ZooKeeper

A highly available, scalable, distributed, configuration, consensus, group membership, leader election, naming, and coordination service

Flavio Junqueira, Mahadev Konar, Andrew Kornev, Benjamin Reed

Observations

1) Distributed systems always need some form of coordination
2) Programmers cannot use locks correctly
3) Message based coordination can be hard to use in some applications

Wishes

1) Simple, Robust, Good Performance
2) Tuned for Read dominant workloads
3) Familiar models and interfaces
4) Wait-Free: A slow/failed client will not interfere with the requests of a fast client
5) Need to be able to wait efficiently

Design Starting Point

Start with the File API and model strip out what we don't need:

1)Partial writes/reads (takes with it open/close/seek)

2)Rename

add what we do need:

1)Ordered updates and strong persistence guarantees

2)Conditional updates

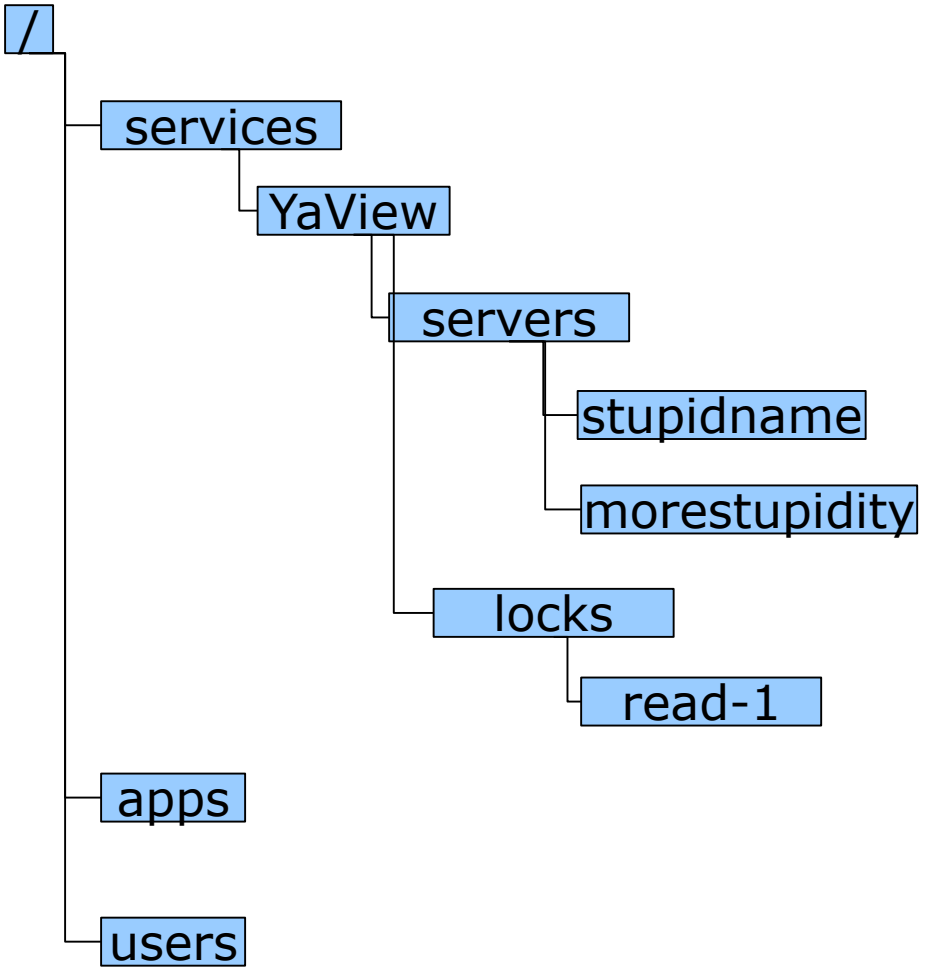
3)Watches for data changes

4)Ephemeral nodes

5)Generated file names

Data Model

1) Hierarchal namespace (like a file system)
2) Each znode has data and children
3) data is read and written in its entirety



ZooKeeper API

String create(path, data, acl, flags)

void delete(path, expectedVersion)

Stat setData(path, data, expectedVersion)

(data, Stat) getData(path, watch)

Stat exists(path, watch)

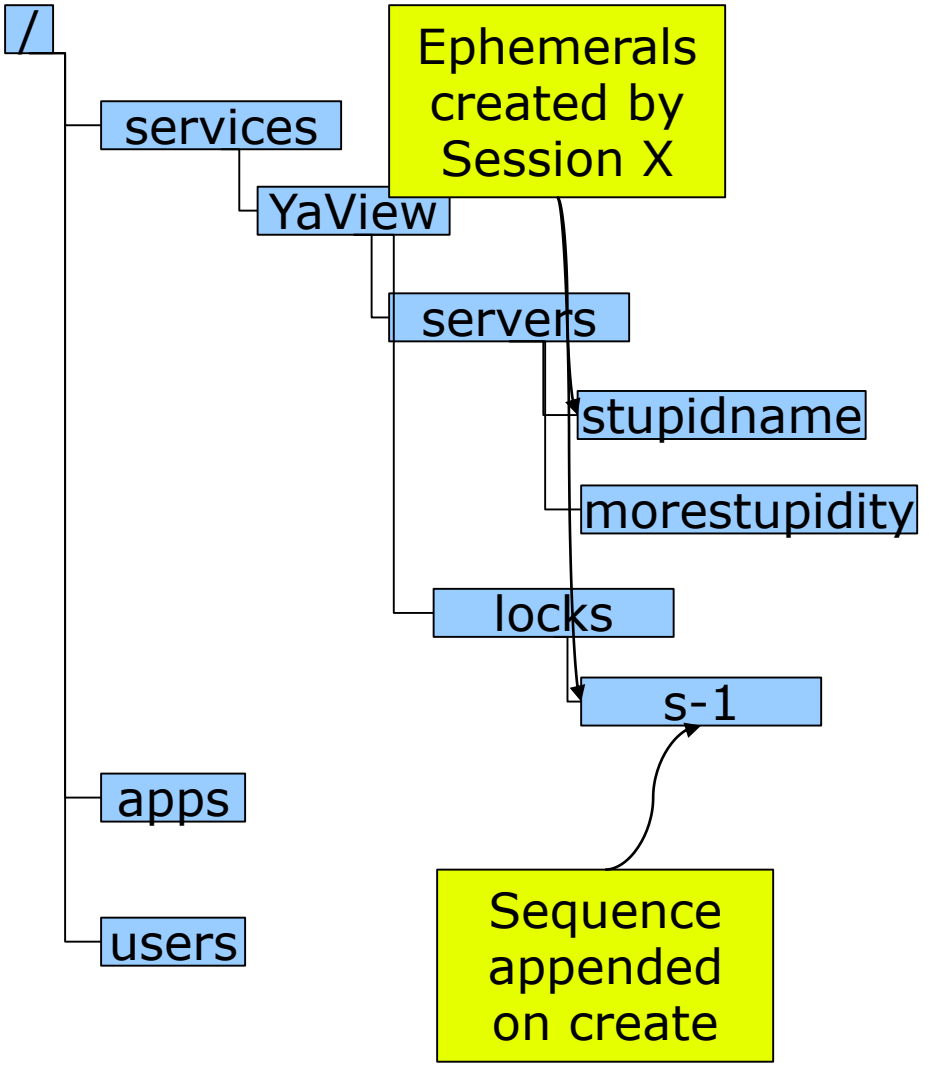
String[] getChildren(path, watch)

void sync(path)

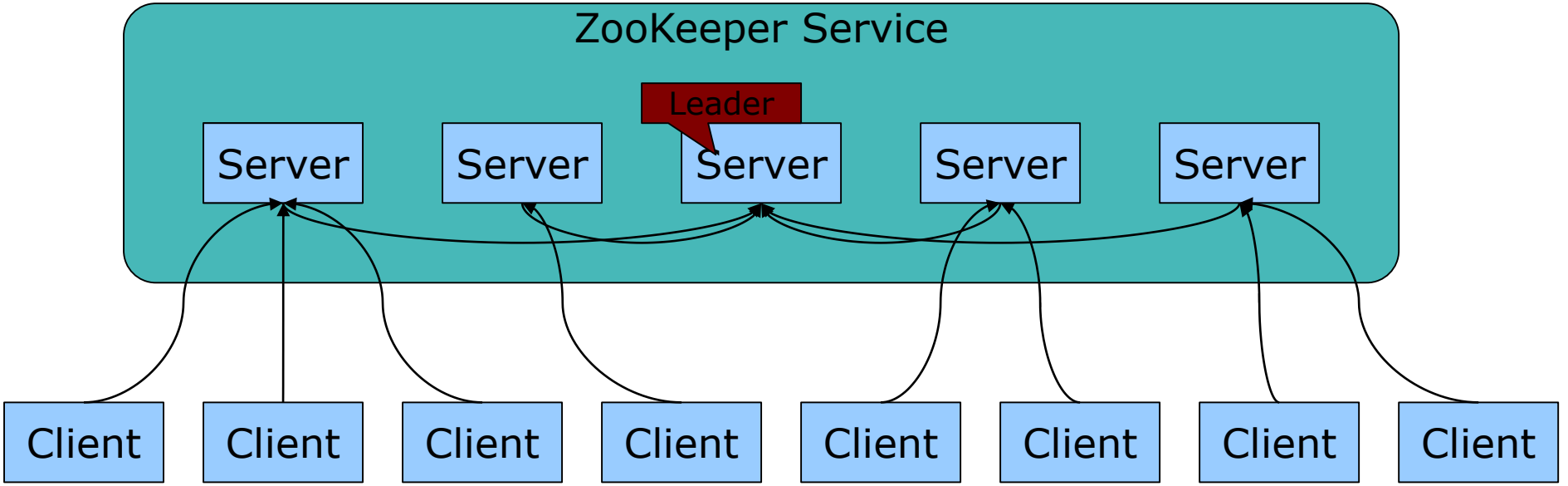
Create Flags

1)Ephemeral: the znode will be deleted when the session that created it times out or it is explicitly deleted

2)Sequence: the the path name will have a monotonically increasing counter relative to the parent appended



ZooKeeper Servers



1)All servers store a copy of the data (in memory)

2)A leader is elected at startup

3)Followers service clients, all updates go through leader

4)Update responses are sent when a majority of servers have persisted the change

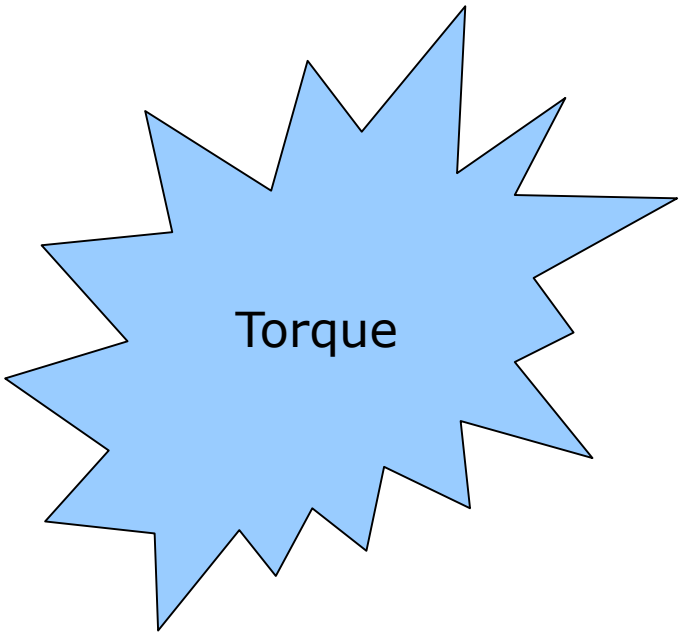
HOD

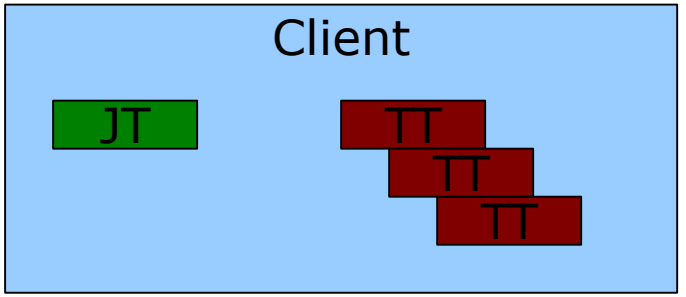
1)A client submits a request to start jobtracker and a set of tasktrackers to torque

2)The ip address and the ports that the jobtracker will bind to is not known apriori

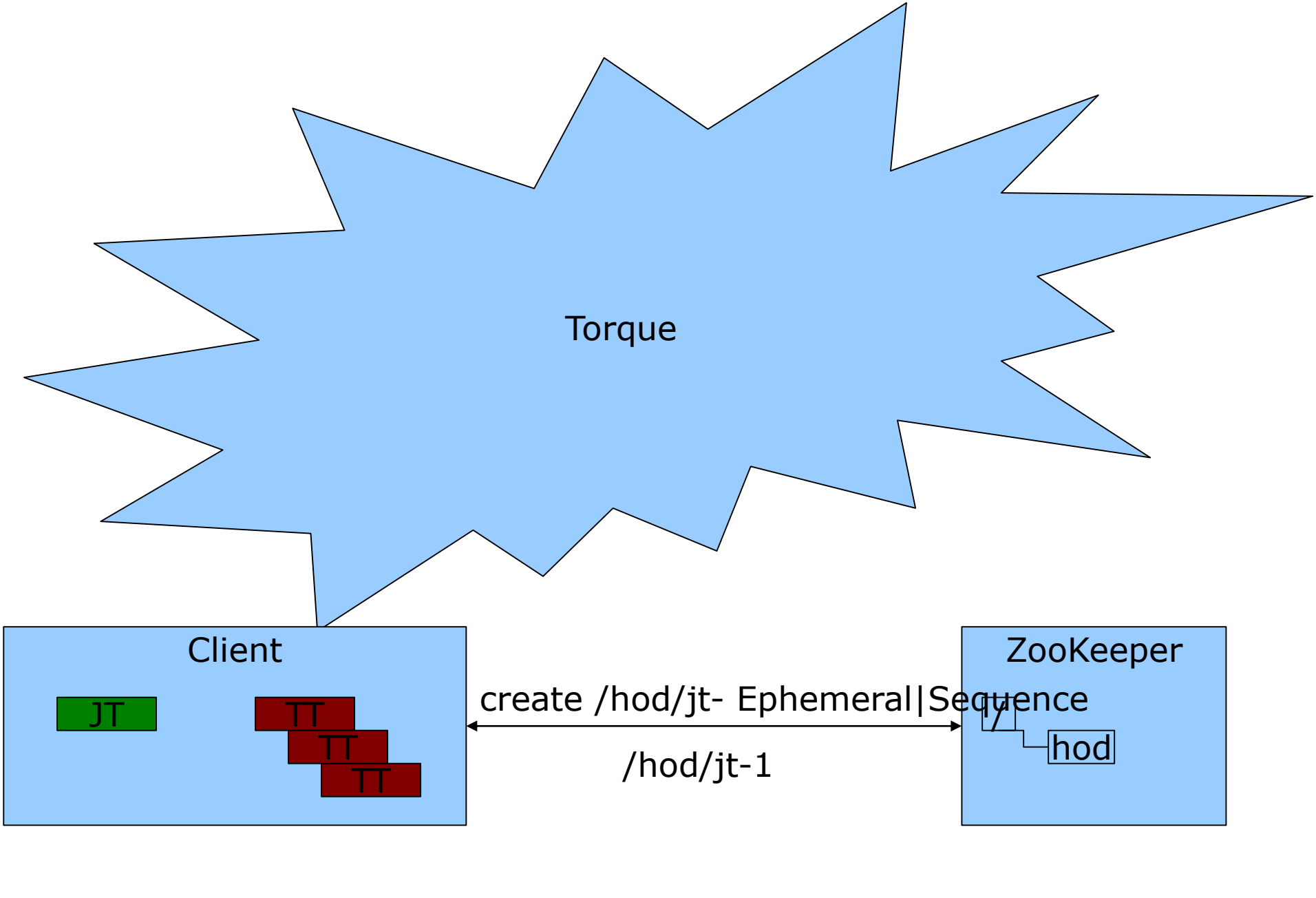
3)The tasktrackers need to find the jobtracker

4)The client needs to find the jobtracker

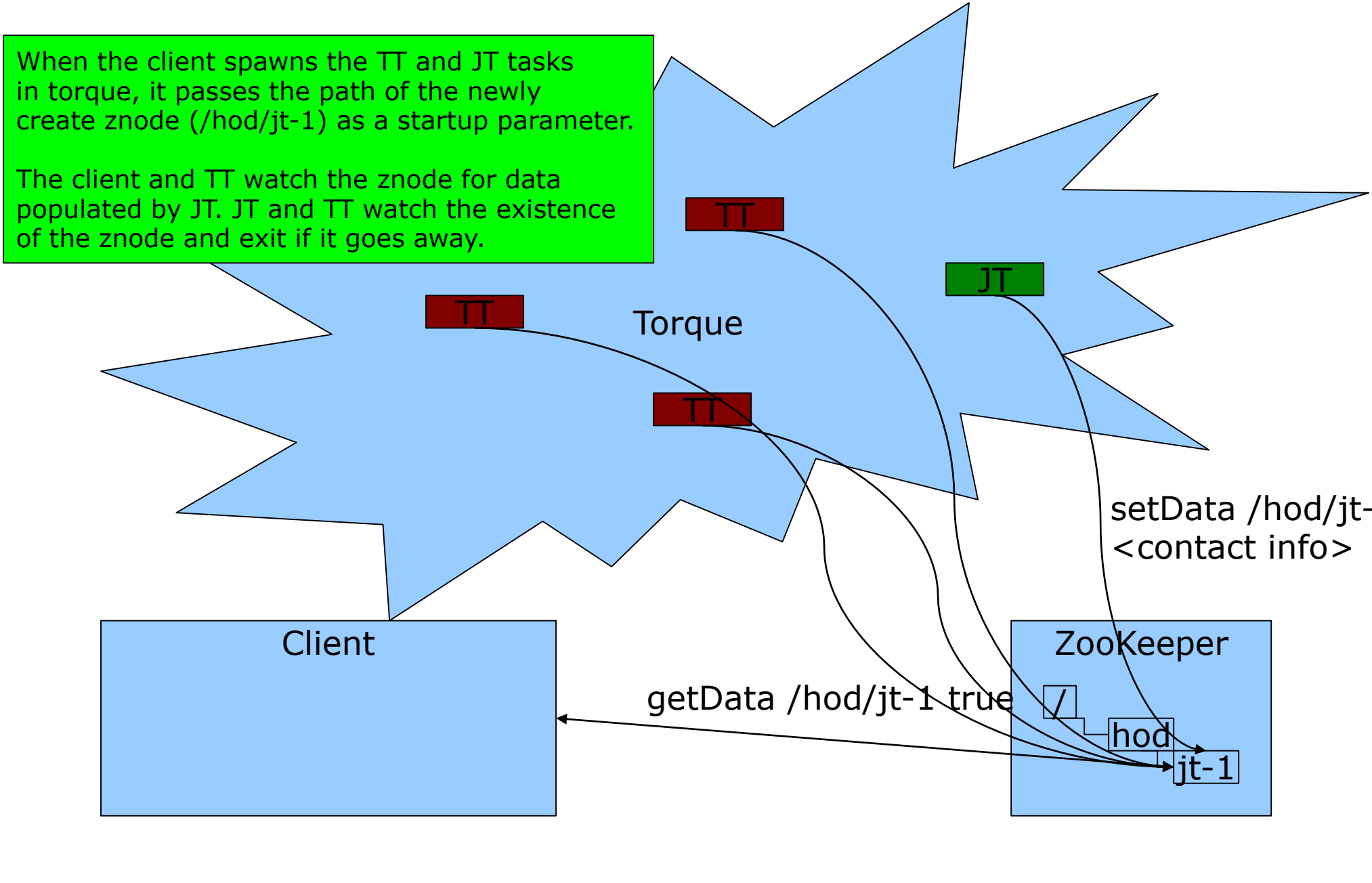




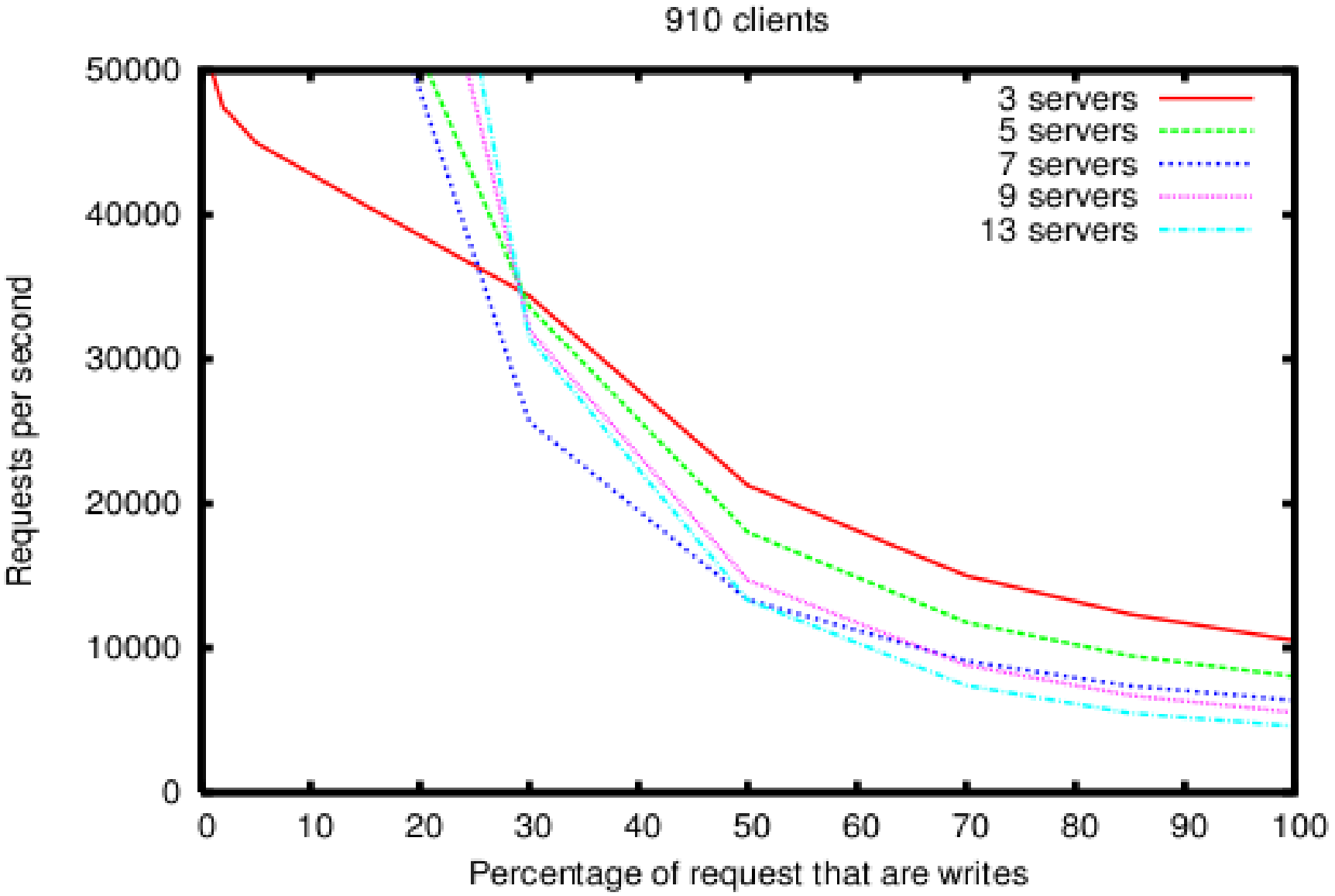
HOD with ZooKeeper



HOD with ZooKeeper



Performance



Performance at Extremes

Servers	1% Writes	100% Writes
13	265115	4592
9	195178	5550
7	147810	6371
5	75308	8048
3	49827	10519

Numbers are operations per second

Status

1) Project started October 2006
2) Prototyped in Fall 2006
3) Initial implementation of production service March 2007
4) Code moved to zookeeper.sf.net and Apache License November 2007
5) Java Quorum and Standalone servers
6) Java and C clients available