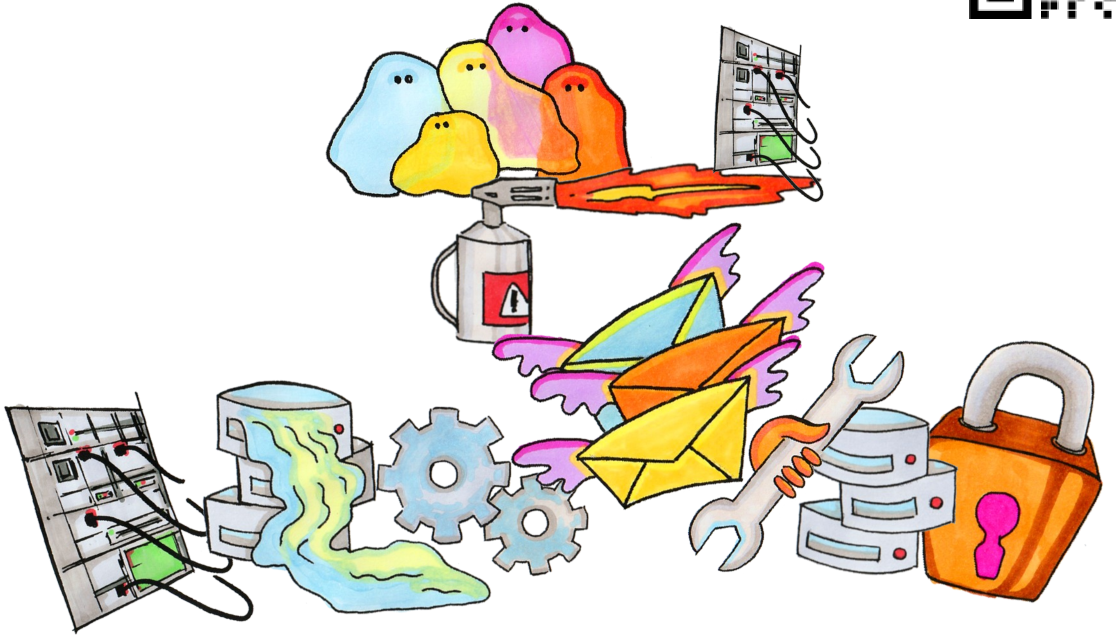
25 Years Of Resilient Systems

Dave Cottlehuber

dch@skunkwerks.at

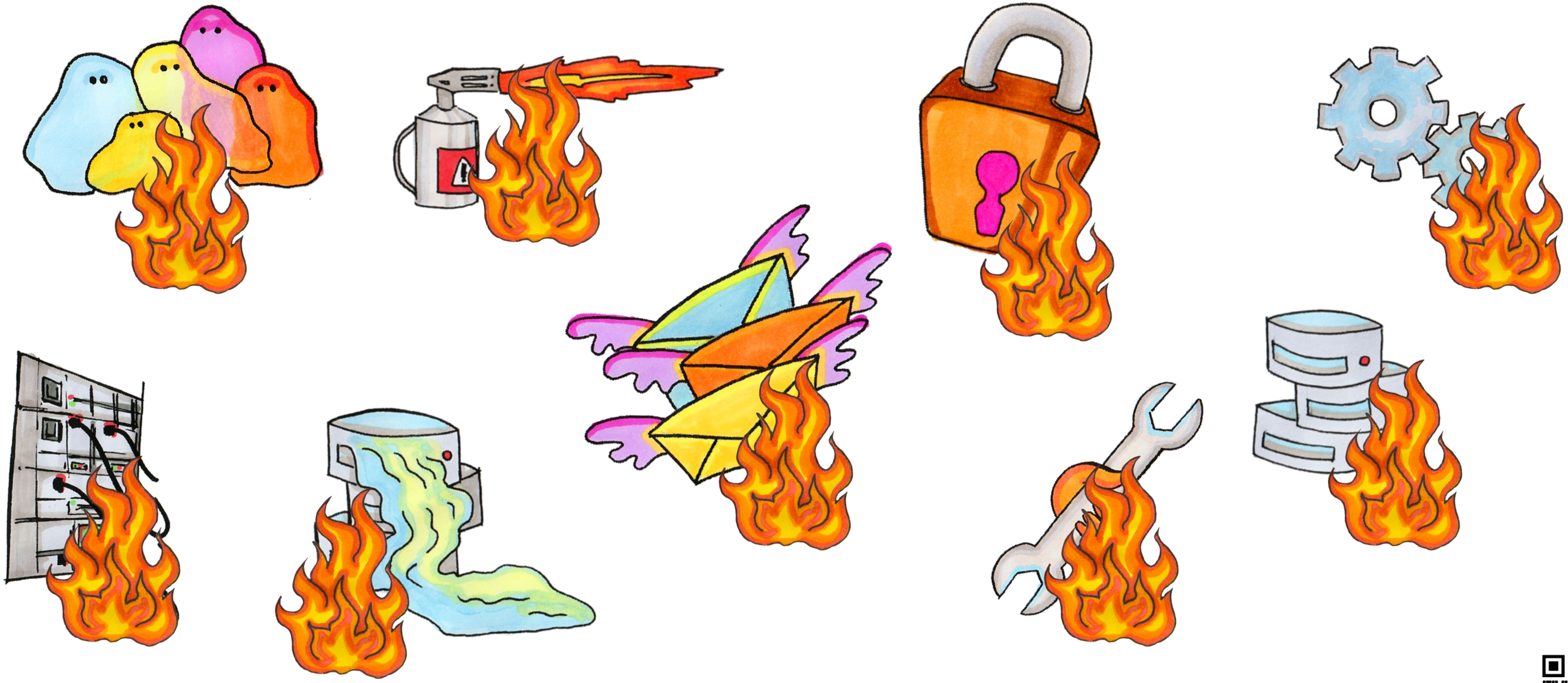
Graphics

@maycontainart



https://people.freebsd.org/~dch/talks/eurobsdcon2025/

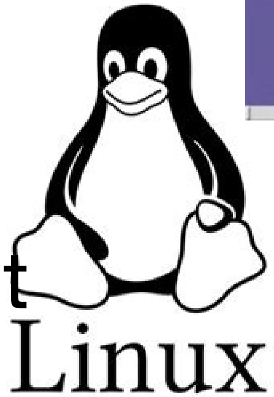
Predictable Modes of Failure

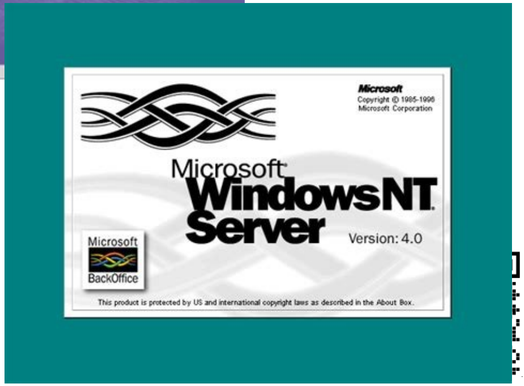


Delayed Enterprise Financial System

- Campus-wide
- Novell NetWare
- OpenVMS
- Various Linux Systems
- Windows NT
- Solaris or SunOS I forget

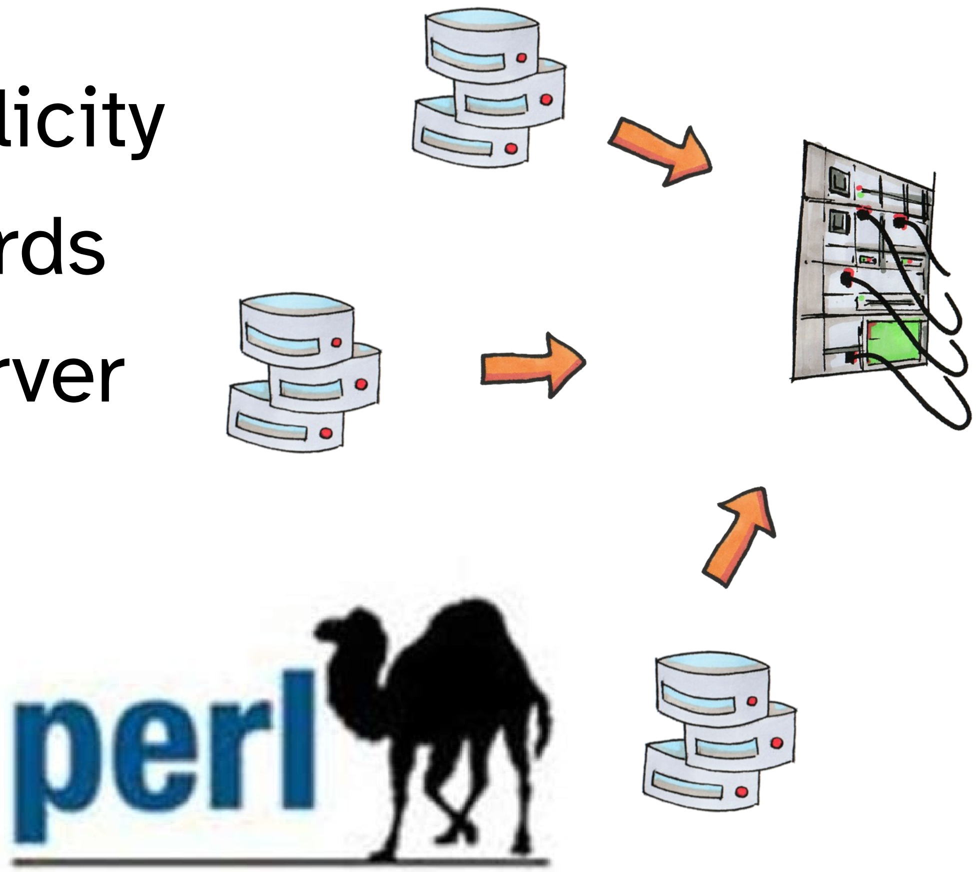
```
NetWare v3.12 (10 user) - 8/12/93                NetWare 386 Loadable Module

                    Information For Server ODYSSEY

  File Server Up Time:     0 Days  0 Hours  0 Minutes  9 Seconds
  Utilization:                   4 |   Packet Receive Buffers:    100
  Original Cache Buffers:    3,078 |   Directory Cache Buffers:    20
  Total Cache Buffers:         507 |   Service Processes:           1
  Dirty Cache Buffers:           1 |   Connections In Use:          0
  Current Disk Requests:         0 |   Open Files:                  4

                    Available Options

                 Connection Information
                 Disk Information
                 LAN Information
                 System Module Information
                 Lock File Server Console
                   ck Activity
                   zation
```



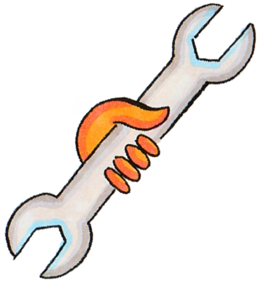


Spoke & Hub Batch Processing

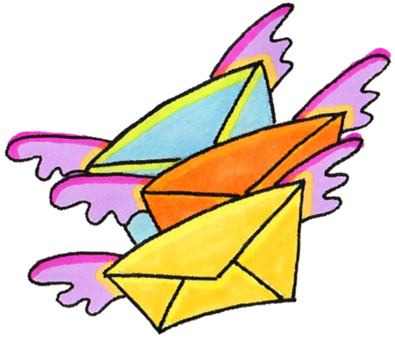
- Operational Simplicity
- Idempotent Records
- Single Central Server
- Collector Agents
- Transfer Agents

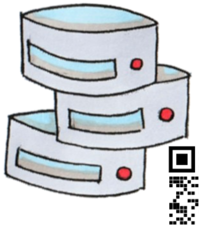


Loosely Coupled



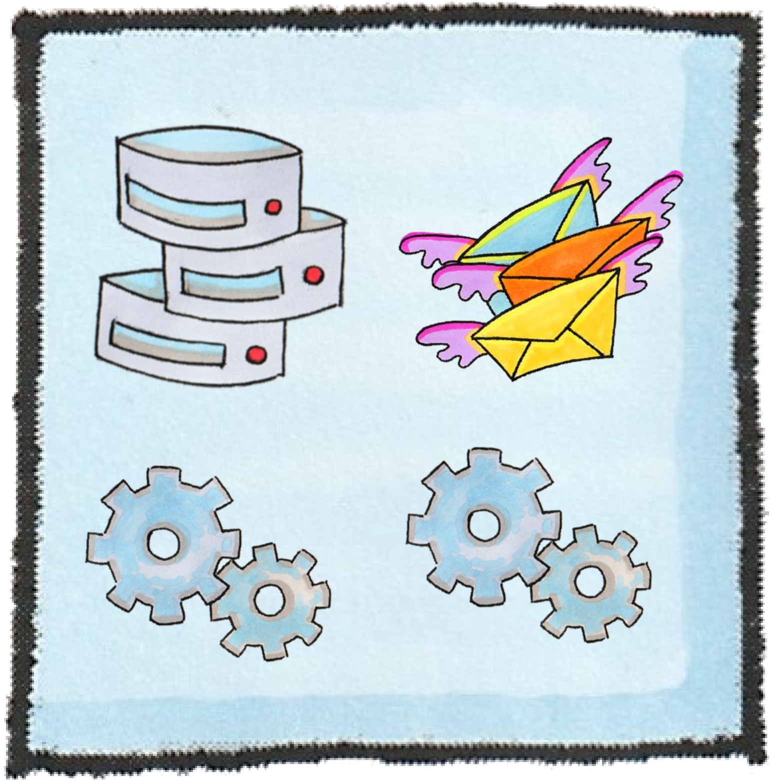
- Operational Simplicity
- Autonomous Agents are Resilient
- Open Source is a 10x advantage
- System Lifespan exceeds Employment Lifespan



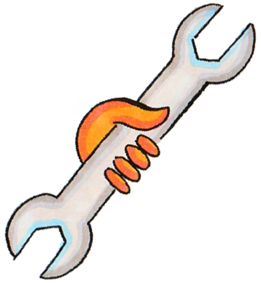


The Single Server

- Conceptually Simple
- Scales Well
- Until It Fails
- Good Performance
- Moore’s Law helps

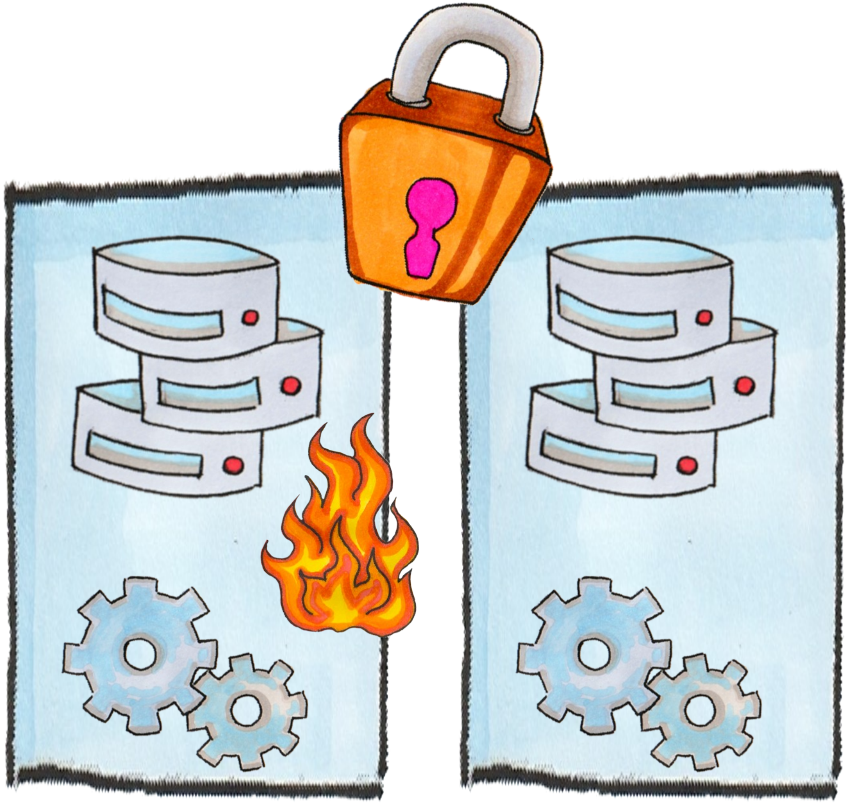


Takeaways



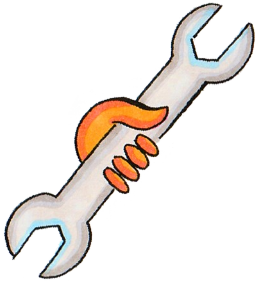
- Co-located Services Are Fast & Easy
- All Your Eggs in a Single Basket
- Upgrades are Hard
- Failure is even Harder
- Infrastructure is Expensive

Double Up On Everything



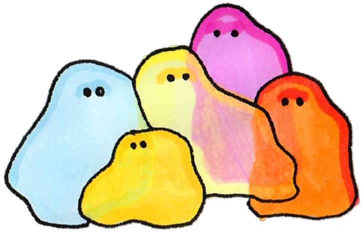
- Redundancy
- But Not Robustness
- Quorum Is Hard
- DB Integrity Is Hard

Takeaways



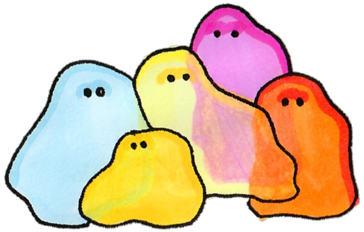
- Traded Simplicity For Redundancy
- Clusters Not Well Understood
- Split Brain Integrity Problems
- Want Load Balancers & Fancy Networks

Theory – In Bounded Time



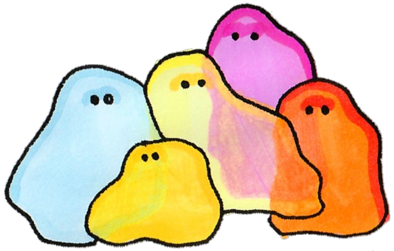
- Byzantine Consensus (Shostak, 1978)
- Impossibility of Distributed Consensus with One Faulty Process (Fischer, Lynch, Paterson, 1985)
- View-stamped Replication (Oki, Liskov, 1988)
- Paxos Parliament (Lamport, 1989)
- Practical Byzantine Fault Tolerance (Castro, Liskov 1999)
- Wait until 2014 for Raft paper (Ongaro, Osterhout, 2014)
- CAP conjecture (Brewer) and theorem (Gilbert, Lynch, 2002)

Byzantine Generals



- Coordination under adversarial conditions
- Multiple generals must agree on attack/retreat to win the battle or risk annihilation
- Some generals may be traitors
- Communication through messengers only
- 3n + 1 nodes to accommodate n failure

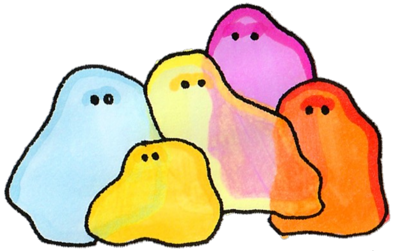
FLP Impossibility Result



- No deterministic algorithm can solve consensus in asynchronous systems **in bounded time**
- Even a single crash/failure/hostile agent is enough
- No bounds on message delays or processing time

- Consensus is impossible without additional assumptions
- Timeouts, failure detection, randomisation
- Partial synchrony required, or eventual synchrony

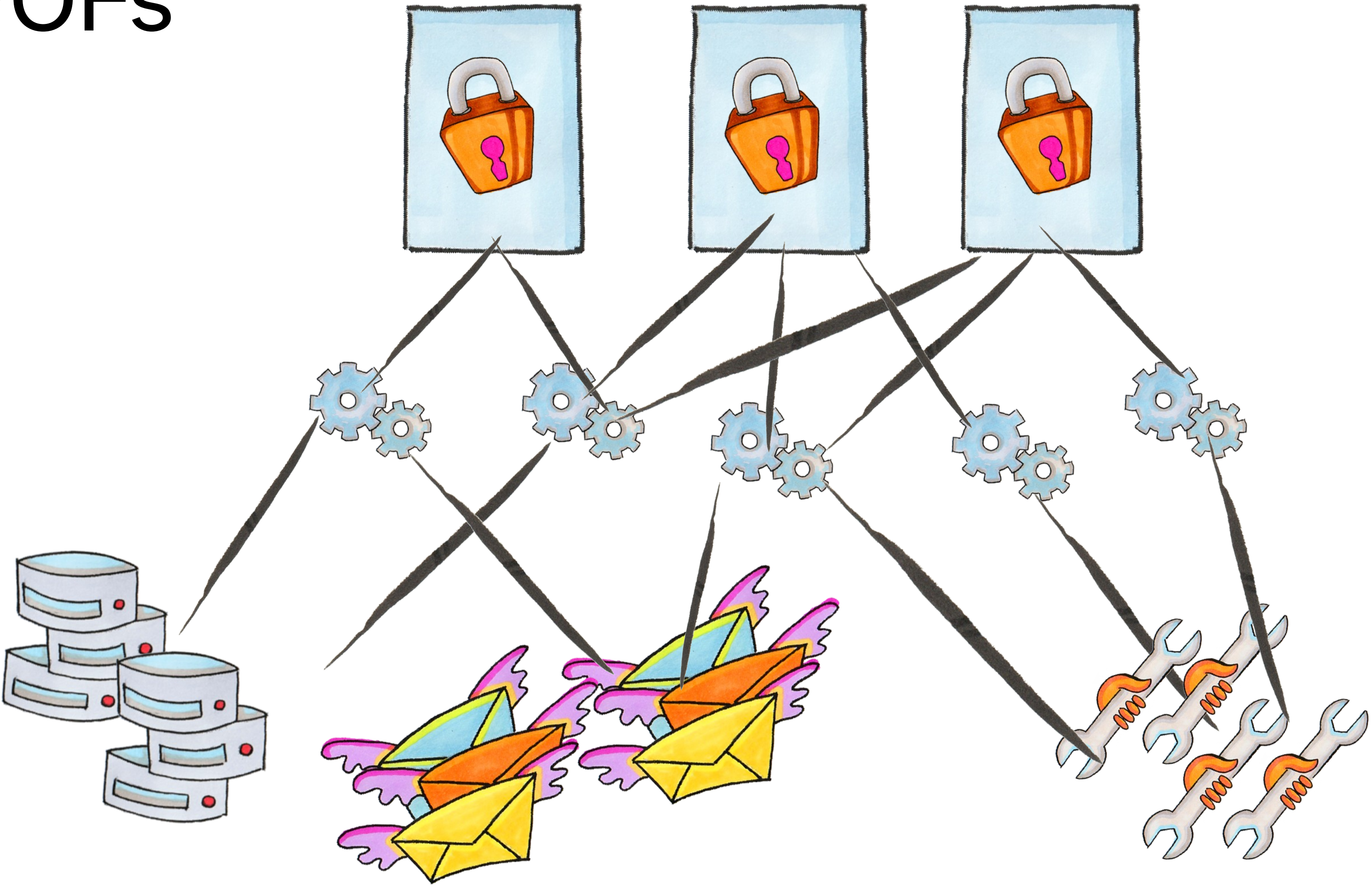
CAP the Impossible Triangle



- Consistency: all node see the same data simultaneously
- Availability: system returns responses despite failures
- Partition Tolerance: system continues to accept writes despite network splits

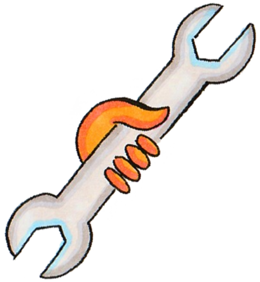
- Only 2/3 properties possible
- Partitions are inevitable
- Thus CP or AP under partition failure
- Bounded time (again!)
- You can’t skip P, so either C or A

No SPOFs



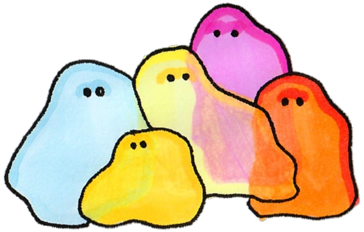
FRA WLG SNG

Takeaways

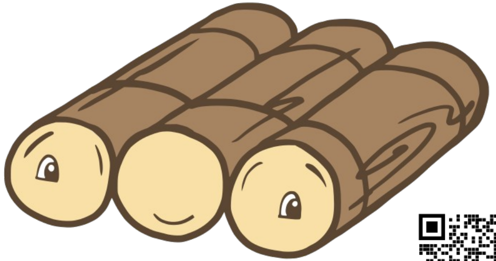


- Definitely Not Operationally Simple
- Excellent Scalability, horizontal & regional
- Database Layer still not ideal
- Consensus is Genuinely Hard

Theory – Distributed Systems

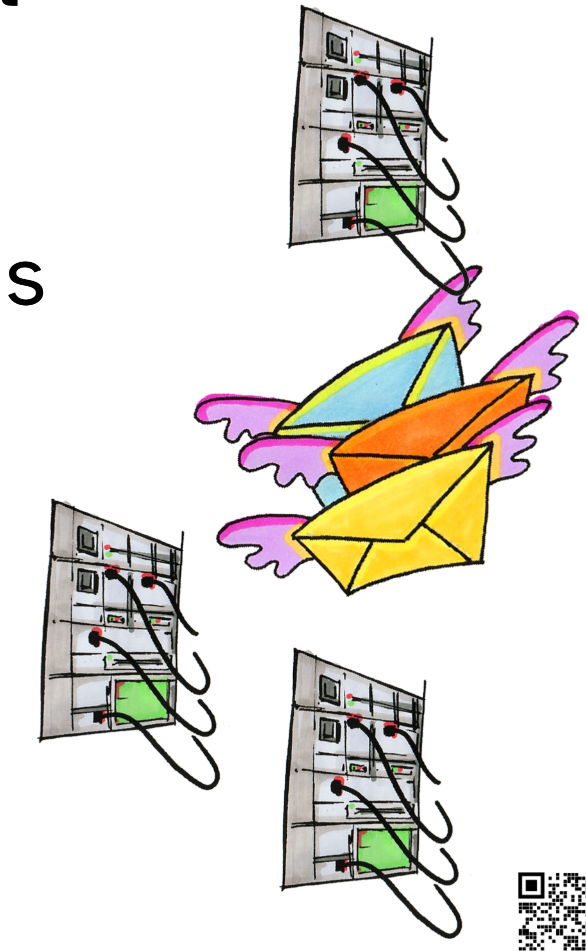


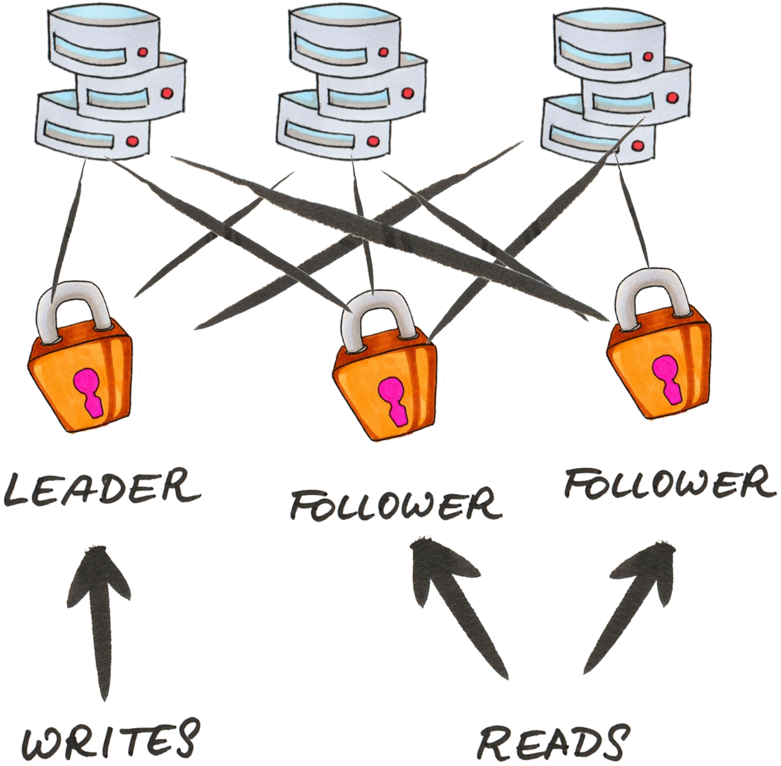
- Convergent & Commutitative Replicated Datatypes
  - Shapiro, 2011
- More Paxos
  - Lamport & Friends
- Raft Algorithm (Ongaro, Osterhout, 2014)
  - Logo by Andrea Ruygt

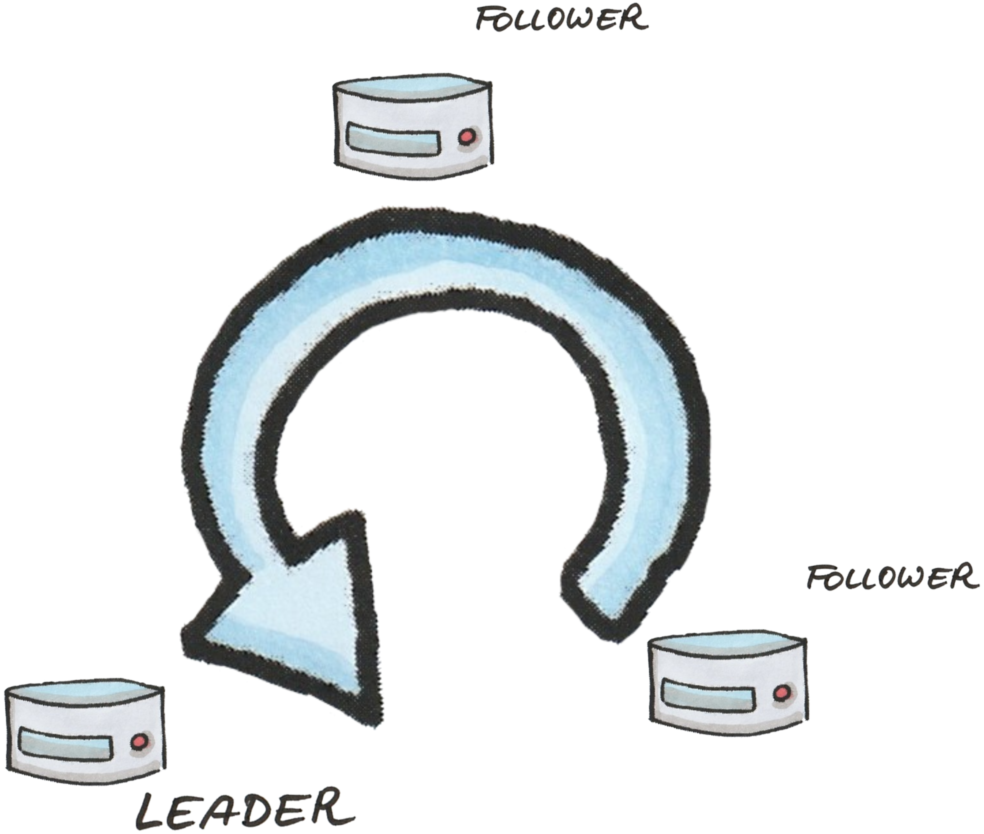


Raft In a Nutshell

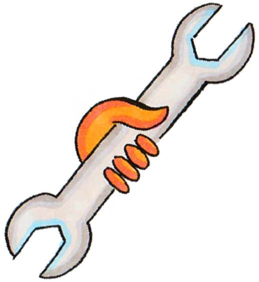
- Replicated State Machine
- Agreement on Ordered Transitions
- Trusted Leaders & Followers
- Log Replication
- Not Byzantine
- Timeouts & Heartbeats




LEADER
FOLLOWER
FOLLOWER
WRITES
READS

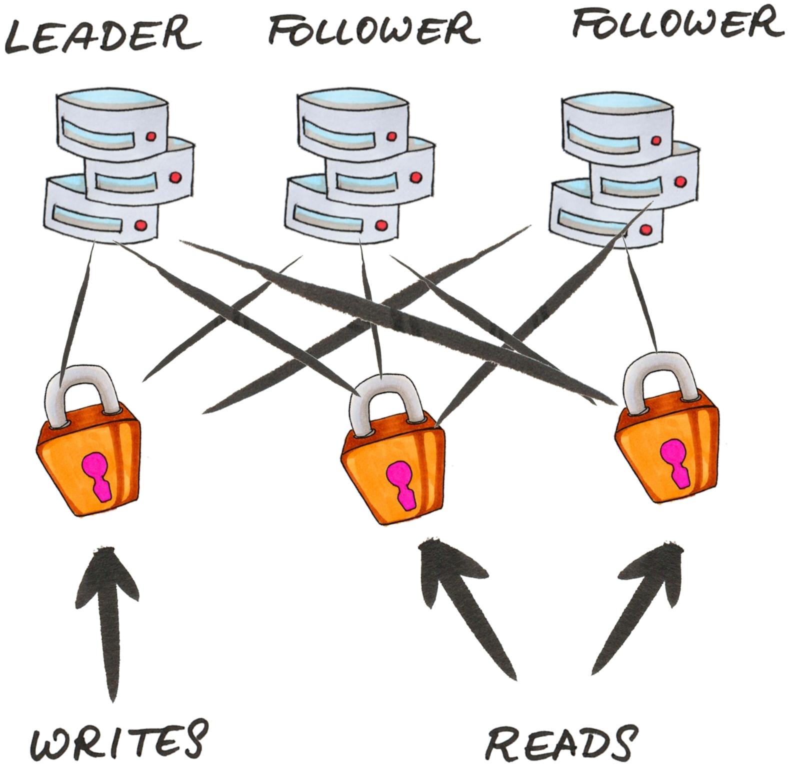

FOLLOWER
FOLLOWER
LEADER

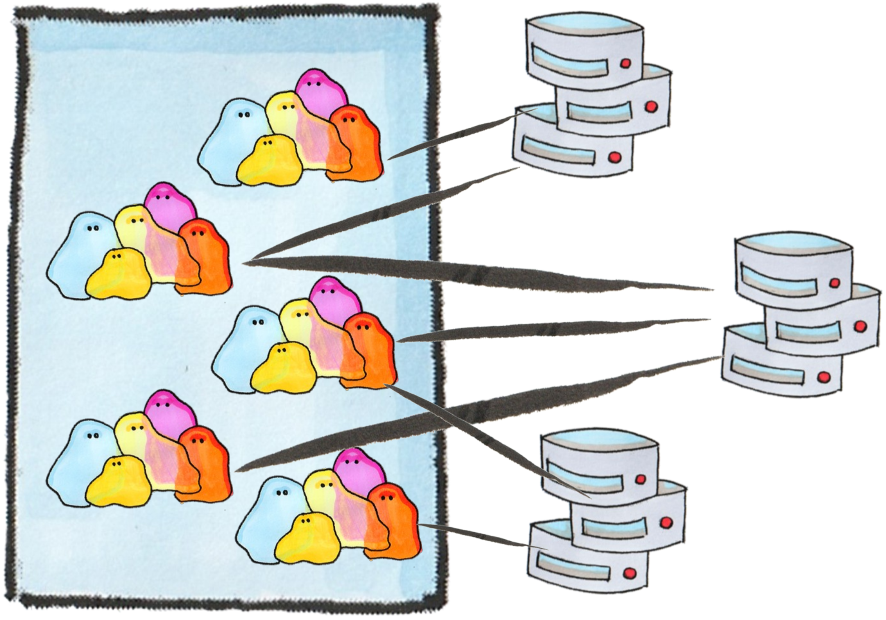
Takeaways



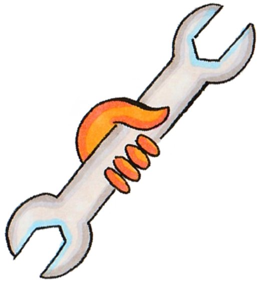
- Solved Cluster Problem
- Operationally Simple
- But Problems Cascade
- Performance & Throughput drastically compromised compared to optimal single-node performance

Raft & Blob Stores





Hacks and Workarounds



- Smart Clients
  - batching writes
  - knowledge of cluster topology
- Reduce need for quorum
  - Partitioned writes, coalesce quorum updates

A Secret Hack

- Squint hard
- Everything looks like a queue
- What happens when the queue is full?
- Model that behaviour
- Monitor & log it

In Bounded Time!

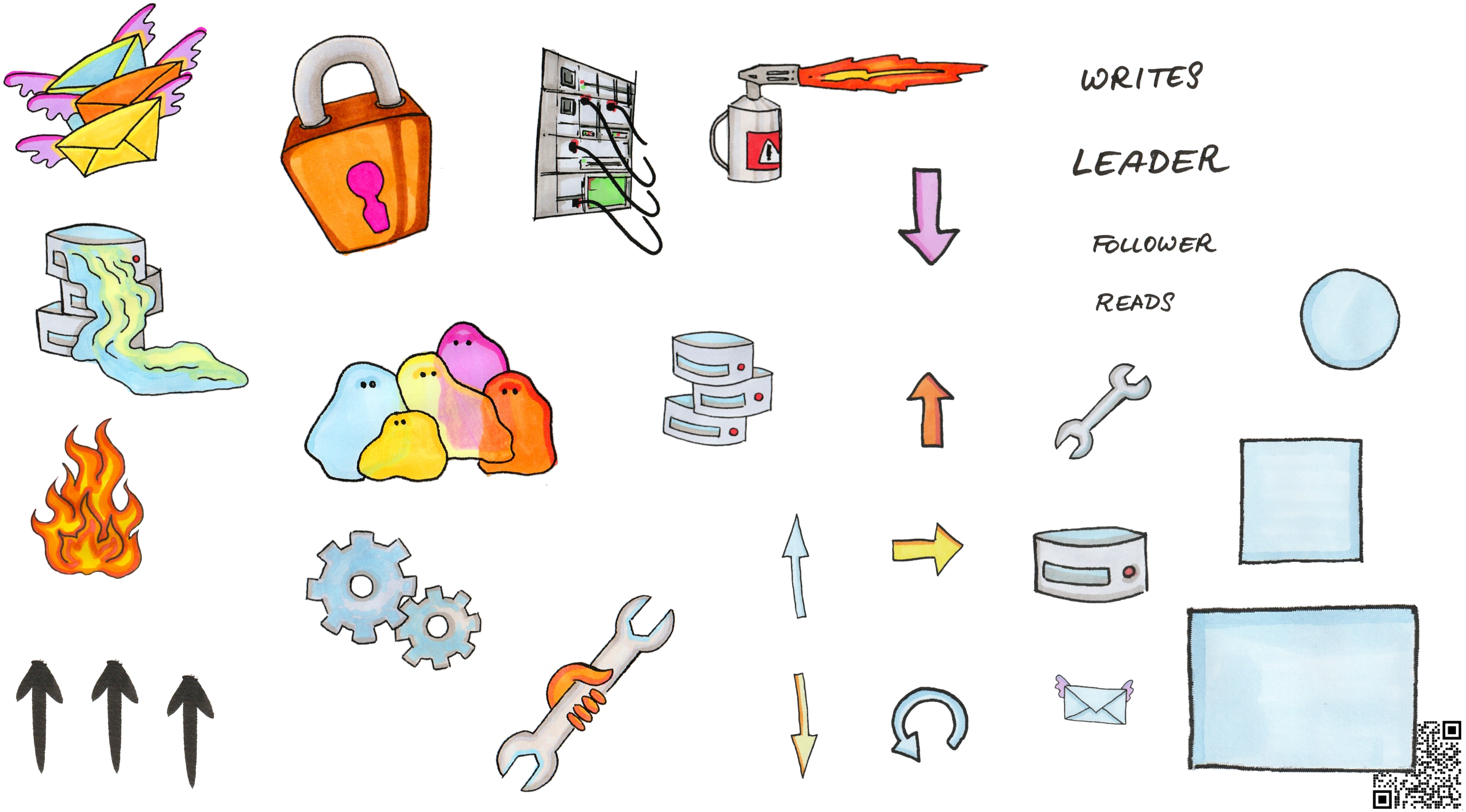
Log-replicated idempotent state machines across loosely coupled standards-based repeatable composable infrastructure, in bounded time.

- CAP
- FLP
- In Bounded Time!

Mastodon: @maycontainart@mastodon.art

E-Mail: contact@maycontain.art

May contain art



WRITES
LEADER
FOLLOWER
READS