expedia group

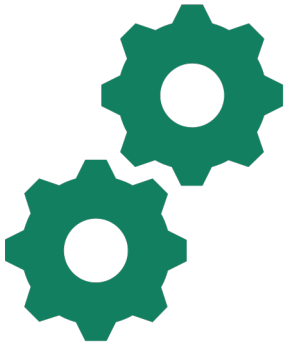
Cloud FinOps and Kubernetes Optimisation at Scale

Matt Callanan

Yow! Conferences Dec 2023

Outline

1. FinOps



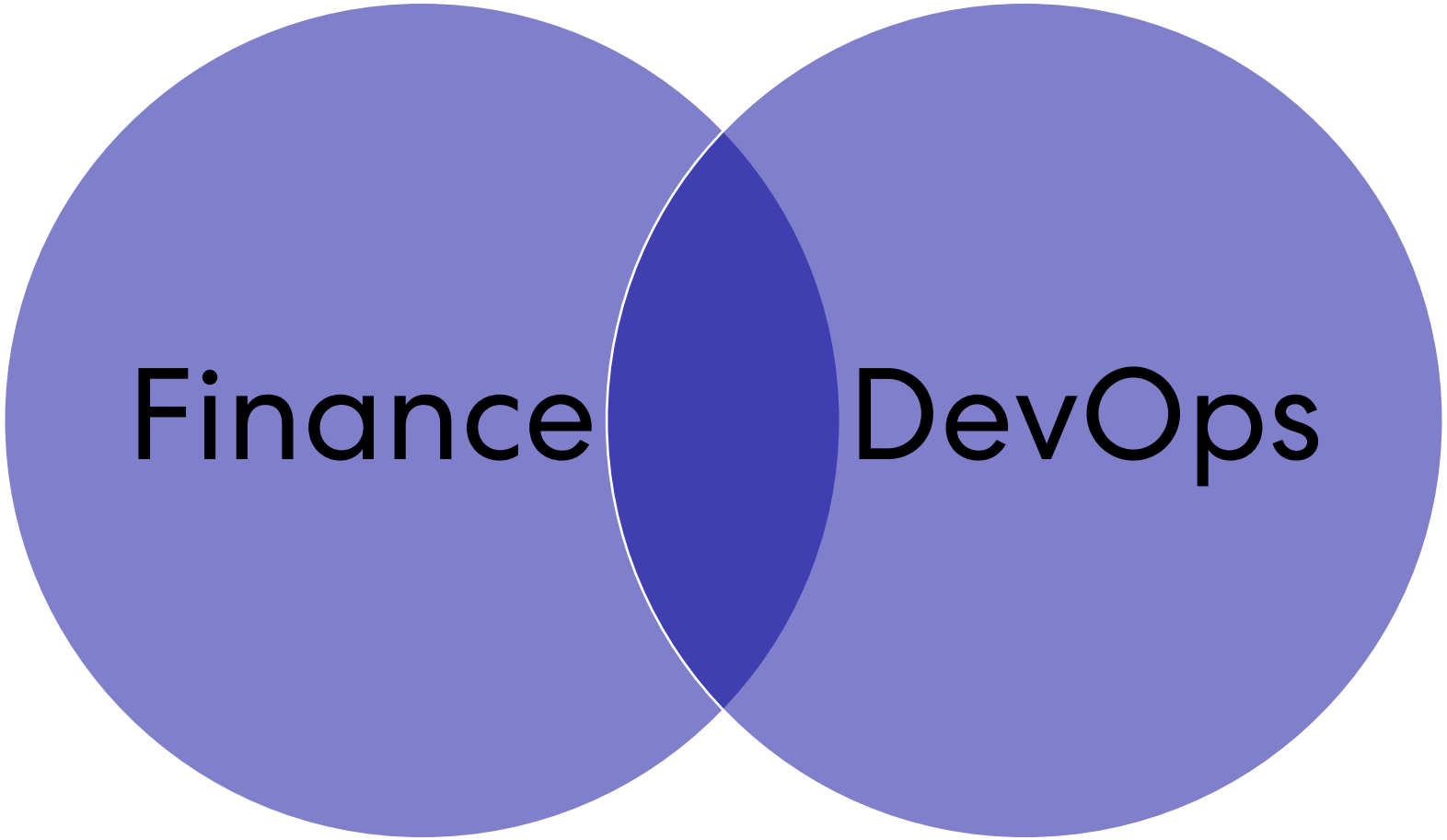
2. Optimising Kubernetes in the Cloud

Expedia Group Company Facts

- Founded in 1996
- 20+ brands
- 200+ sites
- 70+ countries
- 20,000+ applications
- 400+ EKS Kubernetes Clusters

What is FinOps?

What is FinOps?



https://www.finops.org/introduction/what-is-finops/

What is FinOps?

- Portmanteau of "Finance" and "DevOps"
- NOT "Financial Operations” (overloaded)
- Collaboration of Business and Engineering
- AKA
  - ”Cloud Financial Management/Engineering/Optimisation”
  - “Cloud Cost Management/Engineering/Optimisation "
- Getting the most **business value** out of your cloud spend
  - NOT (necessarily) reducing cloud spend

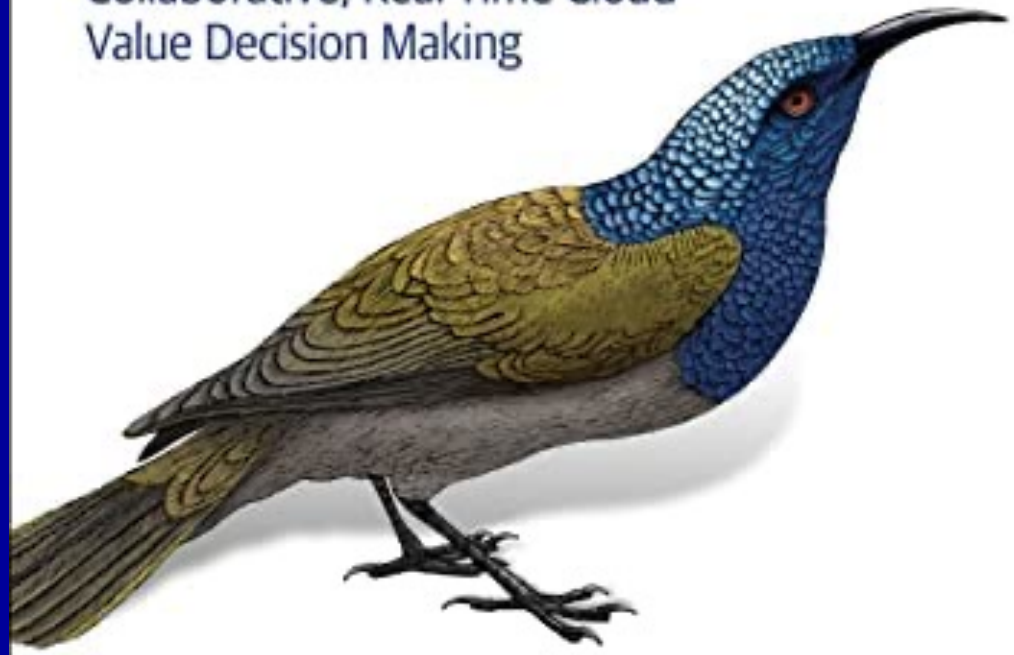
https://www.finops.org/introduction/what-is-finops/

O'REILLY

Second Edition

Cloud FinOps

Collaborative, Real-Time Cloud Value Decision Making



J.R. Storment & Mike Fuller

Why FinOps?

Cloud DevOps
Needs
Cloud FinOps

Without FinOps...

Scale up in cloud with DevOps. Opaque cost

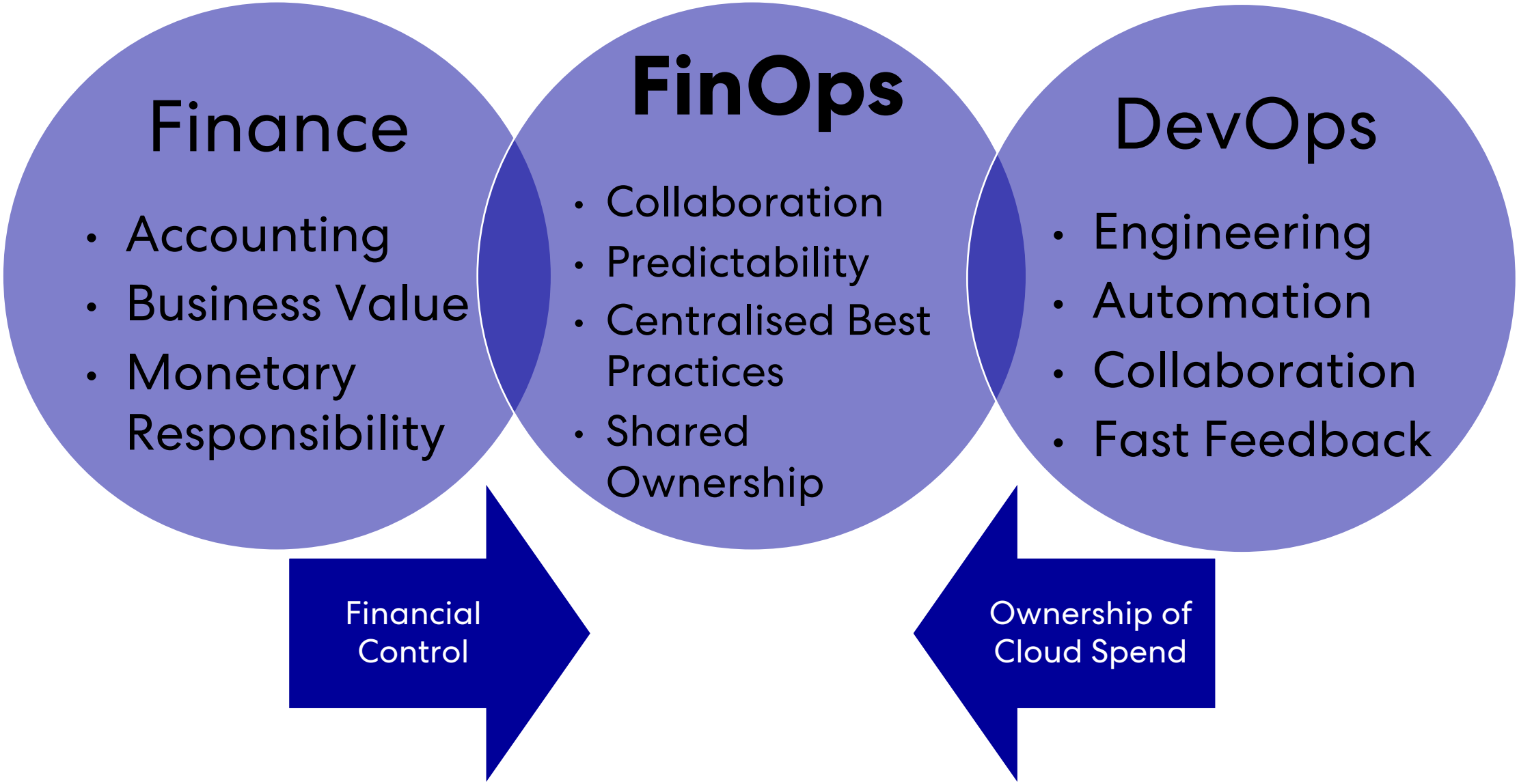
Faster time to market. Increased customer demand. Higher revenue

- 5-year CapEx planning model fails. Spending is no longer predictable/stable
- No visibility into cloud spend or business value
- Positive anomalies → Spike in customer demand → Spike in revenue
- Negative anomalies → Accidental spend → Cost blowouts
- Misinterpretation of anomalies → Kneejerk reaction

Scale back cloud due to unpredictable costs

Slower time to market. Customer dissatisfaction. Reduced revenue.

What is FinOps?



Comparing DevOps and FinOps

- Both started as a conference conversation
  - J.R. Storment, inspired by a DevSecOps talk
- FinOps Definition: A movement that focuses on
  - **Collaboration** between DevOps and Finance
  - Management of cloud spending (**lowering the unit economics of cloud**)
  - Increasing the **cost efficiency** and **profitability** of the cloud environment
- Well-defined by FinOps Foundation (part of Linux Foundation)
- FinOps has certification courses
- FinOps teams are encouraged

FinOps Foundation

Source: Cloud FinOps 2nd Ed, Chapters 1, 26

FinOps Basics

Cloud cost formula:

*cost = usage * rate*

Chargeback formula:

app_cost = cost / app_usage

FinOps Framework

FinOps is an evolving **cloud financial management discipline** and **cultural practice** that enables organizations to get **maximum business value** by helping engineering, finance & business teams to collaborate on data-driven spending decision

Maturity

Principles

- Teams need to collaborate
- Everyone takes ownership for their cloud usage
- A centralized team drives FinOps
- Reports should be accessible and timely
- Decisions are driven by business value of cloud
- Take advantage of the variable cost model of the cloud

Personas

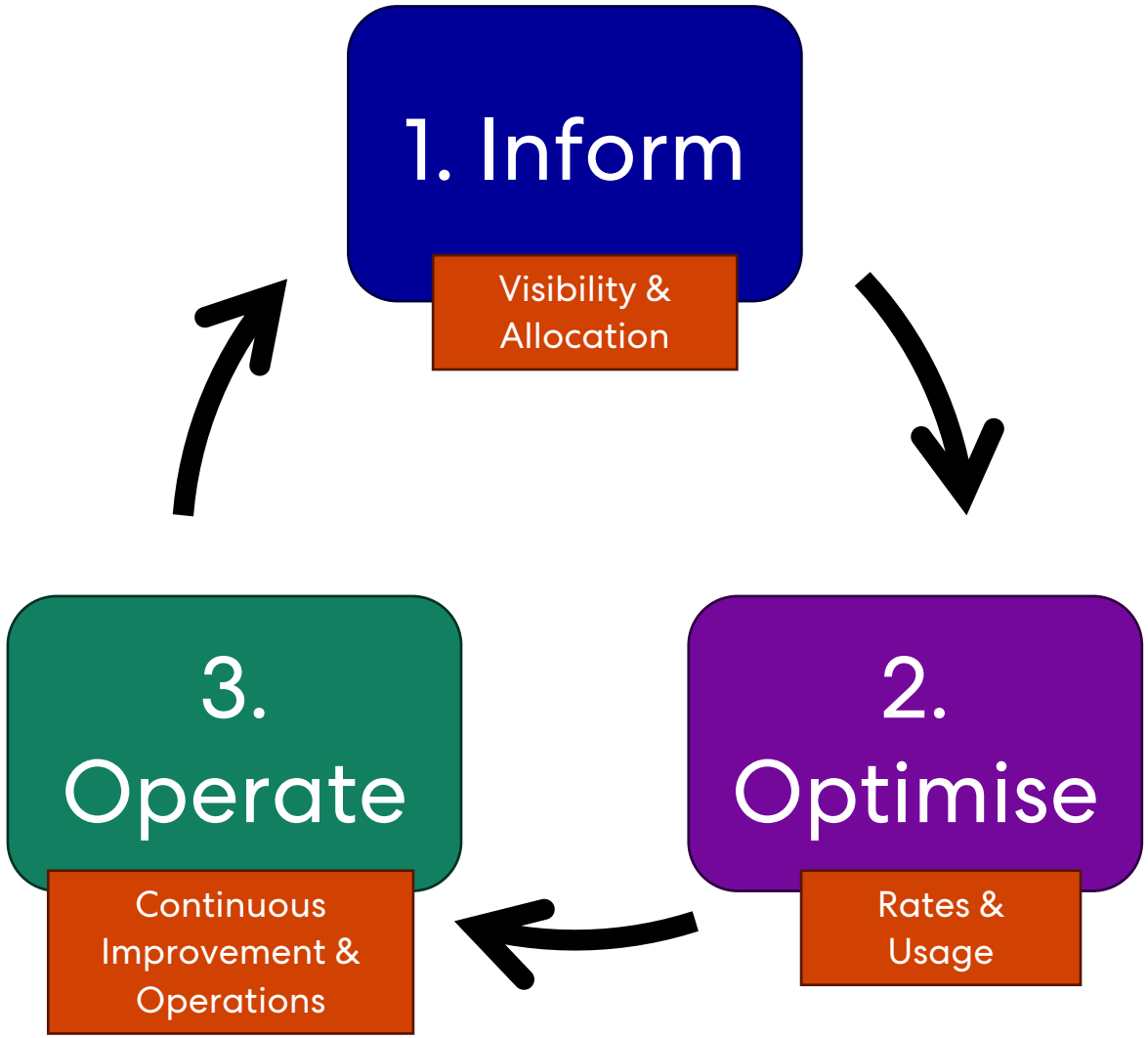
Phases

Domains

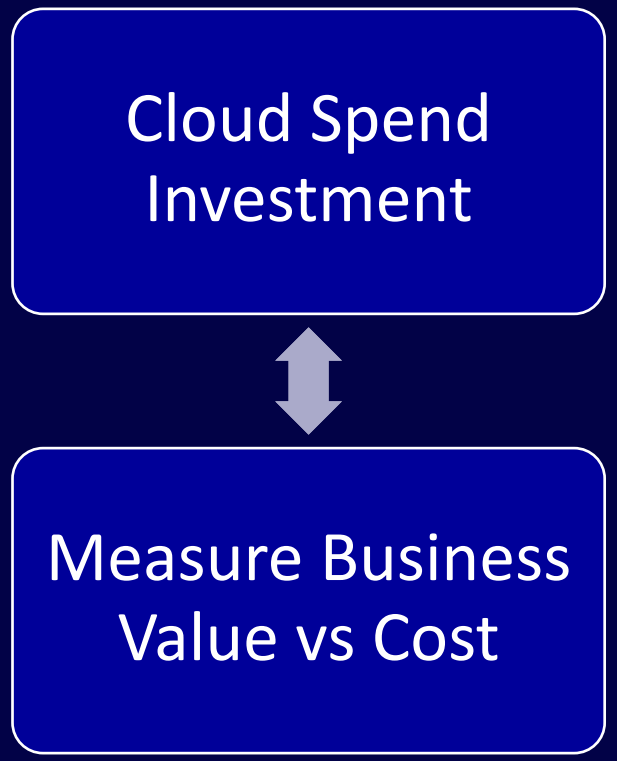
Understanding Cloud Usage and Cost	Performance Tracking & Benchmarking	Real-Time Decision Making	Cloud Rate Optimization	Cloud Usage Optimization	Organizational Alignment

https://www.finops.org/introduction/what-is-finops/

FinOps Lifecycle Phases



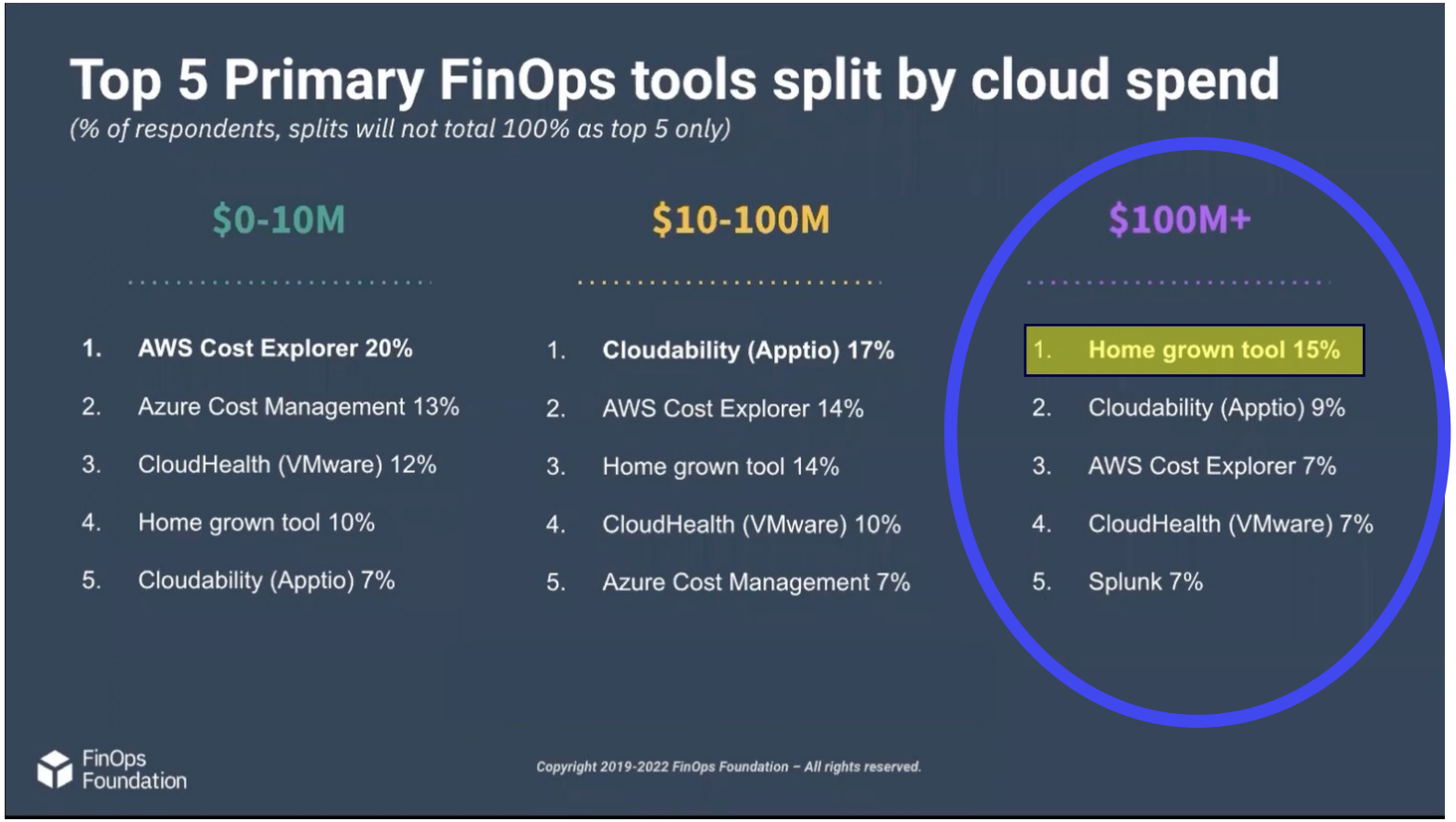
FinOps Nirvana



"Making decisions on the **value of the cloud spend**, no matter what method you use to **measure value** and **compare it to costs**"

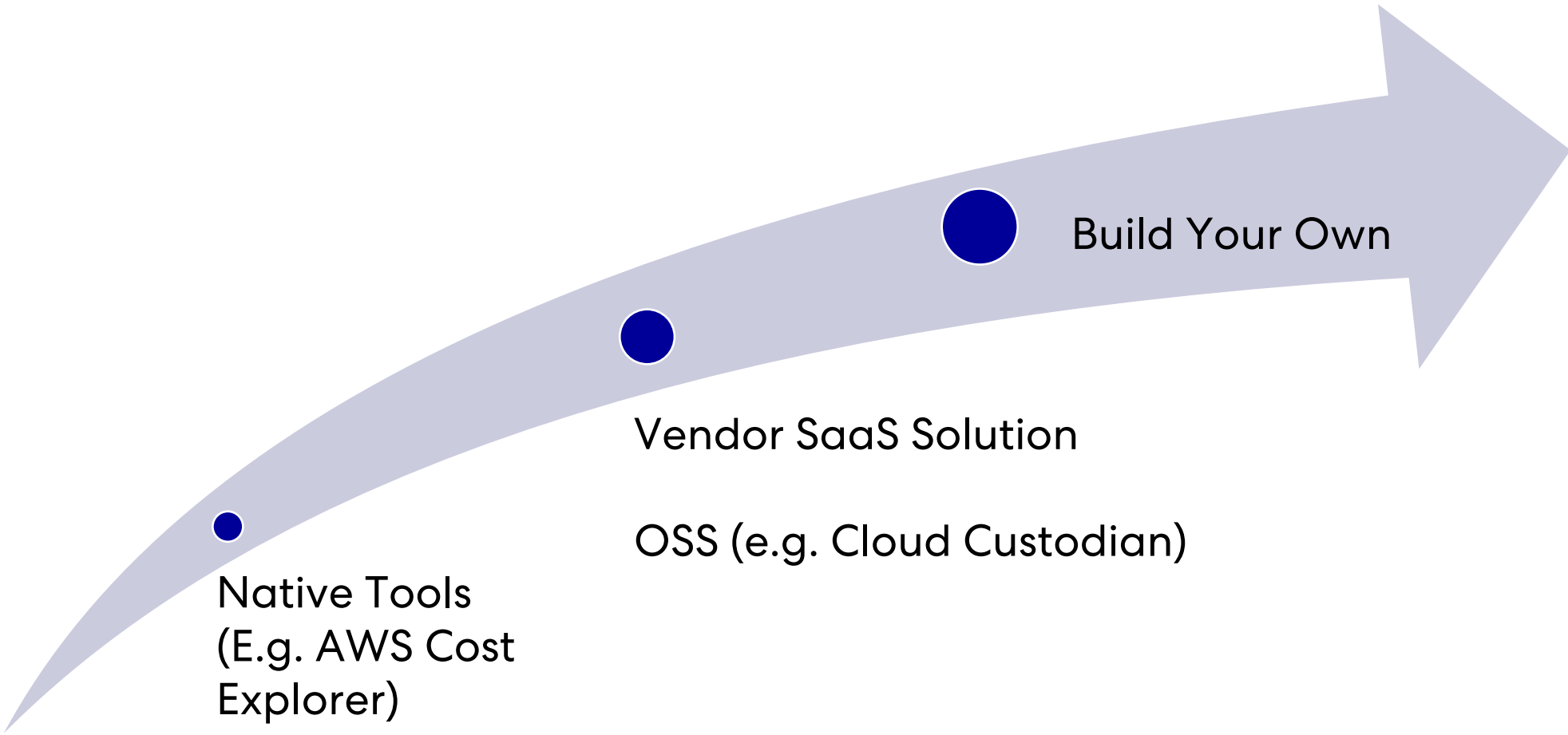
Source: Cloud FinOps 2nd Ed, Chapter 26

FinOps Platform: Buy vs Build?
Big Cloud Spenders Build their Own Tools



Source: “FinOps Foundation Panel Discussion: Which Tools Do You Use to Manage Enterprise Cloud Spend?”
https://www.youtube.com/watch?v=r6MiomkePpk

FinOps Platform: Buy vs Build?



FinOps Platform: Buy vs Build

Buy

- Features out of the box
- Access to industry SMEs
- Absorbs burden of changes and reconciliation

Build

- Align data and reporting with org requirements
- Freedom to Innovate
- Compliance (e.g. security)
- Cost Effectiveness at Scale

FinOps Platform: Building Your Own

- Go in eyes open
- A lot of data
- Need to keep up with changes
- Need to staff with dedicated engineers

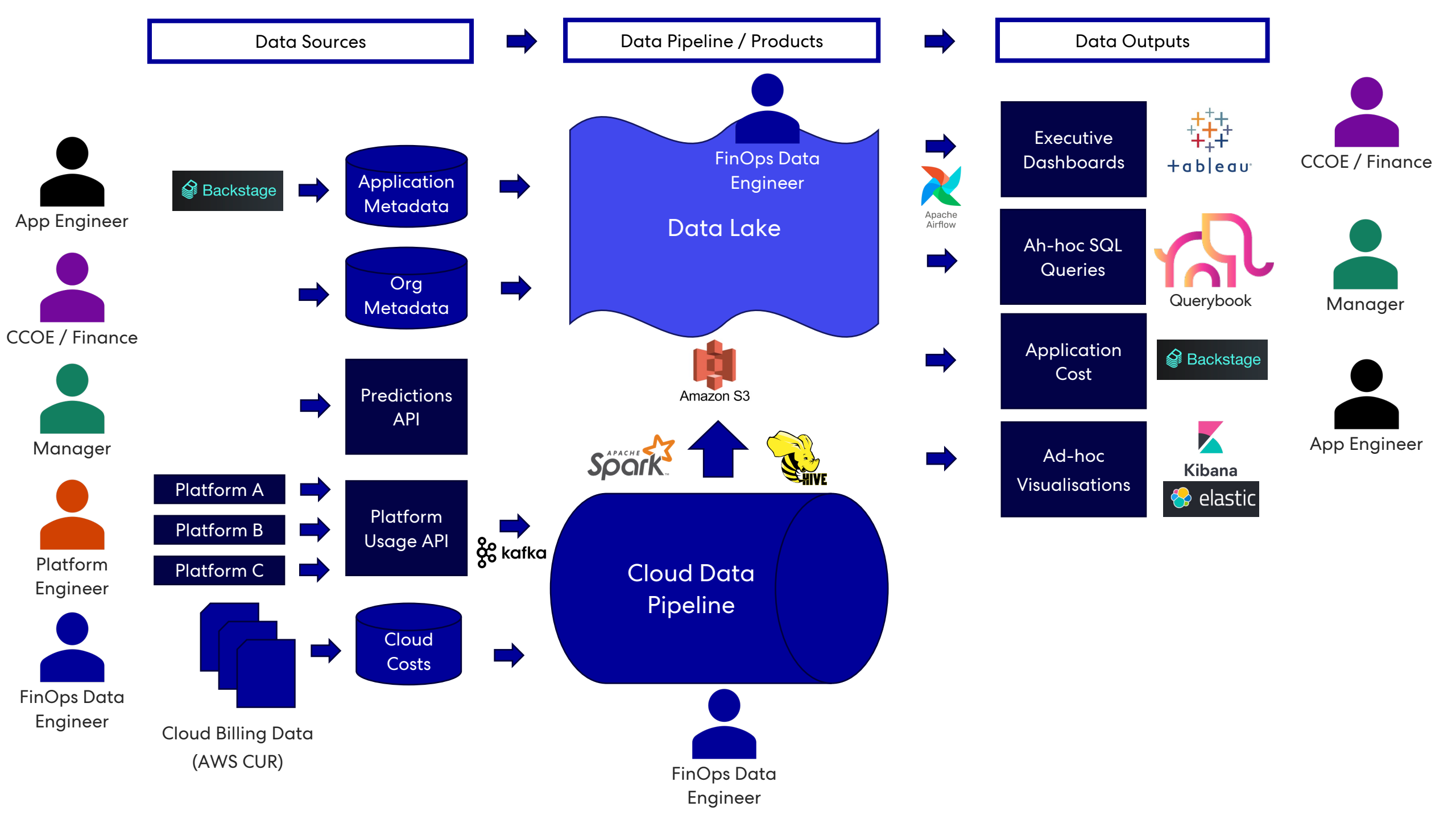
Expedia Group's FinOps Journey

FinOps Journey

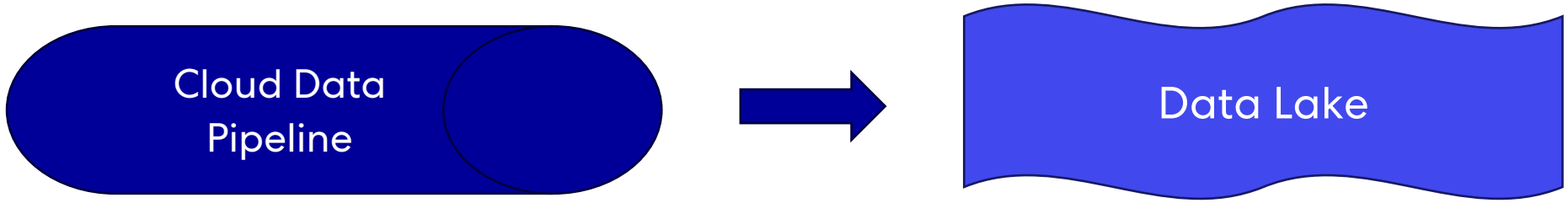
- Now: Unit Economics, Real-time Dashboards, Sub-platform Chargeback
- “What about SaaS licensing cost? Where is it coming from?” L&M Chargebacks
- “Whoa, you're spending a lot on EC2!” ”Actually, we’re running a platform”... Platform Chargebacks
- Costwatch Recommendations Engine: EC2, EBS, RDS, ECS Fargate
- Wait, reorg! Who owns this now? Application Metadata Governance. Decouple Team metadata
- Vendor SaaS to seed Visibility → Built Parallel Cost Platforms: DevOps vs Big Data
- Tagging policies enforced. Tags like “Application” and “Team” must be added to all resources
- Governance by Spreadsheet
- OMG cloud is expensive! Where is all this money going? Who created this stuff?
- 2014: Microservices! Cloud Native! Data Centres are Dead!

Expedia's FinOps Platform Features

- Savings Amortisation
- Fee Sharing (Tax & Premium Fees)
- Platform and L&M Chargeback
- Multi cloud
- Organisational Hierarchy Alignment
- Fix Cloud Provider Data Issues
  - Missing Instance Type
  - Non-taggable Resources
- Budgeting & Predictions; ML Forecasting
- Anomaly Reporting


Data Sources
Data Pipeline / Products
Data Outputs
App Engineer
CCOE / Finance
Manager
Platform Engineer
FinOps Data Engineer
Backstage
Application Metadata
Org Metadata
Predictions API
Platform A
Platform B
Platform C
Platform Usage API
Cloud Costs
Cloud Billing Data (AWS CUR)
kafka
FinOps Data Engineer
Data Lake
Amazon S3
Apache Spark
HIVE
Cloud Data Pipeline
FinOps Data Engineer
Apache Airflow
Executive Dashboards
Ah-hoc SQL Queries
Application Cost
Ad-hoc Visualisations
tableau
Querybook
Backstage
Kibana
elastic
CCOE / Finance
Manager
App Engineer

Pipeline Process



- Raw ingestion of cloud data into raw tables
- Fill missing data, e.g. instance type
- Apply custom pricing, add metadata, platform chargebacks
- Collapse line items into Application-level granularity
- Validate CUR cost against final cloud invoice

FinOps Platform Chargebacks

- Chargeback is just:
  - *app_cost = cost / app_usage*
- Not that simple
- Challenge:
  - Daily cost fluctuations in cluster resources
  - Complicates visualization and estimation
  - E.g. EC2 charges fixed hourly rates
    - Regardless of how many nodes/VMs are in AWS data centres
- Solution:
  - Smooth out with consistent unit rate for resources (e.g. CPU & Memory)

Platform Chargebacks Pre-Requisites

CPU Weight is CPU:Memory ratio

- AWS use 1:9 for Fargate
- Cast.ai use 1:7.3 (based on GCP)

- Platform Breakdown
  - **E.g. CPU 50%, 50% Memory**
  - E.g. CPU 30%, Memory 30%, GPU 20%, Storage 10%, Network 10%
- Platform Compute Unit (PCU)
  - Tenant's proportion of total capacity
  - `= (vCPUs * CPU Weight) + (Memory GiBs)`
- Unit Rate
  - `= Total Platform Cost / Total PCUs`
- Tenant Chargeback per Hour
  - `= PCUs * Unit Rate`

Source: https://docs.aws.amazon.com/cur/latest/userguide/example-split-cost-allocation-data.html

Source: https://cast.ai/blog/how-to-calculate-cpu-vs-memory-costs-for-more-accurate-k8s-cost-monitoring/

Kubernetes Optimisation

Expedia Group's Containerisation Journey

Containerisation Journey

2015-2019

- Amazon ECS
- Mesos / Nomad
- Kubernetes (on EC2)

2020

- RCP: Runtime Compute Platform (EKS)
  - Paved Road
  - Integration
    - Secrets
    - Observability: Splunk, DataDog
    - Service Mesh: Istio

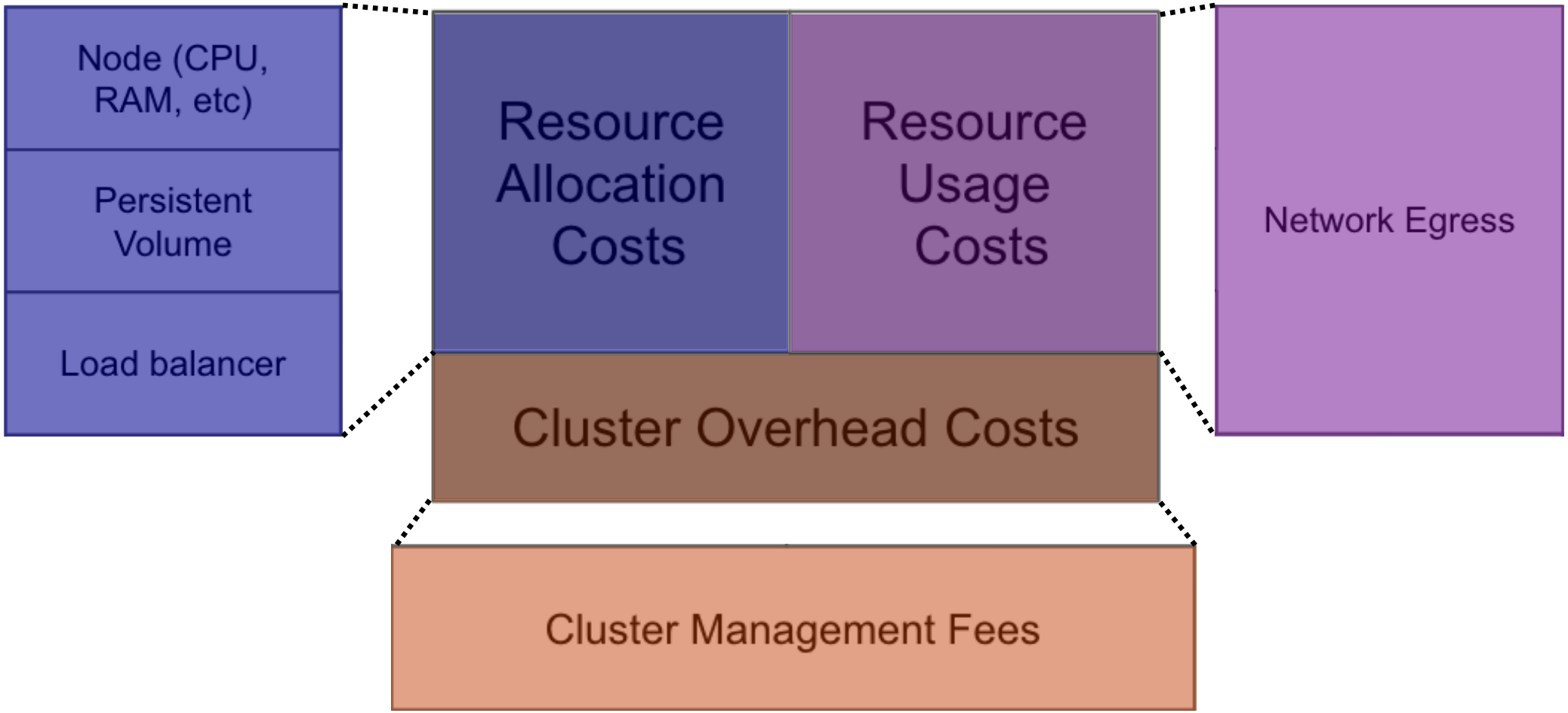
2022

- Cluster Optimisation
  - Cluster Autoscaler
  - Karpenter

Now

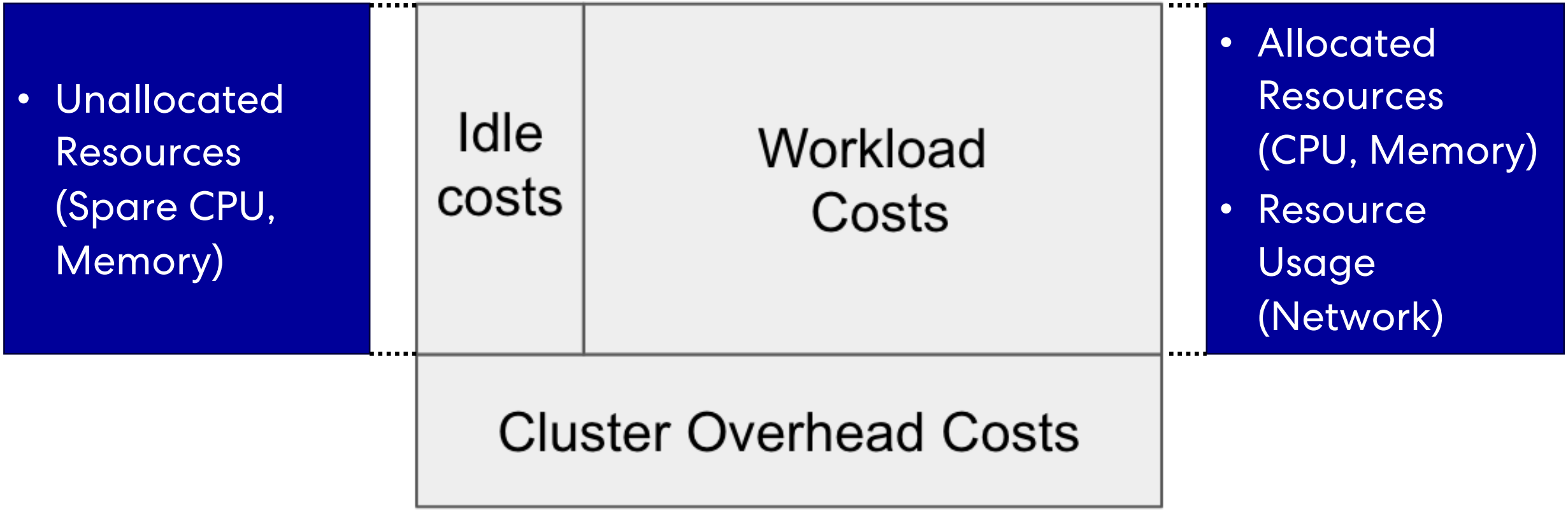
- Workload Optimisation
  - Autonomous Rightsizing

Kubernetes Platform Chargeback



Source: https://www.opencost.io/docs/specification

Kubernetes Platform Chargeback

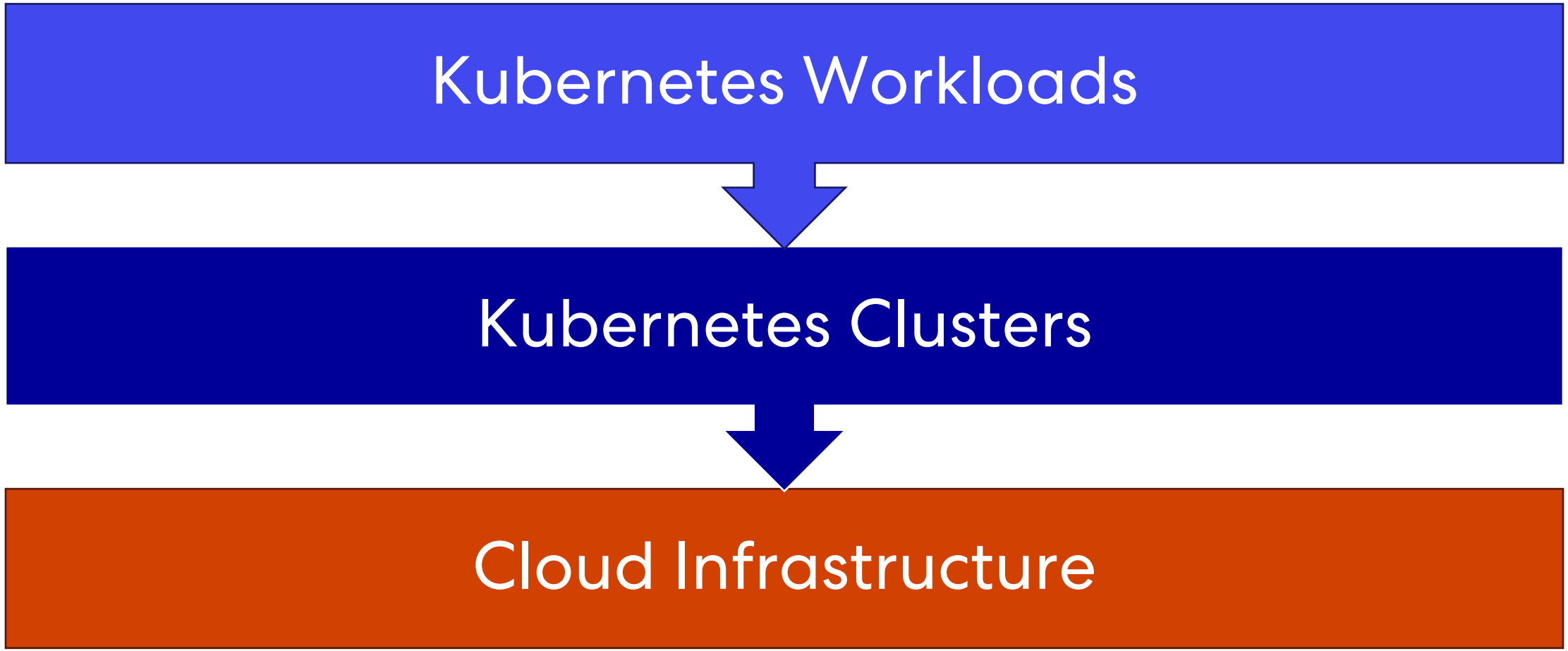


Source: https://www.opencost.io/docs/specification

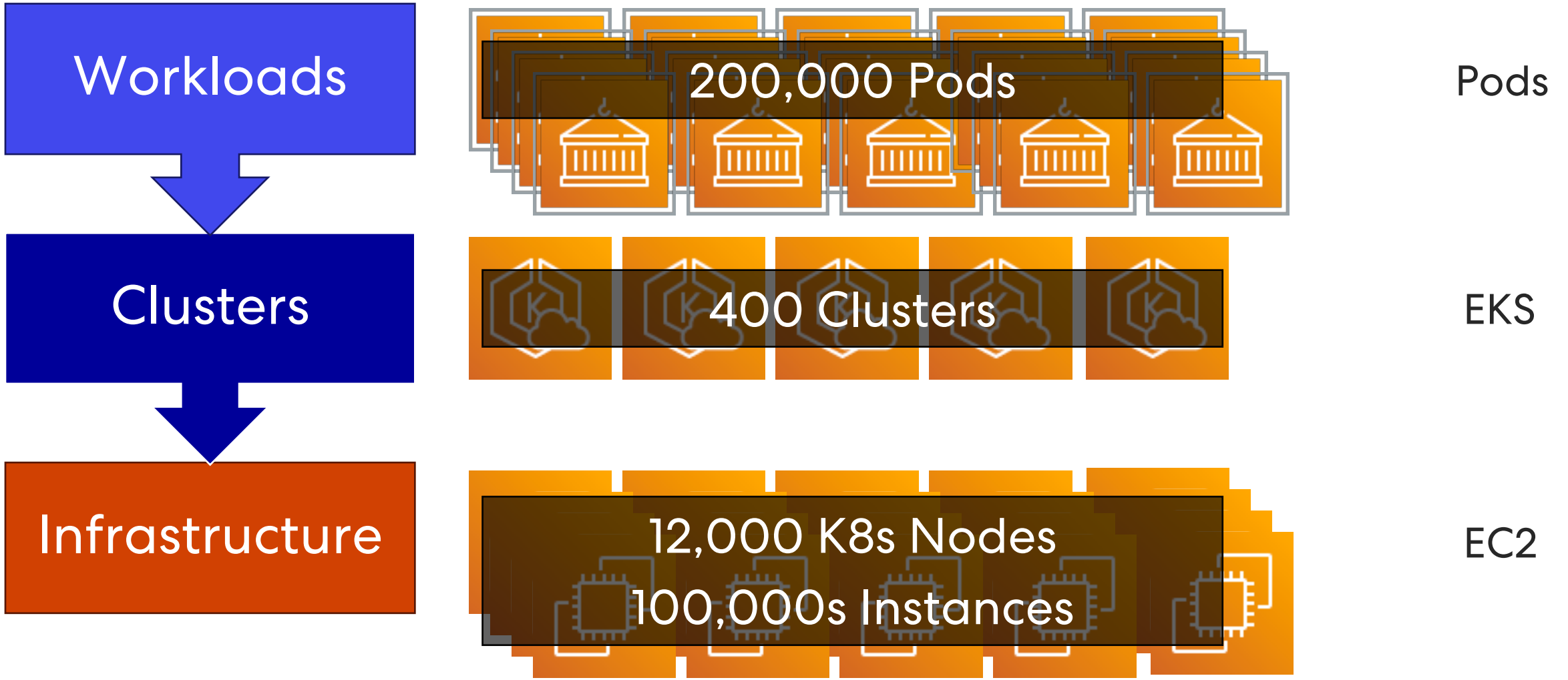
Kubernetes Platform Chargeback – Shared Costs

- Shared Costs:
  - Cluster Idle Costs
  - Overhead Costs (EKS fees)
  - Shared Workload Costs (e.g. kube-system, istio, datadog pods)
- Who pays for Shared Costs?
- Distribution Methods
  - Platform team absorbs the cost
  - Shared uniformly across tenants (regardless of usage)
  - Charged proportionate to tenant's consumption of Cluster Asset costs *

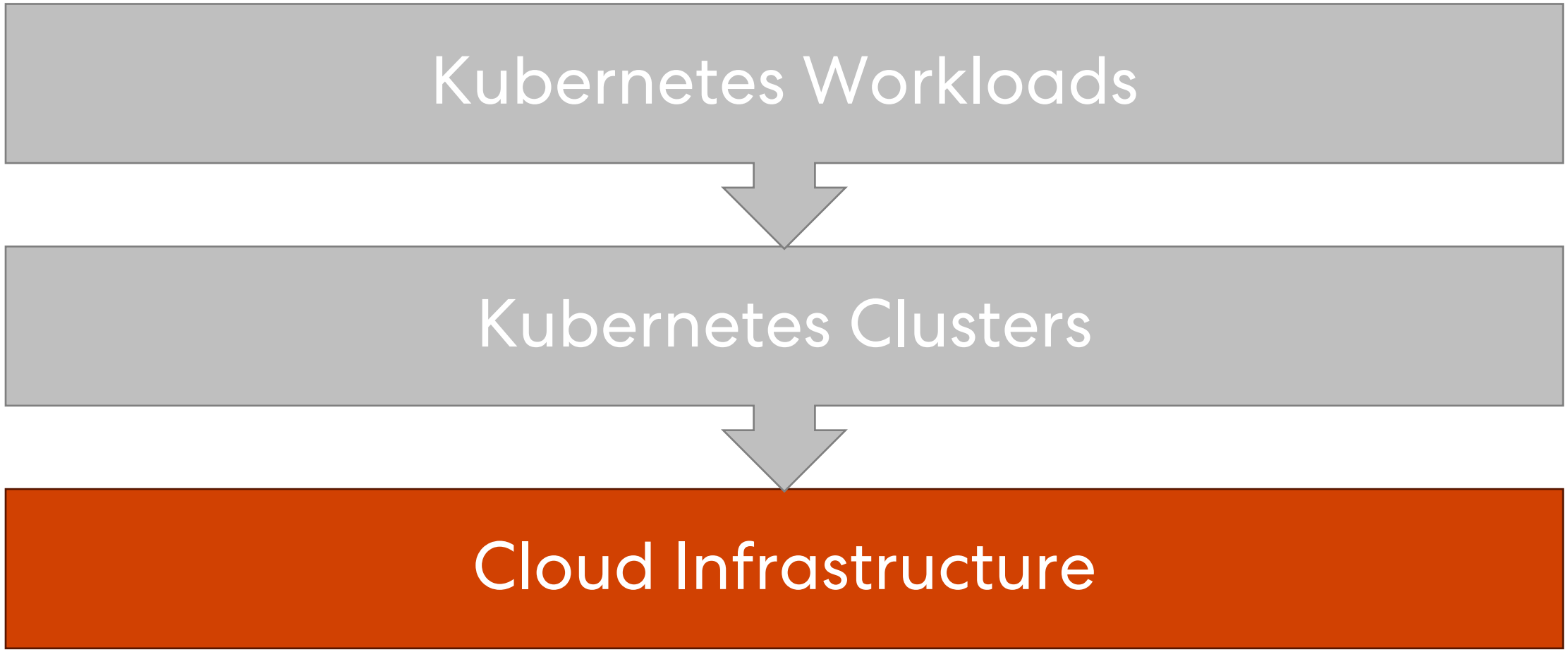
Kubernetes Optimisation

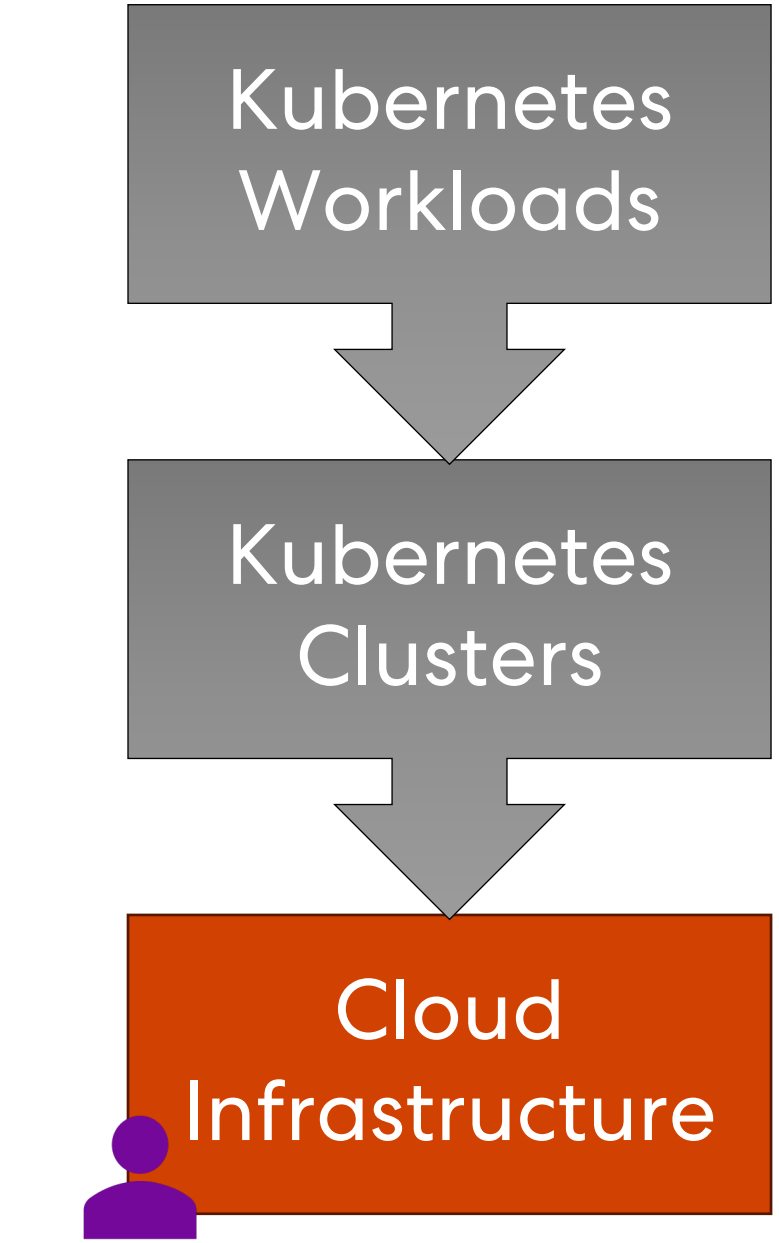


Kubernetes Optimisation



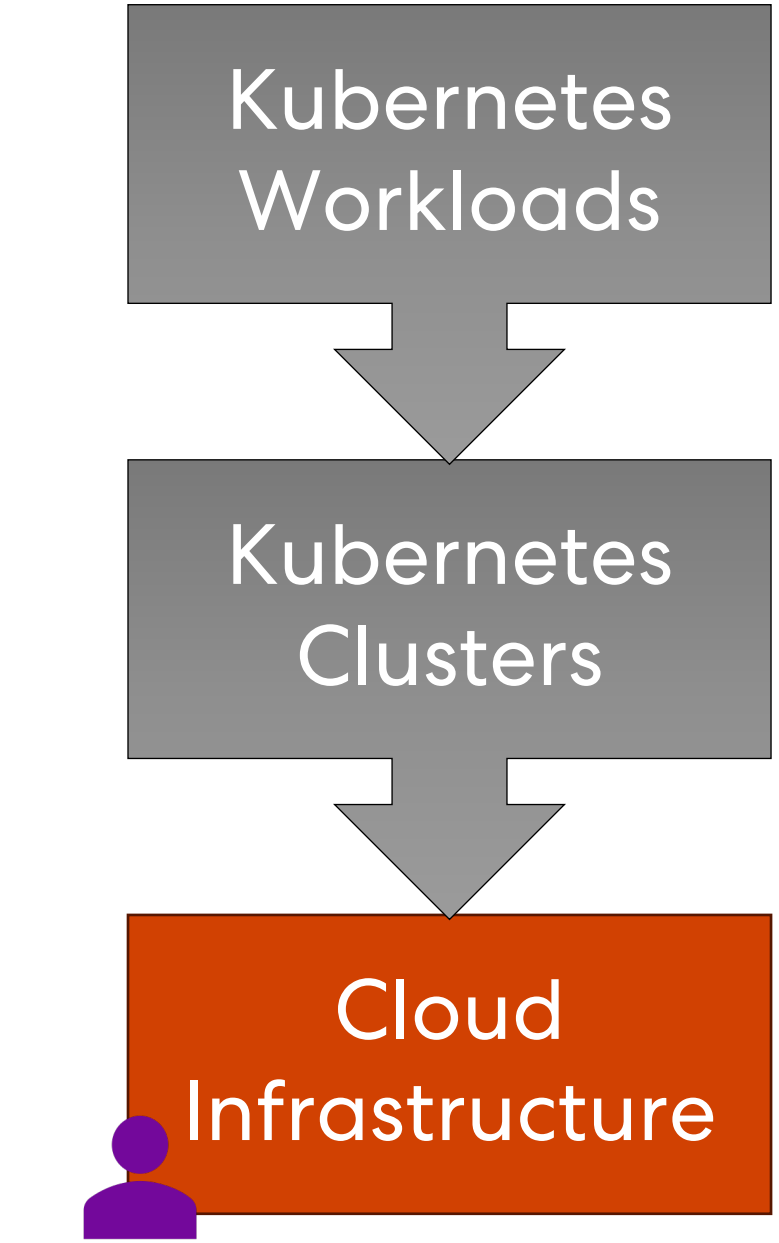
Infrastructure Optimisation





Cloud Infrastructure Optimisation

- Personas
  - FinOps
- Responsibilities
  - Forecasting, Budgeting
  - Work with all platform teams on aligning commitments with projected usage



Cloud Infrastructure Optimisation

- *cost = usage * **rate***
- Rate Negotiations – FinOps Team
  - Savings Plans
  - Convertible Reserved Instances
  - Enterprise Discounts (EDP / PPA)

How to measure effectiveness of FinOps Negotiations?

"Effective Savings Rate"

Effective Savings Rate

$$\textit{Effective Savings Rate (ESR)} = 1 - \left(\frac{\textit{Actual Spend with Discounts}}{\textit{On-Demand Equivalent (ODE) Spend}} \right)$$

Actual Spend with Discounts:

- What you paid with RIs and Savings Plans, amortizing any upfront charges

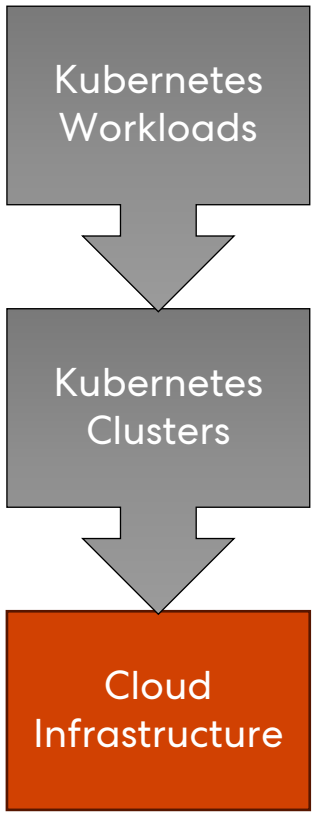
On-Demand Equivalent (ODE) Spend:

- What you would have paid if no discounts were applied

A negative ESR = low utilization wasting the discount benefit

https://www.prosperops.com/blog/effective-savings-rate/

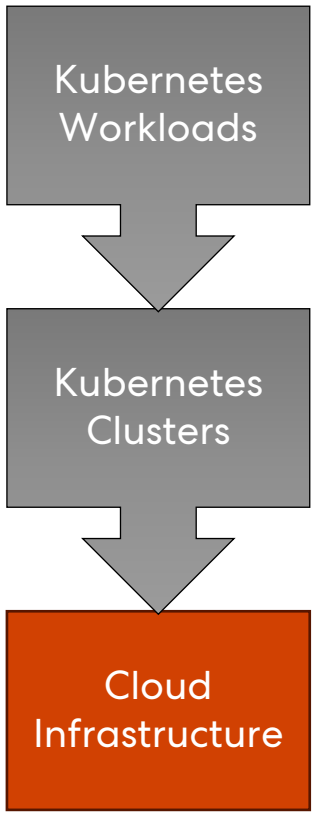
Cloud Infrastructure Optimisation – Purchasing Future Options

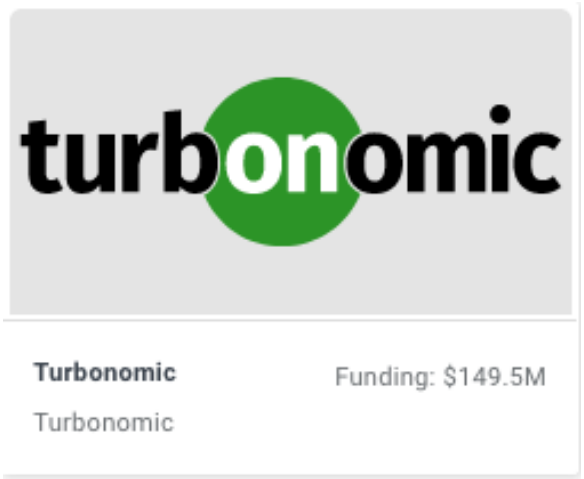


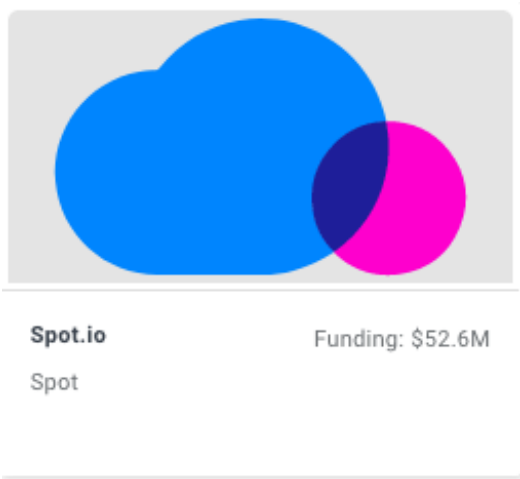
- Automated CRI purchasing
  - ML approach
  - Dynamically repurchases Convertible Reserved Instance commitments to match actual usage

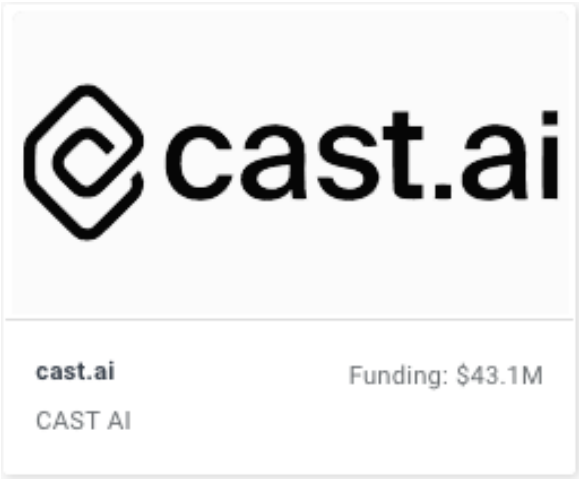
Source: https://www.getapp.com/it-management-software/a/prosperops

Cloud Infrastructure Optimisation – Consolidation and Spot Purchasing

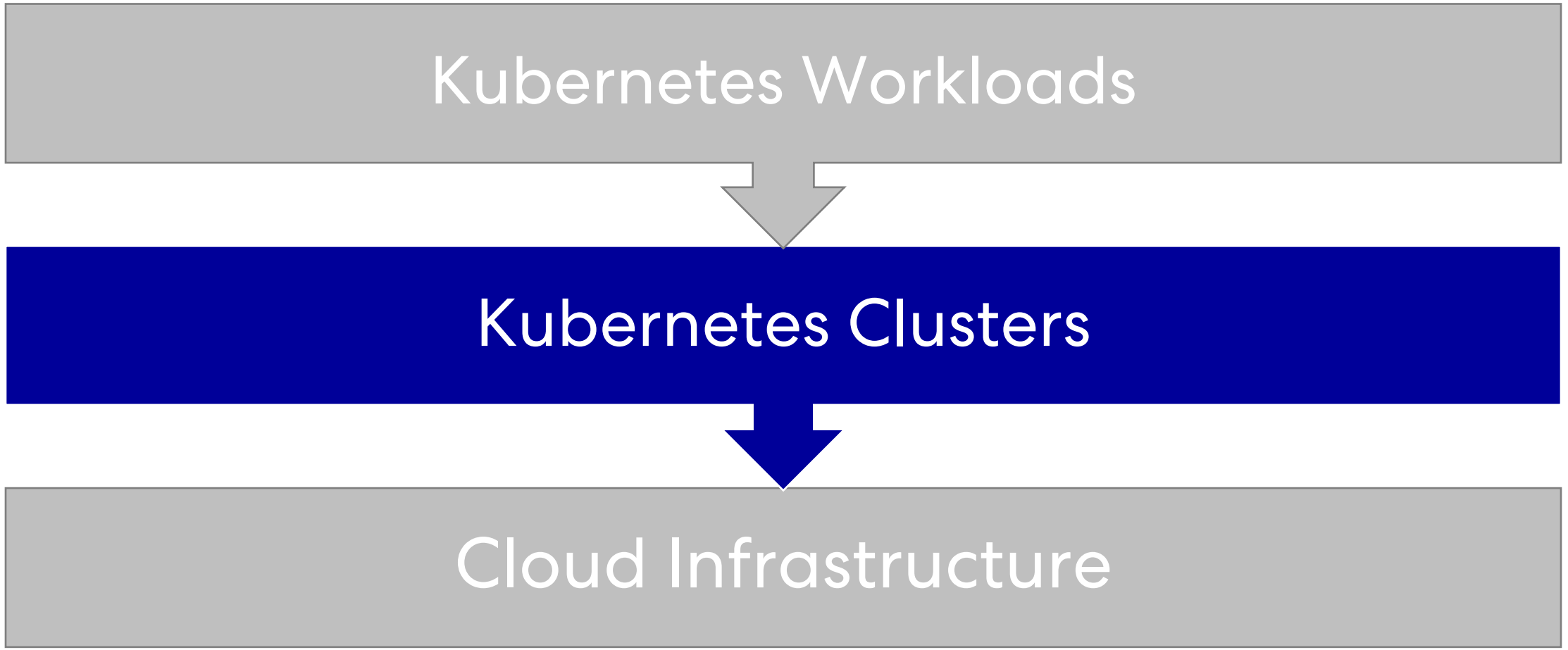


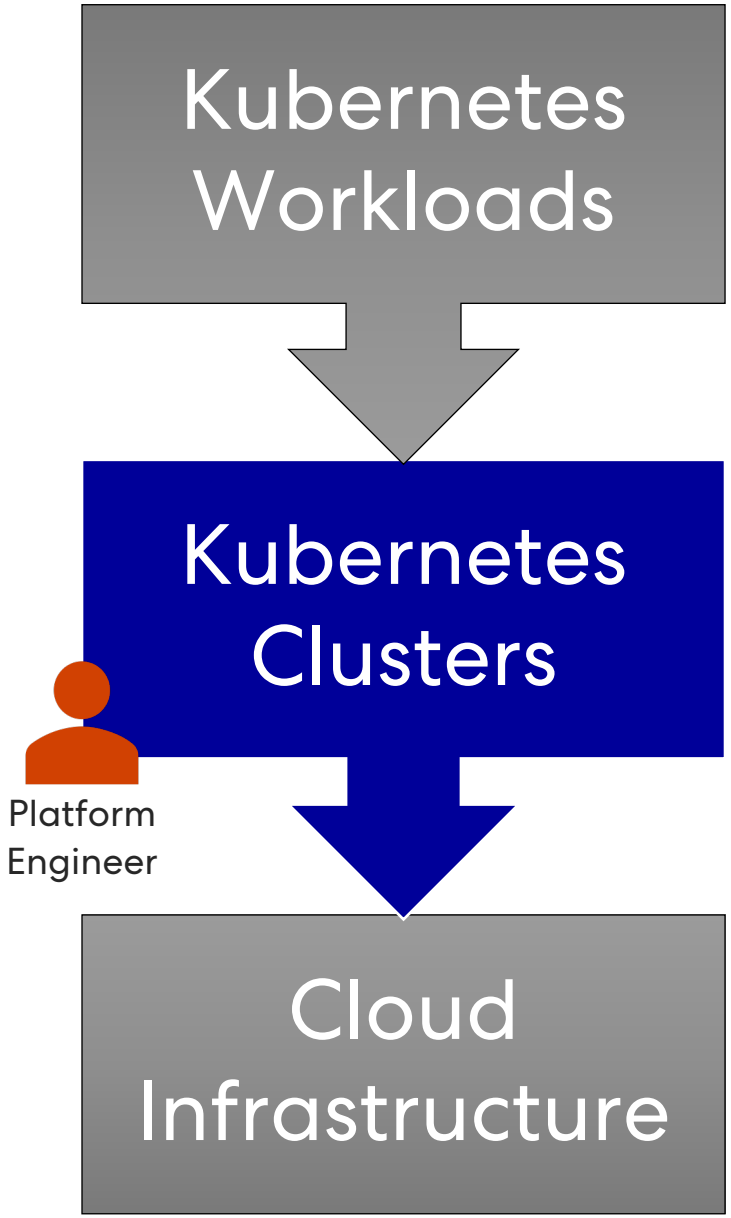






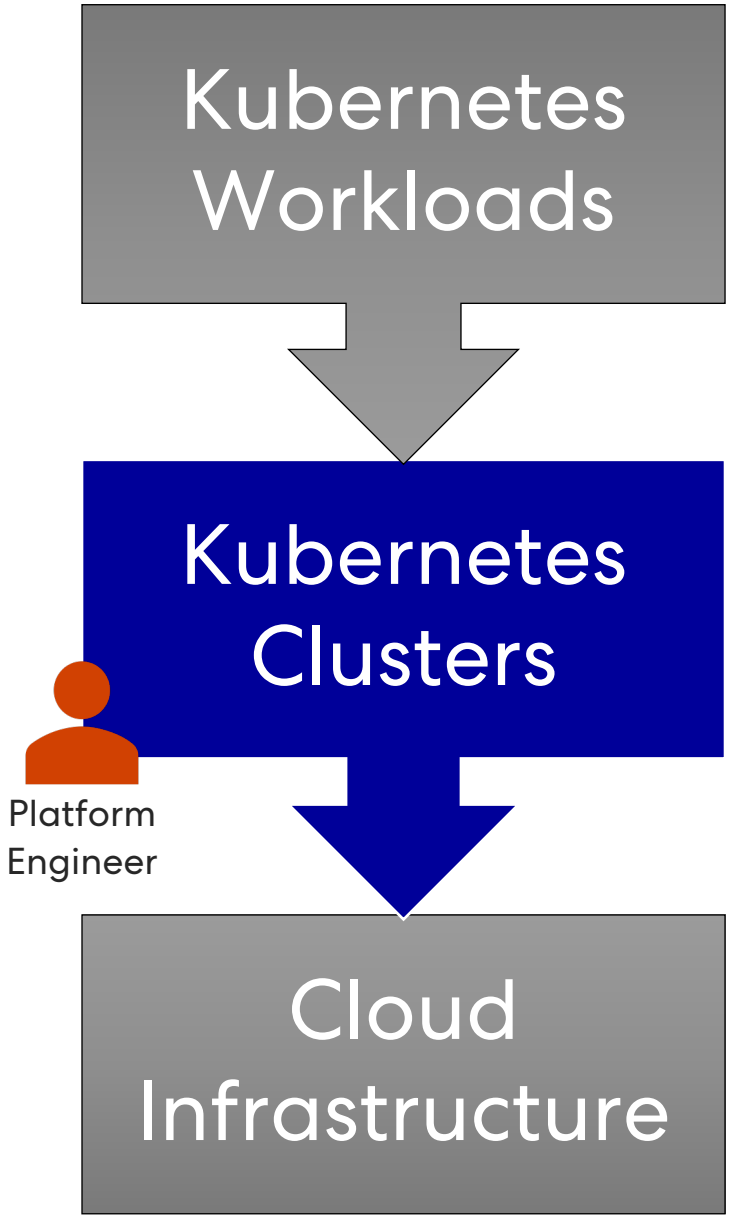
Kubernetes Cluster Optimisation





Kubernetes Cluster Optimisation

- Personas
  - Platform Engineer
- Responsibilities
  - Ensuring nodes in the cluster are rightsized for the workloads
  - Configuring Cluster Autoscaler / Karpenter
    - Take advantage of FinOps negotiated savings
    - Match the demand of workloads on the platform

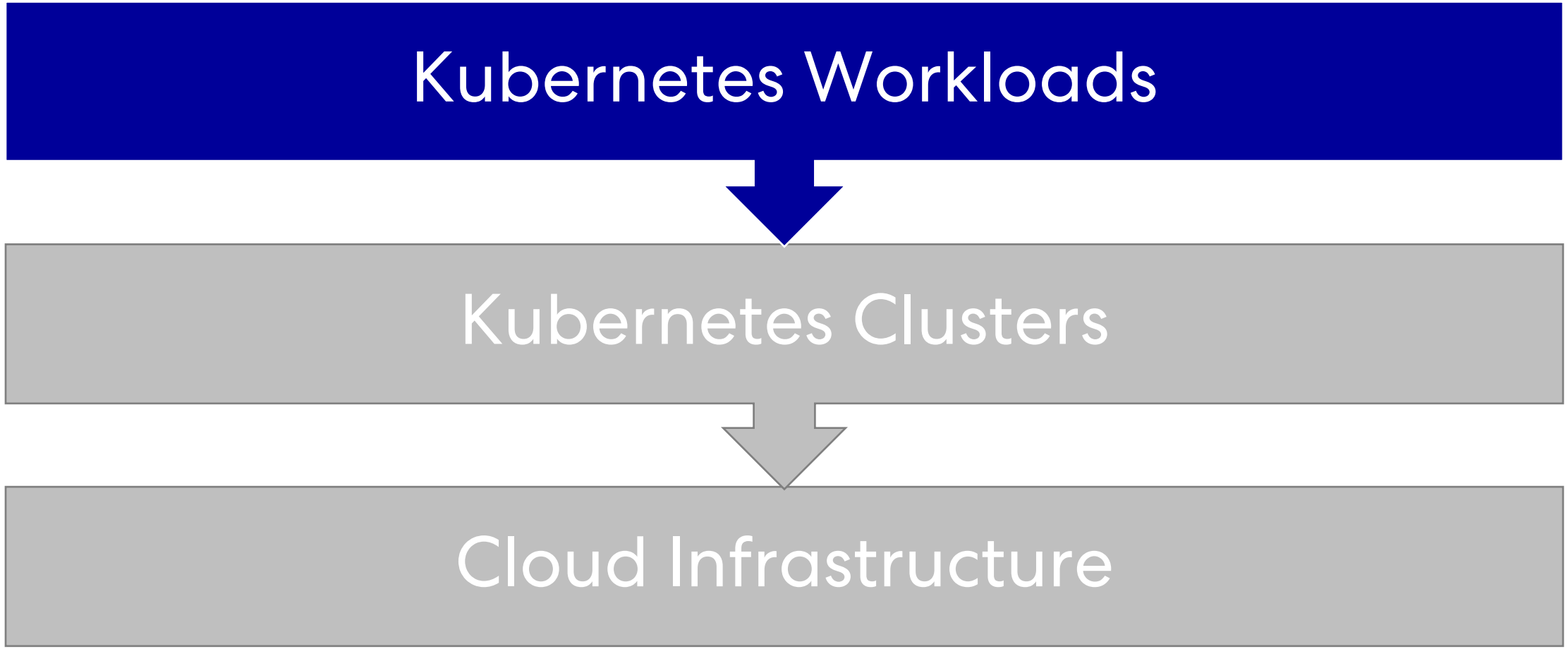


Kubernetes Cluster Optimisation Strategies

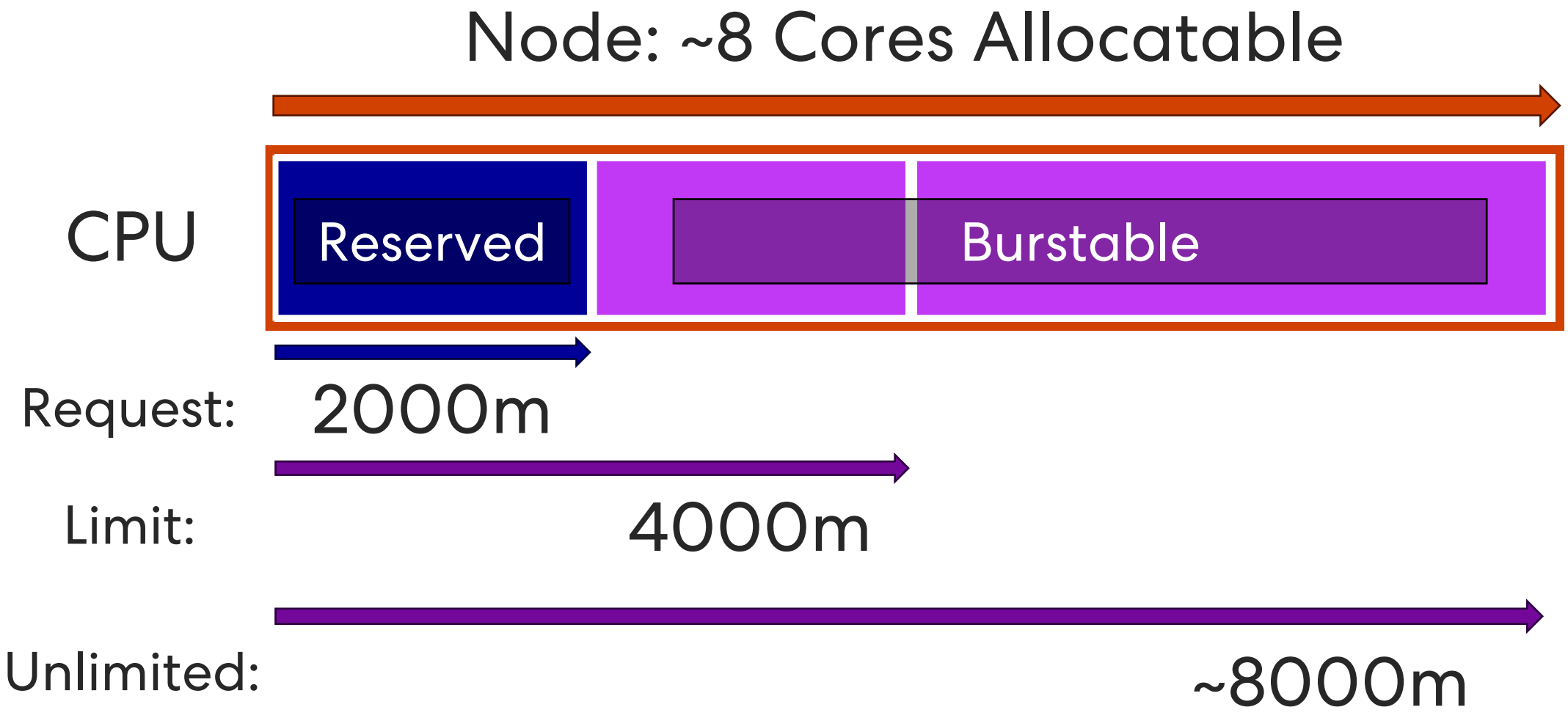
- Cleanup
  - Scale down unused clusters to zero nodes
  - Delete unclaimed volumes
- Rightsizing
  - Keep nodes spread across AZs for redundancy
  - Smaller nodes for smaller clusters
  - Choose instance families with best proportion of CPU:Memory (and storage, networking, GPU) to match average workload proportion
  - Don’t assume newer generations are more cost effective – do your research
  - Compare AMD vs Intel vs ARM (Graviton) *
- Use spot instances *

* Requires workload compatibility

Kubernetes Workload Optimisation

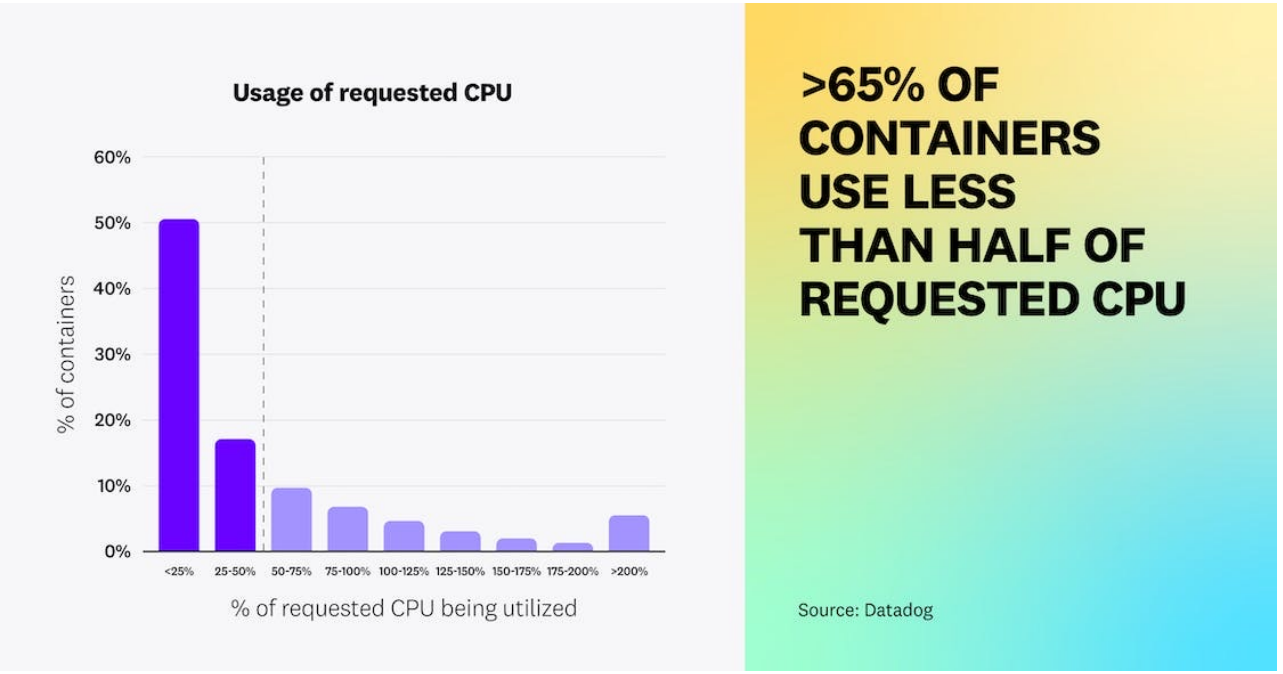


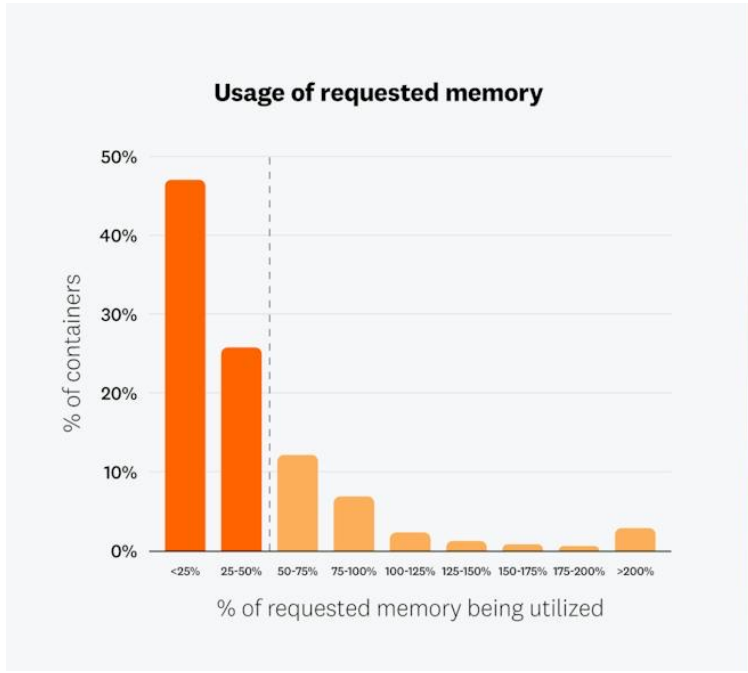
Kubernetes Workload Resourcing Intro

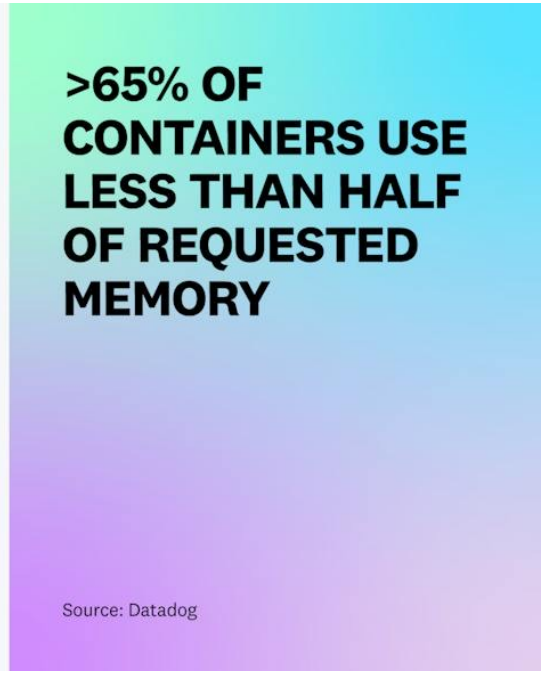


Industry Waste

- What percentage of containers use less than half requested resources?



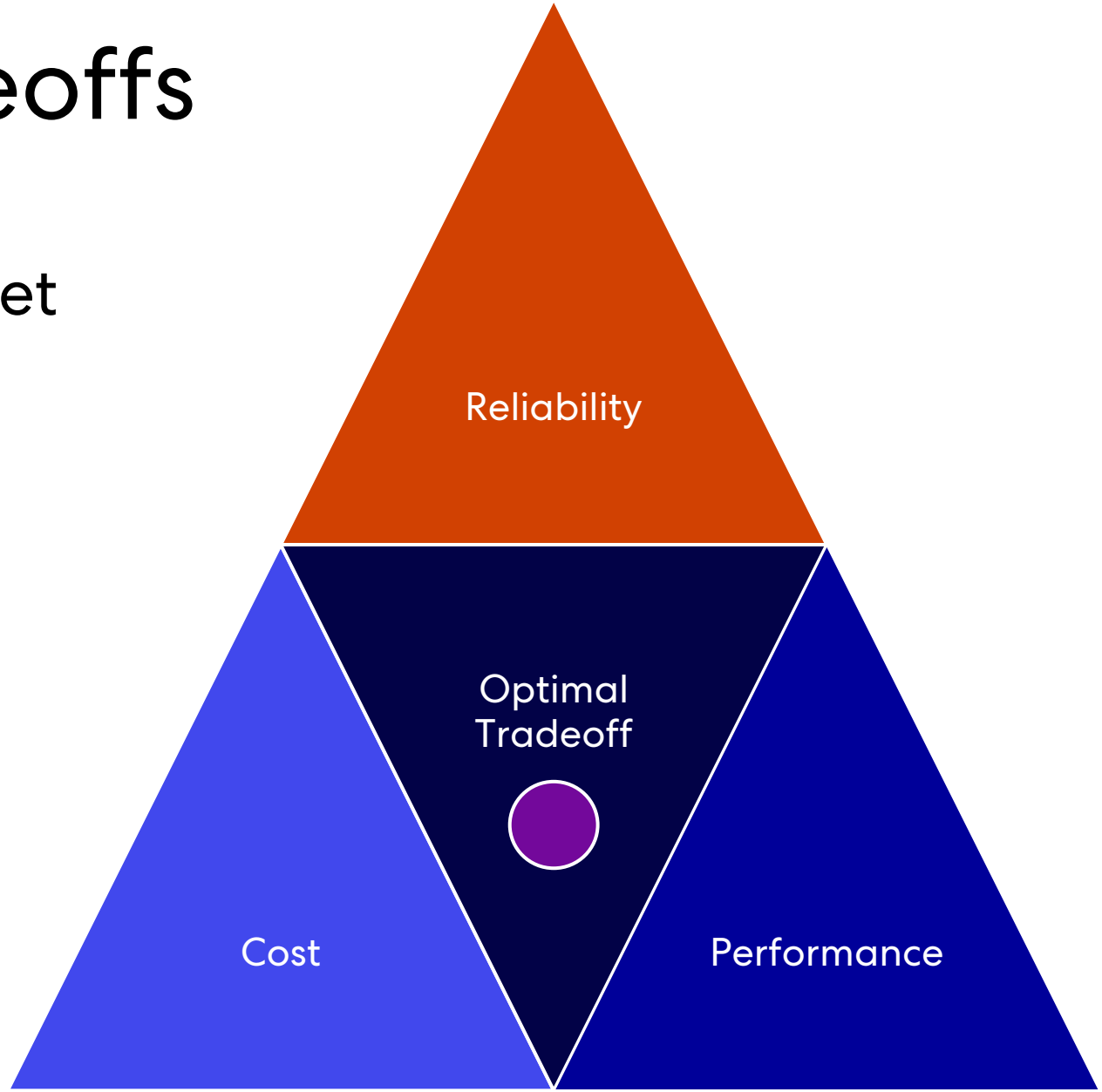




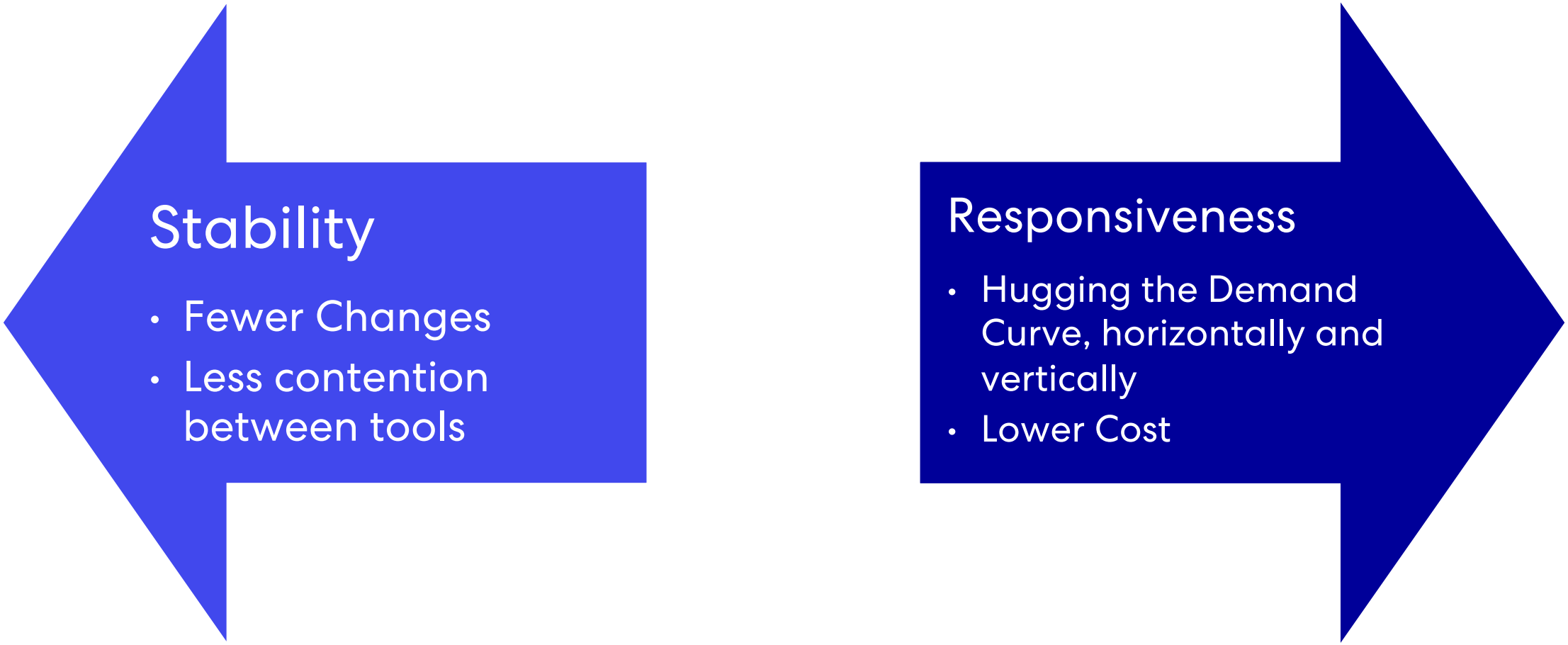
Source: https://www.datadoghq.com/container-report/ (November 2023)

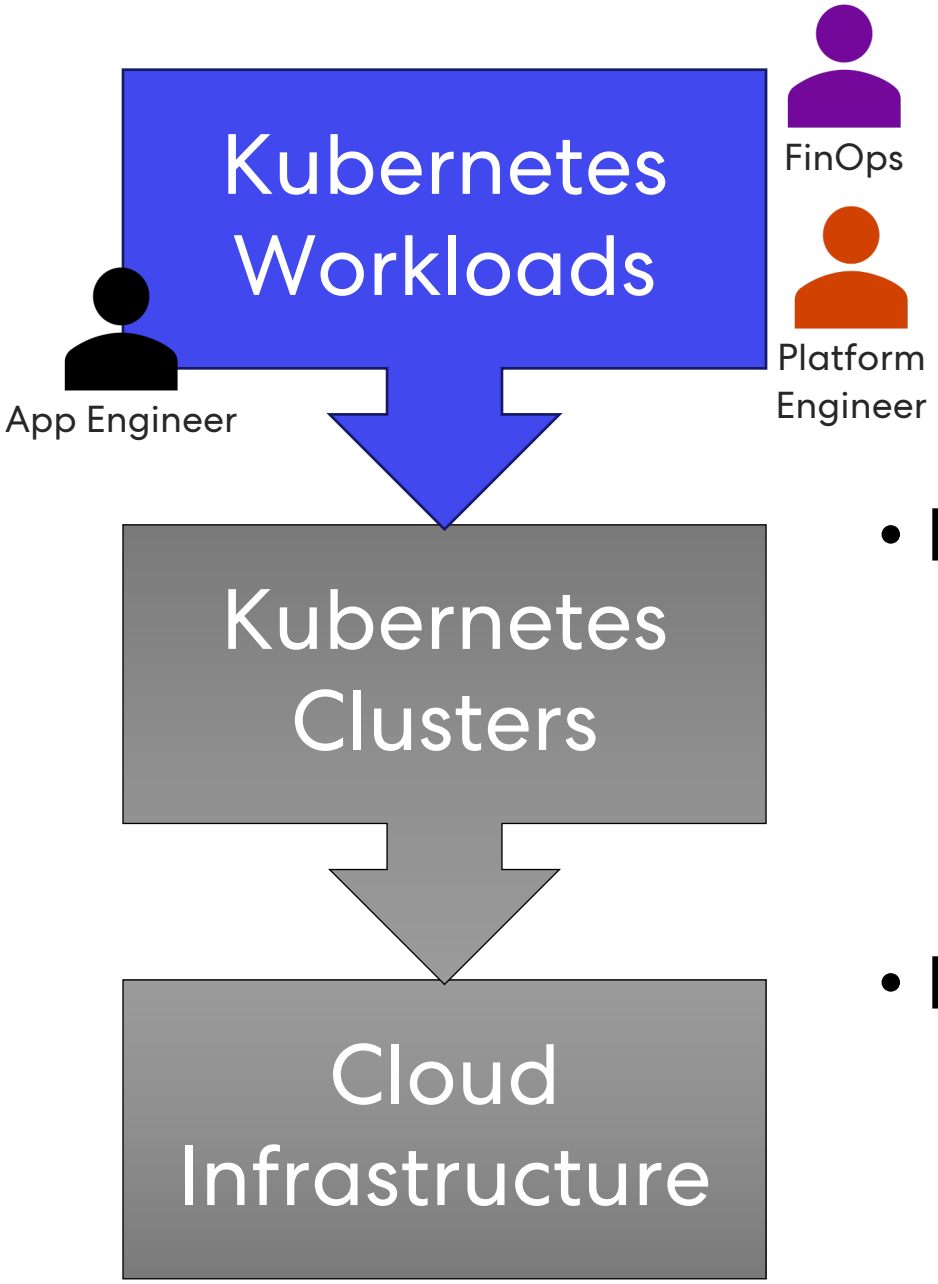
Optimisation Tradeoffs

- Sweet Spot is a Moving Target
- Performance
  - Feature Creep
  - Customer Demand
  - Platform Maturity
- Cost
  - Cloud Spend + Time Spent Optimising



Optimisation Tradeoffs: Responsiveness vs Stability





Kubernetes Workload Optimisation

- Personas
  - Application Engineer
  - Platform Engineer
  - CCOE / FinOps
- Responsibilities
  - Ensuring workloads are not overprovisioned
  - Ensuring workloads are reliable

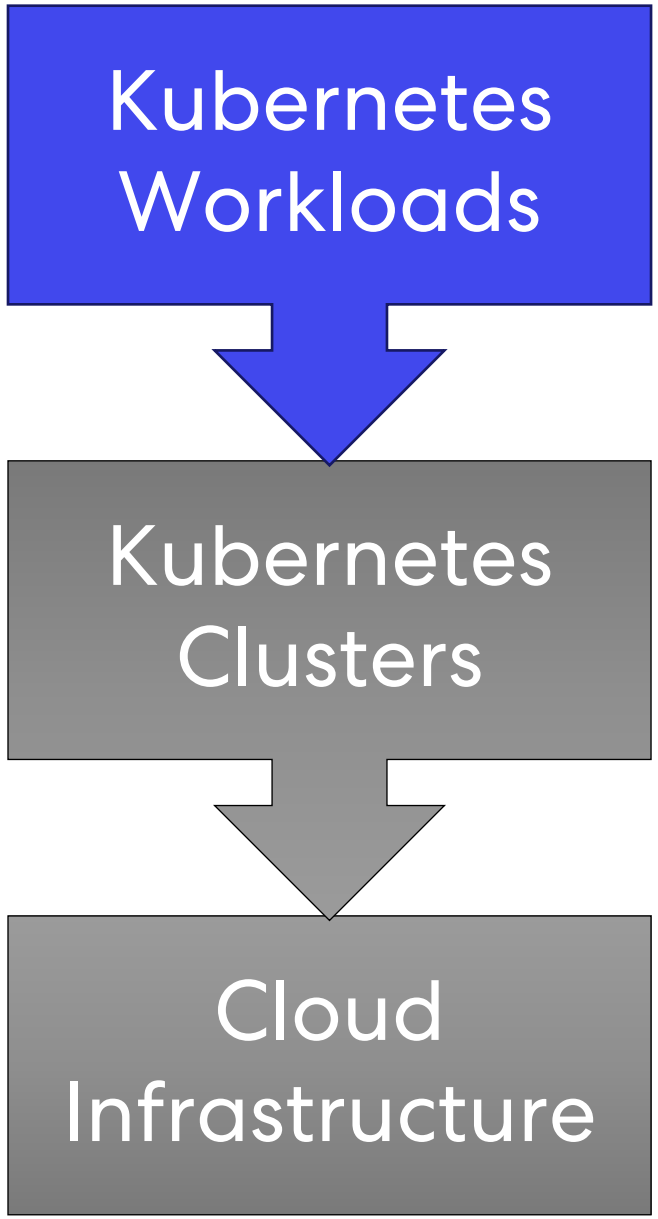
Kubernetes Quality of Service (QoS) Classes

QoS	Definition
Best Effort	No Requests
Guaranteed	Requests == Limits
Burstable	Requests < Limits

Request	Too low	Too high
CPU	Starvation/Throttling – may not get CPU cycles needed	Inefficiency
Memory	Kill risk – Out Of Memory (OOM Kill)	Inefficiency

QoS Best Practices

Resource	"Best Practice"
Memory **(Non-compressible)**	Set memory Requests == Limits (Guaranteed QoS). Avoids OOM Kills.
CPU **(Compressible)**	No industry consensus. Options: 1. Burstable: No Limits or Requests < Limits • Take advantage of spare cycles until HPA kicks in • Know what you're doing. Configure according to your SLO. 2. Guaranteed (Requests == Limits). This avoids: • Inconsistent performance in different environments • Unbalanced (unlimited) CPU allocation vs (limited) memory allocation – i.e. bursting CPU beyond requests could lead to pressure on memory if HPA not configured properly. • Guaranteed and Whole Number CPUs can take advantage of "Static CPU Manager" for Core affinity for higher performance

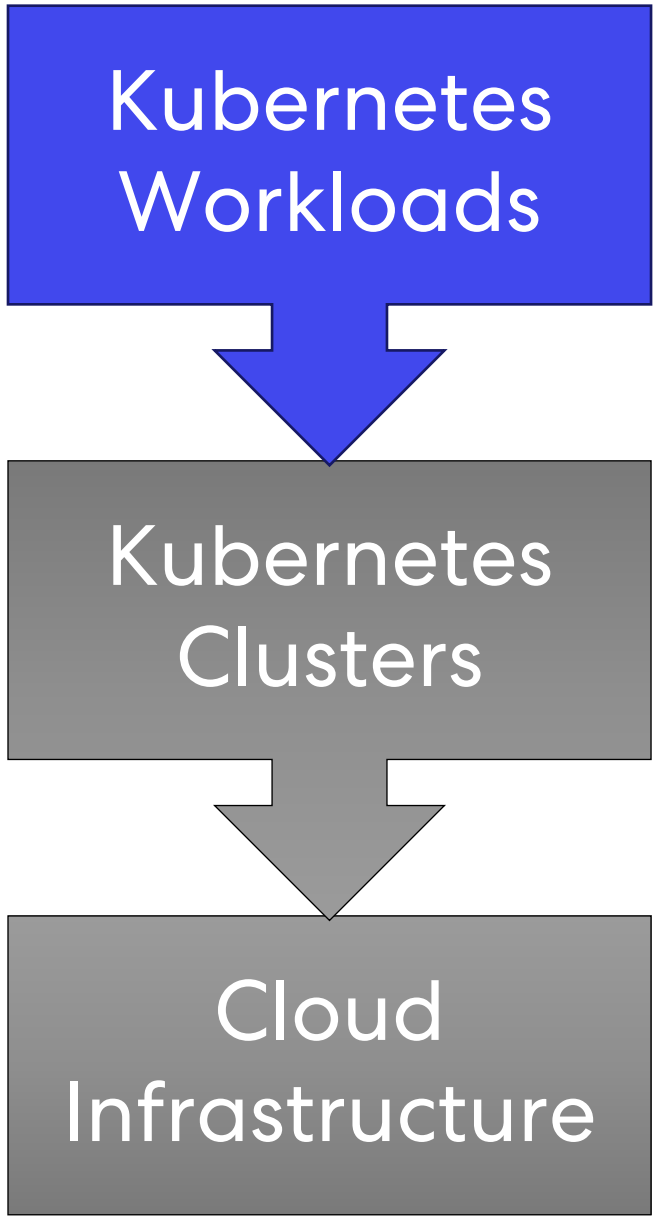


Kubernetes Workload Optimisation Strategies

- Cleanup
  - Delete abandoned workloads/namespaces
  - Delete volumes unattached to containers
- Reward compatibility with cheaper infra
  - Interruptible = spot instances
  - Graviton-compatibility
- Rightsizing Policies
  - Sensible Requests/Limits Defaults and/or Enforce “Best Practice”
- Recommendations
  - Shift Left: Show estimates on PRs; GitOps: Submit PR recommendations
  - CPU: Close gap between Usage and Requests
    - Use TP95 stat, not Average (hides distribution)
    - E.g. Policy: <10% utilisation → Recommend 2x 95th Percentile
    - E.g. Request = 1000m, 95th Percentile Usage = 100m, Recommendation = 200m


Platform Engineer


FinOps



Kubernetes Workload Optimisation Strategies

- Recommendations Example

Tier	Request
Critical / Highly Available	99.99th percentile + 100% headroom
Production / Non-critical	99th + 50% headroom
Dev / Experimental	95th or consider namespace quotas*

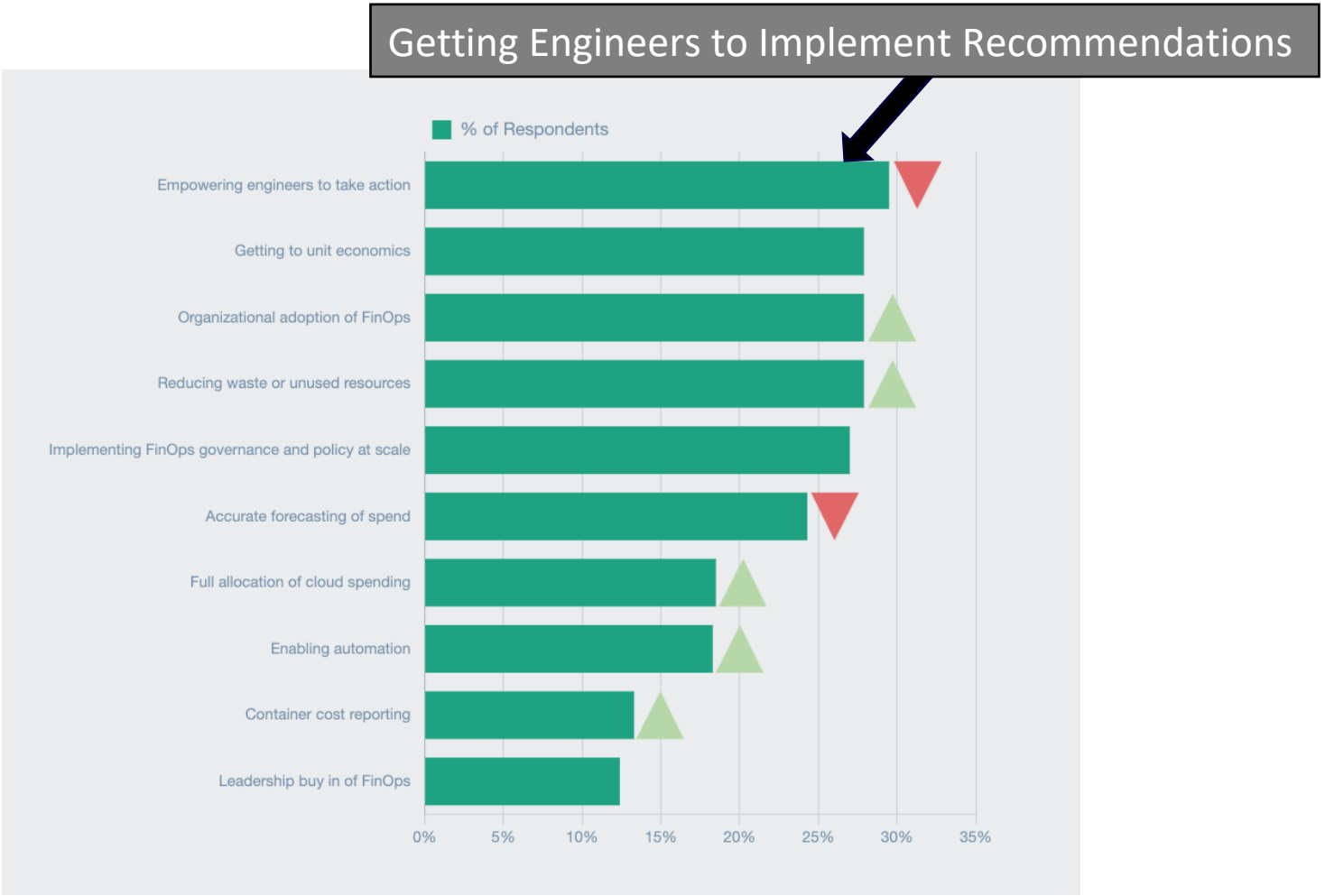
Source: https://blog.kubecost.com/blog/requests-and-limits/

Kubernetes Workloads

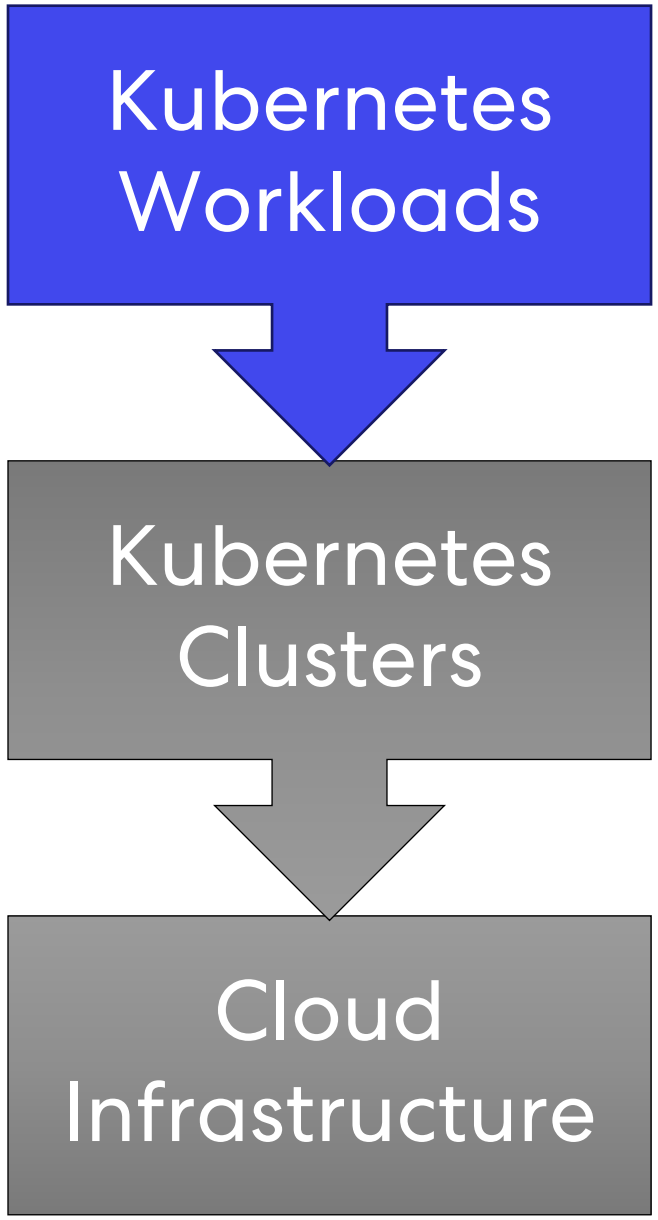
Kubernetes Clusters

Cloud Infrastructure

The Need for Automation



State of FinOps Report 2023: https://data.finops.org/#3388

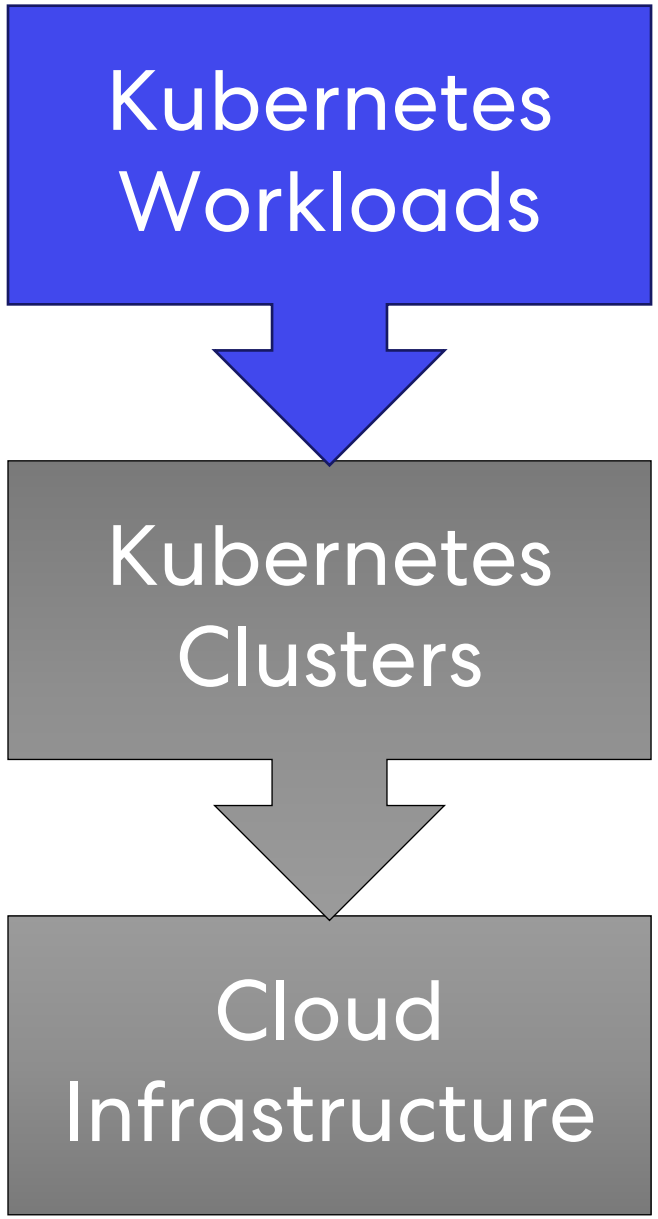


The Need for Automation

- For a team managing 10 microservices, releasing once per week...

10 microservices
x 10 CPU / Memory resource settings
x 10 HPA min / max auto scaling settings
x 10 metrics: CPU, memory, thread count, Latency, error rates, TP99, TP90, garbage collection settings
x 3 warm up settings (e.g. mittens rate, time, concurrency)
x 3 average traffic patterns
x 4 releases

= **360,000 permutations per month** to find optimal cost and performance

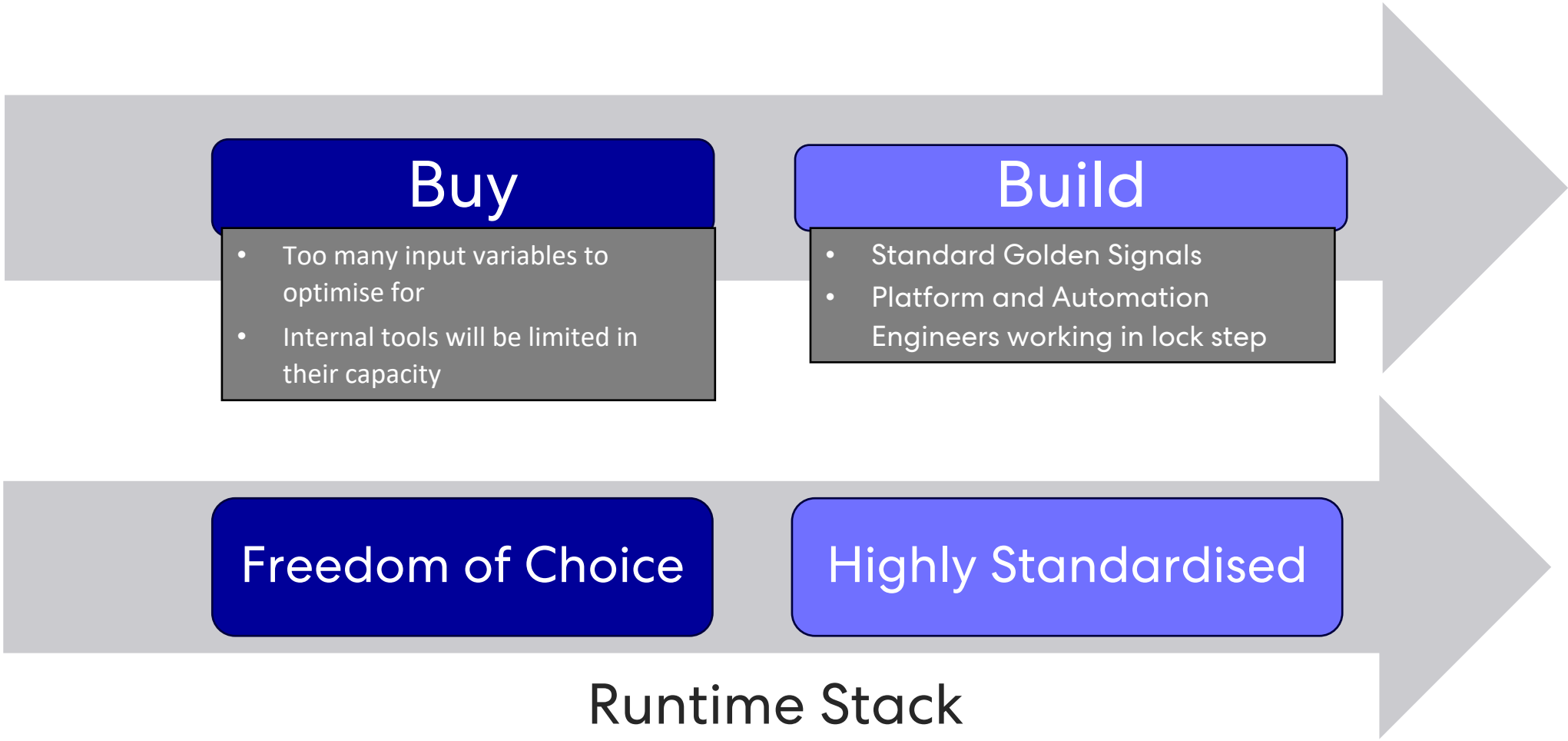


Automation: The Need for Bravery

- Whether you buy or build autonomous solutions…
- Automation is the only way to see measurable outcomes
- Bravery is biggest barrier for success
- Requires leaders willing to learn and iterate
- Savings rates above 70%
- That's the correct use of cloud – the original design for flexibility / elasticity

Kubernetes Workloads

Workload Optimisation Buy vs Build?

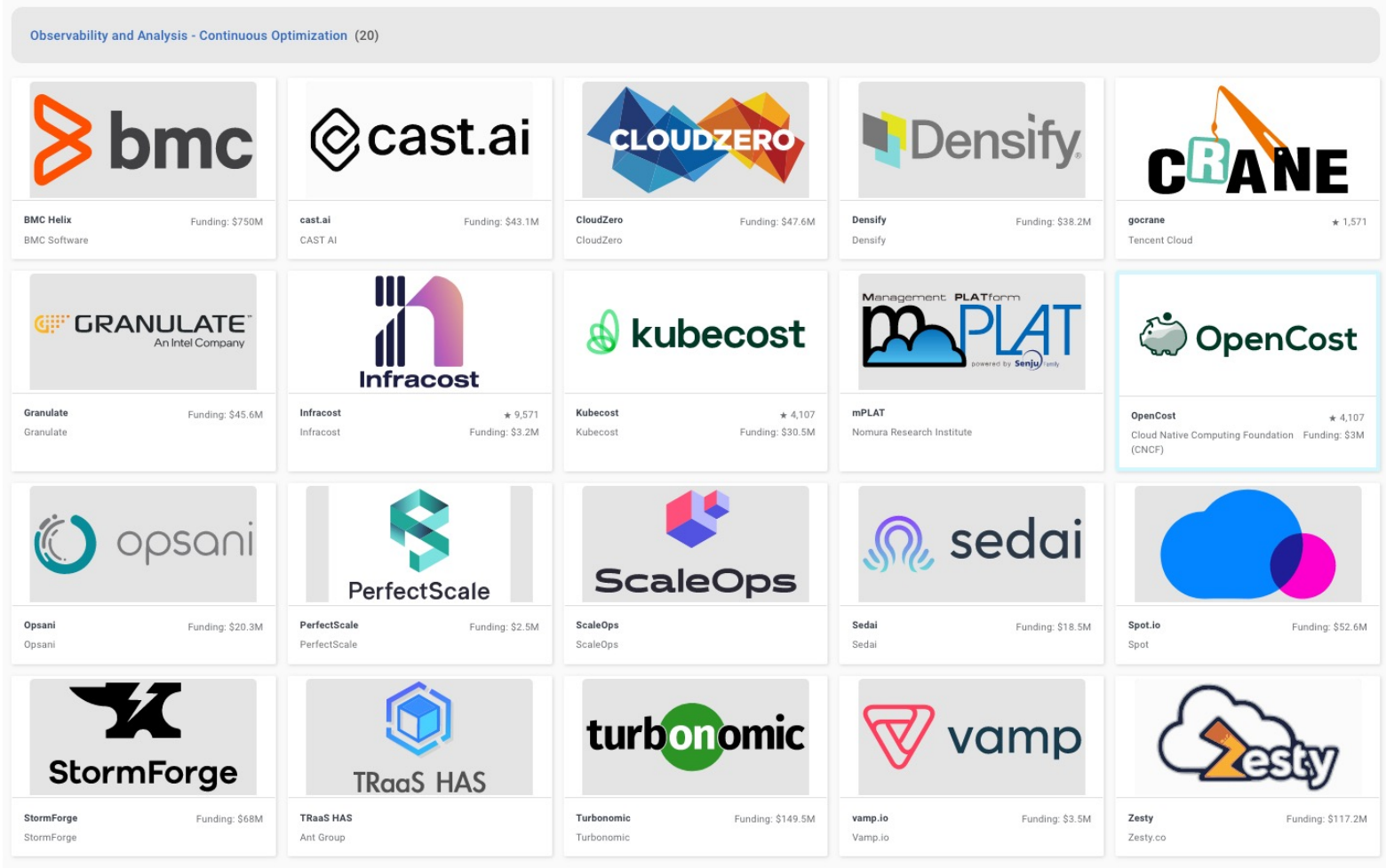


Kubernetes Workloads

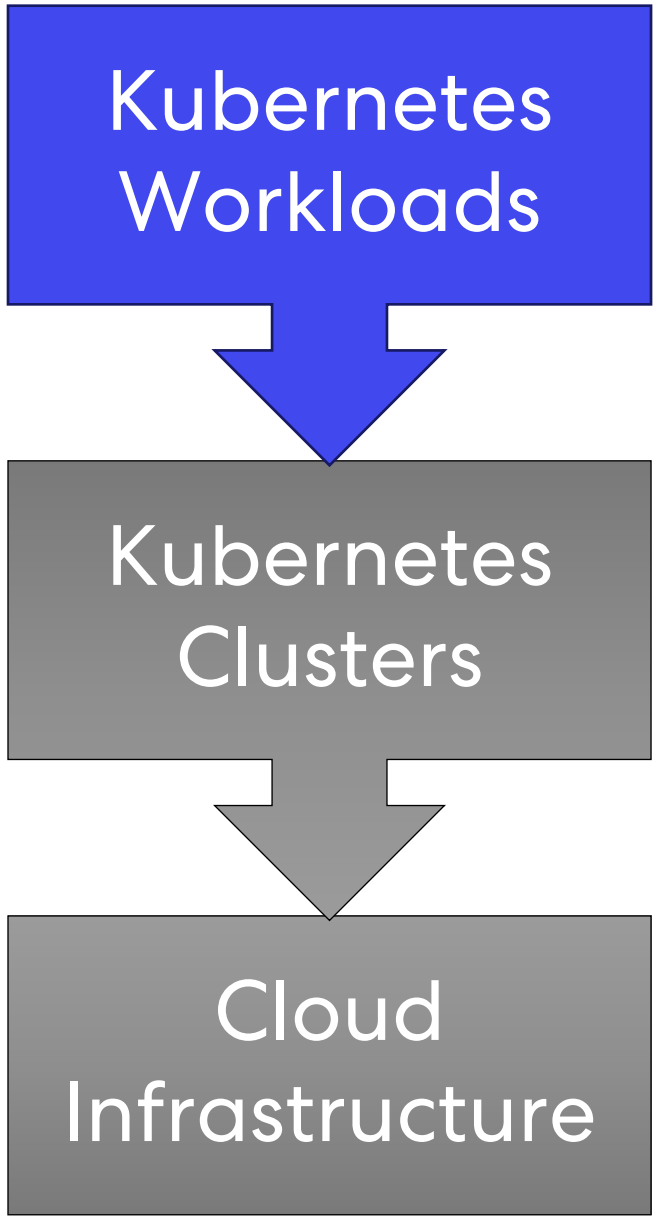
Kubernetes Clusters

Cloud Infrastructure

Optimisation Vendor Solutions

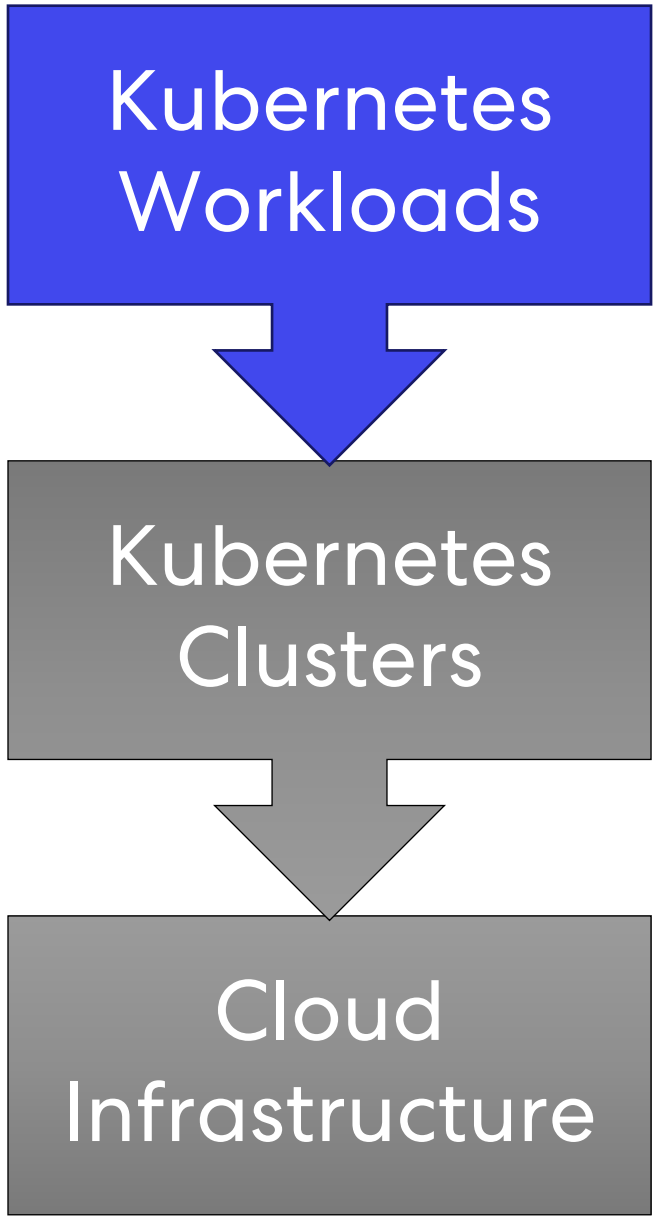


"Continuous Optimization" category in CNCF: https://landscape.cncf.io/card-mode?category=continuous-optimization&grouping=category



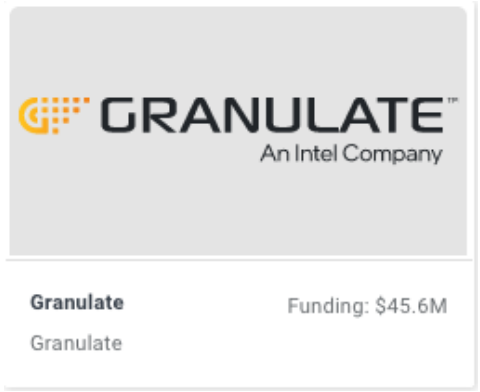
Automated Workload Optimisation Strategies

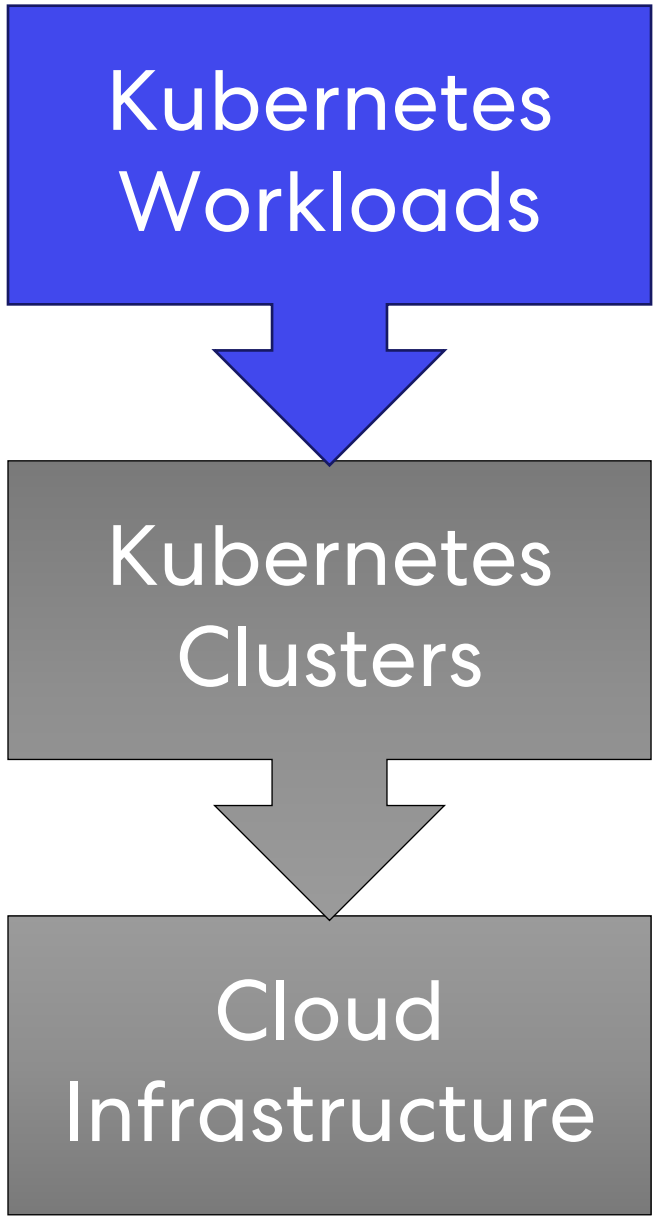
1. Low-level Magic
2. Actively Configure Resources
3. Explore Optimal Configuration



Workload Strategy #1: Low-level Magic

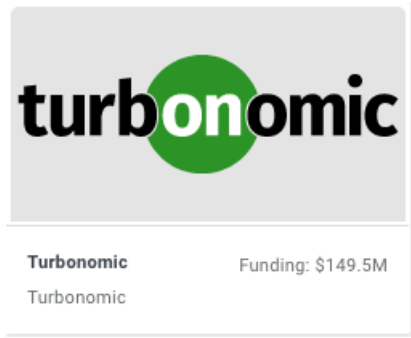
- Based on ML model
- Optimise kernel and runtime (JVM, Golang, etc) decisions
  - Thread scheduling, Lockless networking, IPC, Connection pooling, Congestion control, Memory allocations

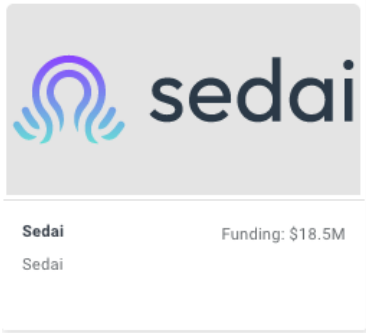


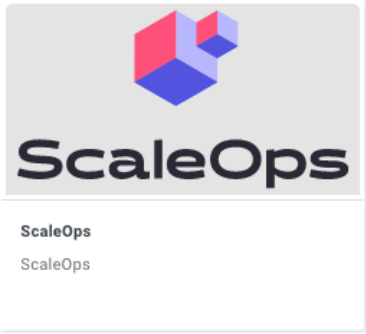


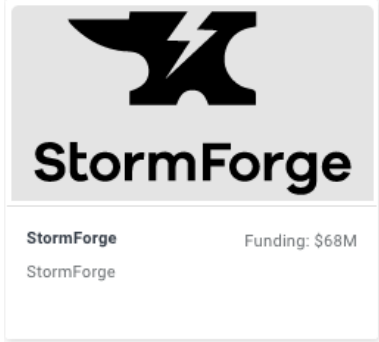
Workload Strategy #2: Actively Configure Resources

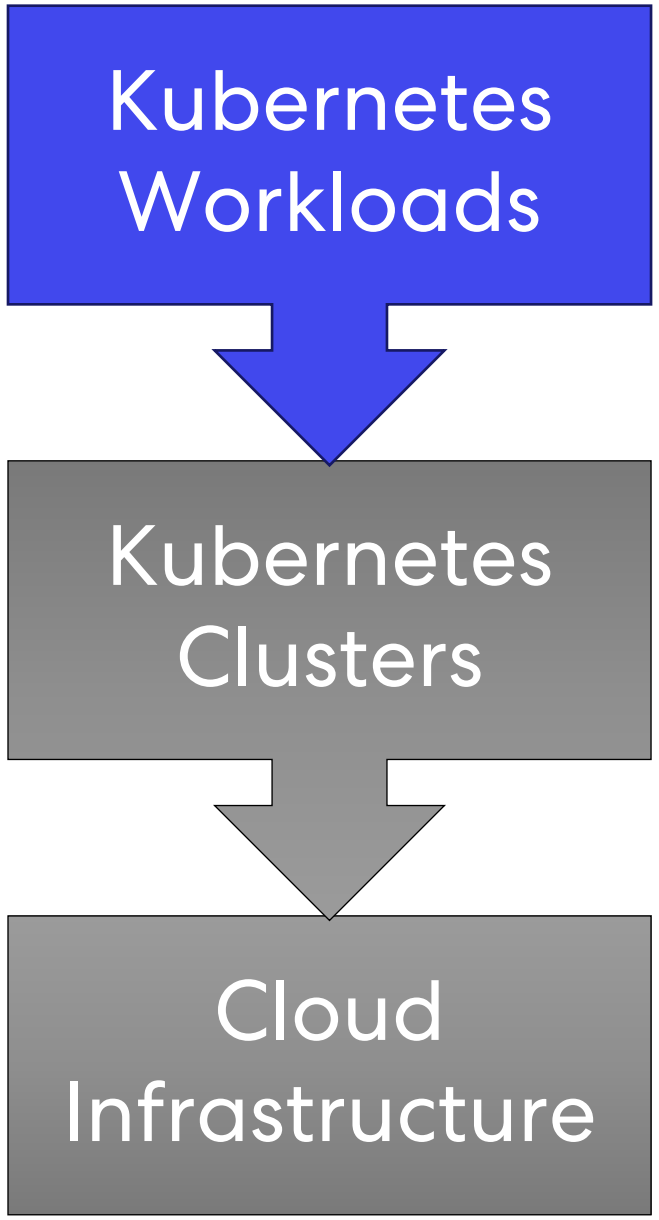
- Vertically Scale (CPU requests/limits)
- Horizontally Scale (HPA)
- Based on ML and RL
- Safety-first, gradual tweaking





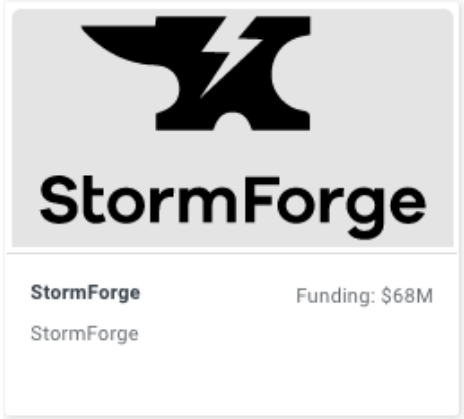


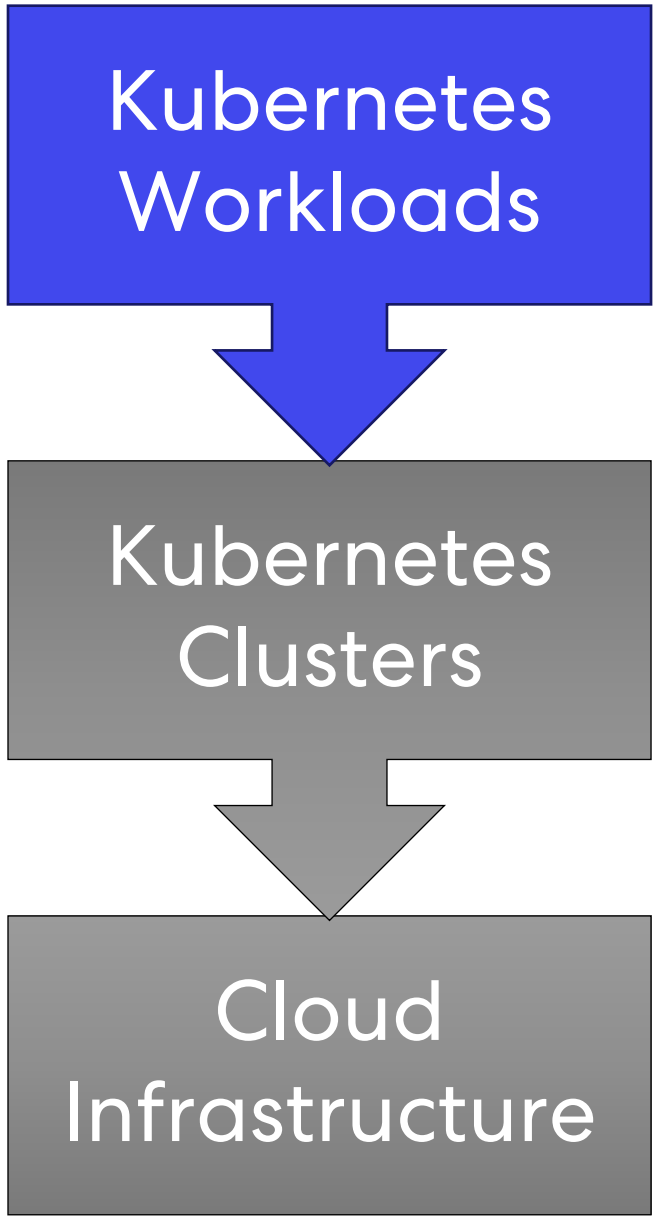




Workload Strategy #3: Explore Optimal Config

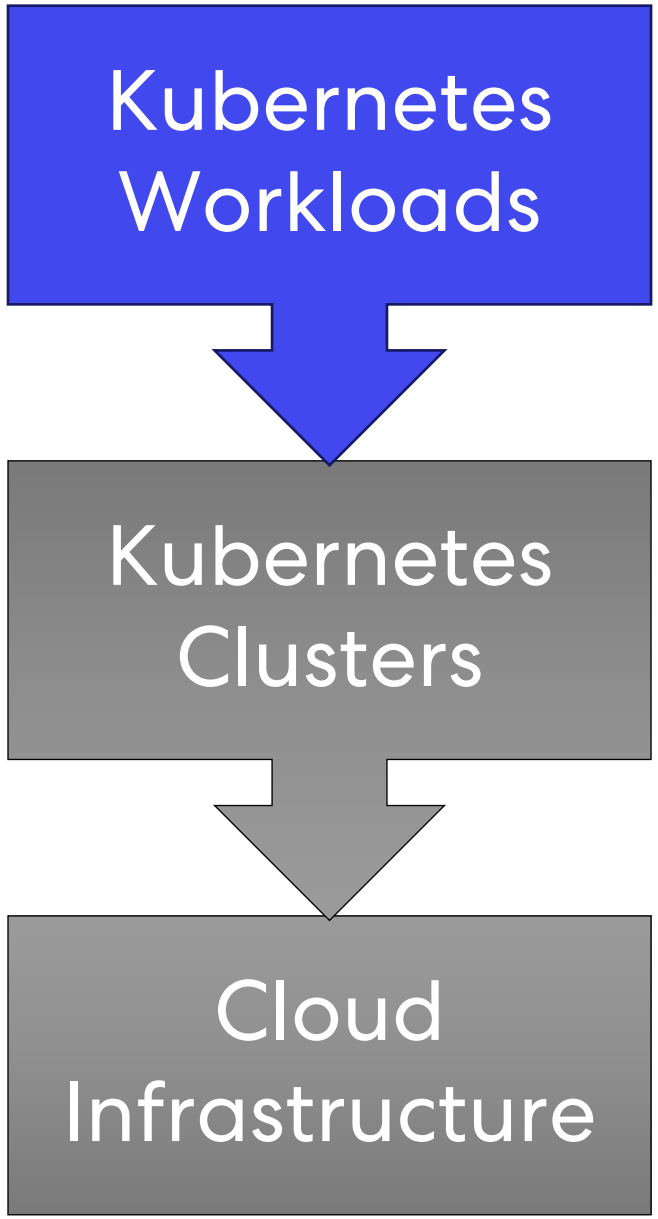
- Find optimal configuration via Data Science in pre-production
- Run trials with performance testing frameworks
- Explore optimal configuration settings





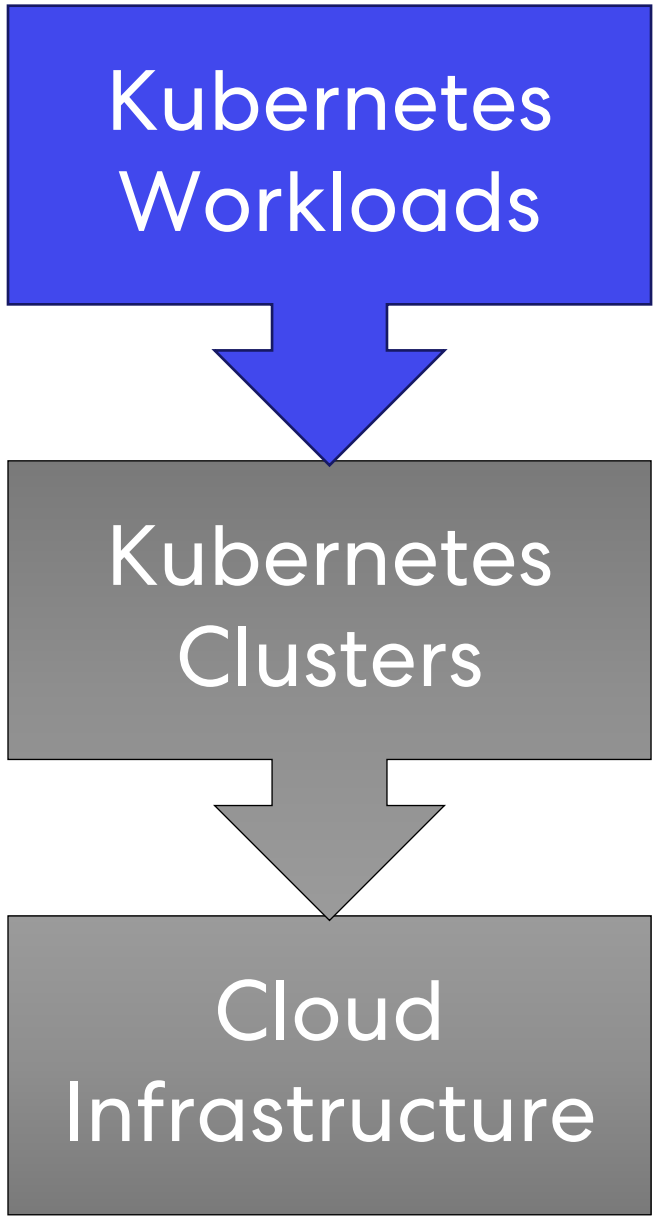
Autonomous Optimisation Adoption Tips

1. Empower Engineers With Choice
2. Be Above Reproach
3. No One-Size-Fits-All Approach
4. Exploring Pareto Front is Hard
5. Timing is Everything



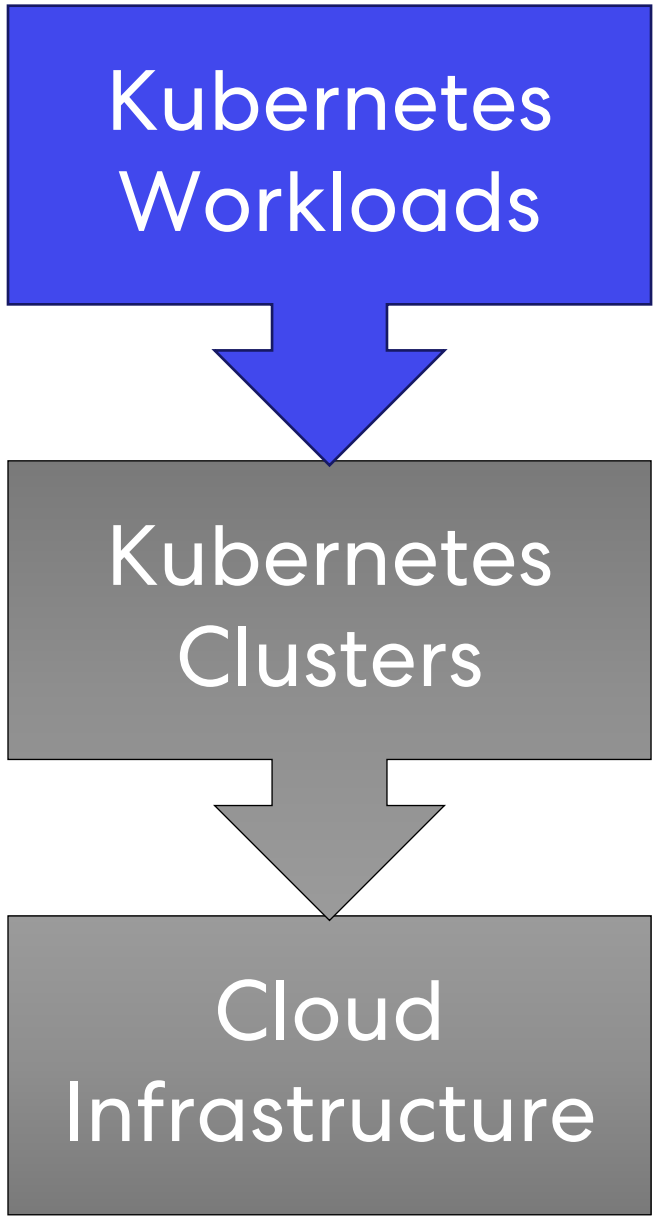
Workload Optimisation Automation Tip #1

1. Empower Engineers With Choice
   - Give them a threshold choice
   - E.g. for CPU Requests recommendation: Choosing 95th Percentile vs 99th Percentile
   - Engineers feel a sense of control and psychological ownership
     - “These are my recommendations because I tuned them”
   - Uptake of recommendations increases



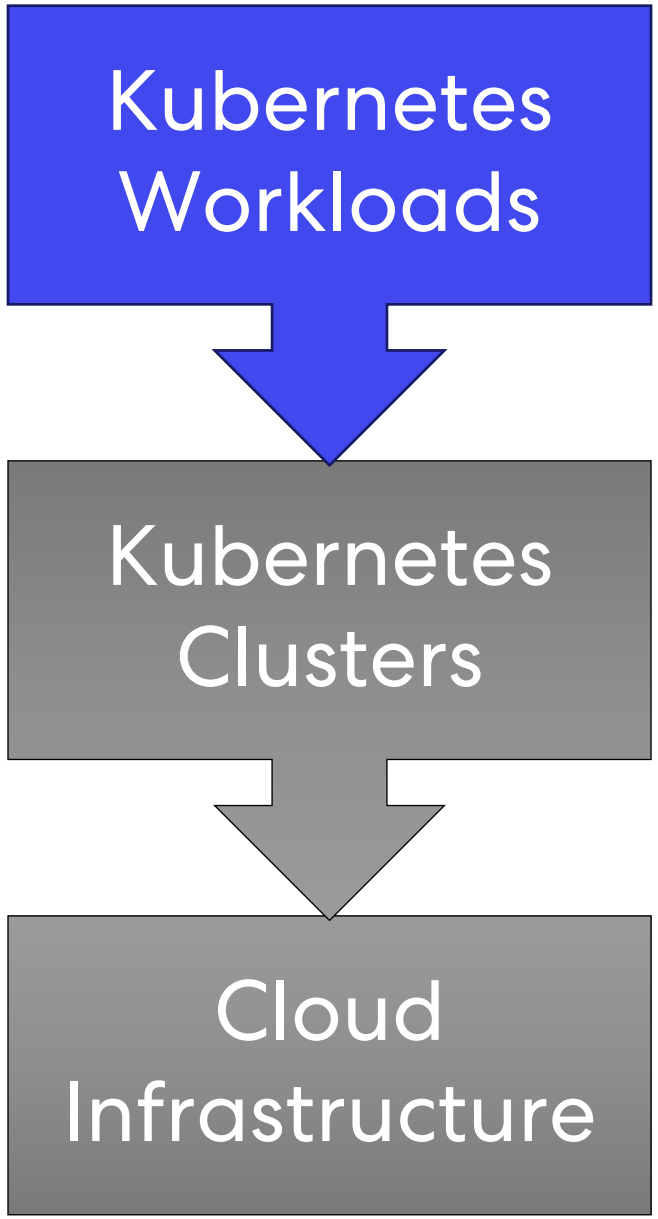
Workload Optimisation Automation Tip #2

2. Be Above Reproach
   - Don't start with unrealistic or simplistic recommendations
   - Two Extremes
     - Blind Trust
     - Zero Trust
   - Earn trust with trustworthy recommendations
   - Err on side of caution. Trust is hard to win back.



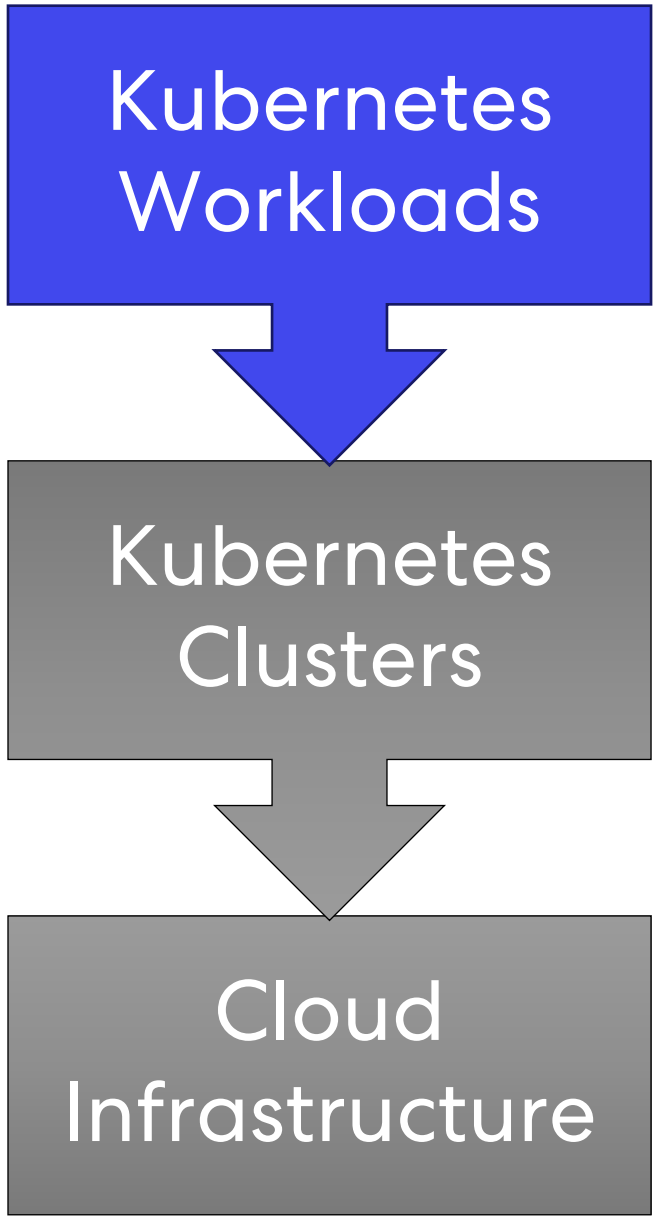
Workload Optimisation Automation Tip #3

3. No One-Size-Fits-All Approach
   - Some workloads are easily interruptible and restartable (12-Factor / Cloud-native)
   - Others expect to be long-running and never want to be interrupted outside a release cycle (lift-and-shift, poor design)
   - Others have long warm up times and need long lead times to scale out (low HPA thresholds)



Workload Optimisation Automation Tip #4

4. Exploring Pareto Front is Hard
   - Optimising for both cost and performance outputs is a difficult problem
   - Many input paramaters to consider
   - CPU and Memory sizing changes can affect each other



Workload Optimisation Automation Tip #5

5. Timing is Everything
   - Scaling both vertically and horizontally requires careful observation times (“bake-in”) between adjustments to avoid thrashing of pods
   - Some periods of time are better than others to apply optimization changes
     - E.g. a slower rate of change may be required during volatile / peaky traffic demand

Summary

- FinOps
  - Collaboration between Finance and Engineering
  - Centralised Rate Negotiation and Best Practices
  - Decentralised Ownership
- Kubernetes Optimisation
  - Personas work together to optimise the entire stack
  - Automation is required for realising true cost savings
  - Be brave, be realistic, be above reproach

Resources

- FinOps
  - Book: “Cloud FinOps” Book 2nd Edition
  - Podcasts: “FinOps Pod”, “The FinOps Guys”
  - Slack: finopsfoundation.slack.com
  - Certification: learn.finops.org
- Optimisation
  - YouTube: @stormforgeio, @sedaiio, @granulate_io
  - PodCast: Screaming in the Cloud Ep 416 “The Complexities of AWS Cost Optimization with Rick Ochs”

Questions?

Matt Callanan
Expedia Group
https://www.linkedin.com/in/matthewcallanan/