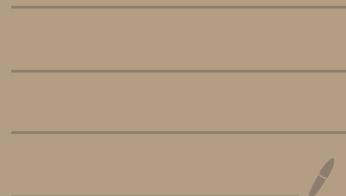


Chapter 9: Consistency and Consensus



- Consensus — getting all of the nodes to agree on something.
- Split brain — in single-leader replication, two nodes both believe that they are the leader
- eventual consistency — if step writing to the database and wait for some time, eventually all reads request will return the same value

← convergence

Linearizability

— make a system appear as if there were only one copy of data, and all operations on it are atomic

• Implementing Linearizable Systems

- Single-leader replication
 - potentially linearizable
- Consensus algorithms
 - linearizable
- Multi-leader replication
 - not linearizable

- Leaderless replication
 - probably not linearizable
- CAP theorem
 - ~~Trade-off~~ between linearizability and availability
 - if an application requires linearizability and some replicas are disconnected from the other replicas due to a network problem, they must either wait until the network problem is fixed or return an error.
 - if an application does not require linearizability, then the application can remain available even if the network problem occurs,

- Sequence Number Ordering
 - Use seq number or timestamps to order events.

- Lamport timestamps
 - a way of generating sequence numbers that is consistent with causality
 - provides total order, (counter, nodeID)

* Total Order Broadcast

- a protocol for exchanging messages between nodes.

features Predictable delivery
No messages are lost

- Totally ordered delivery
Messages are delivered to every node in the same order.

2PC (Two-Phase Commit)

- a coordinator (aka transaction manager)
prepares the transaction.

Phase 1

activities
nodes
whether
they can
commit

1. if all participants (database nodes) reply that they are ready to commit, the coordinator sends a commit request
2. if any of the participants say no, the coordinator sends an abort request to all nodes in phase 2,



Commit T Commit transaction

Cons

if a coordinator fails after receiving replies of a prepare request from participants, they have to wait for a coordinator to recover.
↳ in doubt, uncertain

Membership and Coordination Services

- linearizable atomic operations

- using an atomic compare-and-set

- deviation, you can implement a clock
✓ guarantee

linearizability

- total ordering of deviations

- using this to give each operation a monotonically increasing number

(transaction ID) and version number
to prevent conflicts,

- failure detection

- change notifications

- finding out any changes

(for example, another client joins the cluster)