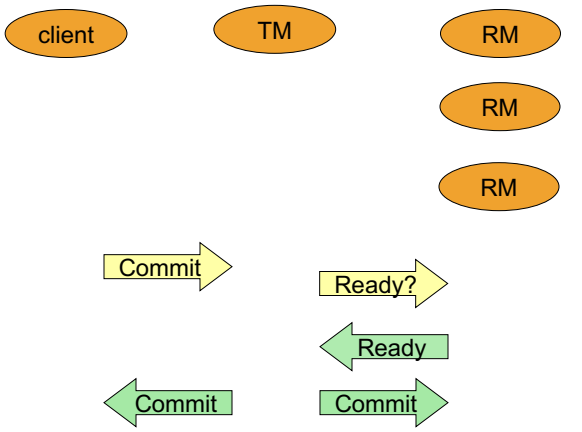
PAXOS AND ATOMIC COMMIT

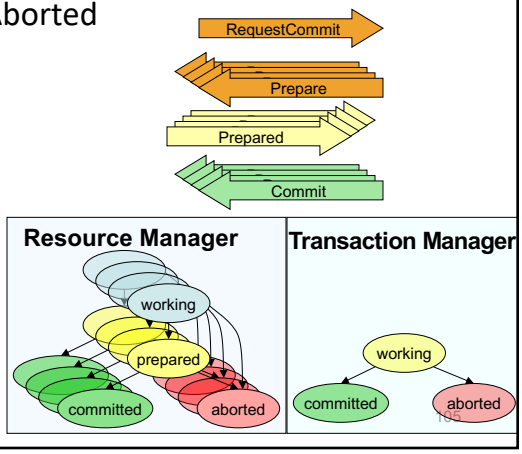
Database Commit



TM: Transaction Manager
RM: Resource Manager

Two Phase Commit

- *N* Resource Managers (RMs)
- Want all RMs to commit or all abort.
- Coordinated by Transaction Manager (TM)
  TM sends Prepare, Commit-Abort
- RM responds Prepared, Aborted
- *3N+1* messages
- *N+1* stable writes
- Delay
  - 4 message
  - 2 stable write
- Blocking:
  if TM fails,
  Commit-Abort stalls



105

The Problem With 2PC

- Atomicity – all or nothing
- Consistency – does right thing
- Isolation – no concurrency anomalies
- Durability / Reliability – state survives failures
- Availability: always up

Blocks if TM fails

106

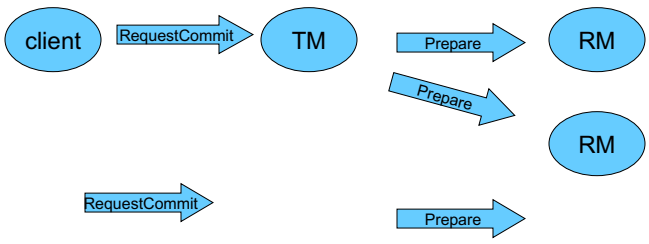
Problem Statement

- ACID Transactions make error handling easy.
- One fault can make 2-Phase Commit block.
- Goal: ACID and Available.
  Non-blocking despite *F* faults.

107

107

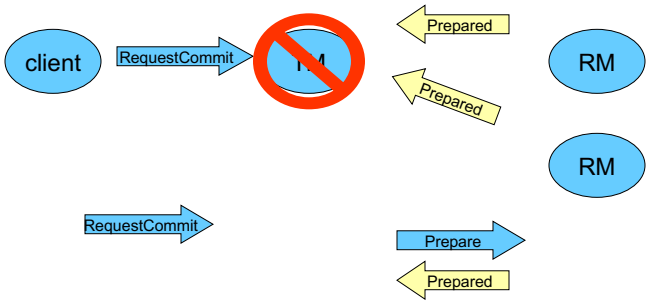
Fault-Tolerant Two Phase Commit



108

108

Fault-Tolerant Two Phase Commit

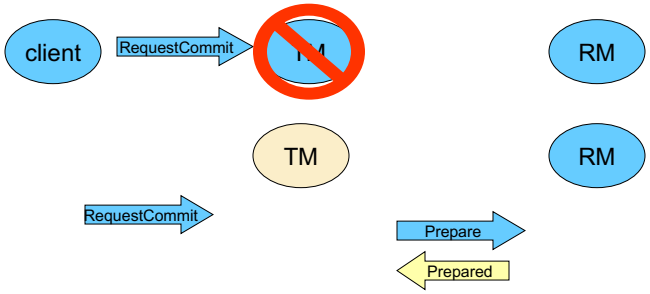


If the 2PC Transaction Manager (TM) Fails, transaction blocks.

109

109

Fault-Tolerant Two Phase Commit



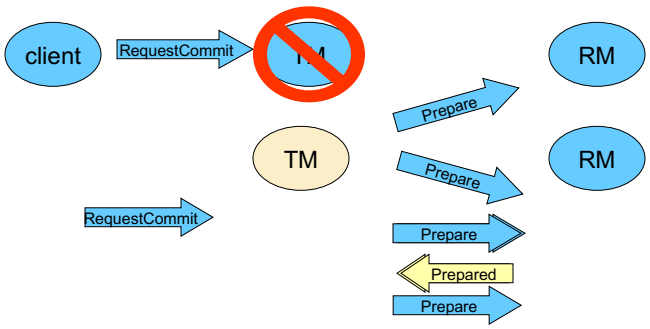
If the 2PC Transaction Manager (TM) Fails, transaction blocks.

Solution: Add a "spare" transaction manager
(non blocking commit, 3 phase commit)

110

110

Fault-Tolerant Two Phase Commit



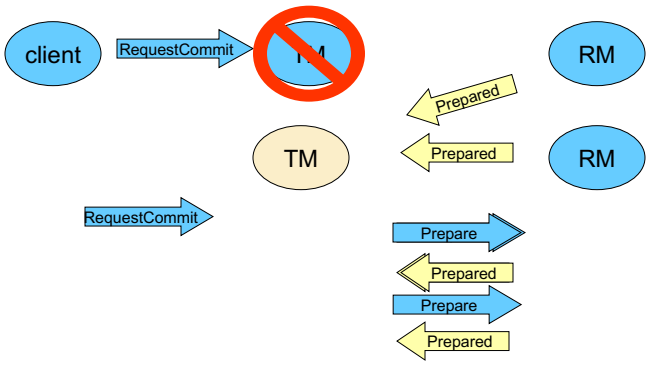
If the 2PC Transaction Manager (TM) Fails, transaction blocks.

Solution: Add a "spare" transaction manager
(non blocking commit, 3 phase commit)

111

111

Fault-Tolerant Two Phase Commit



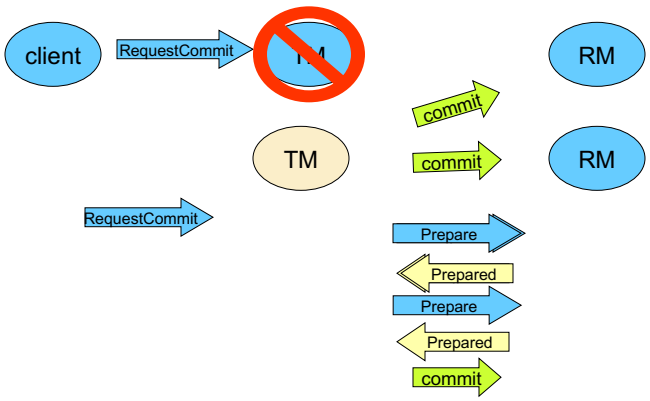
If the 2PC Transaction Manager (TM) Fails, transaction blocks.

Solution: Add a "spare" transaction manager
(non blocking commit, 3 phase commit)

112

112

Fault-Tolerant Two Phase Commit



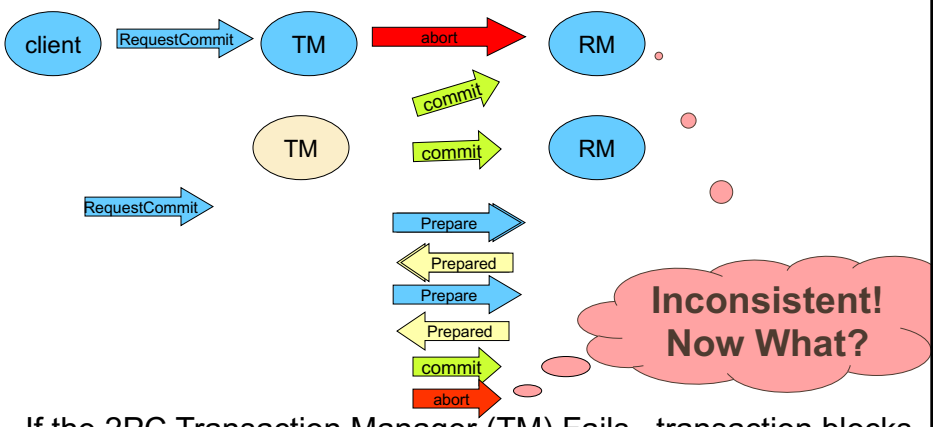
If the 2PC Transaction Manager (TM) Fails, transaction blocks.

Solution: Add a "spare" transaction manager
(non blocking commit, 3 phase commit)

113

113

Fault-Tolerant Two Phase Commit



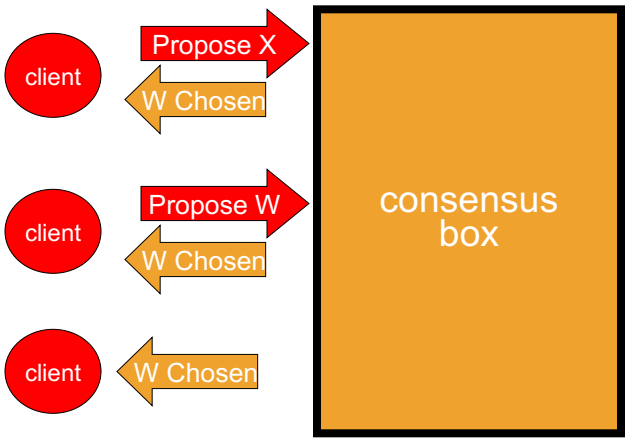
If the 2PC Transaction Manager (TM) Fails, transaction blocks.

Solution: Add a "spare" transaction manager
(non blocking commit, 3 phase commit)

114

114

Paxos Consensus Box

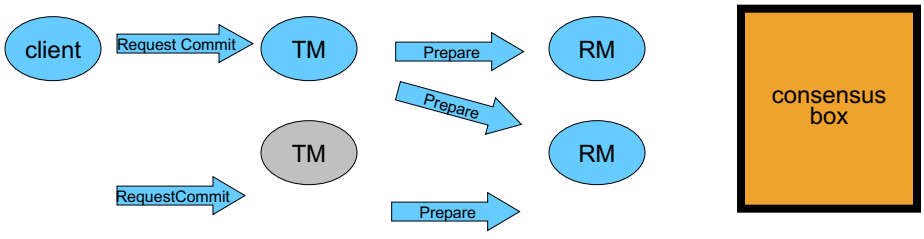


- Collects proposed values
- Picks one proposed value
- Remembers it forever

115

115

Consensus for Commit
The Obvious Approach

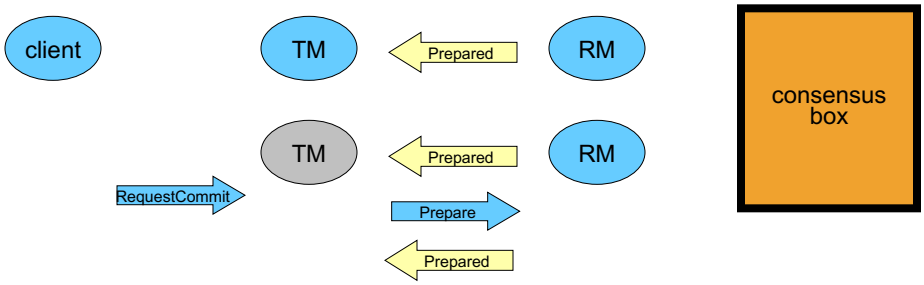


- Get consensus on TM's decision.
- TM just learns consensus value.
- TM is "stateless"

116

116

Consensus for Commit
The Obvious Approach

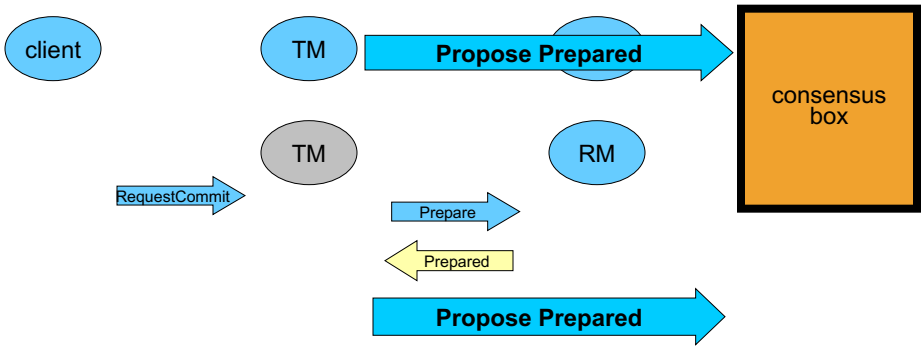


- Get consensus on TM's decision.
- TM just learns consensus value.
- TM is "stateless"

117

117

Consensus for Commit
The Obvious Approach

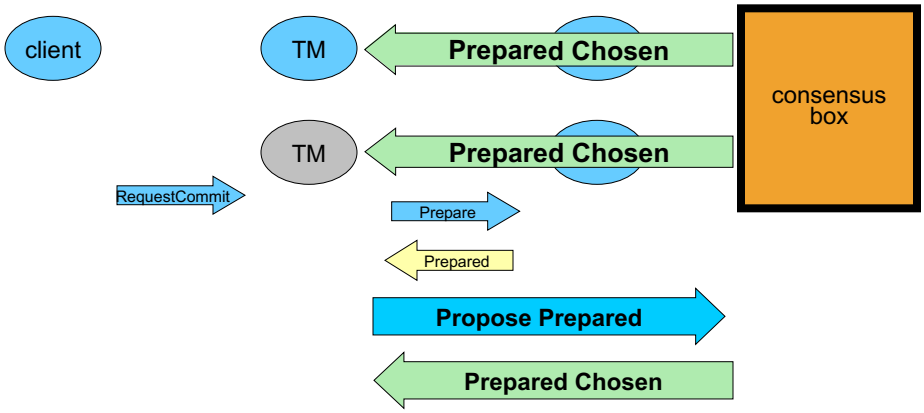


- Get consensus on TM's decision.
- TM just learns consensus value.
- TM is "stateless"

118

118

Consensus for Commit
The Obvious Approach

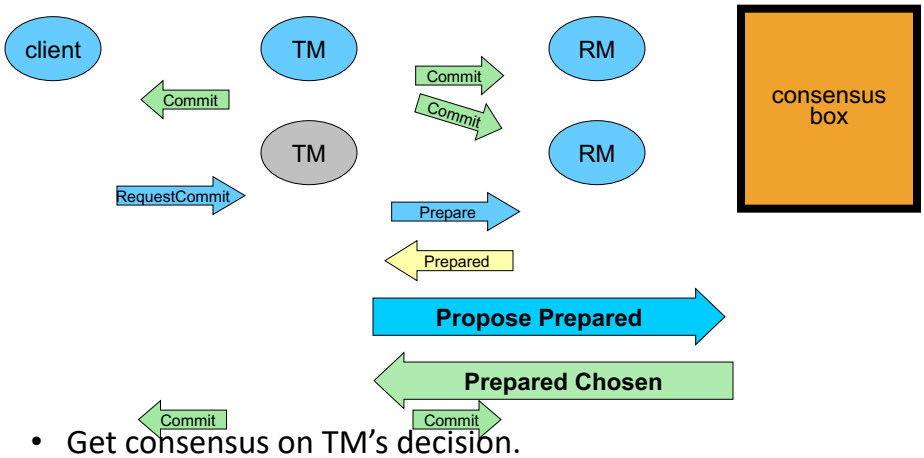


- Get consensus on TM's decision.
- TM just learns consensus value.
- TM is "stateless"

119

119

Consensus for Commit
The Obvious Approach

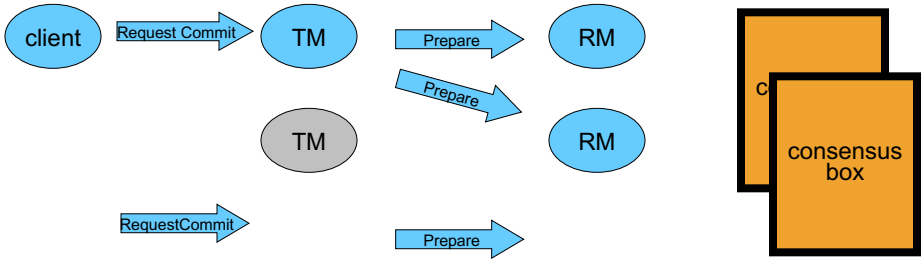


- Get consensus on TM's decision.
- TM just learns consensus value.
- TM is "stateless"

120

120

Consensus for Commit
The Paxos Commit Approach

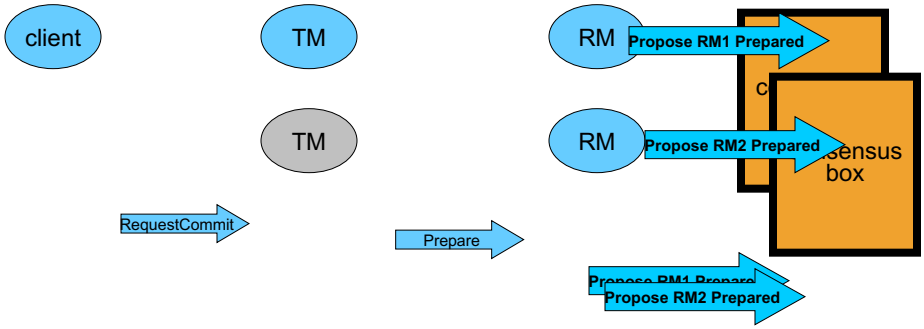


- Get consensus on each RM's choice.
- TM just combines consensus values.
- TM is "stateless"

121

121

Consensus for Commit
The Paxos Commit Approach

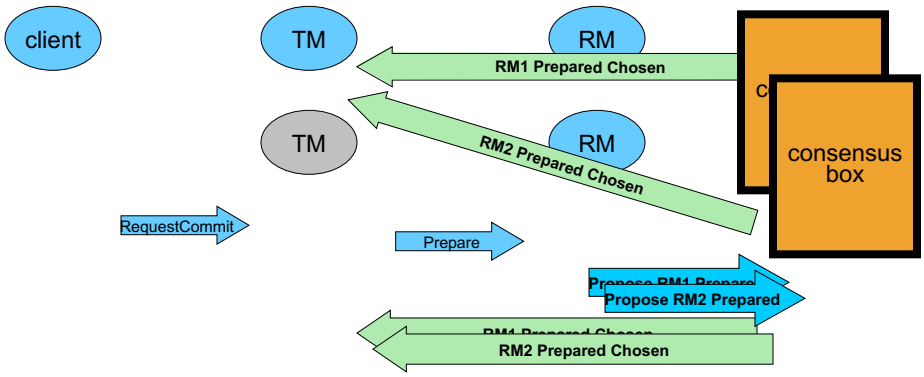


- Get consensus on each RM's choice.
- TM just combines consensus values.
- TM is "stateless"

122

122

Consensus for Commit
The Paxos Commit Approach

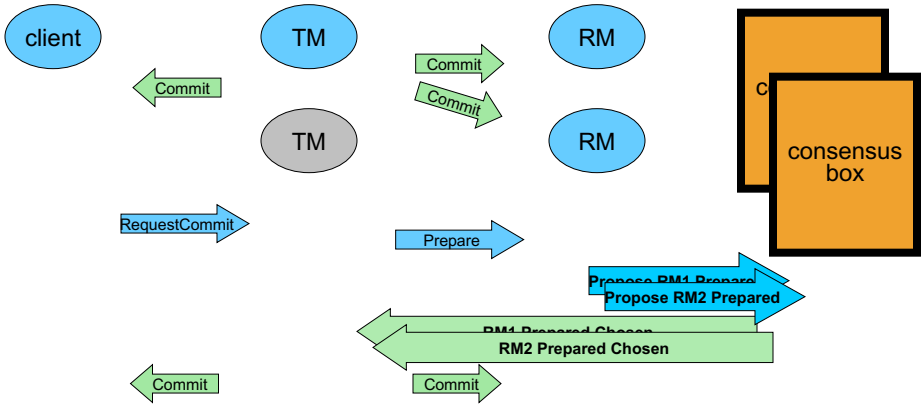


- Get consensus on each RM's choice.
- TM just combines consensus values.
- TM is "stateless"

123

123

Consensus for Commit
The Paxos Commit Approach

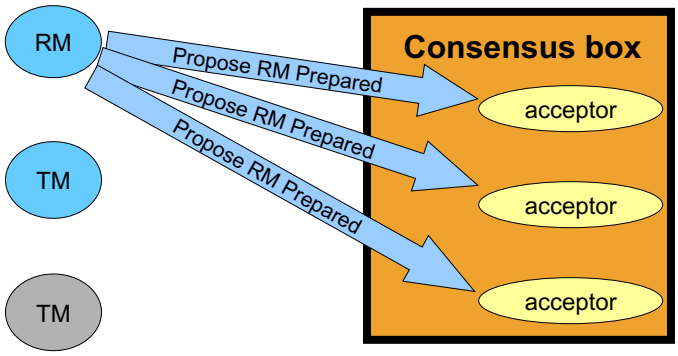


- Get consensus on each RM's choice.
- TM just combines consensus values.
- TM is "stateless"

124

124

Consensus in Action

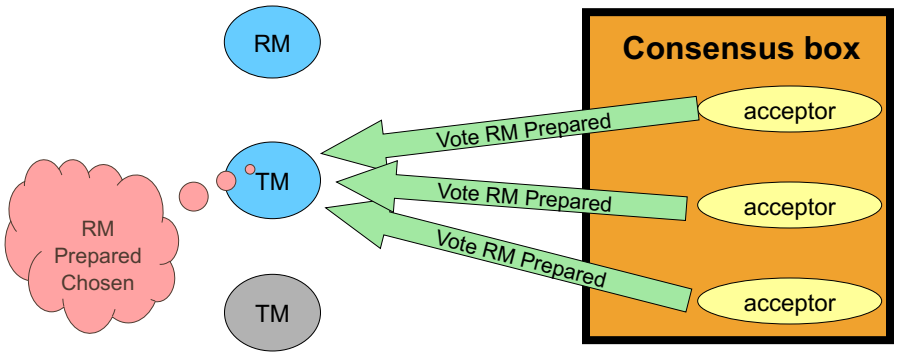


- The normal (failure-free) case

125

125

Consensus in Action

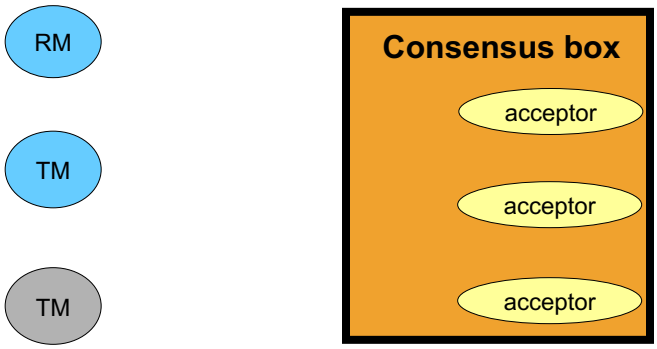


- The normal (failure-free) case
- Two message delays
- Can optimize

126

126

Consensus in Action



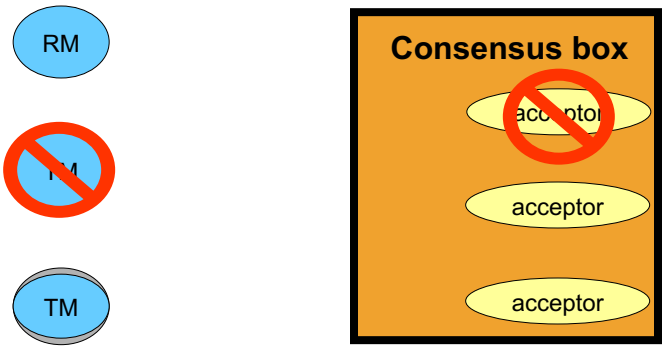
TM can always learn what was chosen,
or get *Aborted* chosen if nothing chosen yet;

.

127

127

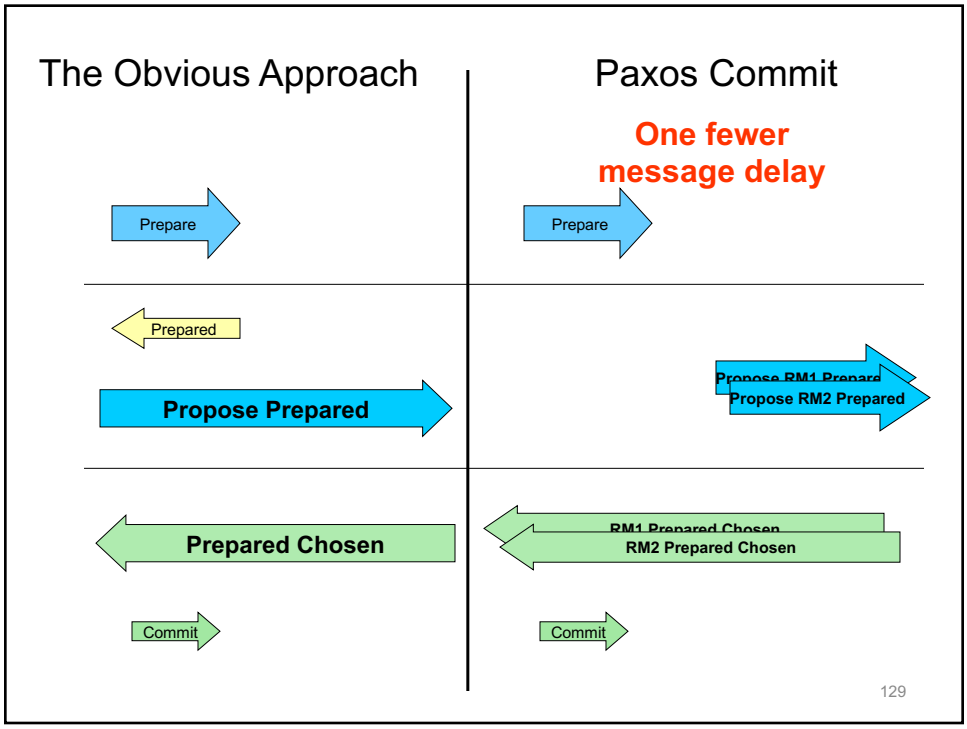
Consensus in Action



TM can always learn what was chosen,
or get *Aborted* chosen if nothing chosen yet;
if majority of acceptors working .

128

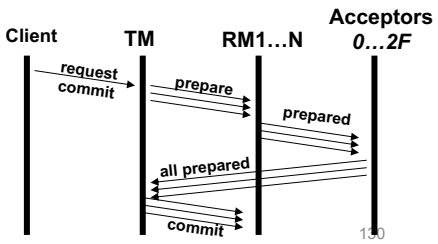
128



129

Paxos Commit

- *N* RMs
- *2F+1* acceptors (~*2F+1* TMs)
- If *F+1* acceptors see all RMs prepared, then transaction committed.
- *(N+1)(F+3) - 2* messages
  - 1 request to commit
  - N - 1 prepare
  - N (F + 1) prepared
  - F + 1 all prepared
  - N commit
- 5 message delays
  2 stable write delays.



130

Two-Phase Commit	Paxos Commit
	tolerates *F* faults
• 3*N*+1 messages	• 3*N*+ 2*F*(*N*+1) +1 messages
• *N*+1 stable writes	• *N*+2*F*+1 stable writes
• 4 message delays	• 5 message delays
• 2 stable-write delays	• 2 stable-write delays

Same algorithm when *F*=0 and TM = Acceptor

131

131