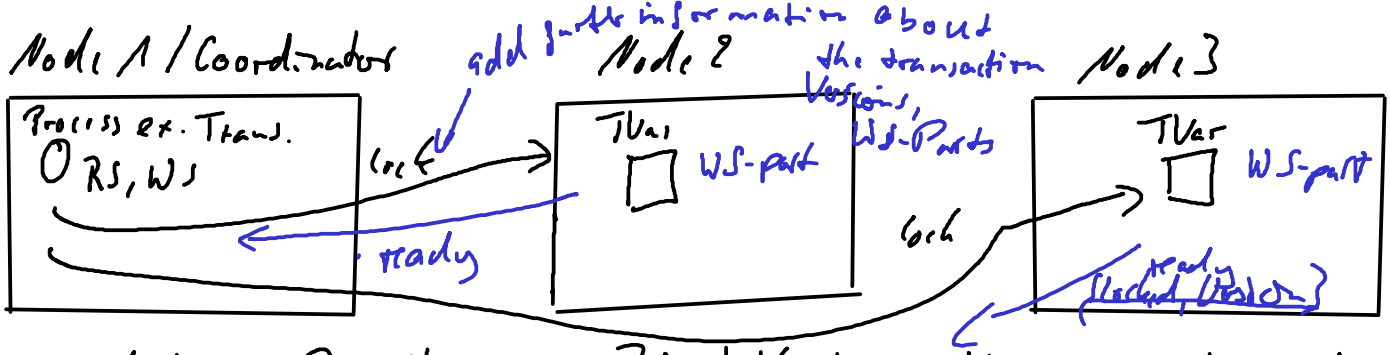


- Validation: Do TVars in RS still have the previously read value or Version?
- Commit: Realize the WS in every TVar
- Lock TVars

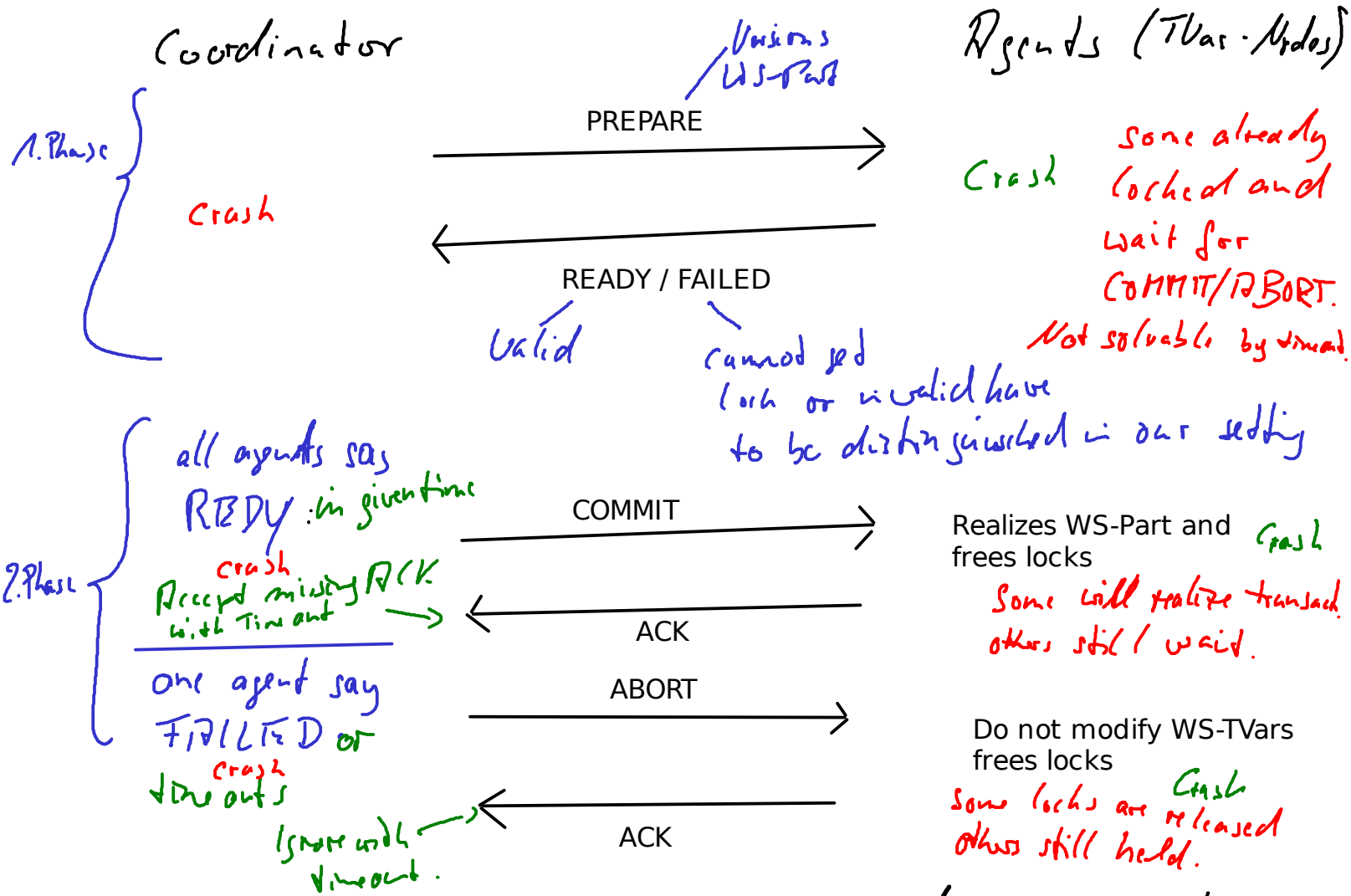
Locking in distributed has also to consider deadlock

Avoiding: Use global order or Layback all ressources if next lock is not available.

defined by coordinator
Consider IP-address + TVar-Id
Locking is performed sequentially:
Lock TVar1 -> Acknowledge -> Lock TVar2
-> Acknowledge -> ...

parallel lock message to all nodes, two possible answers: locked or failed. Coordinator collects all answers:
all locked -> continue
one failed -> inform all about that -> free locks again -> try lock again

Locking and distributing information is called PREPARE in the two-phase-commit-protocoll.



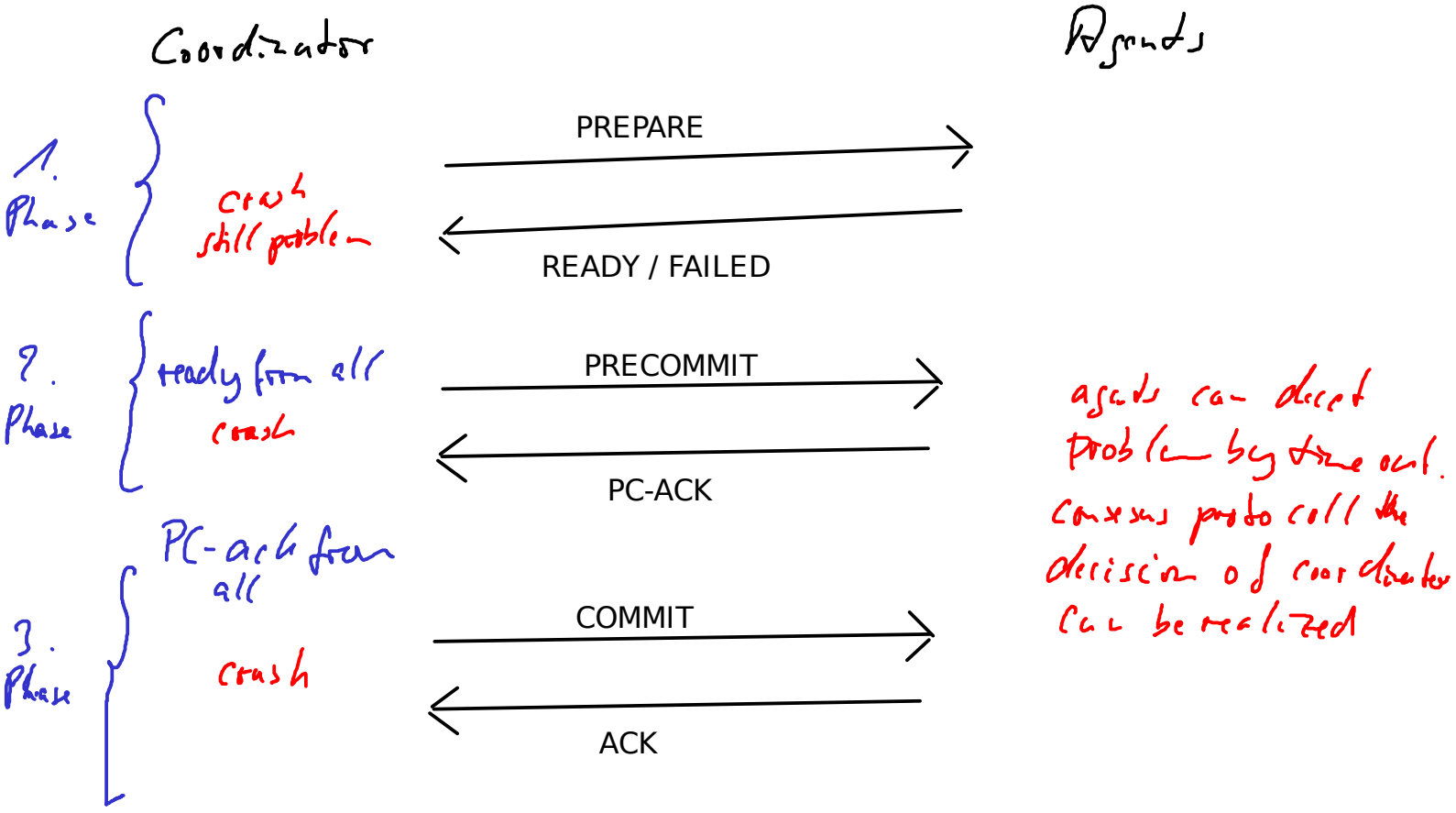
Problems: in distributed systems nodes can crash or connection can get lost.

1. One agent crashes: the coordinator can use timeouts to finish the protocoll without producing an inconsistent global state.

2. the coordinator crashes: blocking of the system is possible an also part of transaction can be committed, but inconsistent view is avoided (by blocking).

Solution: find another coordinator if one agent detects the problematic blocking by a time out. ⇒ Leader election DS.

Another solution: three-phase-commit-protocol



Committing in a distributed setting is closely related to the problem of transaction comm distributed database systems. This is independent of using optimistic oder pessimistic transaction implementations.

The approach of direct notification of making other transaction invalid, cannot easily be transfered to a distributed setting, because of delays in network communication.