Principles for Storage Management

Running Ceph as a Predictable Product

About Me

Benedikt Bürk | Head of Professional Services

- 15+ years in Real-Time Communications (Enterprise Voice, Contact Center, Security)
- Expert in solving complex challenges from Pre-Sales, Design to Implementation
- Worked with Fortune 500 companies, major banks & insurance firms
- Now building a top-notch Ceph Service & Support organization at CLYSO
- *Outside the office:* Proud dad of two and an fisherman

CLYSO

What you will learn?

- Service Orientation
- Architecture for Change
- Data-Driven Operations
- Cost Transparency

„Storage only gets attention when it fails. When it works, nobody sees it - and that makes it hard to manage, justify or fund.“

The Invisibility Problem

- Nobody knows the real €/TB
- No shared SLOs → only assumptions
- Upgrades by panic instead of planning

The Three Views on Storage

Cost Center

Storage needs:

- Capital
- Power
- Floor Space

Tactics:

- Standard HW & SLA
- Track Costs per TB and IOPS
- Right-size pools (Replication vs EC)

Resource

As resource:

- Storage is shared
- Multi-Tenant
- Feeding compute & analytics

Tactics

- Enforce Quota
- Offer Tiers (Gold, Silver, Bronze) with documented latency and durability
- Monitor (publish) capacity, performance and SLO

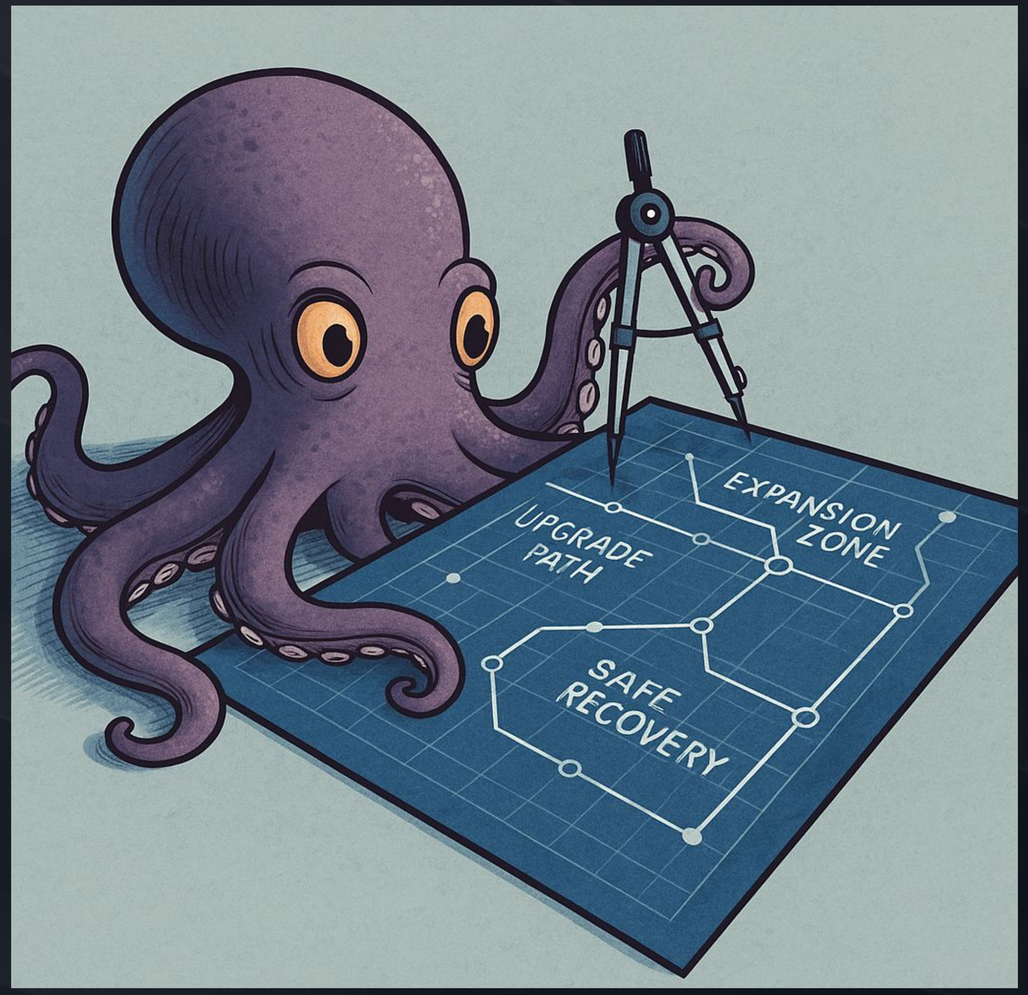
Strategic Weapon

As a strategic asset:

- Storage enables new products
- Storage enables faster time-to-market
- Storage for resilience & sovereignty

Tactics

- Co-Design with other teams for their use-cases
- S3 for decoupled applications
- Multi-Site and tiering to place data near compute (edge - core - cloud)



The Technical Solution… Needs a Management Solution!

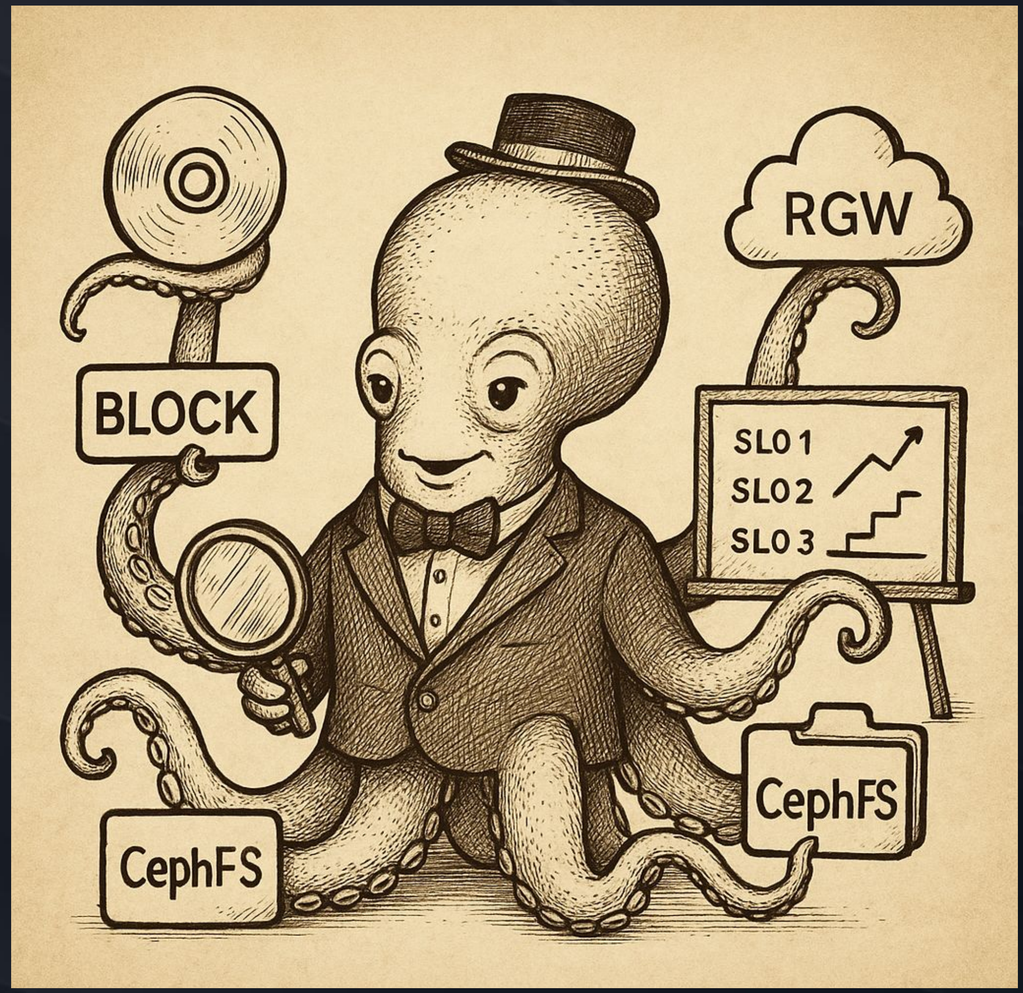
Margolies, *Octopus Car Wash*, 1981, Library of Congress, LC-MA05-691, hdl.loc.gov/loc.pnp/mrg.00691

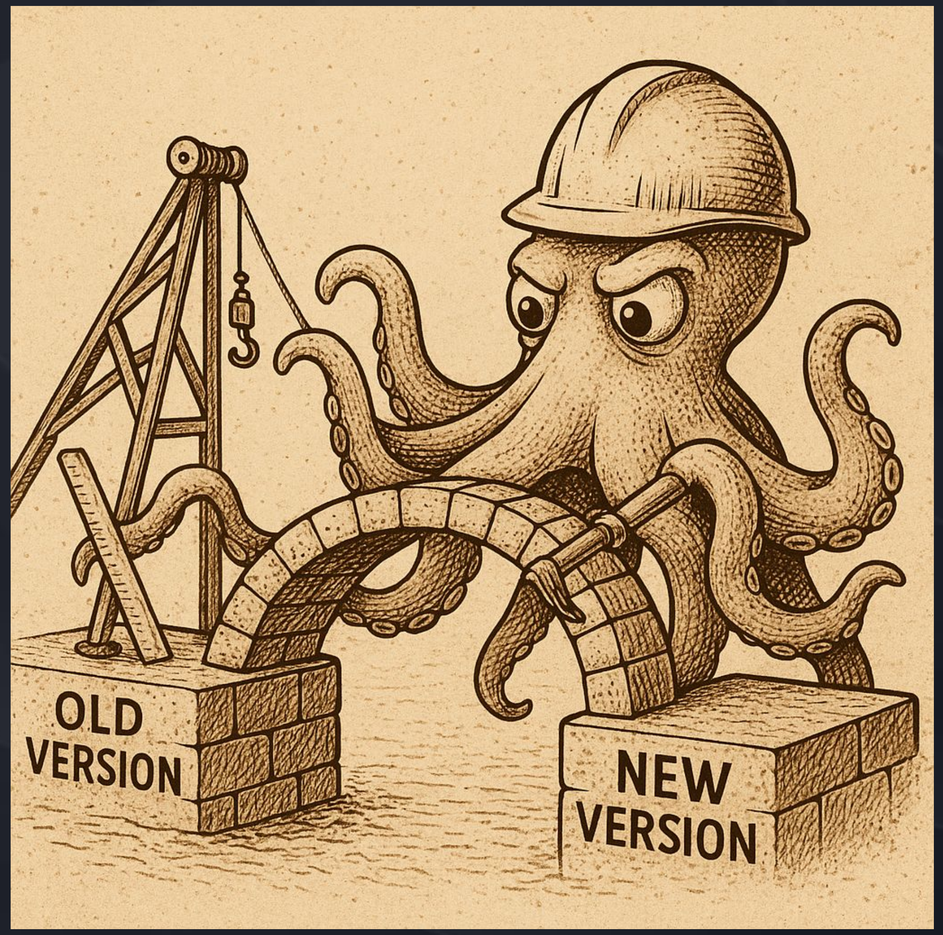
Service Orientation

Principle One

What does this mean?

- Know your (internal) customers
- Create tiers per interface (Block / CephFS / RGW) with…
- …Experience Promise (SLO) + boundary conditions (size, IOPS, Bandwidth, Object Size)



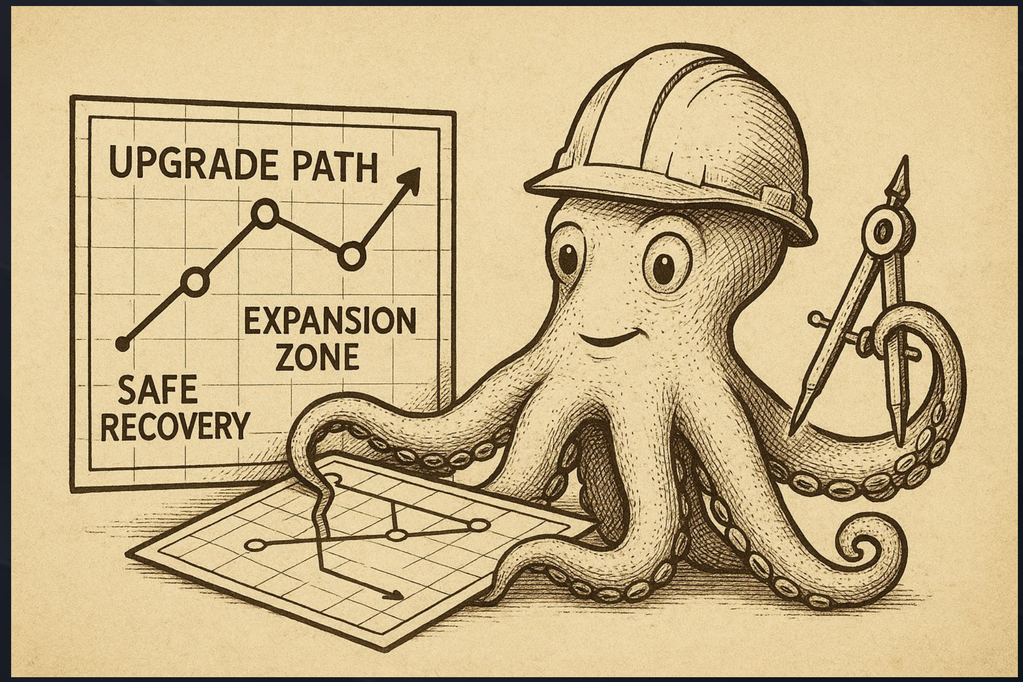


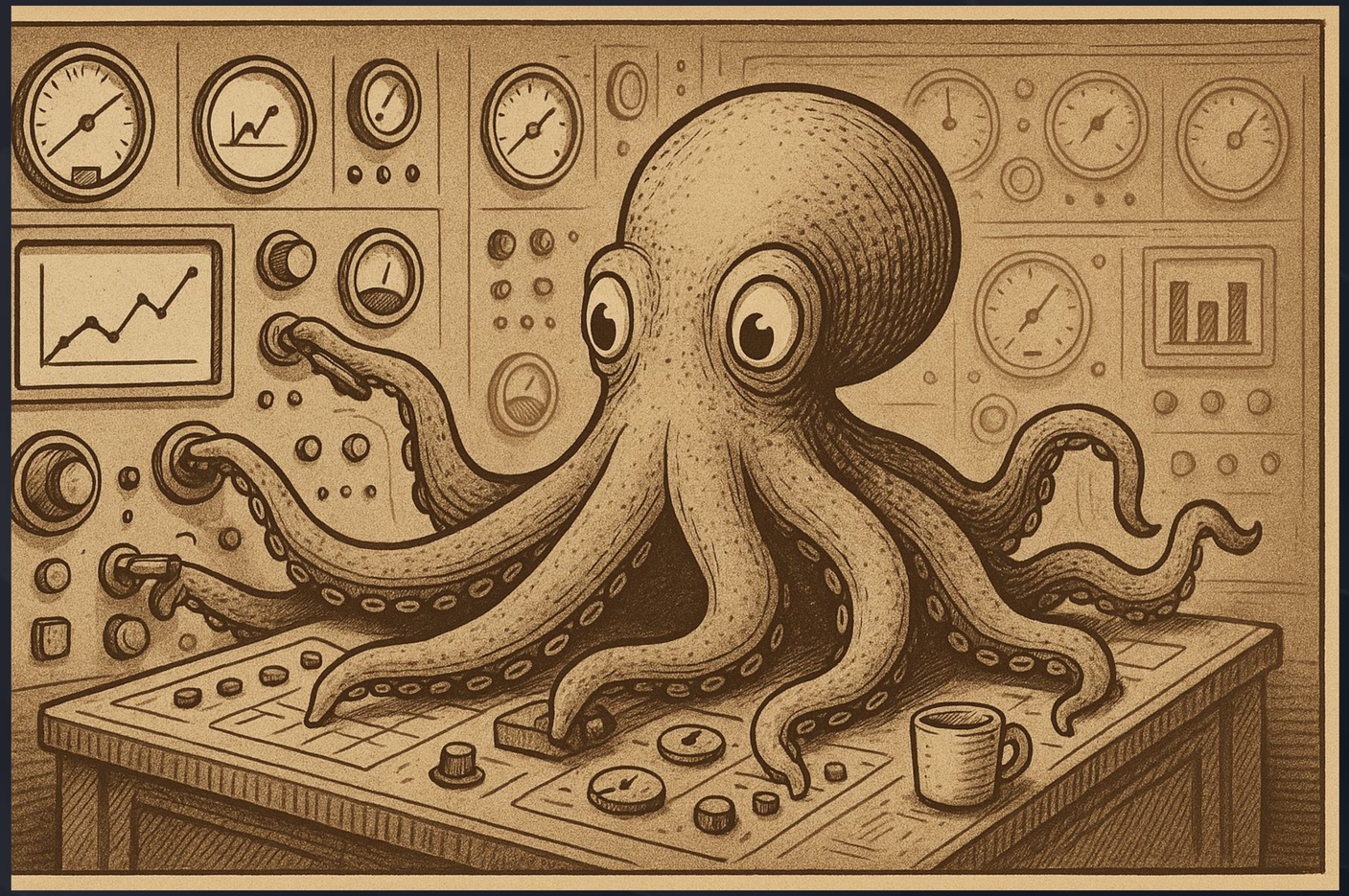
Architect for Change

Principle Two

What does this mean?

- Favor scale-out
- Controlled recovery
- Configuration-as-Code with Cephadm / Rook / Ansible
- Network separation
- Regular Upgrades
- Operation Runbooks
- Recovery tuning
- Headroom by design
- … Talk with your customers!



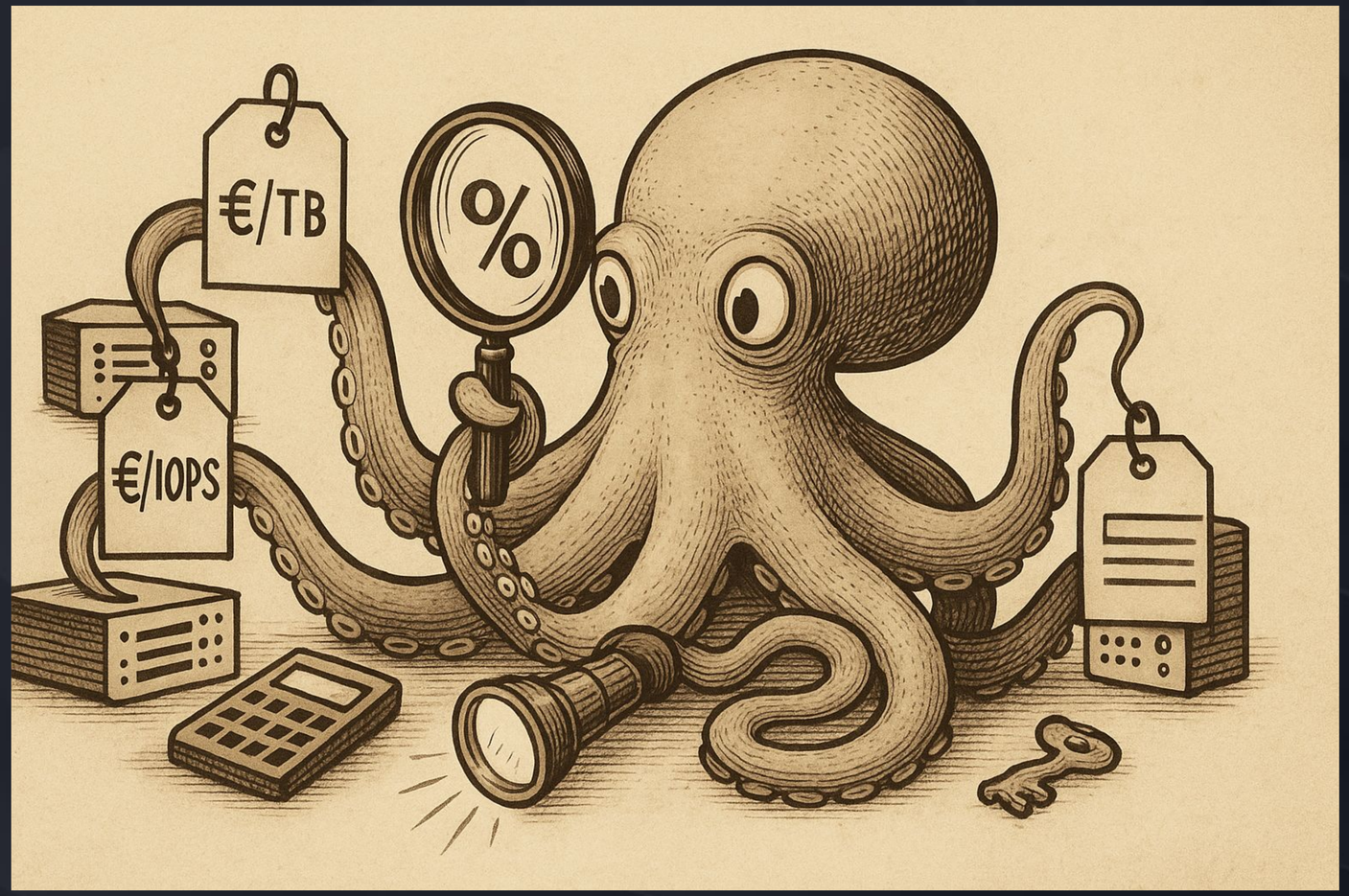


Data-Driven Operations

Principle Three

What does this mean?

- SLIs at the interfaces (RBD/CephFS/RGW): success + p95/p99 latency
- SLOs = target bands (time window + percentile)
- One telemetry, two views; publish the consumer KPIs
- Error budgets gate change; alert on sustained p99 breaches
- Forecast runway (growth + rebuild tax); expand early



Cost Transparency

Principle Four

What does this mean?

- Unit costs per tier (€/TB-mo; €/kIOPS-mo)
- Full cost = HW + power/cooling/space + network/support/ops + headroom
- Showback first - monthly statement (usage, requests, tiers/features)
- Prices signal design - Performance (€€€) for low-latency; Archive (€) for cold data
- One language across catalog, SLOs, rate card
- Trade-offs visible: cross-rack/site ↑cost/bandwidth ↔ better RPO/RTO

To Do List

Capacity - Performance - Durability

Pick two? Not anymore - design tiers to balance all three for a defined use case

- Capacity: €/TB/month target with headroom for safe rebuilds
- Performance: p95/p99 latency/throughput envelopes at the interface
- Durability: explicit policy (Replication vs EC) + failure domains (host/chassis/rack/site)

Outcome #1 - tier policy states the trade-offs

Outcome #2 - dashboards show whether we meet them

SLI & SLO Example

Tier	Focus	SLI	SLO
Object Standard	Standard S3 Storage	Availability p99 get latency (<1 MiB)	Availability > 99.9% per month p99 GET < 150 ms per month
Block Premium	Performance / Latency	p95 write latency Success Rate	p95 write < 5 ms per month Success rate > 99.99% per month
Object Archive	Cost, Long term storage	Availability retrieval time	Availability > 99.9% per month Retrieval completed with minutes (x minutes for y GB)

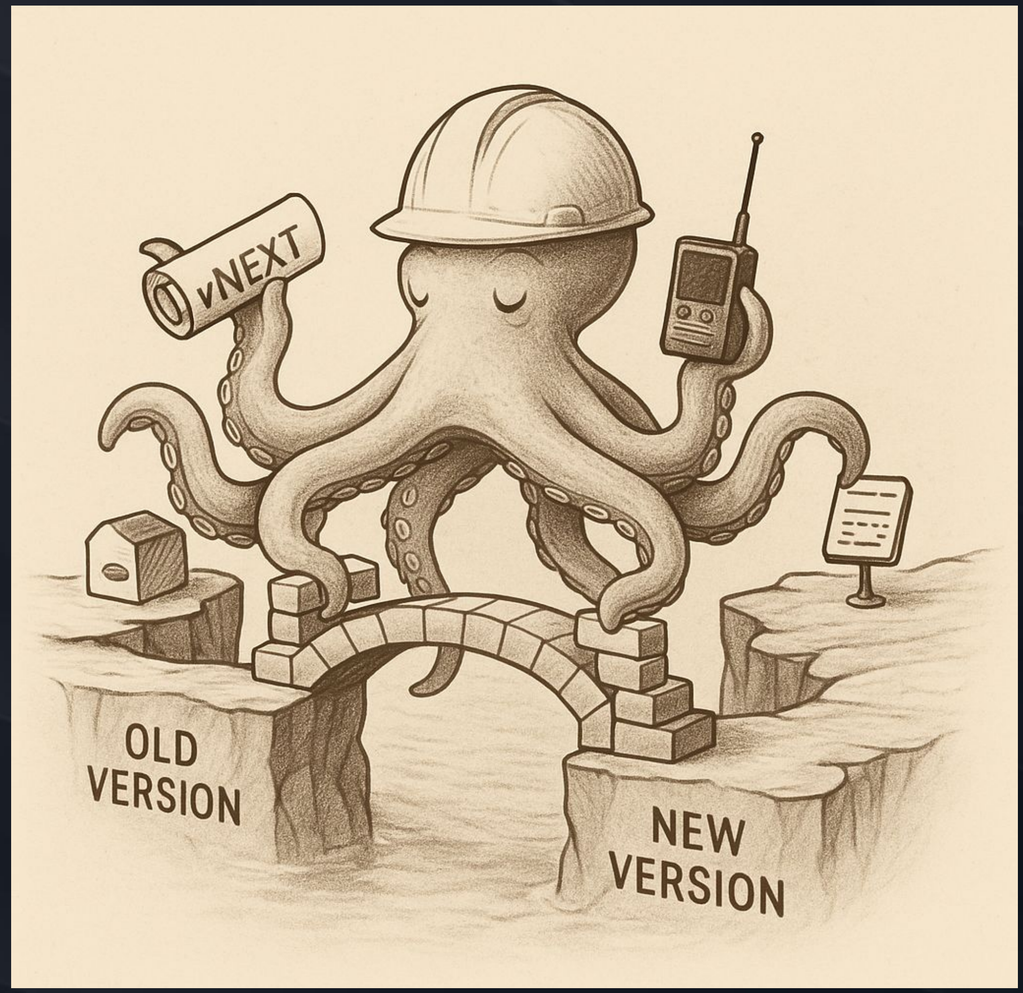
CL'SO

Operating Model: Cadence & Roles

- Weekly: backlog triage, SLO review; Monthly: showback & roadmap checkpoints
- Roles: Service Owner (accountable), SRE/Storage Ops (responsible), Finance/PM (consulted), Tenants (informed)
- Pipelines: change → pilot → staged rollout → GA
- Artifacts: catalog, SLO dashboard, runbooks, rate card, all in one repo
- Rule: no change without owner, rollback, stop-criteria, and comms plan

Change & Incident Management

- Change windows guided by profile; never roll into known spikes
- Pre-prod pilots; hard stop on error-budget breach; post-change verification gates
- Incident playbooks: classify (Severity), assign roles, timebox actions
- Stabilize first: throttle/backoff recovery; protect customer; then root-cause
- Aftercare: blameless review → fixes in code/policy/tests; learning goes back to catalog



From Control to Confidence

Predictable Business Value

- Publish price/perf/durability per tier → buyers know what they get and pay
- Showback drives behavior: data moves to cheaper tiers when visibility exists
- Stable SLOs reduce escalation load and unplanned work → higher engineering ROI
- Forecasting converts ‘surprises’ into planned expansions with vendor-neutral options

Outcome: fewer incidents, fewer emergency upgrades, better usage over time

Prove IT, Then Scale

- Start with a pilot tenant/pool per tier; publish consumer-view KPIs weekly
- Define exit criteria: N weeks within SLOs; 0 critical incidents; forecast runway ≥ X months
- Scale horizontally: add OSDs/nodes; re-weigh/CRUSH
- Automate the routine: node add, pool creation, quotas, QoS via cephadm/Rook/Ansible
- Adjust if SLOs or error budget slips

90-Day Action Plan

	Setup	Validation	Adoption
	Month 1	Month 2	Month 3
Service Orientation	Draft Catalogue with 3-5 Tiers	Roll out Pilot Tenants	Finalize & Publish Rate Card Version 1
Architecture for Change	Set Upgrade Cadence	Implement Quotas/QoS	Execute Upgrade
Data-Driven Ops	Baseline SLI	Monitor and Measure	Define Error Budget Policy & Post Mortem Process
Cost Transparency	Enable Showback (Consumtion)	Publish Consumer Dashboard	Use Rate Card to Guide consumption
Goals & Outcomes	Lay the foundation & establish basic measurements	Maximize the internal visibility	Established risk management & service model
Artefact	Initial Service Catalogue	Published SLO Status	Established Error-Budget Post-Mortem Process Final set of Deliverables

CLOUDSO

How this helps the Ceph Ecosystem

How this helps the Ceph Ecosystem

- Clear tier policies = clearer bug reports and reproducible tests
- Healthy clusters →paced recovery, fewer escalations in community channels
- Predictable upgrades drive broader, safer adoption of new releases
- Shared dashboards & runbooks become templates for others
- Better ops → happier users → stronger contributor base for Ceph

Thank You

...lets make storage boringly reliable!