Easy Commit: A Non-blocking Two-phase Commit Protocol

Suyash Gupta, Mohammad Sadoghi
Dept. of Computer Science
University of California Davis

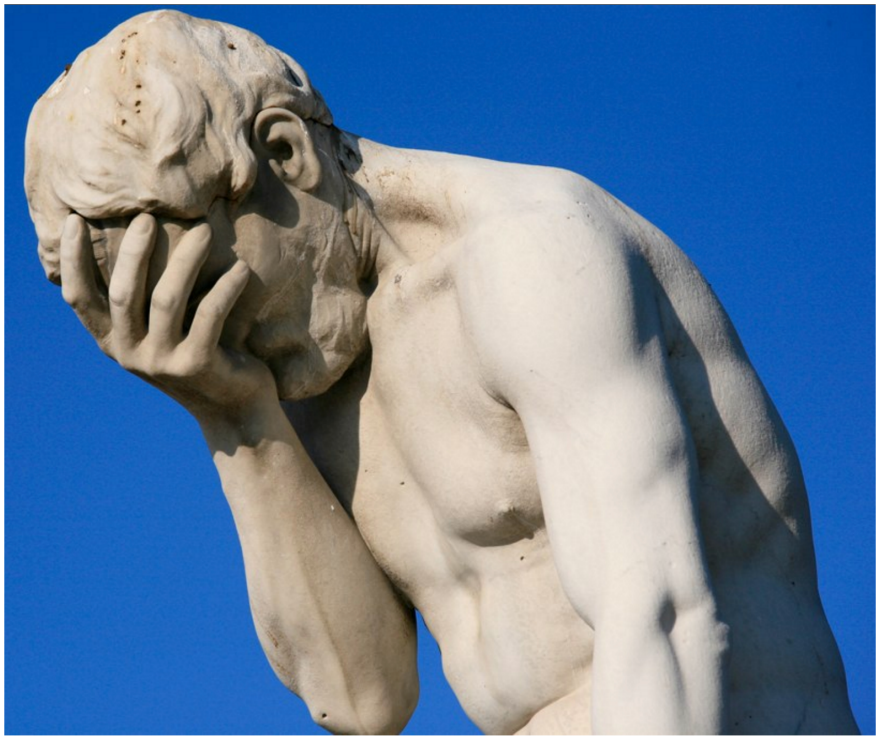
March 28, 2018

UCDAVIS UNIVERSITY OF CALIFORNIA

We are changing the world!!!

copyright emojiterra.com

Just Kidding!



copyright – Alex E. Proimos (taken from Wikipedia)

Motivation

- Concerns of a Distributed Database Designer:
  - Availability
  - Consistency
- Lots of hardware, resources $\Rightarrow$ Availability Solved.
- If Not Consistent $\Rightarrow$ Correctness Compromised!

Motivation

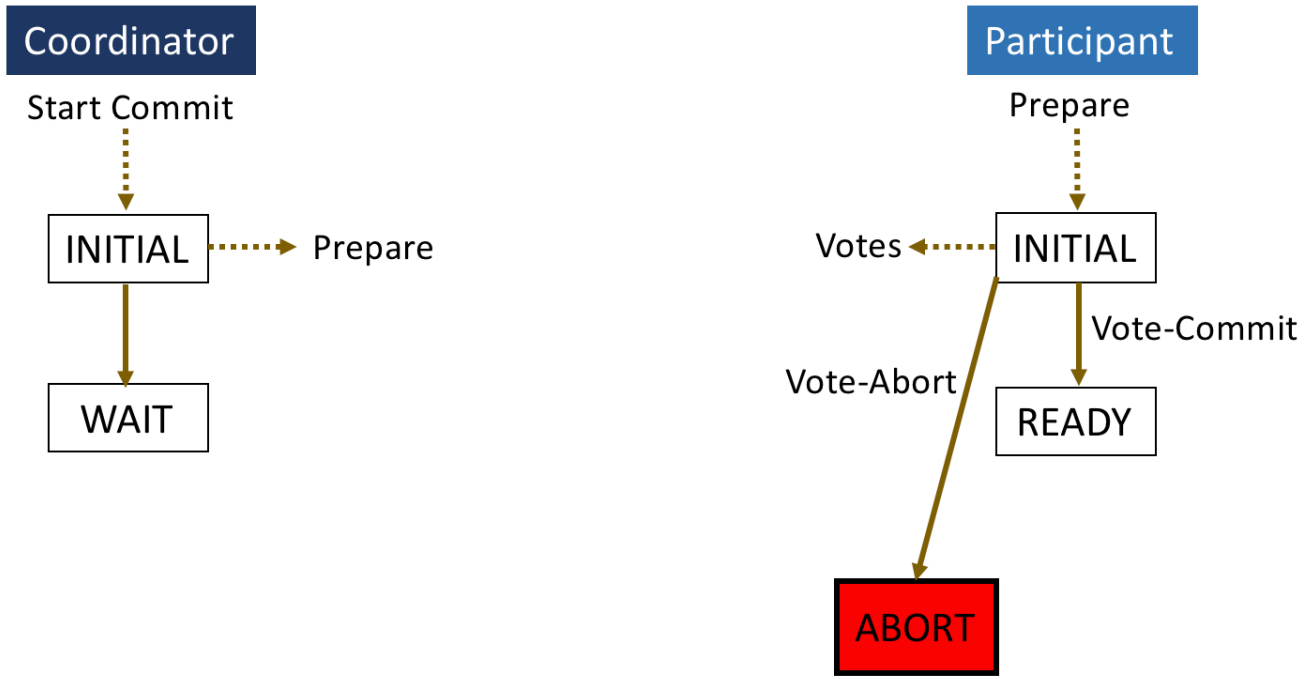
- Concerns of a Distributed Database Designer:
  - Availability
  - Consistency
- Lots of hardware, resources ⇒ Availability Solved.
- If Not Consistent ⇒ Correctness Compromised!
- Need for Agreement!
  - Two Phase Commit Protocol (2PC)? ⇒ Blocked!
  - Three Phase Commit Protocol (3PC)? ⇒ Heavy!

Motivation

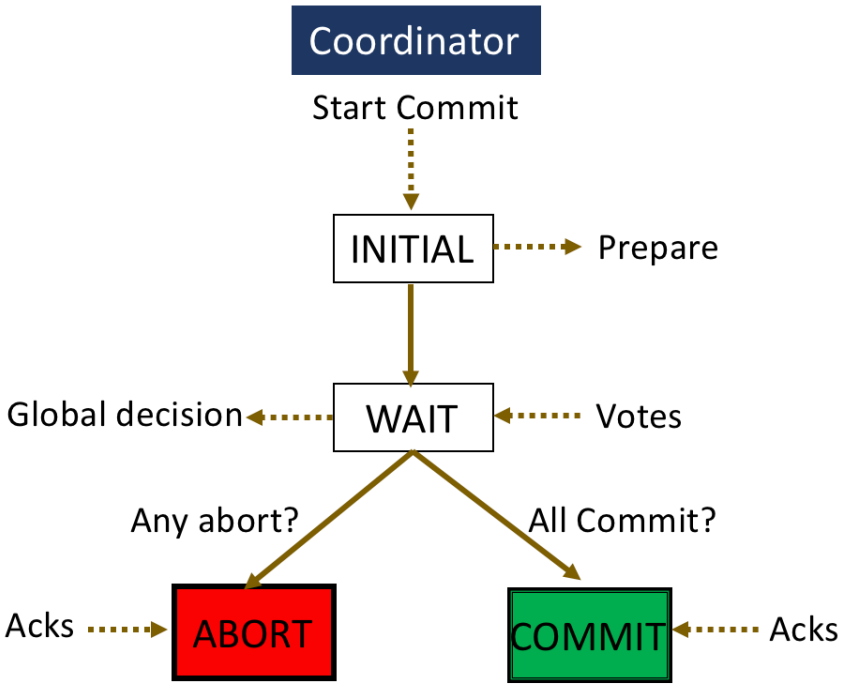
- Concerns of a Distributed Database Designer:
  - Availability
  - Consistency
- Lots of hardware, resources ⇒ Availability Solved.
- If Not Consistent ⇒ Correctness Compromised!
- Need for Agreement!
  - Two Phase Commit Protocol (2PC)? ⇒ Blocked!
  - Three Phase Commit Protocol (3PC)? ⇒ Heavy!
- **Easy Commit** ⇒ Two Phases! Non-Blocking!

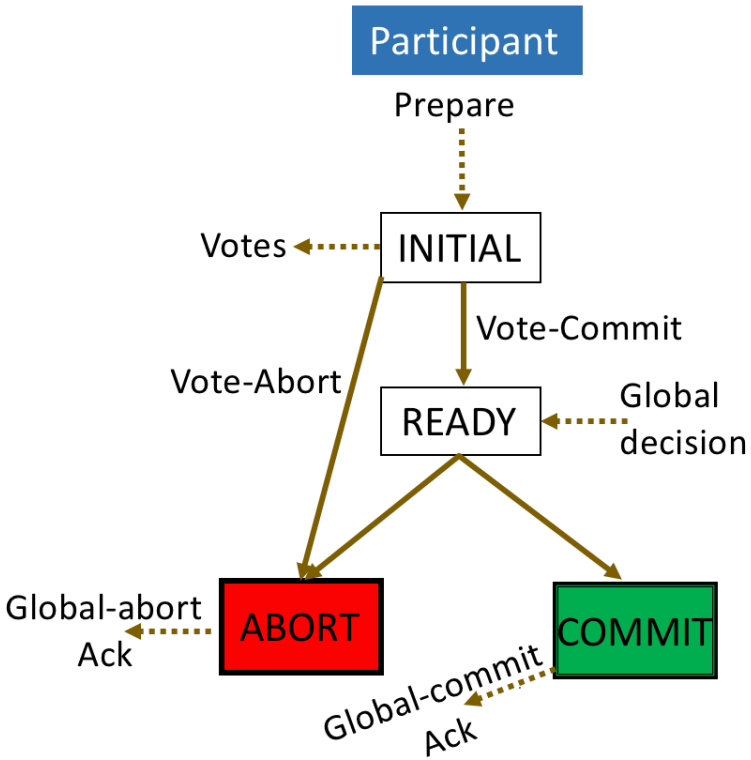
Two Phase Commit Protocol

Two Phase Commit Protocol

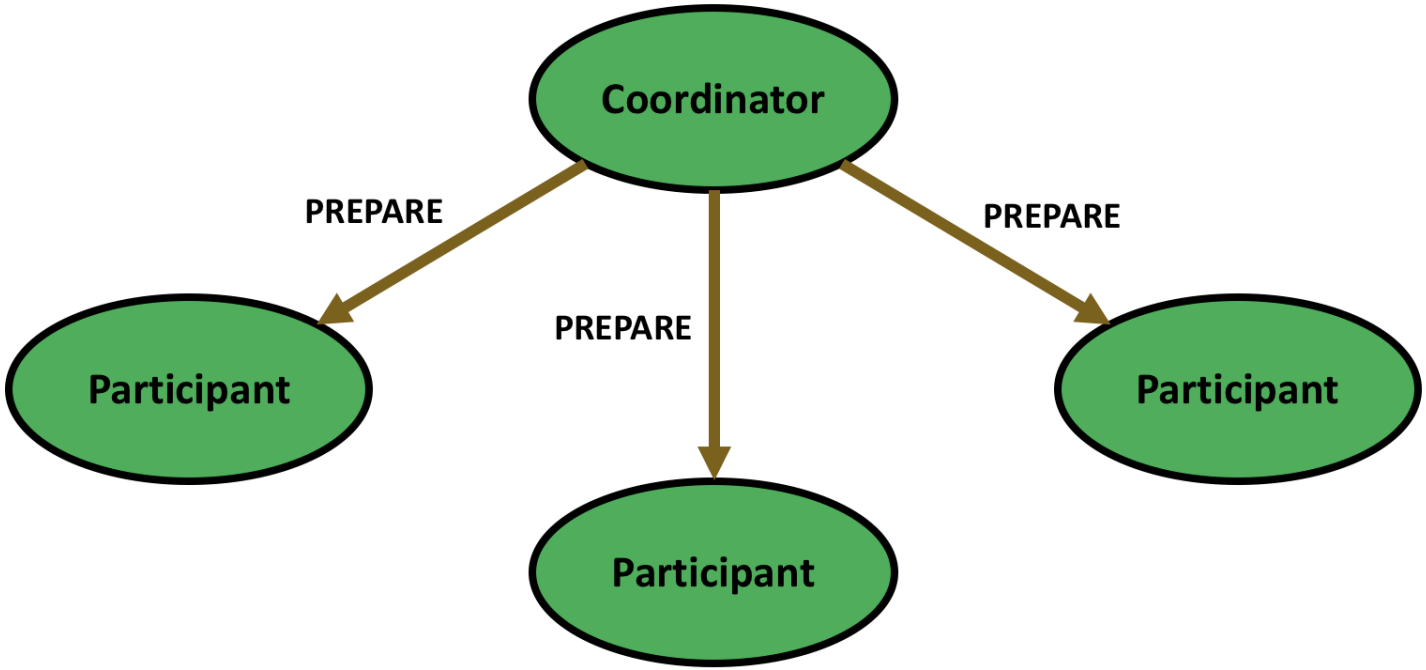


Two Phase Commit Protocol

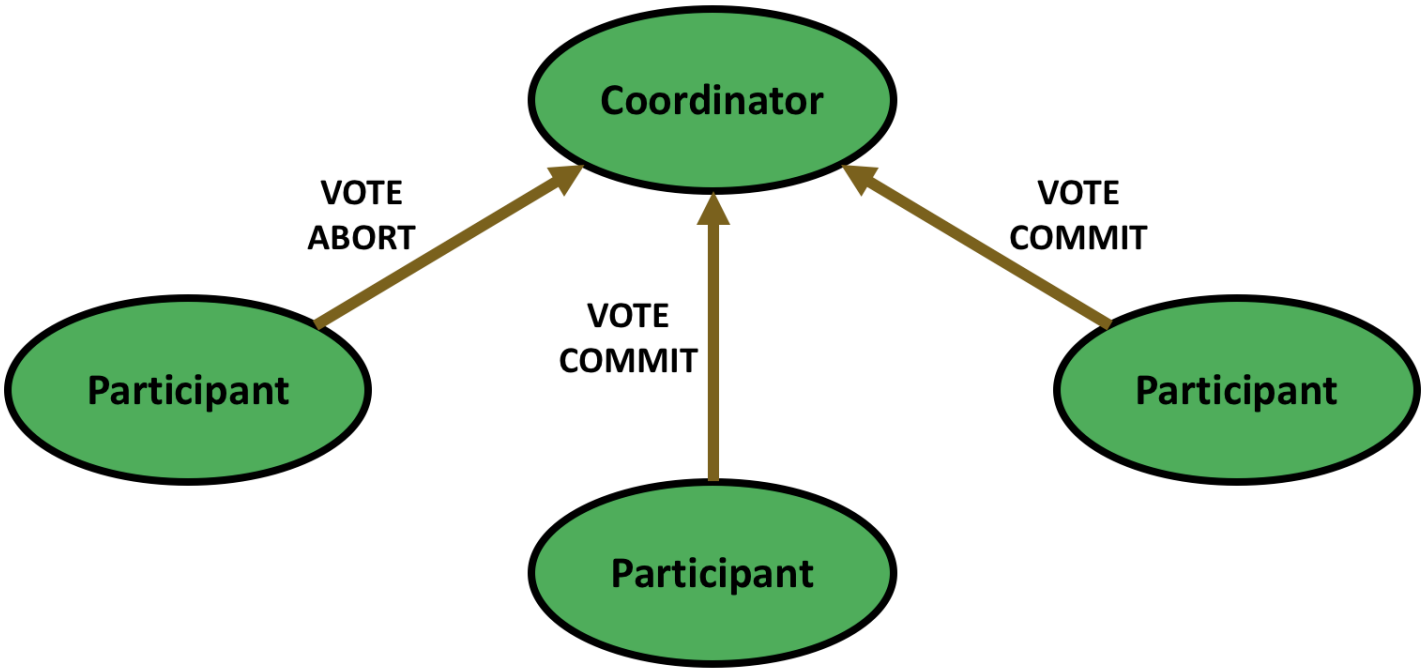




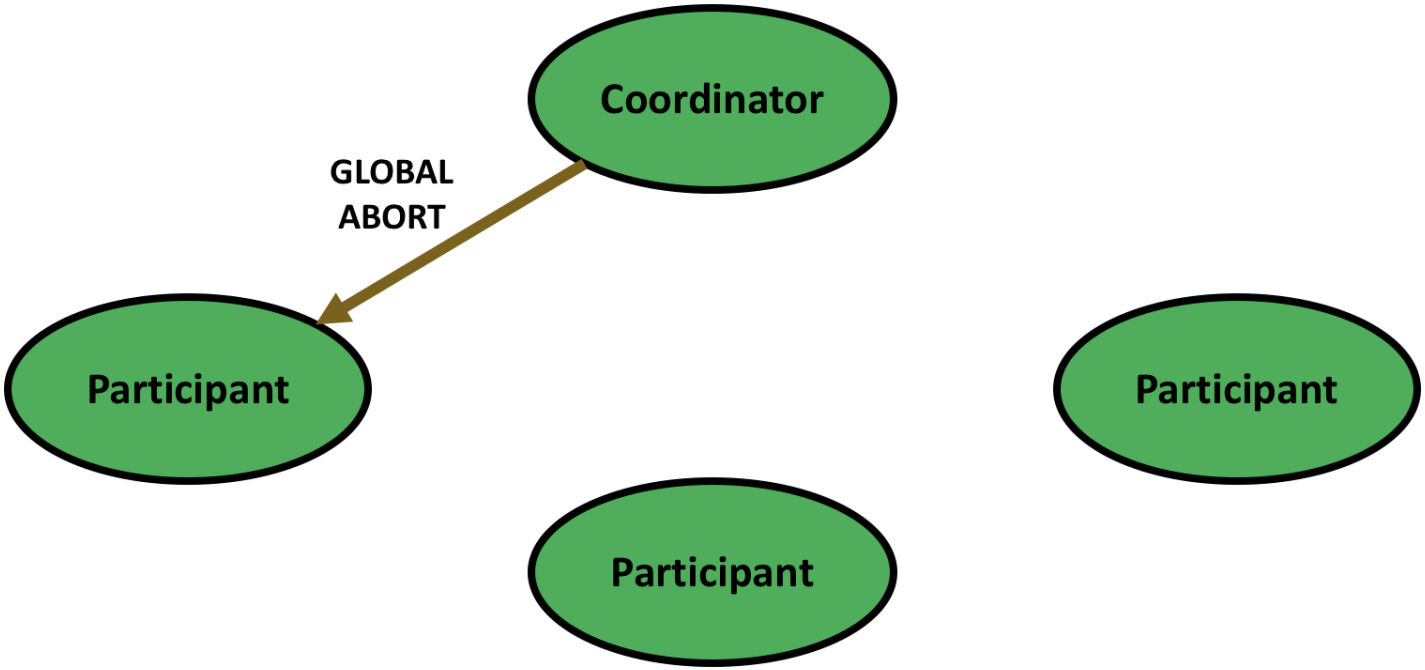
2PC is Blocking



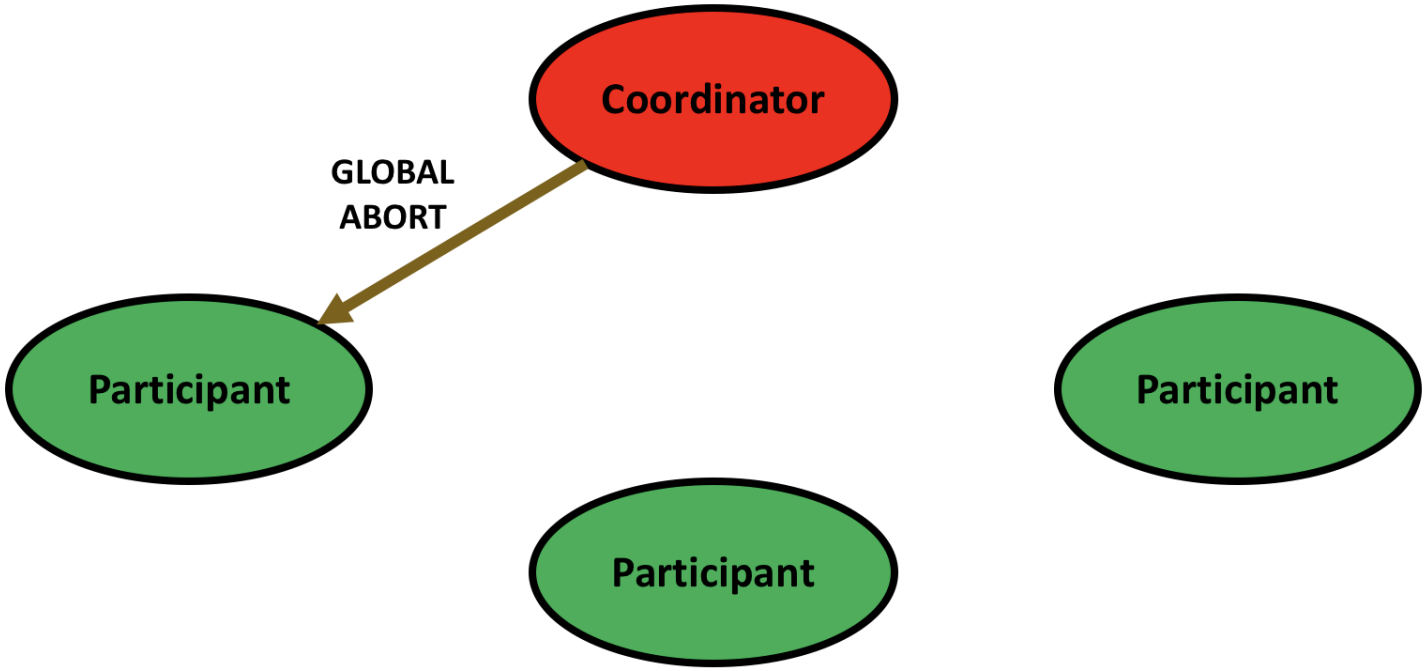
2PC is Blocking



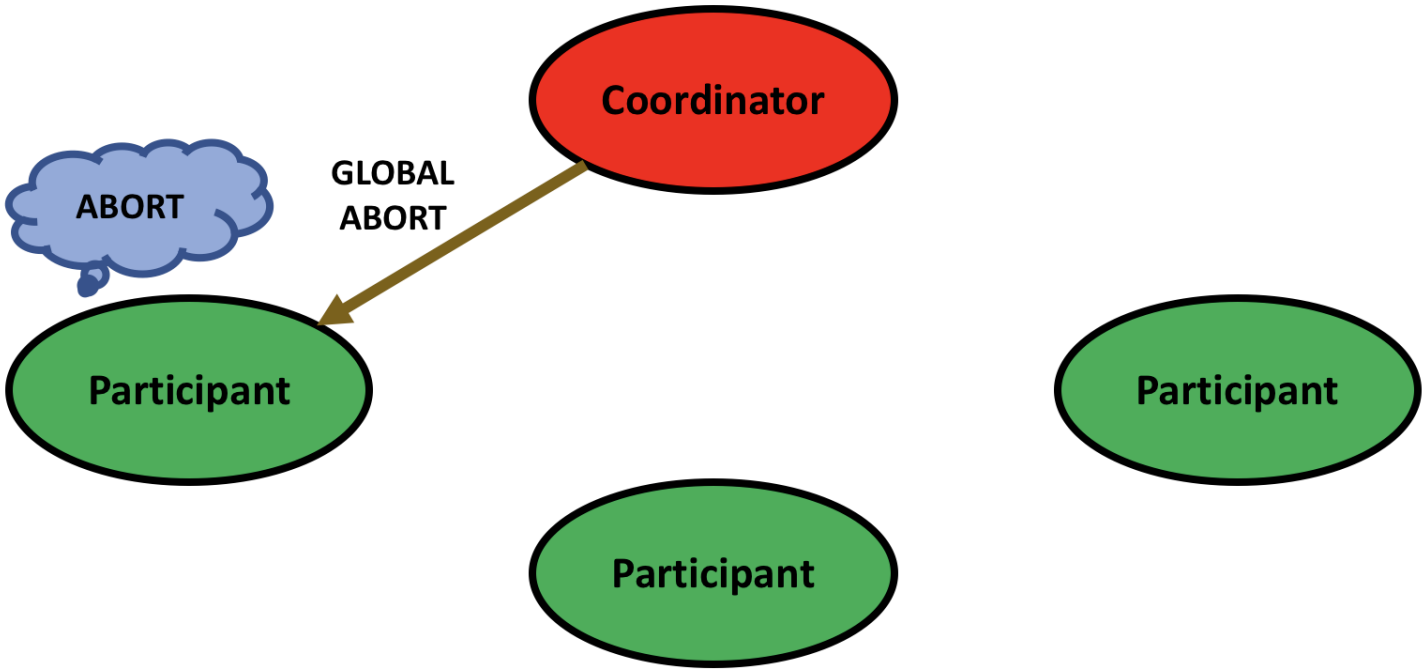
2PC is Blocking



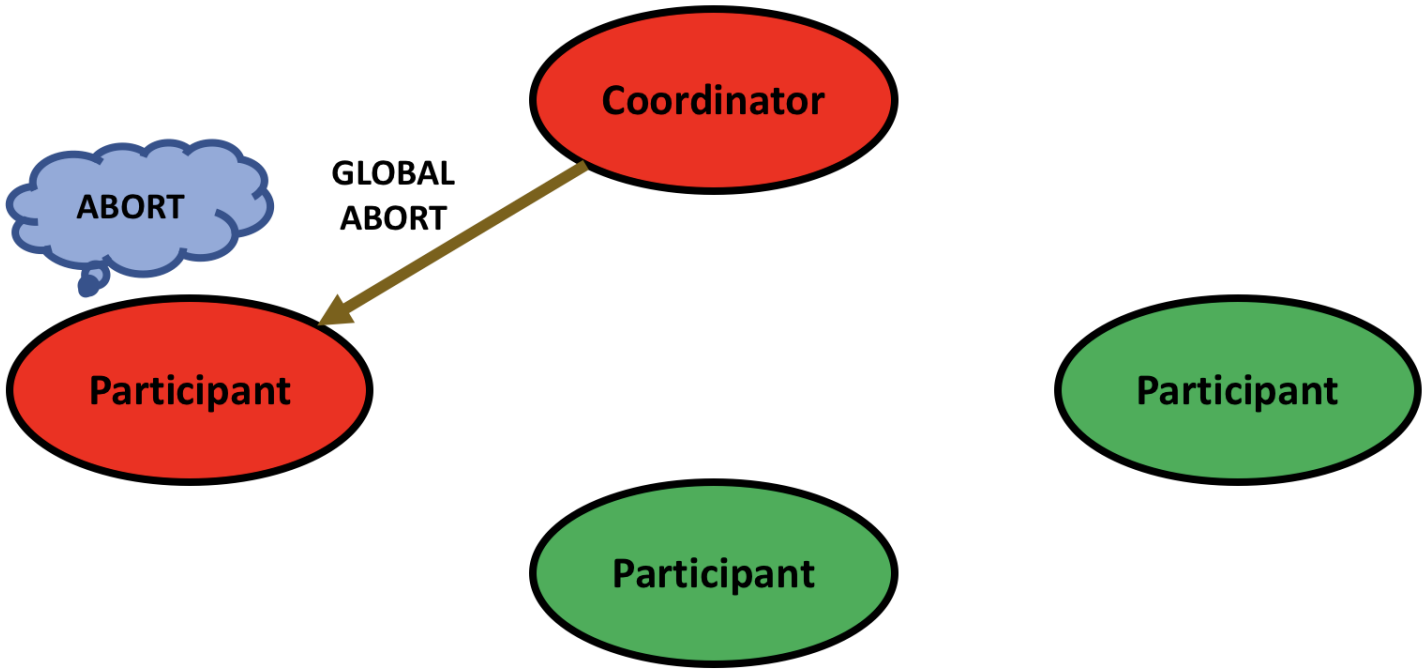
2PC is Blocking



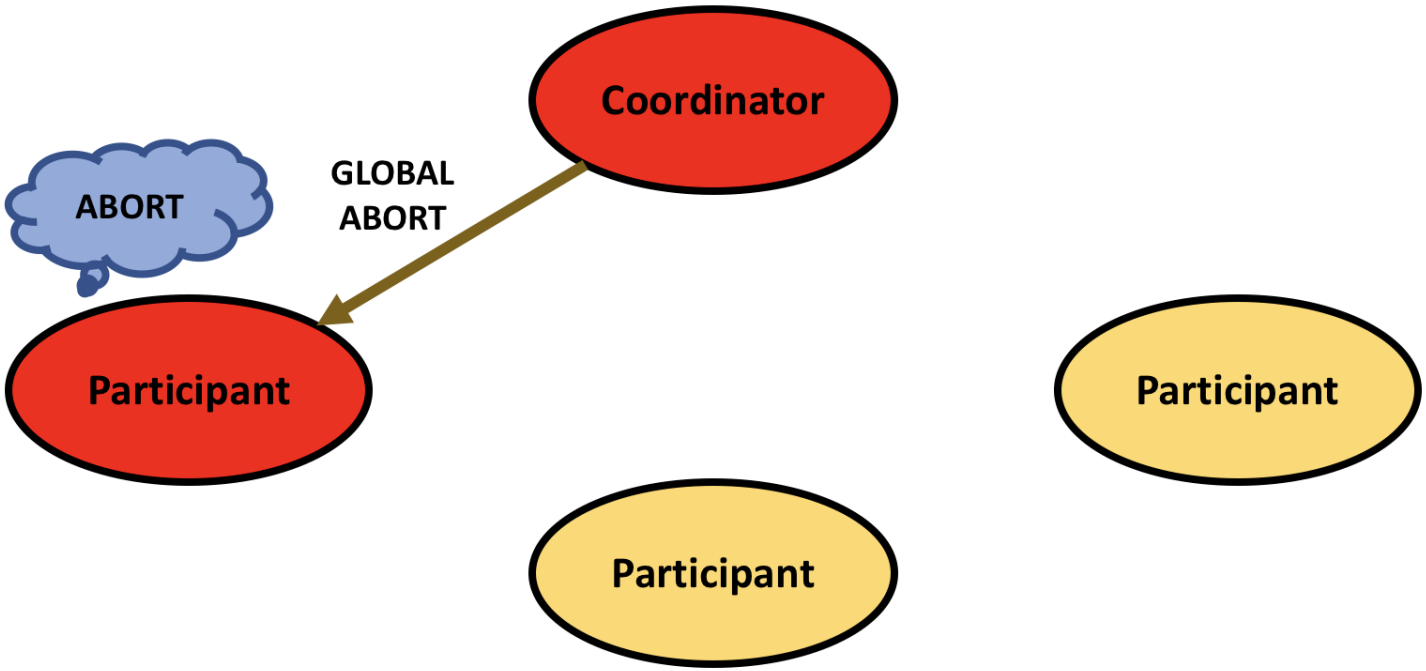
2PC is Blocking



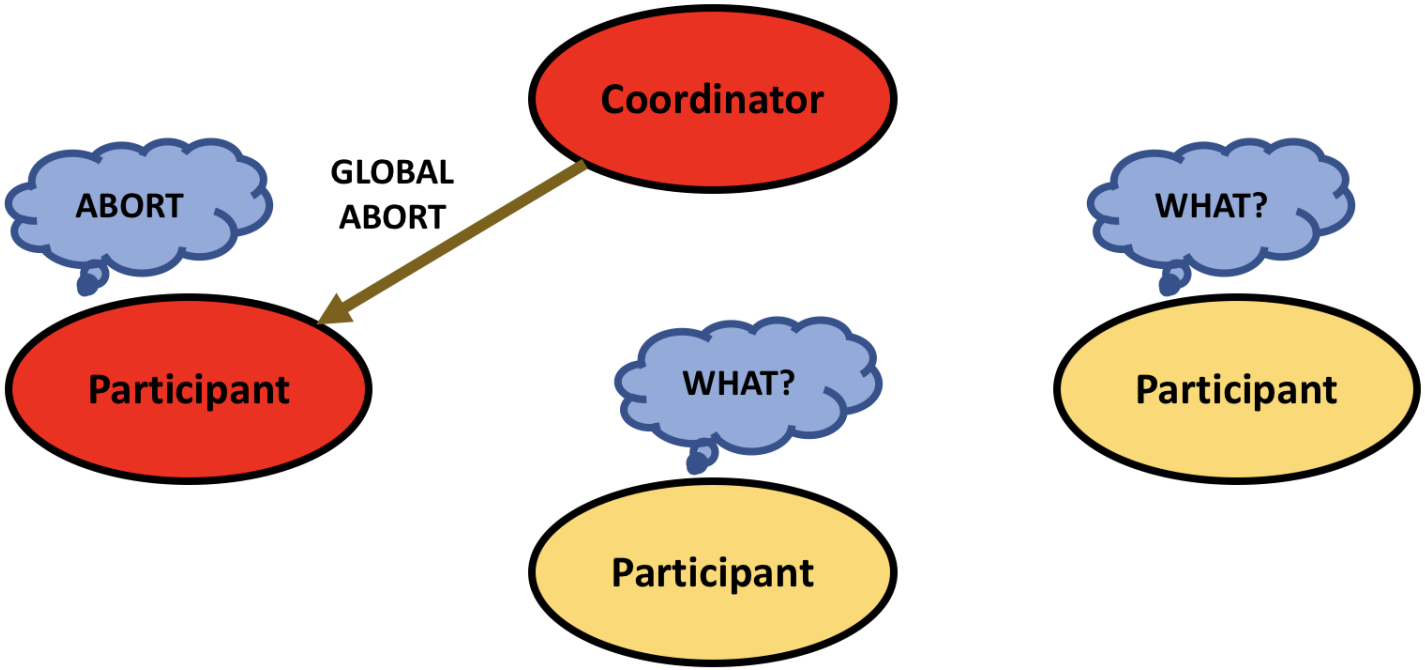
2PC is Blocking



2PC is Blocking



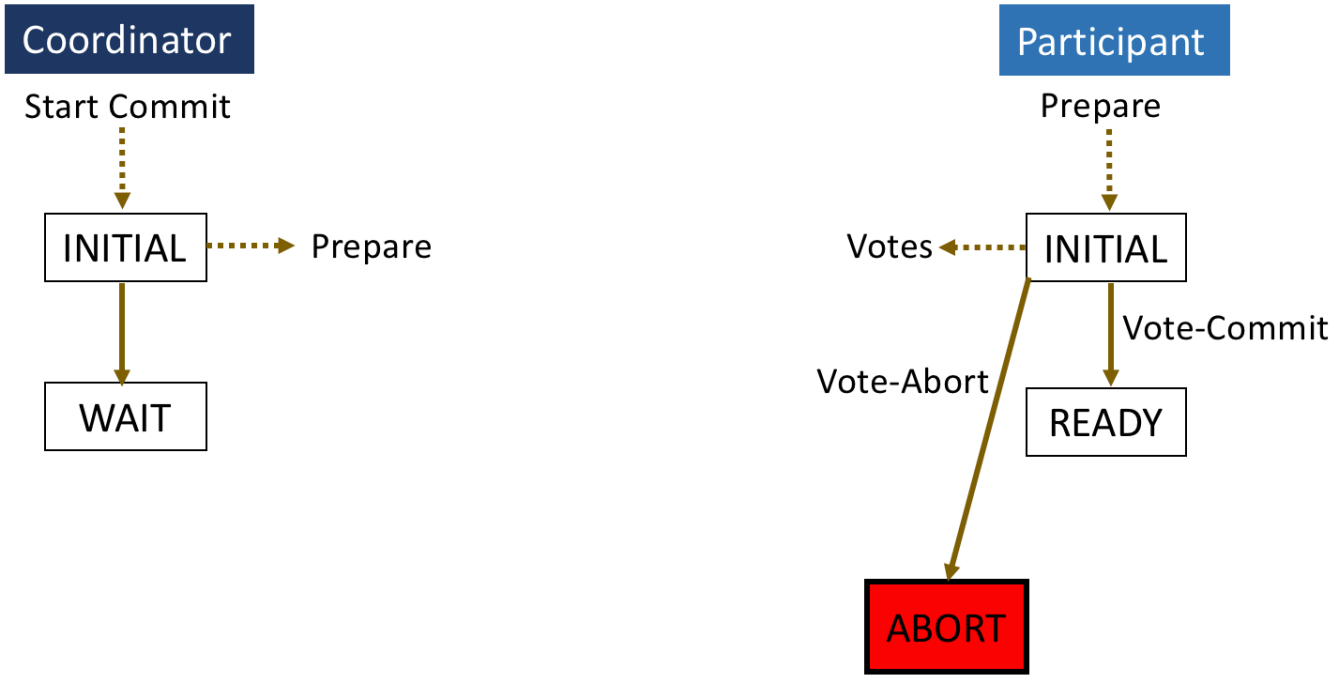
2PC is Blocking



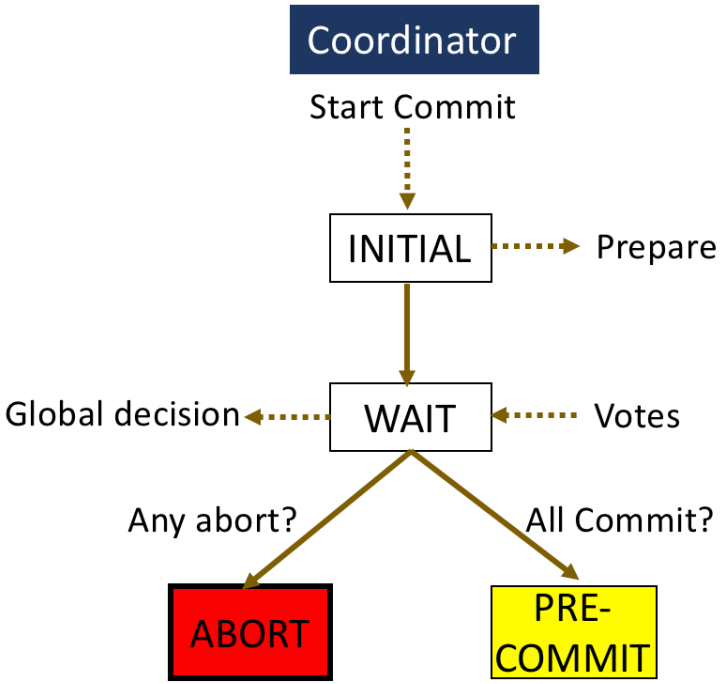
System is in an Unstable State.

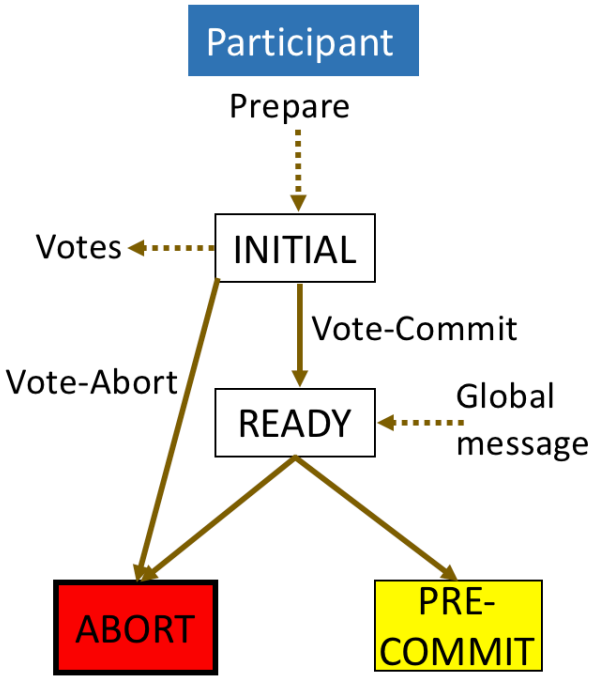
Three Phase Commit Protocol

Three Phase Commit Protocol

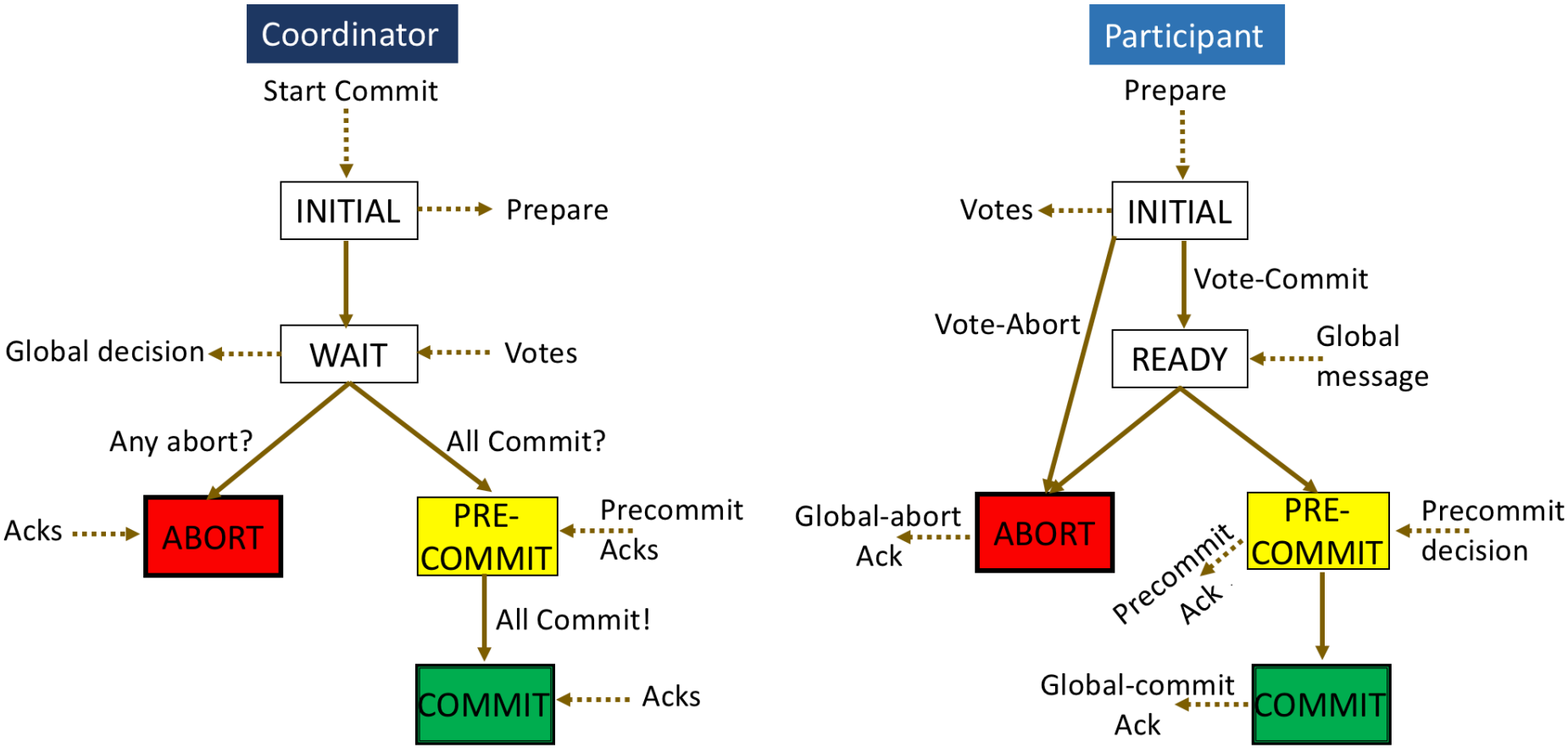


Three Phase Commit Protocol





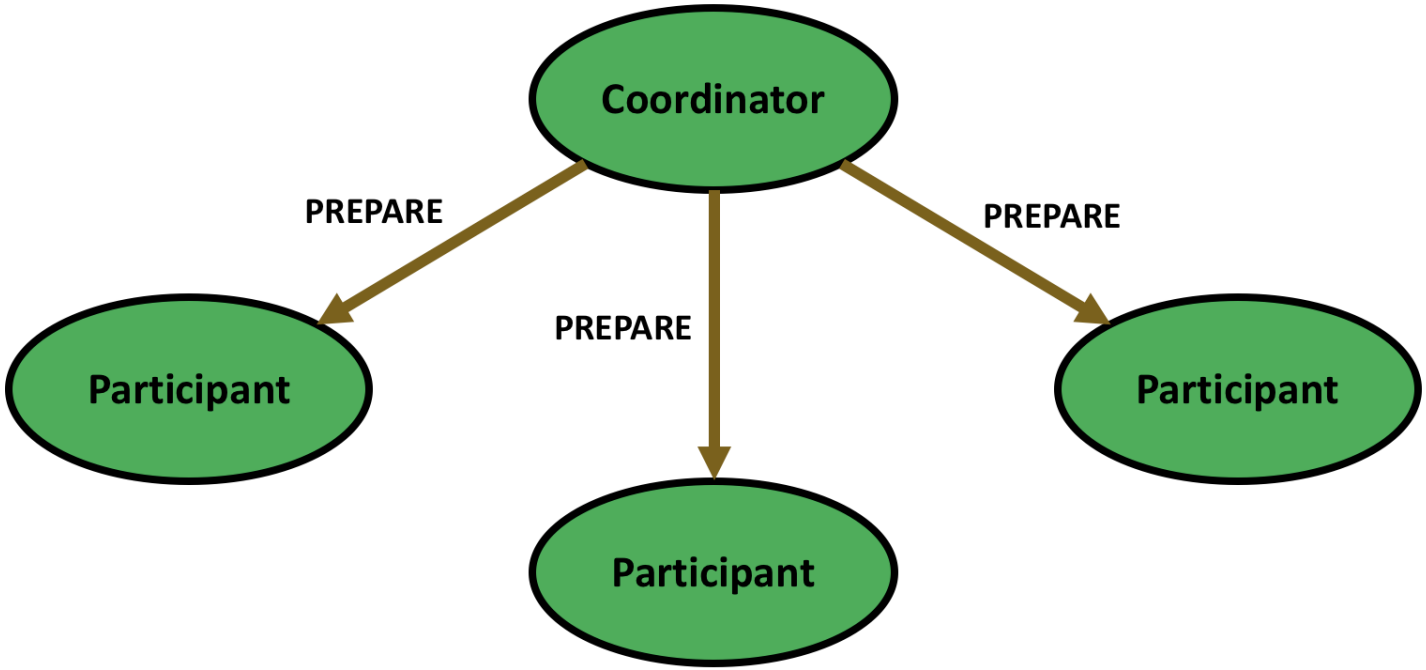
Three Phase Commit Protocol



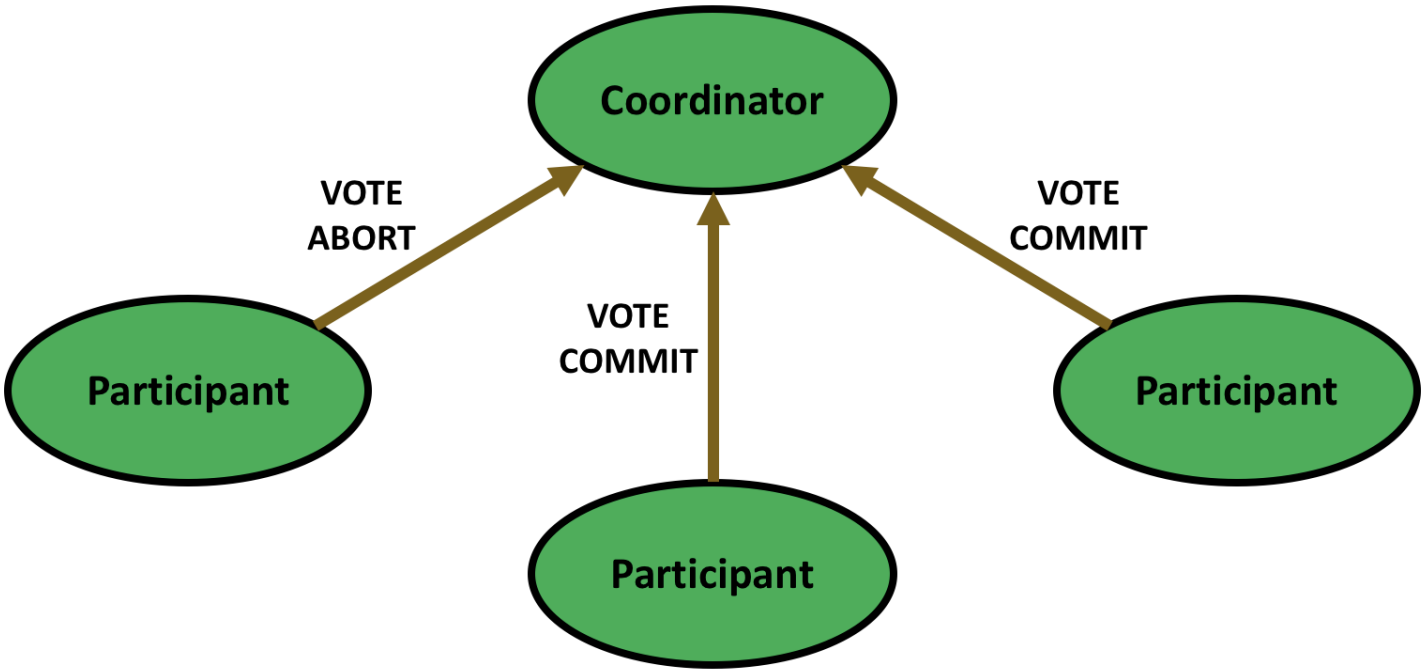
3PC is Non-Blocking

- **Dale Skeen's** Requirements [SIGMOD 1981]:
- No state adjacent to both Abort and Commit states.
- No non-committable state adjacent to the Commit state.

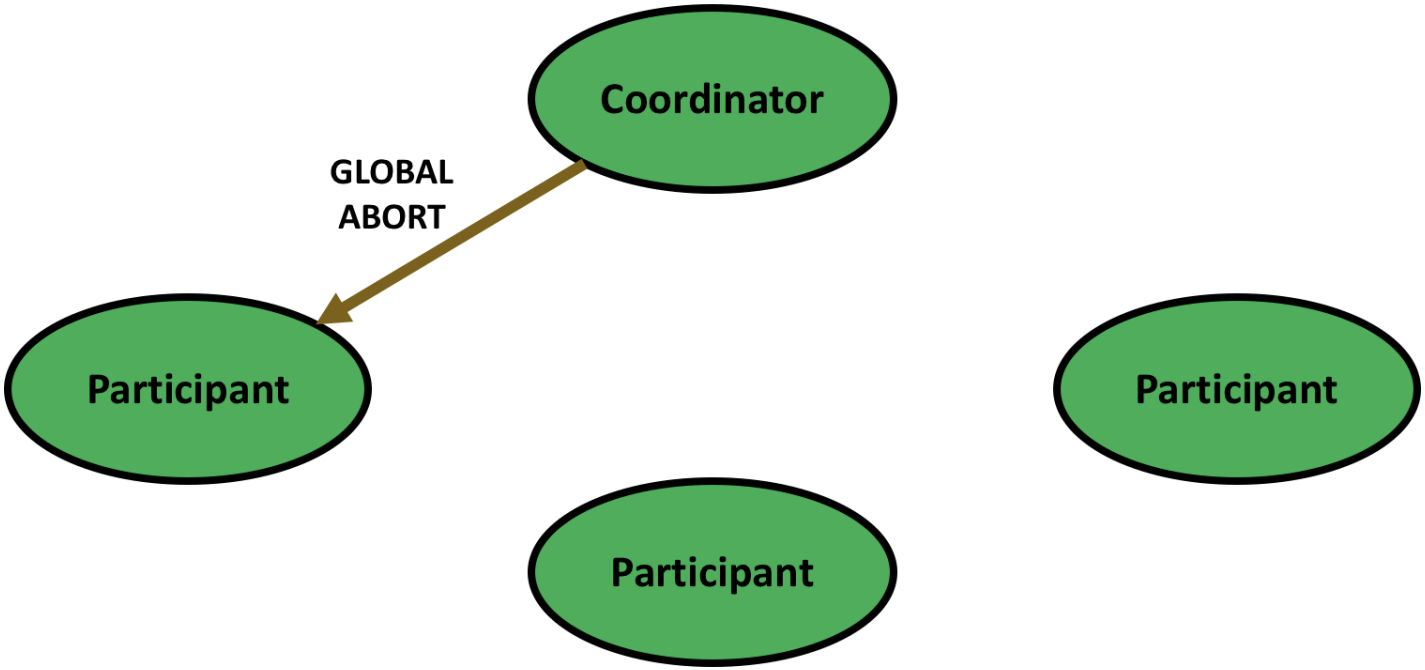
3PC Works


Coordinator
PREPARE
PREPARE
PREPARE
Participant
Participant
Participant

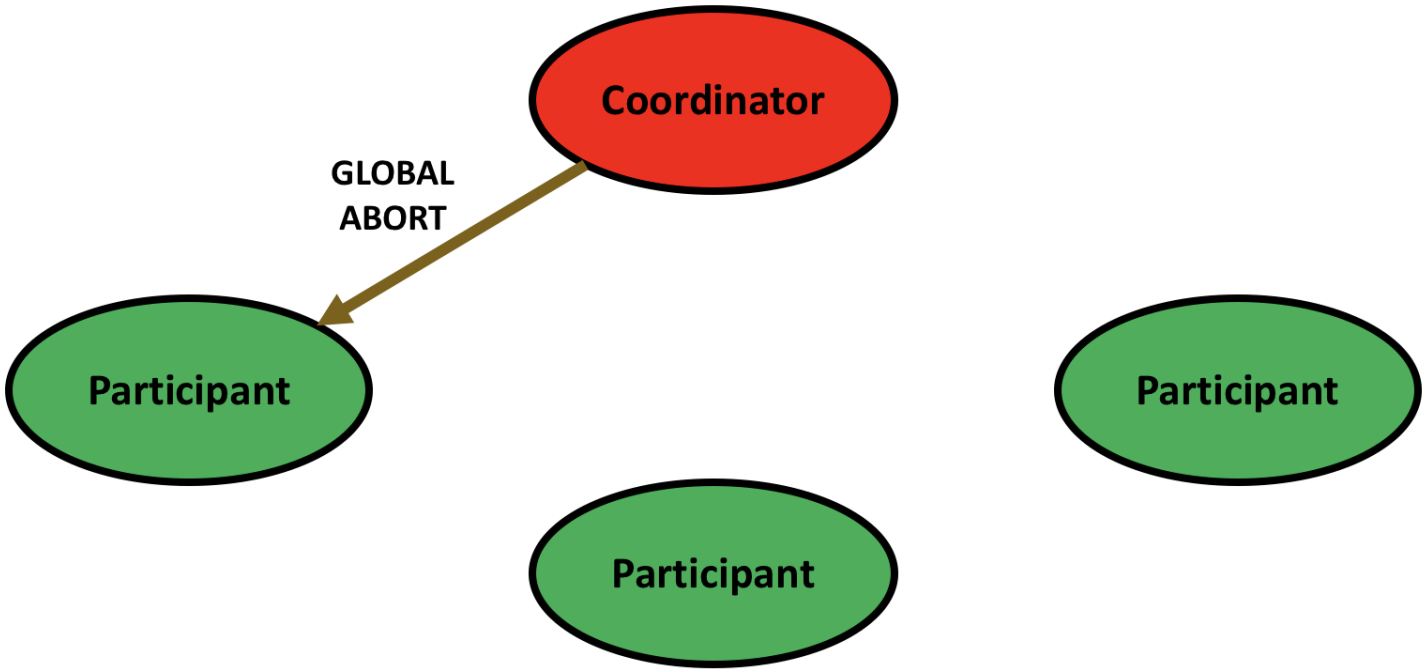
3PC Works


Coordinator
VOTE
ABORT
VOTE
COMMIT
VOTE
COMMIT
Participant
Participant
Participant

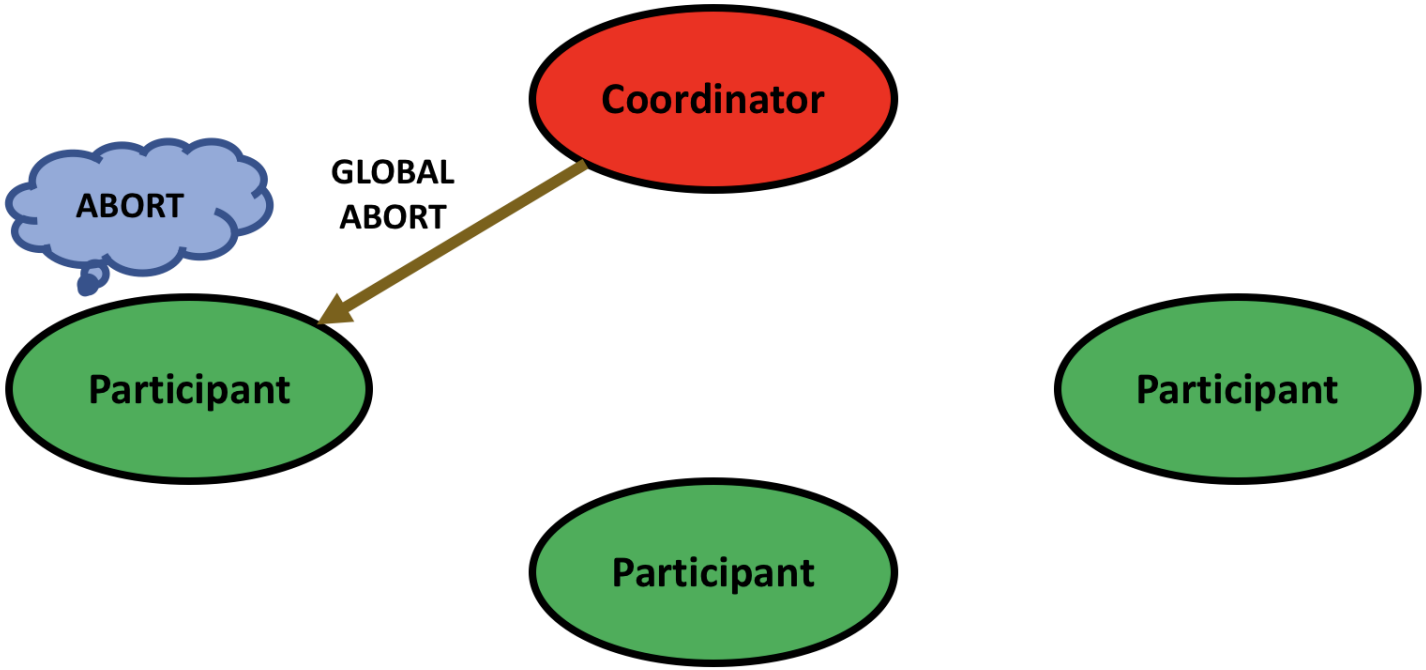
3PC Works


Coordinator
GLOBAL
ABORT
Participant
Participant
Participant

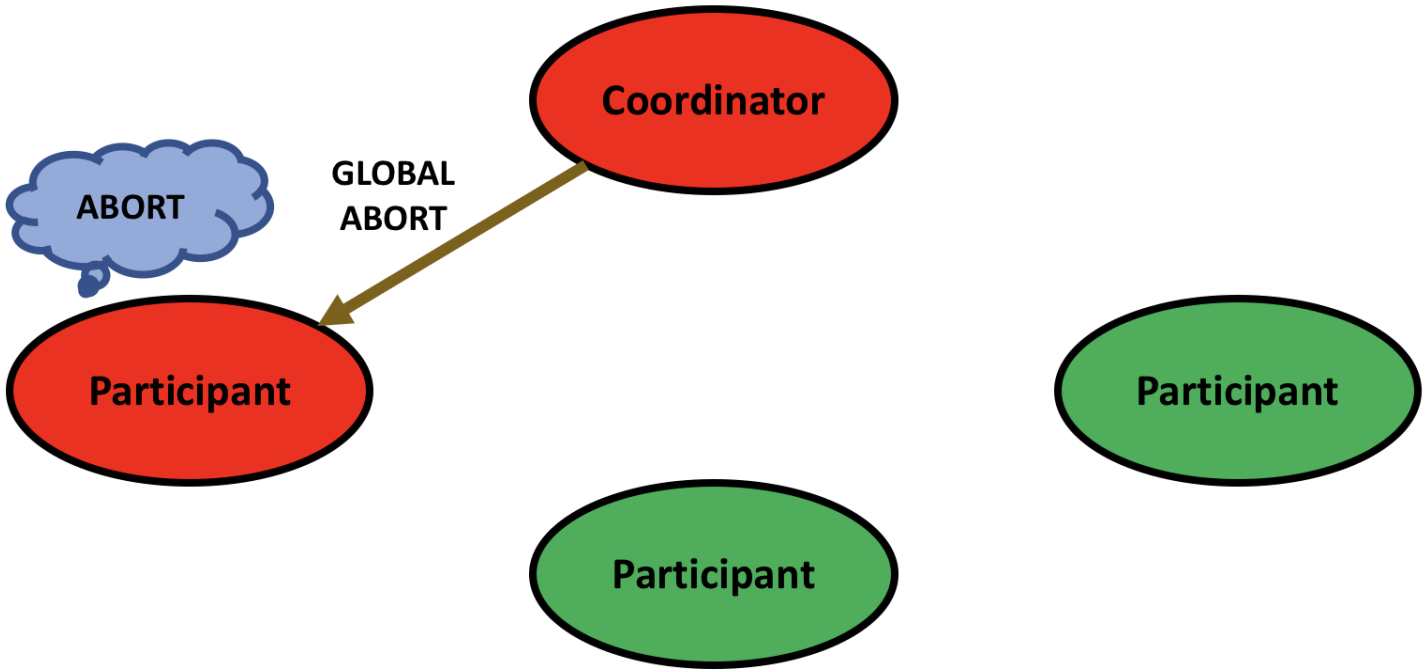
3PC Works


Coordinator
GLOBAL
ABORT
Participant
Participant
Participant

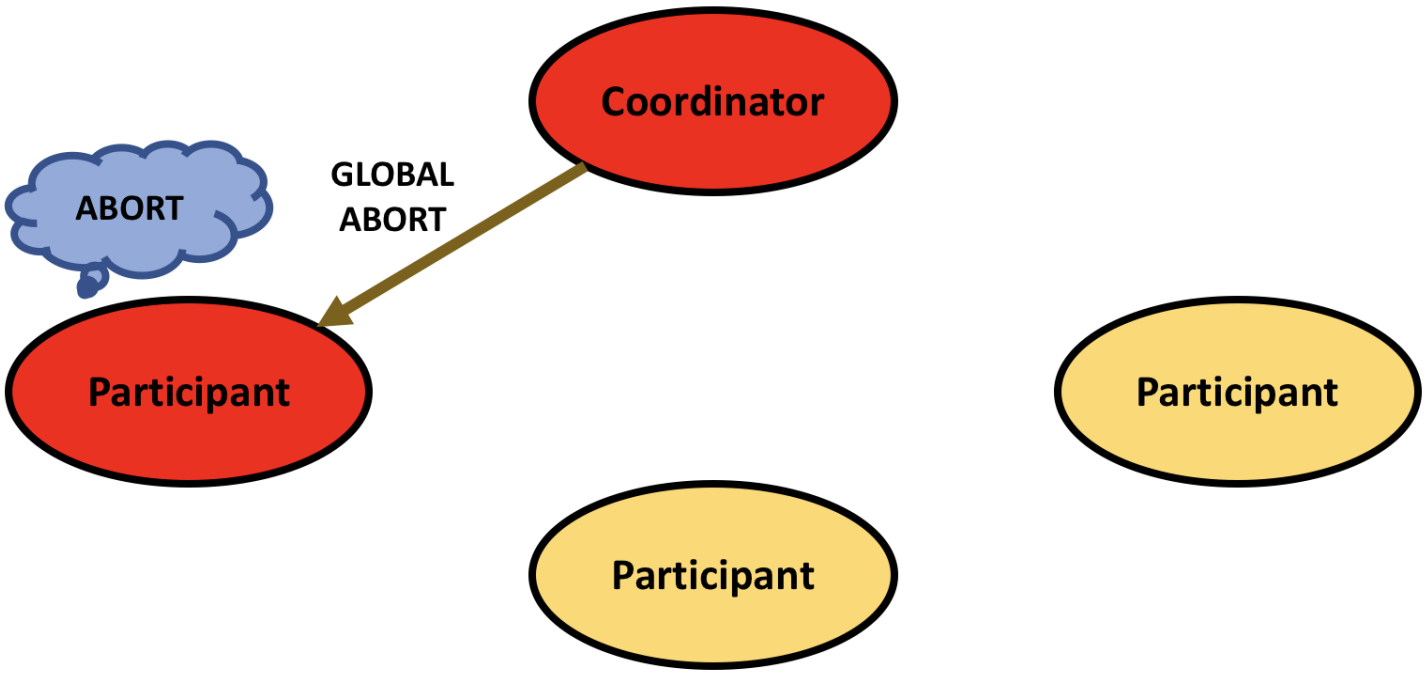
3PC Works


Coordinator
ABORT
GLOBAL
ABORT
Participant
Participant
Participant

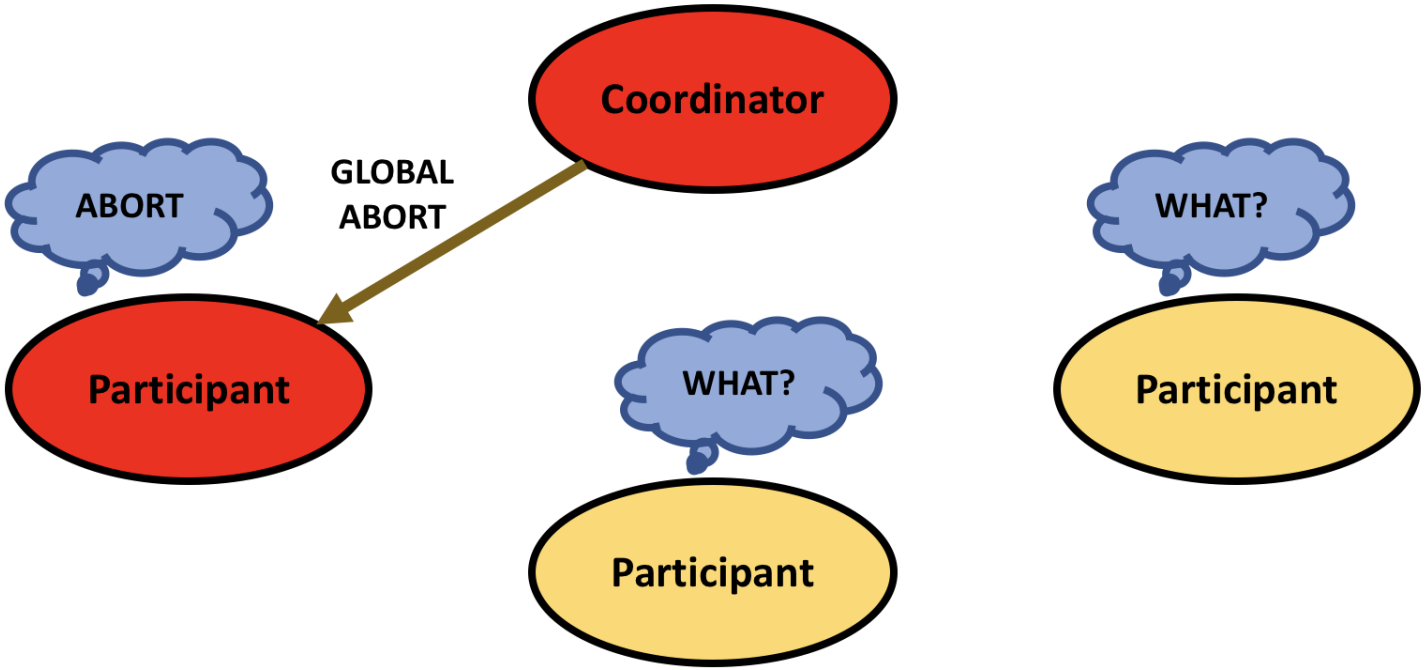
3PC Works


Coordinator
ABORT
GLOBAL
ABORT
Participant
Participant
Participant

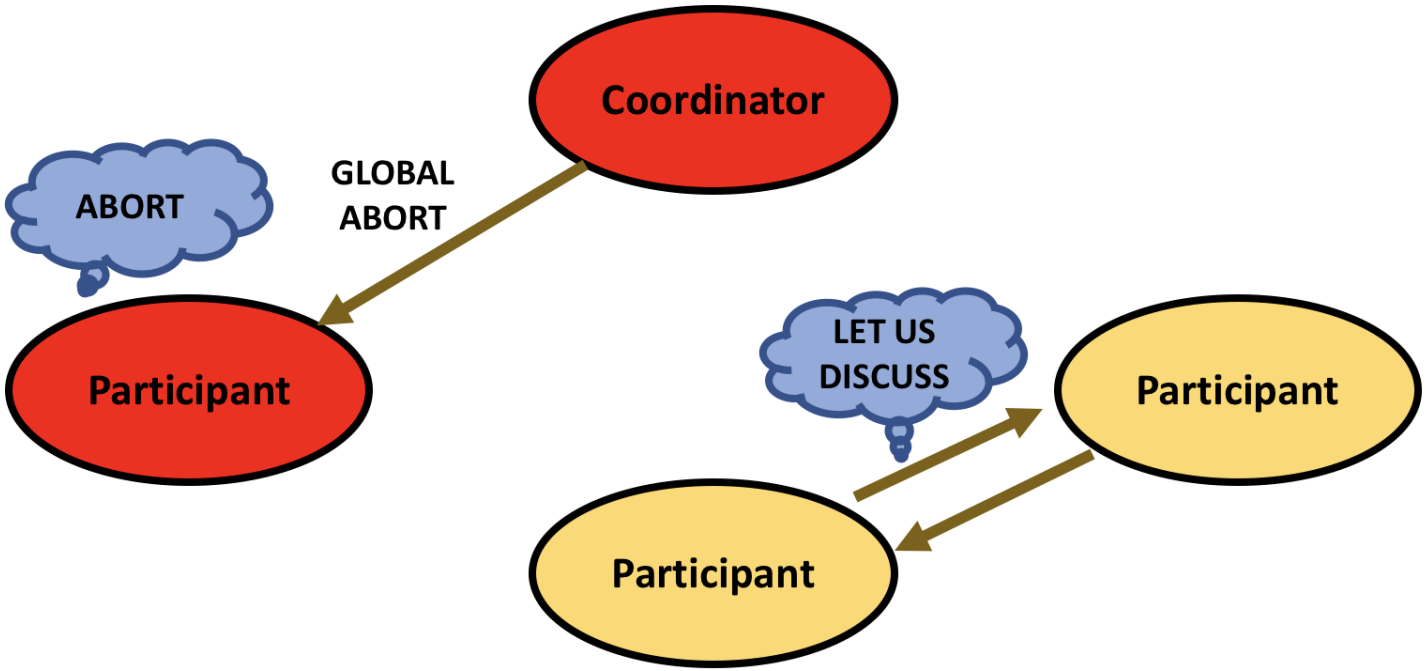
3PC Works


Coordinator
ABORT
GLOBAL
ABORT
Participant
Participant
Participant

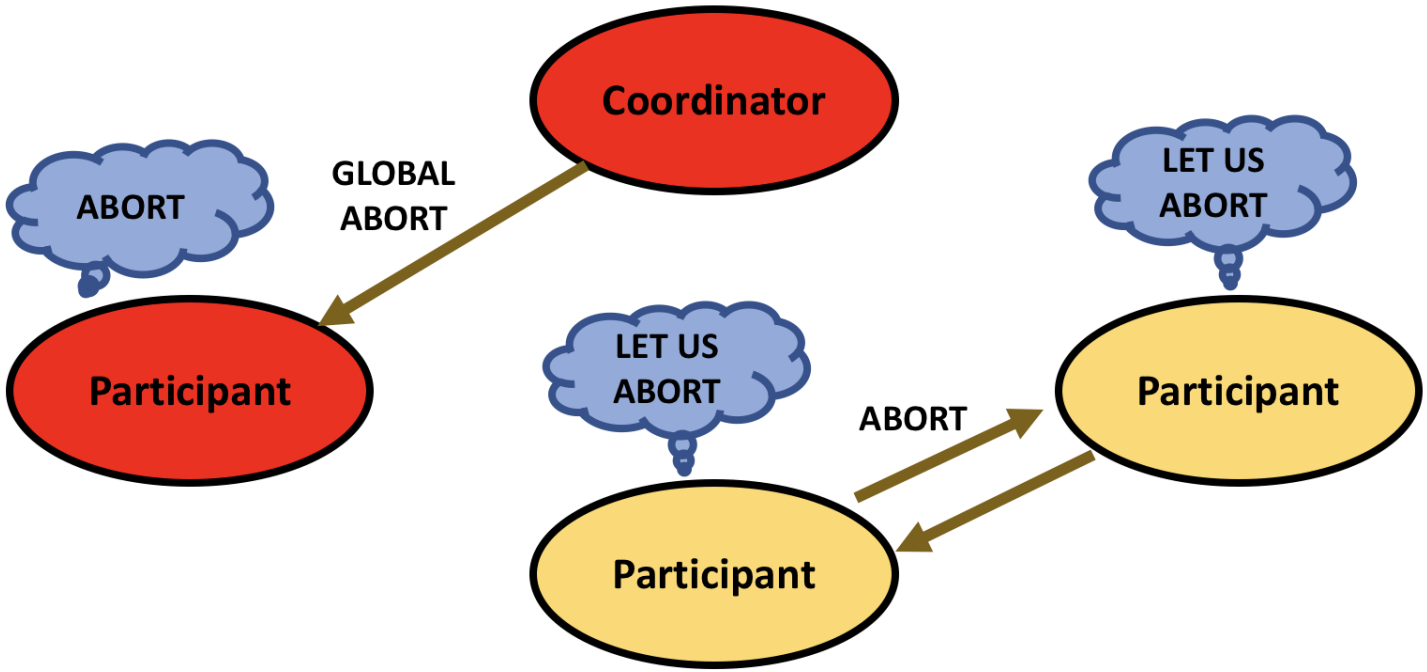
3PC Works


Coordinator
GLOBAL ABORT
ABORT
Participant
WHAT?
Participant
WHAT?
Participant

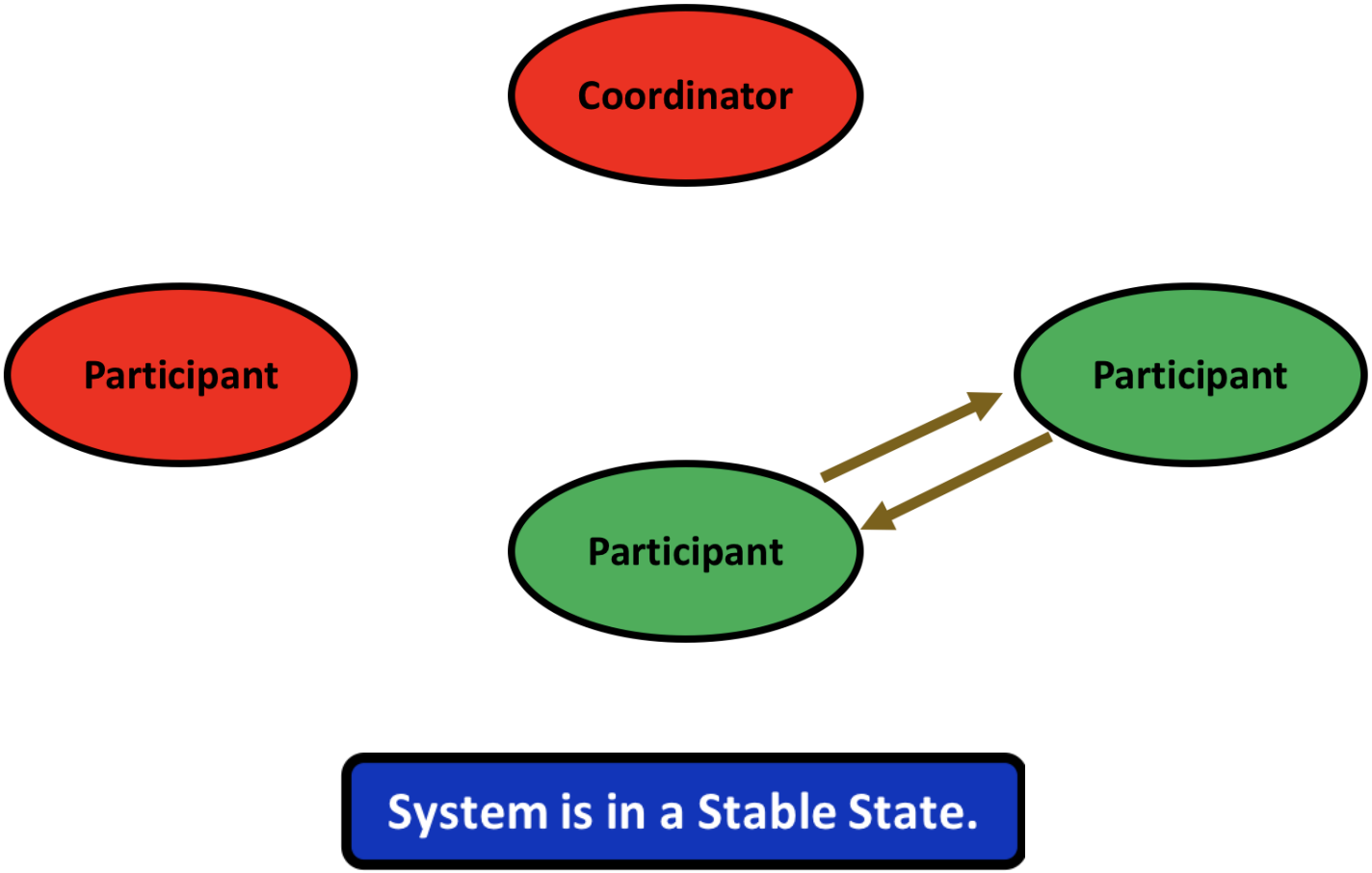
3PC Works


Coordinator
ABORT
GLOBAL
ABORT
Participant
LET US
DISCUSS
Participant
Participant

3PC Works


Coordinator
GLOBAL ABORT
ABORT
Participant
LET US ABORT
Participant
LET US ABORT
ABORT
Participant

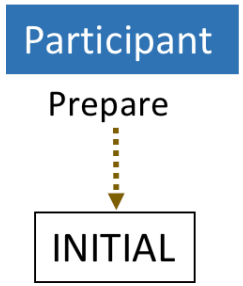
3PC Works


Coordinator
Participant
Participant
Participant
System is in a Stable State.

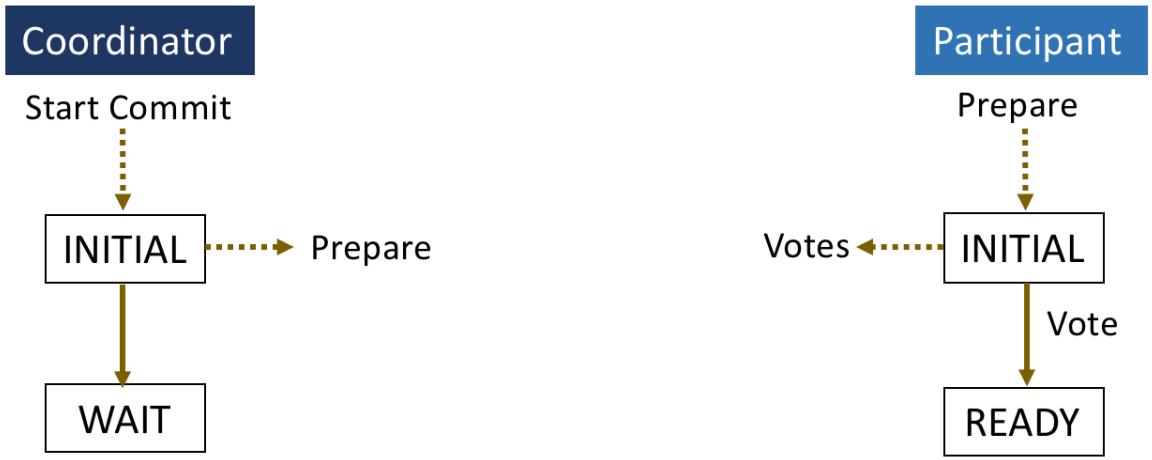
Easy Commit Principle

- First Transmit and then Commit
- Message Redundancy

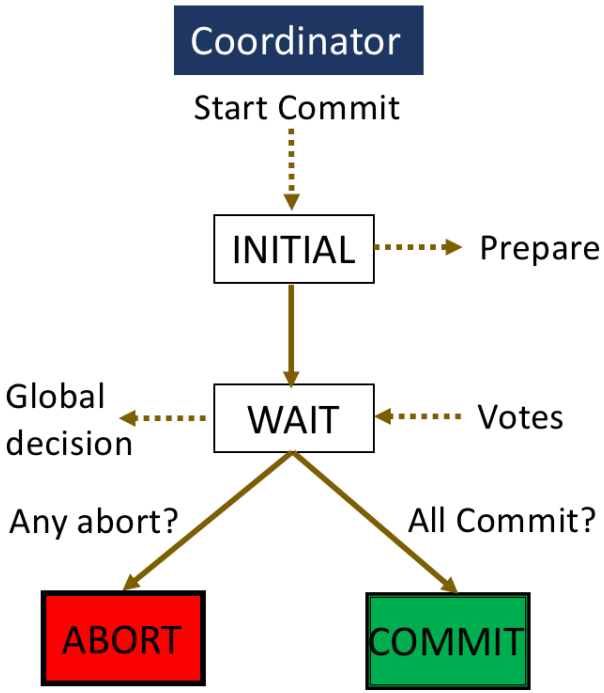
Easy Commit Protocol

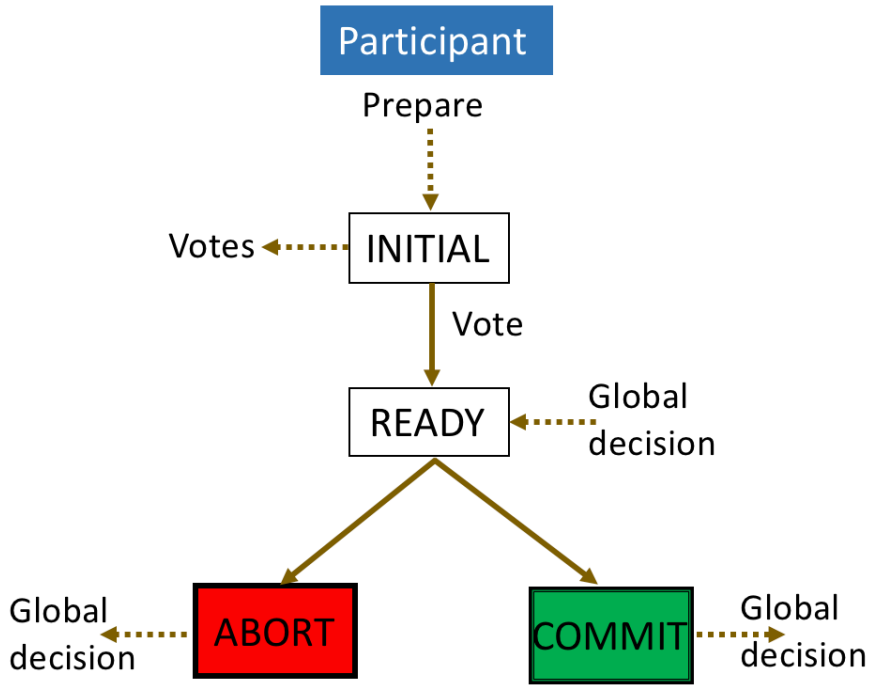


Easy Commit Protocol



Easy Commit Protocol





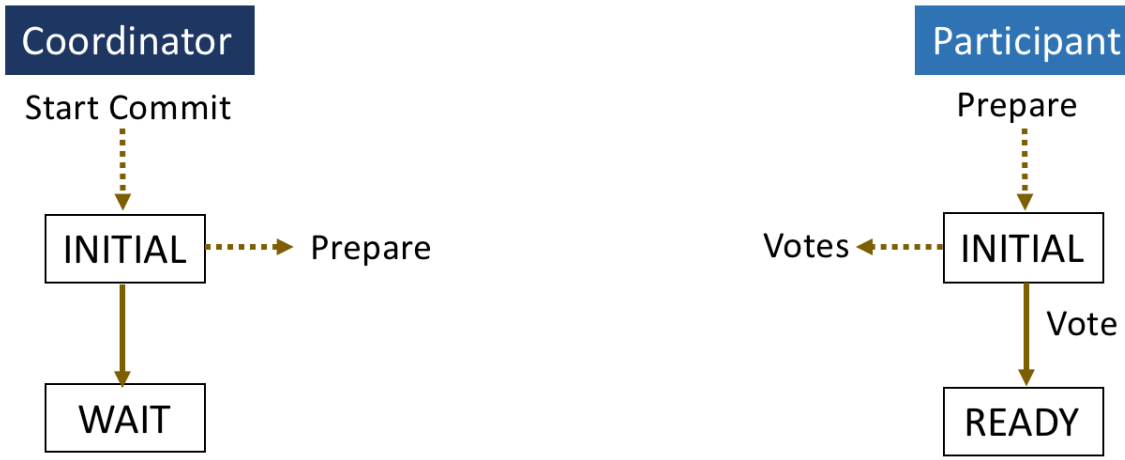
Easy Commit Observations

- Participant cannot directly transition from INITIAL to ABORT.
- Each participant forwards the global decision to every other node.
- Participant need not wait for global decision from coordinator.
- Existence of hidden states: **TRANSMIT-A** and **TRANSMIT-C**.

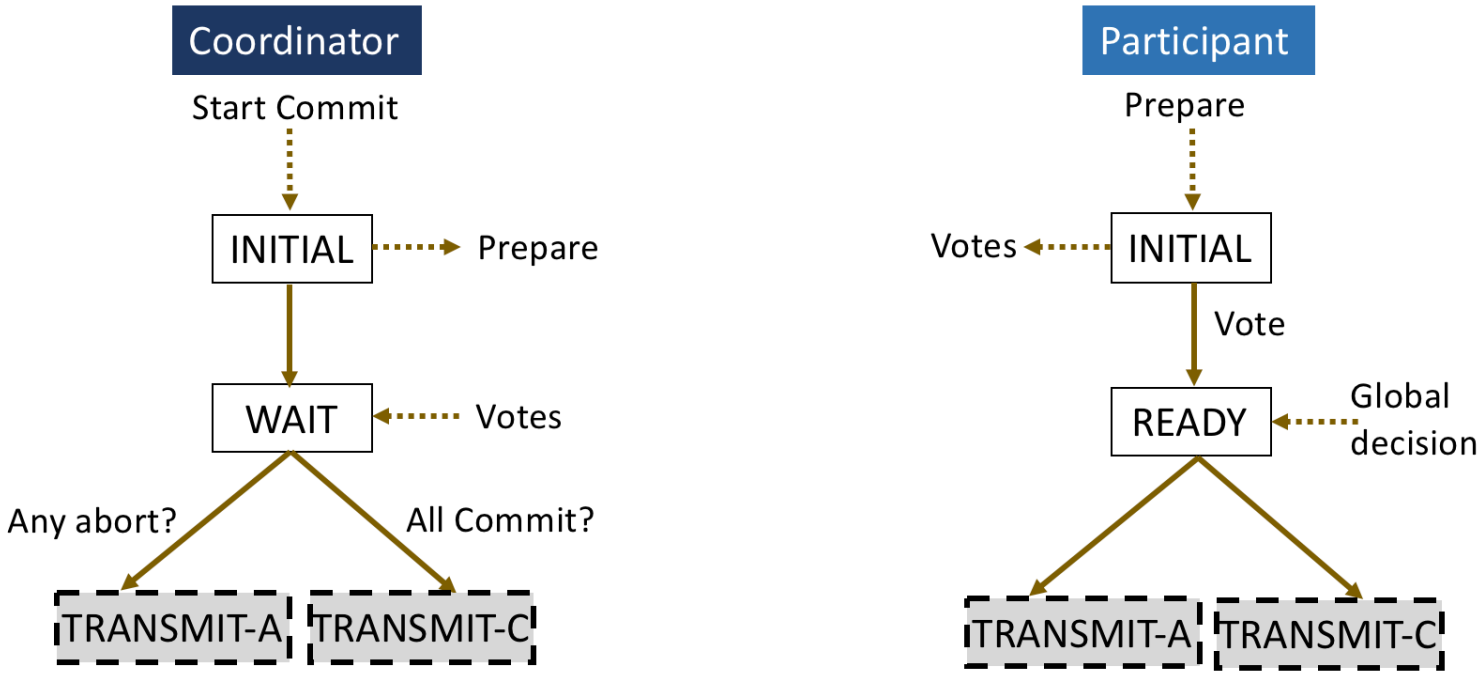
UC DAVIS UNIVERSITY OF CALIFORNIA

Easy Commit Protocol – Logical Expansion

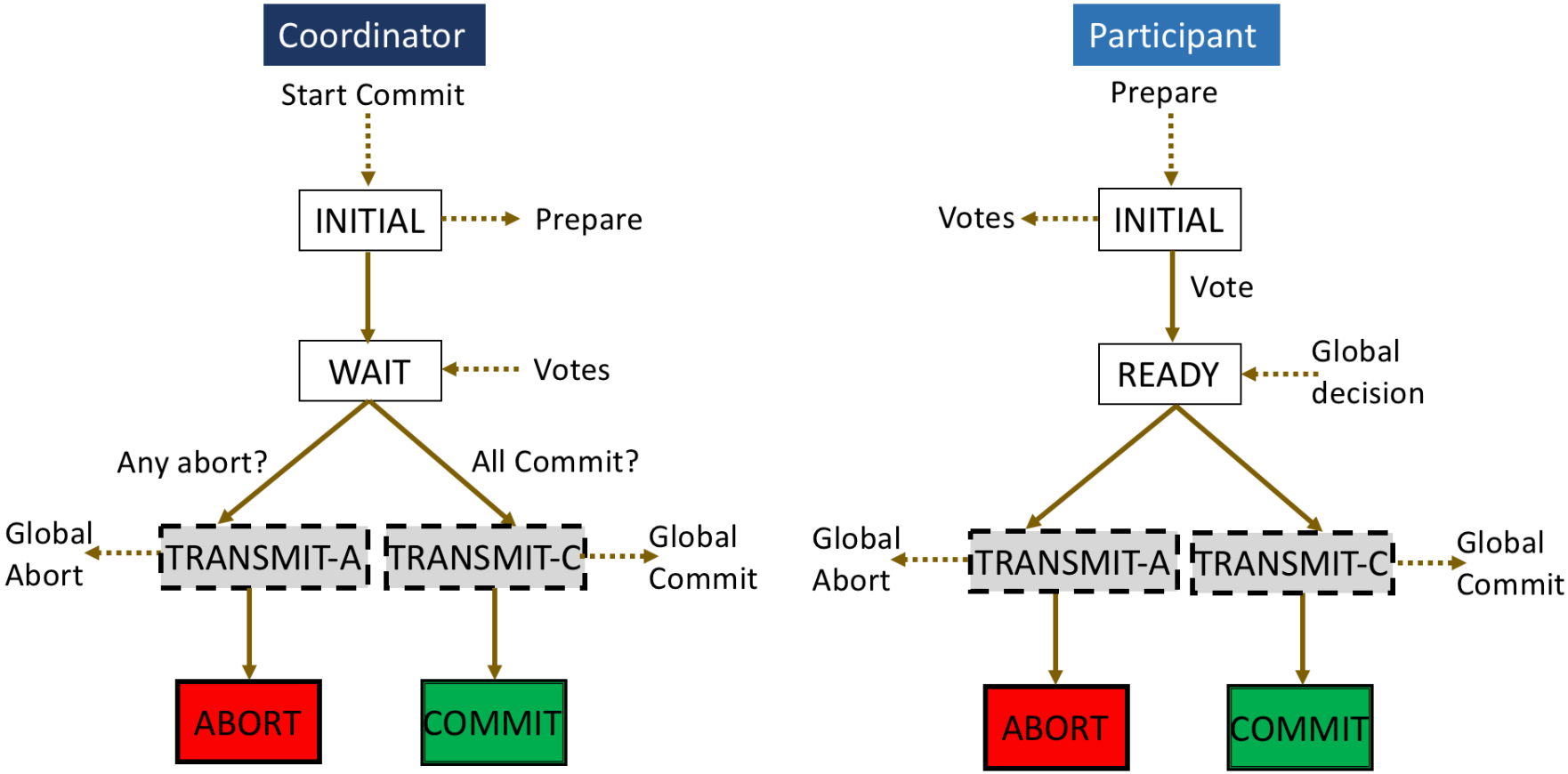
Easy Commit Protocol – Logical Expansion



Easy Commit Protocol – Logical Expansion



Easy Commit Protocol – Logical Expansion

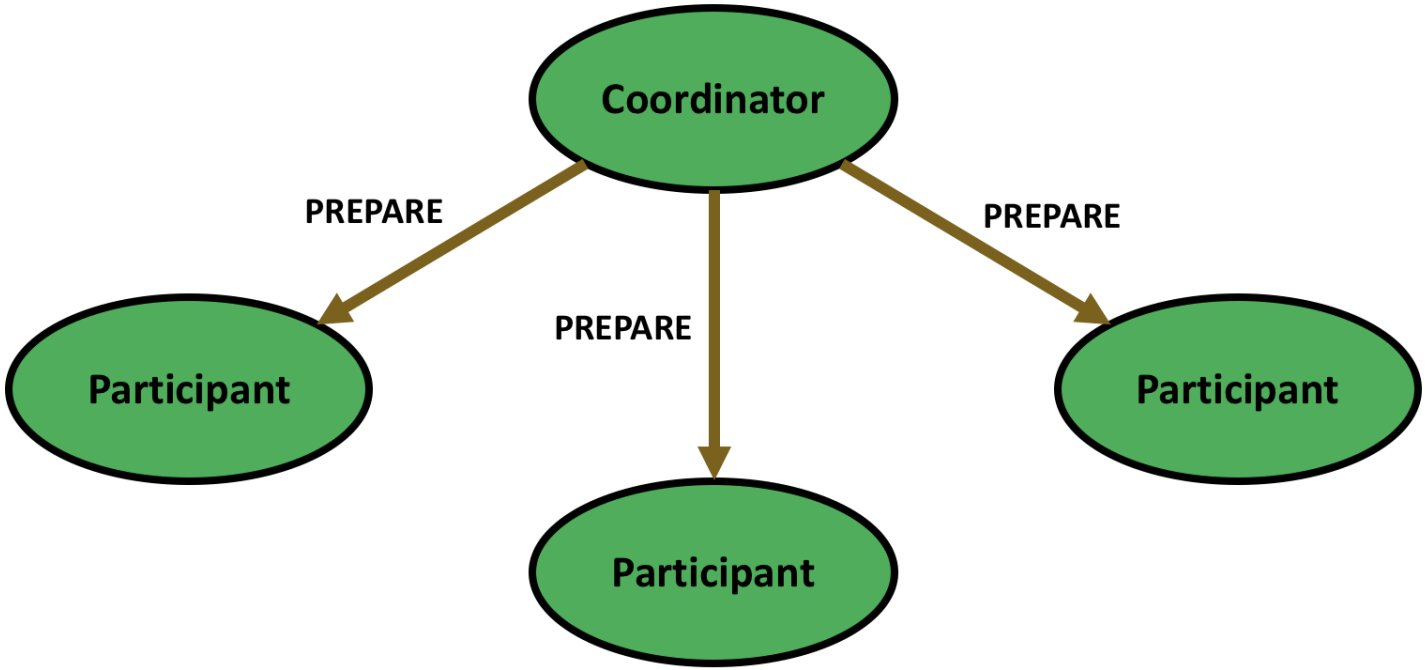


Easy Commit Termination Protocol

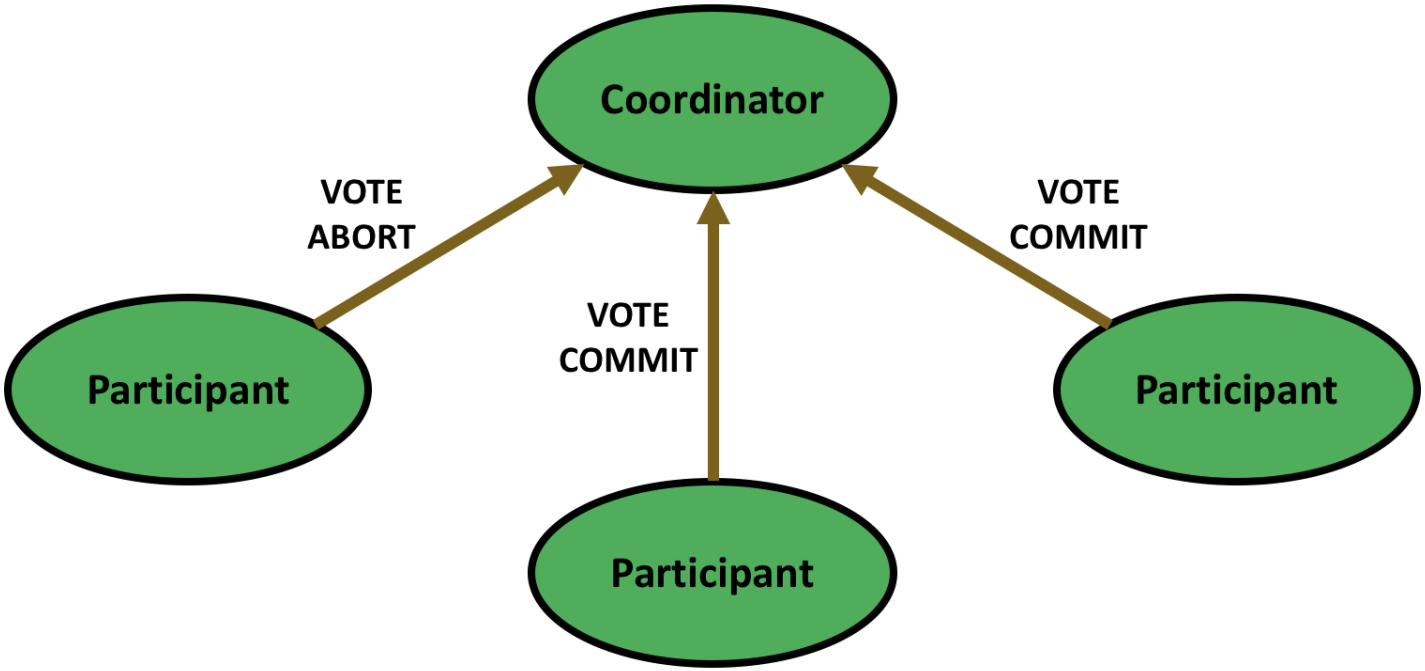
- **Coordinator Timeout in WAIT state**:
  - Coordinator didn't receive **Votes**.
  - Adds a log entry, Transmits **Global-Abort**, Aborts Transaction.
- **Participant Timeout in INITAL state**:
  - Participant didn't receive **PREPARE** message.
  - Communicates with other participants.
- **Participant Timeout in READY state**:
  - Participant didn't receive **Global Decision**.
  - Communicates with other participants.

UC DAVIS UNIVERSITY OF CALIFORNIA

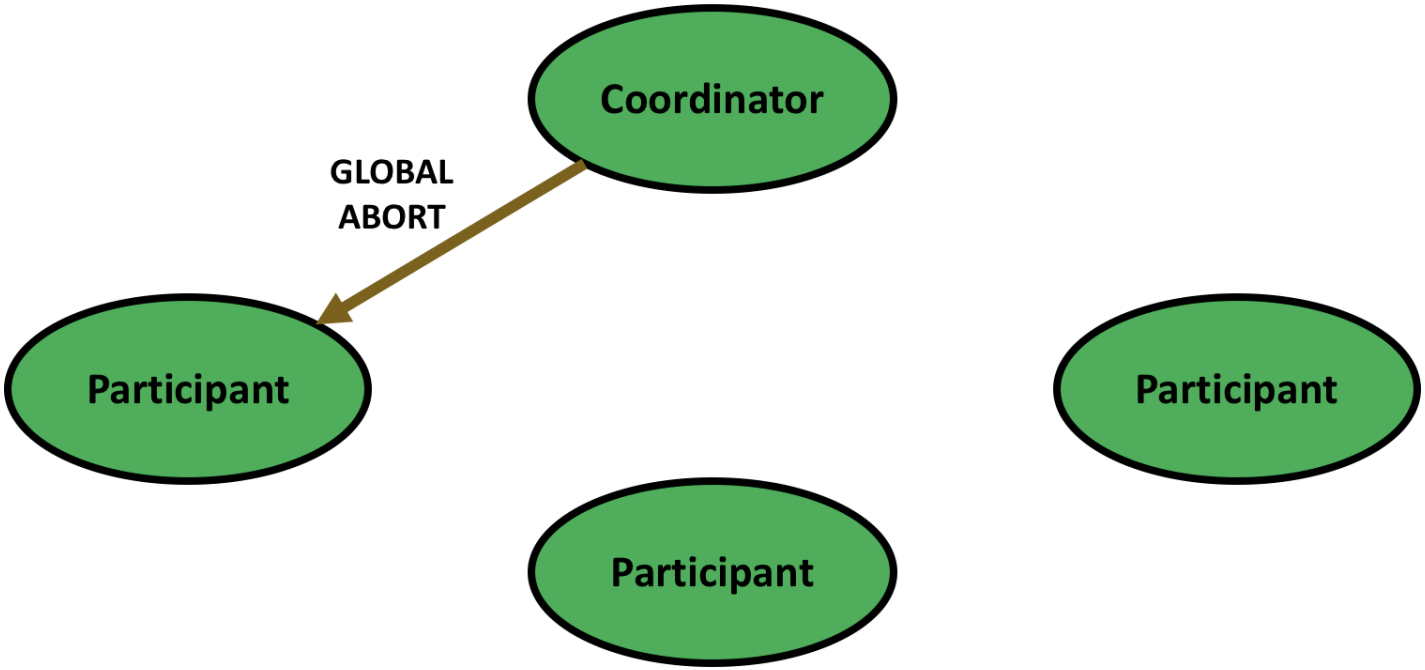
Easy Commit Works



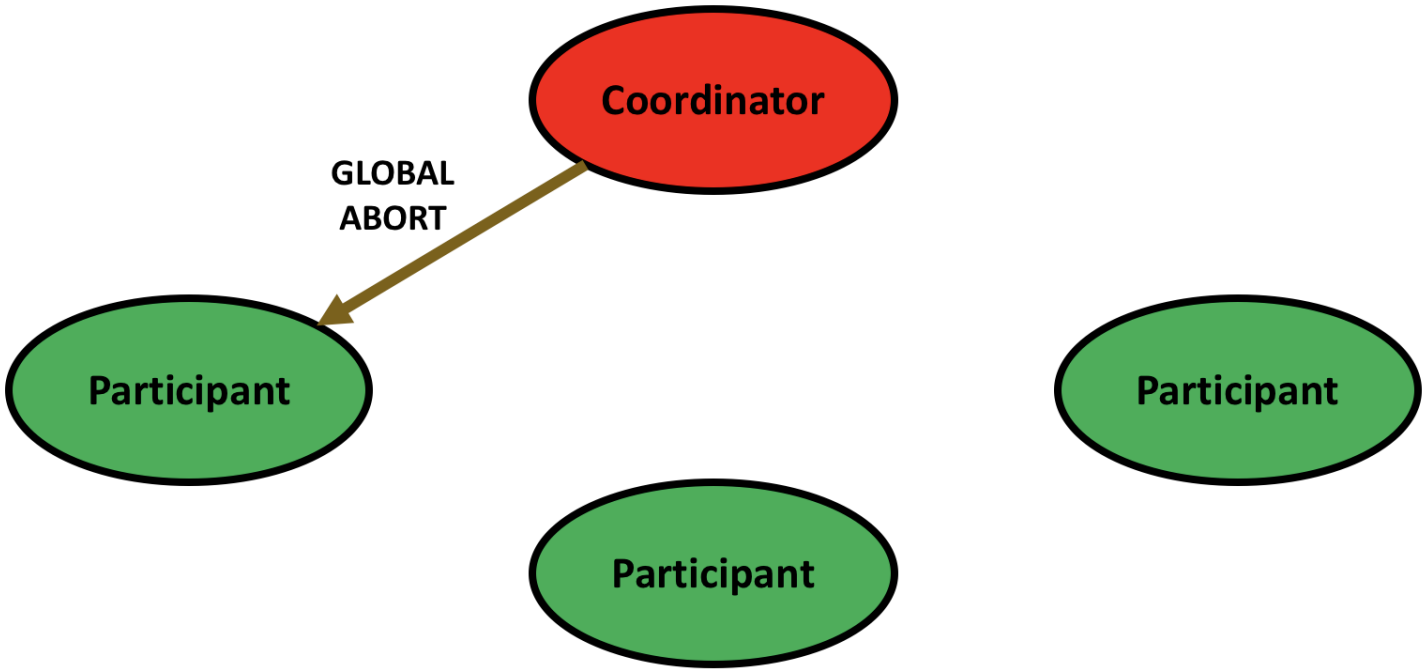
Easy Commit Works



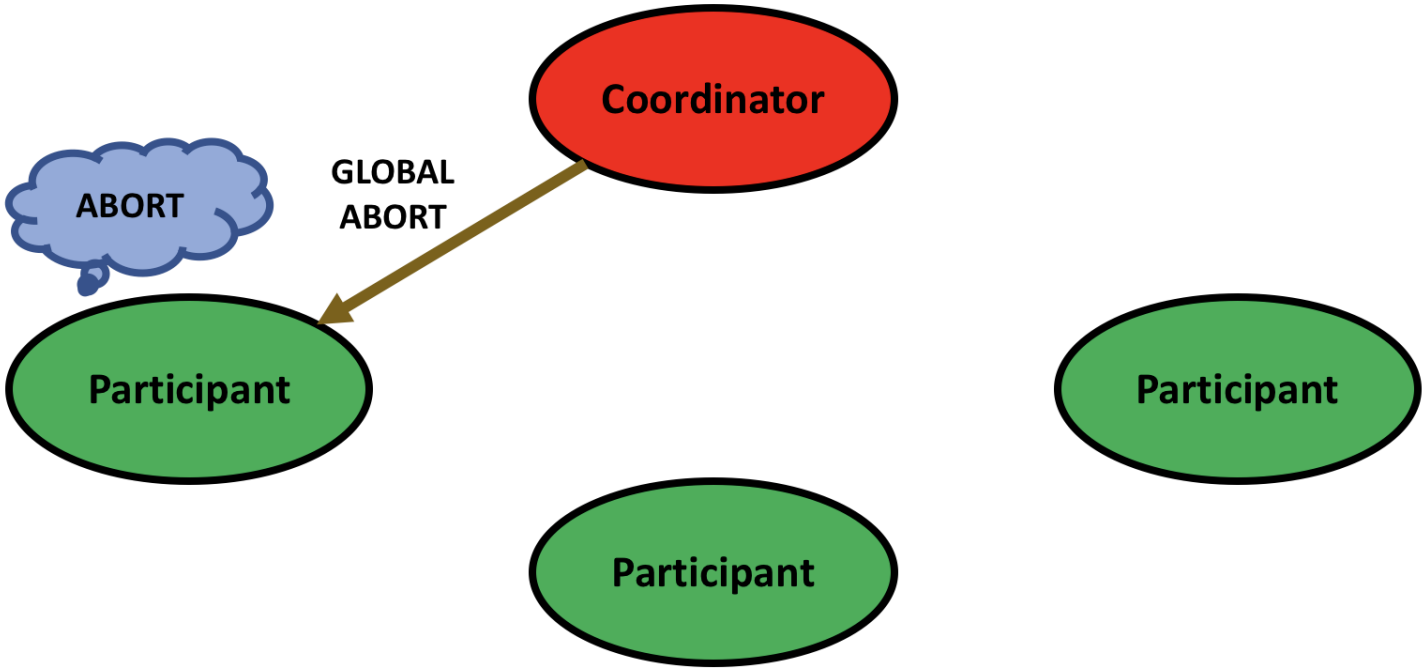
Easy Commit Works



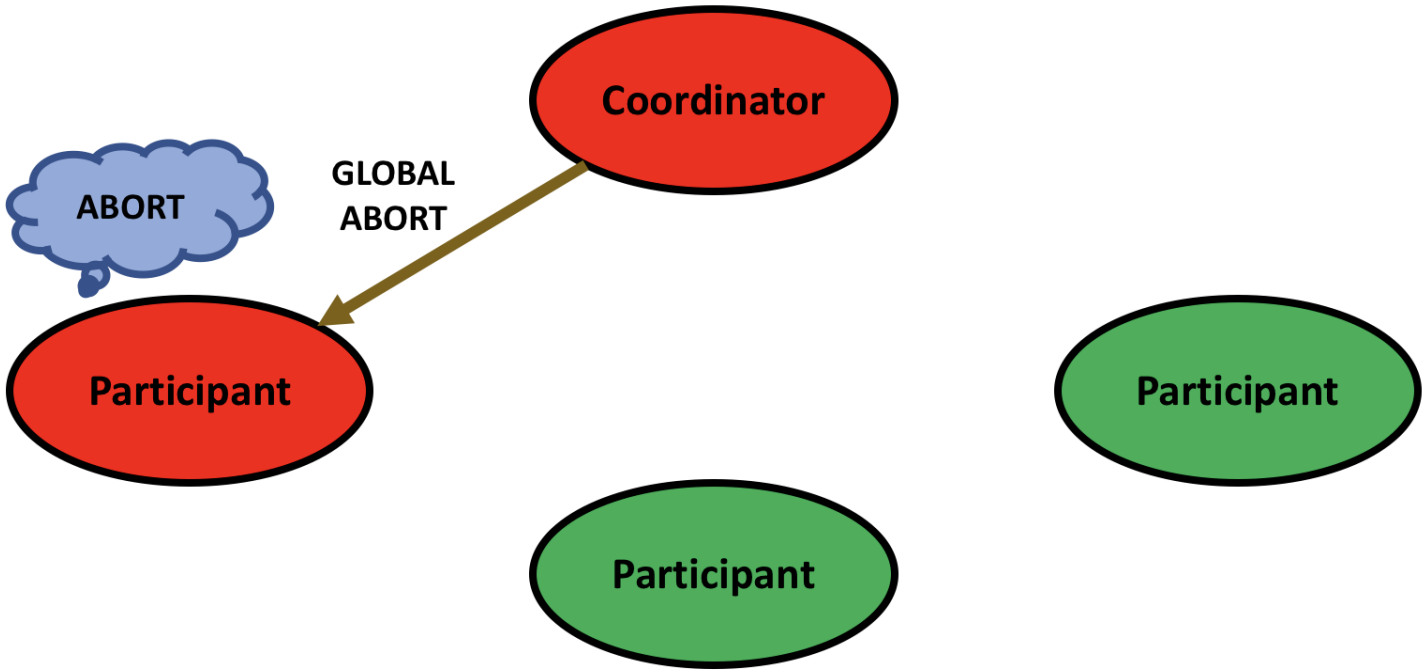
Easy Commit Works



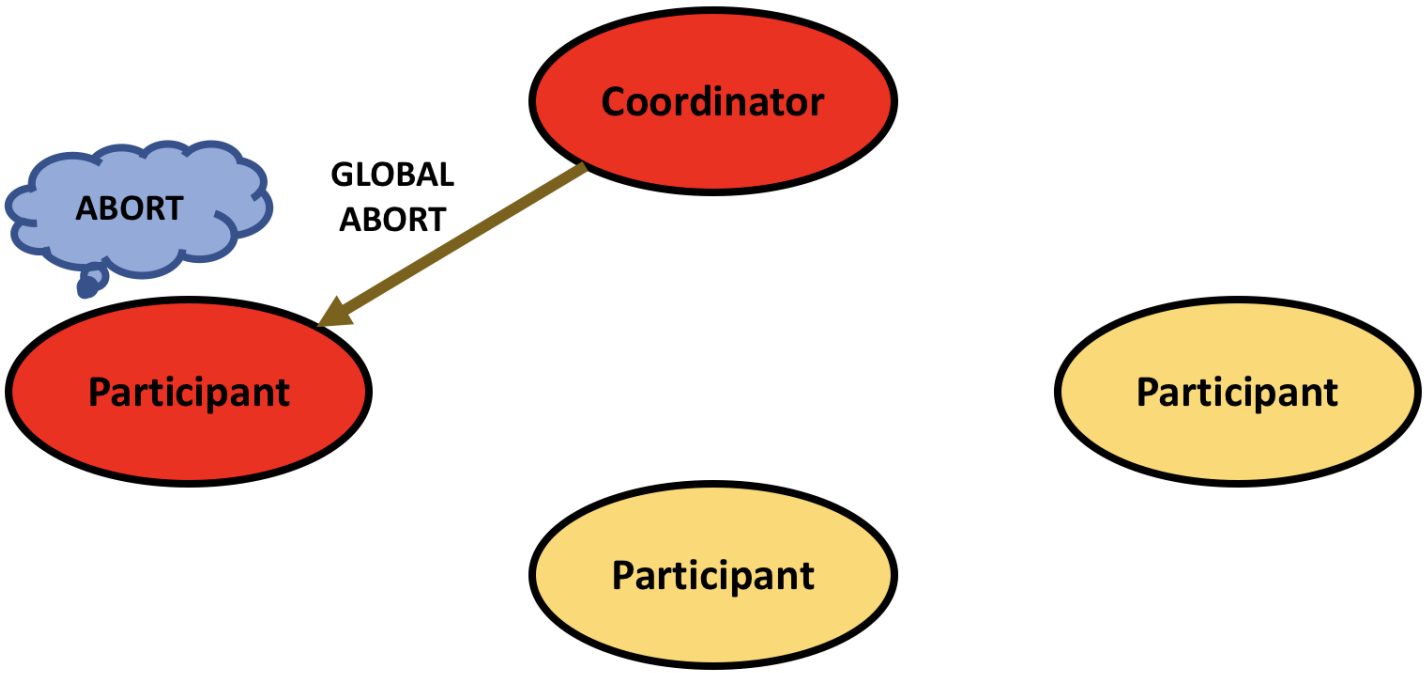
Easy Commit Works



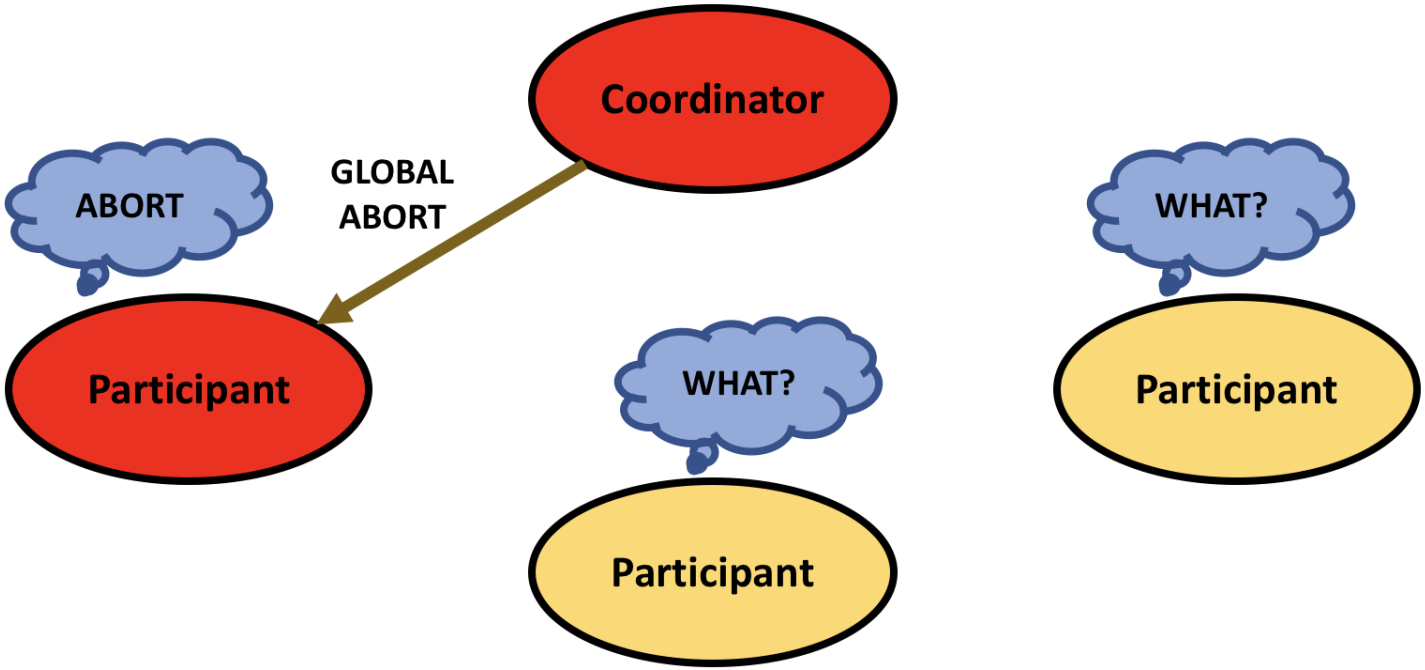
Easy Commit Works



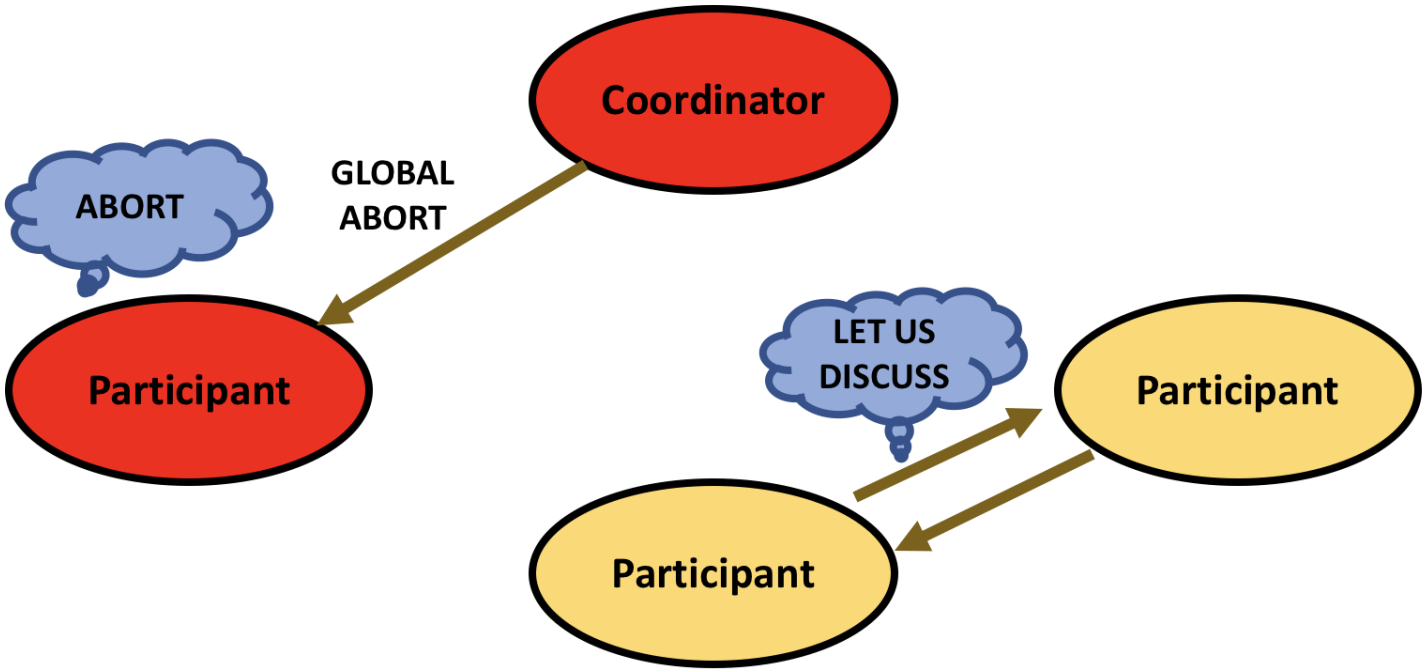
Easy Commit Works



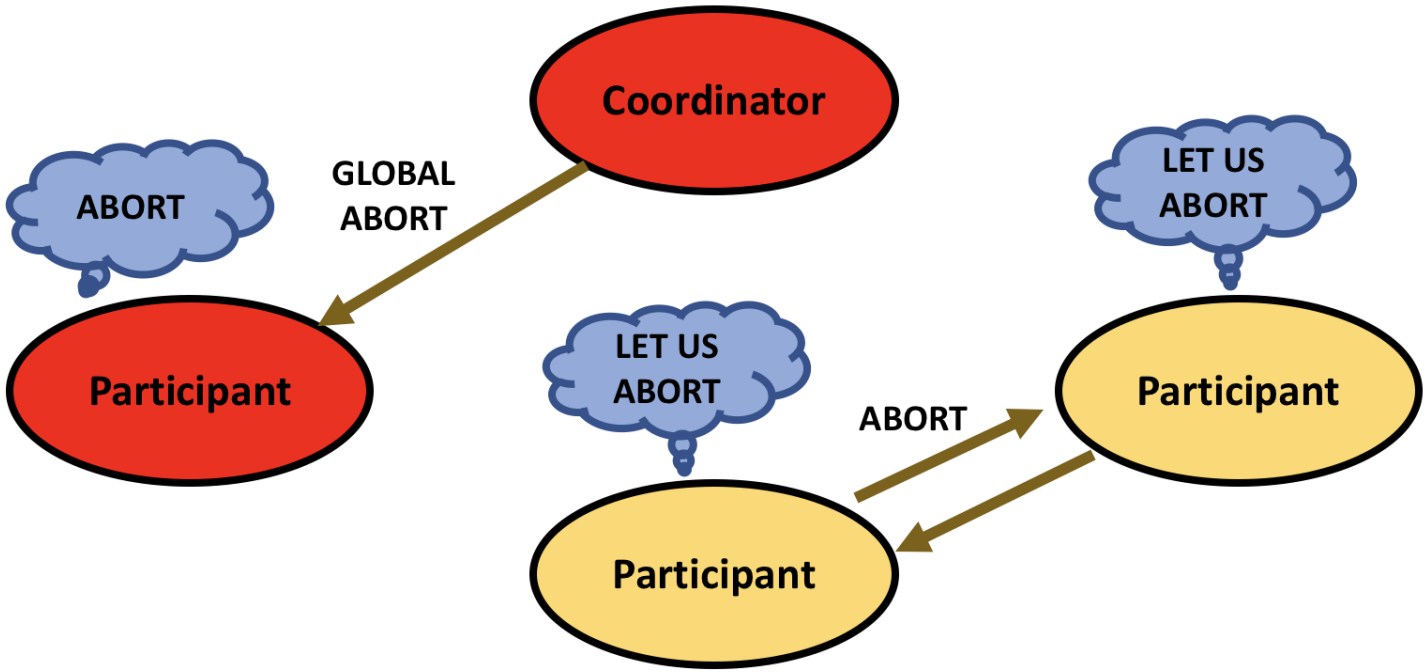
Easy Commit Works



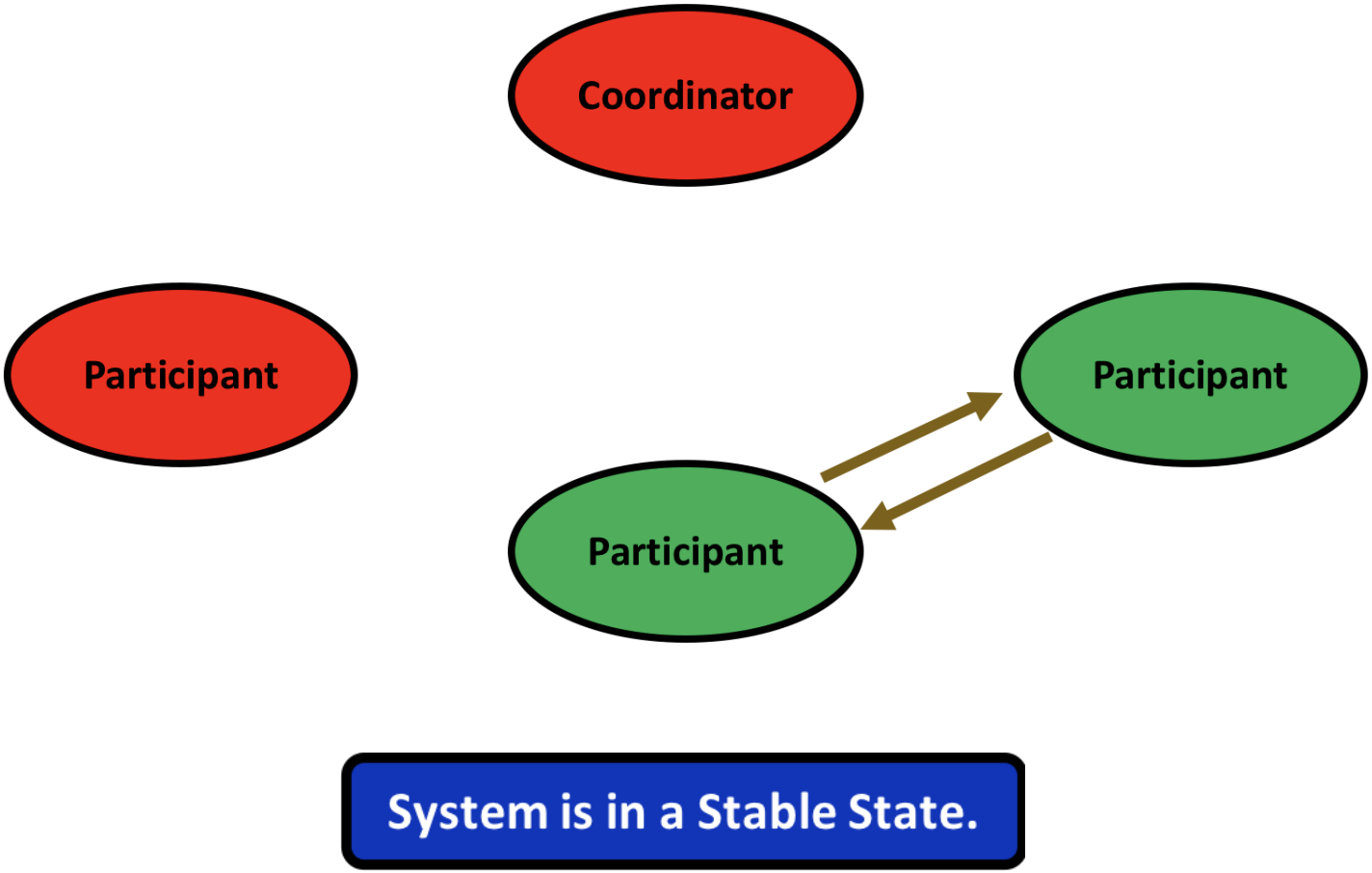
Easy Commit Works



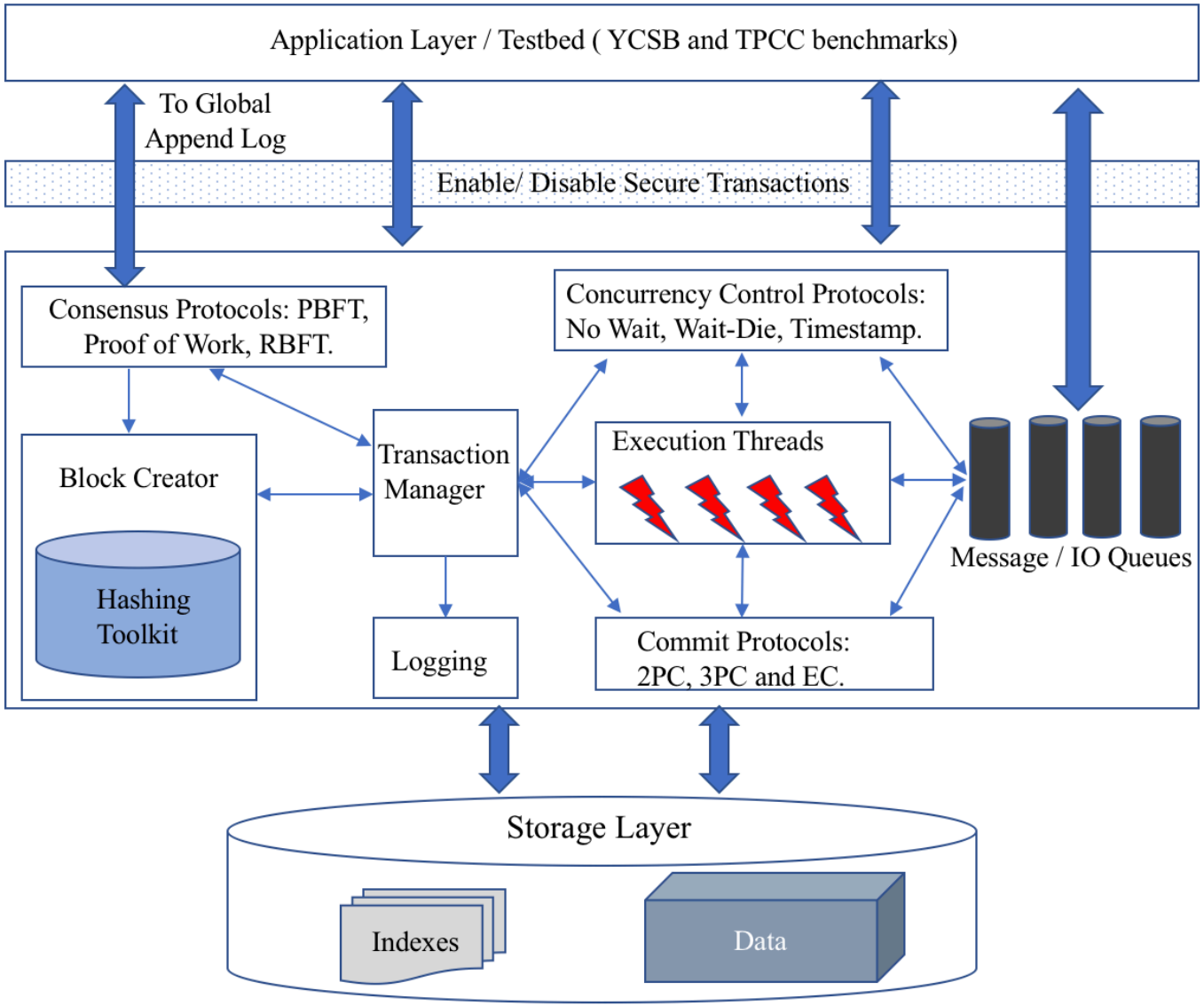
Easy Commit Works



Easy Commit Works



Implementation: ExpoDB

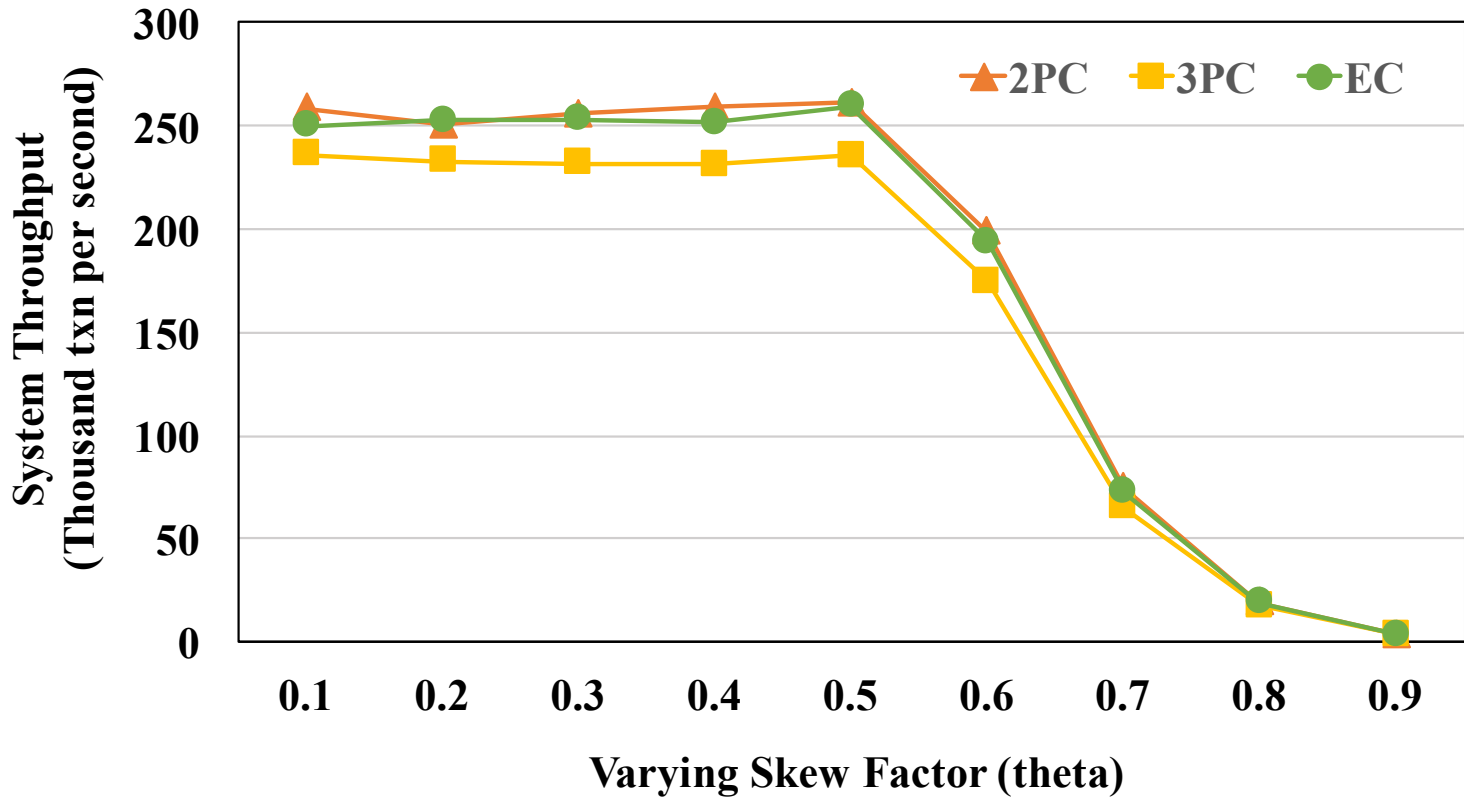


Evaluation

- 64 `Standard_D8S_V3` Azure instances, deployed in US East.
  - Each machine has 8 cores and 32GB memory.
- 4 Worker threads attached to a dedicated core.
- Load of 10000 open client connections per node.
- First 60s warmup and next 60s execution.
- Results averaged over three runs.
- NO-WAIT concurrency control algorithm used.
- Two benchmark suites: **YCSB** and **TPC-C**.

UC DAVIS UNIVERSITY OF CALIFORNIA

Varying Skew Factor – YCSB Zipfian Theta

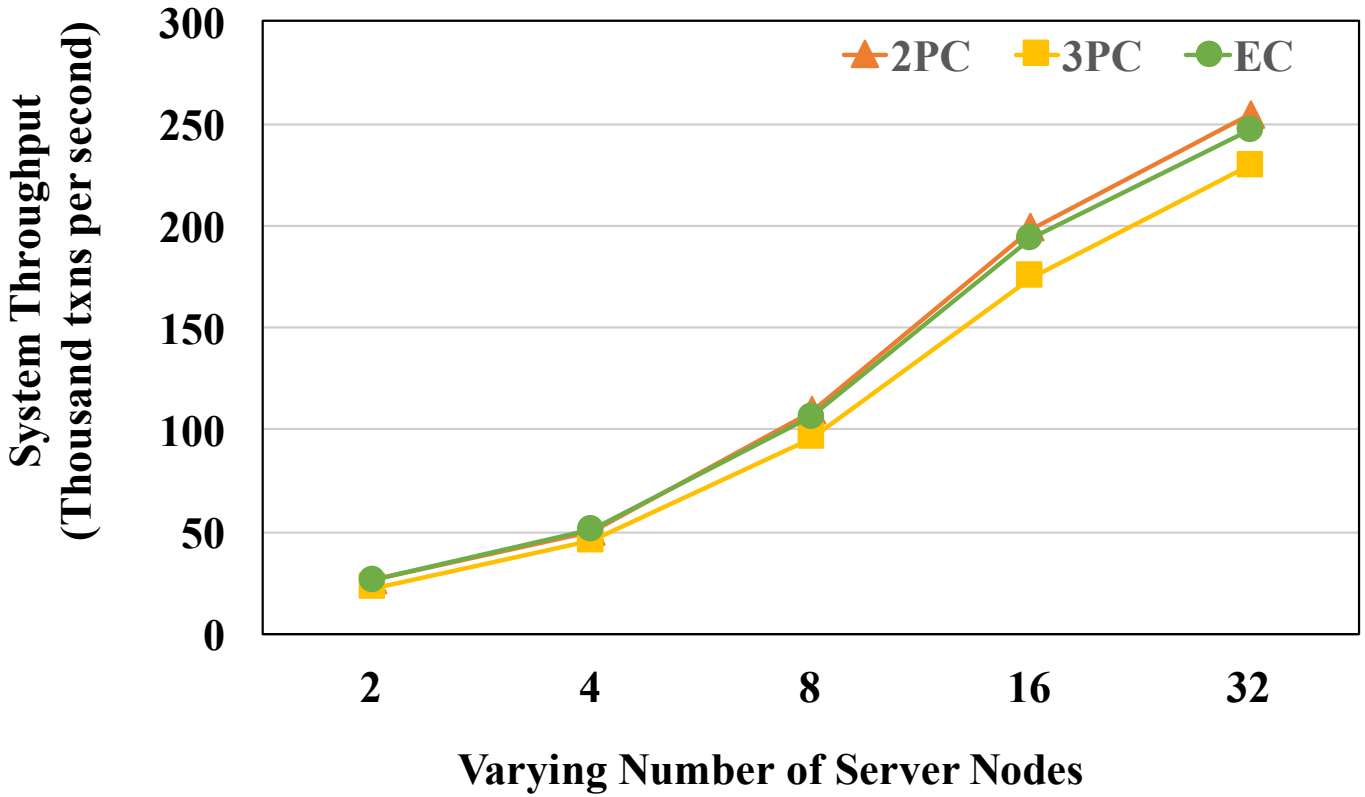


Number of server nodes = 16 and partitions per transaction = 2.

On varying the YCSB skew factor, EasyCommit throughput is equivalent to 2PC.

UCDAVIS UNIVERSITY OF CALIFORNIA

Varying Server Nodes

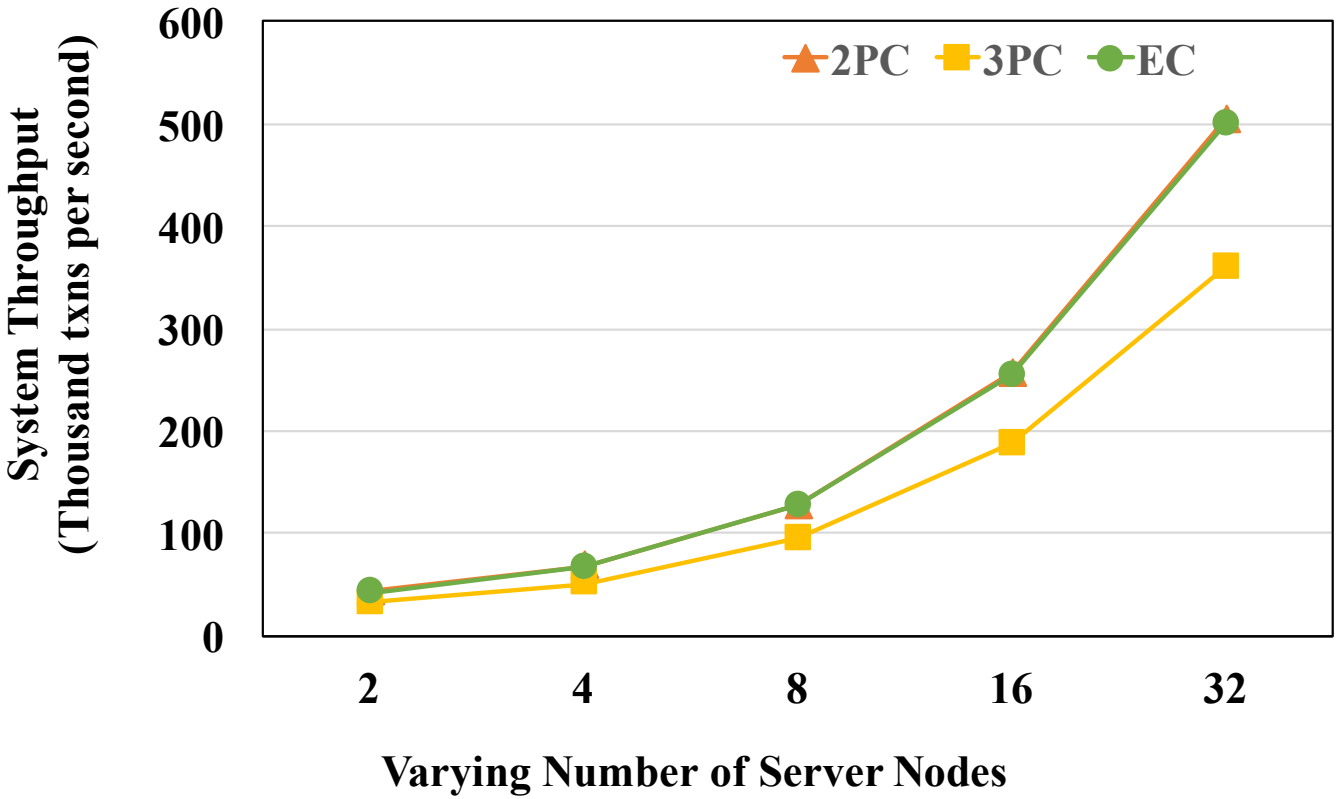


Partitions per transaction = 2 and Skew factor = 0.6.

EasyCommit is scalable as 2PC, on increasing the number of server nodes.

UC DAVIS UNIVERSITY OF CALIFORNIA

TPC-C Payment Transactions

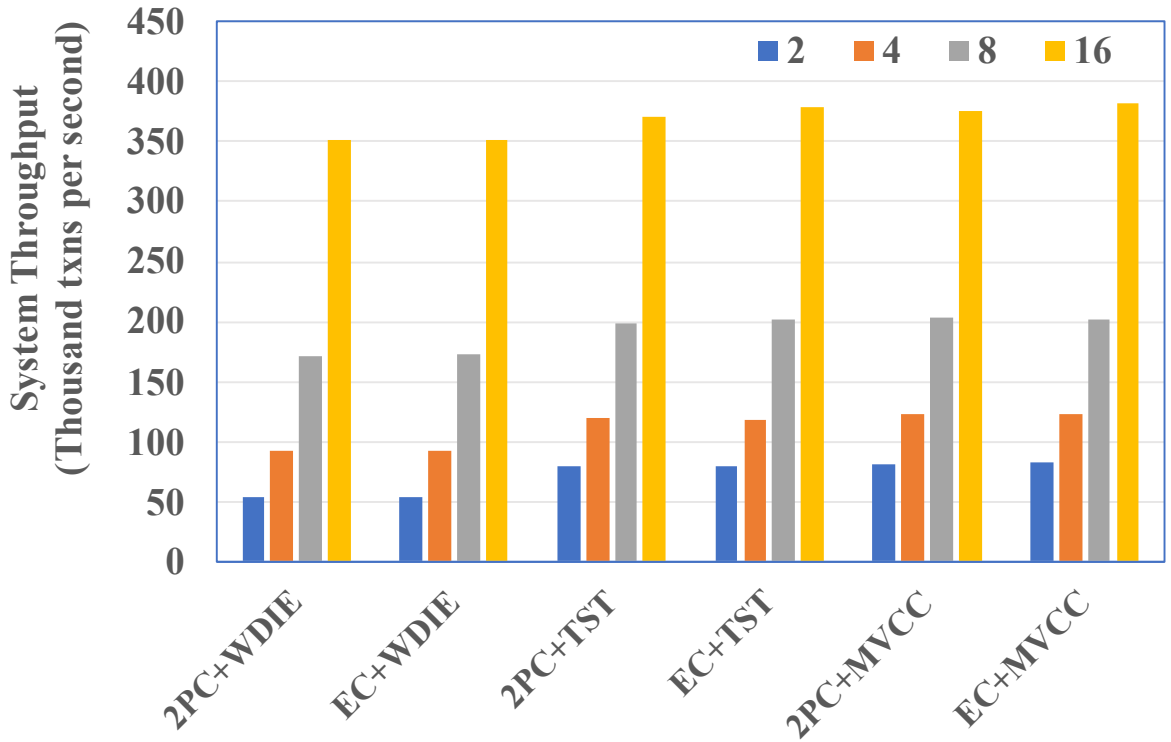


Number of warehouses per server = 128.

EasyCommit is scalable as 2PC, on increasing the number of server nodes.

UC DAVIS UNIVERSITY OF CALIFORNIA

Varying Concurrency Control Algorithms



TPC-C benchmarking for 16 **servers,** 128 **warehouses per server and** 3 **CC algorithms: WDIE (**`WAIT-DIE`**) and TST (**`TIMESTAMP`**).**

EasyCommit design is orthogonal to underlying Concurrency Control.

UC DAVIS UNIVERSITY OF CALIFORNIA

Conclusions

- We present novel commit protocol – **Easy Commit**.
- Leverages best of twin worlds (2PC and 3PC).
- Two key observations:
  - First transmit and then commit.
  - Message Redundancy.
- Easy Commit guarantees both safety and liveness.

Conclusions

- We present novel commit protocol – **Easy Commit**.
- Leverages best of twin worlds (2PC and 3PC).
- Two key observations:
  - First transmit and then commit.
  - Message Redundancy.
- Easy Commit guarantees both safety and liveness.

UC DAVIS UNIVERSITY OF CALIFORNIA