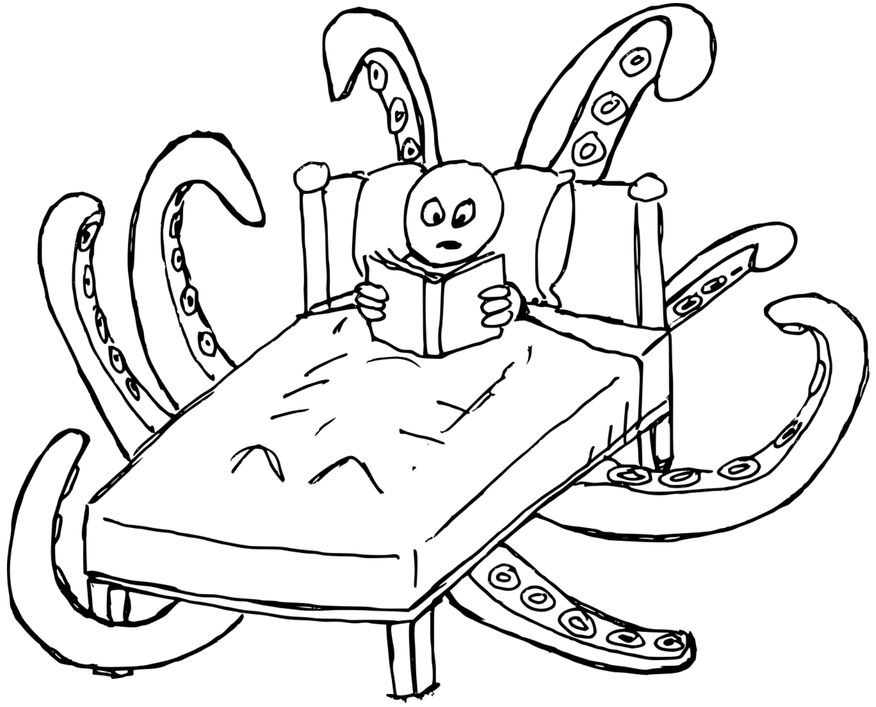
A Gentle Introduction to Ceph



Narrated by Tim Serong
tserong@suse.com

Adapted from a longer work by Lars Marowsky-Brée
lmb@suse.com

Once upon a time there was a Free and Open Source distributed storage solution named Ceph.

Ceph...

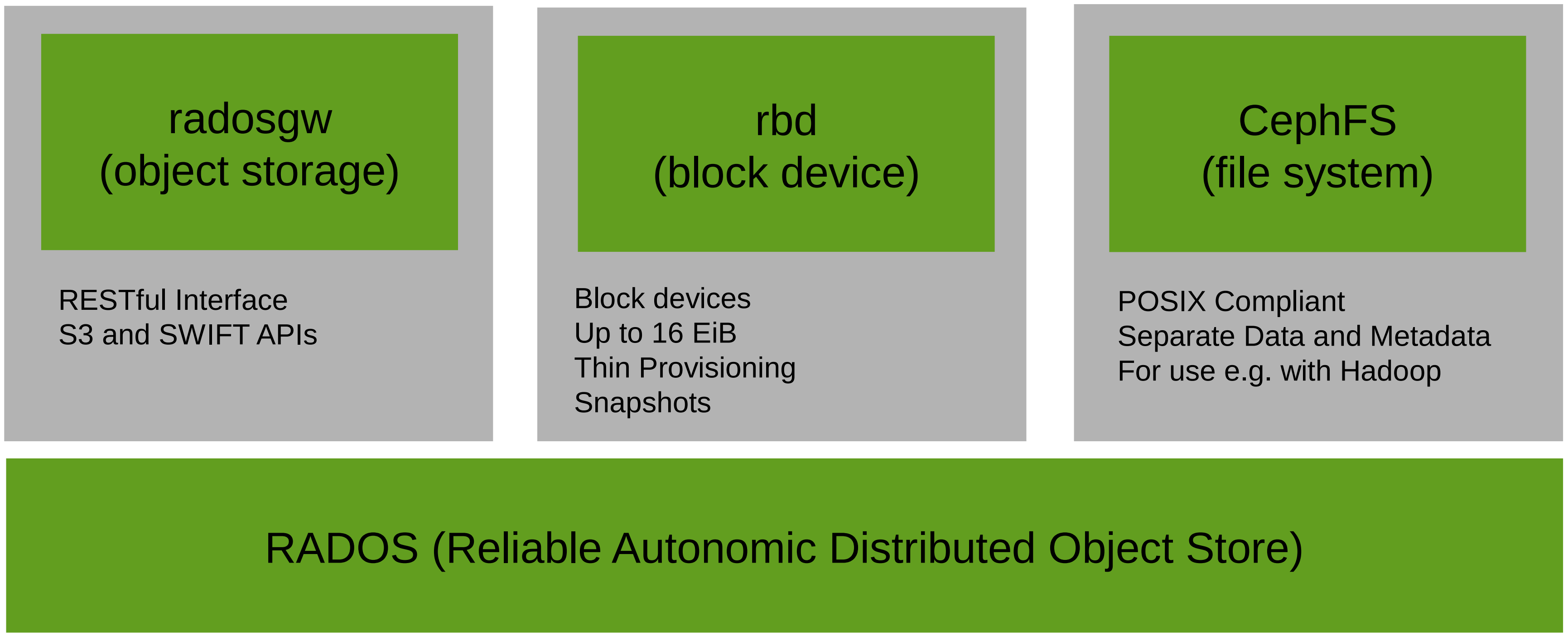
- Has been around for a while (first stable release in July 2012)
- Has lots of goodies:
  - Distributed object storage
  - Redundancy
  - Efficient scale-out
  - Build on commodity hardware
- Most popular choice of distributed storage for OpenStack[1]

[1] http://www.openstack.org/assets/survey/Public-User-Survey-Report.pdf (October 2015)

Ceph Gives Us...

- A Storage Cluster
  - Self healing
  - Self managed
  - No bottlenecks
- Three interfaces
  - Object Access (like Amazon S3)
  - Block Access
  - Distributed File System

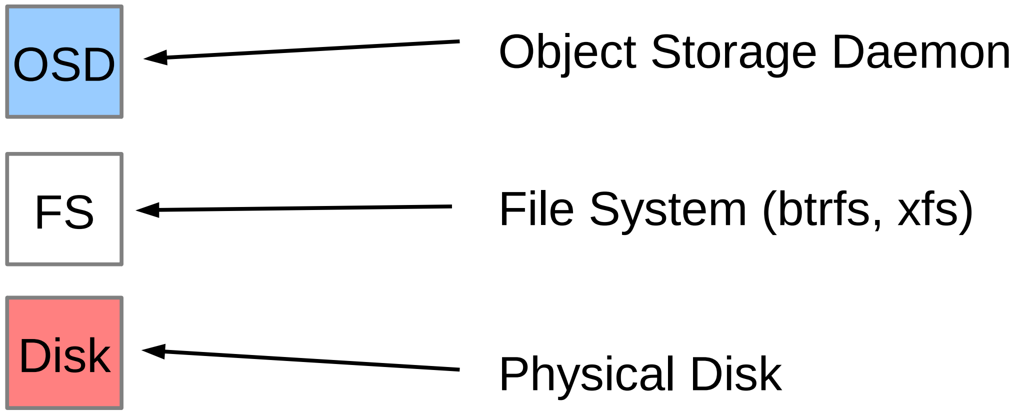
Ceph's Architecture



Once upon a time there was a Free and Open Source distributed storage solution named Ceph.

Sysadmins throughout the land needed to know the components that made up Ceph...

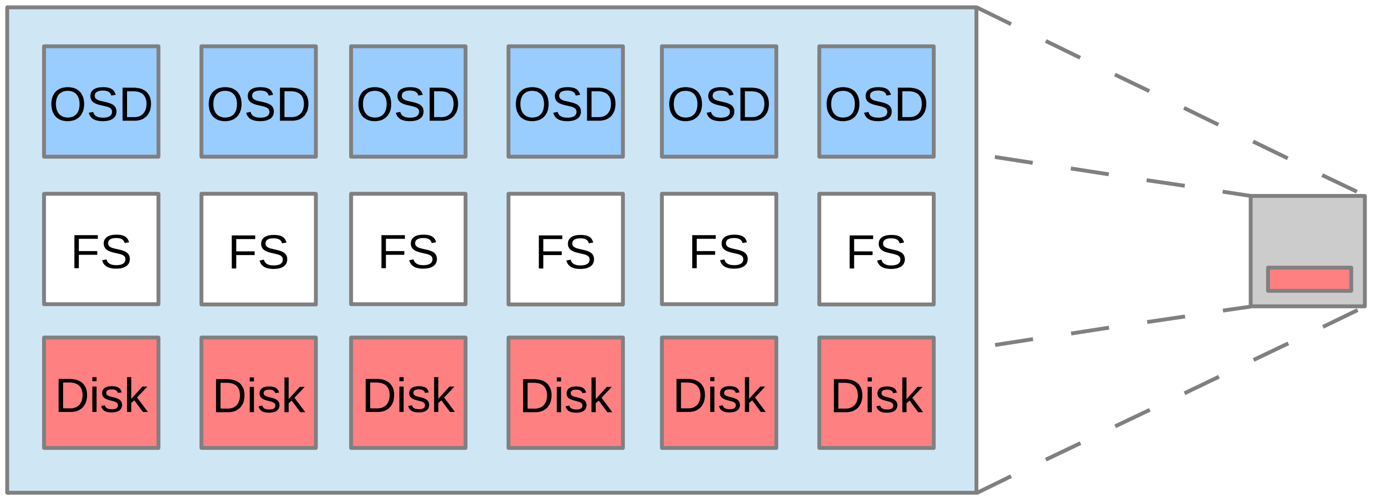
At the Lowest Level



OSDs serve stored objects to clients

Peer to perform replication and recovery

Put Several OSDs in One Node

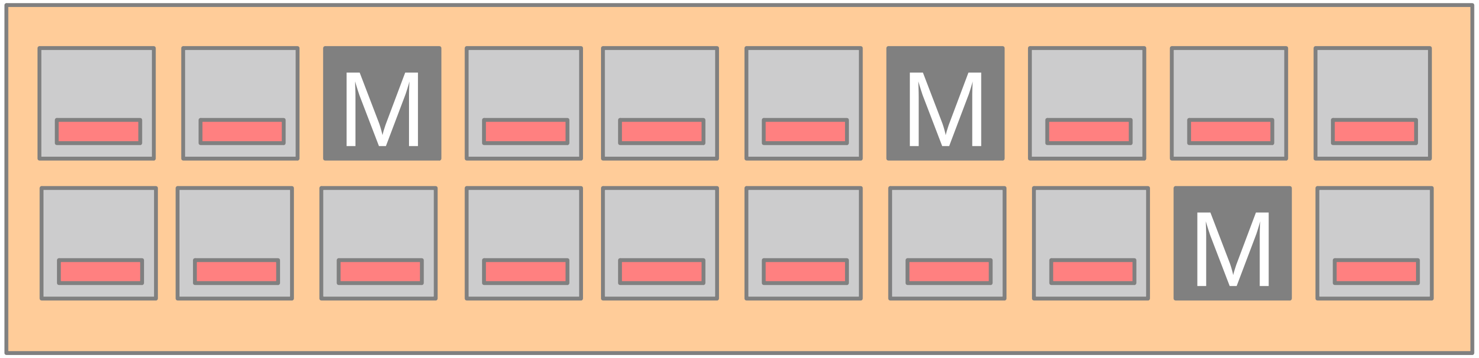


Add a Few Monitor Nodes

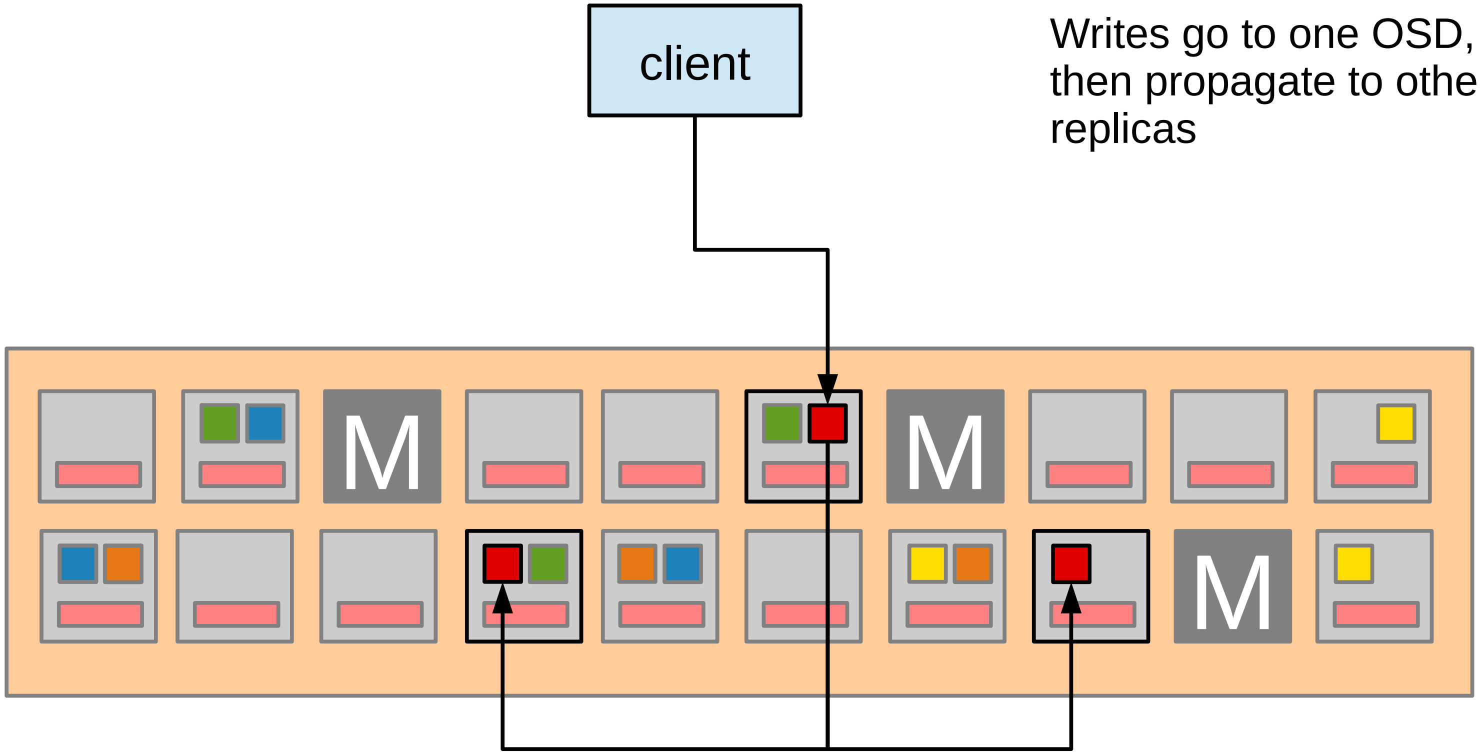
Monitors are the brain cells of the cluster
- Cluster membership
- Consensus for distributed decision making

Do not serve stored objects to clients

...And You Get a Small Ceph Cluster

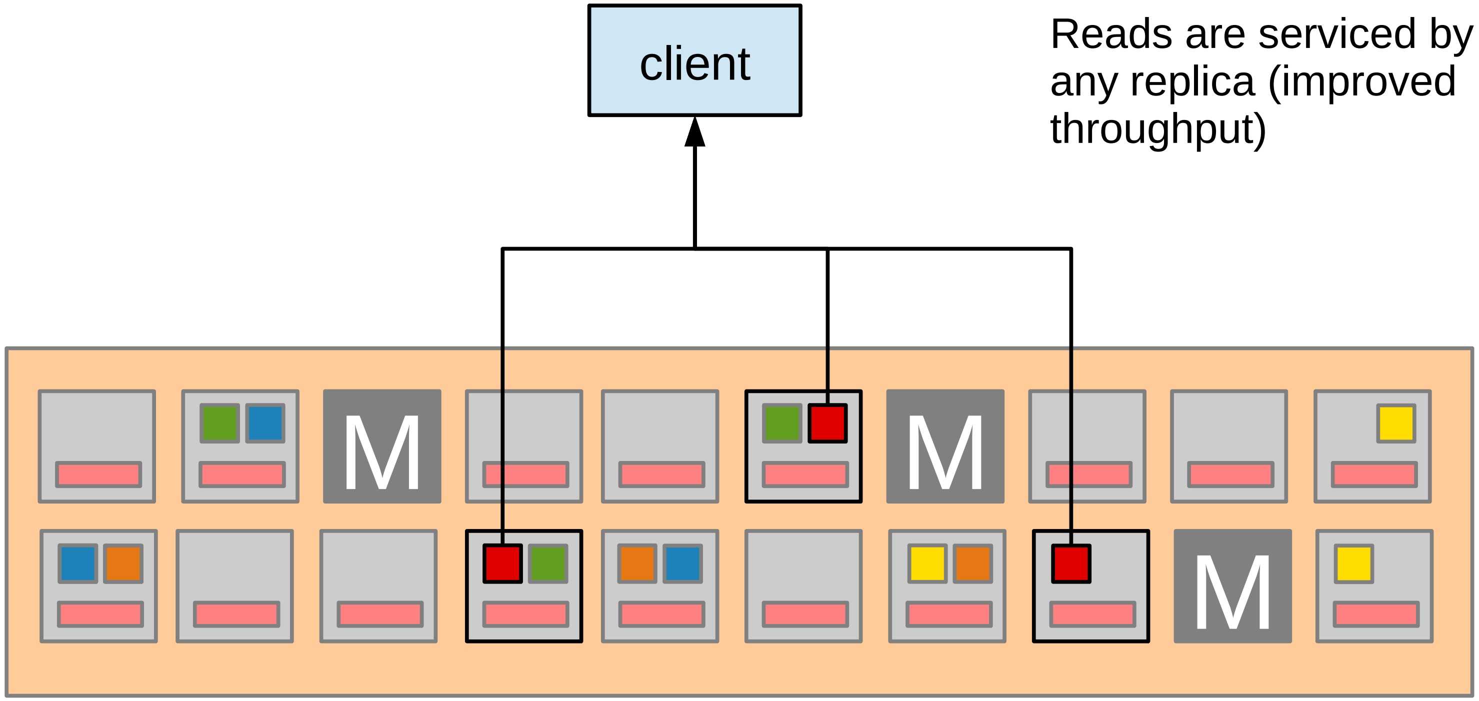


...Which You Can Write To



Writes go to one OSD, then propagate to other replicas

...And Read From



Reads are serviced by any replica (improved throughput)

Three Conceptual Components

- Pools
- Placement Groups
- CRUSH (deterministic, decentralised placement algorithm)

Pools

- Logical container for storage objects
- Have a set of parameters
  - Name, ID
  - Number of replicas or erasure coding settings
  - Number of placement groups
  - CRUSH rules
  - Owner
- Support certain operations
  - Create/read/write objects
  - Snapshot pool

Placement Groups (PGs)

- Help balance data across OSDs
- One PG typically spans several OSDs
- One OSD typically serves many PGs
- Tunable – read the docs! (50-100 per OSD)

CRUSH

- Controlled Replication Under Scalable Hashing
- MONs maintain CRUSH map
  - Physical topology (row, rack, host)
  - Rules for which OSDs to consider for certain pool/PG
- Clients understand CRUSH
  - This is the magic that removes bottlenecks

Once upon a time there was a Free and Open Source distributed storage solution named Ceph.

Sysadmins throughout the land needed to know the components that made up Ceph…

...because they wanted to deploy Software Defined Storage, instead of legacy storage arrays...

Legacy Storage Arrays

- Limits:
  - Tightly controlled environment
  - Limited scalability
  - Few options
    - Certain approved drives
    - Constrained disk slots
    - Fewer memory variations
    - Few networking choices
    - Fixed controller & CPU

- Benefits:
  - Reasonably easy to understand
  - Long-term experience, “gut instincts”
  - Somewhat deterministic in behaviour and pricing

Software Defined Storage (SDS)

- Limits:
  - ?

- Benefits:
  - Infinite scalability
  - Infinite adapability
  - Infinite choices
  - Infinite flexibility

To infinity… and beyond!”
– Buzz Lightyear

Software Defined Storage Properties

- Throughput
- Latency
- IOPS

- Capacity
- Density

- Availability
- Reliability

- **Cost**

Architecting SDS Systems

- Goals often conflict
  - Availability vs density
  - IOPS vs density
  - Everything vs cost
- Many hardware options
- Software topology offers many choices
- There is no “one size fits all”

Once upon a time there was a Free and Open Source distributed storage solution named Ceph.

Sysadmins throughout the land needed to know the components that made up Ceph…

...because they wanted to deploy Software Defined Storage, instead of legacy storage arrays…

...and they found they had many questions regarding configuration choices.

Network

- Choose the fastest you can afford
- Separate public and cluster networks
- Cluster network should be 2x public bandwidth
- Ethernet (1, 10, 40 GigE), or IPoIB

Storage Nodes

- CPU (number & speed of cores)
- Memory
- Storage controller (bandwidth, performance, cache size)
- SSDs for OSD journal (SSD to HDD ratio)
- HDDs (count, capacity, performance)

SSD Journals

- Accelerate bursts & random write IO
- Sustained writes that overflow journal degrade to HDD speed
- Help very little with read performance
- Are costly, and consume storage slots
- Use a large battery-backed cache on storage controller if not using SSDs

Hard Disk Parameters

- Capacity matters (highest density often not most cost effective)
- Reliability advantage of Enterprise drives typically marginal compared to cost
- High RPM increases IOPS & throughput, but also power consumption and cost

Redundancy

- Replication:
  - *n* exact full-size copies
  - Increase read performance (striping)
  - More copies lower throughput
  - Increased cluster network utilisation for writes
  - Rebuilds leverage multiple sources
  - Significant capacity impact

- Erasure coding:
  - Data split into *k* parts plus *m* redundancy codes
  - Better space efficiency
  - Higher CPU overhead
  - Significant CPU & cluster network impact, especially during rebuild
  - Cannot directly be used with block devices

Cache Tiering

- One pool acts as transparent write-back overlay for another
- Can flush on relative or absolute dirty levels, or age
- Additional configuration complexity, requires workload specific tuning
- Some downsides (no snapshots)
- Good way to combine advantages of replication & erasure coding

Adding More Nodes

- Capacity increases
- Total throughput increases
- IOPS increase
- Redundancy increases

- Latency unchanged
- Eventual network topology limitations
- Temporary impact during rebalancing

Adding More Disks to a Node

- Capacity increases
- Redundancy increases
- Throughput might increase
- IOPS might increase

- Internal node bandwidth consumed
- Higher CPU & memory load
- Cache contention
- Latency unchanged

Once upon a time there was a Free and Open Source distributed storage solution named Ceph.

Sysadmins throughout the land needed to know the components that made up Ceph…

...because they wanted to deploy Software Defined Storage, instead of legacy storage arrays…

...and they found they had many questions regarding configuration choices.

But learning which questions to ask enabled them to build sensible proofs-of-concept, which they scaled up and out...

How to Size a Ceph Cluster?

- Understand your workload
- Make a best guess, based on desirable properties & factors
- Build 10% pilot / proof of concept
- Refine until desired performance is achieved
- Scale up (most characteristics retained or even improved)
- It doesn't have to be perfect, you can always evolve it later

Once upon a time there was a Free and Open Source distributed storage solution named Ceph.

Sysadmins throughout the land needed to know the components that made up Ceph…

...because they wanted to deploy Software Defined Storage, instead of legacy storage arrays…

...and they found they had many questions regarding configuration choices.

But learning which questions to ask enabled them to build sensible proofs-of-concept, which they scaled up and out…

...and they all lived happily ever after.

Once upon a time there was a Free and Open Source distributed storage solution named Ceph.

Sysadmins throughout the land needed to know the components that made up Ceph…

...because they wanted to deploy Software Defined Storage, instead of legacy storage arrays…

...and they found they had many questions regarding configuration choices.

But learning which questions to ask enabled them to build sensible proofs-of-concept, which they scaled up and out…

...and they all lived happily ever after.