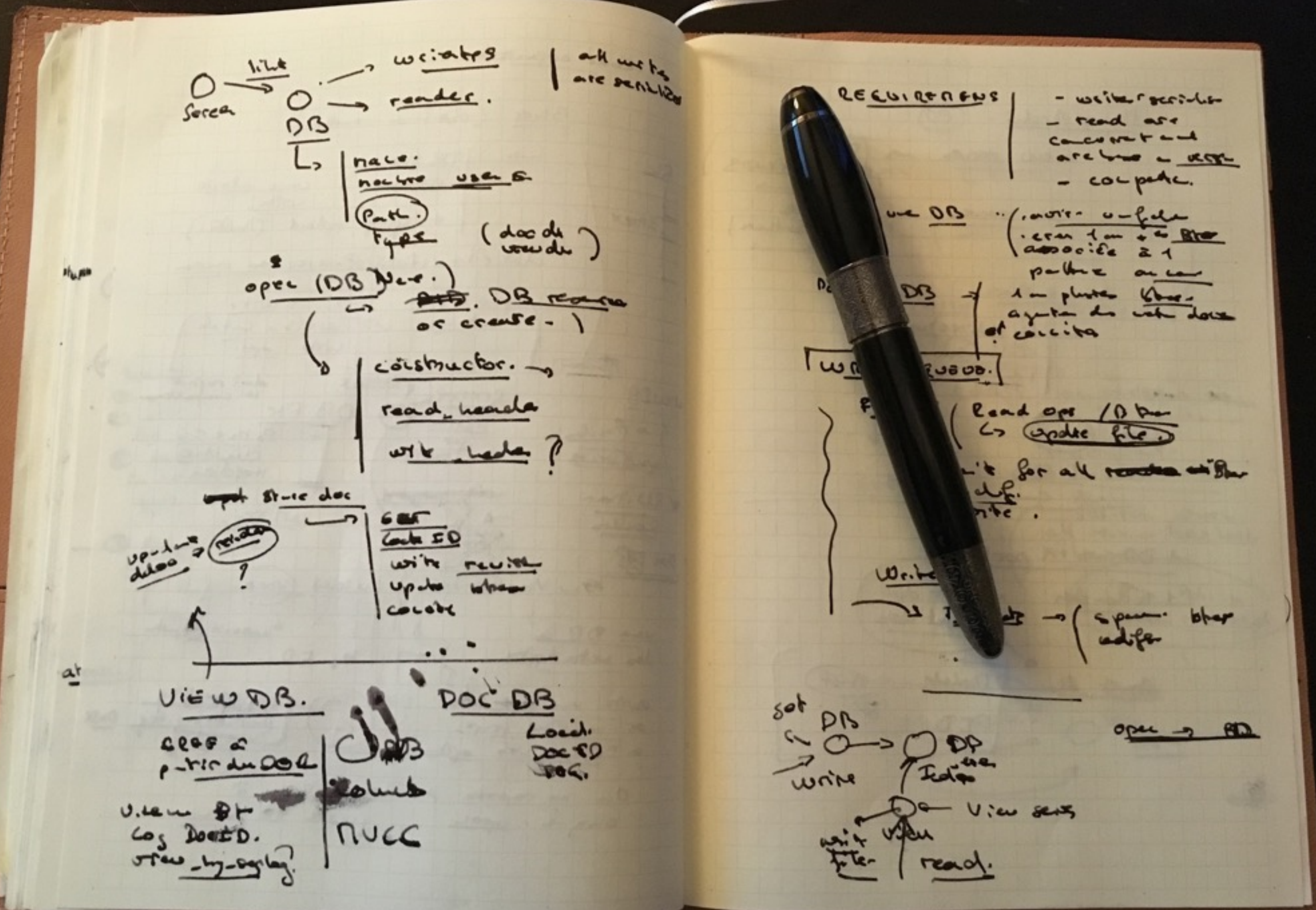
Erlang Factory San Francisco 2016

BUILD A P2P DOCUMENT ORIENTED DATABASE

BARREL

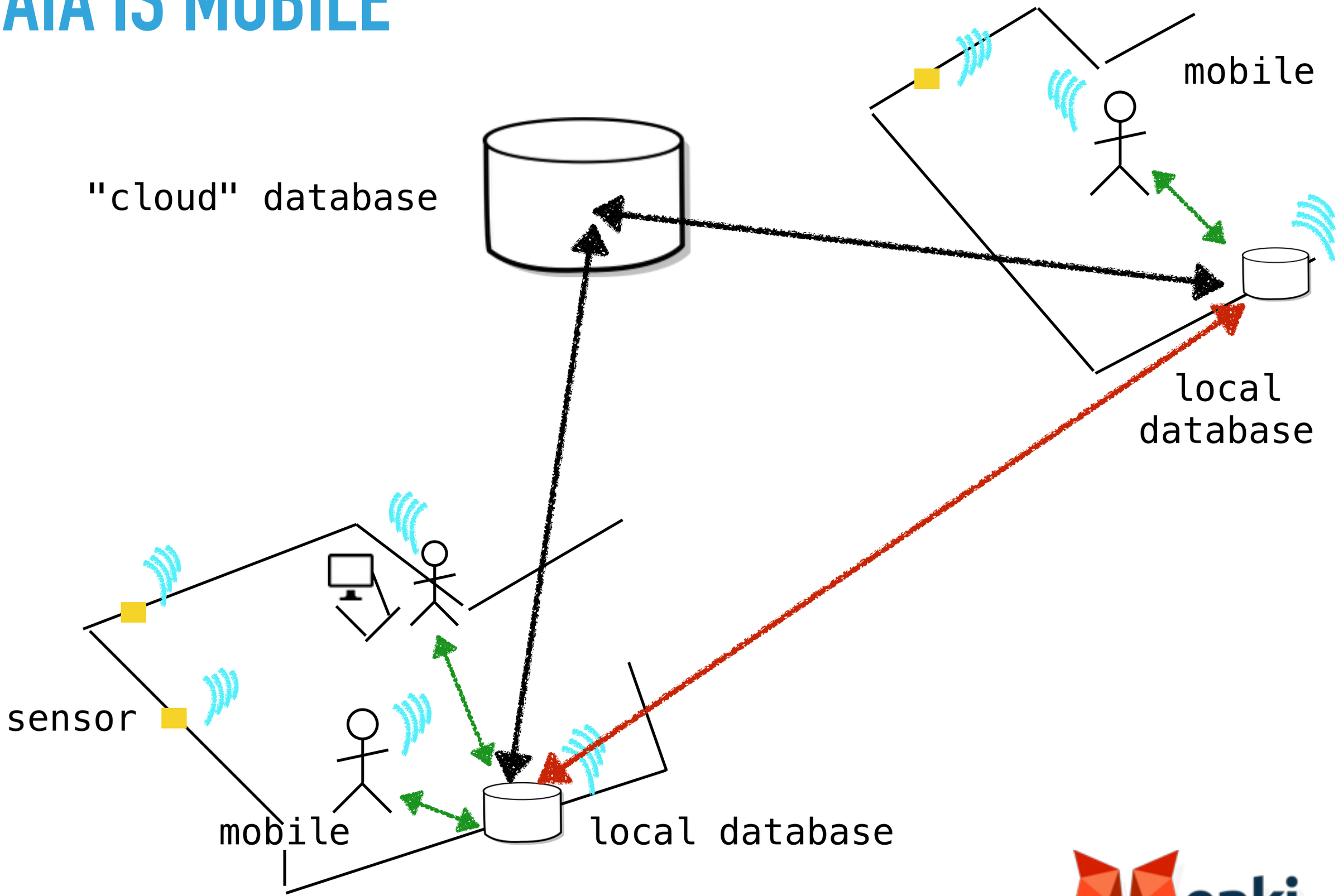
https://barrel-db.org

enki


writes
reader.
DB
name.
Path.
constructor.
read_handle
VIEW DB.
DOC DB
REQUIREMENTS
use DB
read.

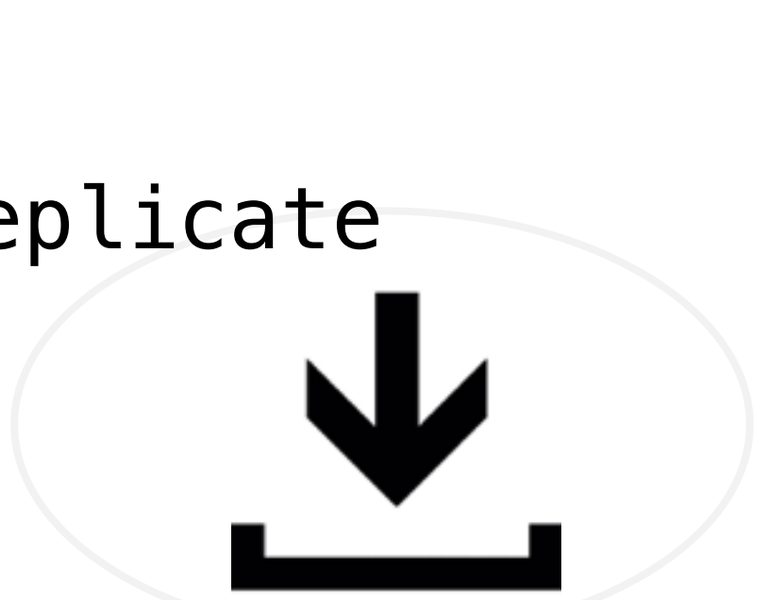
VISION AND CONCEPT

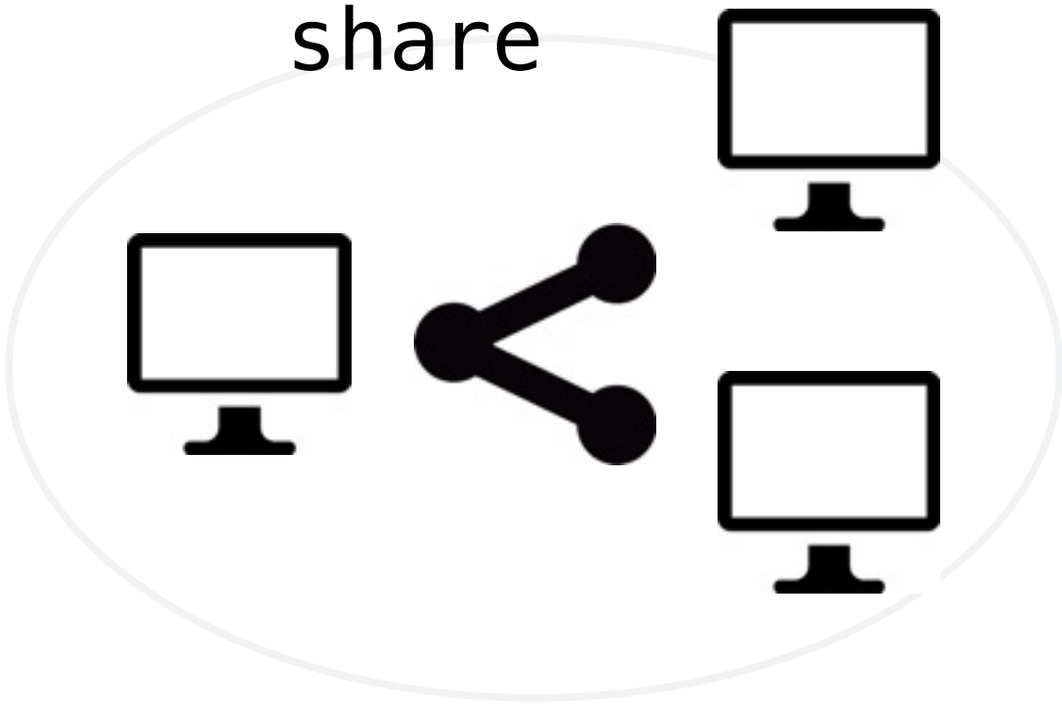
DATA IS MOBILE



enki

PEER TO PEER (P2P)

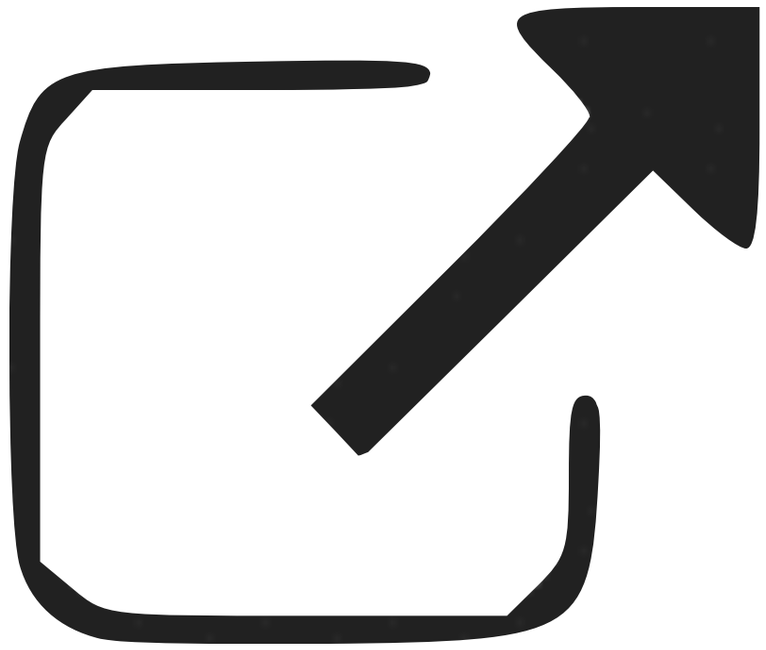




enki

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

WHAT IS BARREL



enki

- a document database
- document are JSON with attachments and links
- changes feed for document and indexes
- replication between any nodes in both way
- views (~ map)
- HTTP 1.1/2 API

WHAT IS BARREL

- DATA: not just blobs
- Replicated APPs
- Couchapps but extended and revisited

REPLICATED APPS

DECONSTRUCT

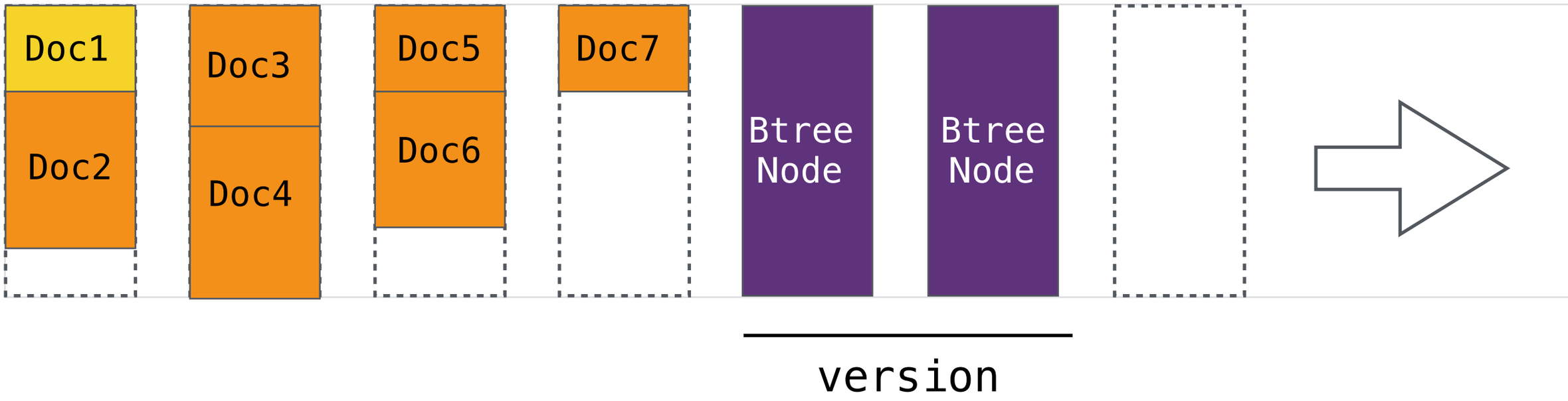
enki

invalid data

Document revision

Block

btree node

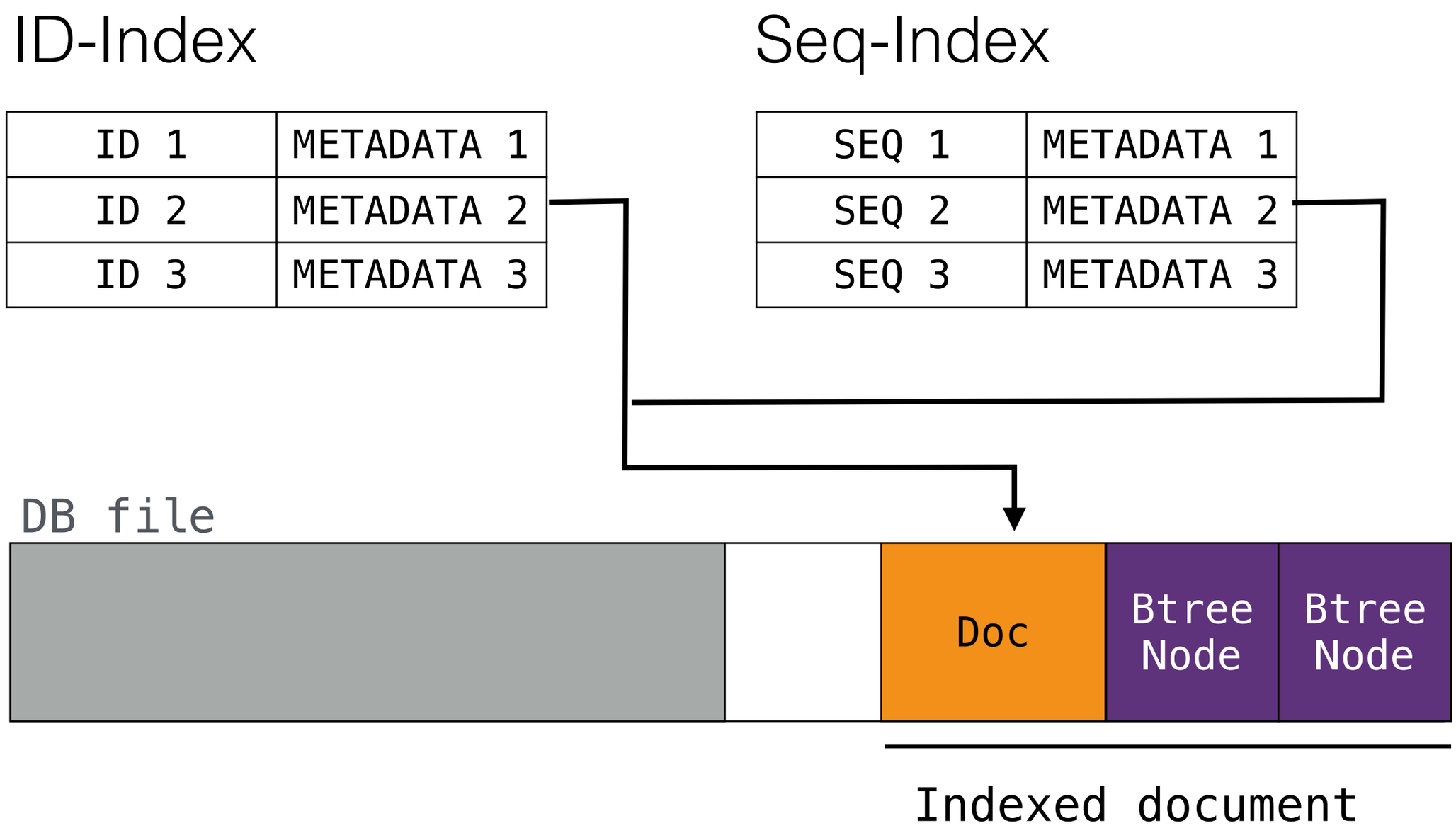


APPEND ONLY & MVCC

enki

- Create a new file to remove the fragmentation
- A race between copy and the addition of new data
- Require at least twice of the storage

THE COMPACTION ISSUE



DOCUMENT STORAGE

- 2 indexes (btree): by sequence, by id
- 1 index for local documents without conflict handling
- A revision tree is stored in indexes pointed to the revisio offset
- The revision is stored in the file separately

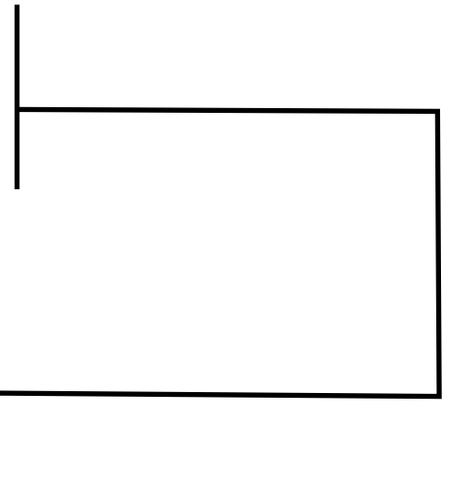
HOW ARE STORED DOCUMENTS

- Reverse index (map)
- Index using a function
- Function in javascripts, erlang, ..
- Incremental index
- Retrieves changes (aka view changes)
- View are regrouped by groups (1 db file/group)

VIEWS

Log-Index

DOCID	View 1	KEY 1	SEQ 1	ADD
		KEY 2	SEQ 2	DEL
	View 2	KEY 1	SEQ 1	ADD



Key-Index

[KEY 1, DOCID]	[VALUE, DOCREV, SEQ]
[KEY 2, DOCID]	[del, DOCREV, SEQ]
[KEY 3, DOCID 2]	[VALUE, DOCREV, SEQ]

SEQ-Index

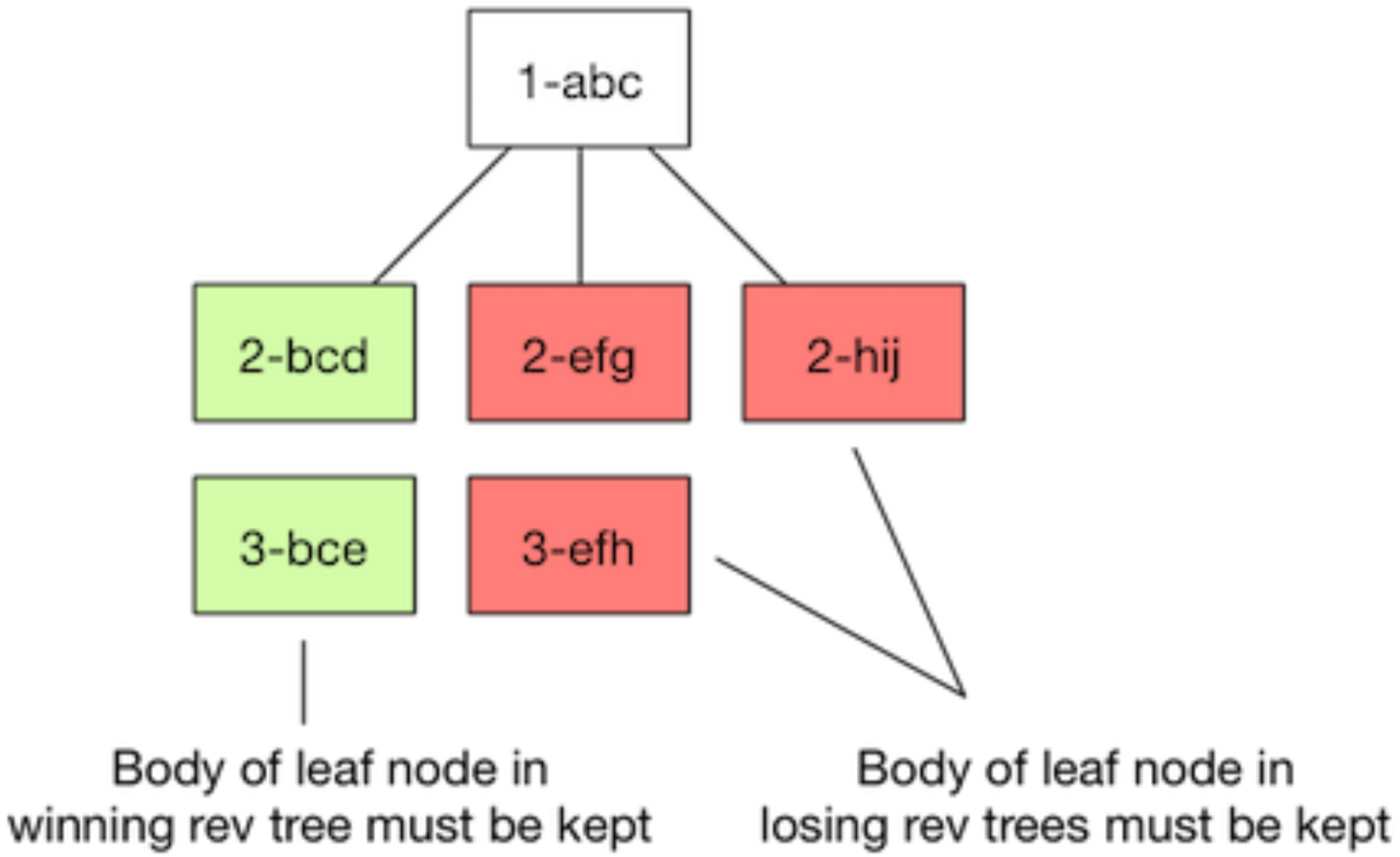
[SEQ 1, KEY]	[VALUE, DOCREV, SEQ]
[SEQ 2, KEY 2]	[del, DOCREV, SEQ]
[SEQ 3, KEY]3	[VALUE, DOCREV, SEQ]

view

VIEW STORAGE

enki

REVISION TREE



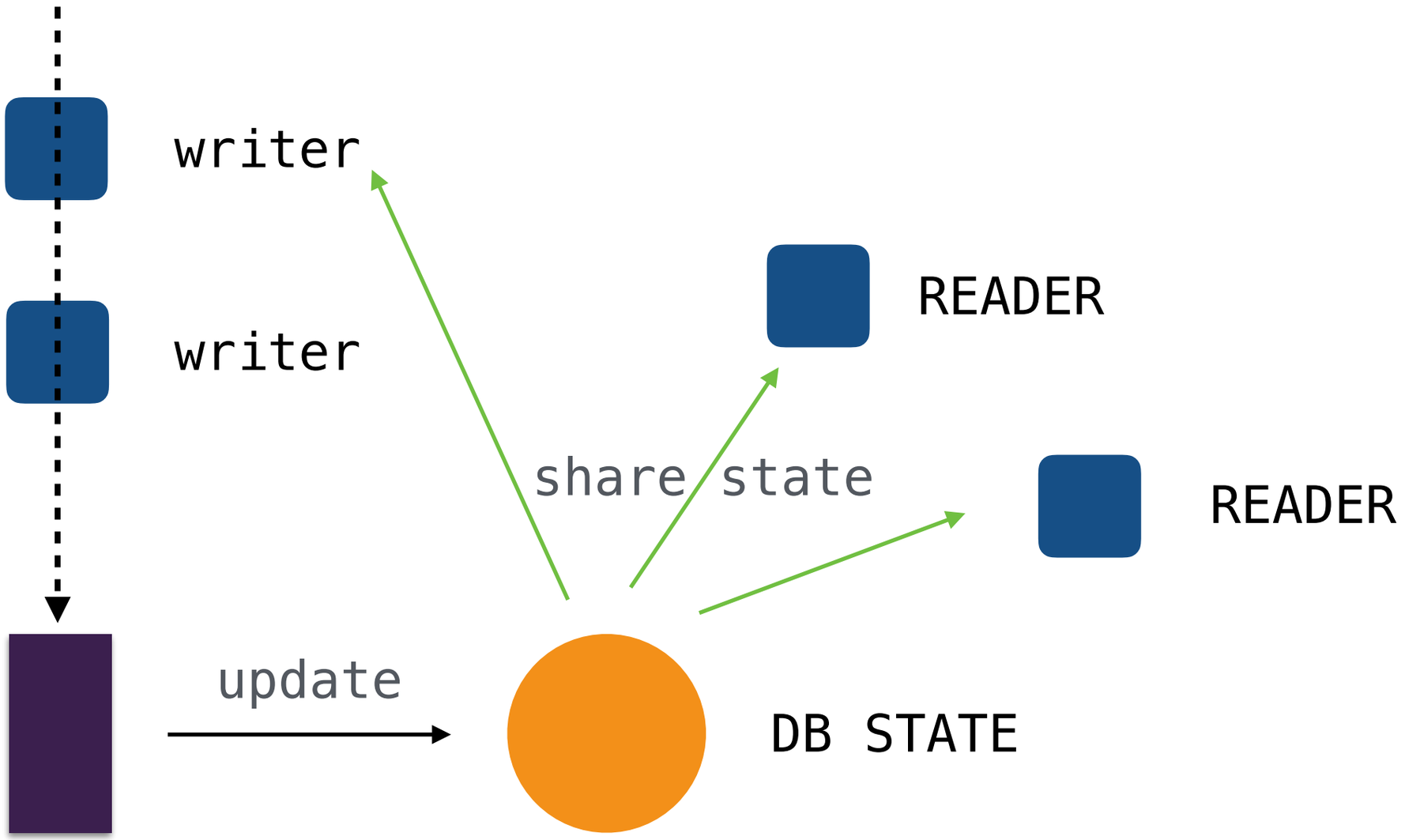
enki

BUILT IN ERLANG

enki

- Write is slow
- Read should not being blocked by writes
- No shared memory
- No atomic integer trick
- Only actors and message passing
- Operations on a doc are atomic

CHALLENGES



READ/WRITE OPERATIONS

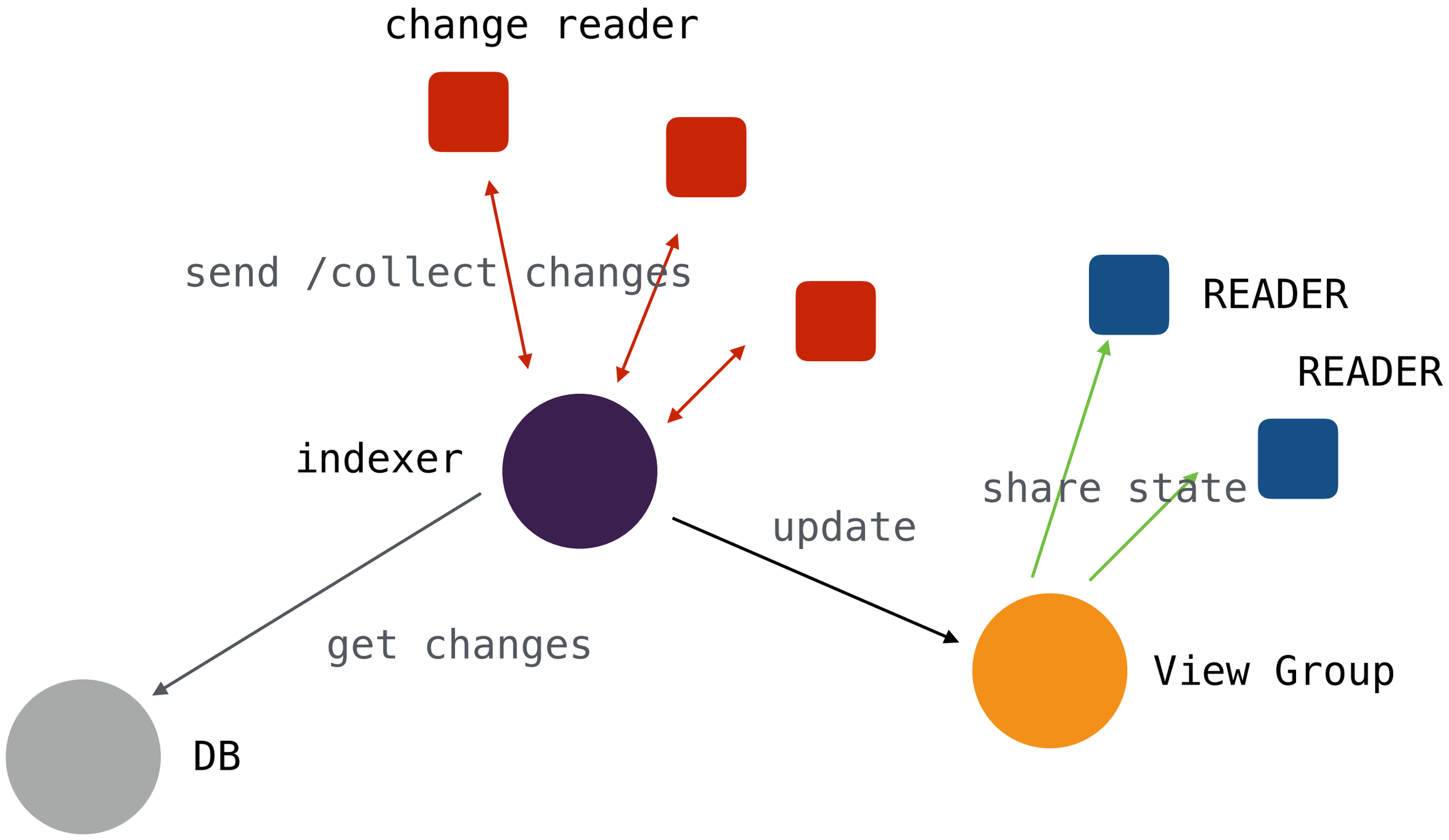
enki

- LRU to cache blocks
  https://github.com/barrel-db/erlang-lru
- 1 File process, Operations are limited
- DB users are linked to the database process
- Optional Write buffer to reduce the latency
- Optional wal

READ/WRITE OPERATIONS

- STORE SEGMENTS of data for compaction
- IO is "relatively" slow in erlang
- USE a “native KV store” as a nif.

SPEEDUPS



INDEX OPERATIONS

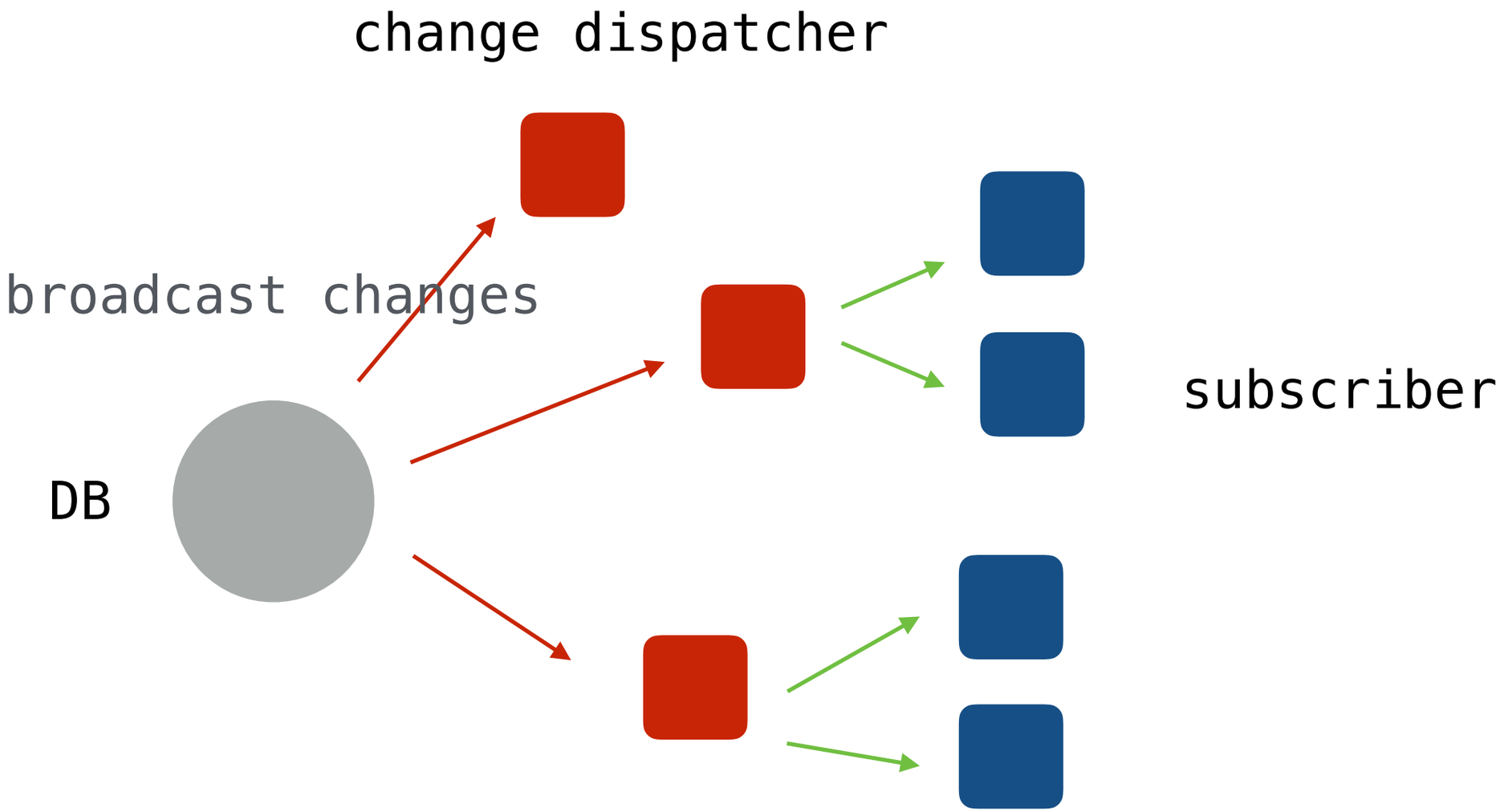
enki

- Credit Flow Based
- The View group keep the state
- View group is created on demande
- kept open until it has readers
- Indexer ask for updates
- Read functions (Map functions) are processed in //

INDEX OPERATIONS

- Added 2 features:
  - MOVE: move doc(s) to another node or database (like copy but with delet
  - User hooks functions (run in background) using hooks:
    https://github.com/barrel-db/hooks
- Partition on demand
- Decision depends on the application needs

NEW FUNCTIONS



CHANGES HANDLER

enki

- Use the sequence index
- changes load balancing
- consumer subscribe on patterns (delete, update, ...)
- Create changes Load Balancer on demand

- Allows remote nodes to subscribe to a queue
- Based on primer (release on March 2016)

CHANGES EVENTS

- Use the sequence index
- changes load balancing
- consumer subscribe on patterns (delete, update, ...)
- Create changes Load Balancer on demand

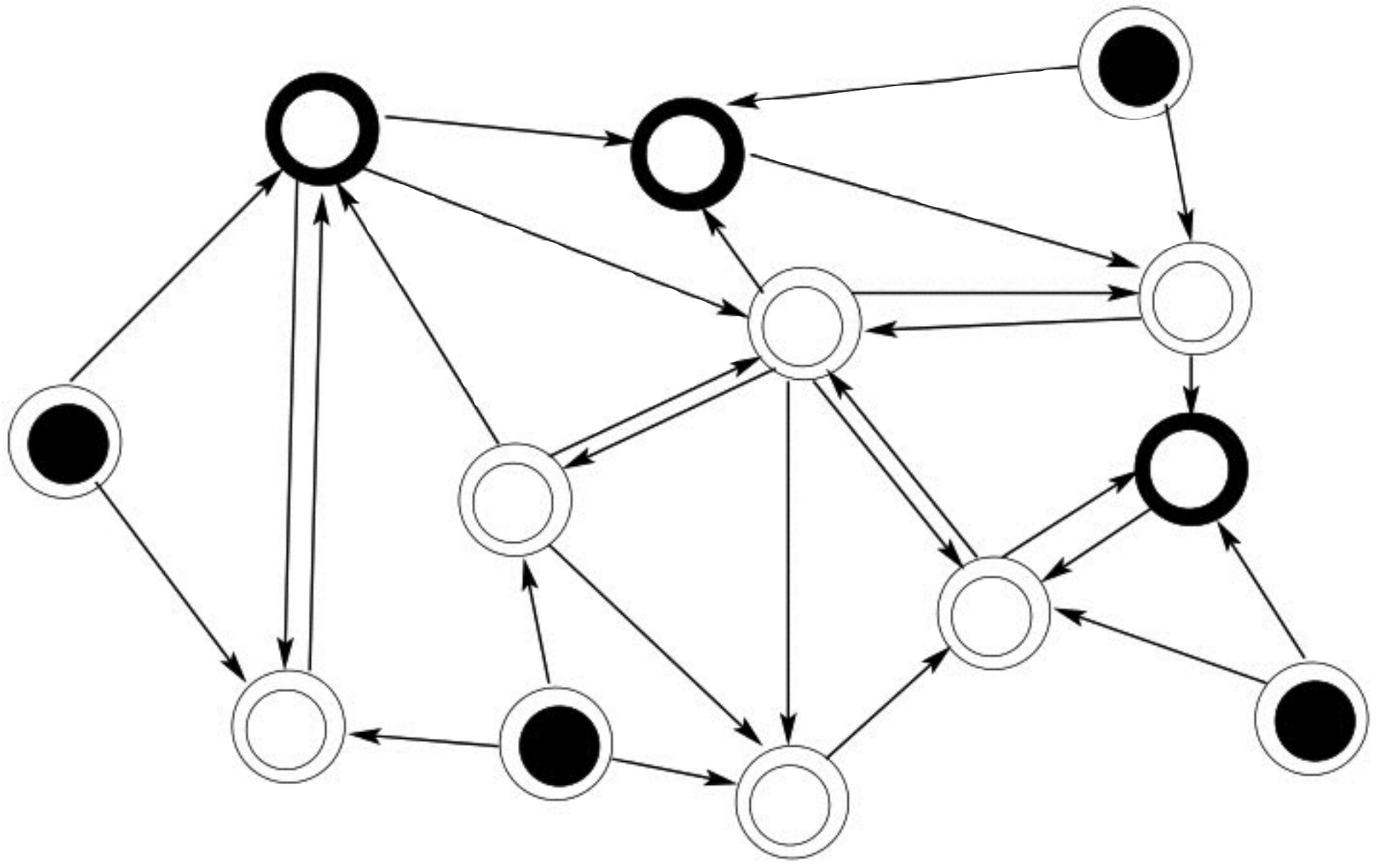
- Allows remote nodes to subscribe to a queue
- Based on primer (release on March 2016)

CHANGES EVENTS

- inherited the HTTP api in mochiweb
- small changes to makes the server more resilient
- chatterbox
- wip in cowboy.
- yaws ?

HTTP API

P2P



enki

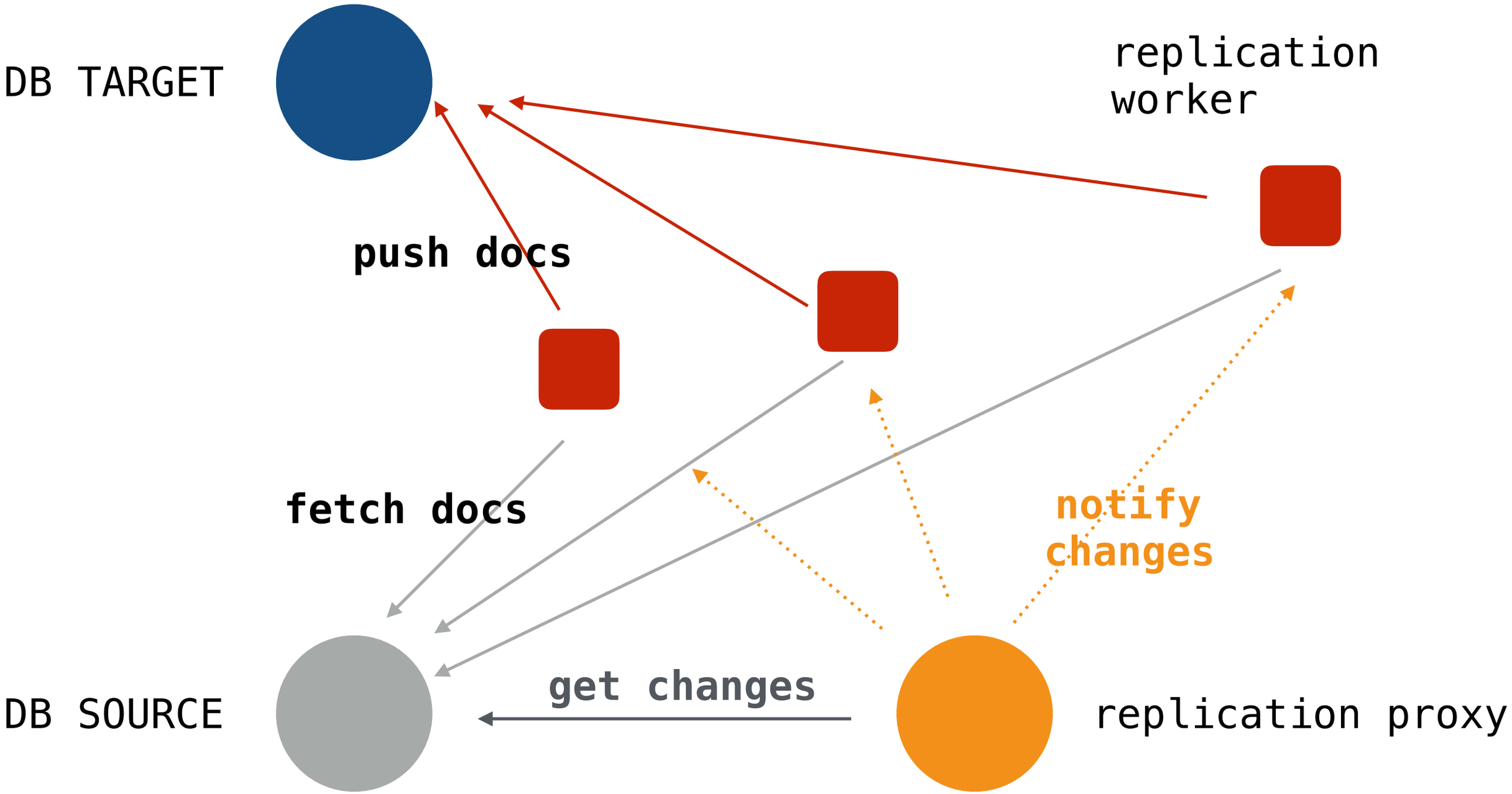
- Over HTTP
- Replication is the core
- Each nodes can replicate each others
- PUSH/PULL
- Chained replication

P2P

enki

- Based ont the change feed
- fetch the revisions and their attachments not present on the node
- continuous or not
- try to collect multiple docs at once
- use hackney: http://github.com/benoitc/hackney
- Use a Flow-based pattern instead of a classic pool

REPLICATION



REPLICATION OPERATIONS

enki

- Replication state is stored a least on one node
- checkpoints
- get the revisions not actually stored on the nodes (“_rev_diffs”
- the replication proxy maintains routes
- build replication chains, by replicating status

REPLICATION

?

enki

Barrel

HTTPS://BARREL-DB.ORG

Enki Multimedia

HTTP://ENKIM.EU