# The Concept-to-Console Playbook: SRE in Google Cloud

## Operationalizing the PDE Section 3 Architecture

Professional Cloud DevOps Engineer (PDE) Reference Guide


Architecture: Cloud Run & GKE Autopilot

## Governing Velocity

Metrics, SLIs, SLOs, and the Error Budget deployment gate.

Section 3.1 Inter

## Lifecycle & Capacity

Autoscaling dimensions, limits, and resource optimization on GKE & Cloud Run.

Section 3.2 Inter

## Continuous Evolution

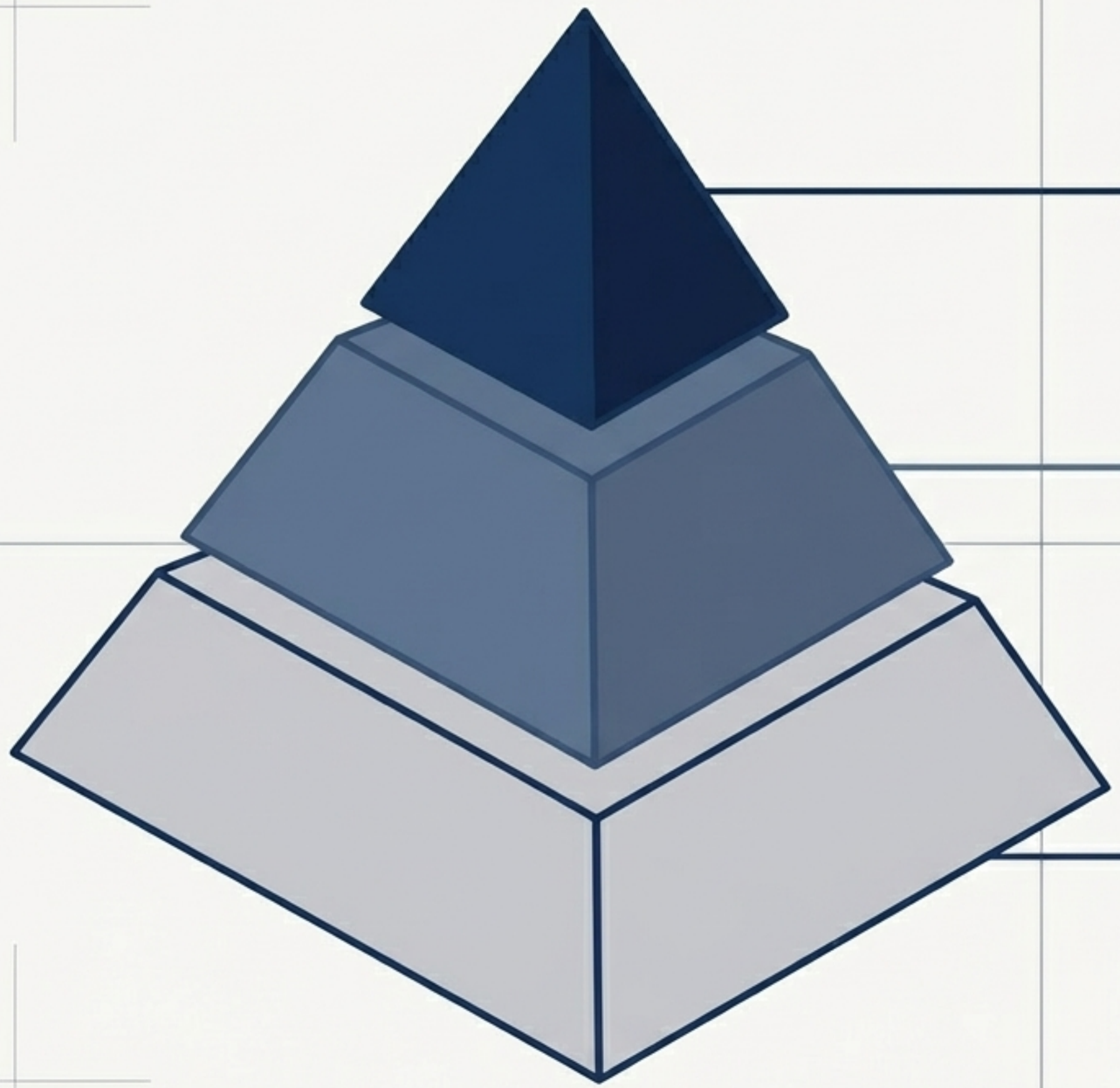
Eliminating toil, chaos engineering, and blameless post-mortems.

Culture & Practice Inter

## Incident Mitigation

Traffic shaping, blast radius reduction, and rapid rollbacks.

Section 3.3 Inter



## **SLI** (The Metric)

**Definition:** A quantitative measure of service behavior.

**Raw Telemetry:**
`run.googleapis.com/container/cpu/utilizations`

**Console:** Monitoring > Dashboards (p50/p95/p99 latency, error rates)

## **SLO** (The Target)

**Definition:** Internal engineering target for the SLI.

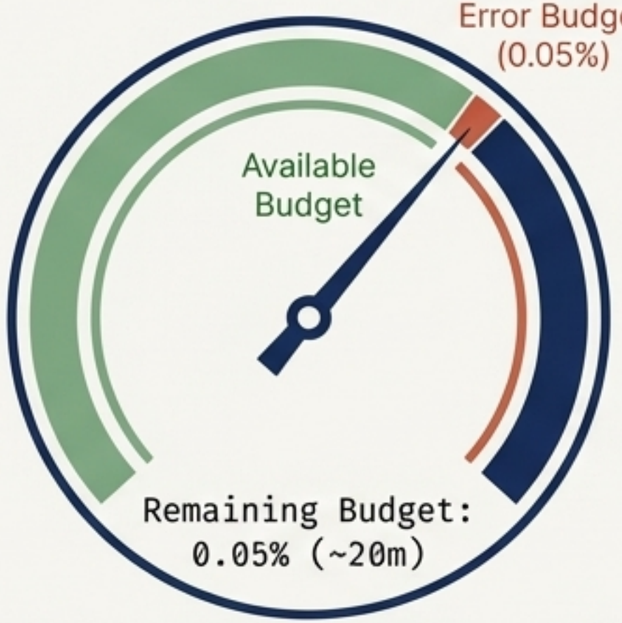
**Example:** 99.9% of requests to /checkout return HTTP 2xx within 500ms over a 30-day window.

## **SLA** (The Contract)

**Definition:** External business commitment, typically 10–20% looser than the SLO.

**Example:** 99.5% guaranteed availability. (Buffer protects against financial penalties).

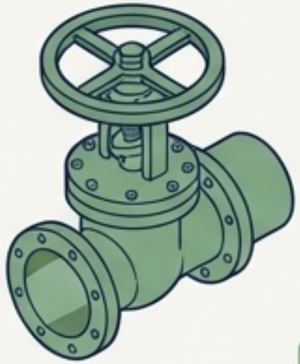
## 28-Day Rolling Window




**Streaming Media API Target:** 99.95% availability.
Error Budget = 0.05% (~20 minutes of allowed downtime per month).

## Deployment Pipeline Valve: Error Budget States

### Valve OPEN

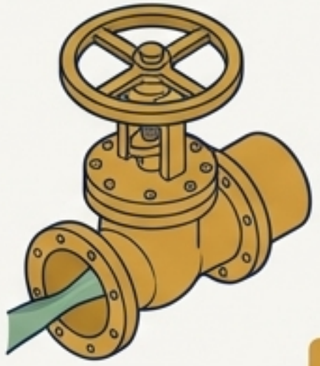


**Budget > 50%**

**Normal velocity.**

Feature deployments are automatically approved.

### Valve RESTRICTED



**Budget < 50%**

**Slow down.**

Only bug fixes and reliability improvements allowed.

### Valve CLOSED

**Budget = 0%**

**Feature freeze.**

All feature work stops until the 28-day window rolls forward.

**Key Takeaway:** The budget replaces subjective management approval with an objective, automated governance mechanism.

# The Code

modules/app_monitoring/monitoring.tf

```
cpu_threshold = 0.9
```

When this threshold is breached, the error budget is actively consumed.

# The Console

Monitoring > SLOs & Alerting

Current Compliance: 99.94%

Remaining Budget: 0.06%

## Multi-Burn-Rate Alerts

**Fast Burn Alert**: 14x sustainable rate over 1 hour → Action: Page on-call immediately.

**Slow Burn Alert**: 6x sustainable rate over 6 hours → Action: Create a ticket for next business day.

# Defining Capacity Contracts

## Cloud Run

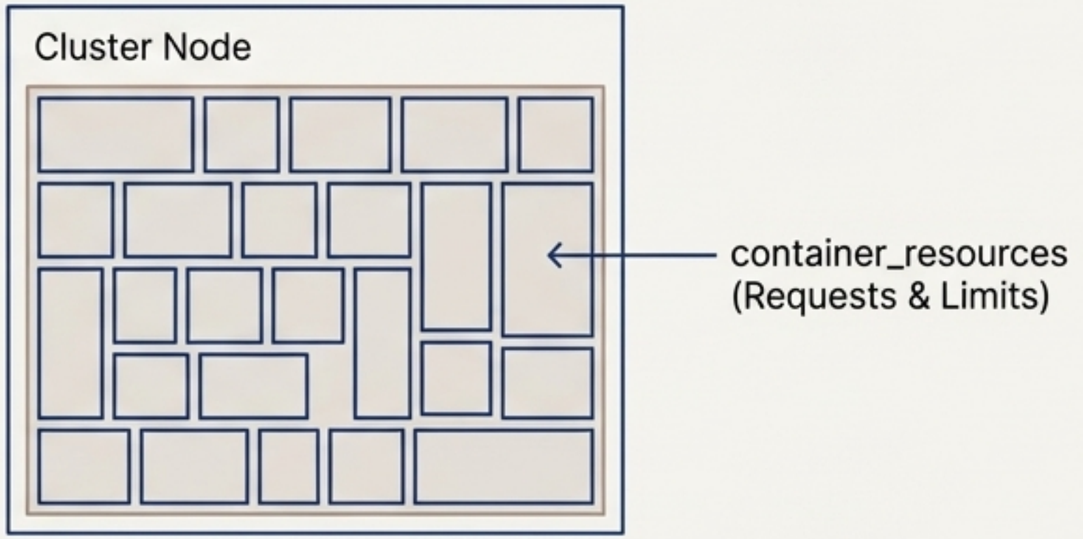
`modules/App_CloudRun/service.tf`
`modules/App_CloudRun/service.tf`


Inter
min = 0 (Lowest cost, cold starts) vs. min >= 1 (Always warm, higher baseline cost).

Inter
Prevents runaway scaling costs under unexpected load.

## GKE Autopilot

`modules/App_GKE/deployment.tf`
`modules/App_GKE/deployment.tf`



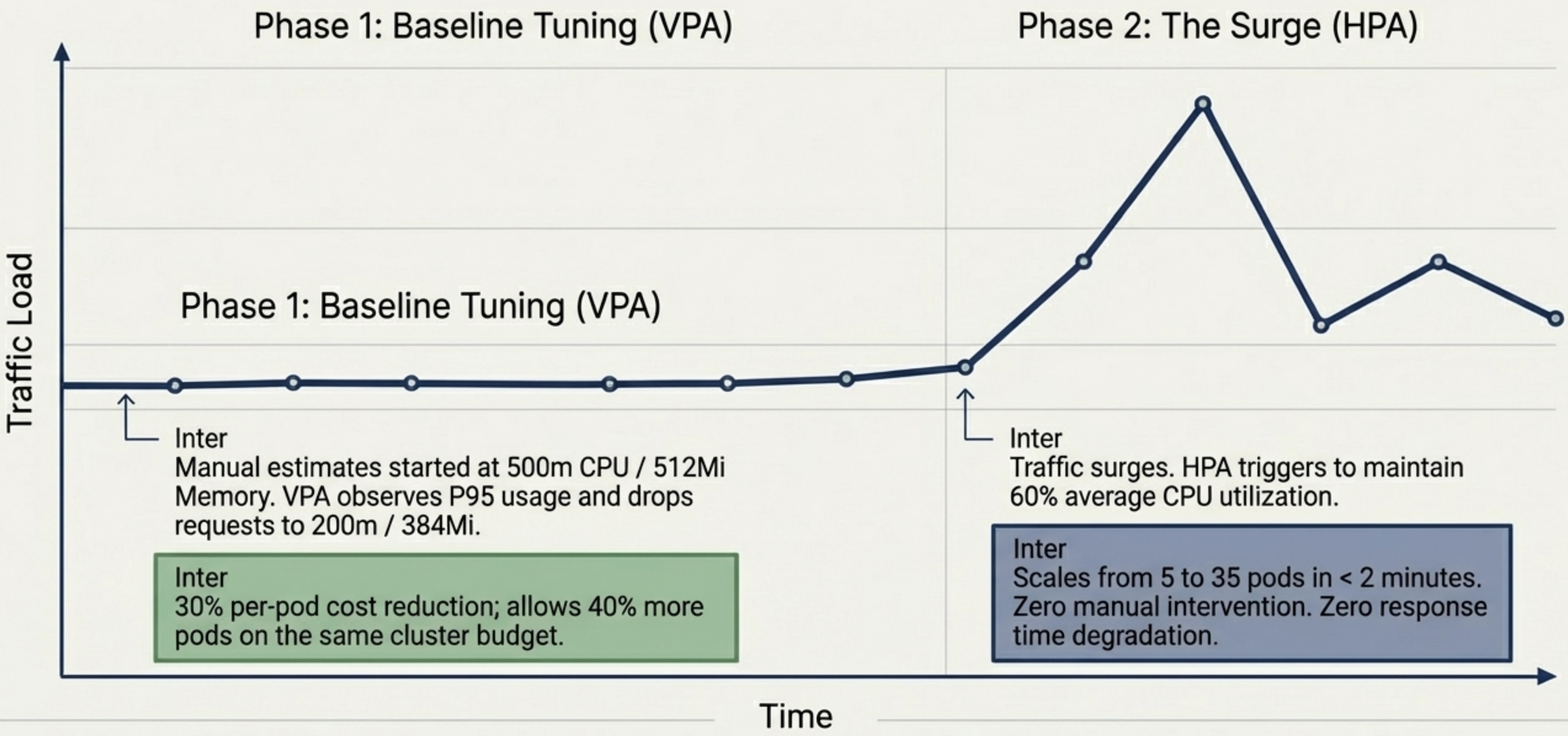

Inter
The cluster provisions underlying node capacity strictly to satisfy the aggregate requests of all scheduled pods.

# HPA vs. VPA: Scaling Strategies Compared

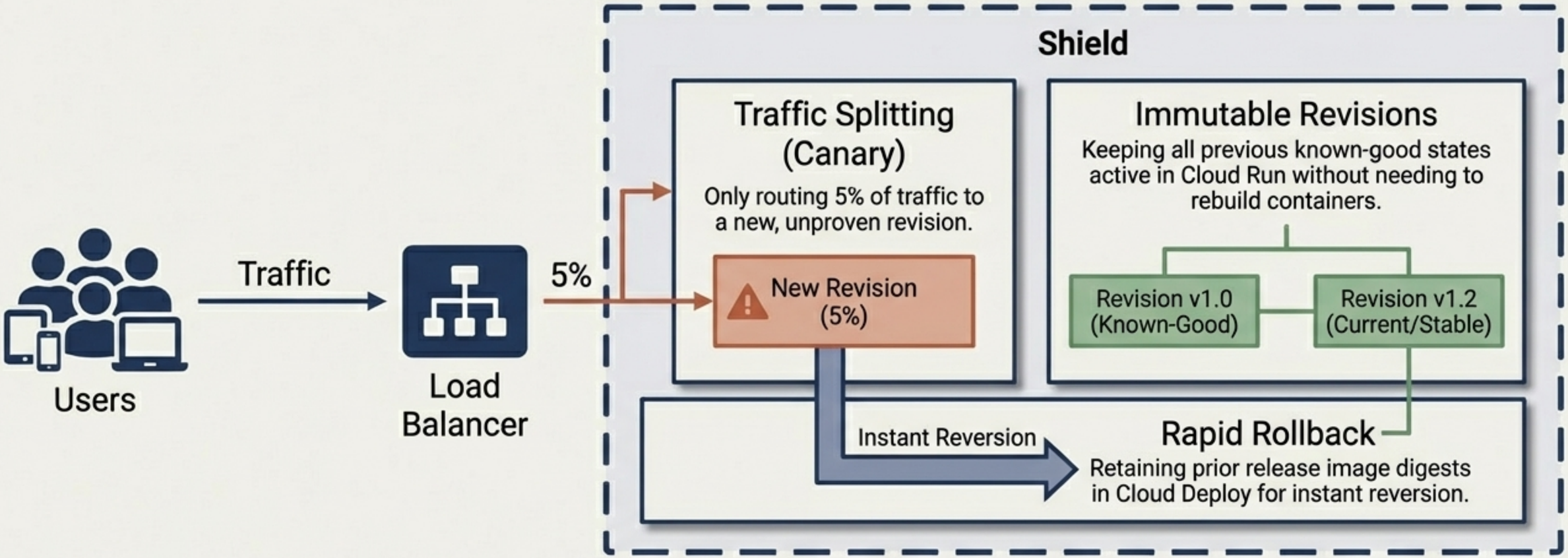
| |  Horizontal Pod Autoscaler (HPA) |  Vertical Pod Autoscaler (VPA) |
|---|---|---|
| Dimension Scaled | Number of pod replicas | CPU/Memory resource requests per pod |
| Trigger | Observed CPU/memory utilization relative to targets | Historical resource consumption analysis |
| Primary Value | Distributing unexpected traffic loads | Right-sizing capacity to prevent over-provisioning waste or under-provisioning evictions |
| Console Location | `Workloads > Details > Autoscaling` | `Workloads > Observability charts vs. YAML resources.requests` |

VPA and HPA address completely different scaling dimensions and can run simultaneously.

# Scaling in Action (Real-World E-Commerce)



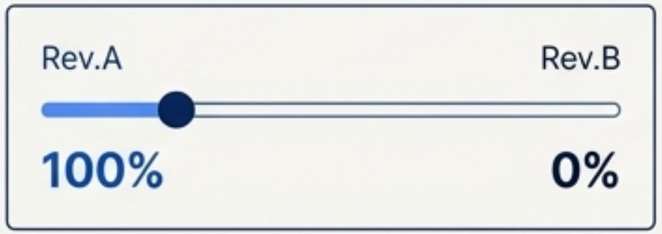
# The Blast Radius Framework




Reliability isn't about preventing all failures; it's about minimizing the percentage of users affected and the seconds it takes to recover.

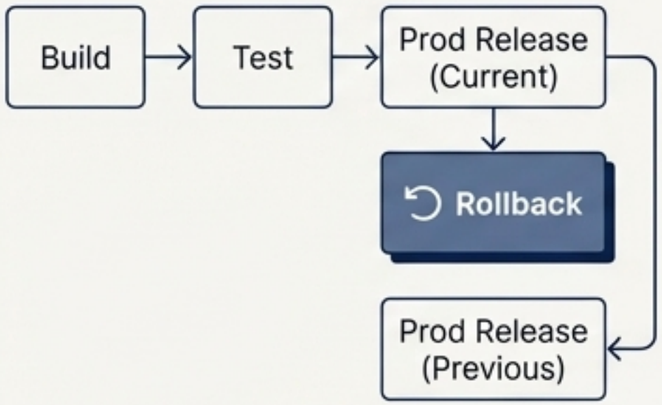
# Routing & Rollbacks in GCP

## Cloud Run (Manage Traffic)




Redistribute traffic between revisions in under 30 seconds without redeploying.

## Cloud Deploy (Rollouts)



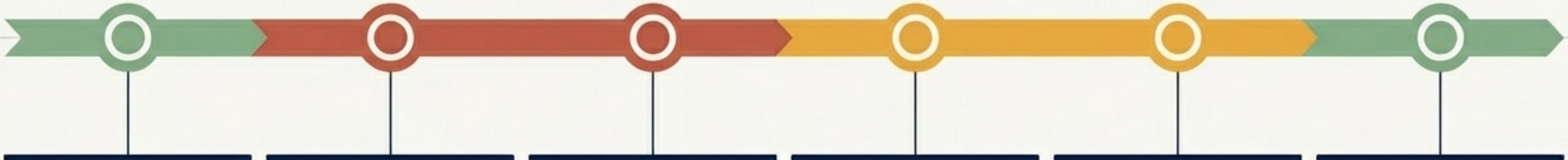

Re-targets the prior image digest instantly; bypasses the Cloud Build execution phase.

## GKE (Rolling Updates)

```
kubectl rollout undo deployment/<name>
```

Reverts to the previous ReplicaSet if incremental pod replacements show errors.

# Anatomy of a Mitigation



### T=14:37 (Failure)

New Cloud Run payment revision deployed. Returns HTTP 500s for 3% of requests.

### T=14:38 (Detection)

Error budget burn rate hits 60x sustainable rate.

### T=14:39 (Alerting)

Fast-burn SLO alert triggers. PagerDuty notifies on-call engineer.

### T=14:43 (Intervention)

Engineer navigates to Cloud Run > Revisions, clicks clicks Manage Traffic.

### T=14:44 (Mitigation)

100% of traffic shifted to prior revision in 20 seconds. Error rate drops to 0.

Total user impact: 7 minutes.

### T=14:50+ (Investigation)

Logs Explorer reveals missing DB index on a new query.

# The War on Toil

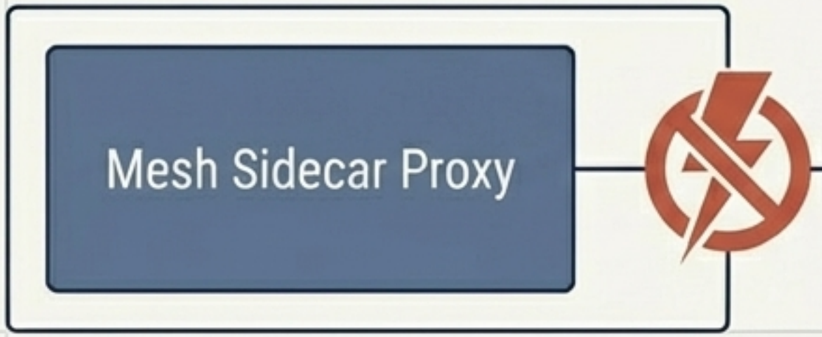
| Manual Toil | GCP Automation Solutions | SRE Value |
| --- | --- | --- |
| Manually updating image tags in manifests. | Cloud Deploy automation rules. | Eliminating repetitive, non-value-adding operational work so engineers can focus on systemic reliability improvements. |
| Manual deployment approvals. | Cloud Scheduler jobs. | |
| Responding to predictable scaling events manually. | HPA / VPA Autoscalers. | |

# Proactive Failure Testing in GCP

## Method 1: Cloud Run Traffic Splitting


Deliberately routing to a slow revision to monitor frontend timeout handling.

## Method 2: Cloud Service Mesh Fault Injection




Navigate to Cloud Service Mesh > Traffic Management.

**Faults:**
Inject HTTP Delays (latency) or HTTP Abort Codes (errors) into specific downstream traffic paths to verify graceful degradation.

# The Blameless Post-Mortem

## The 3 Pillars of a Blameless Post-Mortem

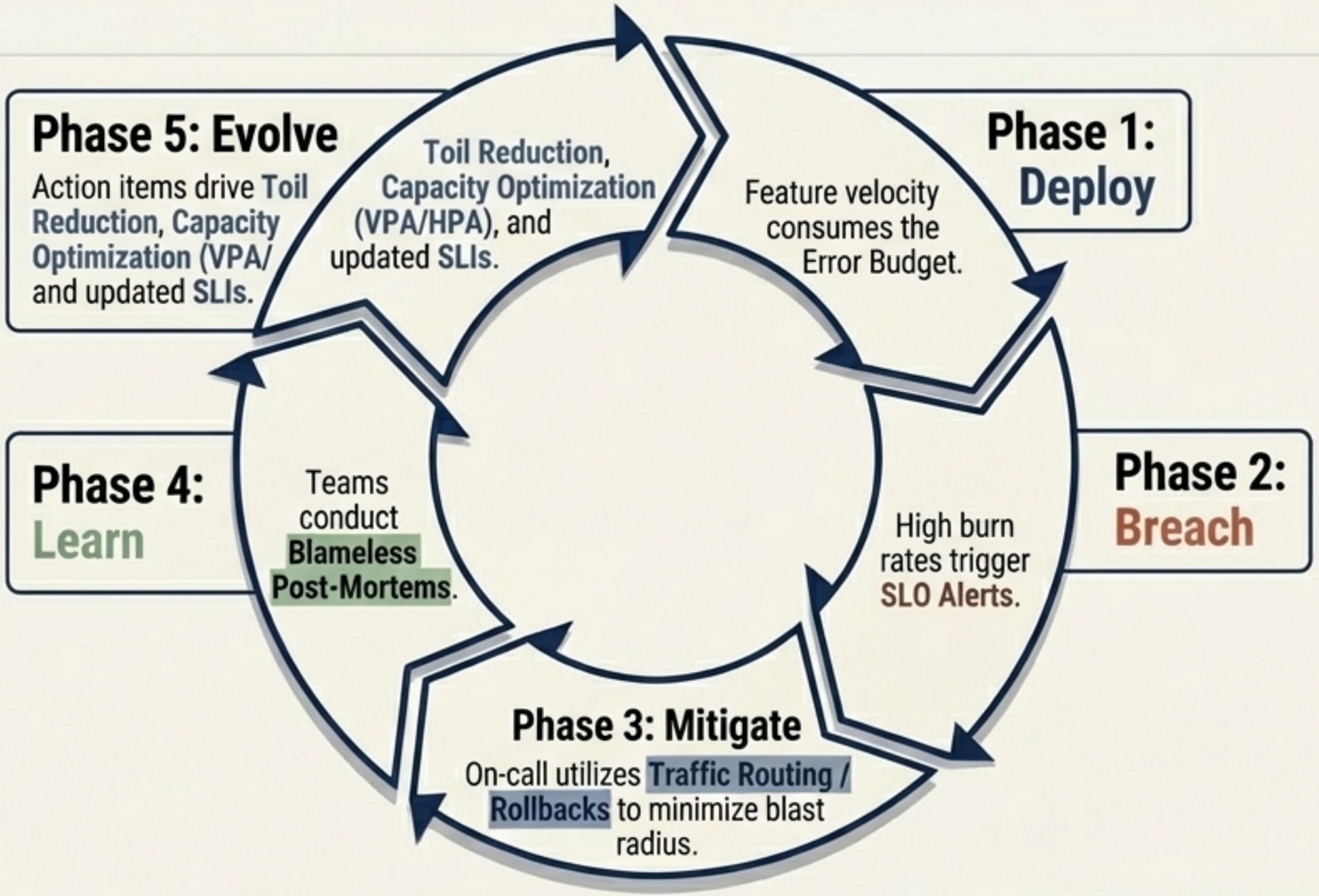
1. **The Timeline**: Objective sequence of events from symptom detection to resolution.
2. **Contributing Factors**: Crucial SRE rule: **There is never a single root cause**. Complex systems fail due to multiple contributing factors.
3. **Action Items**: Specific remediations with assigned owners and deadlines to prevent recurrence.

## The Tool: Cloud Logging > Log Analytics

Run SQL queries over historical log data to objectively construct the incident timeline and isolate system state changes.

```
SELECT timestamp,
       resource.labels.pod_name,
       severity,
       textPayload
FROM `google.cloud.audit.log/`
WHERE resource.type = 'k8s_pod' AND
    timestamp BETWEEN '2023-10-27T10:00:00Z'
      AND '2023-10-27T11:00:00Z'
ORDER BY timestamp ASC
LIMIT 50;
```

# Synthesis: The SRE Continuous Loop




**Reliability is not a static state; it is a continuously engineered loop balancing velocity with operational risk.**