The Raft Consensus Algorithm

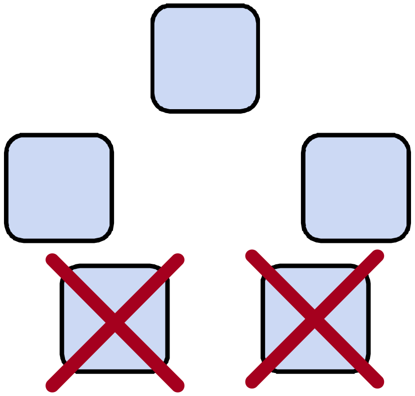
Diego Ongaro   John Ousterhout

Stanford University

http://raftconsensus.github.io

What is Consensus?

- **Agreement on shared state (single system image)**
- **Recovers from server failures autonomously**
  - Minority of servers fail: no problem
  - Majority fail: lose availability, retain consistency



Servers

- **Key to building consistent storage systems**

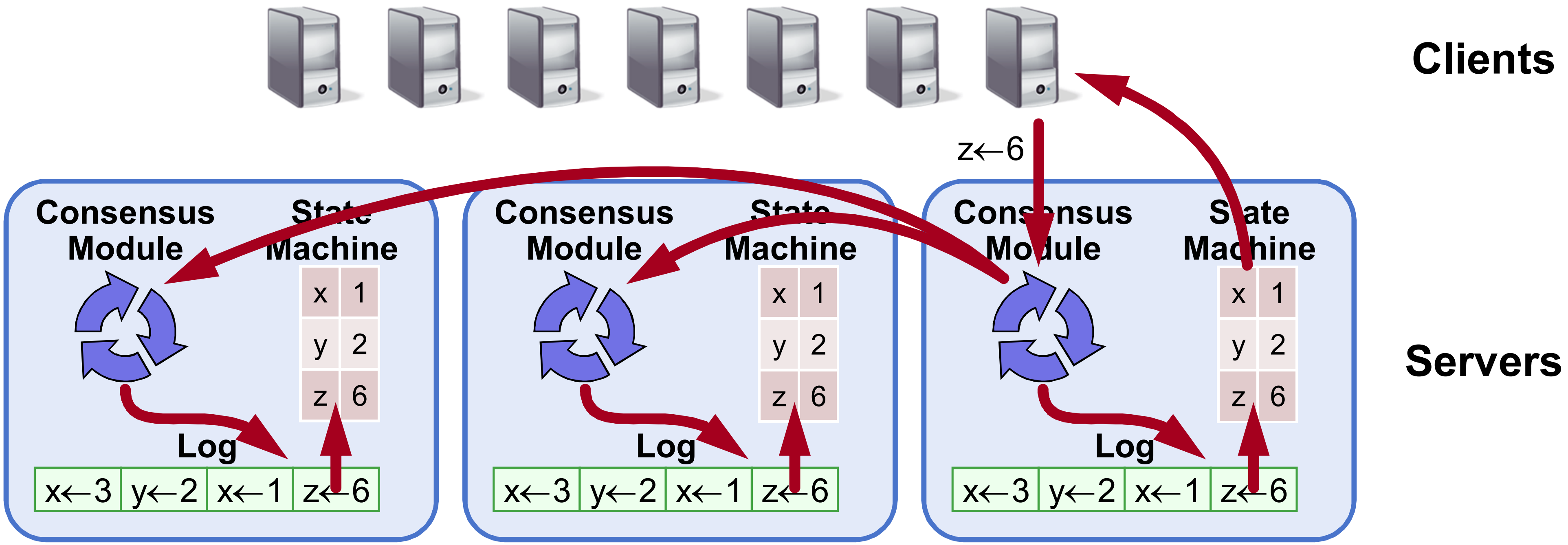
Inside a Consistent System

- **TODO: eliminate single point of failure**
- **An ad hoc algorithm**
  - "This case is rare and typically occurs as a result of a network partition with replication lag."

– OR –

- **A consensus algorithm (built-in or library)**
  - Paxos, Raft, …
- **A consensus service**
  - ZooKeeper, etcd, consul, …

Replicated State Machines



Clients

z←6

Servers

Consensus Module

State Machine

x	1
y	2
z	6

Log

x←3	y←2	x←1	z←6

Consensus Module

State Machine

x	1
y	2
z	6

Log

x←3	y←2	x←1	z←6

Consensus Module

State Machine

x	1
y	2
z	6

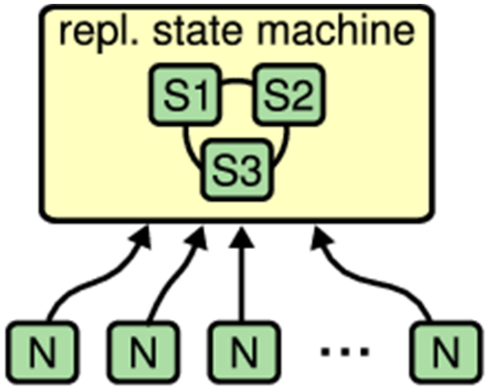
Log

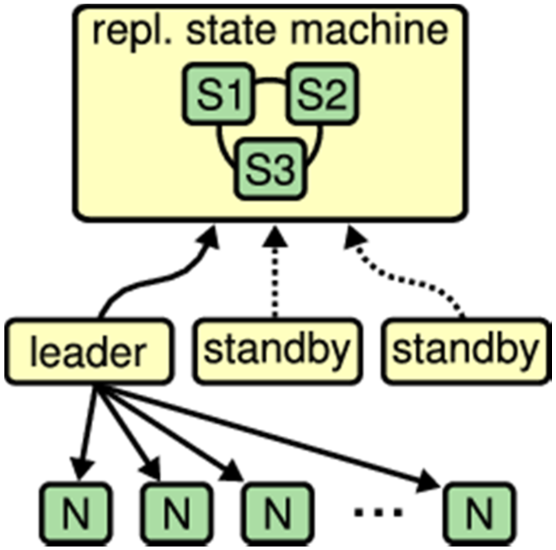
x←3	y←2	x←1	z←6

- **Replicated log ⇒ replicated state machine**
  - All servers execute same commands in same order
- **Consensus module ensures proper log replication**
- **System makes progress as long as any majority of servers are up**
- **Failure model: fail-stop (not Byzantine), delayed/lost messages**

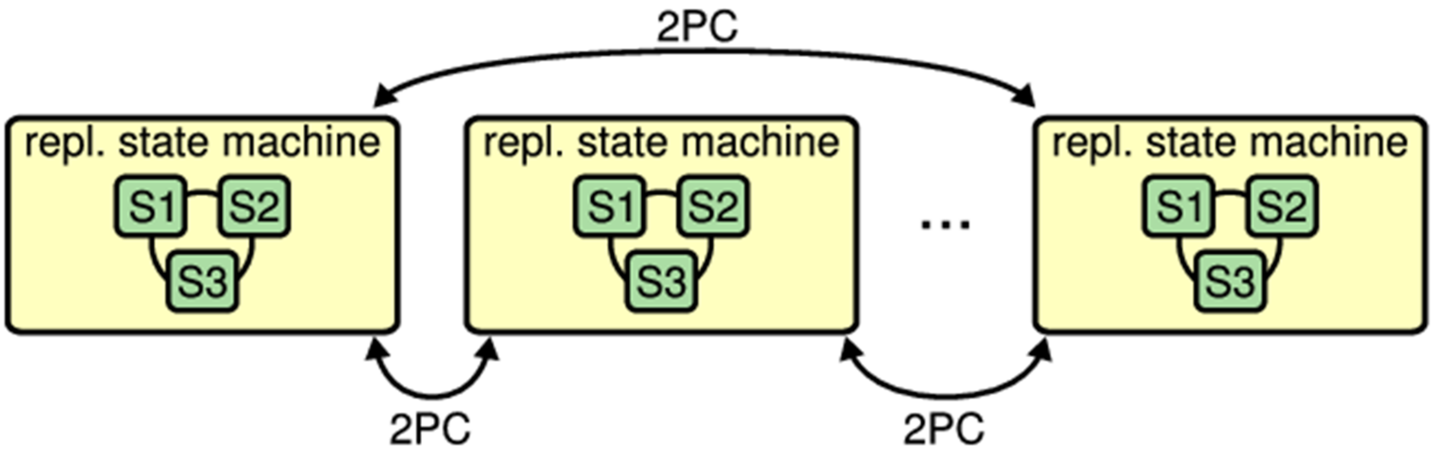
How Is Consensus Used?

- Top-level system configuration





- Replicate entire database state



Paxos Protocol

- **Leslie Lamport, 1989**
- **Nearly synonymous with consensus**

"The dirty little secret of the NSDI community is that at most five people really, truly understand every part of Paxos ;-)." – NSDI reviewer

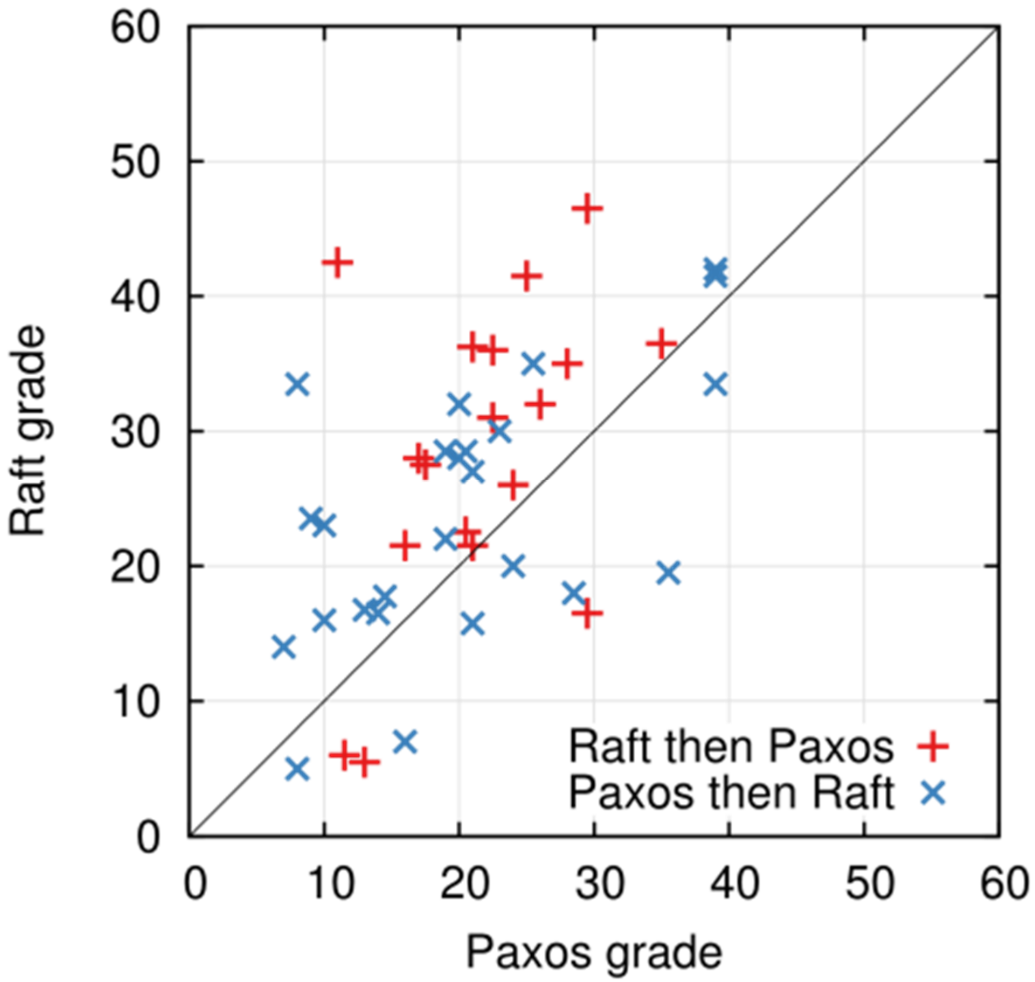
"There are significant gaps between the description of the Paxos algorithm and the needs of a real-world system…the final system will be based on an unproven protocol." – Chubby authors

Raft’s Design for Understandability

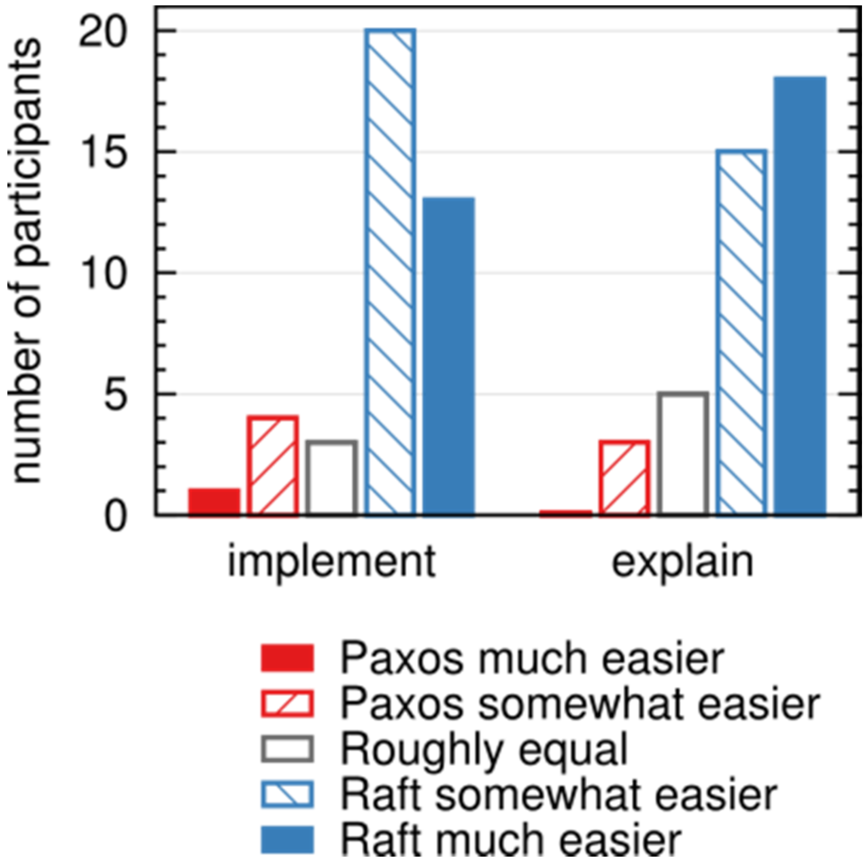
- **We wanted the best algorithm for building real systems**
  - Must be correct, complete, and perform well
  - Must also be understandable
- **“What would be easier to understand or explain?”**
  - Fundamentally different decomposition than Paxos
  - Less complexity in state space
  - Less mechanism

Raft User Study

Quiz Grades



Survey Results



Raft Overview

1. **Leader election**
   - Select one of the servers to act as cluster leader
   - Detect crashes, choose new leader
2. **Log replication (normal operation)**
   - Leader takes commands from clients, appends them to its log
   - Leader replicates its log to other servers (overwriting inconsistencies)
3. **Safety**
   - Only a server with an up-to-date log can become leader

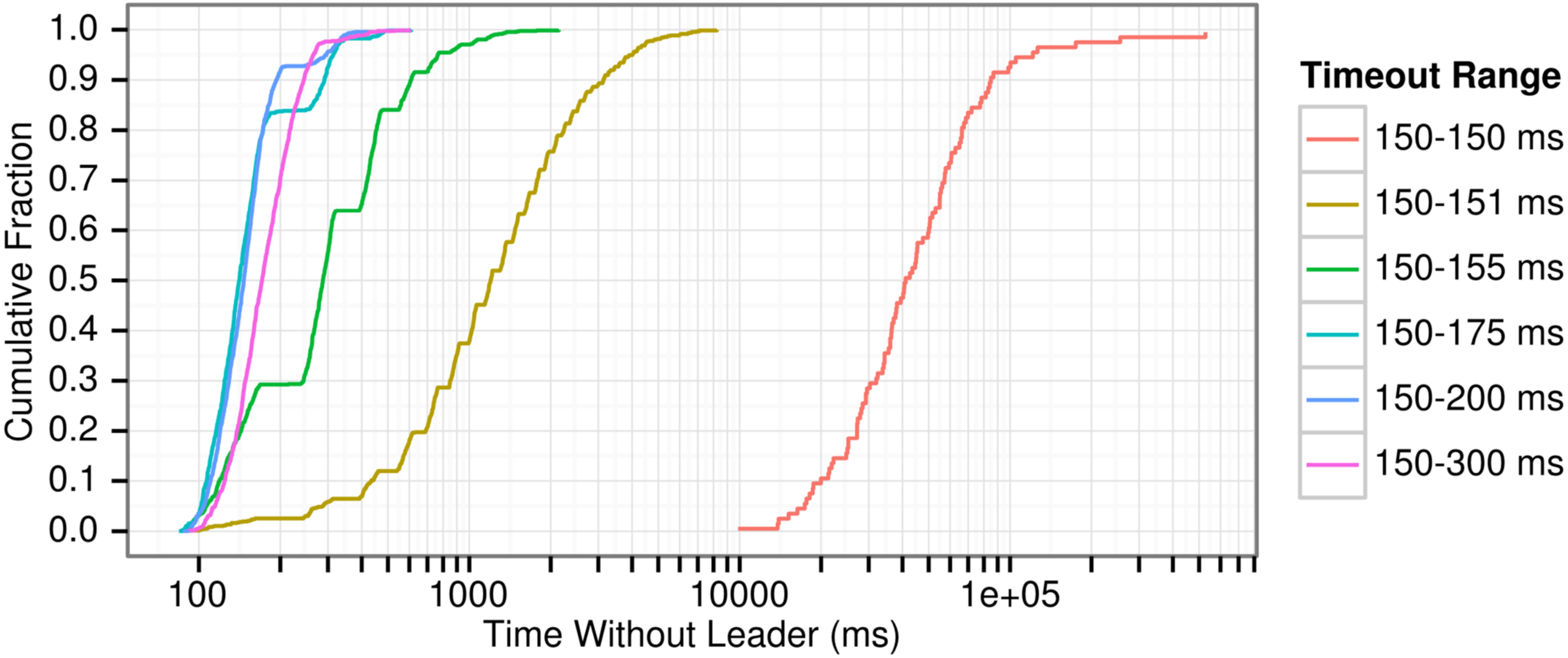
RaftScope Visualization

Core Raft Review

1. **Leader election**
   - Heartbeats and timeouts to detect crashes
   - Randomized timeouts to avoid split votes
   - Majority voting to guarantee at most one leader per term
2. **Log replication (normal operation)**
   - Leader takes commands from clients, appends them to its log
   - Leader replicates its log to other servers (overwriting inconsistencies)
   - Built-in consistency check simplifies how logs may differ
3. **Safety**
   - Only elect leaders with all committed entries in their logs
   - New leader defers committing entries from prior terms

Randomized Timeouts

- **How much randomization is needed to avoid split votes?**

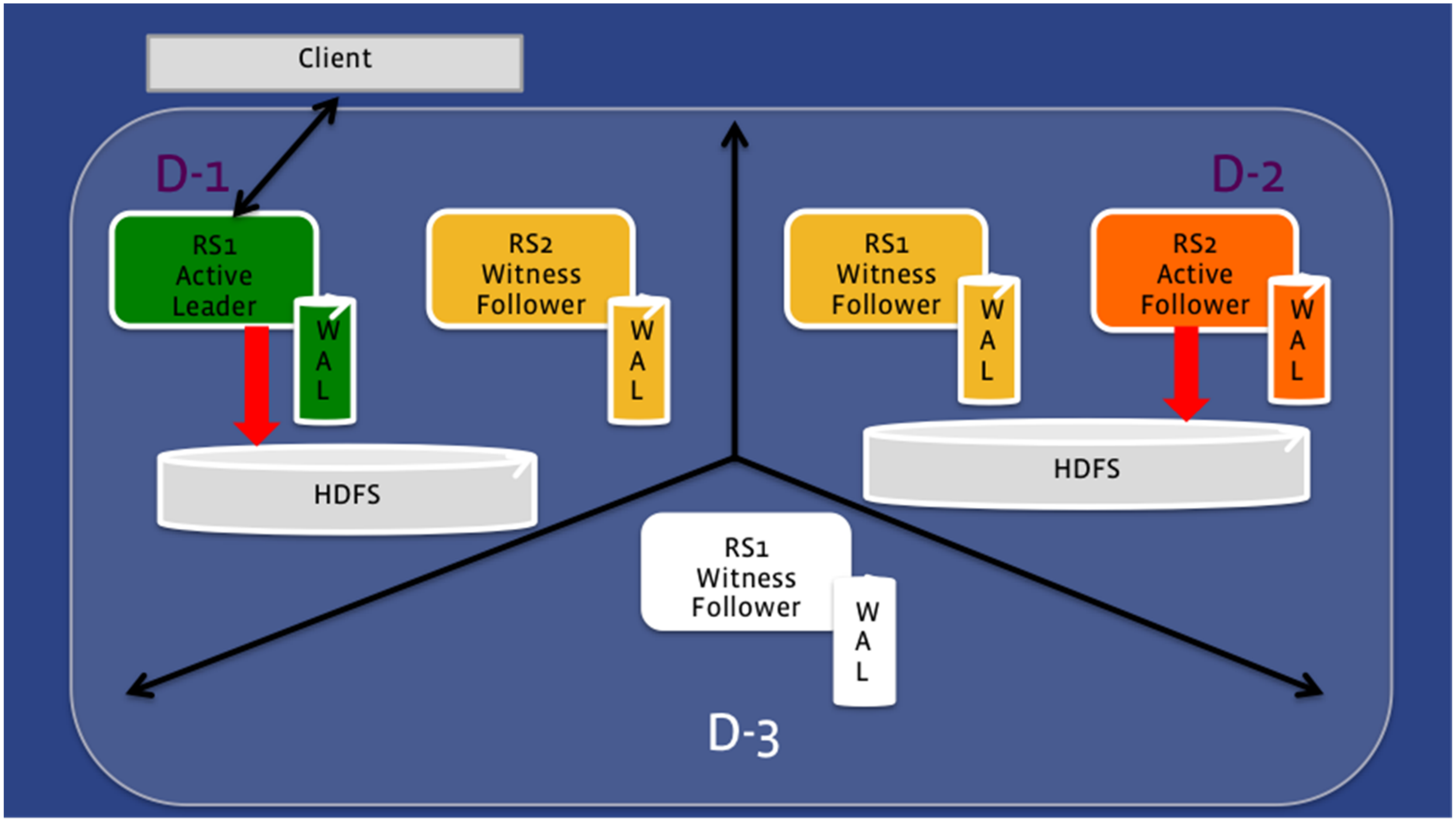


- **Conservatively, use random range ~10x network latency**

Raft Implementations (Stale)

go-raft	Go	Ben Johnson (Sky) and Xiang Li (CoreOS)
kanaka/raft.js	JS	Joel Martin
hashicorp/raft	Go	Armon Dadgar (HashiCorp)
rafter	Erlang	Andrew Stone (Basho)
ckite	Scala	Pablo Medina
kontiki	Haskell	Nicolas Trangez
LogCabin	C++	Diego Ongaro (Stanford)
akka-raft	Scala	Konrad Malawski
floss	Ruby	Alexander Flatten
CRaft	C	Willem-Hendrik Thiart
barge	Java	Dave Rusek
harryw/raft	Ruby	Harry Wilkinson
py-raft	Python	Toby Burress
…		

Facebook HydraBase Example



https://code.facebook.com/posts/321111638043166/hydrabase-the-evolution-of-hbase-facebook/

Conclusions

- **Consensus widely regarded as difficult**
- **Raft designed for understandability**
  - Easier to teach in classrooms
  - Better foundation for building practical systems
- **Paper/thesis covers much more**
  - Cluster membership changes (simpler in thesis)
  - Log compaction (expanded tech report/thesis)
  - Client interaction (expanded tech report/thesis)
  - Evaluation (thesis)

Questions

raftconsensus.github.io