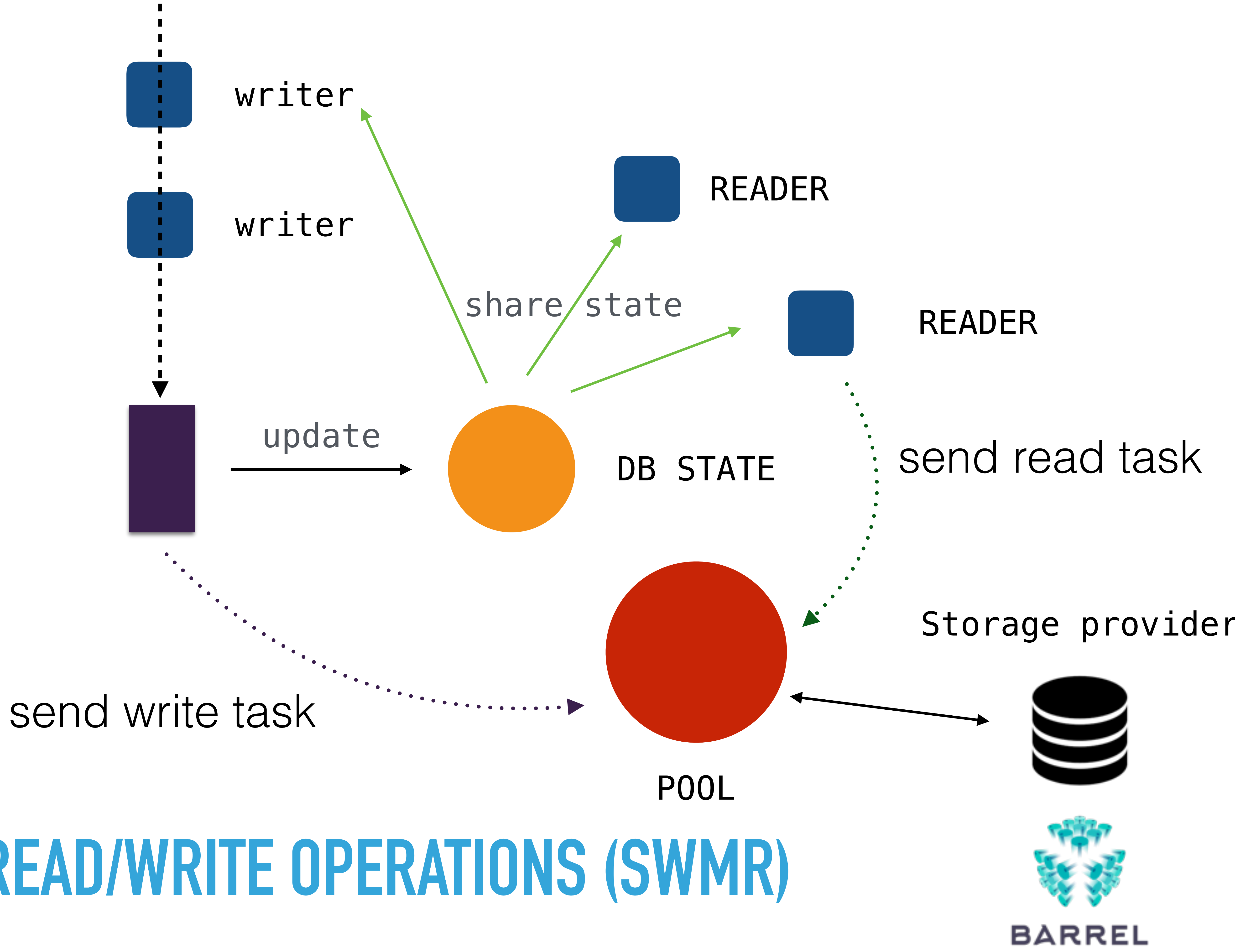
enki

- considered only 2 models
- concurrent writers racing for a writer: optimistics write
- Single Writer, Multiple Readers (SWMR)

DIFFERENT PATTERNS

- akka: persistent actors
- orlans: storage provides
- only actors. no concurrent dictionary

HOW OTHERS DO?



READ/WRITE OPERATIONS (SWMR)

WRAP-UP

- 1 gen_server as the interface (barrel_db)
- A simple process to locks writes
- 1 pool to handle concurrency / storage providers. All reads and write are done over it.

enki

- Simple abstraction (mostly KV)
- Configured at startup

```
{stores, [
  {barrel_test_rocksdb, barrel_rocksdb_store, [
    {workers, 71},
    {dir, "testdb"}
  ]}
]}
```

PLUGGABLE STORAGE

- Configured at startup
- dirty nifs: https://gitlab.com/barrel-db/barrel-rocksdb
- or https://gitlab.com/barrel-db/erocksdb

ROCKSDB STORAGE

enki

https://gitlab.com/barrel-db/barrel

PLAY

Barrel

[HTTPS://BARREL-DB.ORG](https://barrel-db.org)

[CONTACT@BARREL-DB.ORG](mailto:contact@barrel-db.org)