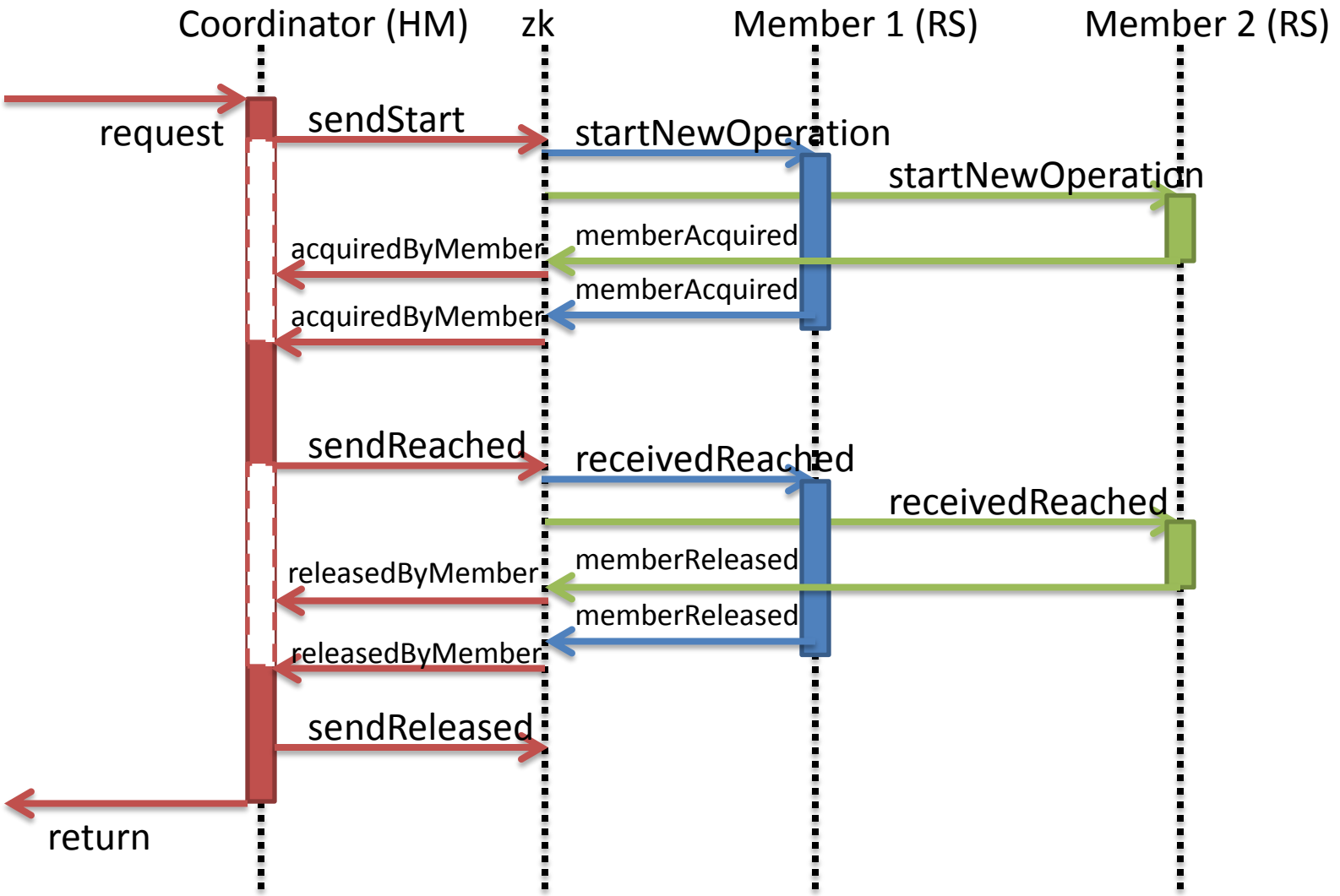
DISTRIBUTED BARRIER PROCEDURE

Barrier Procedure

- Problem:
  - Globally consistent snapshot requires the ability to quiesce a set of regionservers before allowing progress.
  - A failure in one regionserver should result in a cancelation on all others
  - Need to be able to force failure after a specified timeout elapses.
- Solution:
  - Framework that only requires user to implement:
    - The Acquisition Operation
    - The Barriered Operation (and release)

Barrier Procedure coordination

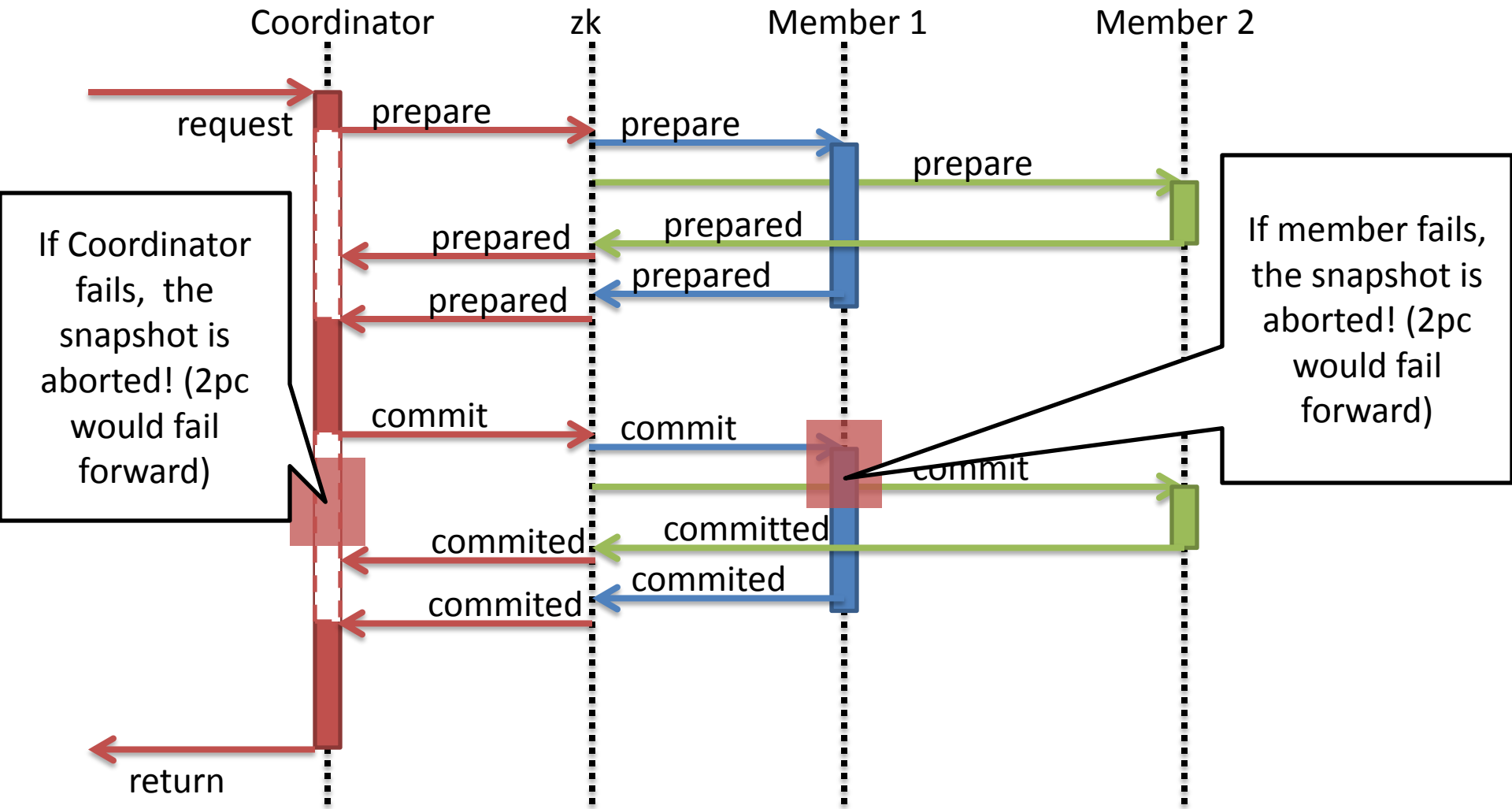


Procedure vs 2 Phase Commit

- 2PC is a distributed transaction protocol that supports ACID semantics.
- After Commit is decided at the Coordinator the commit **must recover** and fail forward.

- Procedure has similar communications phases but does not support ACID semantics
- **Does not recover on failures**
- For now this is ok – if we fail anywhere the snapshot fails, and another can be taken without adversely affecting original table.

Procedure coordination



If we aren't doing proper 2PC do we need all this infrastructure?

Procedure / Subprocedure

ProcedureCoordinator

- ***ProcedureCoordinatorComms**
- **ZKProcedureCoordinatorComms**
  - ZKProcedureUtil

Procedure

ProcedureFactory

ProcedureMember

- ***ProcedureMemberComms**
- **ZKProcedureMemberComms**
  - ZKProcedureUtil

Subprocedure

SubprocedureFactory

ZK interactions: Acquire Barrier

- Coordinator starts by wiping out, then creating and watching these znodes
  - .../class/acquired
  - .../class/reached
  - .../class/abort
- Coordinator drops a new id (snapshot id) in acquired
  - …/class/acquired/snapshot121127
- Members see this and do their local acquire and complete by inserting an acquired node
  - …/class/acquired/snapshot121127/server1
  - …/class/acquired/snapshot121127/server2

ZK interactions: Reached barrier

- If **all** members successfully drop nodes in, the coordinator notifies that the global barrier has been reached by dropping a new znode.
  - …/class/reached/snapshot121127
- Members see this and then start their reached in-barrier operation. When complete they insert znodes:
  - …/class/reached/snapshot121127/server1
  - …/class/reached/snapshot121127/server2
- When Coordinator sees all are completed, it delete all these znodes.

ZK interactions: Aborting

- If anybody encounters an error is unable to complete due to timeout, it will drop a node in the aborted znode dir. (Note that the source member is not part of the name!)
  - …/class/abort/snapshot121127
- It contains a protobuf serialized ExternalException. This contains the source name. It is deserialized by all others and everyone gets receiveError calls with the exception. Everyone bails out.
- Eventually the coordinator will delete all znodes related to this Procedure