Google Cloud

Next '24

How ANZ delivers reliable applications on Google Cloud using archetype-based platforms

Hello Next !

- Joined ANZ in **2015**
- 12 years as a hands-on Software Engineer
- 5 years as a leader in Cloud Platforms
- Live in Melbourne, Australia
- Passionate about putting the ‘dev’ in devops
- Avid gardener & sports fan!

Daniel McKeown

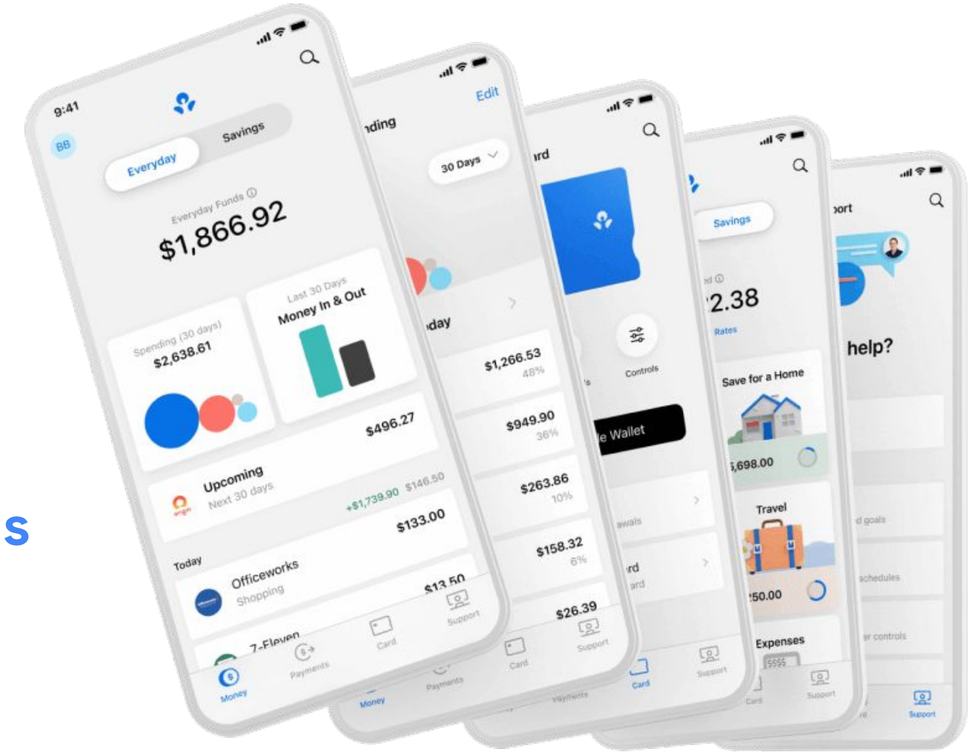
Area Lead Cloud & Orchestration Platforms, ANZ Bank

https://www.linkedin.com/in/**danielmckeown**/

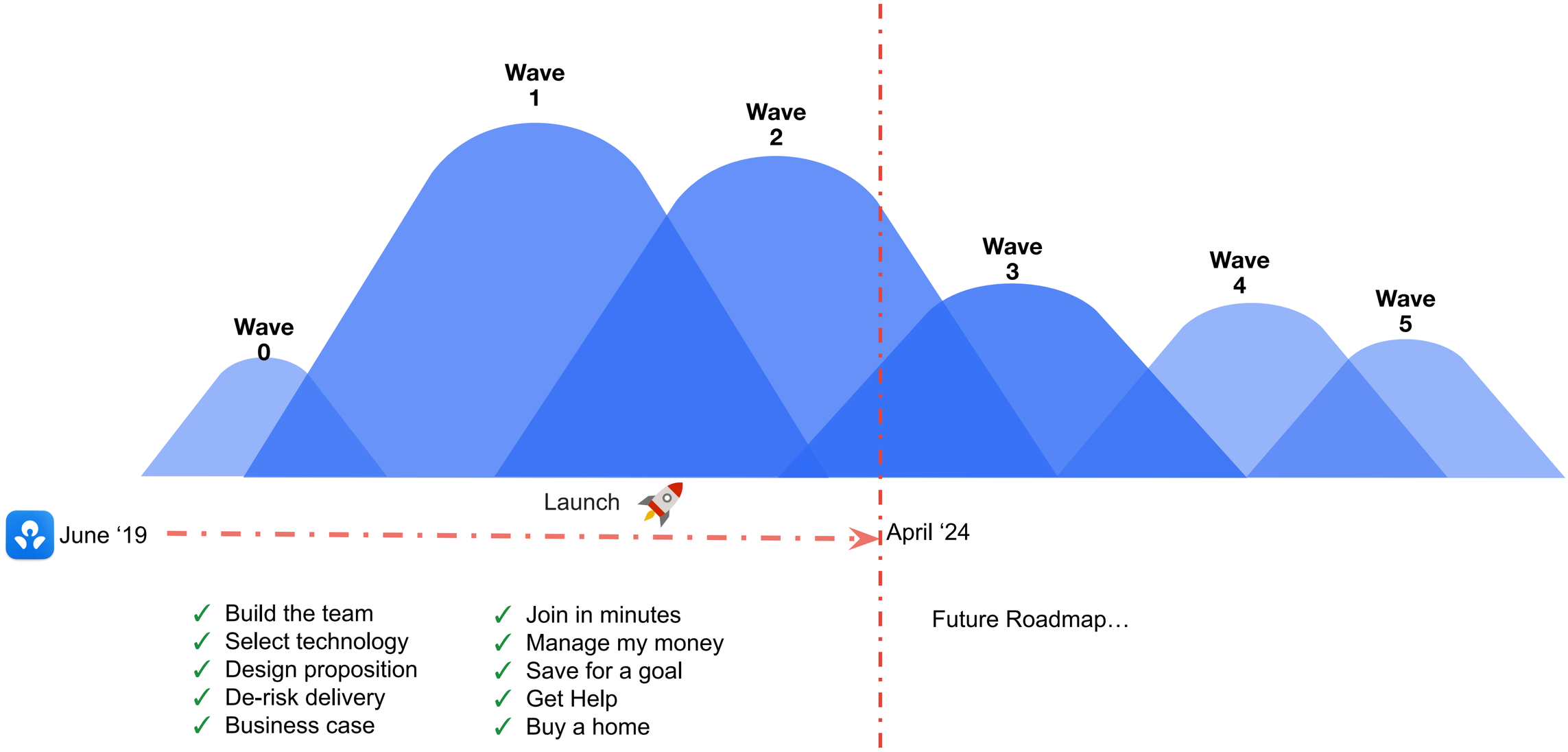
Building a cloud-native digital bank

- ANZ is an 186 year old Australian Bank
  - $77.1B in market capitalisation
  - 40K employees
  - $1.1T in funds under management
  - 8.5M customers
  - Operations in 29 countries

- Building a new digital banking proposition called **ANZ Plus**
  - New mobile and web applications
  - Simplified product offering
  - Cloud-native and hosted on Google Cloud



We are part way through a multi-year transformation



- ✓ Build the team
- ✓ Select technology
- ✓ Design proposition
- ✓ De-risk delivery
- ✓ Business case

- ✓ Join in minutes
- ✓ Manage my money
- ✓ Save for a goal
- ✓ Get Help
- ✓ Buy a home

Future Roadmap…


Access
card
Limits
Settle Home Loan
Digital wallet
Refinance their home loan
Set up account
Money in money out
Card blocks
Explore the loan proposition
Activate Card
In app Card details
View and accept loan offer
Set up offset accounts
Pay off and close home loan
Appointment Booking
Manage their loan
Manage Payees
Transfer money BSB and ACCT#
Manage loan book pricing
Step up ID
Manage Pin
See interest rates
Fitbit Pay
Verify identity
Round Ups
Access a refinance coach
Loans
Spending Limits
Submit property insurance
Deceased Estates
Purchase a property and advise ANZ
Upcoming Expenses
Automated credit risk assessment
Power of Attorney
Create PayID
Share and export transactions
Financial wellbeing nudges
Finalise loan
Coaches workbench
Calculate, display and manage charges and interest on loans
Voice
Complaints
Exit a customer
Transact & Save
Replace Card
BPAY
Create Digital Identity
Recover credentials
Receive Card
NPS
Customer can be uniquely identified
WFM
International Money Transfer
Garmin Pay
Detect fraud, scams and identity events
Open Banking
Manage transaction categories
Virtual Goals
Install the app
Welcome pack
Disputes
Social Media
Collections
OB data out obligations
Migrated from ANZ
Set up account
Observability of systems
Assess customer risk profile
Update personal information
View and manage their portfolio
Migrate
Statements
Complete and application
Publish to market
Coach evaluation
Chat
Coach
Content as a service
Leads
Process outstanding payments
Fit for purpose platforms, developer tooling and frameworks
Send notifications and nudges
Close Account
Split payments
Access a coach
Explore offers
Brokers
Journey support
PayID transfers
Wellbeing insights
Dashboards metrics and reporting
View and manage their financial position
Block gambling transaction
Centralised pricing management
Waitlist
Marketplace Add ons
Approval Confidence
Accreditation with ANZ
Apra Compliance
Broker Managed settlement
Vulnerable Access support
Hardships support
Security controls
Central product management
Credit oversight
QA program
Get help on 'classic' applications

ANZ Plus **scale** on Google Cloud

- 2500 **Deployments**
- 1700 **Namespaces**
- 1500 **Cloud Run Services and Jobs**
- 1000 **Cloud Functions**
- 1200 **Code repositories**

...

Steve McGhee

Reliability Advocate, Google Cloud

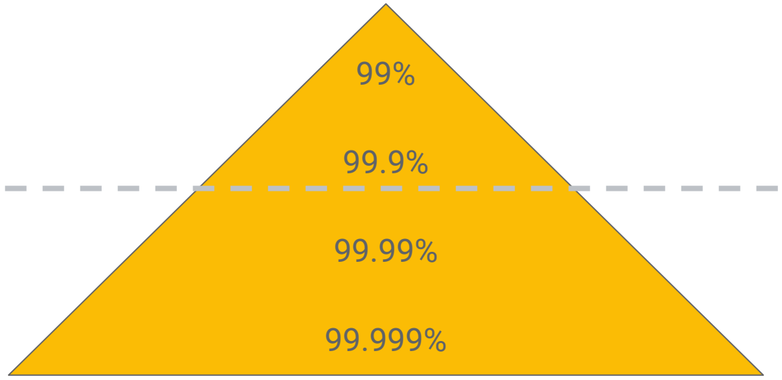
Ameer Abbas

Outbound Product Manager, Google Cloud

Can you build 99.99 services on 99.9 infrastructure?

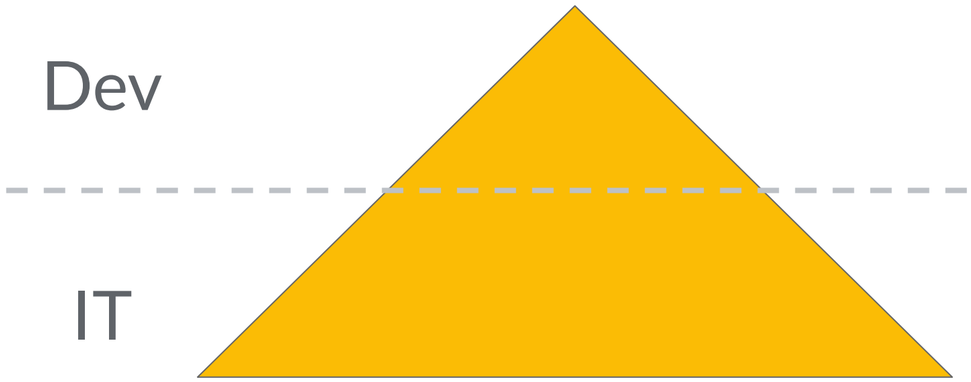
Dev

IT



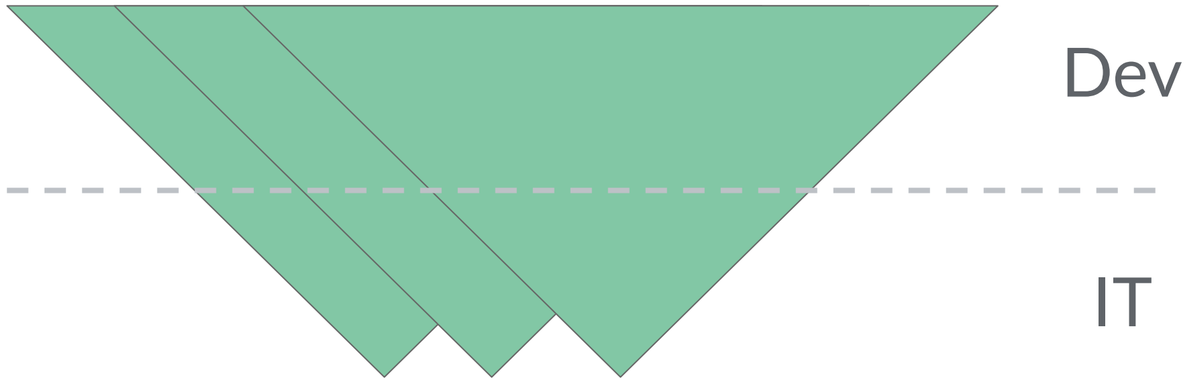
Worked great, for a long time

Common mental model



This works!

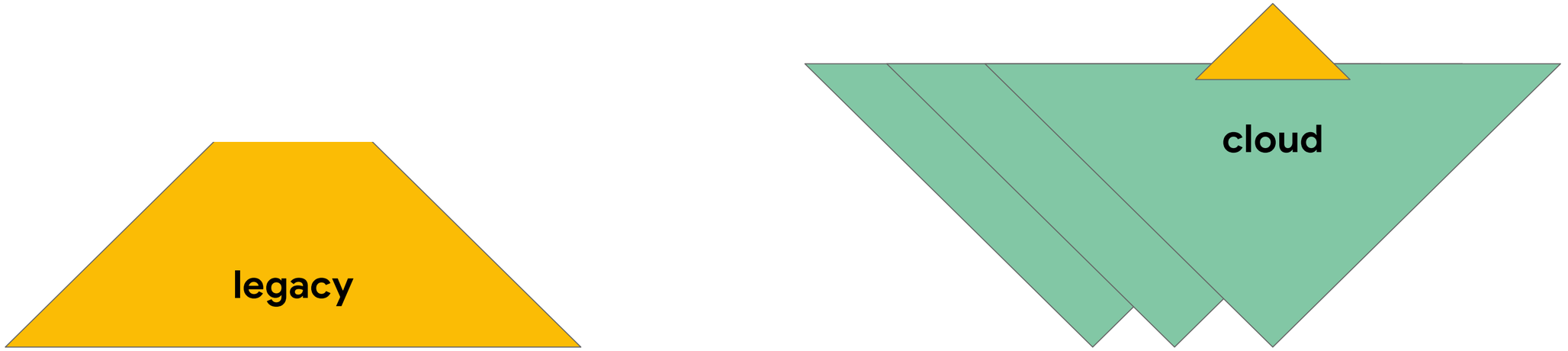
"Makes sense"



Cloud is here, though.

Because: Scale

A common failure mode we've all seen:



Infrastructure changes **can't fix** the app.

Platforms, Personas, and Outcomes

"Are we getting better?"

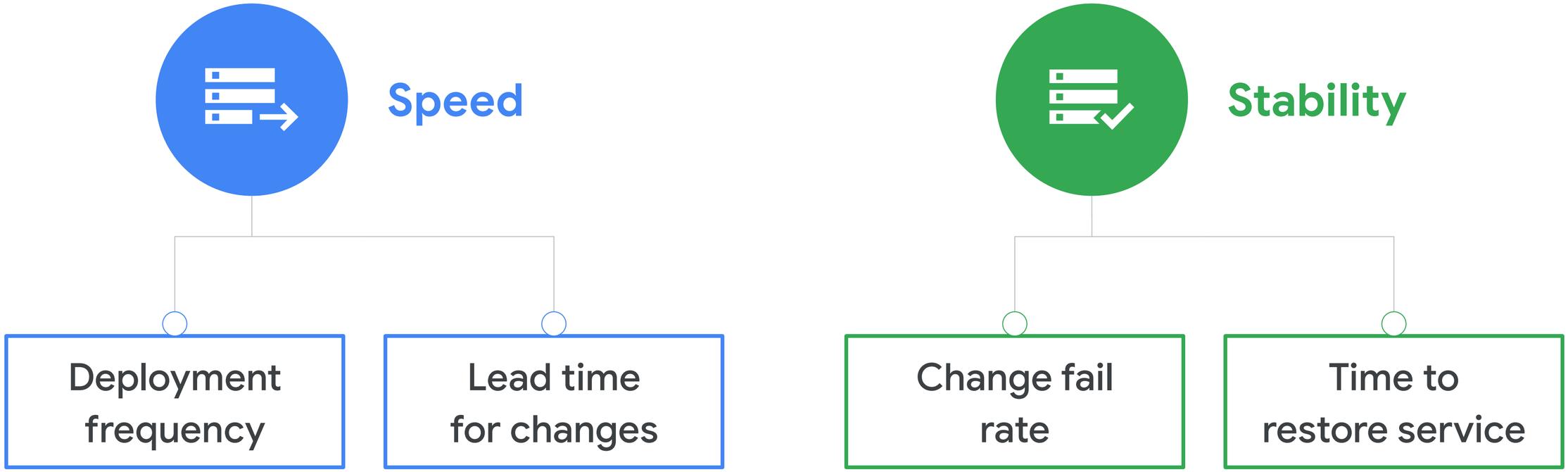
Speed

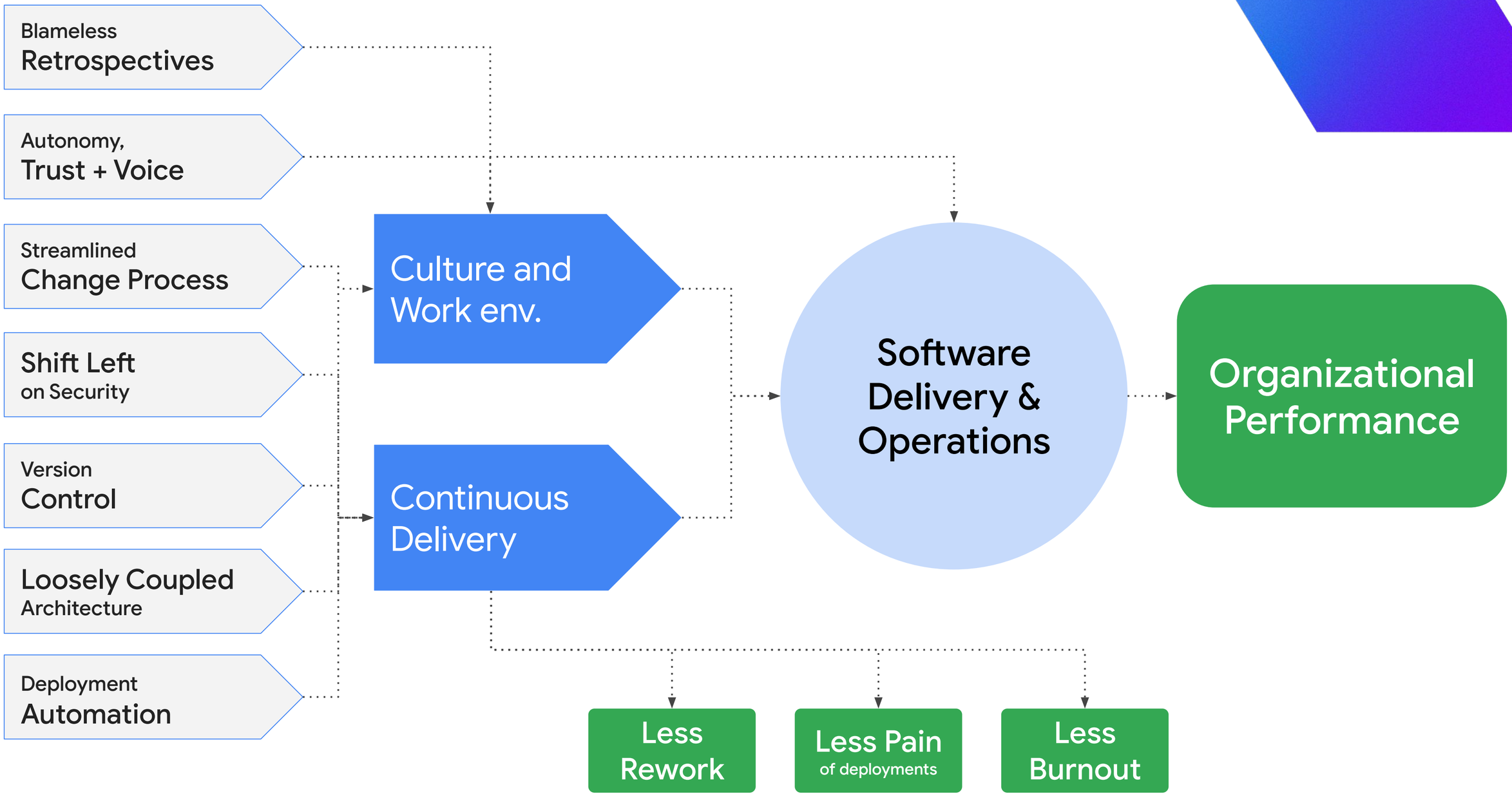
Stability

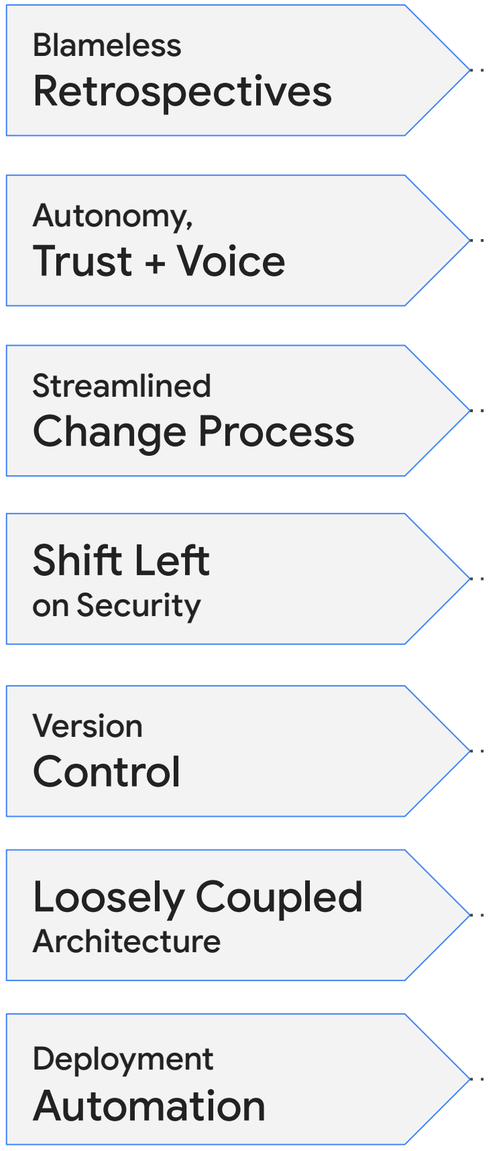
DORA
DEVOPS RESEARCH & ASSESSMENT

"Are we getting better?"

The 4 DORA Metrics




Blameless
Retrospectives
Autonomy,
Trust + Voice
Streamlined
Change Process
Shift Left
on Security
Version
Control
Loosely Coupled
Architecture
Deployment
Automation
Culture and
Work env.
Continuous
Delivery
Software
Delivery &
Operations
Organizational
Performance
Less
Rework
Less Pain
of deployments
Less
Burnout



Capabilities

- technical
- process
- cultural

→ build, buy, or adopt

Measure

1. Measure 4 DORA metrics

2. Determine bottleneck(s)

3. Choose **capability** to improve next

Improve

4. Build / buy capabilities

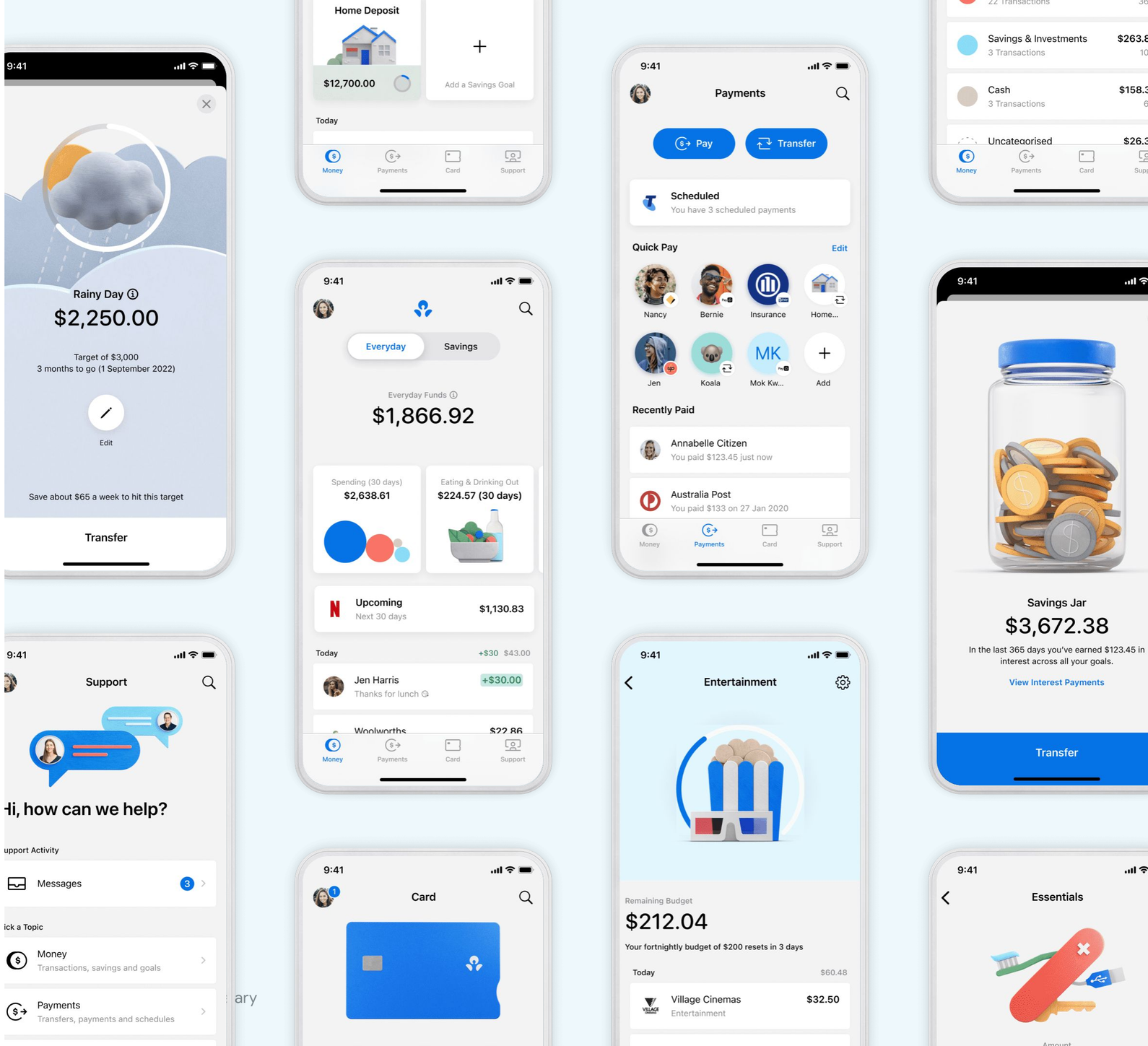
5. Enable capability in platform, document

6. Gather early dev feedback

7. Release next version of platform

Measuring **success**

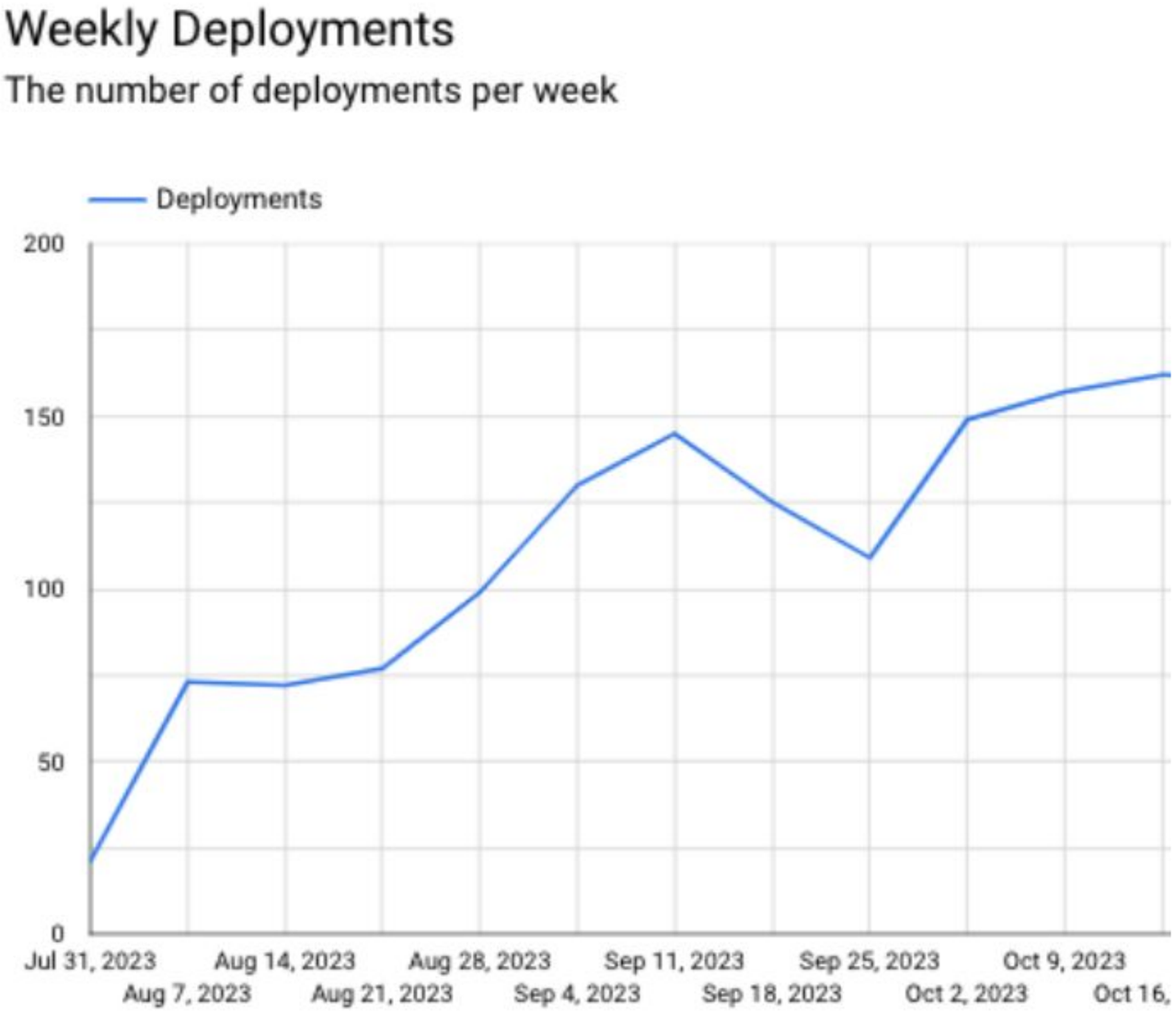
- To know what good looks like, you must first establish **baseline** measures
- Measures aren't only useful for building **platforms**
- We use the **four key** DORA metrics at ANZ

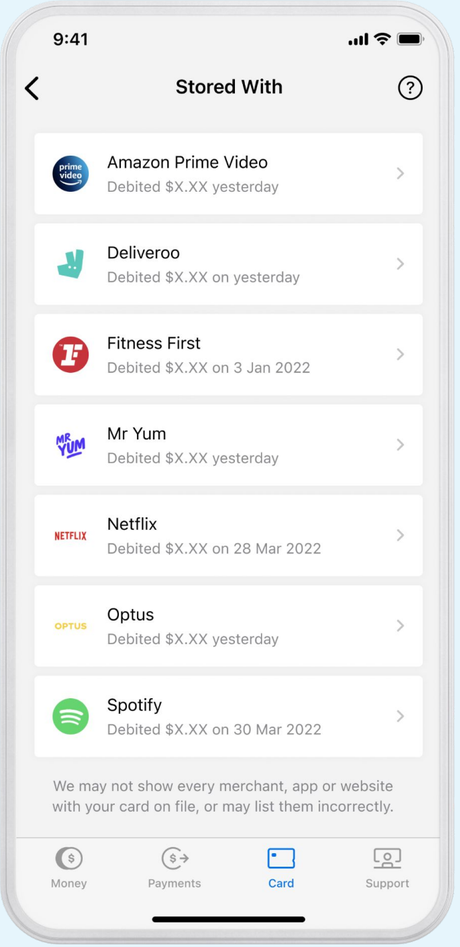


How have we done it?

- Scheduled batch loads of historical data from **Jira**, **GitHub** and **Service Now**
- **dbt** for ETL
- **Big Query** as our Data Warehouse
- **Looker Studio** as our Data Presentation layer

Platforms drive devops health



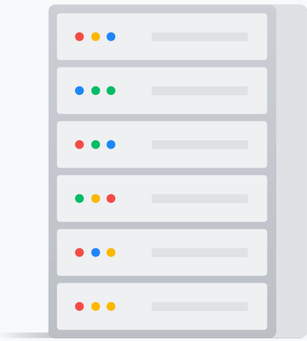


Platform Engineering

More focus
More creativity
More agility

Software Developer

Platform Engineer



Google Cloud Resources

Infrastructure
Dev and Prod

Public Cloud

Platform Engineering

More focus
More creativity
More agility

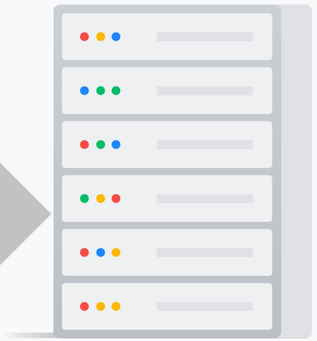
Software Developer

SSH / UI / API

Internal Developer Platform

Platform Engineer

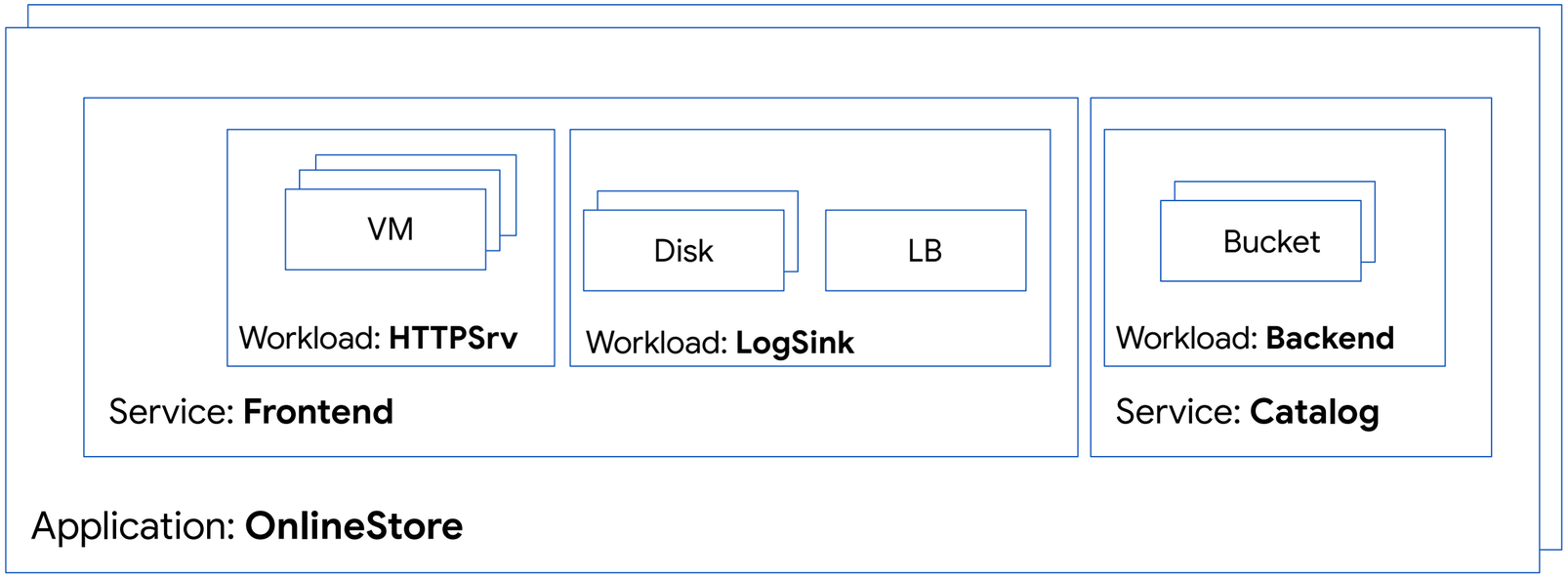
Golden Paths



Google Cloud Resources

Infrastructure
Dev and Prod

Public Cloud


VM
Workload: HTTPSrv
Disk
LB
Workload: LogSink
Service: Frontend
Bucket
Workload: Backend
Service: Catalog
Application: OnlineStore

VM

Workload: **HTTPSrv**

Disk

LB

Workload: **LogSink**

Service: **Frontend**

Bucket

Workload: **Backend**

Service: **Catalog**

Application: **OnlineStore**

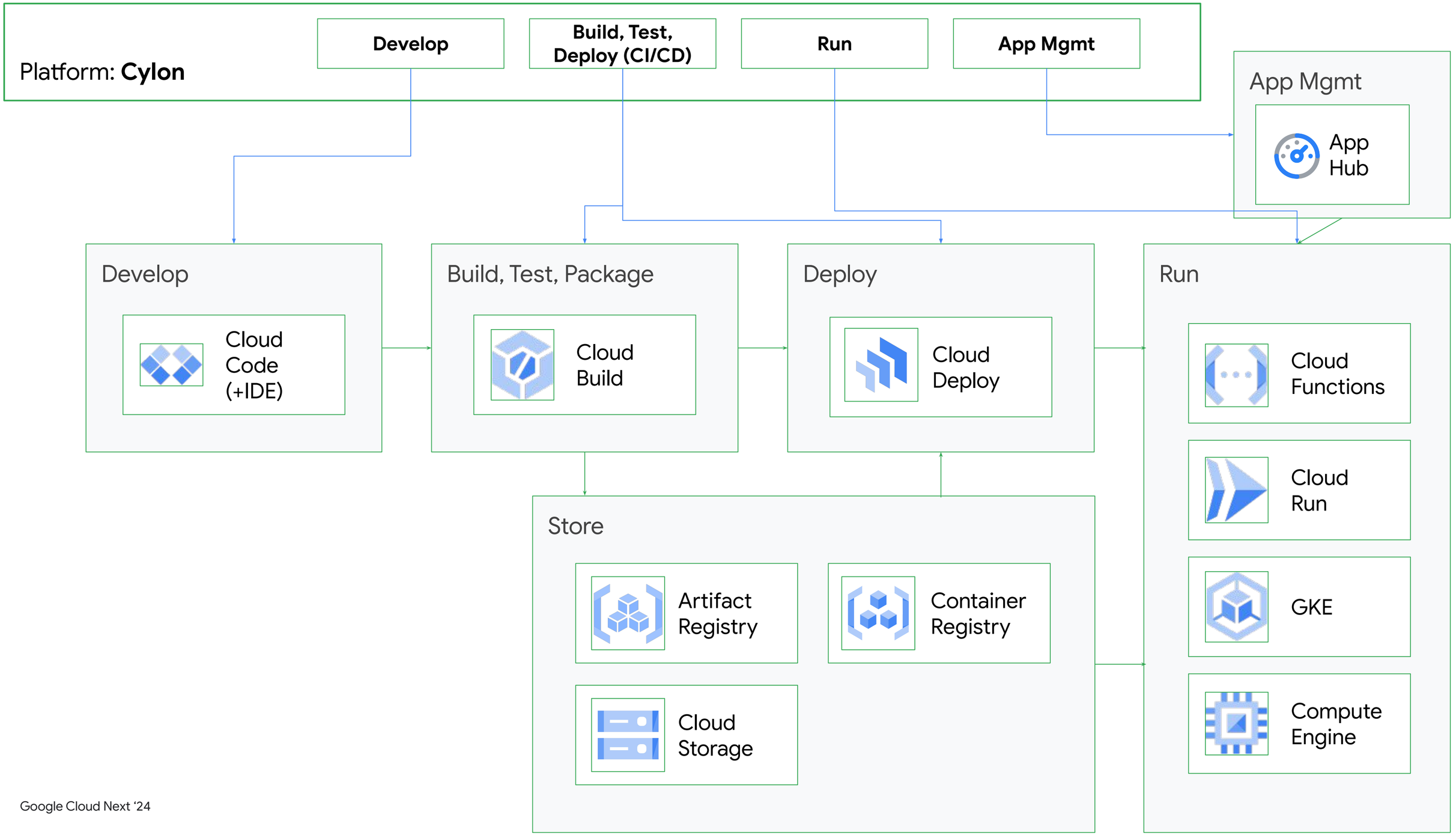
Platform: **Cylon**

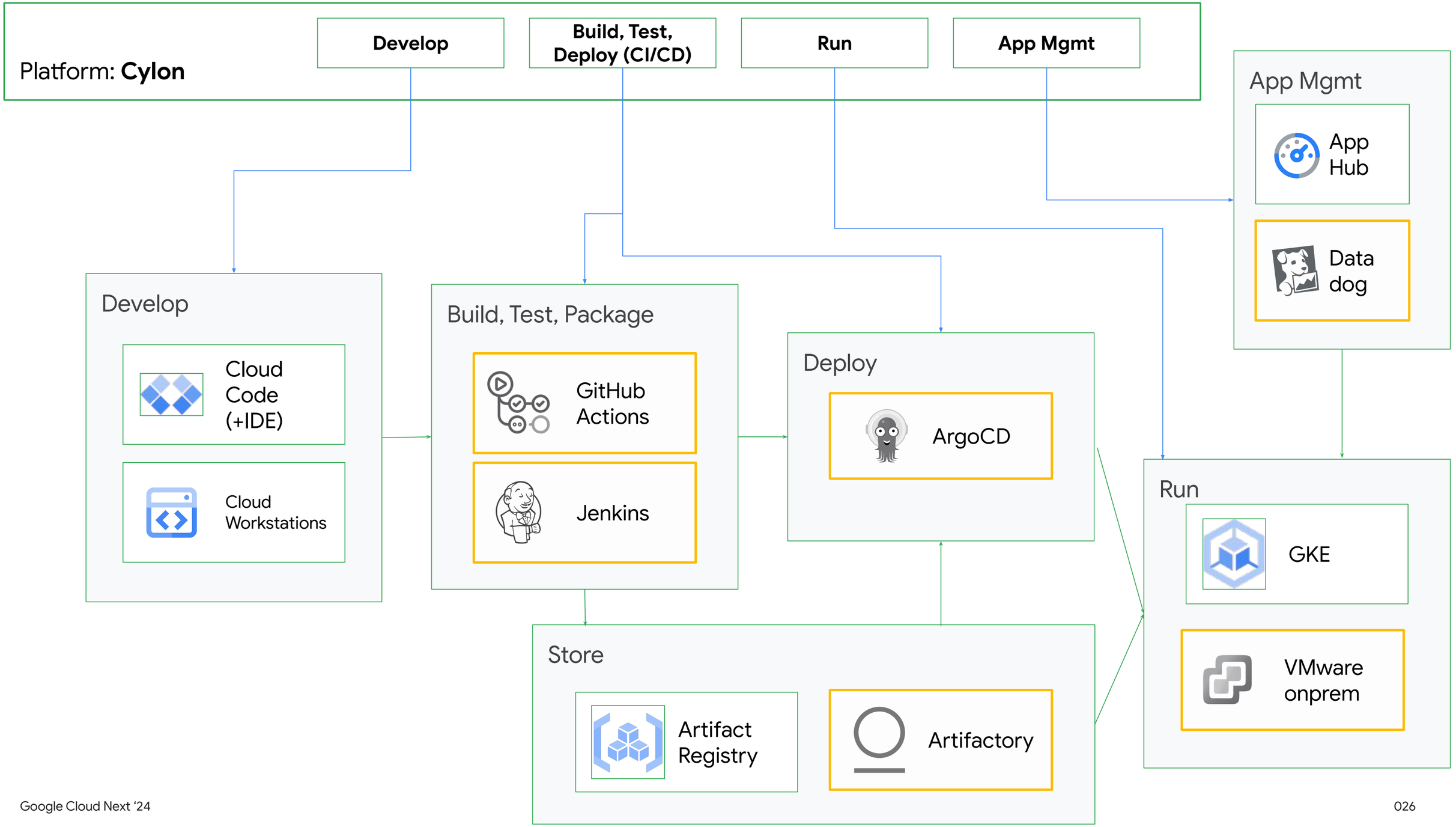
Develop

Build, Test, Deploy (CI/CD)

Run

App Mgmt


Platform: Cylon
Develop
Build, Test, Deploy (CI/CD)
Run
App Mgmt
App Mgmt
App Hub
Develop
Cloud Code (+IDE)
Build, Test, Package
Cloud Build
Deploy
Cloud Deploy
Run
Cloud Functions
Cloud Run
GKE
Compute Engine
Store
Artifact Registry
Container Registry
Cloud Storage


Platform: Cylon
Develop
Build, Test, Deploy (CI/CD)
Run
App Mgmt
App Mgmt
App Hub
Data dog
Develop
Cloud Code (+IDE)
Cloud Workstations
Build, Test, Package
GitHub Actions
Jenkins
Deploy
ArgoCD
Run
GKE
VMware onprem
Store
Artifact Registry
Artifactory

We’ve built an API-driven Platform known internally as the X Framework

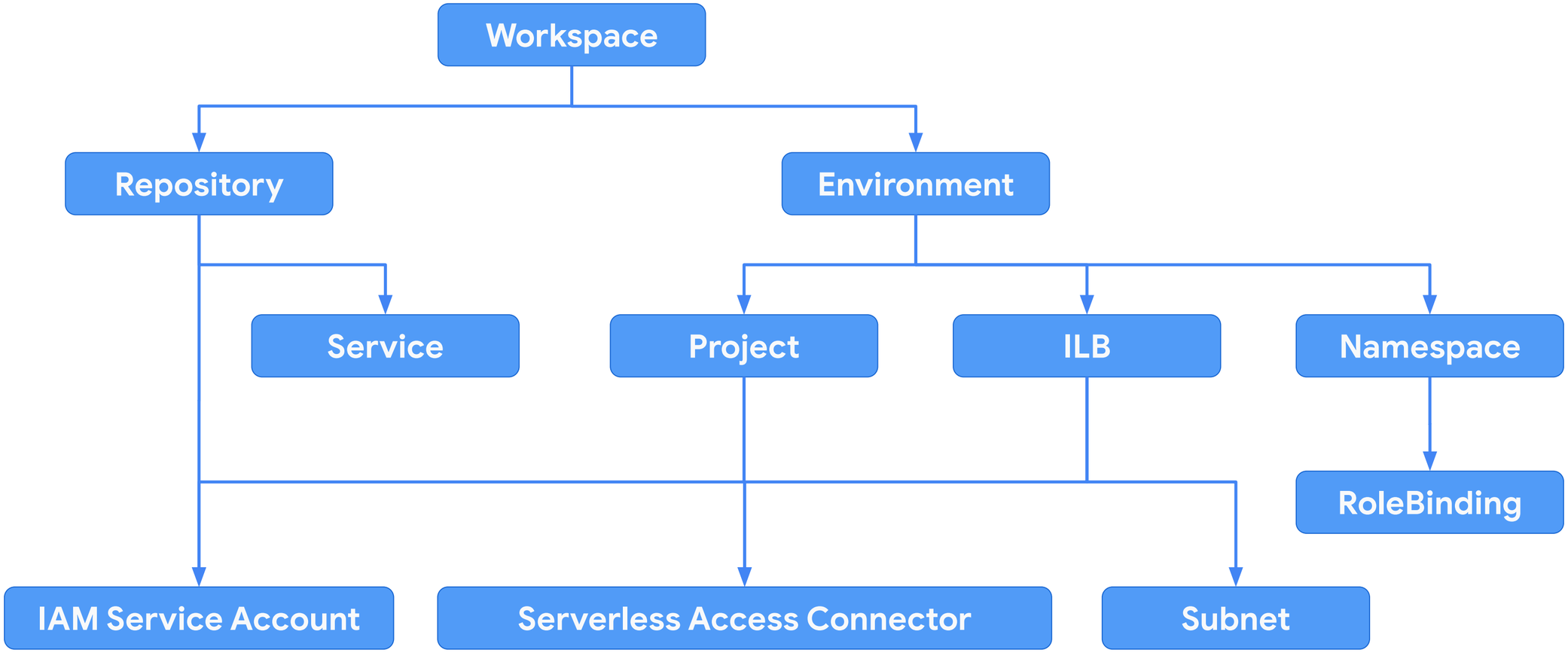
Asset transparency

Reduced developer cognitive load

Reduced operator toil

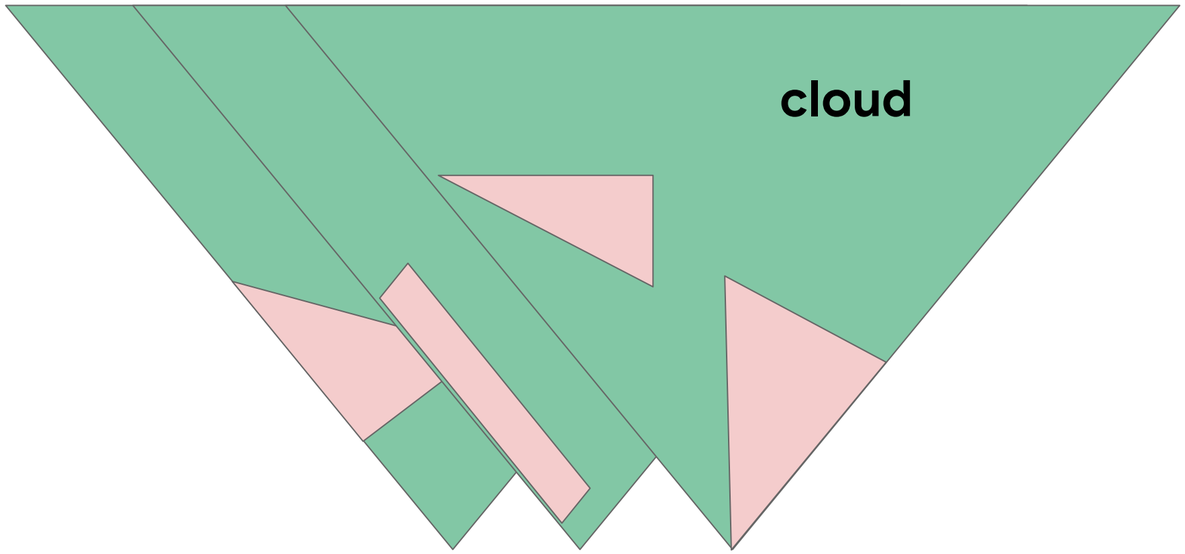
Community Governance

X Framework Domain Model

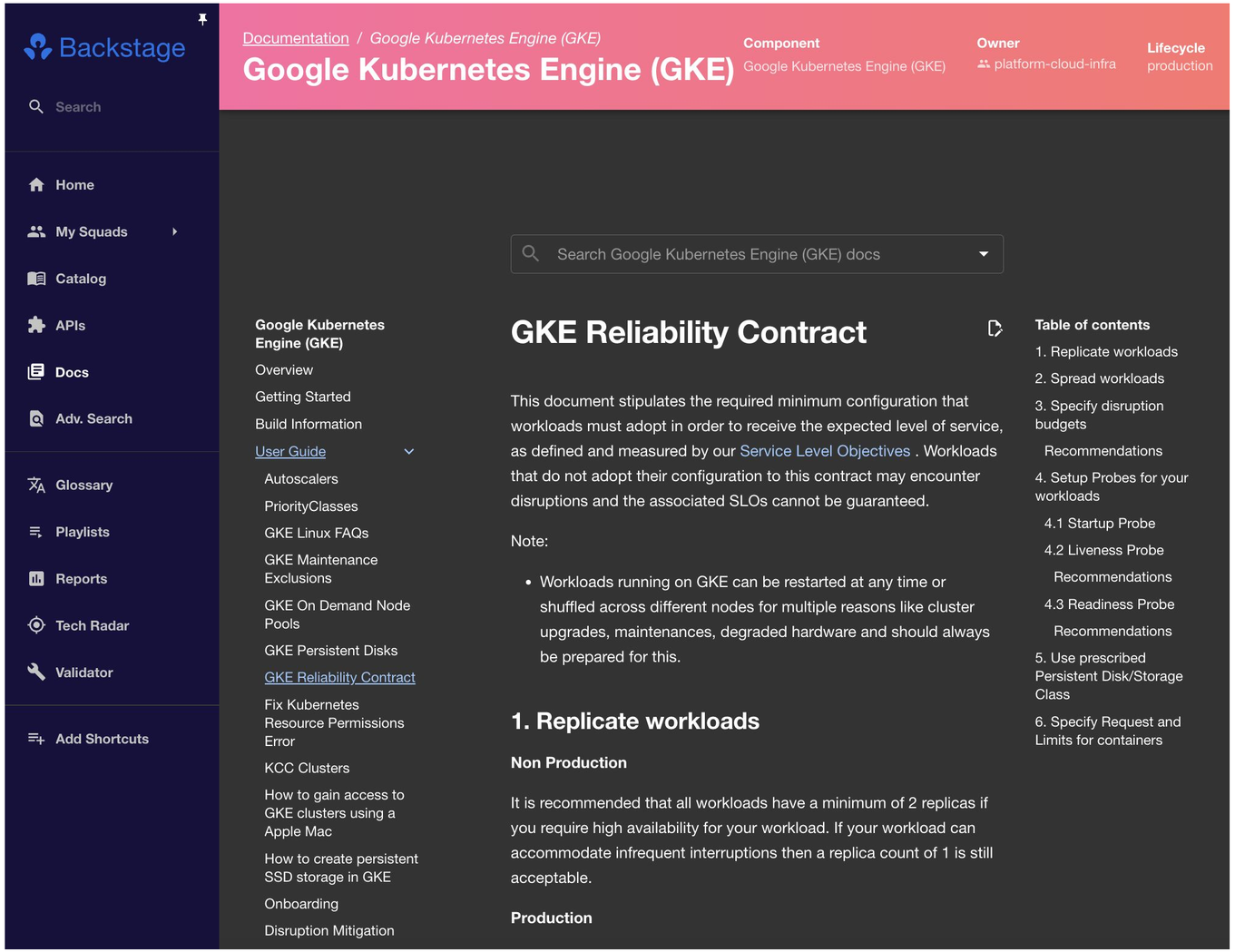


Platforms enable reliable applications

- **Chaos Engineering**
  - You don’t need an off the shelf tool
  - We simulated losing a zone in production
  - Use taints and tolerances move applications between zones
- Non-production clusters use **Spot VMs**
- Stateless **cloud-native** approaches



Platforms enable reliable applications



- Publish a **Reliability Contract**
- Embrace **SRE** Principles
- Keep **competitive tension** between infrastructure **reliability**, **cost** and developer **productivity**

https://sre.google/workbook/slo-document/

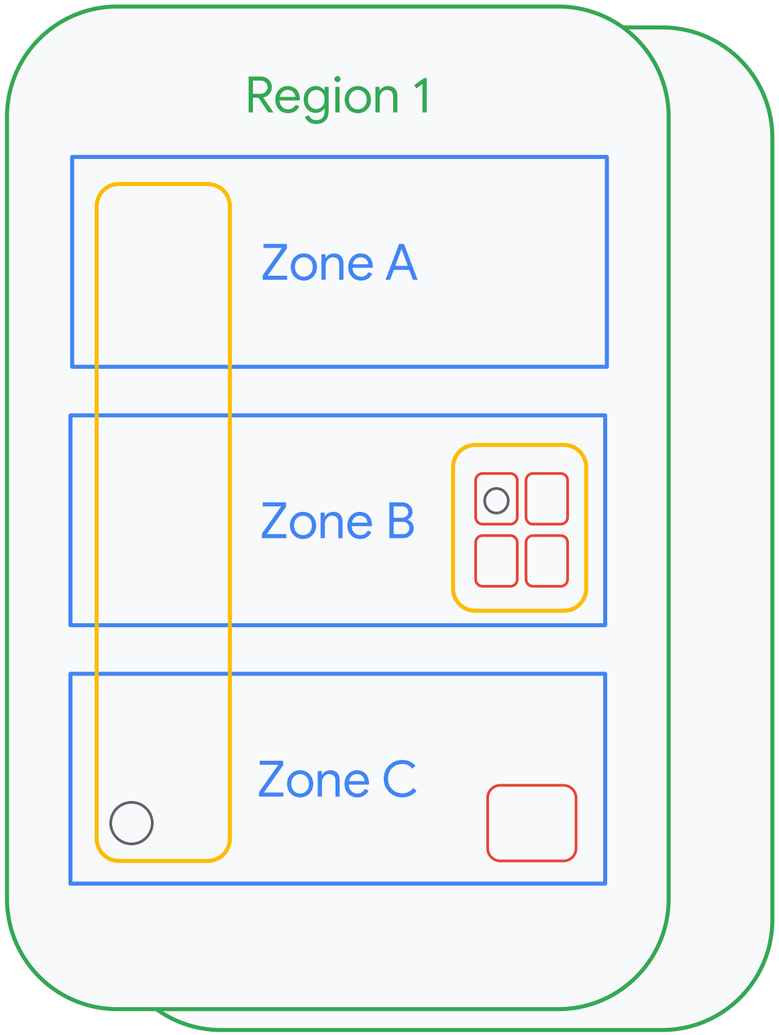
https://sre.google/workbook/error-budget-policy/

Platforms enable reliable applications

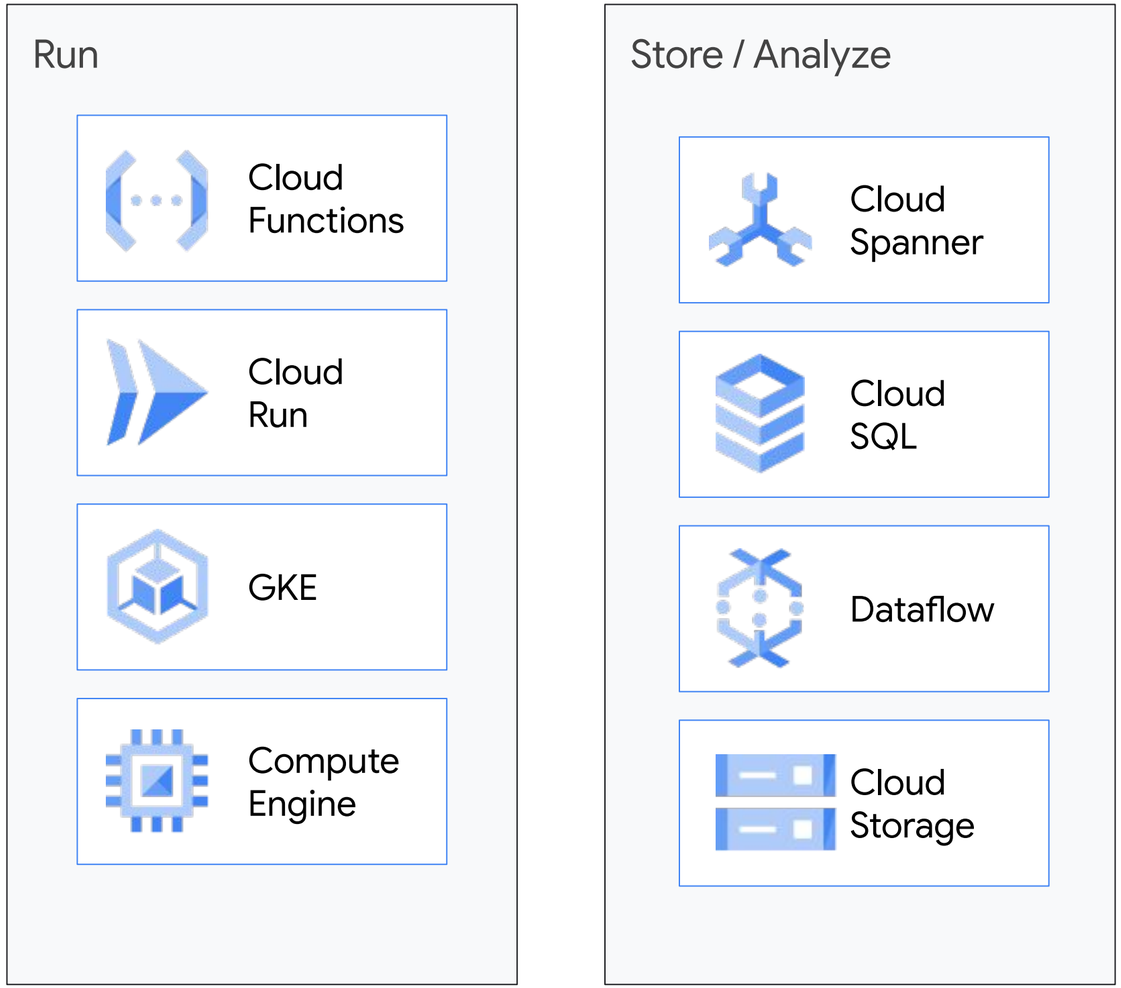
- Only a **single region** in Australia available at launch, second came later.
- Currently finalizing **dual region** design
- Only most critical services will be deployed **active-active**
- We have to **rewrite** some things.
- Extreme reliability is **hard**

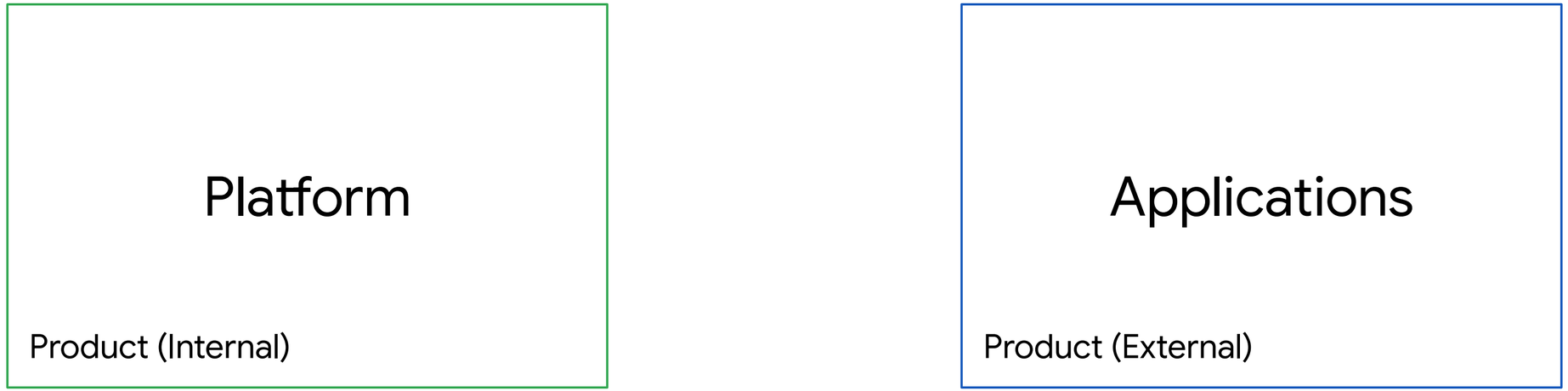
Cloud Failure Domains: Regions, Zones

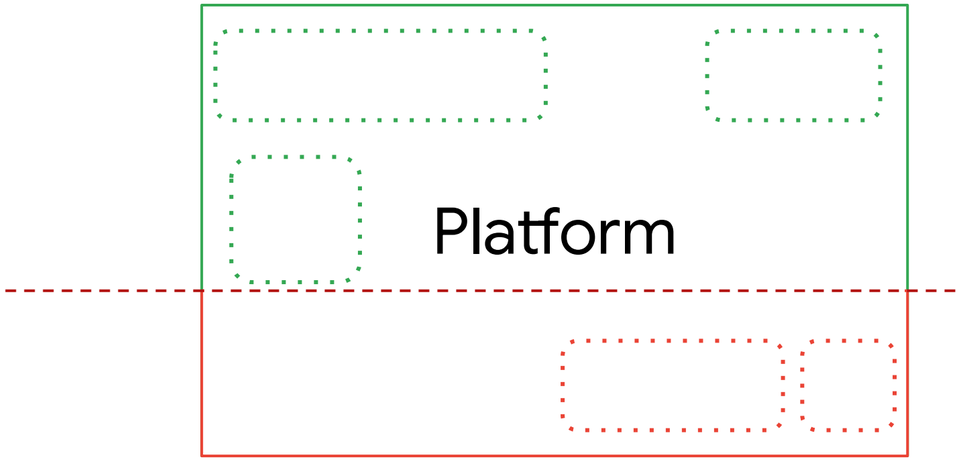
Clusters, Pods, Apps, Functions, VMs

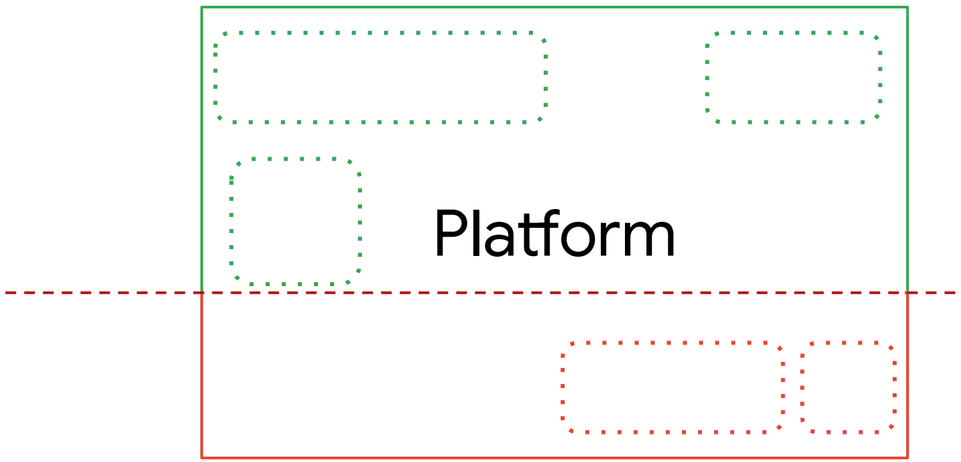


Products expose their failure domains differently




Platform
Product (Internal)
Applications
Product (External)


Platform



- **Critical** platform capabilities deployed as active-active, dual region
- **Observability** stack also critical
- **Feature toggling** is critical - for outage tiles and disabling UI components

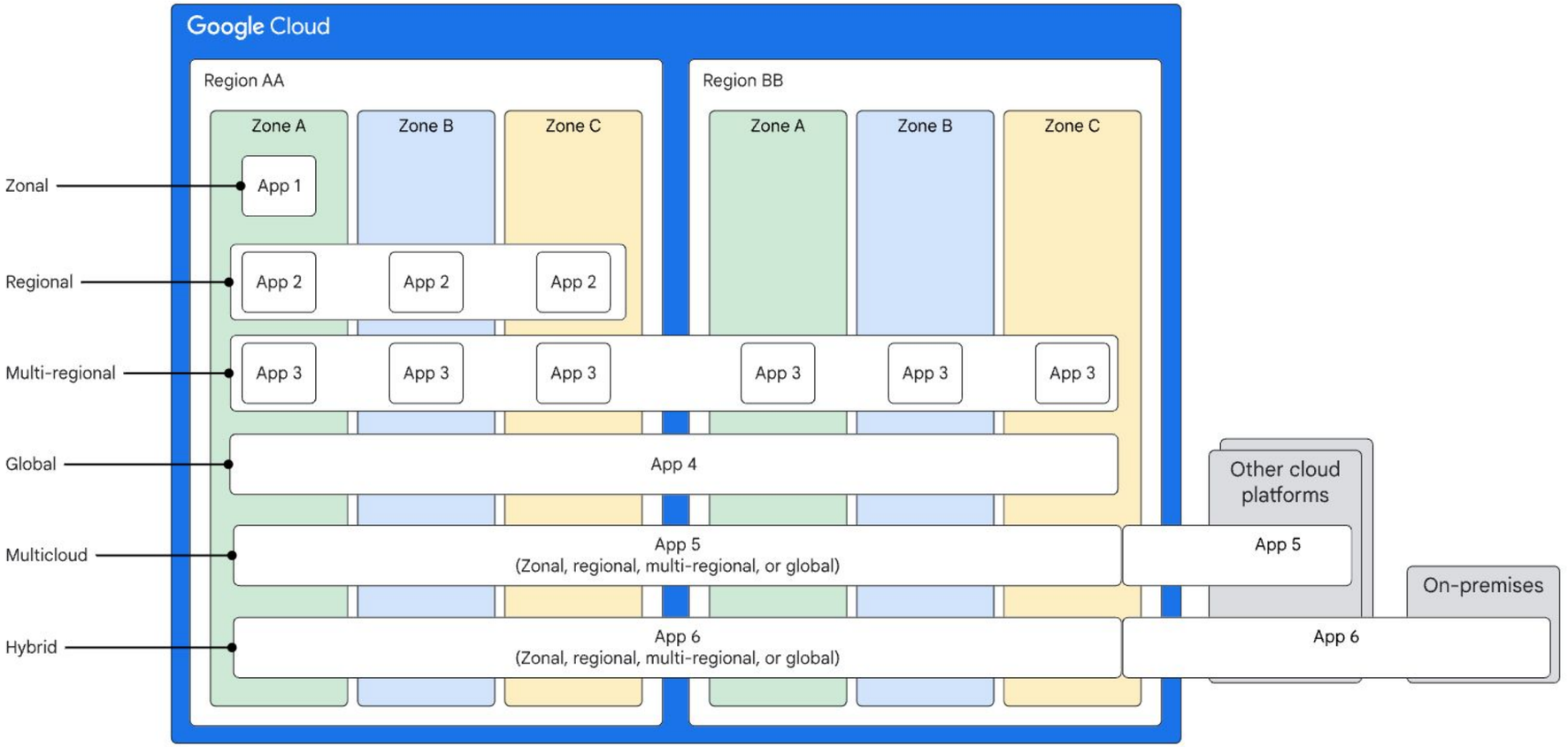
- **CI/CD** not dual-region initially
- **Manual failover** using **break-glass** procedures
- Dual region implementation may come **later**

5 Application Archetypes

5 Application Archetypes

goo.gle/app-archetypes

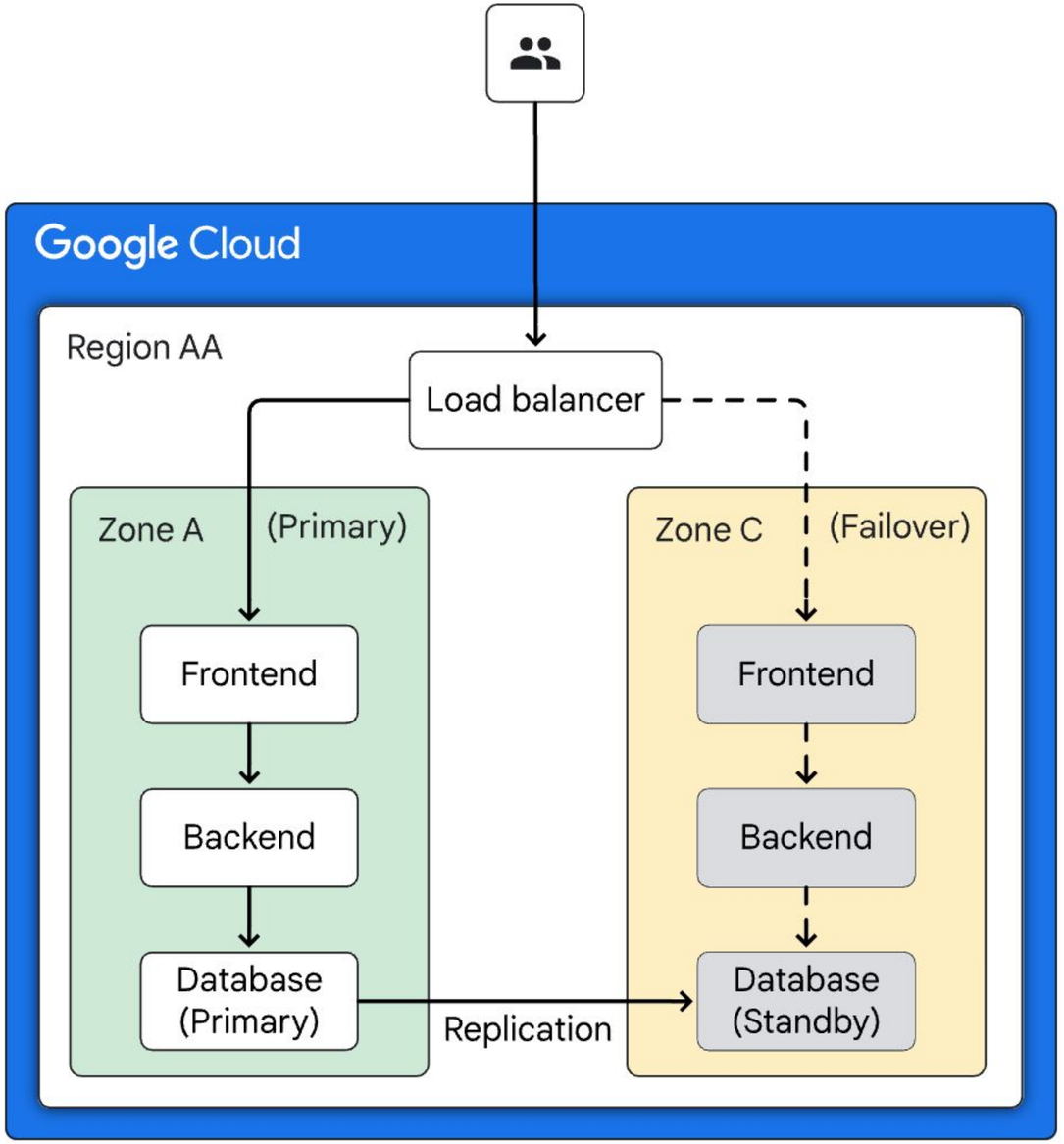
Cloud Architecture Center



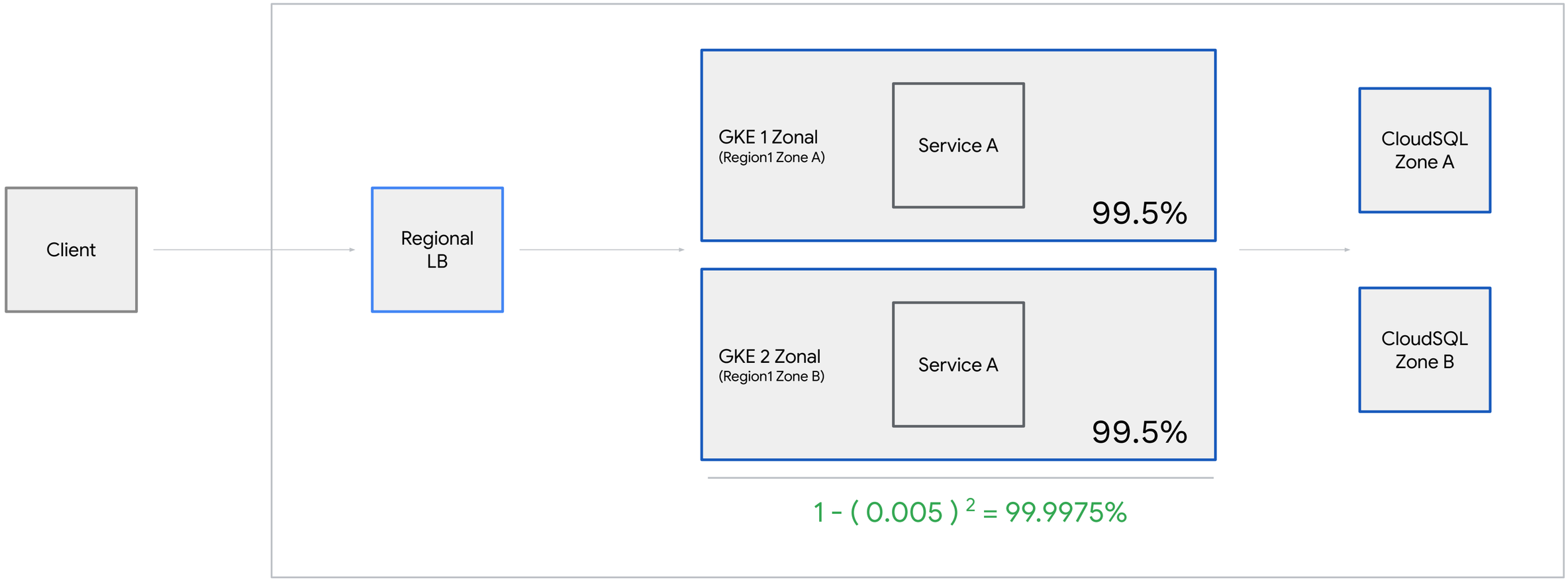
Archetype 1

Active Passive Zones

- **Survives zone** failure.
- **Fail-Ops**: Change LB backend, promote read replica
- **Cost**: 2x serving + 2x data (1 replica)
- **Complexity**: Low
- **App Refactoring**: None (lift and shift)
- **Type**: COTS, licensing



GKE clusters with Cloud SQL HA



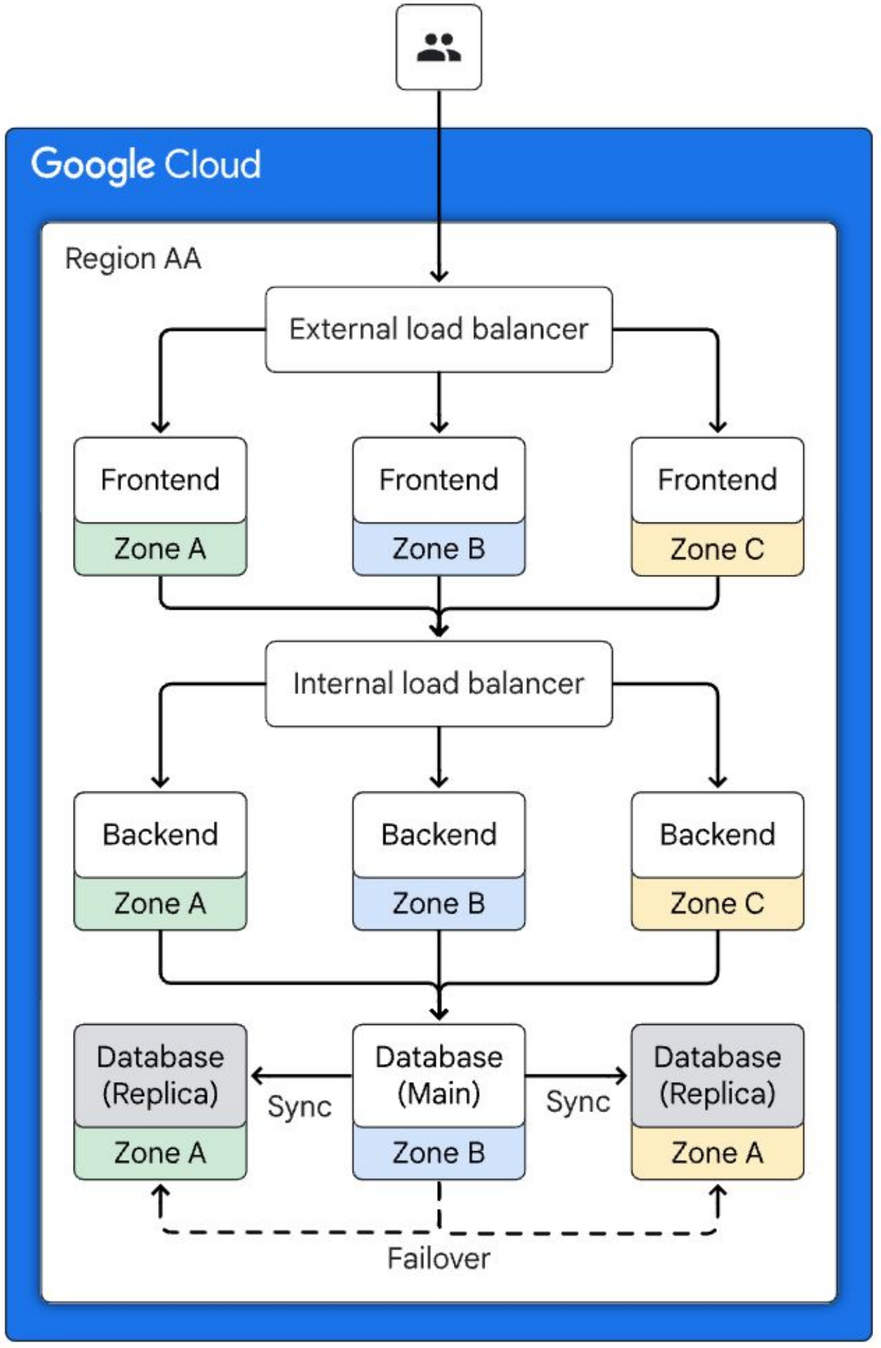
GCP SLAs

99.98% SLO
<2h / year

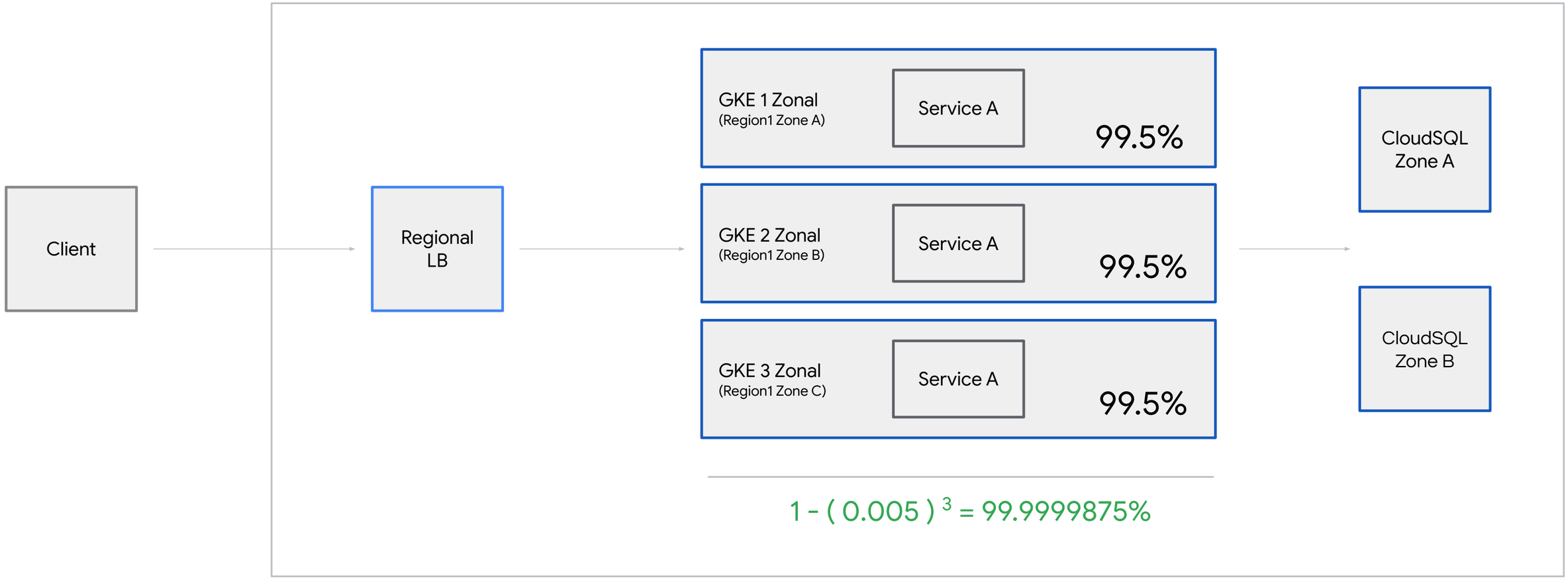
Archetype 2

Multi Zonal

- **Survives zone** failure.
  Does not survive region failure.
- **Fail-Ops**: Initiate DB failover
- **Cost**: 1.5x serving + 3x data (HA SQL)
- **Complexity**: Medium
- **App Refactoring**: Low (multi instance)
- **Type**: Web services



GKE clusters with Cloud SQL HA



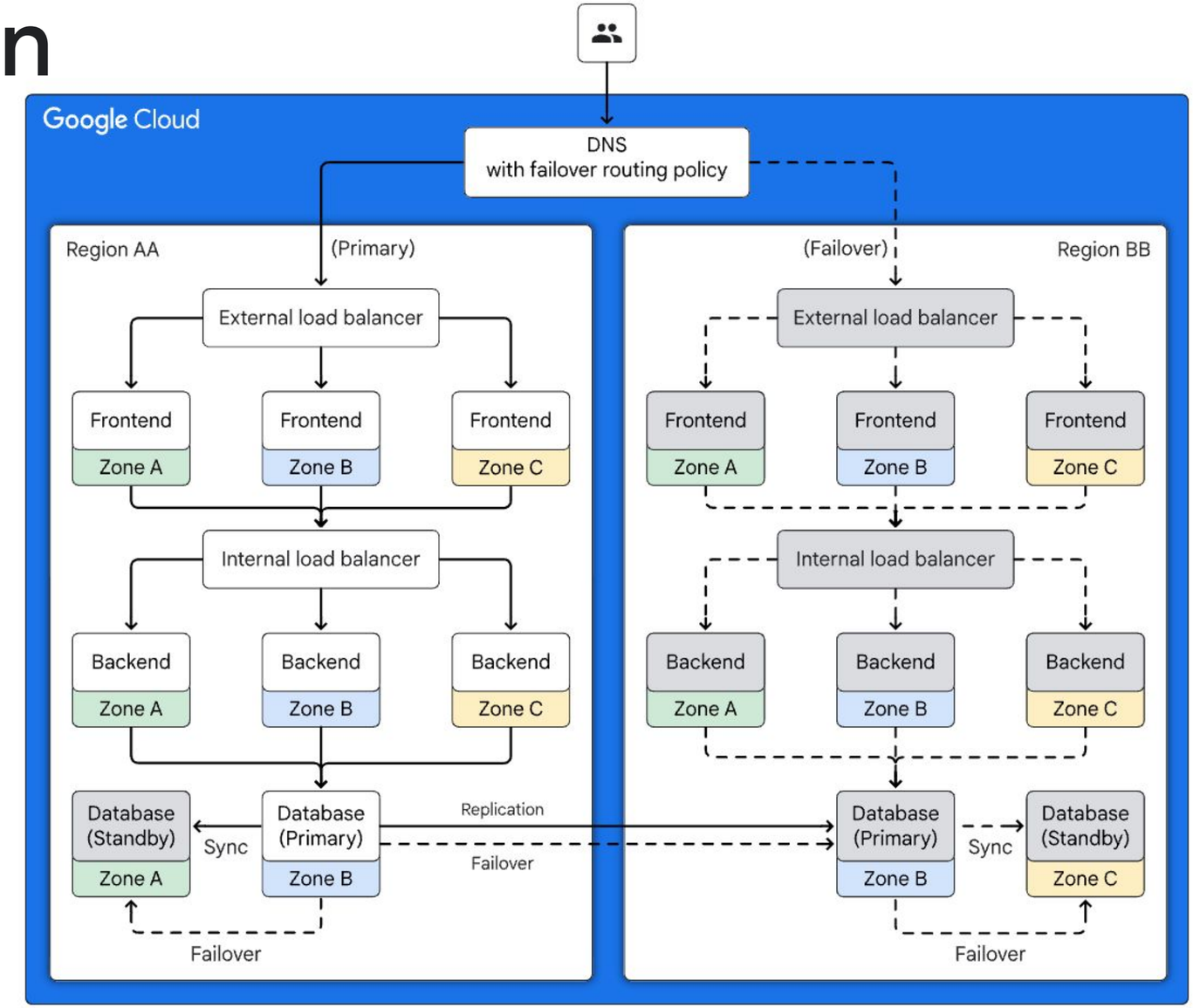
GCP SLAs

99.99% SLO

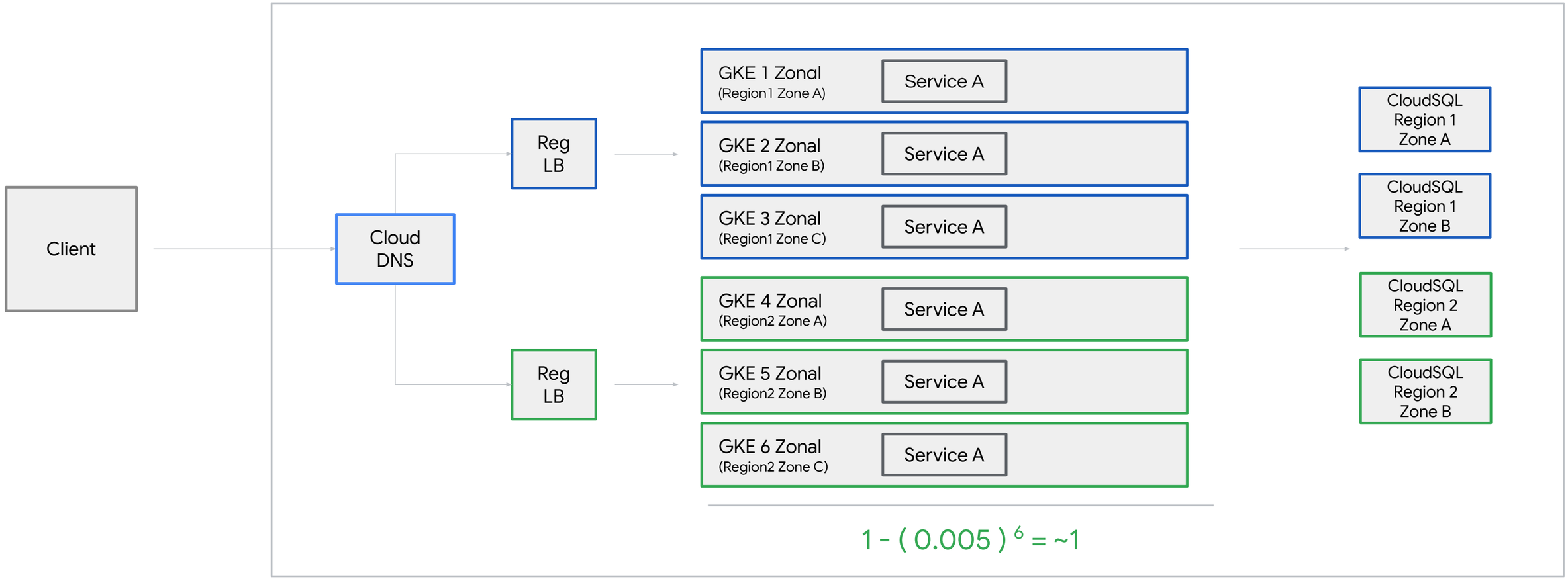
Archetype 3

Active Passive Region

- **Survives zone and region** failures
- **Fail-Ops**: No action for zone failure.
  - Update DNS to point at standby LB
  - DB: Cross region DR failover process
- **Cost**: 3x serving + 6x data (HA SQL)
- **Complexity**: Medium
- **App Refactoring**: Medium (multi instance, multi regional data)
- **Type**: HA web services



GKE clusters in multiple regions with Cloud SQL HA



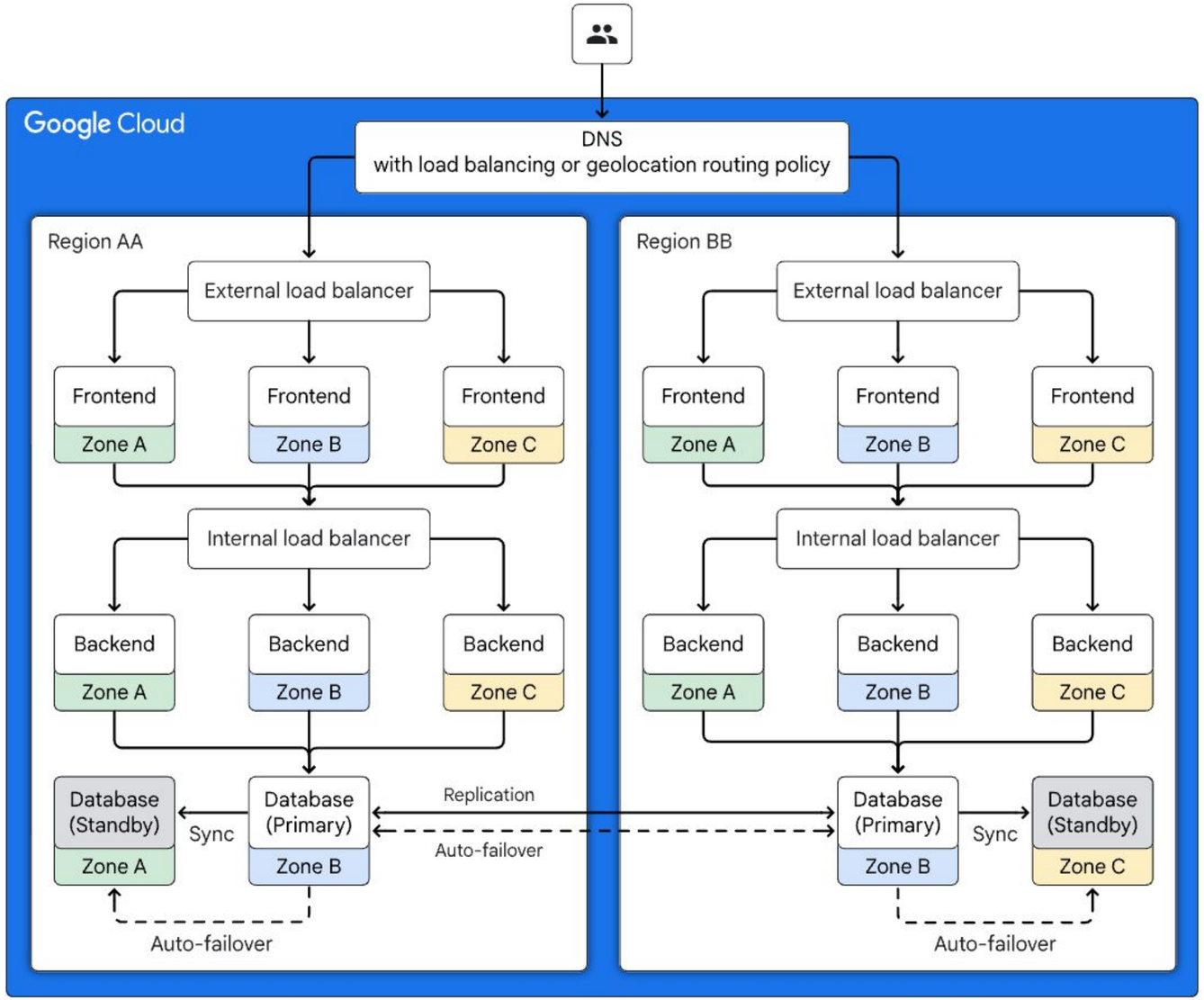
GCP SLAs

99.999999% SLO

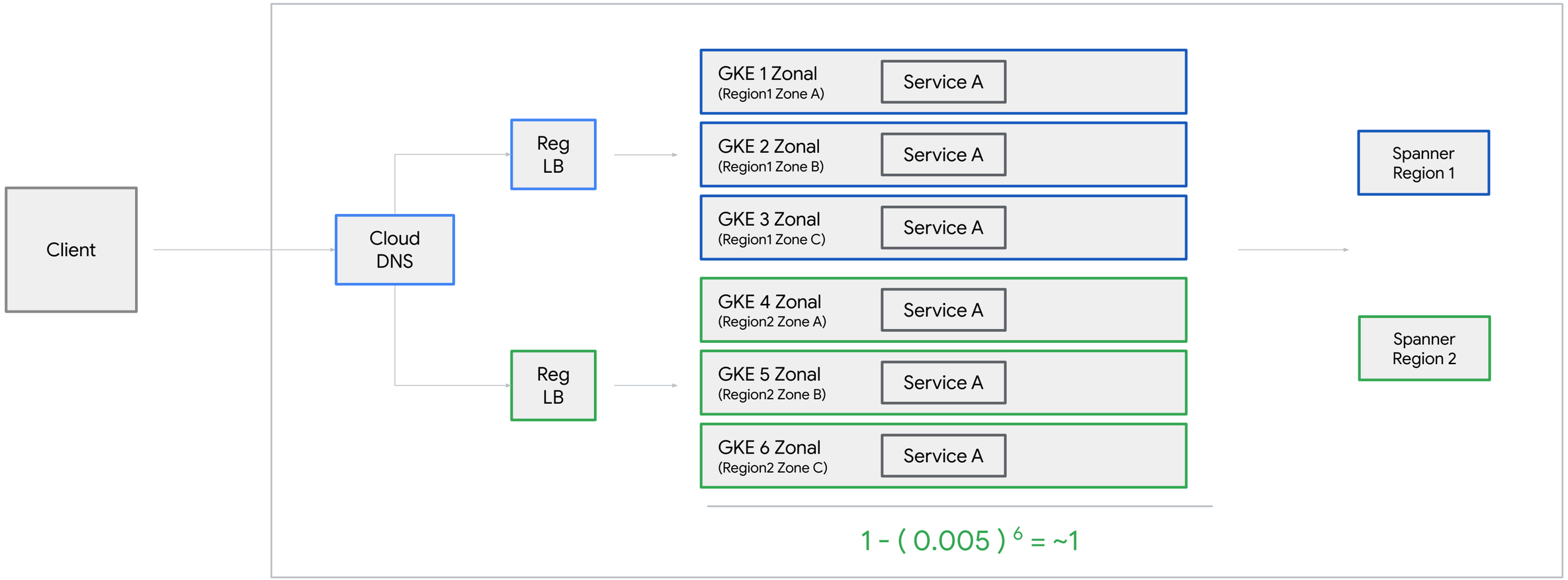
Archetype 4

Isolated Regions

- **Survives zone and region** failures. No impact for ½ consumers. Possible manual failover
- **Fail-Ops**: No action for zone failure. Optional regional failover like Arch 3.2
- **Cost**: 1.5 cost per region for zone failure
- **Complexity**: Medium/High
- **App Refactoring**: Medium (multi instance, multi regional data)
- **Type**: Regulated HA services



GKE clusters with Cloud Spanner



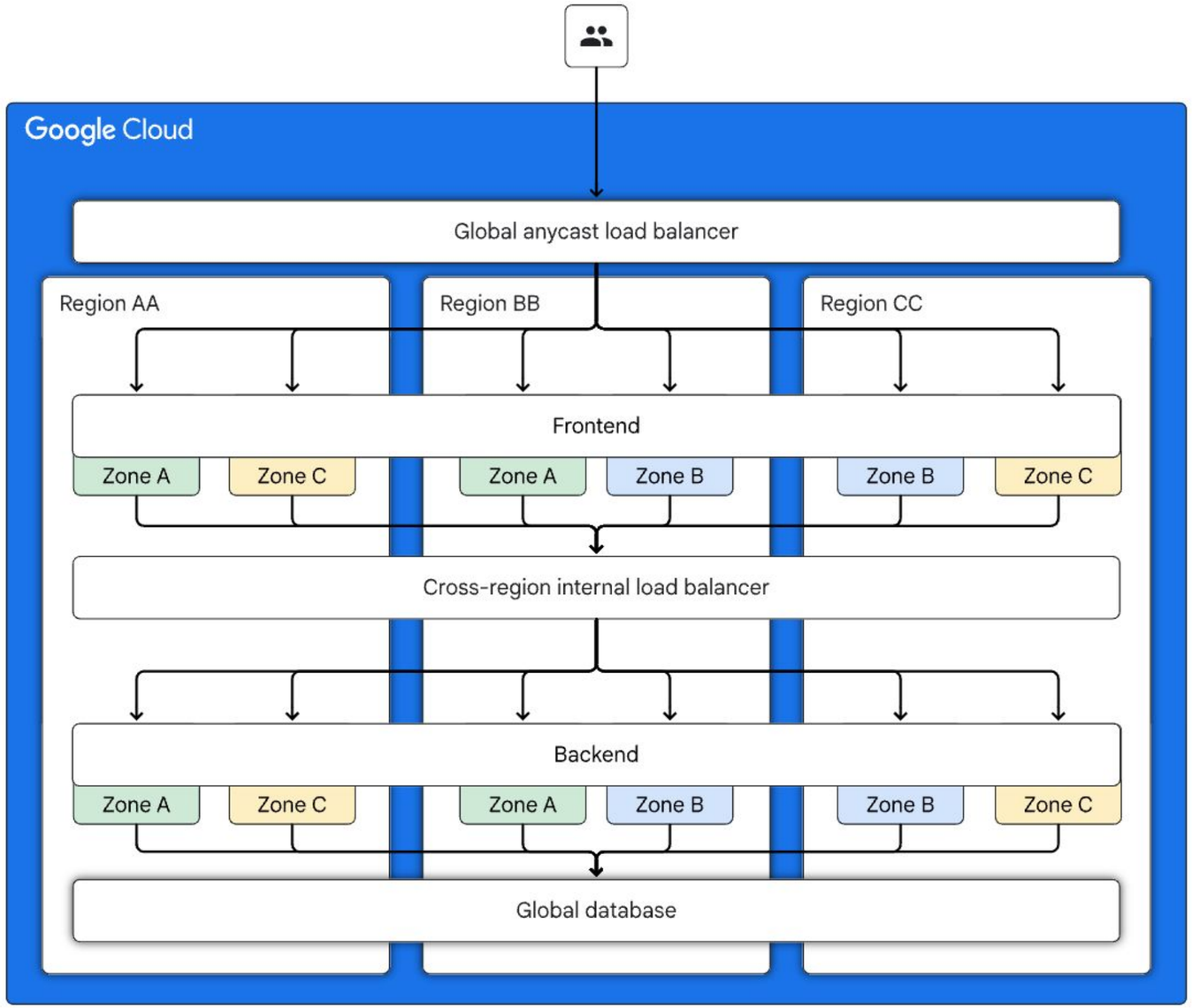
[GCP SLAs]

99.999998% SLO

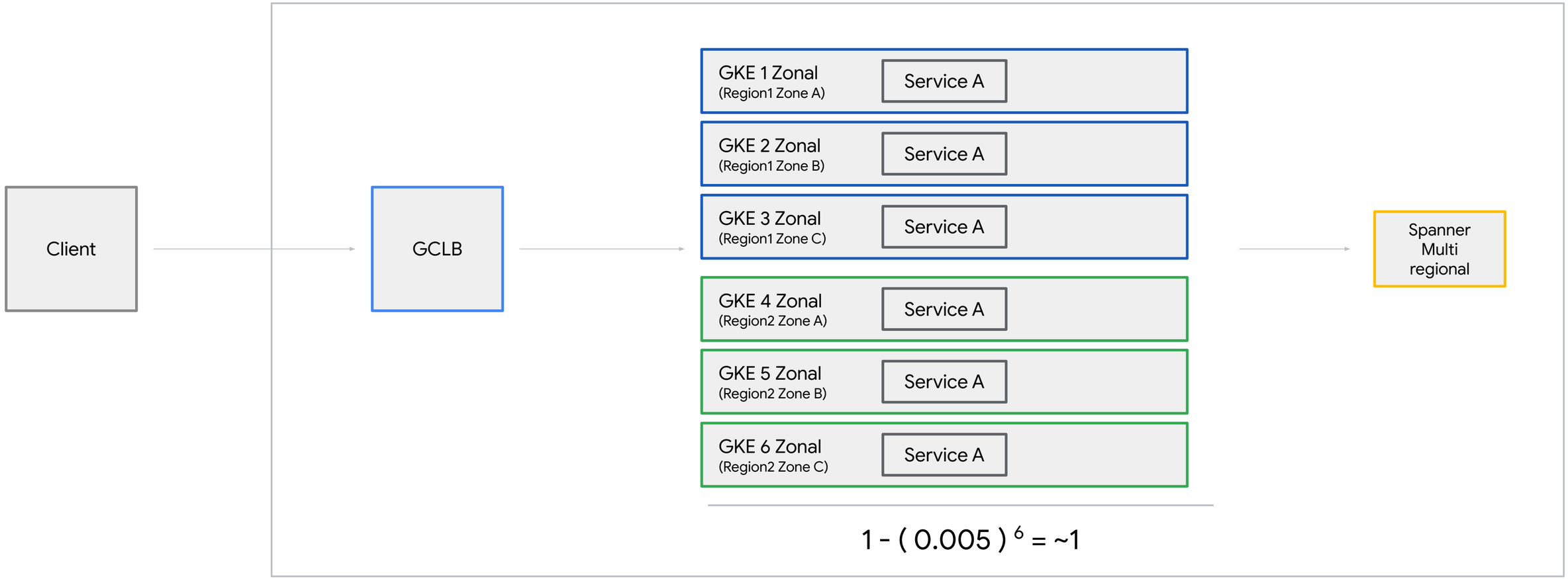
Archetype 5

Global

- **Survives zone and region** failures
- **Fail-Ops**: None
- **Cost**: N+m cost modelling. Global DBs are more expensive
- **Complexity**: High
- **App Refactoring**: High (multi instance, global DBs)
- **Type**: Global consumer services

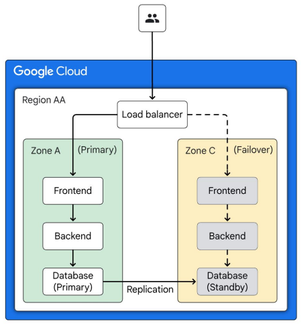


GKE clusters with Cloud Spanner (Multi regional)



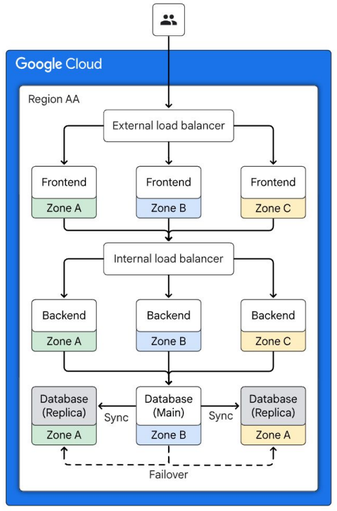
GCP SLAs

99.99% SLO



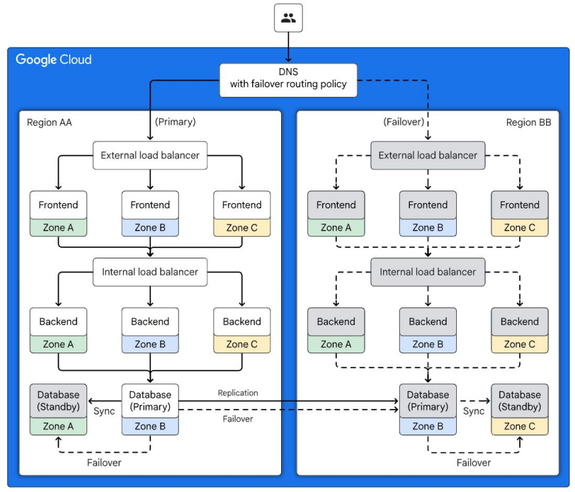
Active Passive Zones

SLO: 99.98



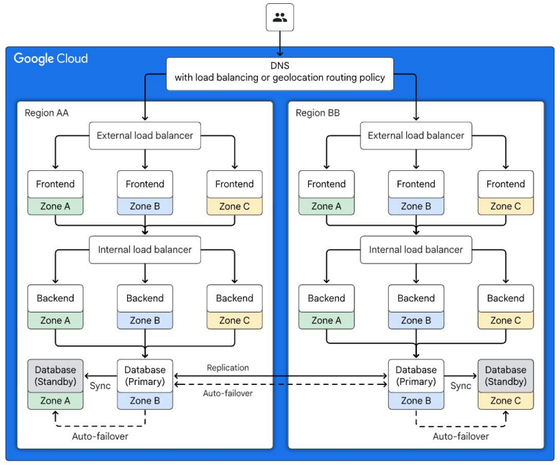
Multi-Zonal

SLO: 99.98



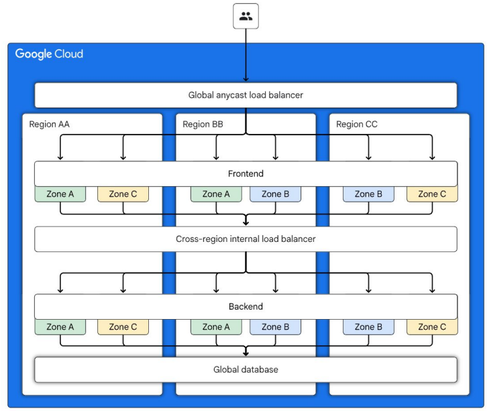
Active Passive Regions

SLO: 99.999999



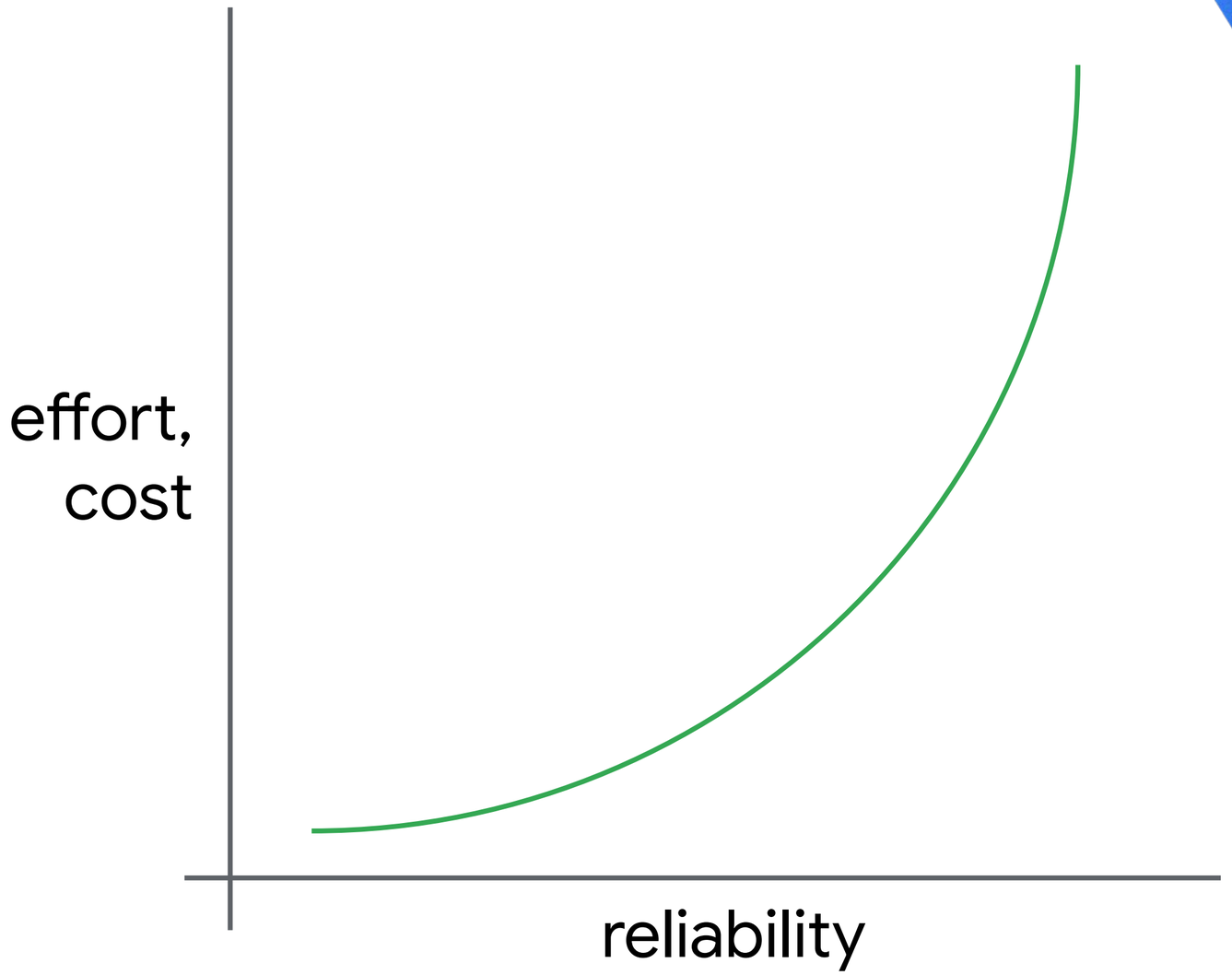
Isolated Regions

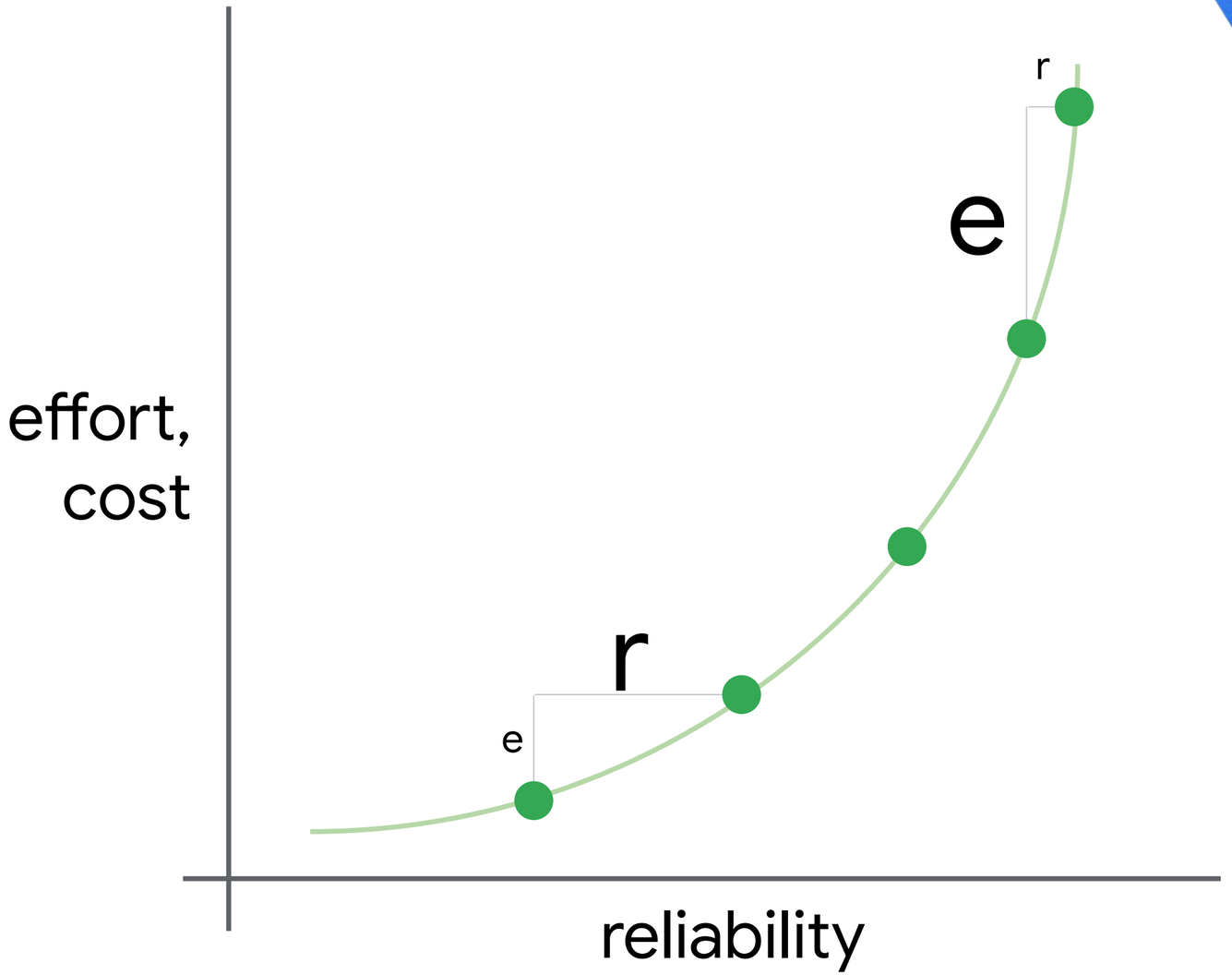
SLO: 99.999998



Global

SLO: 99.99


effort,
cost
reliability


r
e
effort,
cost
r
e
reliability

```yaml
apiVersion: apps.x.anz/v1beta1
kind: CloudRun
metadata:
  isPublic: false
  name: xxxxx-registration
  workspace: xxxxx
  deploymentArchetype: DualRegion
pipeline:
  gha: {}
  monolith: {}
  rolloutStrategy: bluegreen
spec:
  application:
    image:
      repository: australia-southeast1-docker.pkg.de
    livenessProbe:
      failureThreshold: 3
      httpGet:
        path: /healthz
        port: 8080
      initialDelaySeconds: 30
      periodSeconds: 30
      timeoutSeconds: 10
    ports:
    - containerPort: 8080
      name: h2c
  ...
```

Deployment Archetype can be configured on runtime CRDs

Google Cloud

NOBL9™

GitHub Actions

flux

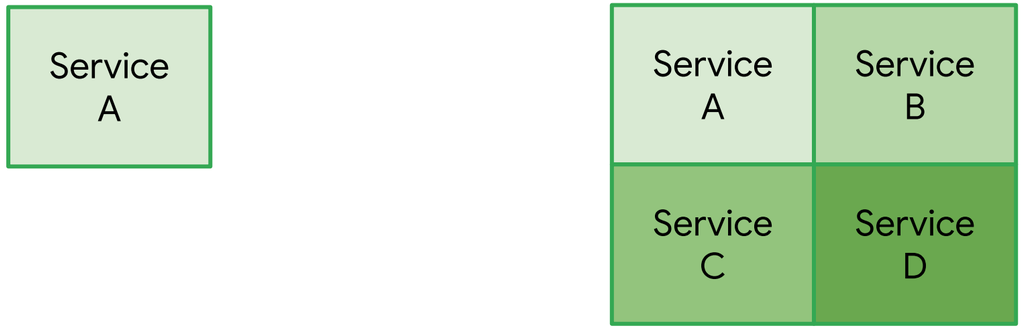
How to use Archetypes?

Services can be deployed to a **single archetype**

How to use Archetypes?

Services can be deployed to a **single archetype**

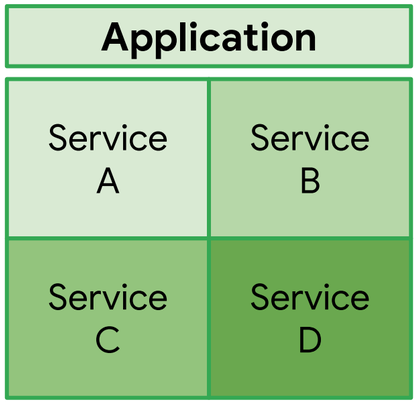
Applications can use services across **multiple archetypes**



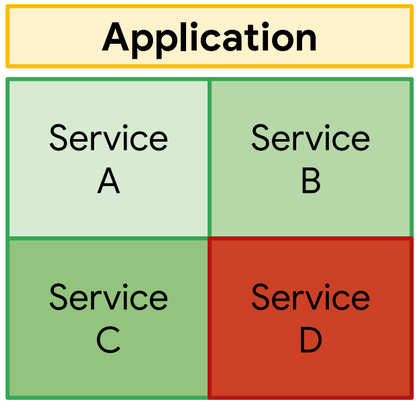
How to use Archetypes?

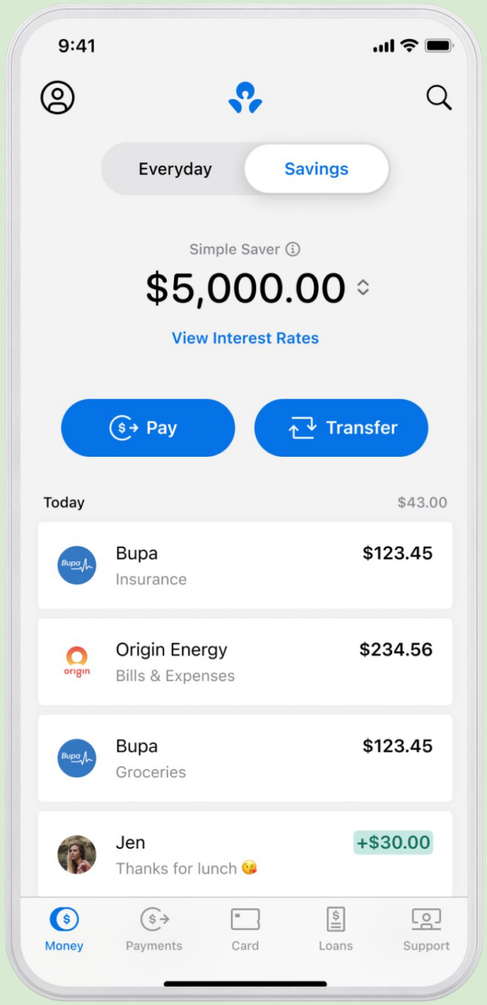
Services can be deployed to a **single archetype**

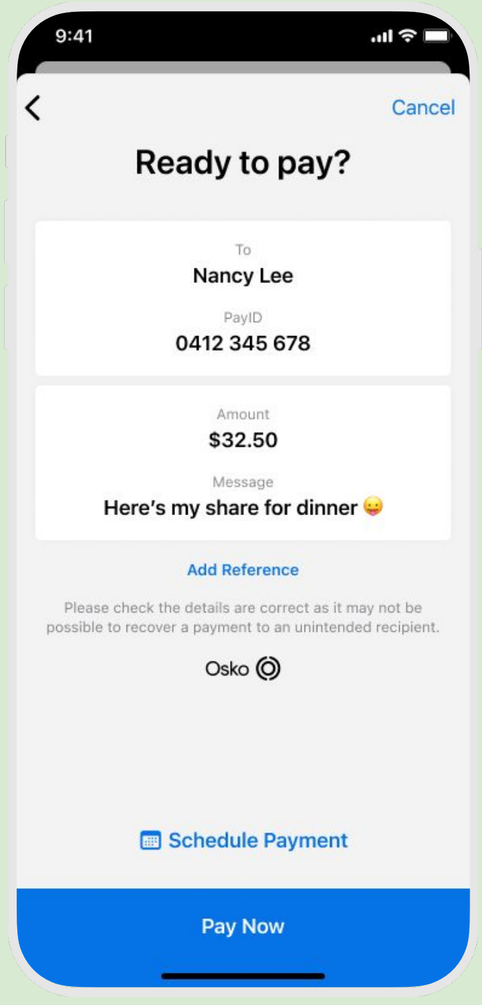
Applications can use services across **multiple archetypes**



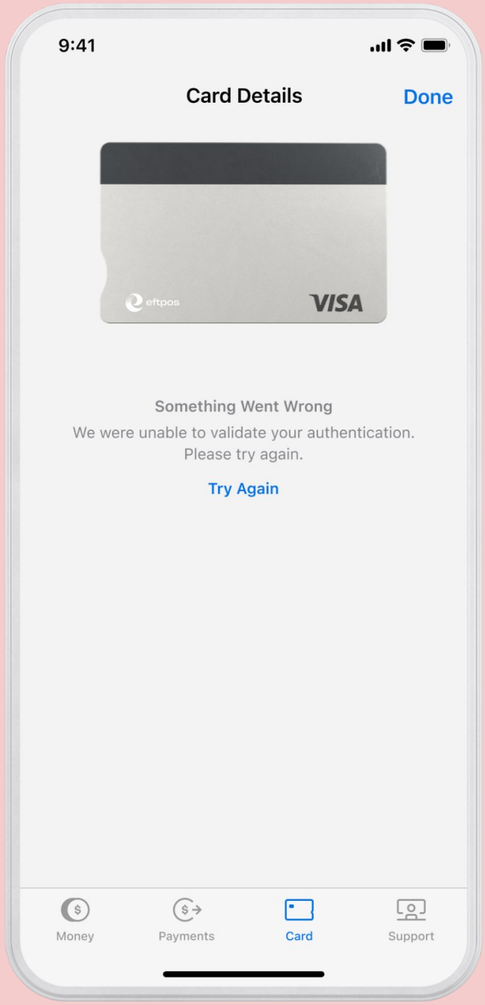
Applications should be designed for **graceful degradation**

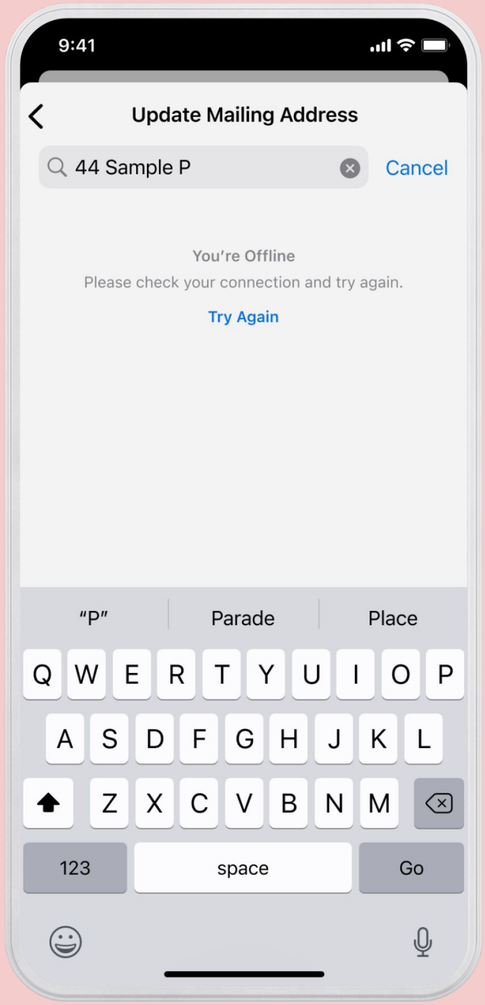






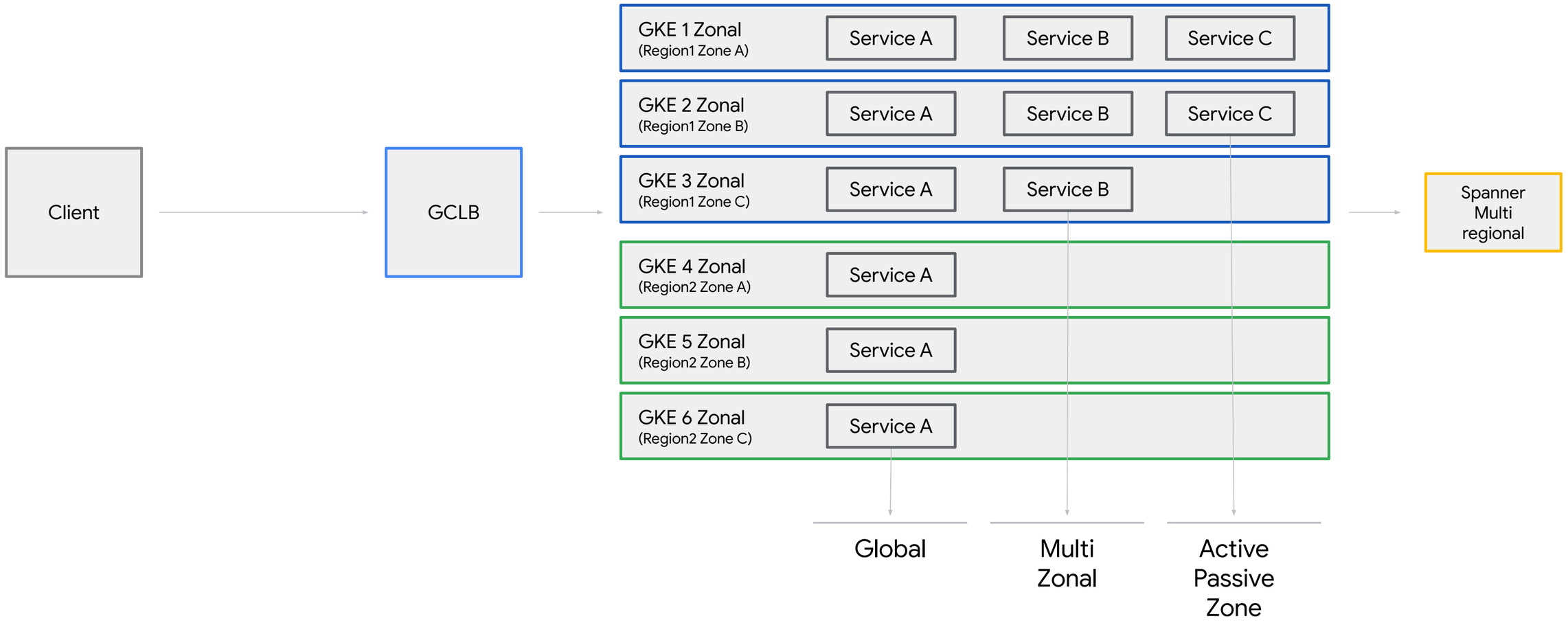
Accounts, Transactions and Payments
remain operational





Card and Profile operations
can fail

Application with Services using multiple Archetypes



Stepping back...

Cloud is complicated!

Stepping back...

Cloud is complicated!

Teams are swamped.

Stepping back...

Cloud is complicated!

Teams are swamped.

Platforms can help.

Platforms can help, but there are challenges..

- Platforms need to cater for the masses, and allow for graceful **opt-outs**.
- We have had mass adoption quickly, but parts of the platform still **need improvement**
- Production-like **test environments** aren’t always possible

Would I do it again?

- **Absolutely!** And our DORA metrics back this up.
- Platforms enable **reliable applications,** improve **developer experience** and **reduce toil** for operators
- **A Platform is software.** Solve problems with data and code

Keep going

reliable-app-platforms Public

Code

Clone, Fork, Contribute:
goo.gle/reliable-app-platforms

Community

Monthly Video Discussions

→ r9y.dev/discuss

Chat on Discord!

→ r9y.dev/chat

Sessions

ARC103 - The reality of reliability: Big differences between hyperscalers, explained

OPS221 - What's new in DevOps tools

OPS105 - The future of platform engineering is application-centric

Ready to build what's next?

Tap into **special offers** designed to help you **implement what you learned** at Google Cloud Next.

Scan the code to receive personalized guidance from one of our experts.

Or visit **g.co/next/24offers**

Continue your learning journey!

Sessions

Session ID - Title

Session ID - Title

Sessions

Session ID - Title

Session ID - Title

Sessions

Session ID - Title

Session ID - Title

Your Feedback is greatly appreciated!

Complete the session survey in the mobile app


Thank you

honeycomb.io

NOBL9

- **Critical services** deployed as Active-active, dual region
- **Observability** stack also dual-region
- **Feature toggling** for outage tiles and disabling UI components

Observability, Feature Toggling remain operational

- **CI/CD not** dual-region initially
- Changes during primary-region-failure: **applied manually** using **break-glass** procedures
- Dual region implementation later

CI/CD can fail

Steve McGhee

Reliability Advocate, Google Cloud

Ameer Abbas

Outbound Product Manager, Google Cloud

Daniel McKeown

Area Lead, Cloud and Orchestration Platform, ANZ Bank

“

So what, Steve?

What do we actually DO.”

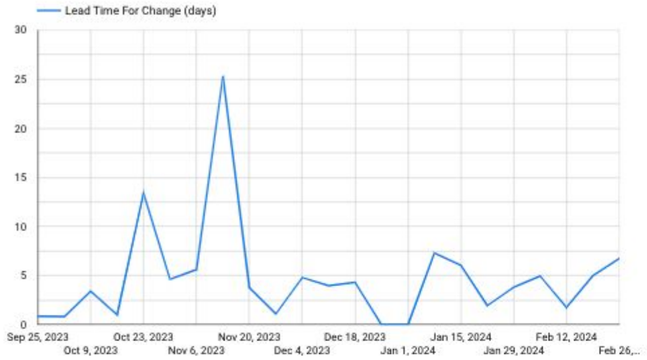
Adopt a platform:

give your team
the means to
run from problems
and
run toward opportunities

The Four Key DORA metrics

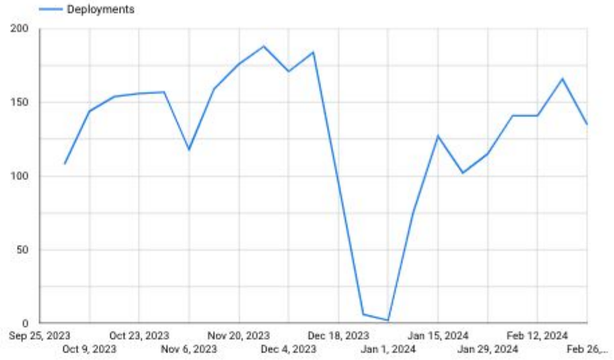
Weekly Average Lead Time for Changes

The average amount of time for a commit to be deployed into production



Weekly Deployments

The number of deployments per week



Lead Time for Changes Only

- GitHub Repository
- Approved by
- Harness Application
- Harness Environment
- Harness Workflow
- IsFirstDeployment

Average Lead Time for Changes
< 1 week (High)

Deployment Frequency
Daily (High)

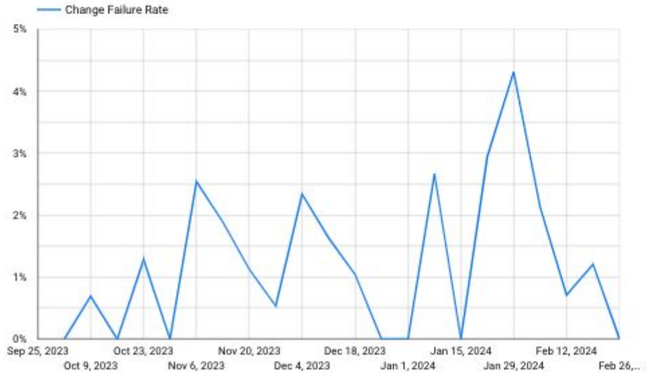
Change Failure Rate Rating
0-15% (High)

Time to Restore Service Rating
< 1 day (High)

- Oct 1, 2023 - Mar 1, 2024

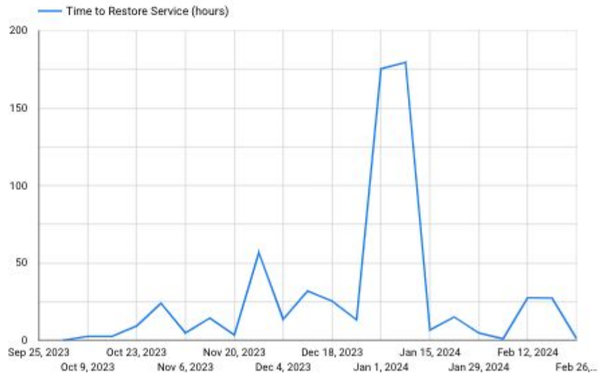
Weekly Change Failure Rate

The number of Failures per the number of deployments. e.g. if there are four deployments in a day and one causes a failure, that would be a 25% change fail.



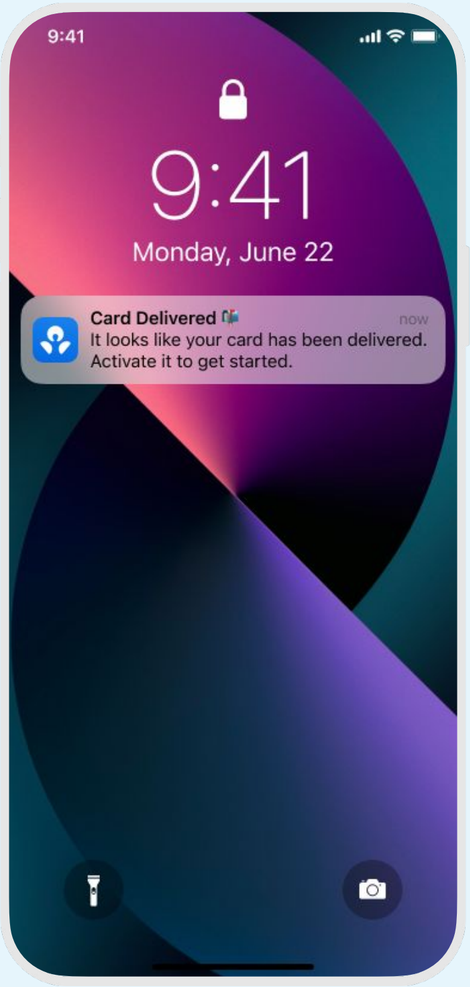
Weekly Average Time to Restore

Represents the time taken to restore a production service from a failed state (triggered by incident being raised and resolved)



Non-Lead Time for Changes Only

- Service Now Assignment Group
- Service Now Business / Tech Area
- Service Now Change Type
- Service Now Configuration Item
- Service Now Change Risk Level
- Priority: 1, 2, 3 (3)



Lead Time for Change

Production Deploy Lead Time

0.13

1.22

% diff from previous month

Merge Lead Time

0

-0.51

% diff from previous month

Approval Lead Time

1.43

-0.52

% diff from previous month

First Review Lead Time

0.09

-0.96

% diff from previous month

Build Lead Time

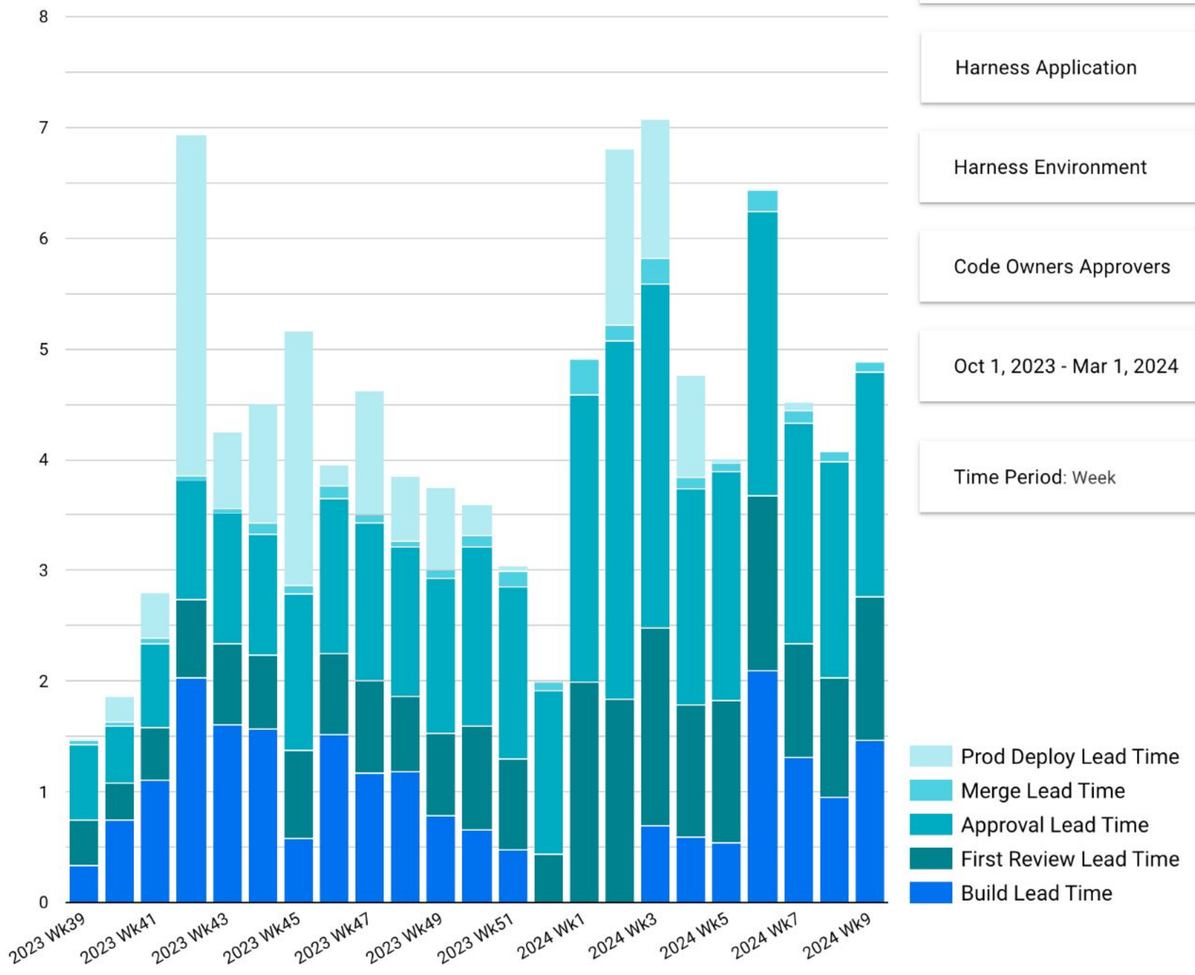
0.01

-1

% diff from previous month

Current Monthly Averages (days)

ANZx Delta Lead Times



GitHub Repository

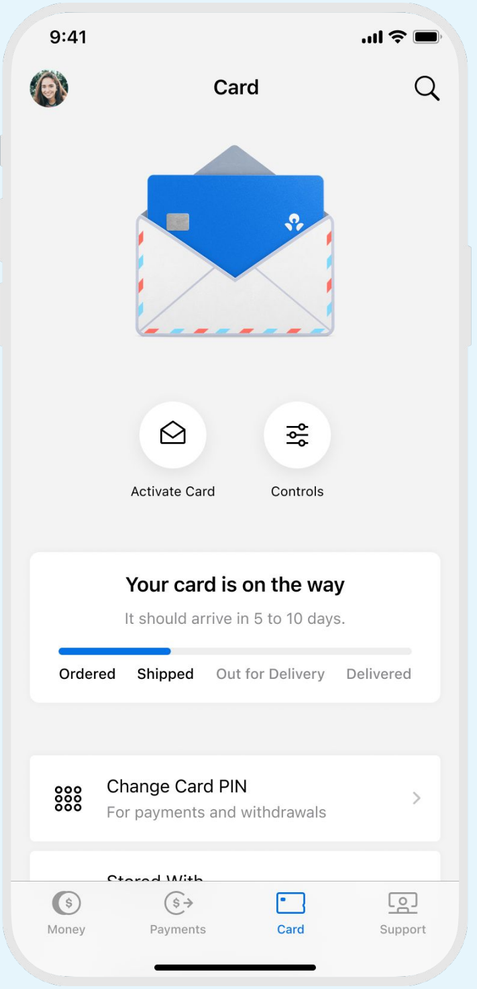
Harness Application

Harness Environment

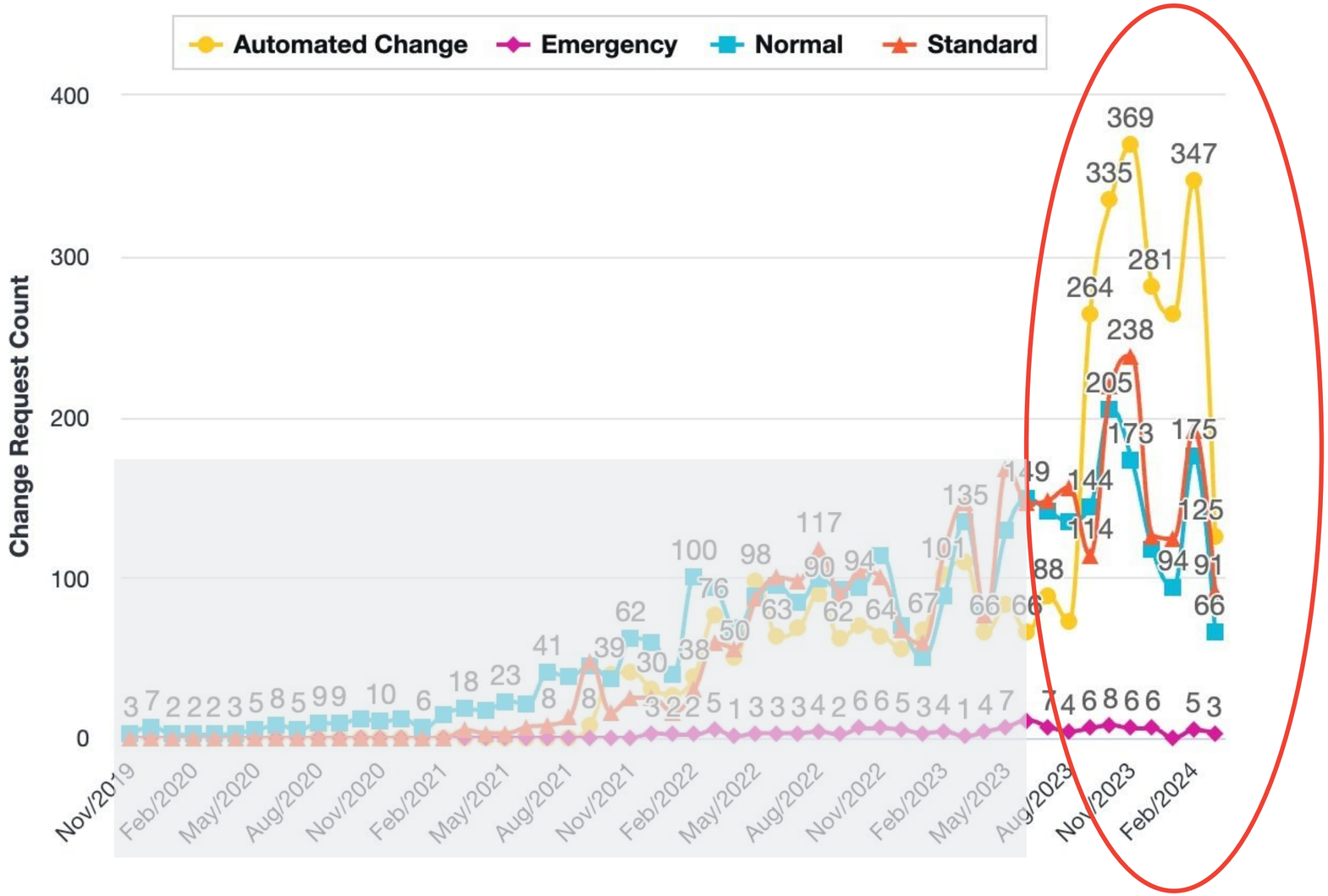
Code Owners Approvers

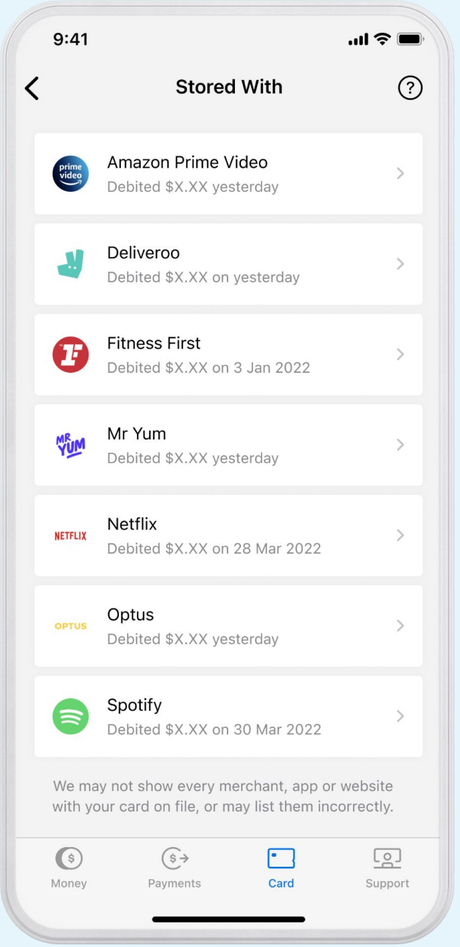
Oct 1, 2023 - Mar 1, 2024

Time Period: Week



Platforms enable Elite teams





Benefits of The X Framework

Asset transparency

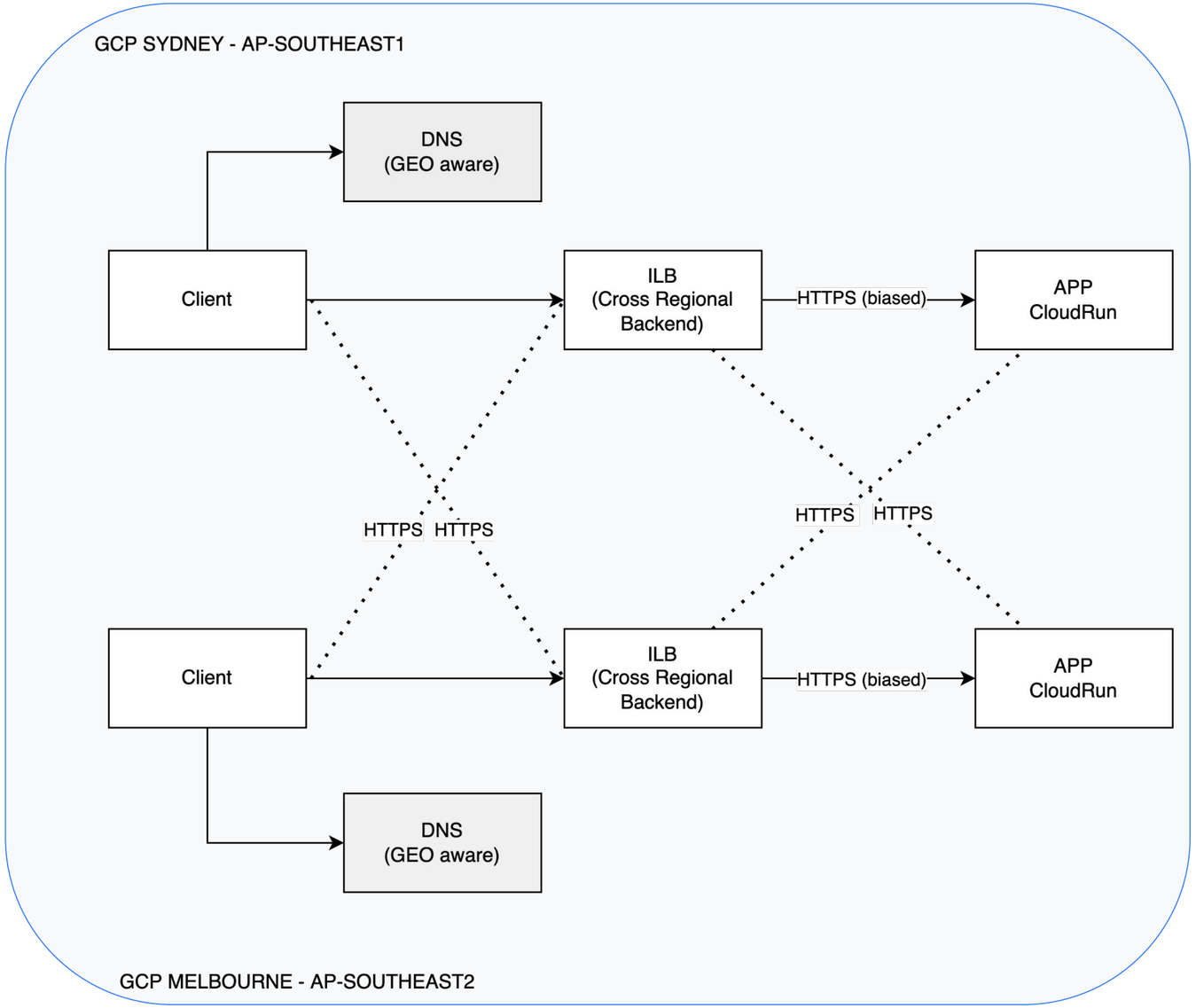
Reduced developer cognitive load

Reduced operator toil

Community Governance

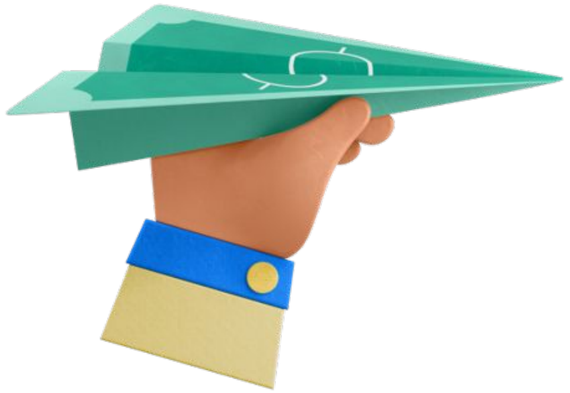
Reliability and failure domains in ANZ Plus

- Achieving multi-region, **active-active** is complex
- Google Services **operate differently** in HA configurations
- Leveraging:
  - dual-region **Cloud Spanner**
  - stateless **Cloud Run** instances
  - region aware **Load Balancing**



X Framework - Adoption

- 95% of KRM is now managed by gitops, drastically improving our **DR posture** and our ability to **operate** the platform
- 85% of repos have **fully generated** CD pipelines
- 80% of workloads are hosted using preferred **runtime patterns**



X Framework - Progress and Plans

Service creation

- Project creation
- Kubernetes Config Connector
- GitHub repository template
- Namespace creation
- Service Archetypes

Cloud features

- Google Secret Manager
- K8s ingress & egress
- Load Balancers and Subnets
- Cloud Run
- DataFlow
- Cloud Functions
- Cloud DNS
- Pub/Sub
- BigQuery
- Cloud KMS
- Spanner

Governance

- Azure AD Groups
- Service accounts
- Google Cloud IAM
- Application Portfolio Management
- Risk & Controls

Dev tooling

- GitHub Actions
- Backstage Component & Techdocs
- Google Artifact Registry
- Code Quality
- Software Composition Analysis
- Static Application Security Testing

Integration & Ops

- Temporal
- Feature Flagging
- Identity integration
- SLO Management
- Cloud Monitoring

blue = done
orange = in progress
grey = future

Criticality

- **Critical platform capabilities** deployed as Active-active, dual region
- **Observability** stack
- **Feature toggling** for outage tiles and disabling UI components

Observability, feature toggling must remain operational

- **CI/CD not** dual-region initially
- **Manual failover** using **break-glass** procedures
- Dual region implementation will come later

CI/CD allowed to fail

Platform

“

Platforms are a means of centralizing expertise while decentralising innovation to the customer or user. This happens through commoditization of services, but in a way which relinquishes an amount of control and is open to extension.

This is a fundamental change to many organizations core business models, which tend to favor standard unification and work centralization.”

Peter Gillard-Moss
ThoughtWorks

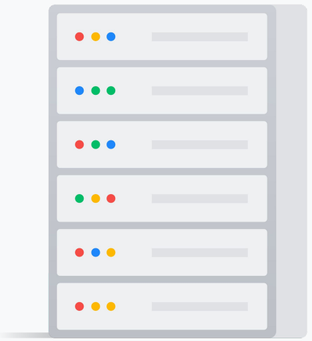
Platform Engineering

More focus
More creativity
More agility

Software Developer

Internal Developer Platform

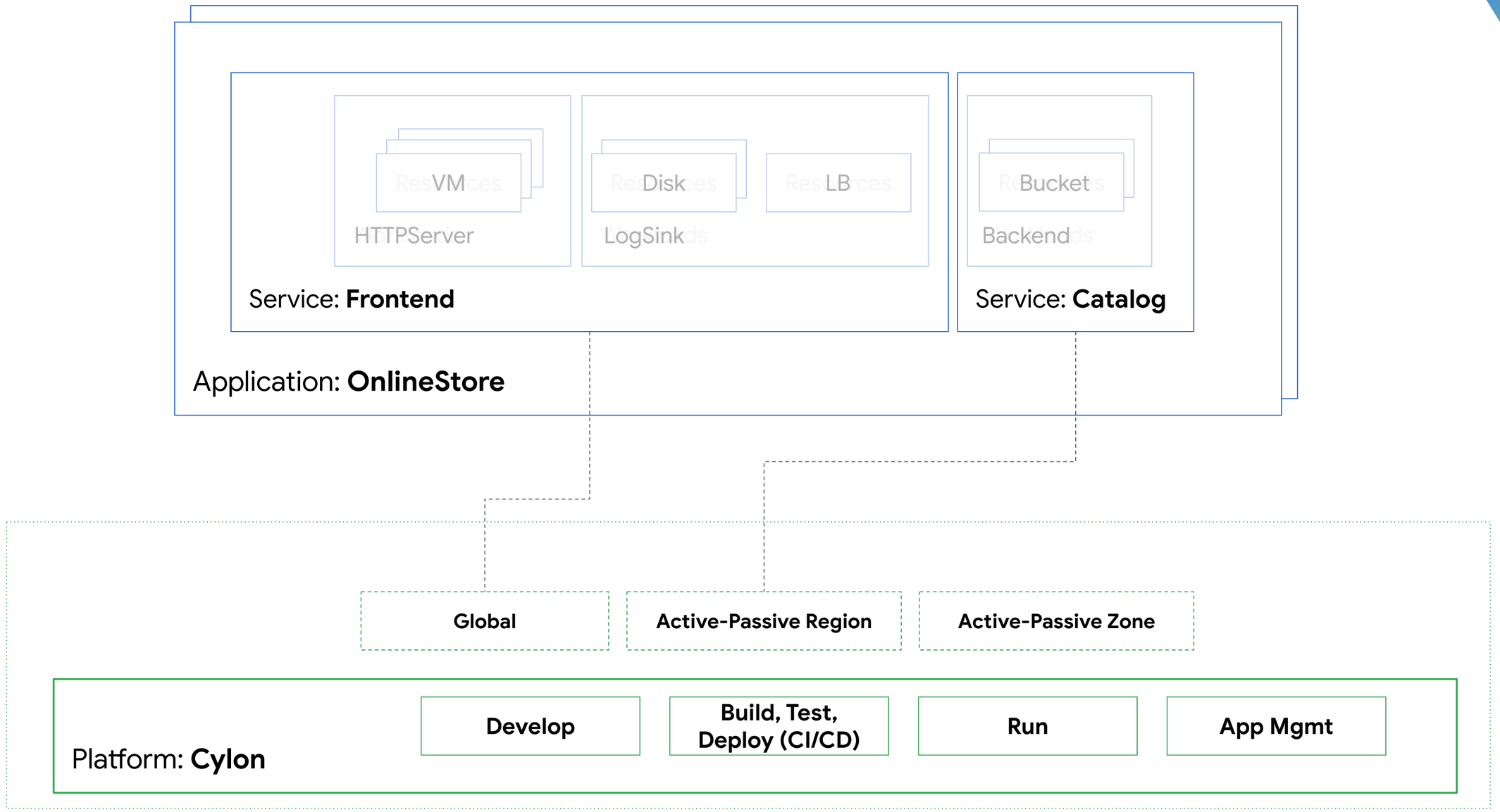
Platform Engineer



Google Cloud Resources

Infrastructure
Dev and Prod

Public Cloud


VM
HTTPServer
Disk
LB
LogSink
Bucket
Backend
Service: Frontend
Service: Catalog
Application: OnlineStore
Global
Active-Passive Region
Active-Passive Zone
Platform: Cylon
Develop
Build, Test, Deploy (CI/CD)
Run
App Mgmt

Platform: **Cylon**

Develop	Build, Test, Deploy (CI/CD)	Run	App Mgmt

Reliability and Failure Domains

Robustness

Scale

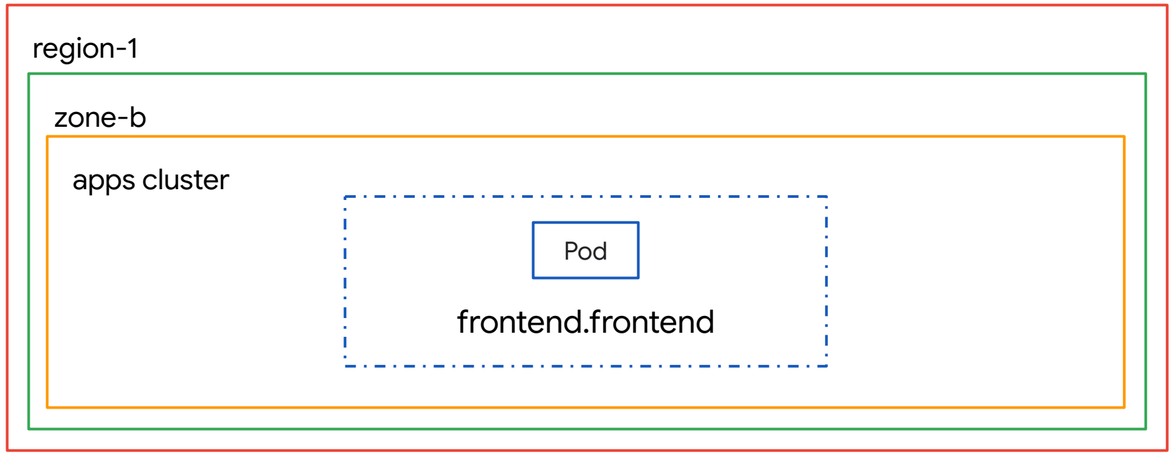
Policy / Security

Manageability

A singleton inside multiple **failure domain envelopes**

❌ **Robustness**

Scale ❌



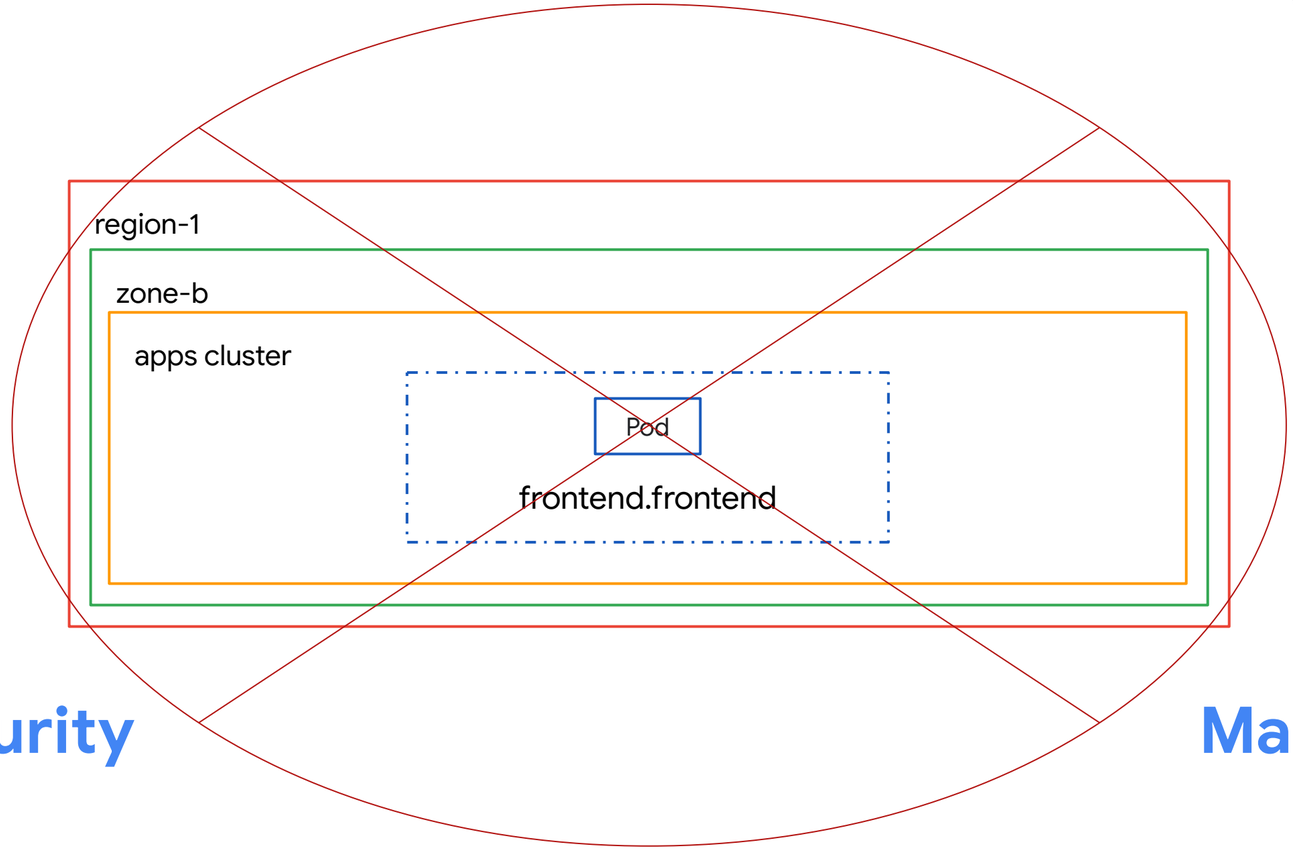
✅ **Policy / Security**

Manageability ✅

Singletons are **insufficient** for Robustness and Scale

❌ **Robustness**

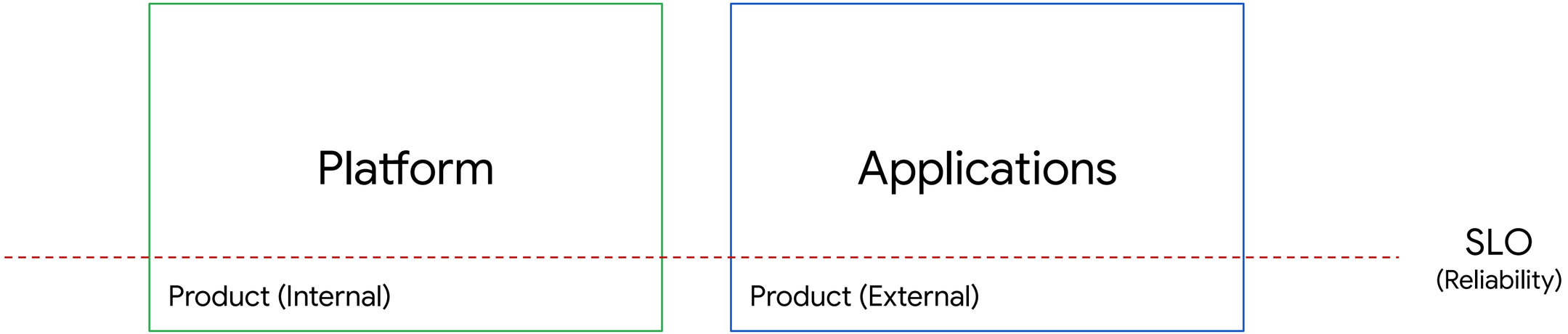
Scale ❌

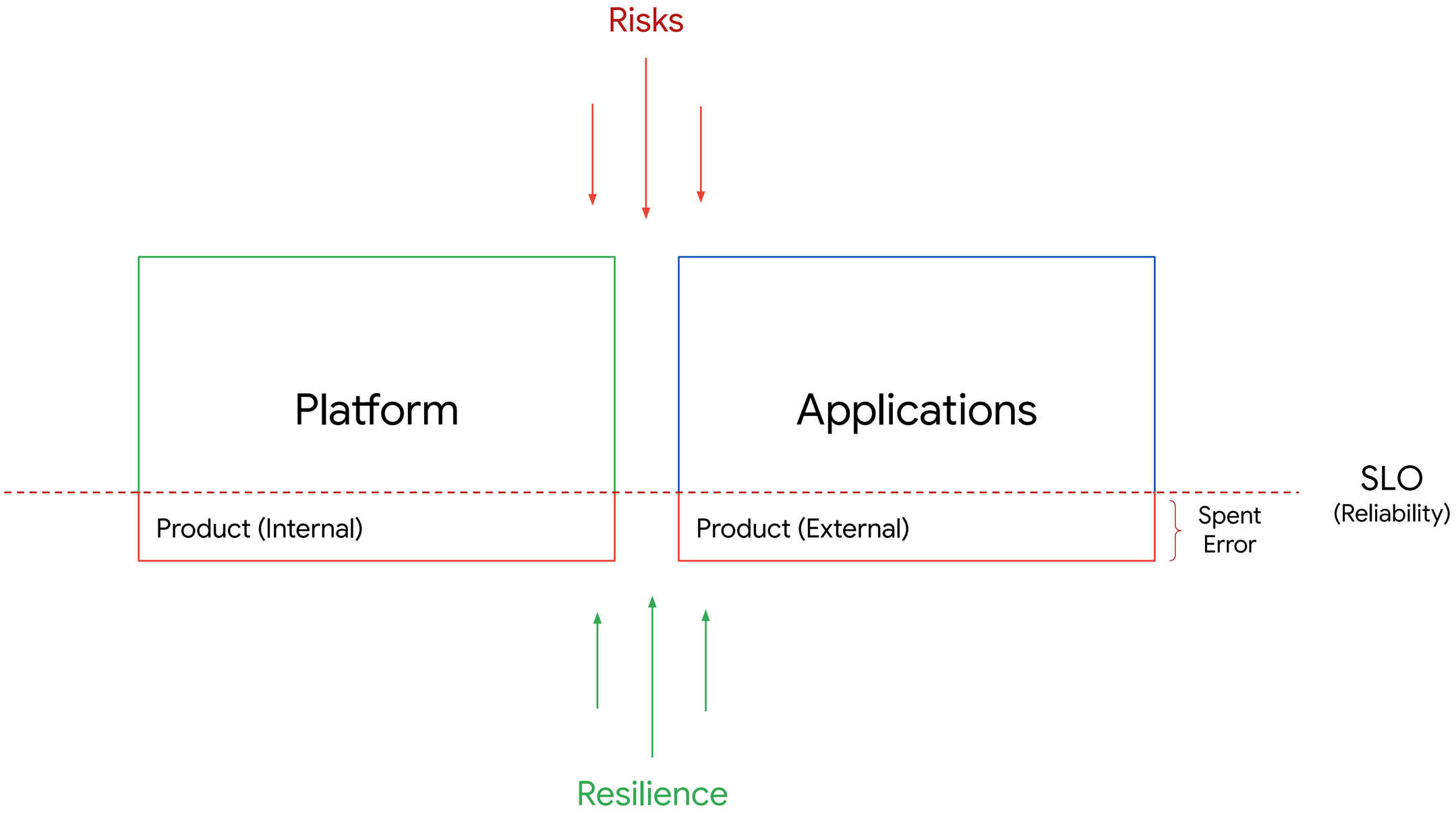


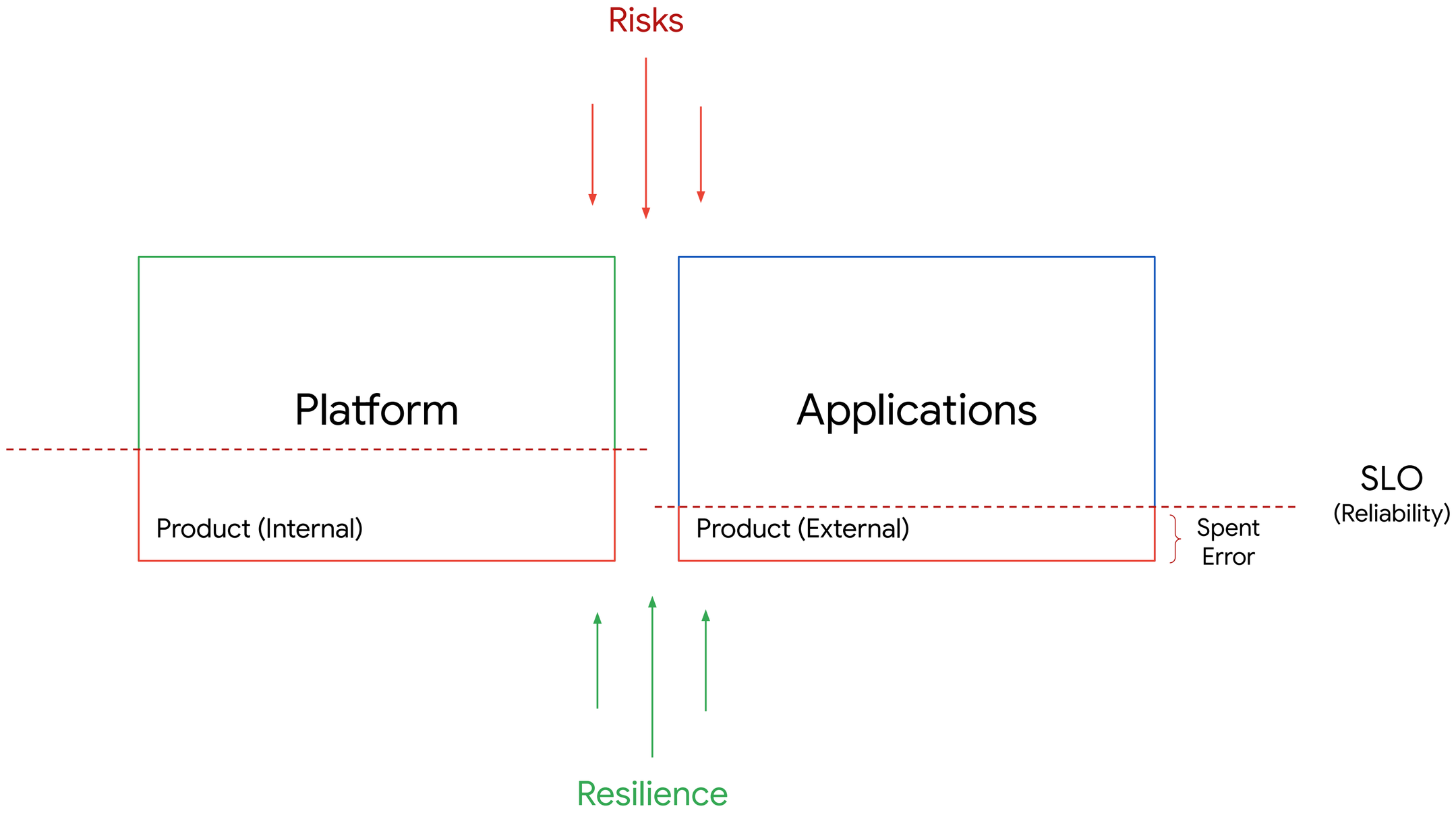
✅ **Policy / Security**

Manageability ✅

(but may simplify Policy / Security + Manageability)


Platform
Applications
Product (Internal)
Product (External)
SLO
(Reliability)


Risks
Platform
Applications
Product (Internal)
Product (External)
Spent
Error
SLO
(Reliability)
Resilience


Risks
Platform
Applications
Product (Internal)
Product (External)
SLO
(Reliability)
Spent
Error
Resilience

Your Deck is Under Review

Please refrain from making changes

Application with Services using multiple Archetypes