

FinOps Fundamentals

Controlling cloud spend without slowing teams:
Inform, Optimize and Operate in practice

IT INFRASTRUCTURE & MANAGEMENT · FINANCE

Published November 2024 · cybertechinsights.com

EXECUTIVE SUMMARY

Cloud bills reflect thousands of small decisions

In a traditional data centre, spending decisions were made upfront by a small group: hardware was purchased, depreciated and largely fixed. In the cloud, every engineer who launches a service, chooses an instance size or leaves a test environment running is making a spending decision. The variable cost model is a strength — organisations pay for what they use — but without the right practices it produces bills that are hard to explain, forecast or control.

FinOps is the discipline that brings financial accountability to this model. The FinOps Foundation describes it as an operational framework and cultural practice that maximises the business value of cloud and technology spending through collaboration between engineering, finance and business teams. This whitepaper explains the framework's three phases — **Inform, Optimize and Operate** — and the practical techniques behind them: tagging and allocation, showback and chargeback, commitment discounts, rightsizing and unit economics.

Key takeaways

- FinOps is about maximising value, not simply cutting cost; engineering speed matters as much as savings.
- Accurate cost allocation through tagging and account structure is the foundation for everything else.
- Showback builds awareness quickly; chargeback adds accountability but needs trustworthy data first.
- Optimisation combines using less (usage) with paying less for what you use (rate).
- Unit economics — cost per customer, transaction or product — connect cloud spend to business outcomes.

FRAMEWORK

The FinOps Framework at a glance

The FinOps Foundation, part of the Linux Foundation, maintains an openly published FinOps Framework built from practitioner experience. It combines a set of principles, a lifecycle of phases, capability domains and a maturity model. The principles emphasise that teams need to collaborate, that business value drives technology decisions, that everyone takes ownership of their usage, that cost data should be accessible and timely, that FinOps should be enabled by a central team, and that organisations should take advantage of the variable cost model of the cloud.

Phase	Core question	Typical activities
Inform	Where is the money going, and who is responsible for it?	Cost allocation and tagging, showback reports, budgets and forecasts, benchmarking, anomaly detection.
Optimize	Where can we get more value for the same or less spend?	Rightsizing, removing idle resources, scheduling non-production, storage tiering, commitment discounts and rate negotiation.
Operate	How do we make good decisions continuously and at scale?	Policies and guardrails, automation, accountability for teams, governance cadence, training and culture.

The phases are a loop rather than a sequence: teams cycle through them continuously, and different parts of the organisation can be in different phases at the same time. The framework's **Crawl, Walk, Run** maturity approach encourages starting small, proving value and then expanding scope.

INFORM

Visibility and allocation

You cannot manage what you cannot attribute. The Inform phase gives every team an accurate view of its own spending, in language that both engineers and finance understand. The foundation is a consistent allocation strategy built from account or subscription structure, resource tags or labels, and agreed rules for shared costs.

Tag or label	Purpose	Example values
cost-centre	Maps spend to the finance structure for reporting and budgets.	CC-1042, marketing-ops
application or service	Groups resources that deliver one product or service.	checkout, data-platform
environment	Separates production from development and testing.	prod, staging, dev
owner	Identifies the accountable team or person.	payments-team
data-classification (optional)	Links cost to data handling requirements.	public, internal, confidential

- **Define a small mandatory set.** Four or five required tags applied consistently are worth more than twenty applied occasionally.
- **Enforce at creation.** Use infrastructure-as-code templates and cloud policy features to block or flag untagged resources.
- **Handle shared costs explicitly.** Support plans, networking, shared clusters and platform teams should be split using a documented rule — even, proportional or based on measured usage.
- **Report untagged spend.** Track the share of unallocated cost as a key metric and drive it down.

Showback and chargeback

	Showback	Chargeback
What it is	Teams see the cost of the resources they use, but budgets are not formally charged.	Costs are formally billed to the consuming team's budget through internal accounting.
When it fits	Early maturity, building awareness, data still being cleaned up.	Mature allocation, trusted data, organisations that already use internal cross-charging.
Main risk	Reports are ignored if no one is accountable for acting on them.	Disputes and distrust if allocation rules are unclear or data is inaccurate.

Most organisations should begin with showback, improve data quality, and only move to chargeback once teams accept the numbers as fair. Budgets, forecasts and automated anomaly alerts complete the Inform phase by turning reports into early warnings.

OPTIMIZE

Using less, and paying less for what you use

Optimisation has two levers. **Usage optimisation** reduces the amount of resource consumed. **Rate optimisation** reduces the price paid for each unit. The order matters: commit to discounts only after removing obvious waste, otherwise you may lock in payment for capacity you do not need.

Usage optimisation

- **Rightsizing:** compare provisioned capacity with observed utilisation over a representative period and move workloads to smaller or better-suited instance types. Validate performance with the owning team before changing.
- **Idle and orphaned resources:** unattached storage volumes, old snapshots, unused IP addresses, idle load balancers and forgotten test environments.
- **Scheduling:** switch off development and test environments outside working hours where they are not needed.
- **Storage tiering and lifecycle policies:** move infrequently accessed data to cheaper tiers and expire data that is no longer required.
- **Architecture choices:** autoscaling, managed services and serverless options can reduce cost for variable workloads, but should be assessed case by case.

Rate optimisation and commitment discounts

Cloud providers offer lower prices in exchange for a commitment to a level of usage or spend over a fixed term, typically one or three years. They go by different names — reserved instances, savings plans or committed use discounts — and differ in how flexible they are across instance families, regions and services.

- Commit against a stable baseline of usage, not peaks; leave variable demand on on-demand or spot capacity.
- Prefer more flexible commitment types where workloads are likely to change.
- Manage commitments centrally so that savings are shared and coverage is monitored.

- Track utilisation and coverage of commitments; unused commitments are pure waste.
- Use spot or preemptible capacity for fault-tolerant, interruptible workloads such as batch processing.
- Review enterprise agreements and marketplace purchases with procurement as part of the same process.

OPERATE

Making good decisions the default

The Operate phase turns one-off savings exercises into everyday practice. The aim is not for a central team to police spending, but to give engineering teams the data, guardrails and incentives to make cost-aware choices without slowing down.

- **Guardrails, not gates:** use policy-as-code to enforce tagging, restrict unapproved regions or very large instance types, and require approval only for genuinely exceptional spend.
- **Cost in the engineering workflow:** show estimated cost impact during design reviews and in infrastructure change pipelines, and surface team dashboards where engineers already work.
- **Regular cadence:** a short monthly review per product team, and a quarterly review with finance and leadership covering forecasts, commitments and unit costs.
- **Clear roles:** engineering owns usage, finance owns budgeting and forecasting, procurement owns rates and contracts, and leadership sets priorities and trade-offs. A small central FinOps function enables all of them.

Unit economics: connecting spend to value

Total cloud spend can rise for good reasons — more customers, new products, higher usage. Unit economics separate healthy growth from inefficiency by dividing cost by a business driver. Choose metrics that the business already tracks and that engineers can influence.

- Cost per active customer or per tenant for a SaaS product.
- Cost per transaction, order or API call for a platform service.
- Cost per gigabyte stored or processed for a data platform.
- Cost per environment or per developer for internal engineering platforms.

A falling unit cost alongside rising total spend is usually a sign of success, not a problem to be fixed.

BUYER GUIDANCE

Questions to ask FinOps tool vendors

- Which cloud providers, SaaS platforms and data sources can the tool ingest, and how quickly is data refreshed?
- Does it support the FinOps Open Cost and Usage Specification (FOCUS) for normalised billing data?
- How are shared costs and untagged resources allocated, and can we define our own rules?
- How are rightsizing and commitment recommendations calculated, and can engineers see the evidence behind them?
- Can it detect cost anomalies and route alerts to the owning team's existing channels?
- How does it calculate and report unit economics using our own business metrics?
- How is the tool itself priced — flat fee, per user, or a percentage of cloud spend?

ROADMAP

A phased FinOps roadmap

Phase	Focus	Outcome
1. Crawl: visibility	Consolidate billing data, define mandatory tags, agree account structure, publish showback reports to teams.	Each team can see its own spend.
2. Crawl: quick wins	Remove idle resources, schedule non-production, apply storage lifecycle policies, set budgets and anomaly alerts.	Early savings that build credibility.
3. Walk: optimisation	Rightsizing cycles with engineering owners; centrally managed commitment discounts on stable baselines.	Lower usage and better rates.
4. Walk: accountability	Monthly team reviews, forecasting with finance, guardrails as code, move towards chargeback where appropriate.	Cost-aware decisions become routine.
5. Run: value	Unit economics per product, cost in design reviews, continuous automation, regular maturity reassessment.	Spend linked to business outcomes.

Checklist: FinOps fundamentals

- An executive sponsor and a named FinOps lead are in place.
- A mandatory tag set is defined and enforced at resource creation.
- Unallocated spend is measured and reported monthly.
- Every team receives a showback report it understands.
- Budgets, forecasts and anomaly alerts are configured.
- Idle resources and non-production schedules are reviewed regularly.
- Commitment coverage and utilisation are tracked centrally.
- At least one unit cost metric is reported for each major product.

About Cyber Tech Insights: a B2B publication providing research and resources for technology and business decision-makers. This document is for general information only and does not constitute professional advice. © 2024 Cyber Tech Insights.