

[<- Back to NucleoCore](#)

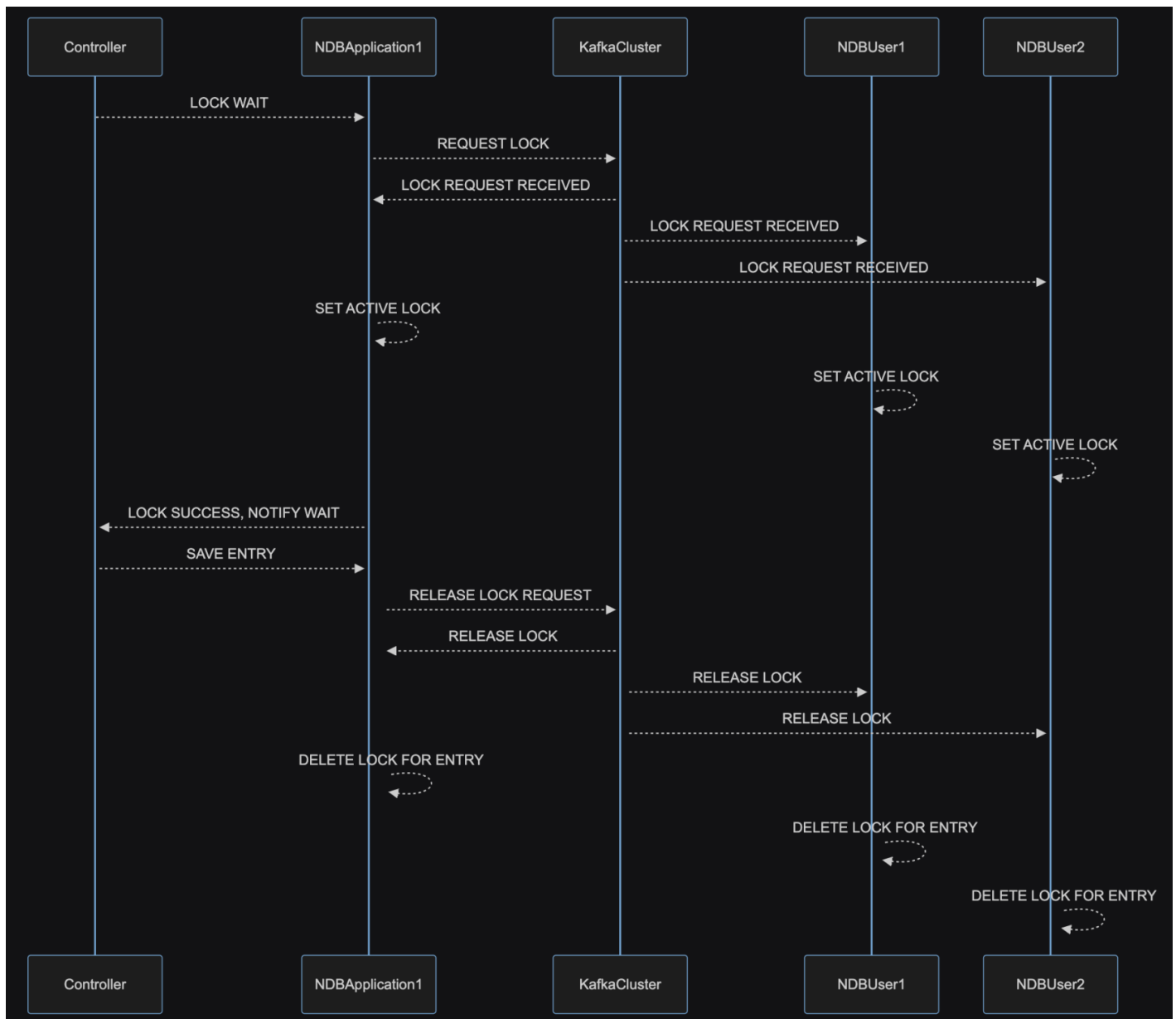
# Locks

- [Cluster Locks](#)

# Cluster Locks

## Cluster wide DataEntry locks.

You can lock a DataEntry by specifying in the copy step that you want to lock the entry. This will send a request to lock the entry to all of the running instances of NucleoDB until that entry has been saved ( Or a timeout of 1 second).



```
BookDE book = (BookDE)nucleoDB.getTable(BookDE.class).get("name", "test").iterator().next();
```

```
// true indicates a cluster wide lock
```

```

// the thread will wait until the lock is successful for this specific request
BookDE bookWritable = book.copy(BookDE.class, true);

// entry is now locked for the entire cluster
// any changes will be thread safe
bookWritable.getData().setName("Odyssey");

// save to cluster.
// NDB will release the lock and other locks will have a chance to successfully lock
nucleoDB.getTable(BookDE.class).saveSync(bookWritable);

```

sequenceDiagram

```

Controller-->> NDBApplication1: LOCK WAIT
NDBApplication1-->>KafkaCluster: REQUEST LOCK
KafkaCluster-->>+NDBApplication1: LOCK REQUEST RECEIVED
KafkaCluster-->>+ NDBUser1: LOCK REQUEST RECEIVED
KafkaCluster-->>+ NDBUser2: LOCK REQUEST RECEIVED
NDBApplication1 -->>+ NDBApplication1: SET ACTIVE LOCK
NDBUser1 -->>+ NDBUser1: SET ACTIVE LOCK
NDBUser2 -->>+ NDBUser2: SET ACTIVE LOCK
NDBApplication1-->>+Controller: LOCK SUCCESS, NOTIFY WAIT
Controller-->>+NDBApplication1: SAVE ENTRY
NDBApplication1-->>+KafkaCluster: RELEASE LOCK REQUEST
KafkaCluster -->>+NDBApplication1: RELEASE LOCK
KafkaCluster -->>+NDBUser1: RELEASE LOCK
KafkaCluster -->>+NDBUser2: RELEASE LOCK
NDBApplication1 -->>+NDBApplication1:DELETE LOCK FOR ENTRY
NDBUser1 -->>+NDBUser1: DELETE LOCK FOR ENTRY
NDBUser2 -->>+NDBUser2: DELETE LOCK FOR ENTRY

```

