MD-SAL Clustering Internals

Moiz Raja

Open Daylight Summit 2015

My Collaborators

Tom Pantelis

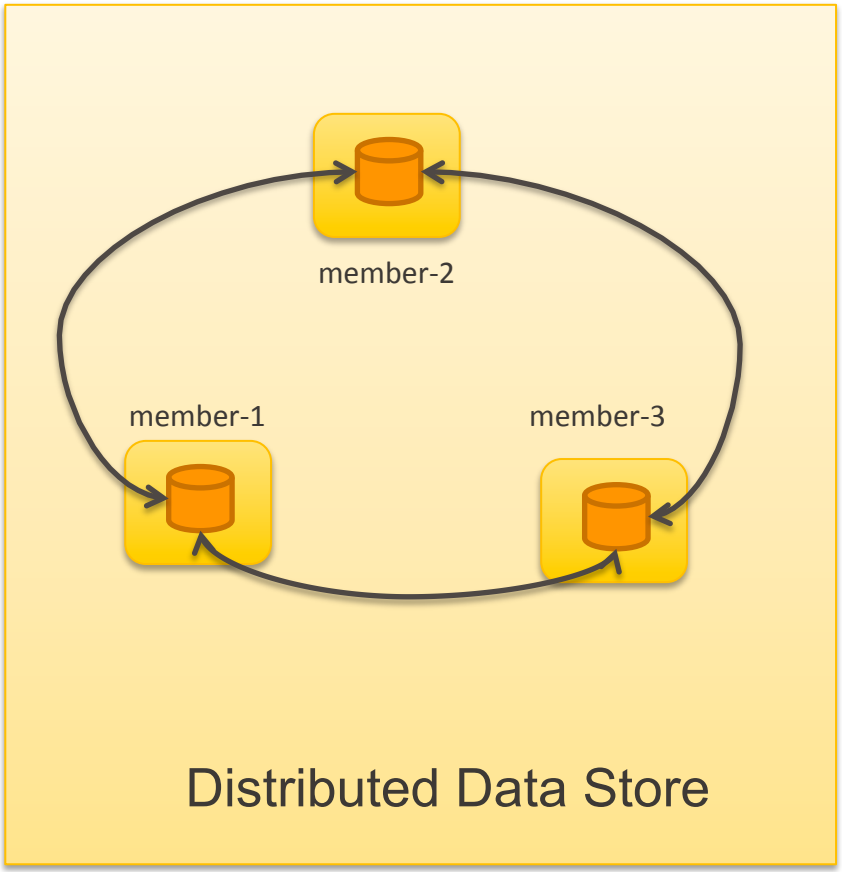
- Abhishek Kumar
- Basheeruddin Ahmed
- Colin Dixon
- Harman Singh
- Kamal Rameshan
- Robert Varga
- Tony Tkacik
- Luis Gomez
- Phillip Shea
- Radhika Hirannaiah
- and many more…

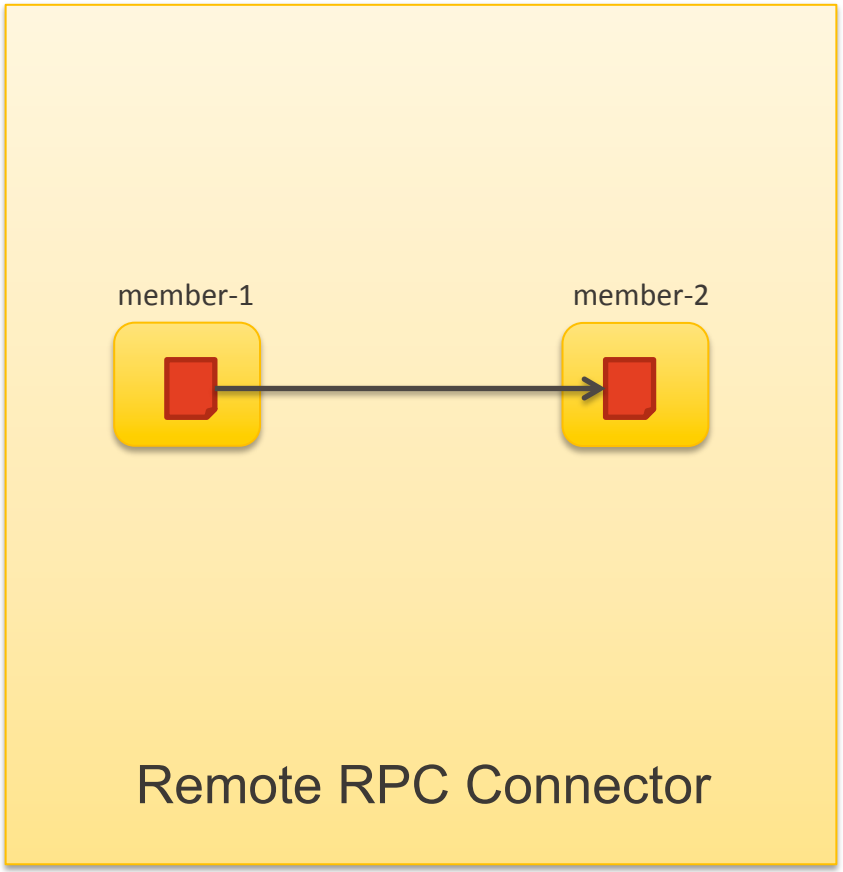
Agenda

- Architecture
- Modules
- Flows
- Diagnostics
- Questions

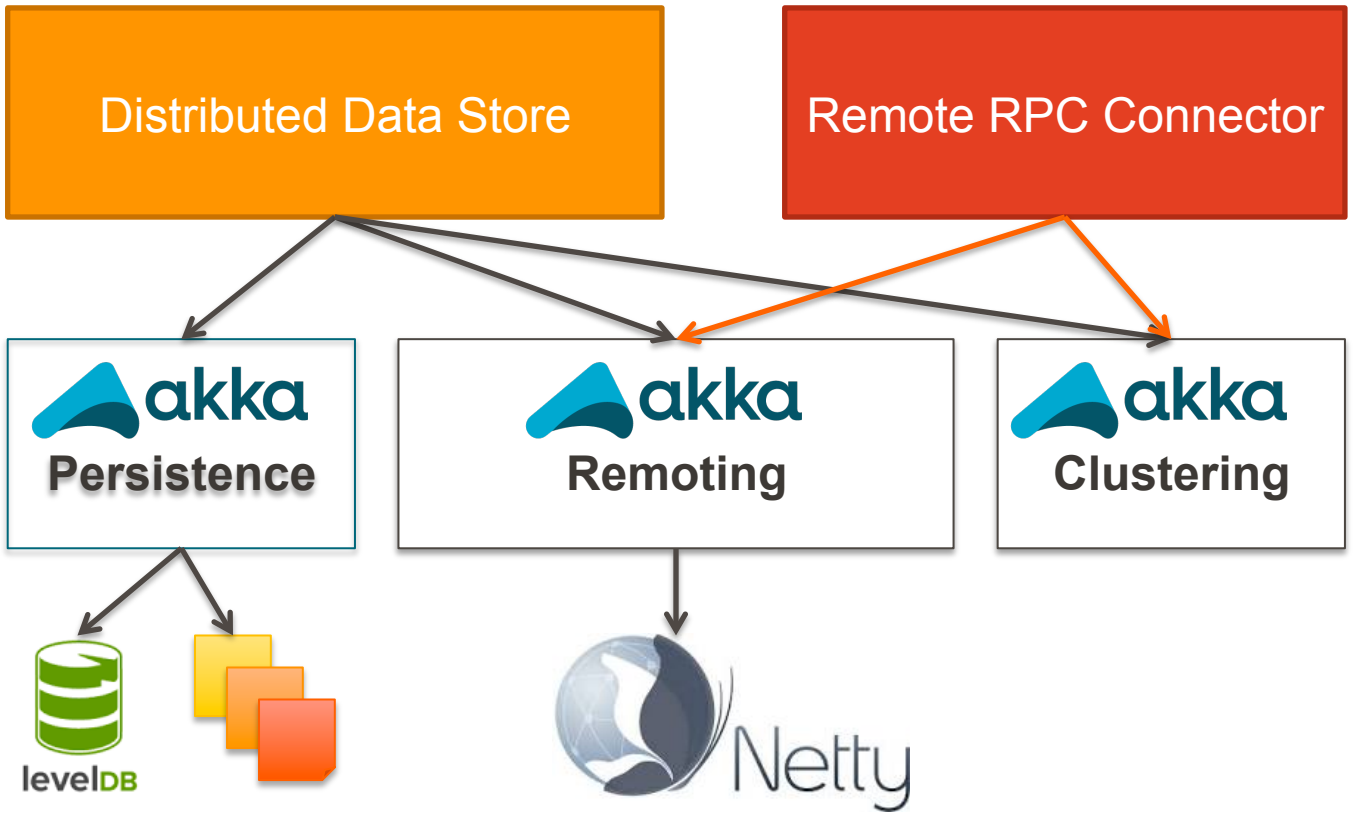
Architecture

Subsystems

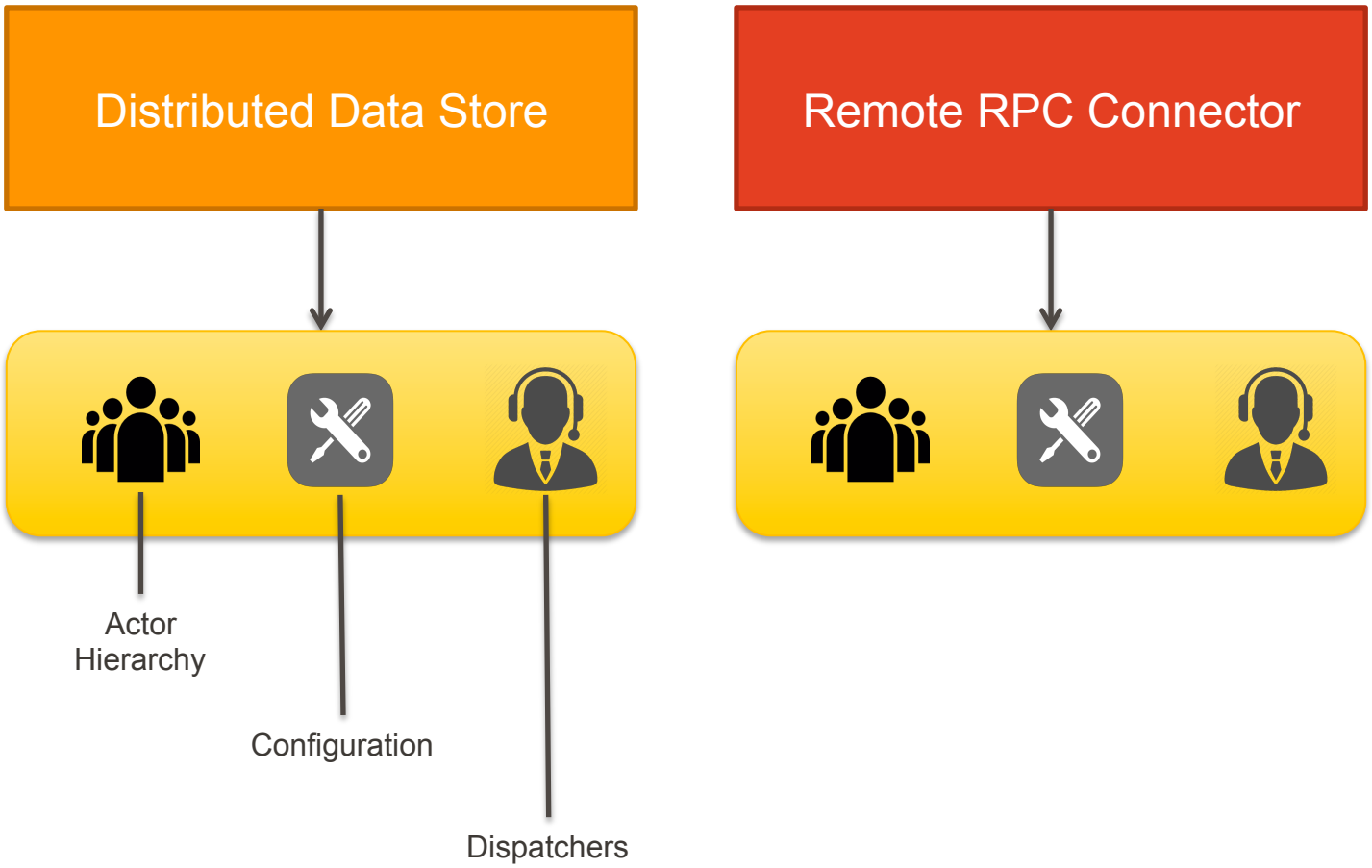

member-2
member-1
member-3
Distributed Data Store


member-1
member-2
Remote RPC Connector

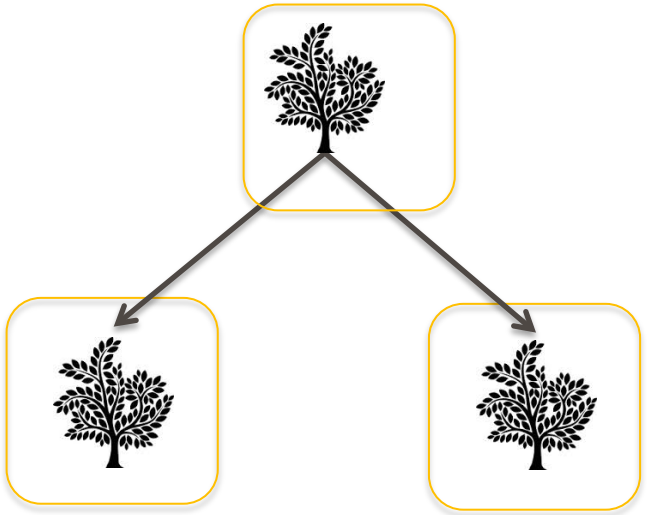
High Level Architecture

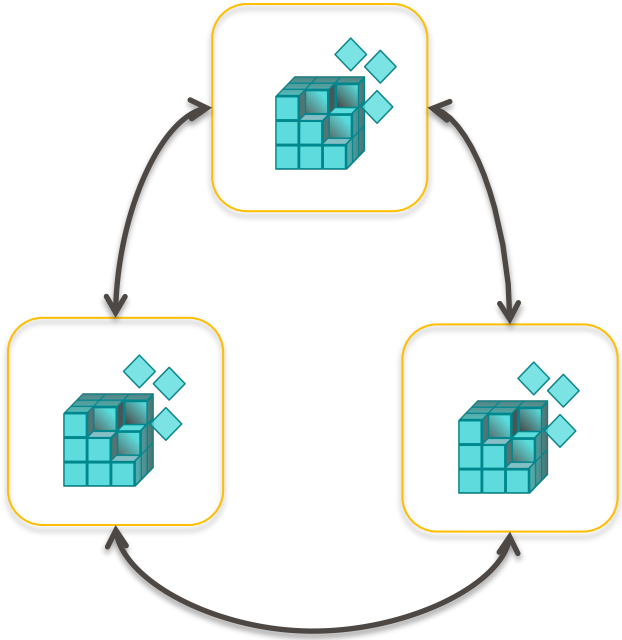


Actor Systems



Data Synchronization



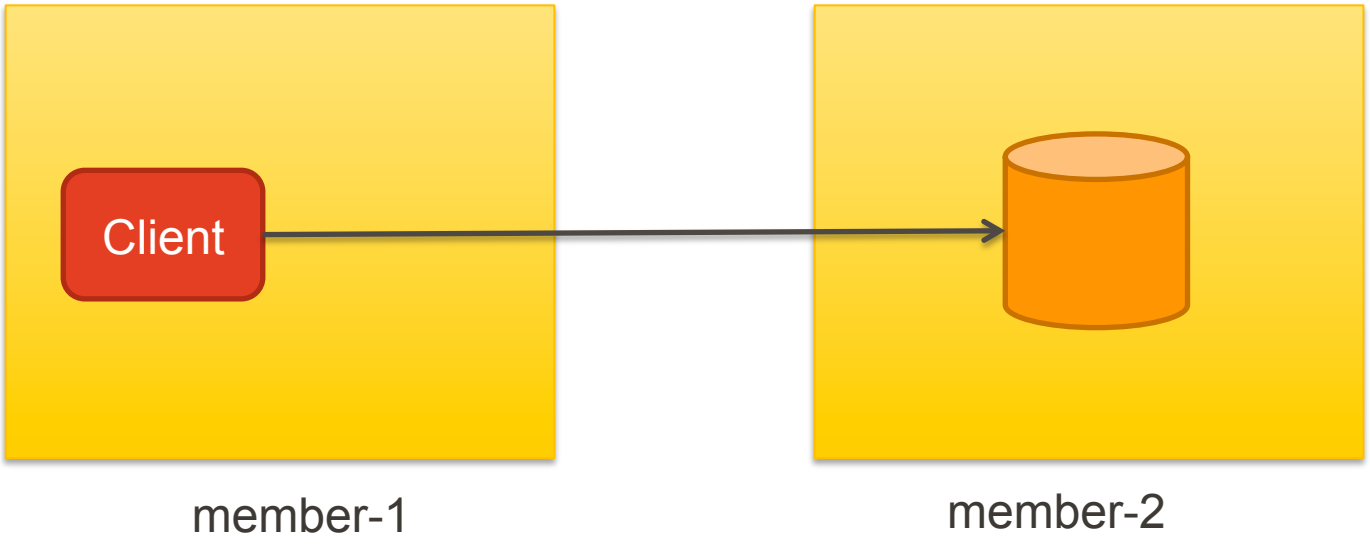


Data store
Synchronized Data Tree
Raft for Distributed Consensus

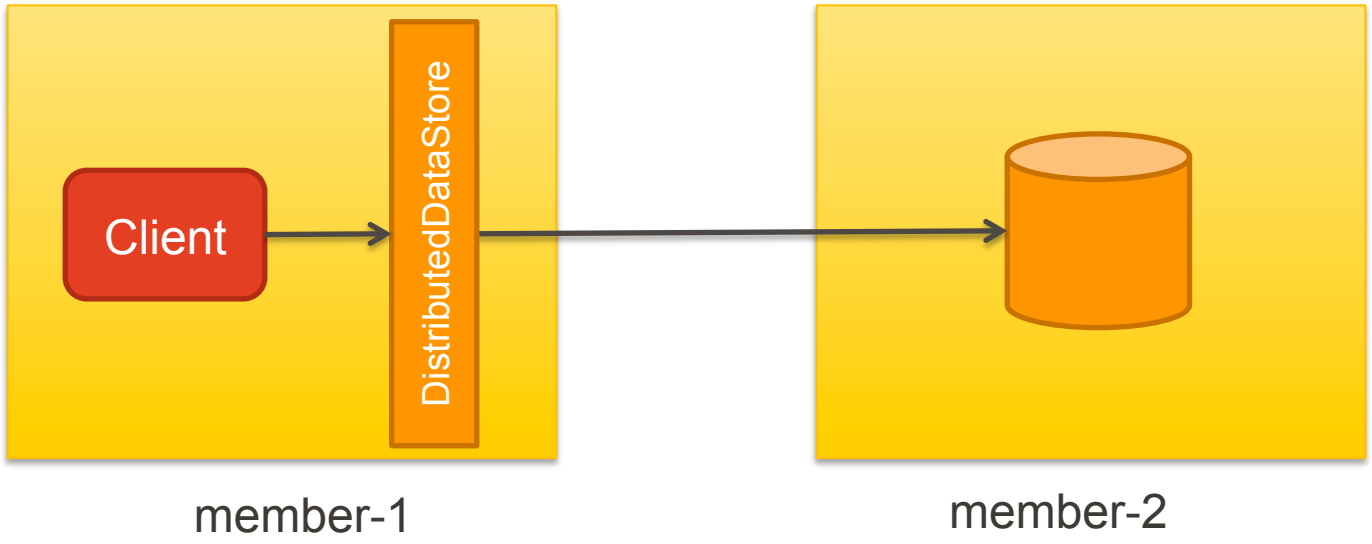
Remote RPC
Synchronized RPC Registry
Gossip for data distribution

Distributed Data Store Architecture

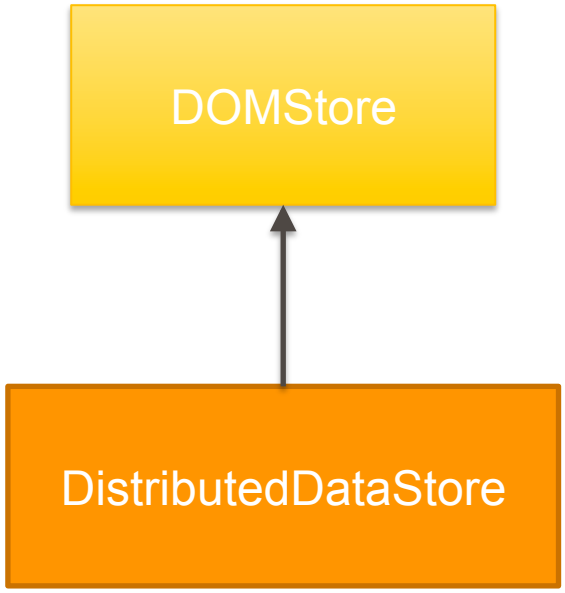
Accessing Remote Data



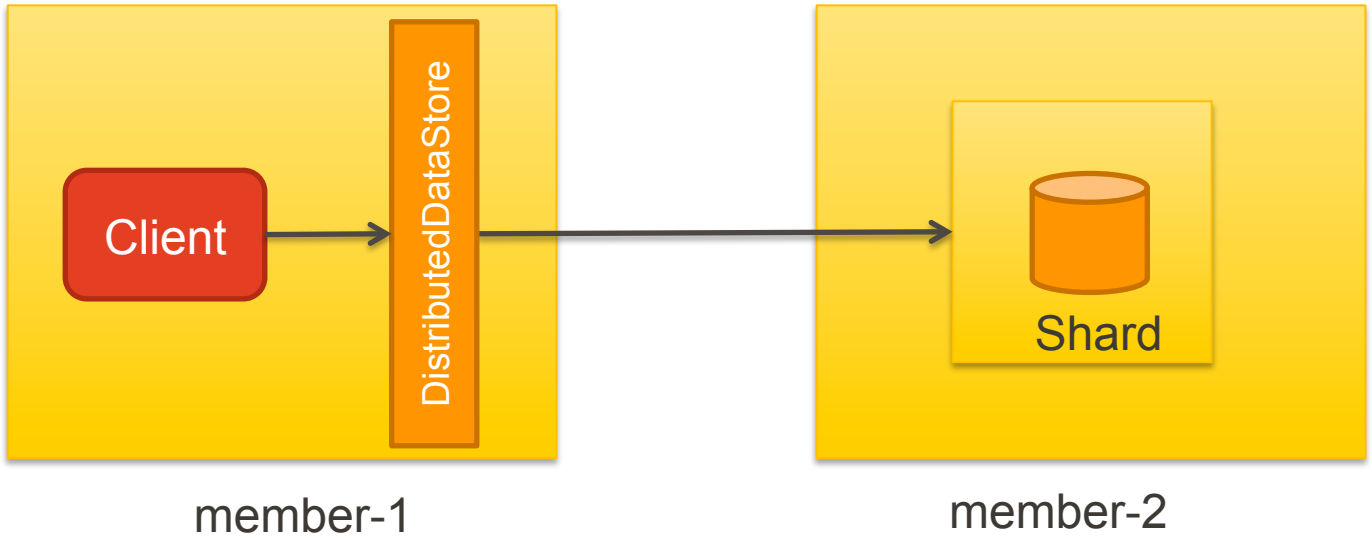
Location Transparency



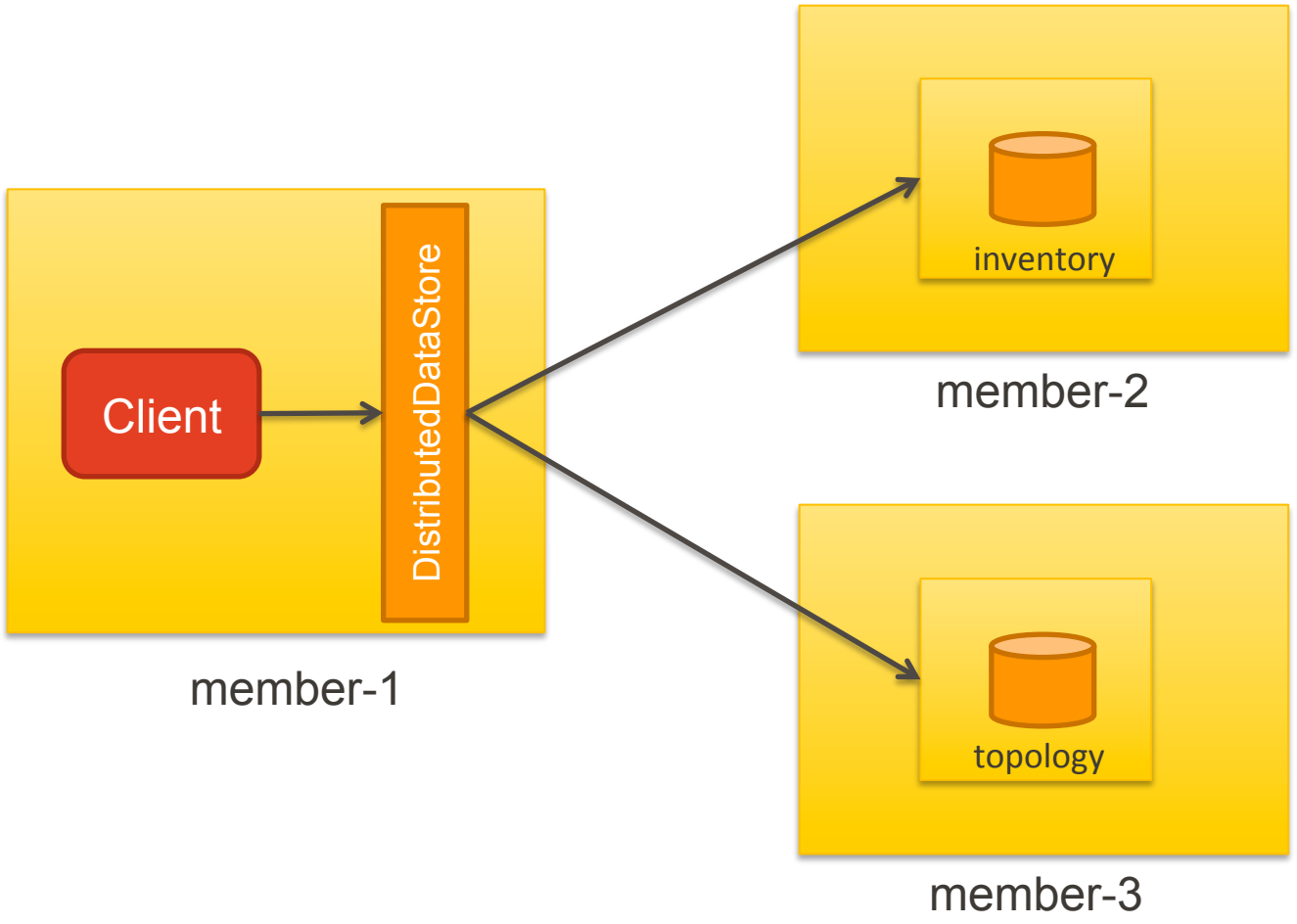
DistributedDataStore



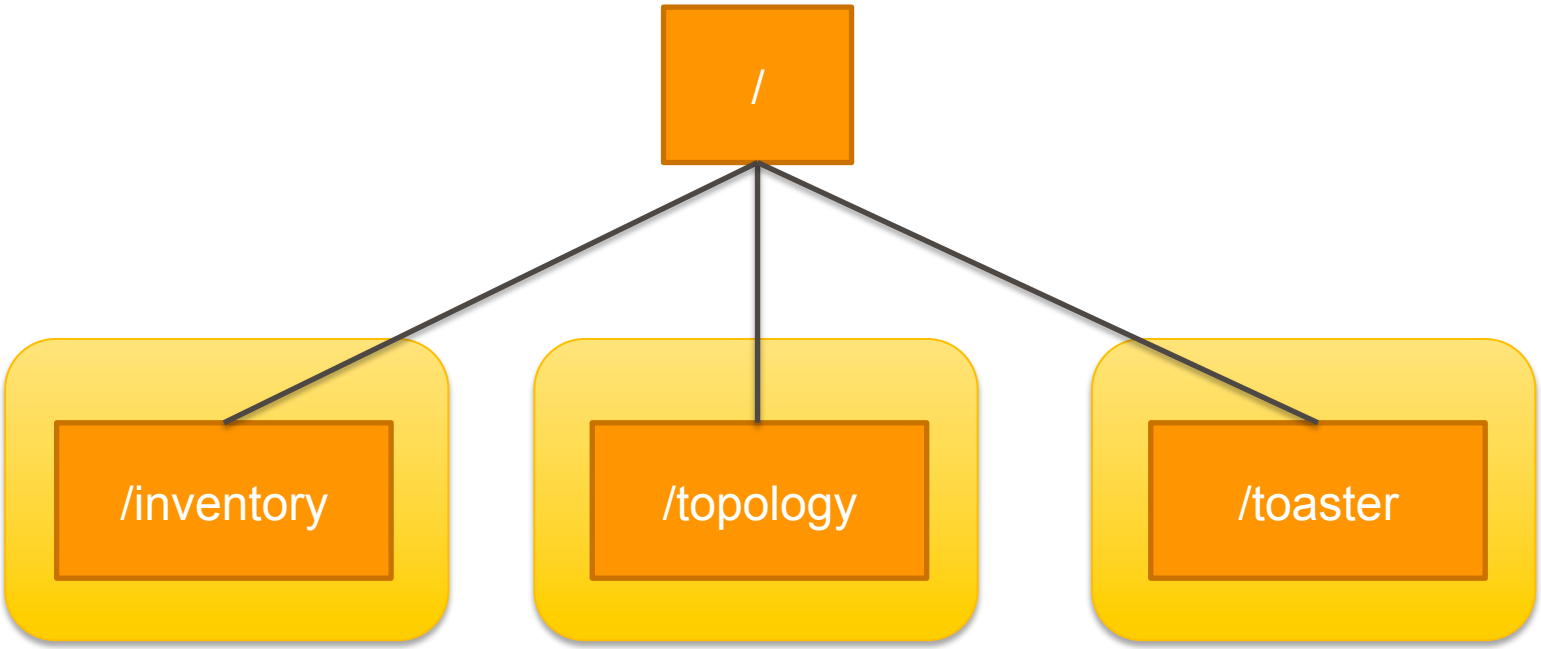
Communication



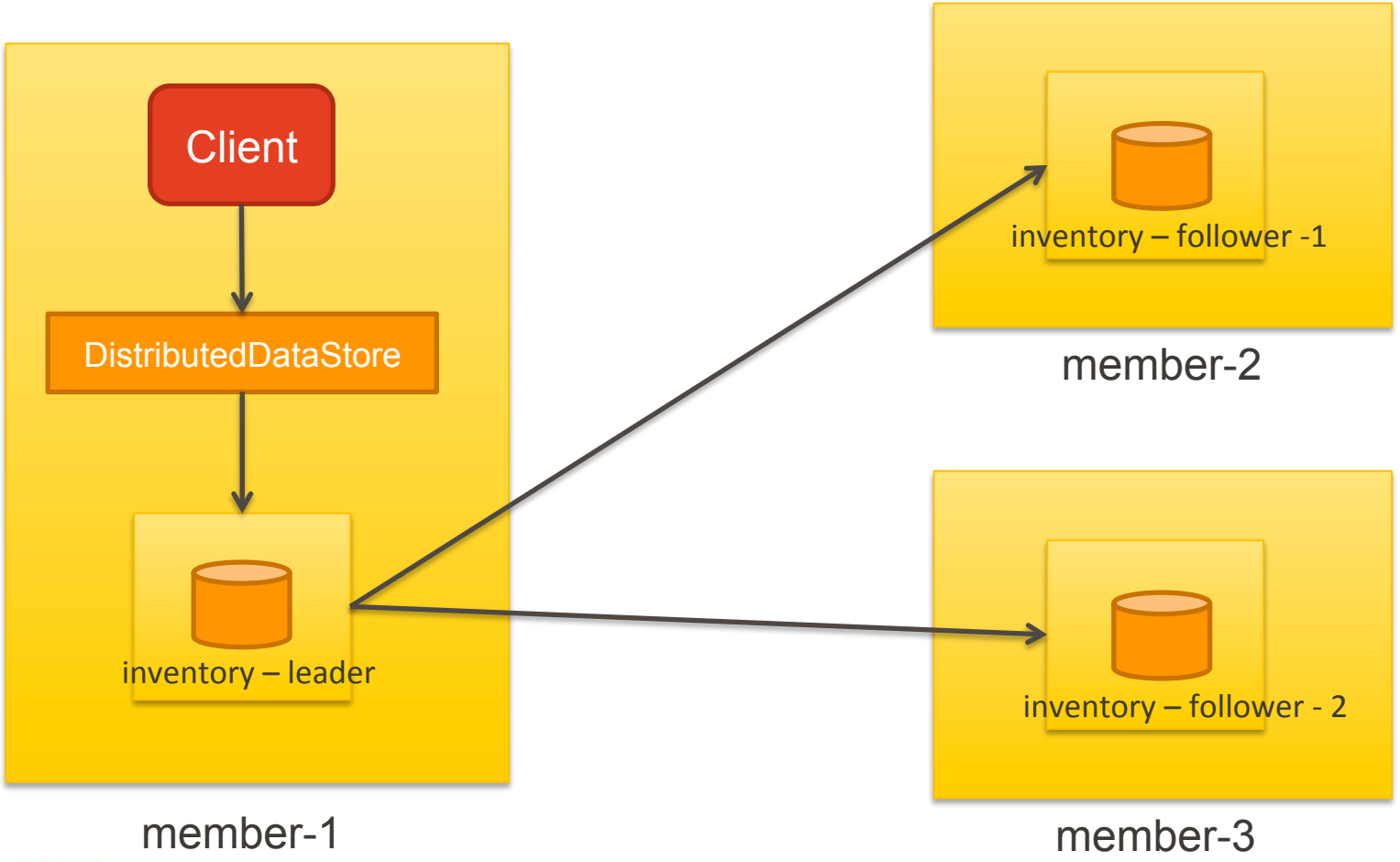
Data Distribution



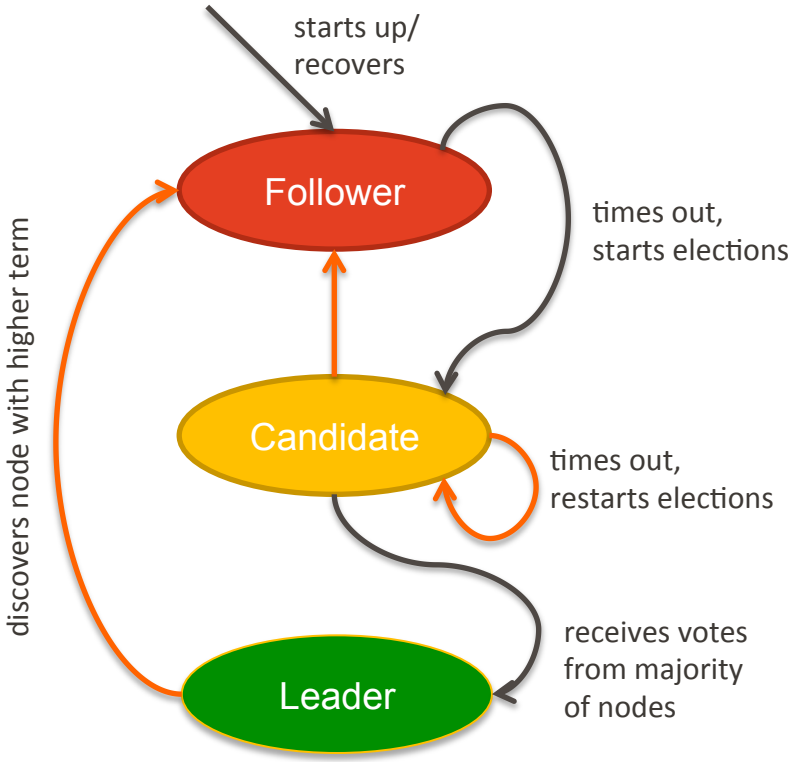
Module Based Shards



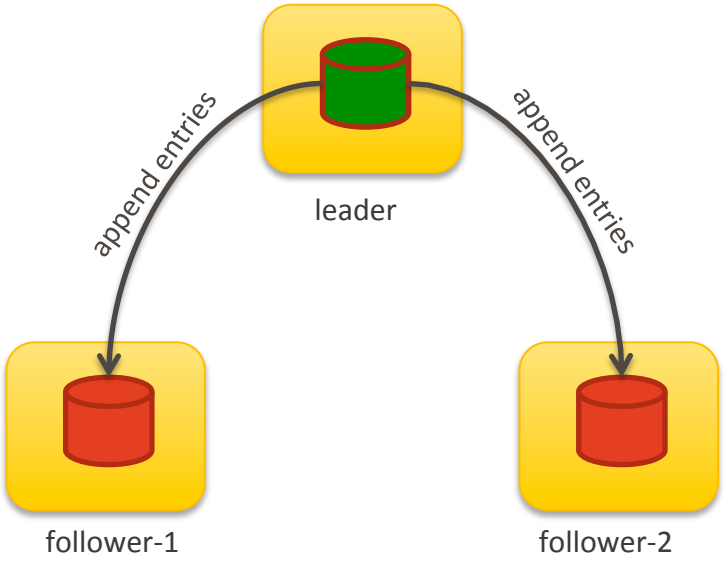
HA


Client
DistributedDataStore
inventory – leader
member-1
inventory – follower -1
member-2
inventory – follower - 2
member-3

Raft Distributed Consensus

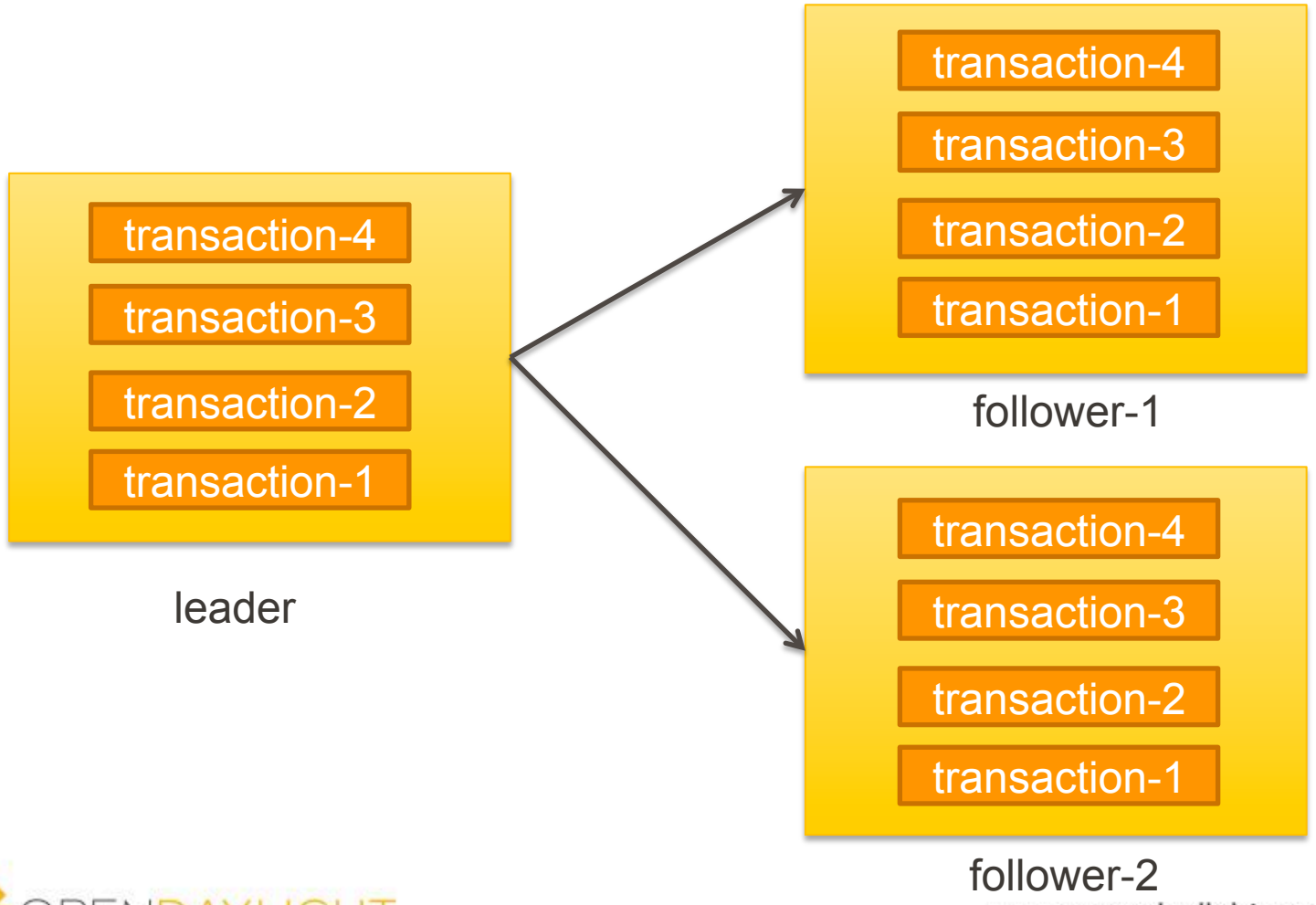


Election

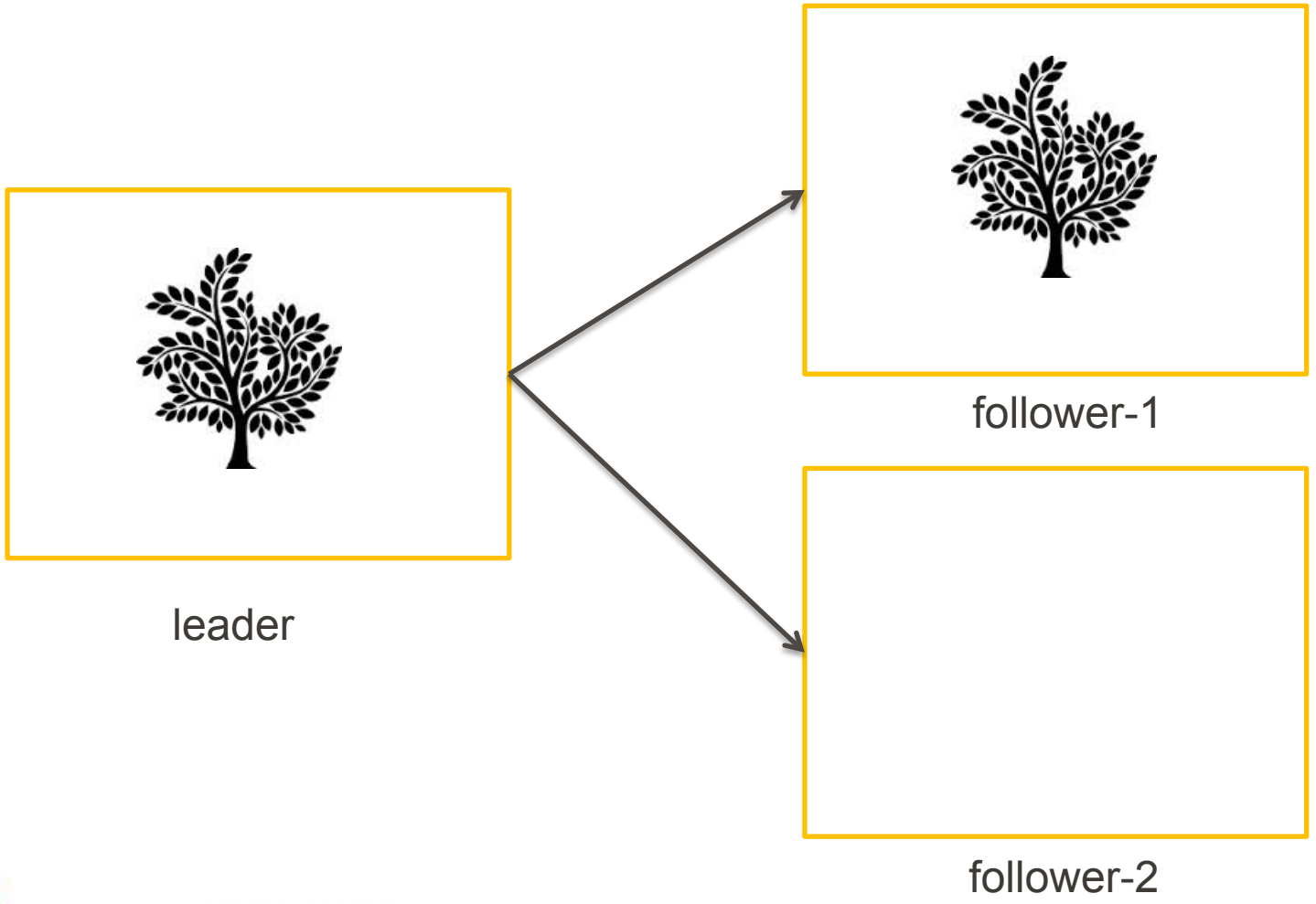


Replication/Consensus

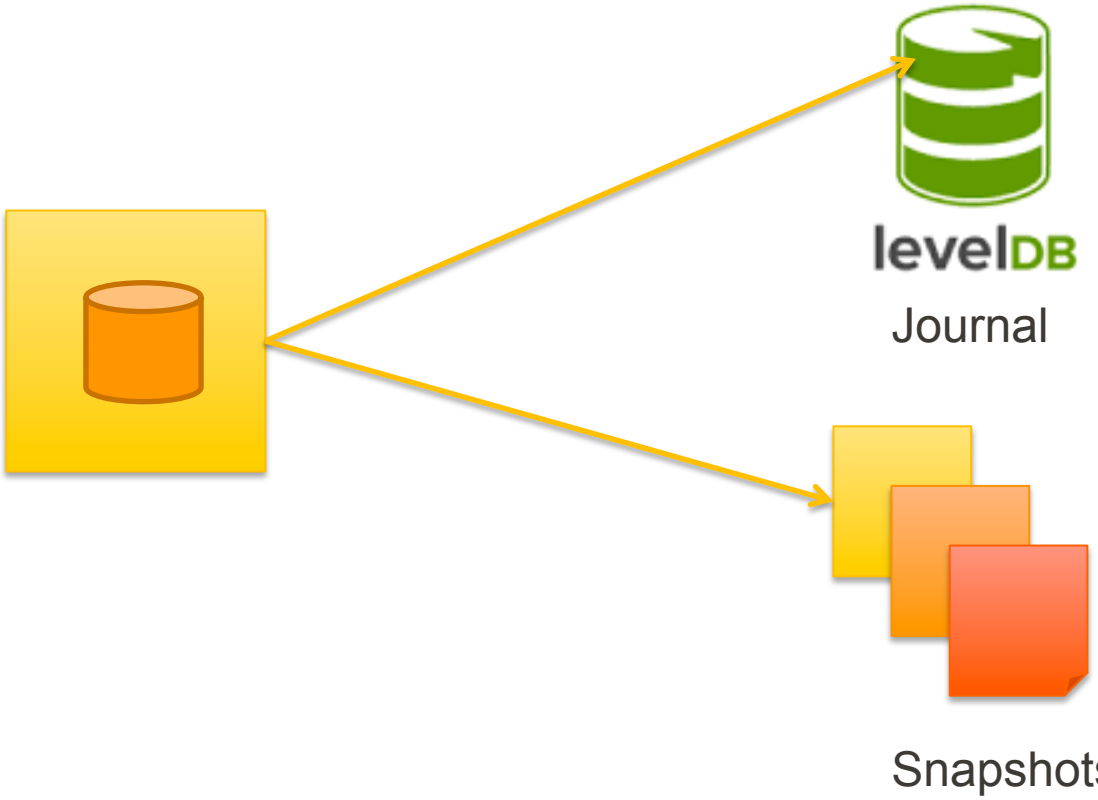
Journal replication



Snapshot Replication

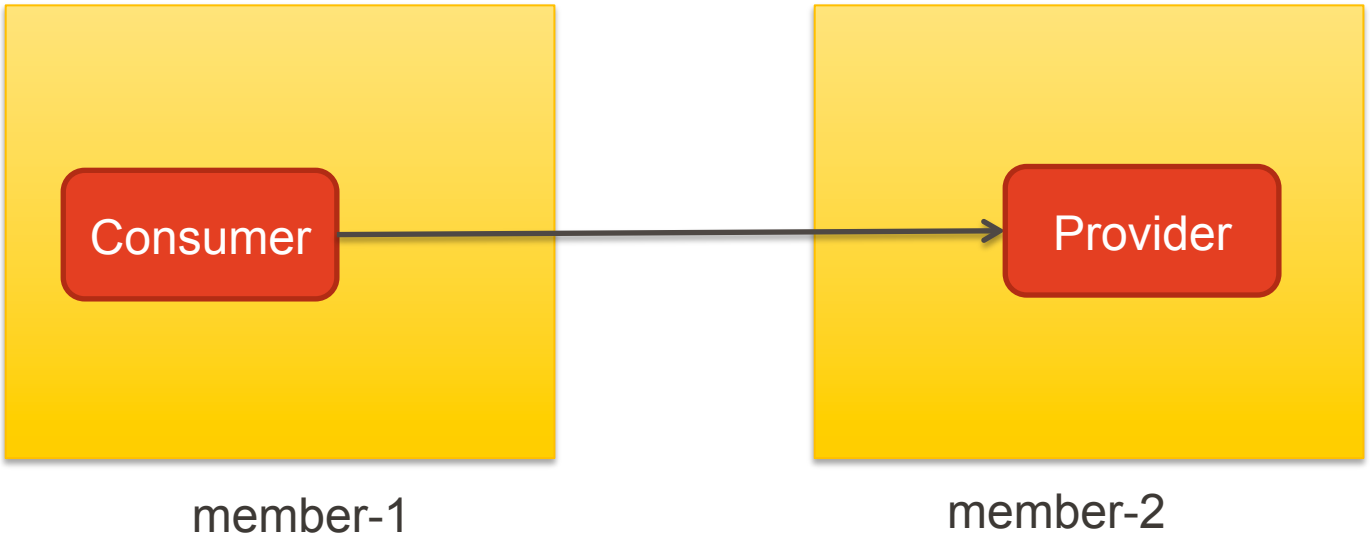


Durability/Recovery

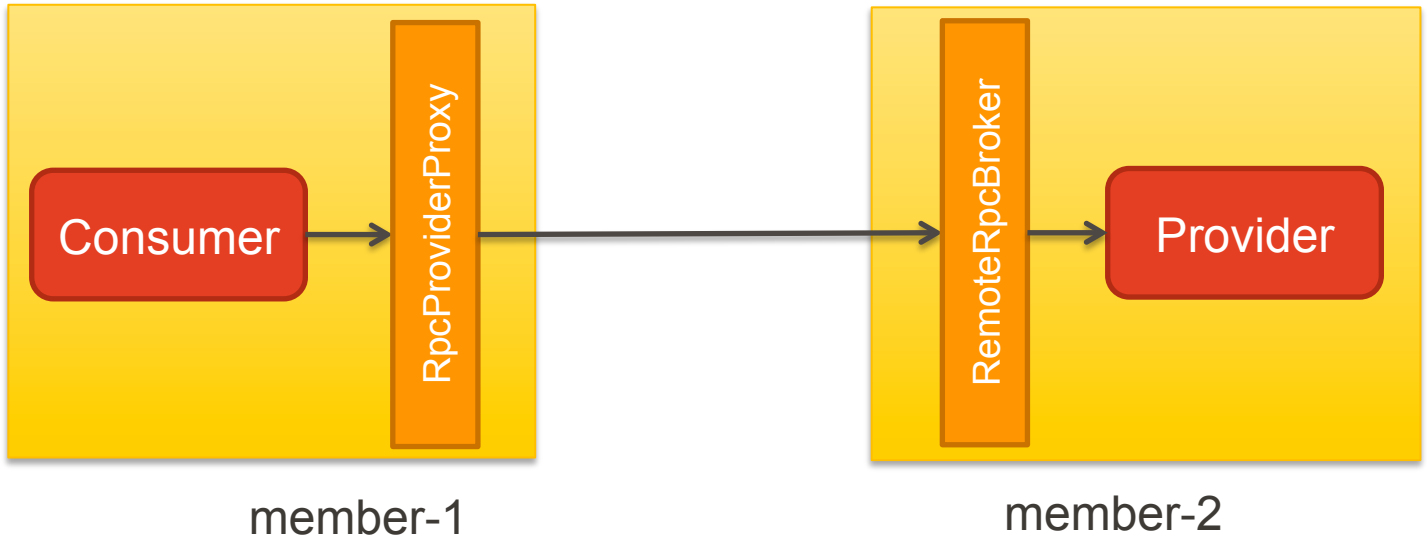


Remote RPC Architecture

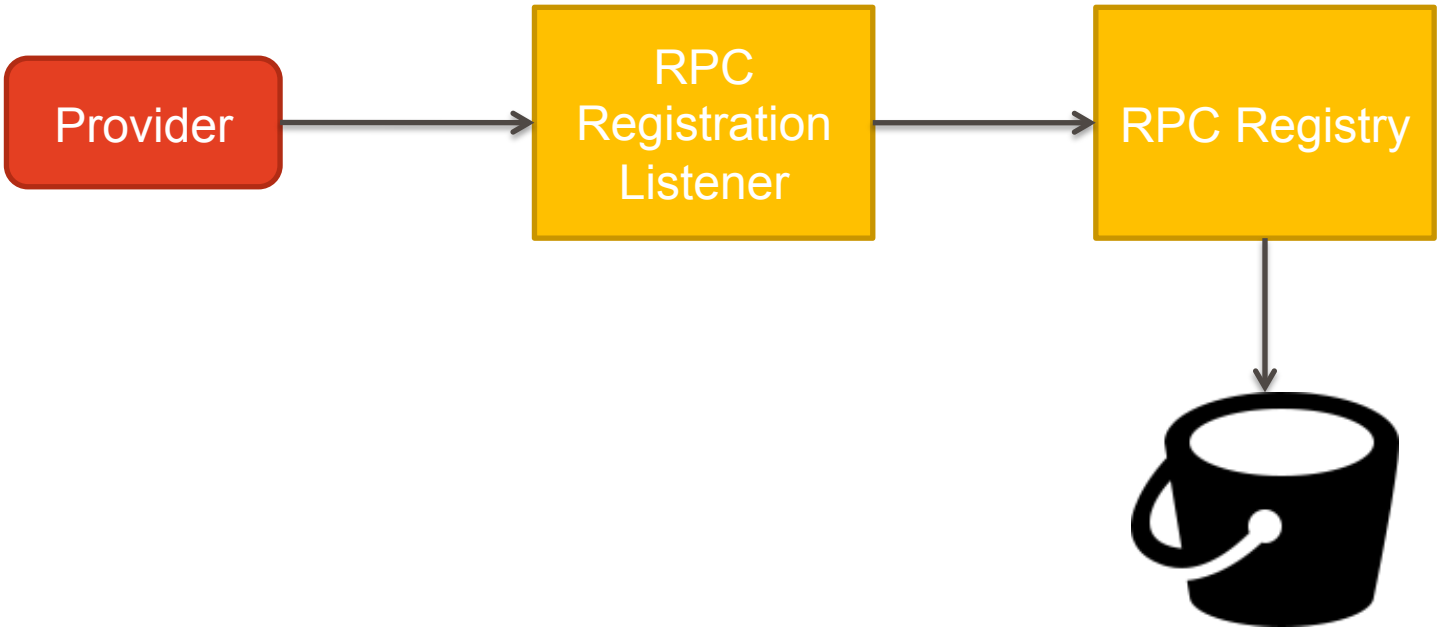
Invoking a Remote RPC



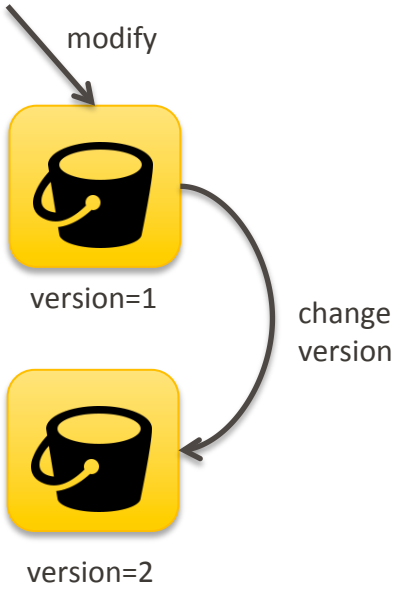
Location Transparency



RPC Registry

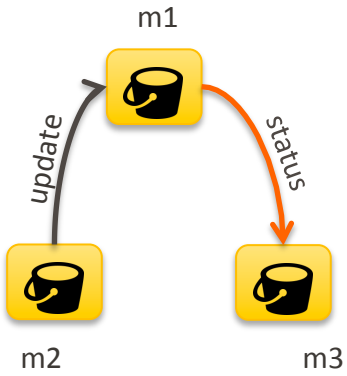


RPC Registry Replication - Gossip

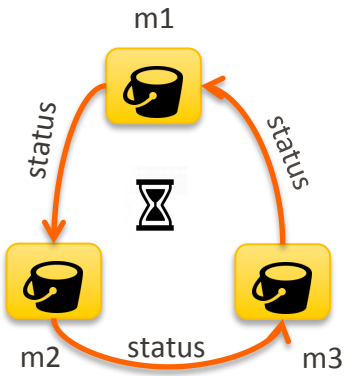


Local bucket updates
change version

All buckets and their
versions known to all
members



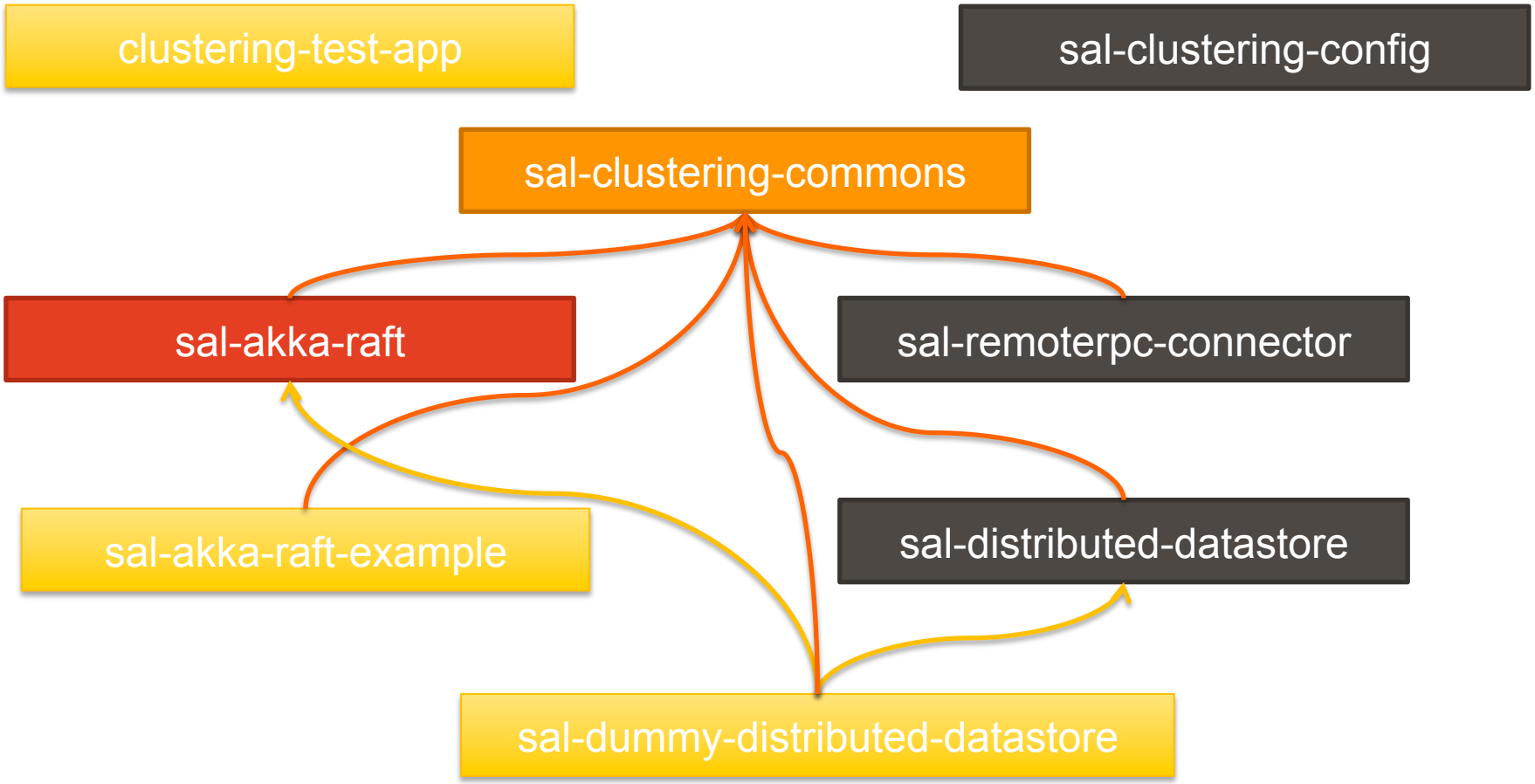
local versions higher – send update
local versions lower – send status to sender



Every 1 second members
send all known bucket
versions to any one peer

Modules

Modules


clustering-test-app
sal-clustering-config
sal-clustering-commons
sal-akka-raft
sal-remoterpc-connector
sal-akka-raft-example
sal-distributed-datastore
sal-dummy-distributed-datastore

sal-clustering-commons

- Some common messages
- Actor base classes
- The Protobuf messages used in Helium
- The Protobuf NormalizedNode serialization code
- The NormalizedNode streaming code
- Other miscellaneous utility classes

sal-akka-raft

- Implementation of the Raft Algorithm on top of akka
- Uses akka-persistence for durability
- Provides a base class called **RaftActor** which when can be extended by anyone who wants to replicate state
- See sal-akka-raft-example which provides a simple implementation of a replicated **HashMap**

sal-distributed-datastore

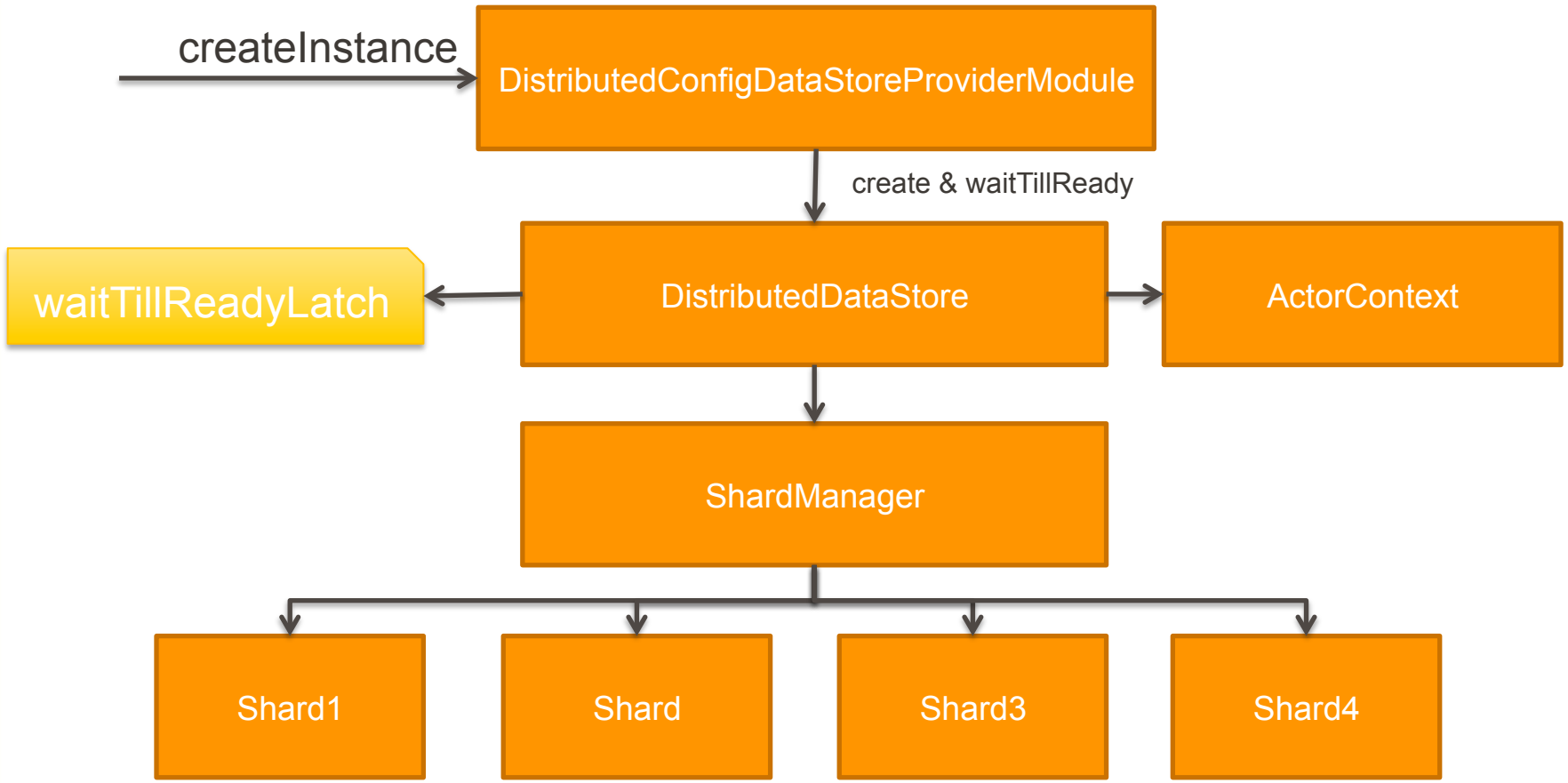
- **ConcurrentDOMDataBroker**
- **DistributedDataStore**
- Implementation of the **DOMStore** SPI
- Shard built on top of RaftActor
- Creates Shards based on Sharding strategy
- Code for a client to interact with the Shard Leader

sal-remoterpc-connector

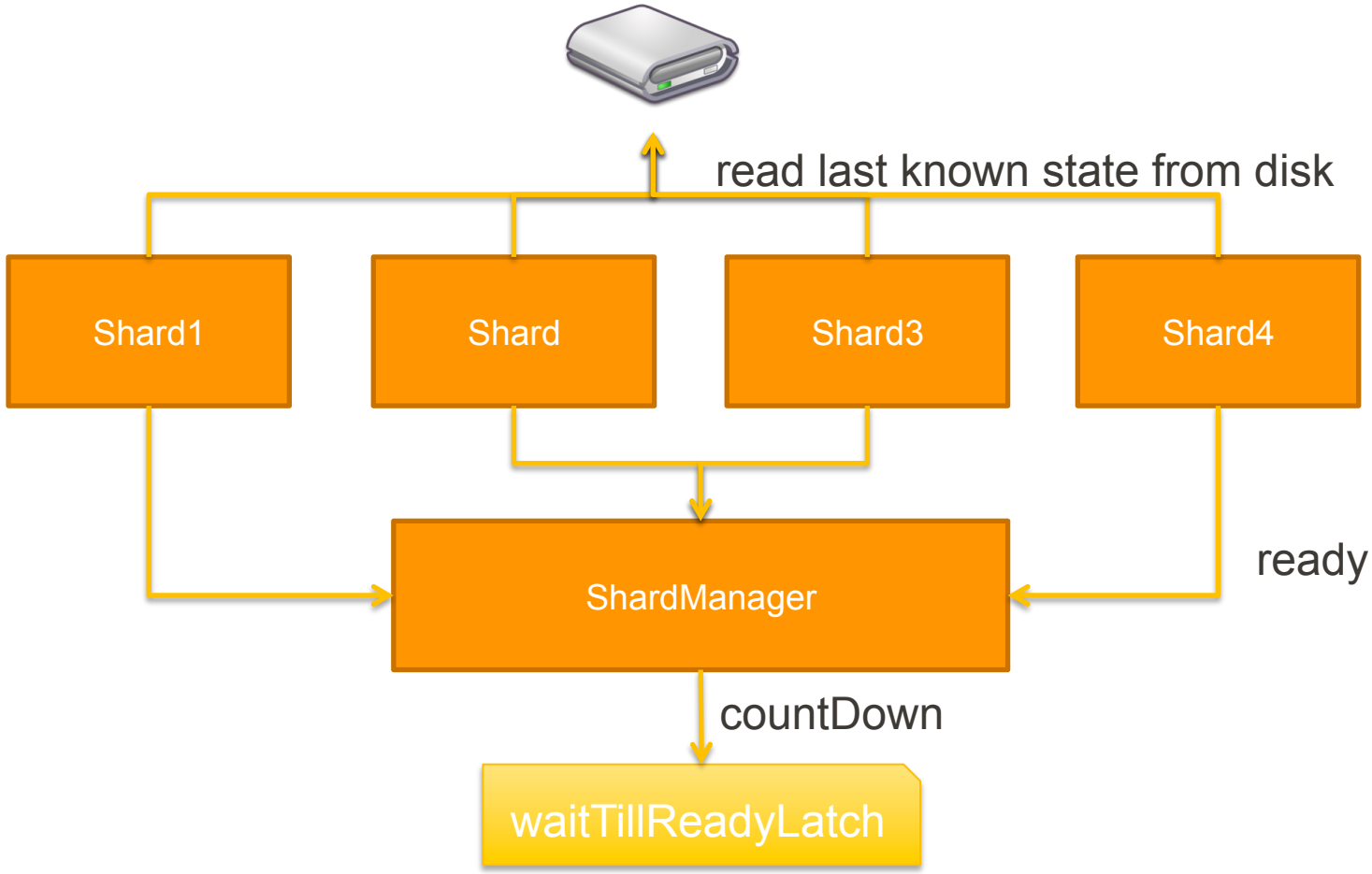
- **RemoteRpcProvider**
- Default RPC Provider. Invoked when an RPC is not found in the local MD-SAL registry.
- Code for BucketStore which provides a mechanism to replicate state based on Gossip
- Code for RpcBroker which allows invoking a remote rpc

Data store flows

Startup


createInstance
DistributedConfigDataStoreProviderModule
create & waitTillReady
waitTillReadyLatch
DistributedDataStore
ActorContext
ShardManager
Shard1
Shard
Shard3
Shard4

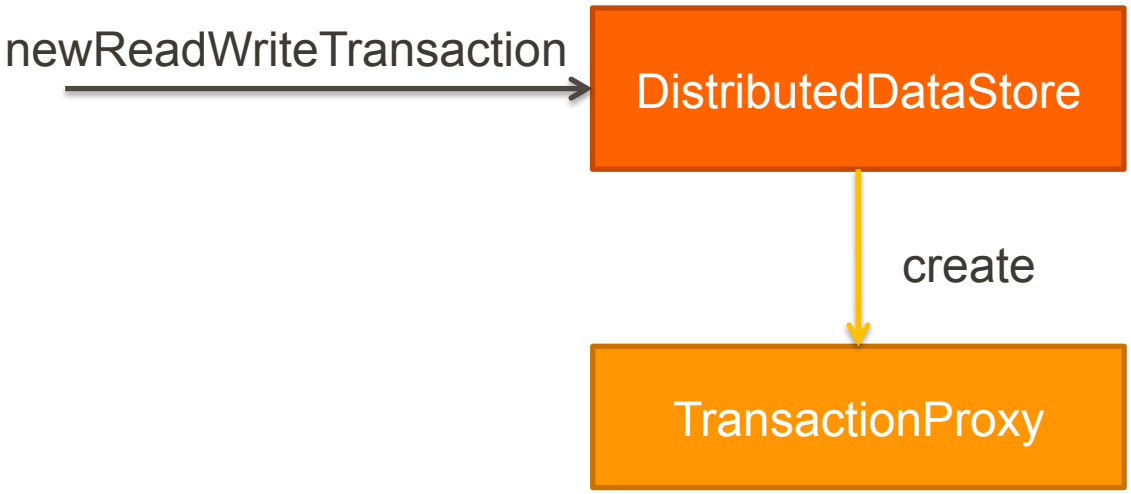
Recovery


read last known state from disk
Shard1
Shard
Shard3
Shard4
ShardManager
ready
countDown
waitTillReadyLatch

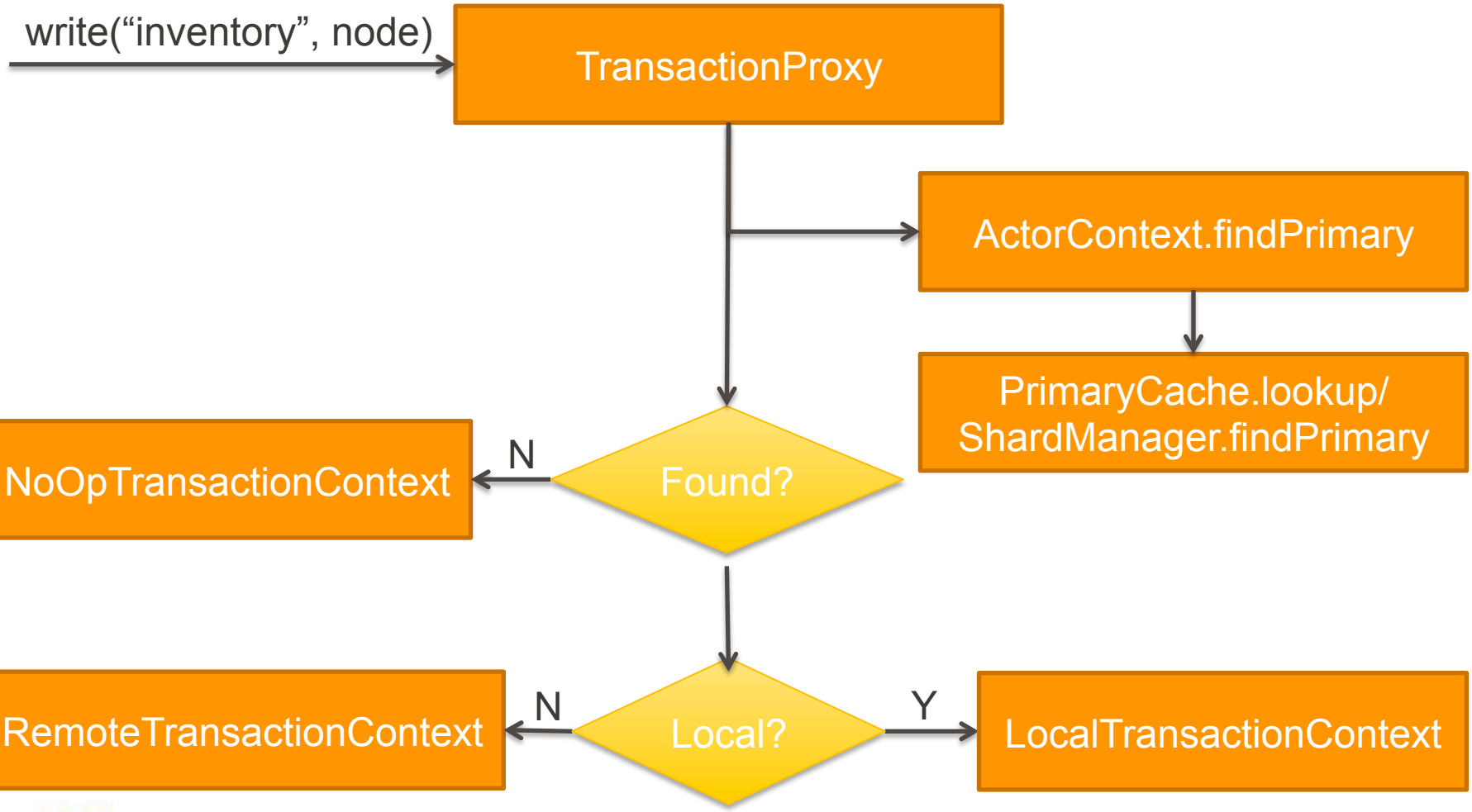
Waiting for Ready

- Recovery must be complete
- All Shard Leaders must be known
- Three messages are monitored by ShardManager
  - Cluster.MemberStatusUp
    - Used to figure out the address of a cluster member
  - LeaderStateChanged
    - Used to figure out if a Follower has a different Leader
  - ShardRoleChanged
    - Use to figured out any changes in a Shard's Role
- Waiting is not infinite, by default it lasts only 90 seconds but is configurable
- Will block config sub-system

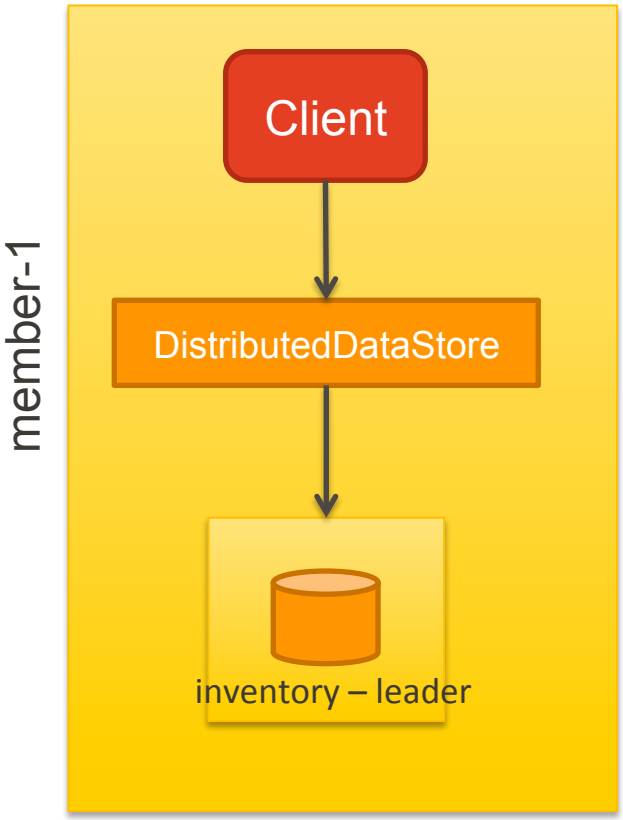
Creating a Transaction



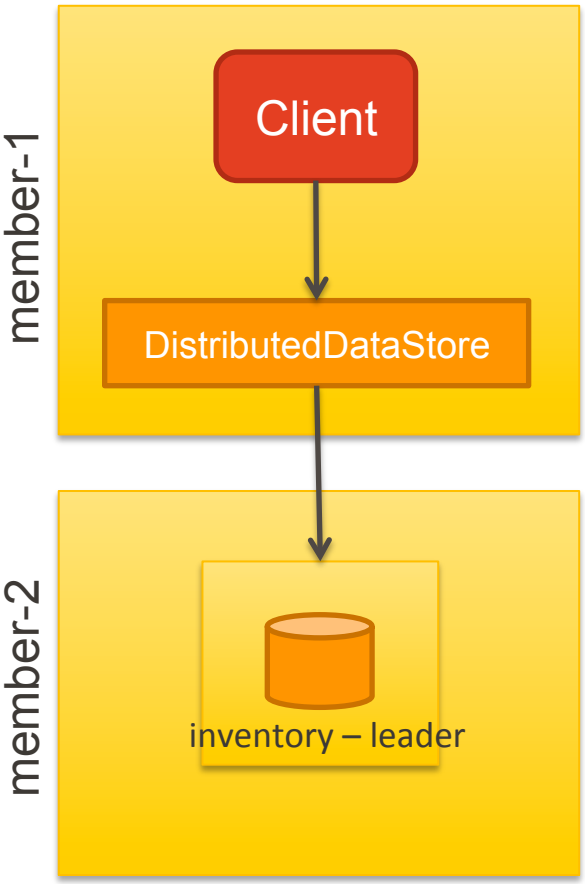
First Operation



Transactions

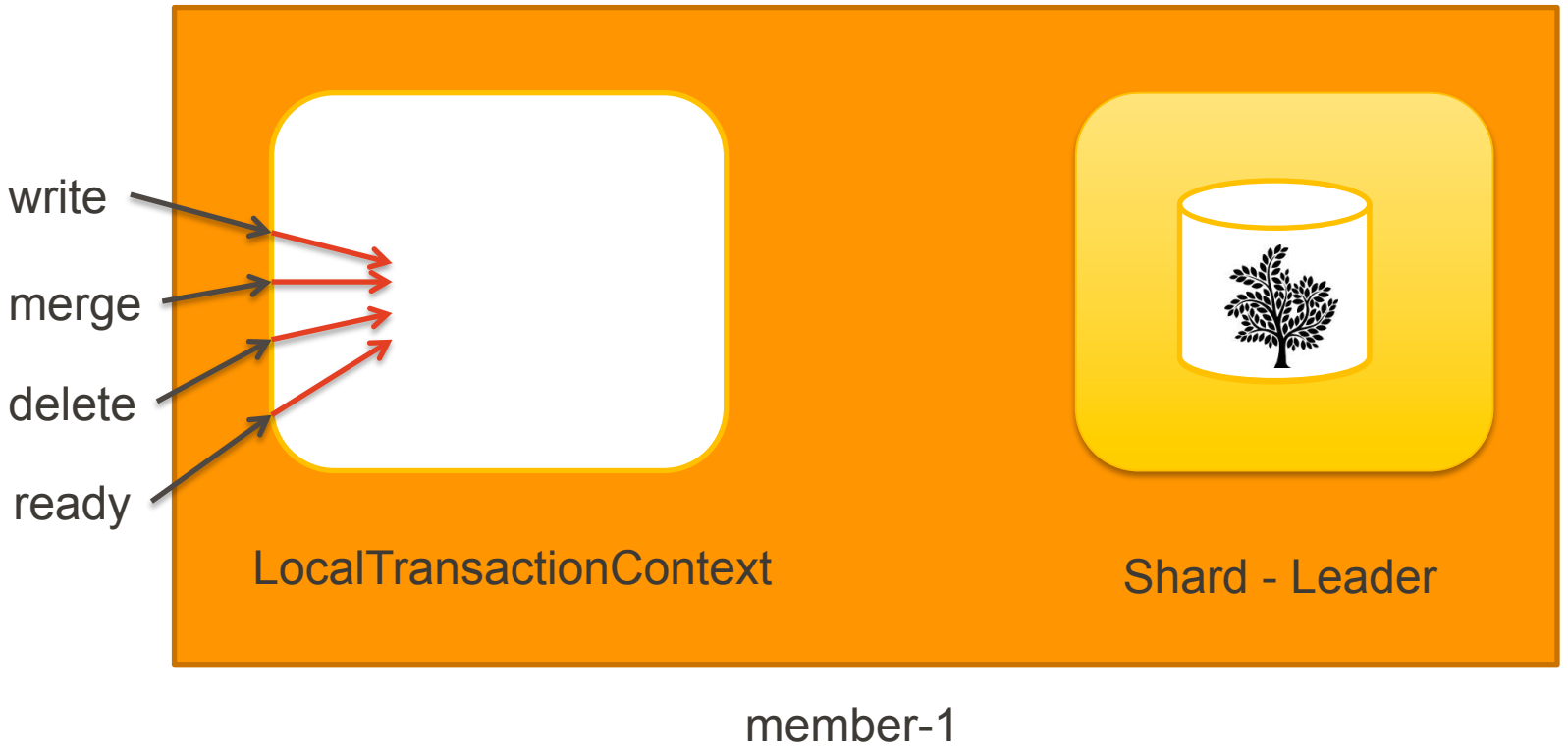


Local Transaction

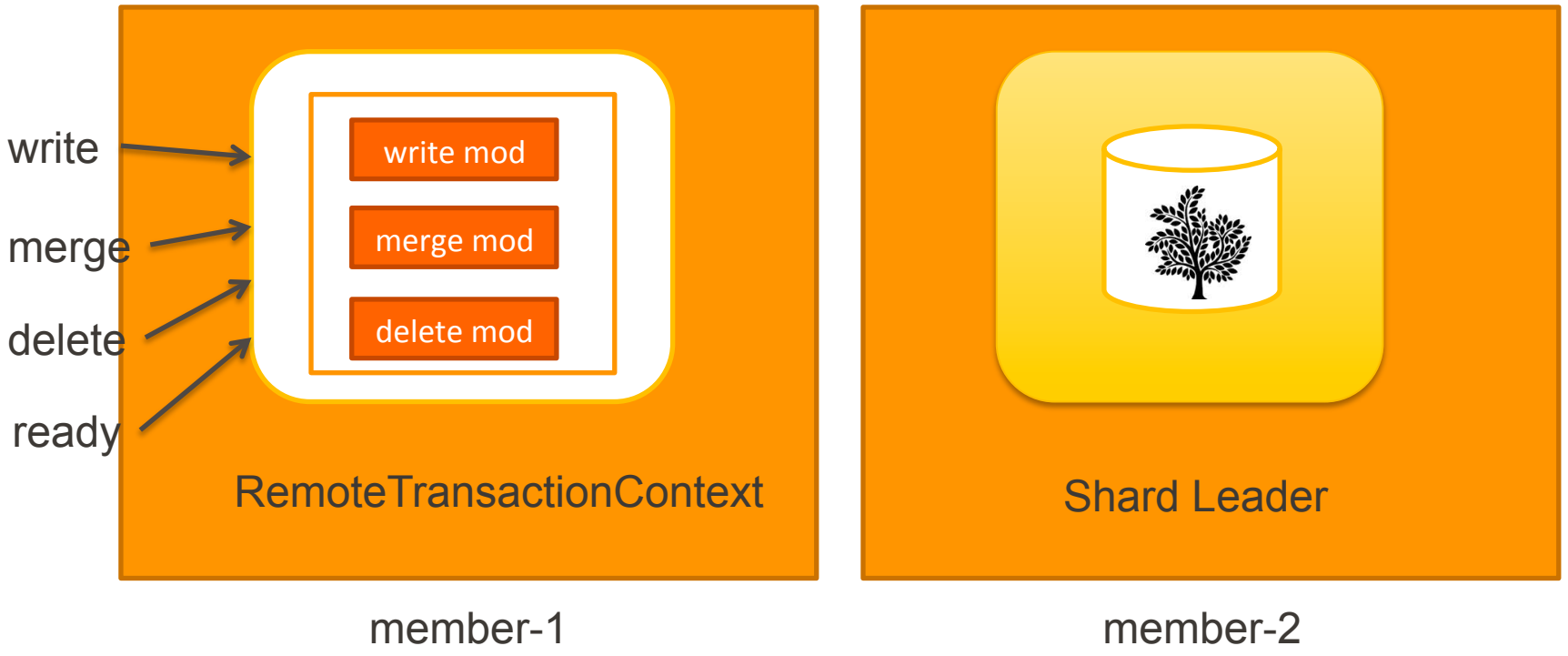


Remote Transaction

Local Transaction Optimization

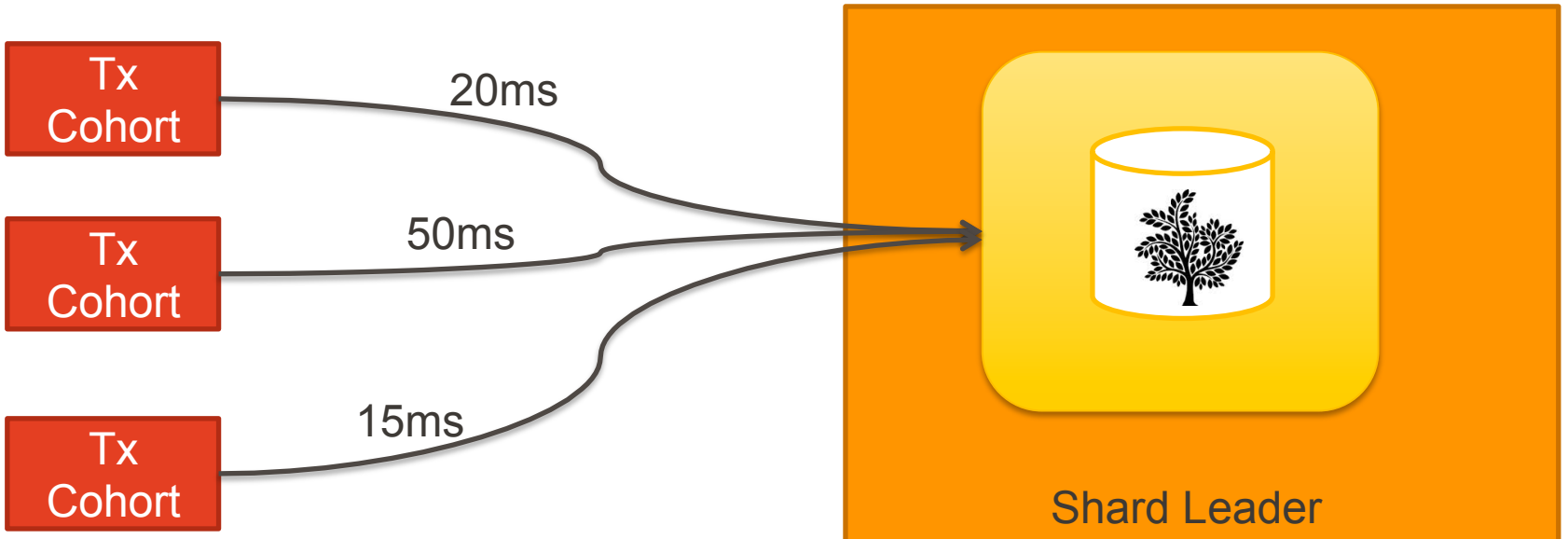


Remote Transaction Optimization



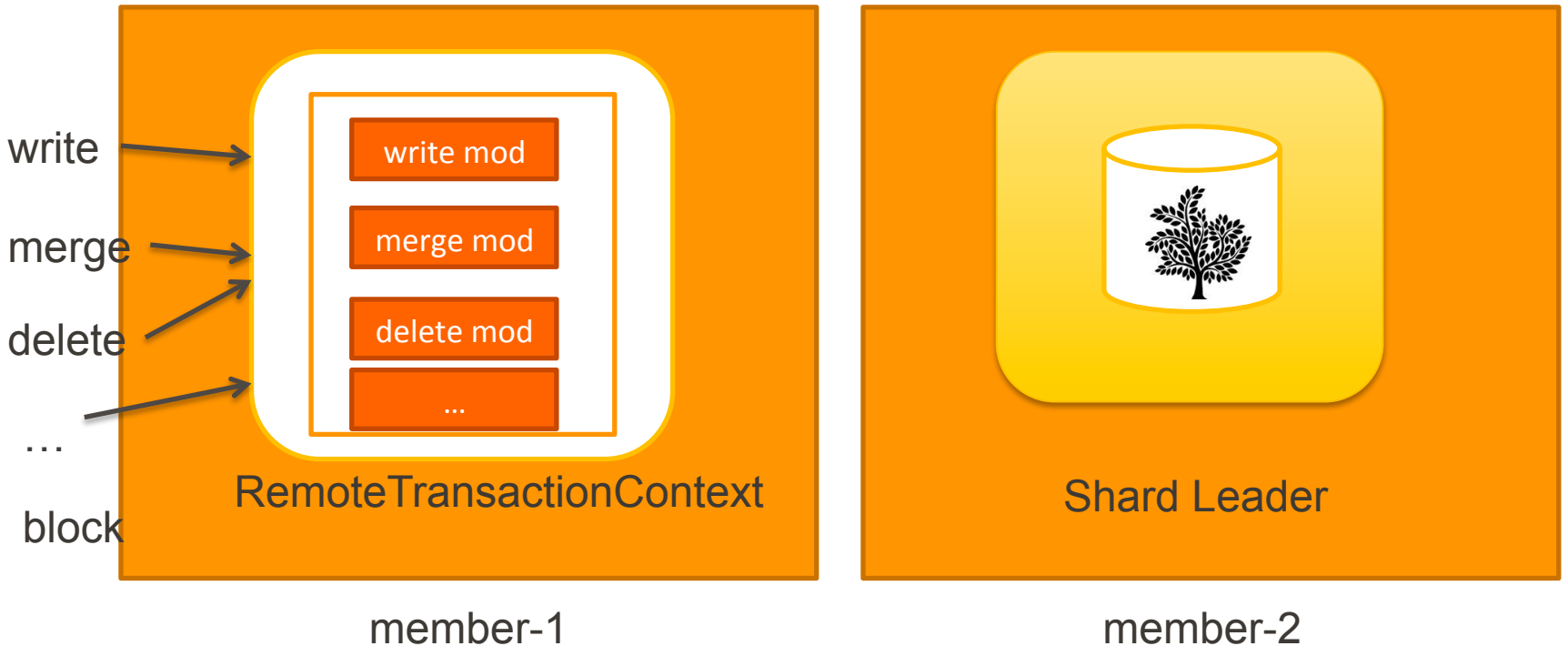
Transaction Rate Limiting

rate-limit = 100 Tx/Sec

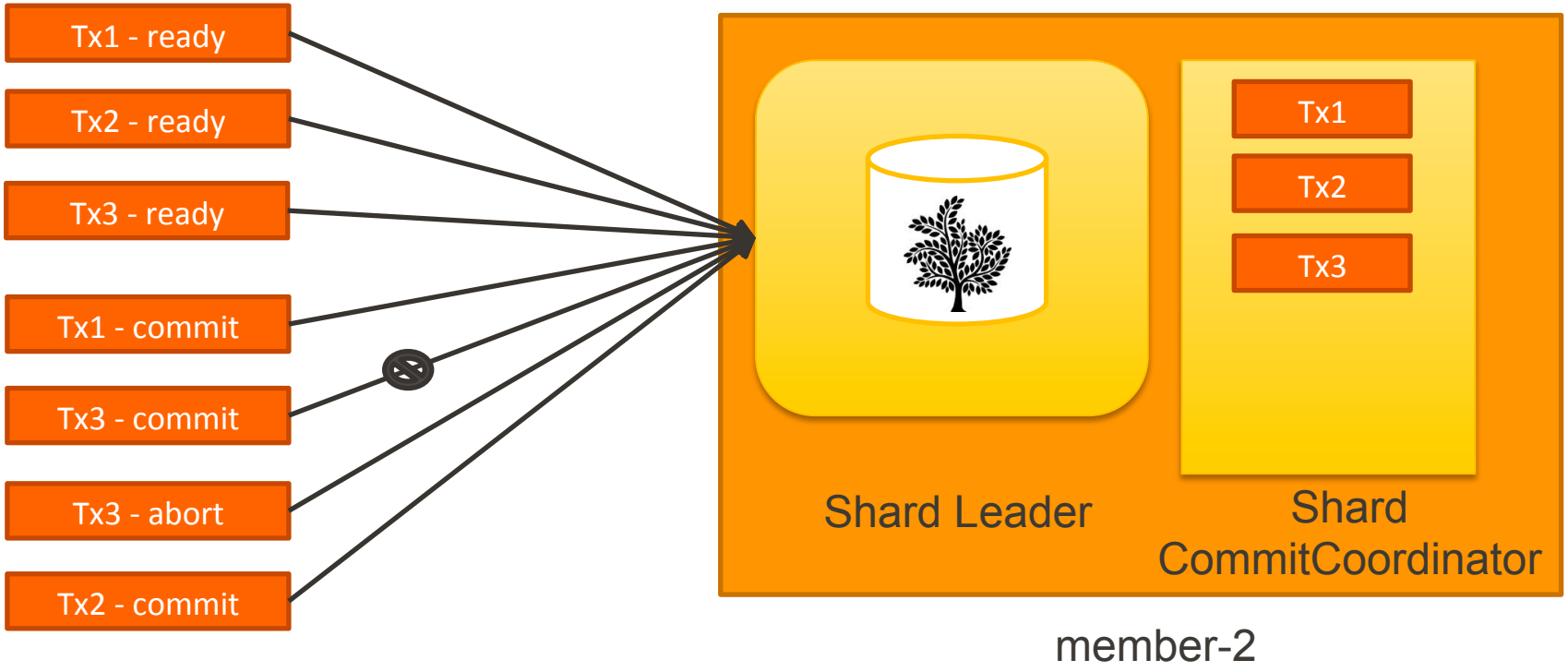


after rate-limit/2 transactions done….
new-rate-limit = 25 Tx/Sec

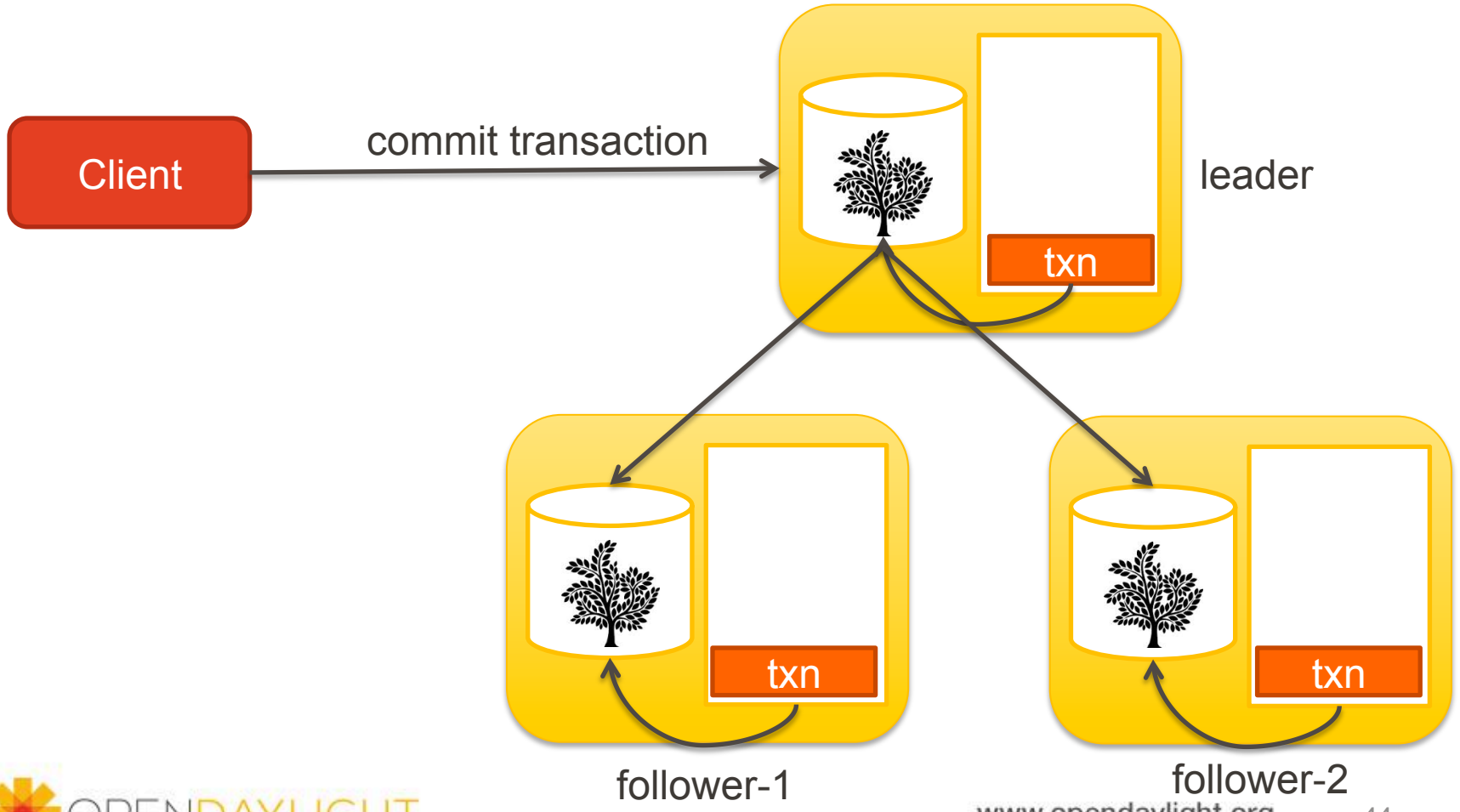
Operation Limiting



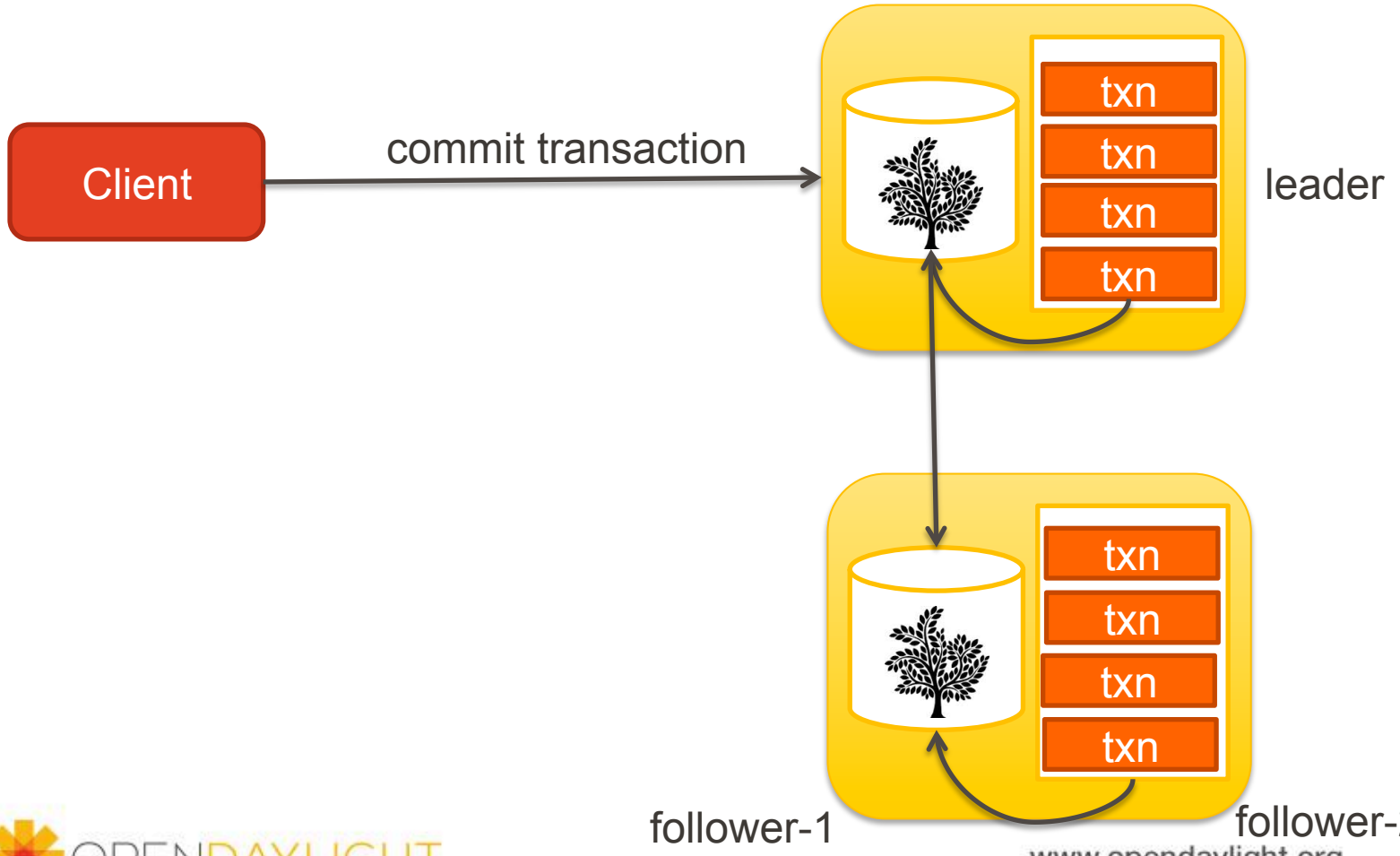
Commit Coordination



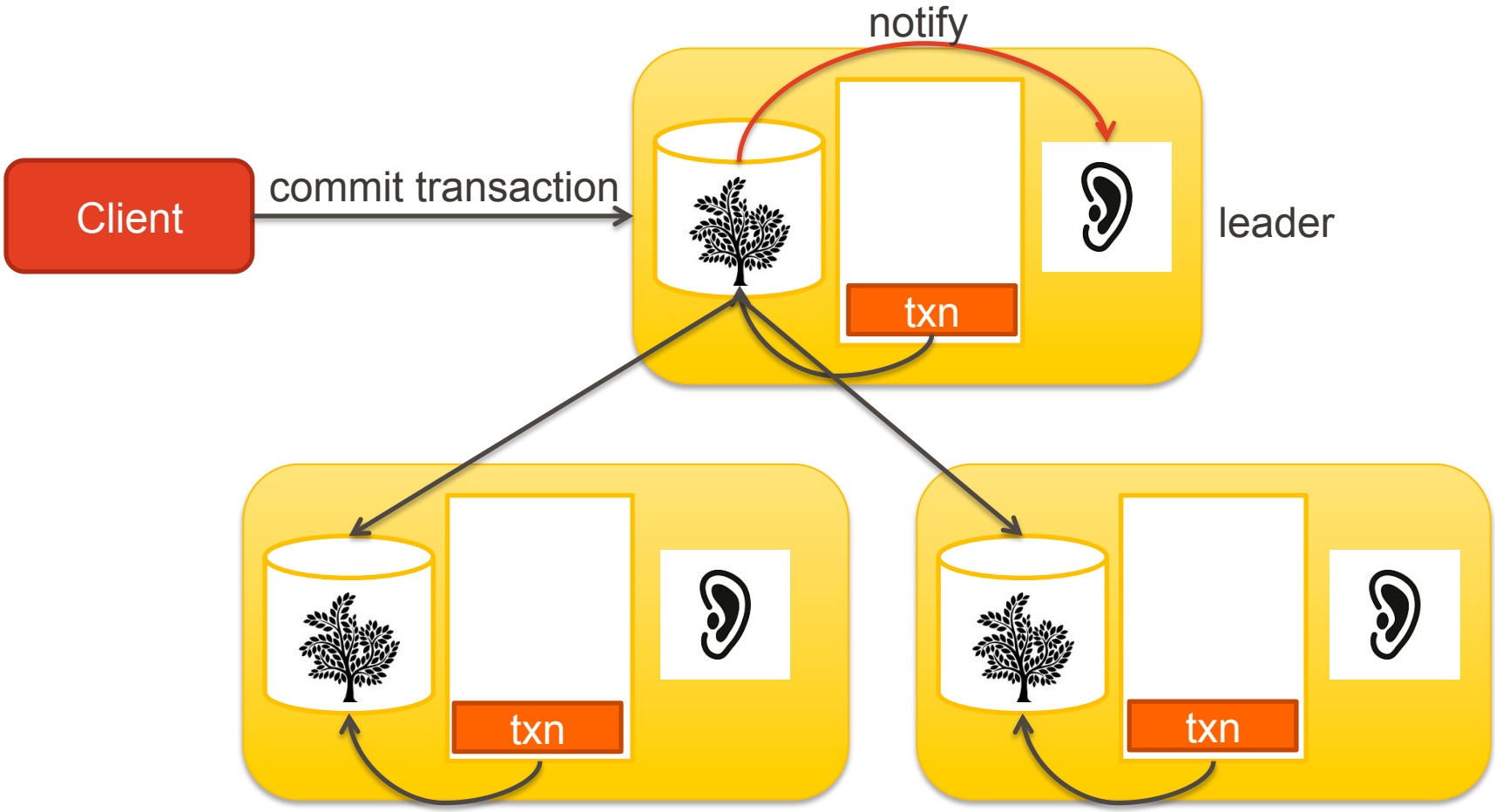
Managing the in-memory journal Replicated To All



Managing the in-memory journal
Cluster member unavailable

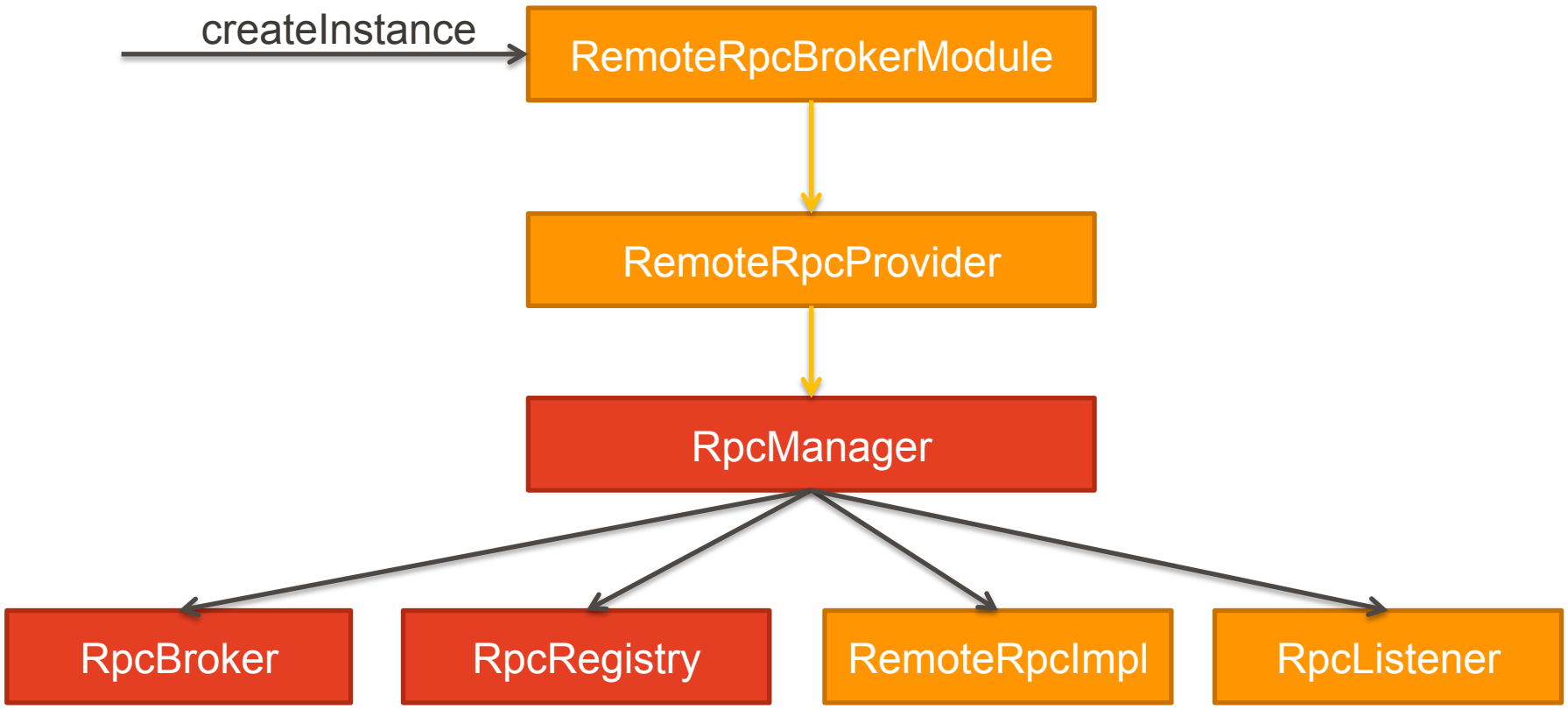


Data Change Notifications

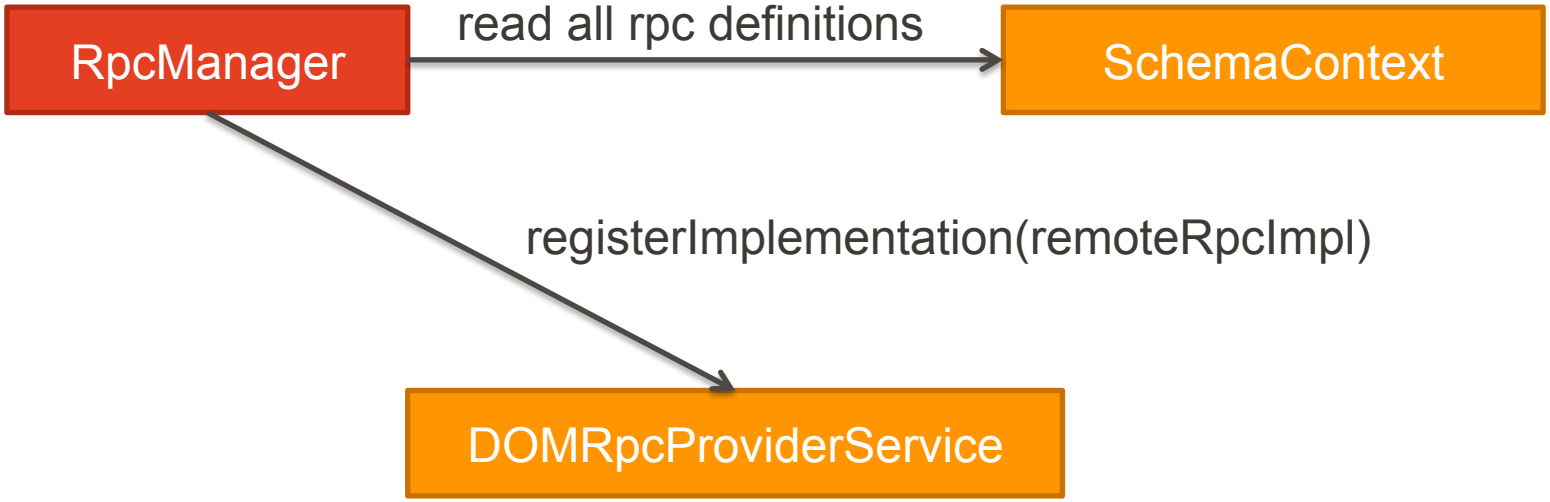


RPC Connector flows

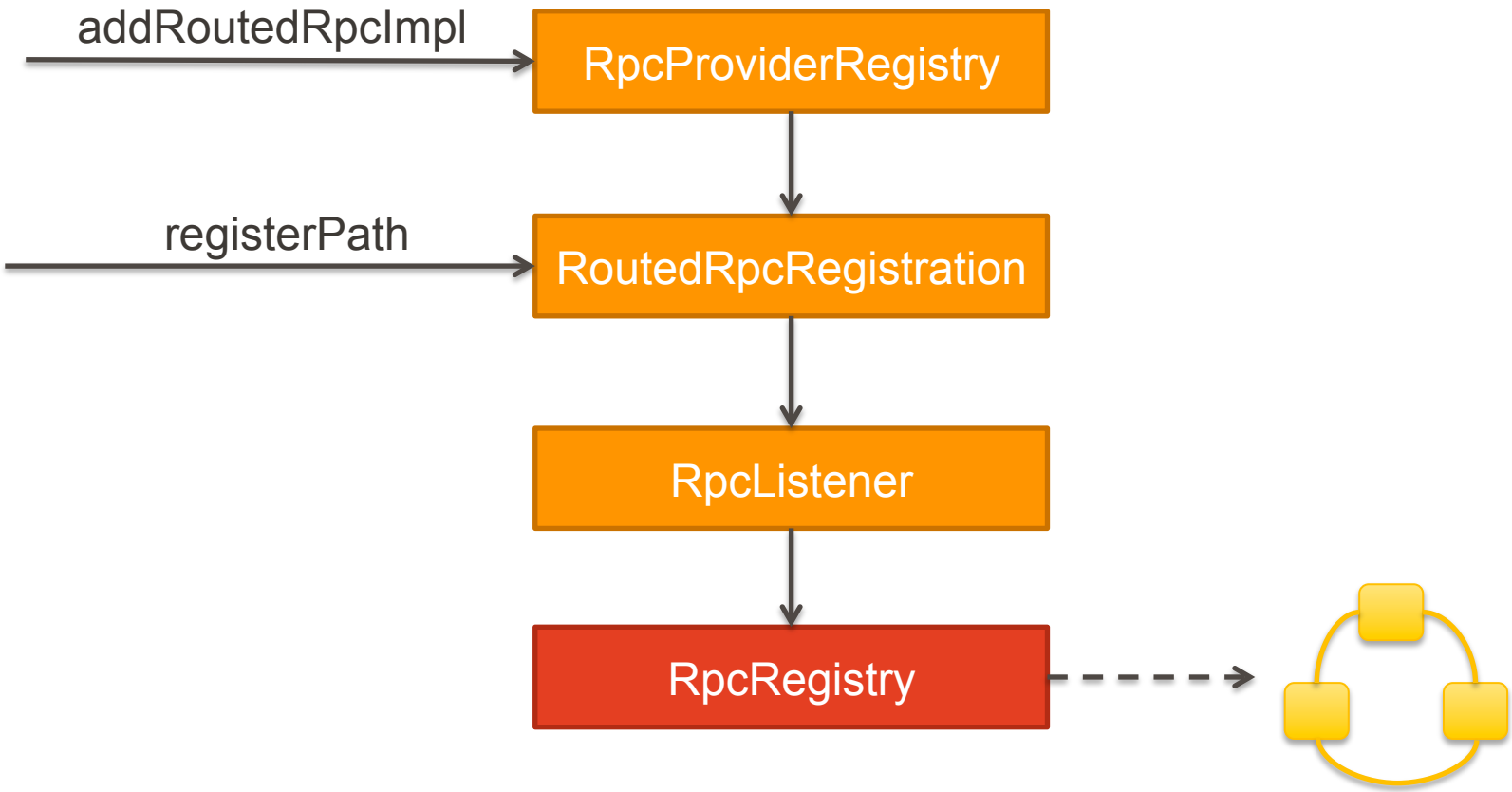
Startup


createInstance
RemoteRpcBrokerModule
RemoteRpcProvider
RpcManager
RpcBroker
RpcRegistry
RemoteRpcImpl
RpcListener

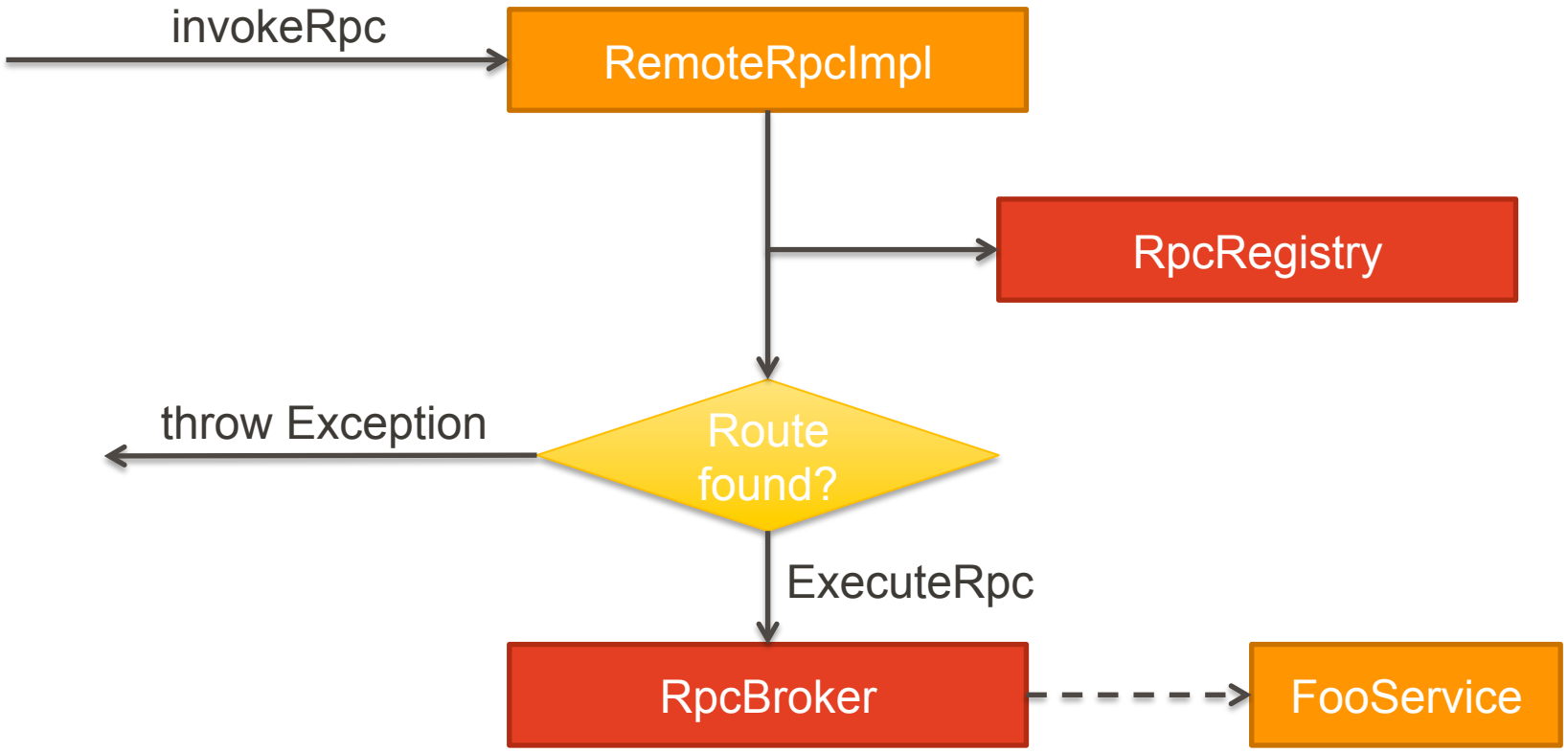
Default RPC Delegate



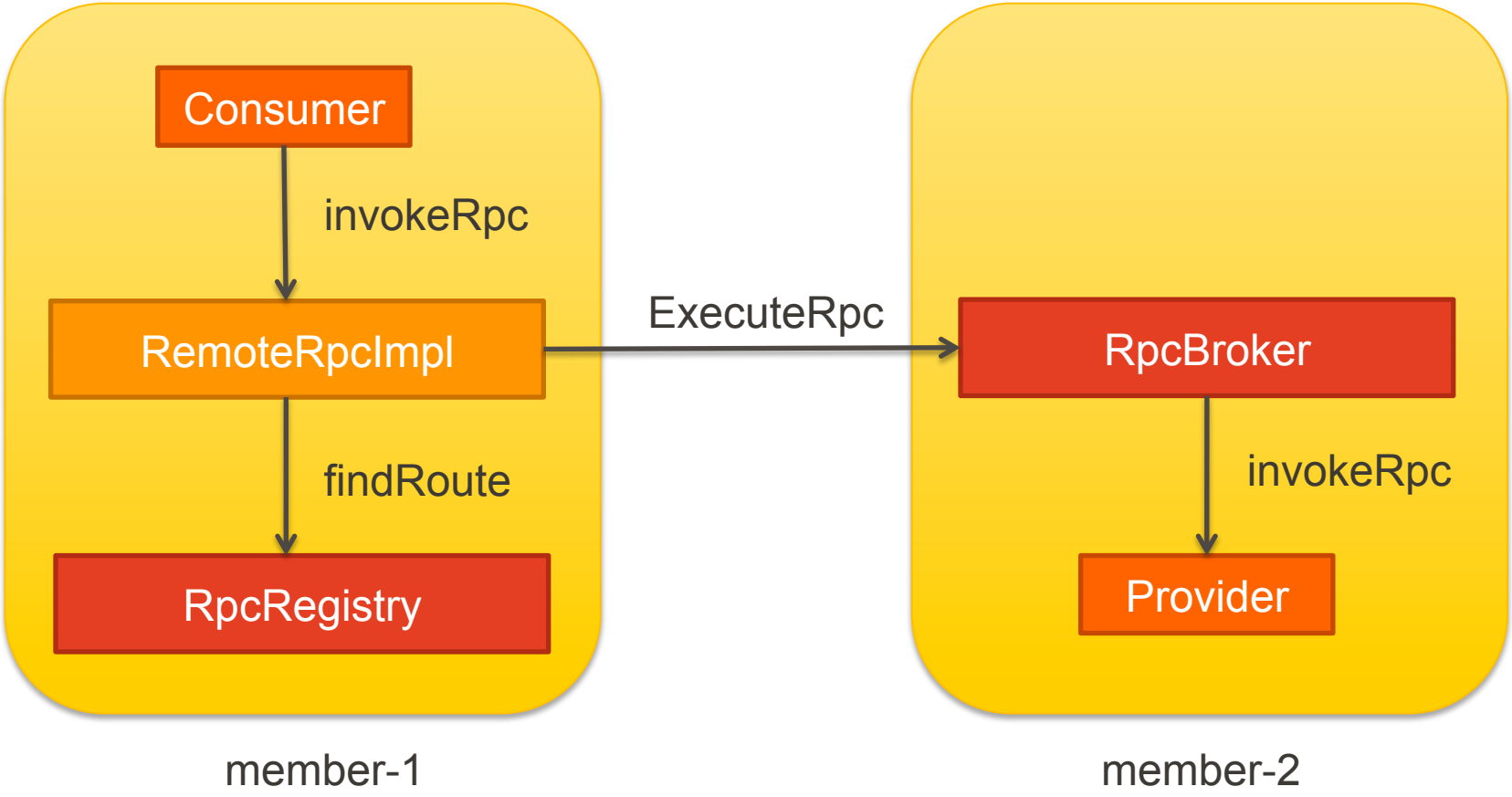
RPC Registered



Invoking a Remote RPC



Invoking a Remote RPC



Data store Diagnostics

Transaction Tracing

Client

Created txn member-2-txn-9400 of type READ_WRITE on chain member-2-txn-chain-13

Tx member-2-txn-9400 read /(urn:opendaylight:inventory?...

Tx member-2-txn-9400 Readying 1 transactions for commit

Tx member-2-txn-9400 commit

Tx member-2-txn-9400: commit succeeded

Server

member-3-shard-inventory-operational: Creating transaction : shard-member-2-txn-9400

member-3-shard-inventory-operational: Readying transaction member-2-txn-9400

member-3-shard-inventory-operational: Committing transaction member-2-txn-9400

Cluster Member Initiator

Counter

Transaction Type

Module

Data store type

Replication Tracing

Leader

Sending AppendEntries to follower member-2-shard-topology-operational: AppendEntries [term=2, leaderId=member-1-shard-topology-operational, prevLogIndex=520, prevLogTerm=2, entries=[Entry{index=521, term=2}], leaderCommit=520, replicatedToAllIndex=-1]

handleAppendEntriesReply - FollowerLogInformation for member-2-shard-topology-operational updated: matchIndex: 521, nextIndex: 522

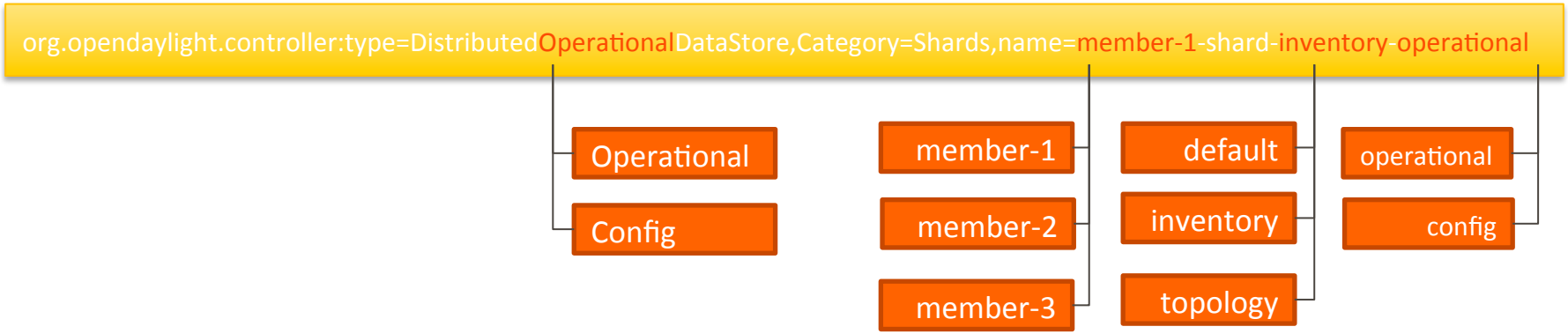
handleAppendEntriesReply from member-2-shard-topology-operational: applying to log – commitIndex: 521, lastAppliedIndex: 520

Follower

handleAppendEntries: AppendEntries [term=2, leaderId=member-2-shard-topology-operational, prevLogIndex=520, prevLogTerm=2, entries=[Entry{index=521, term=2}], leaderCommit=520, replicatedToAllIndex=-1]

handleAppendEntries returning : AppendEntriesReply [term=2, success=true, logLastIndex=521, logLastTerm=2, followerId=member-1-shard-topology-operational]

Shard MBean



Attributes

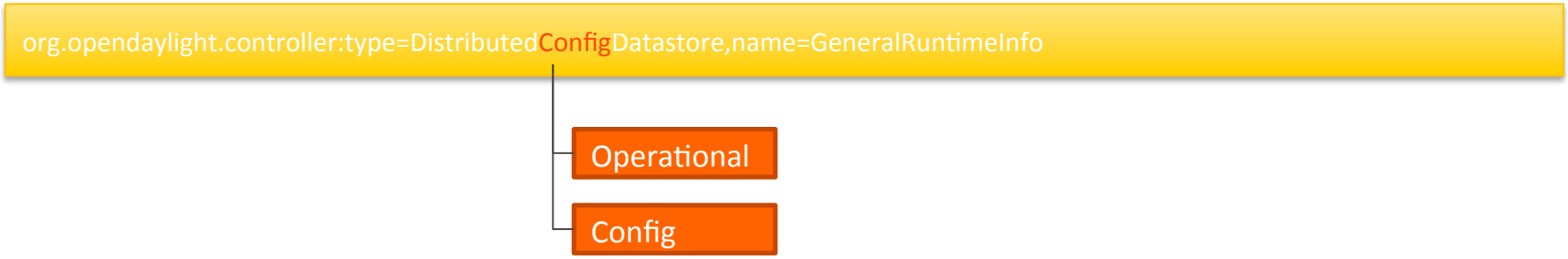
AbortTransactionsCount	CommitIndex	CommittedTransactionsCount	CurrentTerm	FailedTransactionsCount
FollowerInfo	FollowerInitialSyncStatus	InMemoryJournalDataSize	InMemoryJournalLogSize	LastApplied
LastCommittedTransactionTime	LastIndex	LastTerm	Leader	RaftState
ReadOnlyTransactionCount	ReadWriteTransactionCount	WriteOnlyTransactionCount	VotedFor	and more….

ShardManager MBean

Attributes

- LocalShards
- SyncStatus

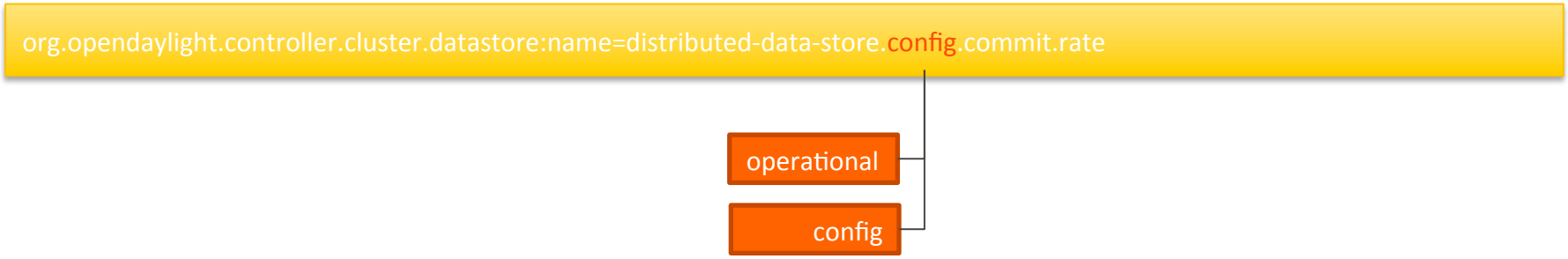
Data store GeneralRuntimeInfo MBean



Attributes

- TransactionCreationRateLimit

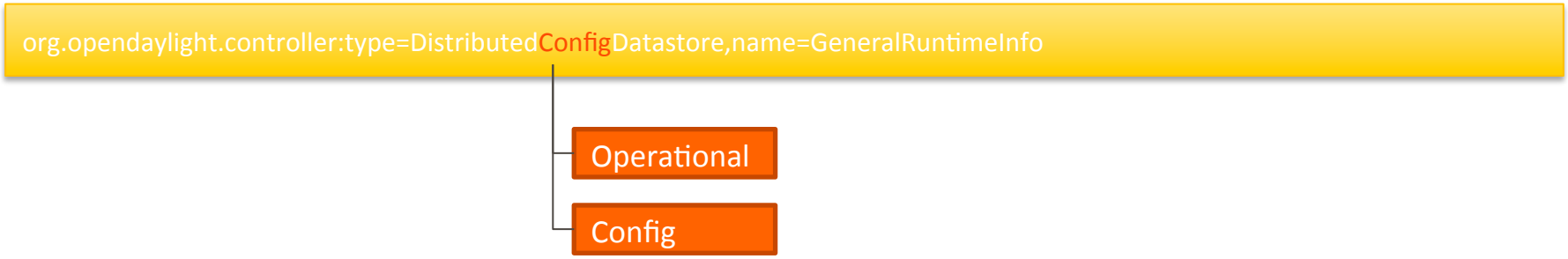
Transaction Commit Rate MBean



Attributes

- Count
- Min
- Max
- StdDev
- 50thPercentile
- 75thPercentile
- 90thPercentile
- and so on…

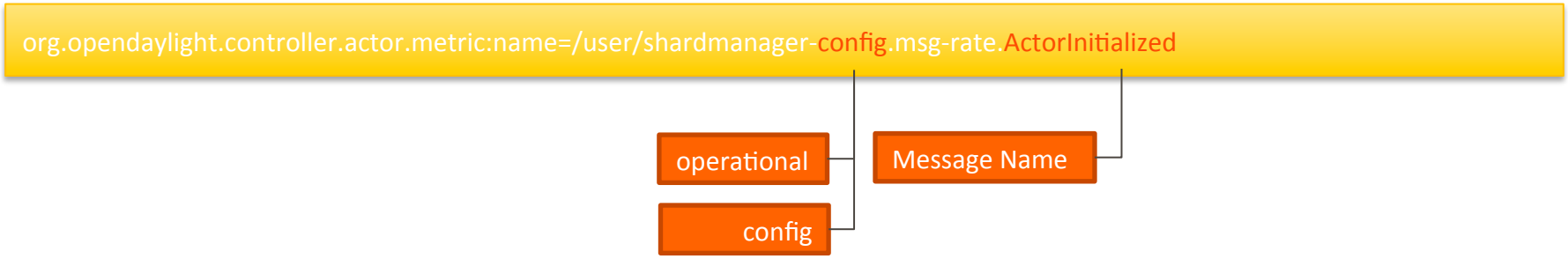
Data store GeneralRuntimeInfo MBean



Attributes

- TransactionCreationRateLimit

Message Statistics MBean



Attributes

- Count
- Min
- Max
- StdDev
- 50thPercentile
- 75thPercentile
- 90thPercentile
- and so on…

Remote RPC Broker Diagnostics

RemoteRpcBroker MBean

org.opendaylight.controller:type=RemoteRpcBroker,name=RemoteRpcRegistry

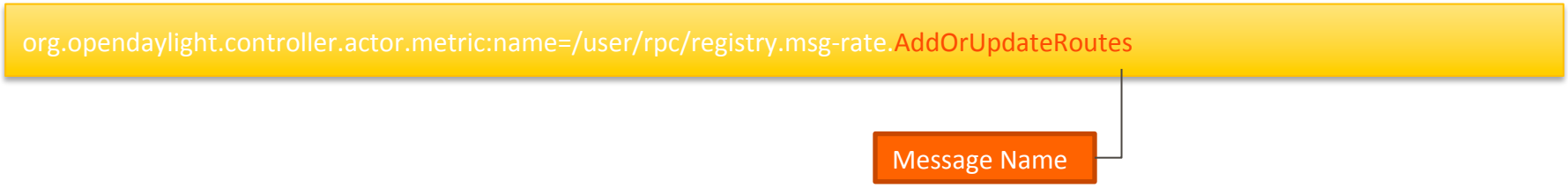
Attributes

- BucketVersions
- GlobalRpc
- LocalRegisteredRoutedRpc

Operations

- findRpcByName
- findRpcByRoute

Message Statistics MBean



Attributes

- Count
- Min
- Max
- StdDev
- 50thPercentile
- 75thPercentile
- 90thPercentile
- and so on…

Suggested Next Steps…

- Deploy a cluster
- Run clustering integration tests
- Write an application that works in the cluster
- Write bugs to report features which you find missing
- Try running dsBenchMark on a cluster
- Test out replication using the dummy data store
- Check out the code
- Send email to controller-dev@lists.opendaylight.org with questions