20TH ANNIVERSARY EDITION

WSO2CON ASIA

PLATFORMLESS MODERNIZATION

Operating at Scale: **Platform Engineering & Strategies with Choreo**



Lakmal Warusawithana

Vice President & Distinguished Engineer

WSO2



Binura Gunasekara

Associate Director & Architect

WSO2





First, a quick backstory...

Scaling up! 🚀

- Since the last last IDP lab, your organization has seen incredible growth!
  - You went from a few devs to multiple teams working across projects and locations.
- Your apps transitioned from prototypes to **critical, production-grade services.**
- **NFRs** have now become non-negotiables:
  - Reliability
  - Security
  - Observability
  - Scalability
  - (and regulatory/ compliance requirements as well!)

...You've hit a pivotal point:

A more structured platform strategy is now essential.

✨ Welcome to the life of a

Platform Engineer! ✨

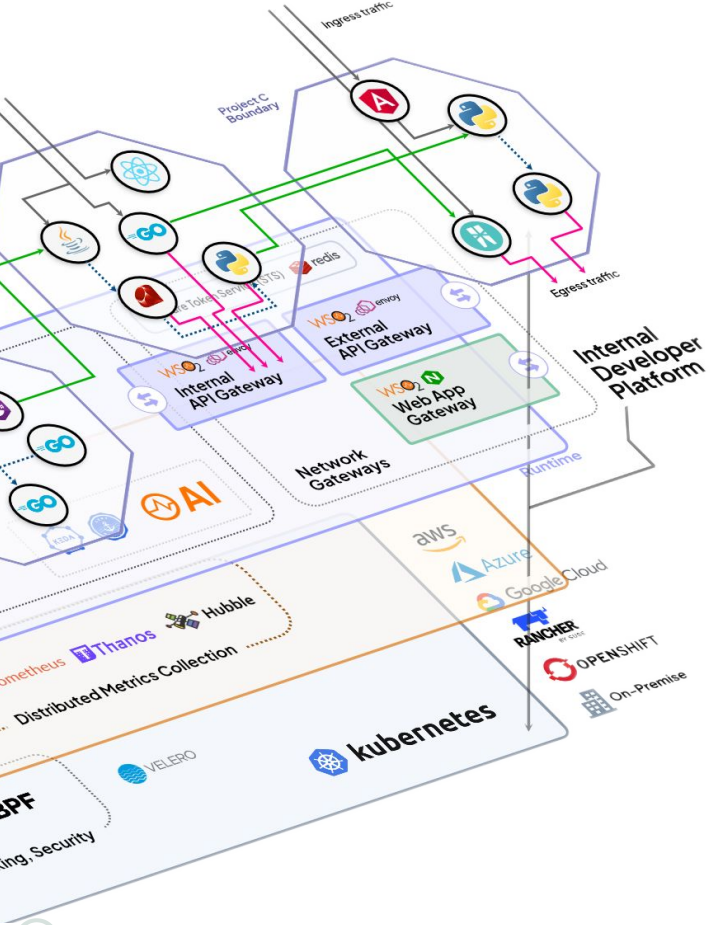


Agenda

Agenda

1. 🏰 **Introduction to Platform Engineering (PE).**
2. 🌱 **Infrastructure:** Building a foundation for scale.
3. 🌐 **Environments & CI/CD pipelines**: Reliable releases at scale.
4. 🔒 **Zero-trust & governance**: Securing your growing platform.
5. 📈 **Observability & site reliability engineering (SRE).**
6. 📊 **Platform insights:** Measuring to improve.
7. 🎯 **Q&A.**

Introduction to
Platform Engineering (PE)



By definition:

Platform engineering is the discipline of designing and building self-service **(internal developer) platforms** that optimize the software development lifecycle. It enables developers to focus on writing and delivering high-quality code without getting bogged down by operational complexities.

Learn more:
https://wso2.com/choreo/what-is-platform-engineering/

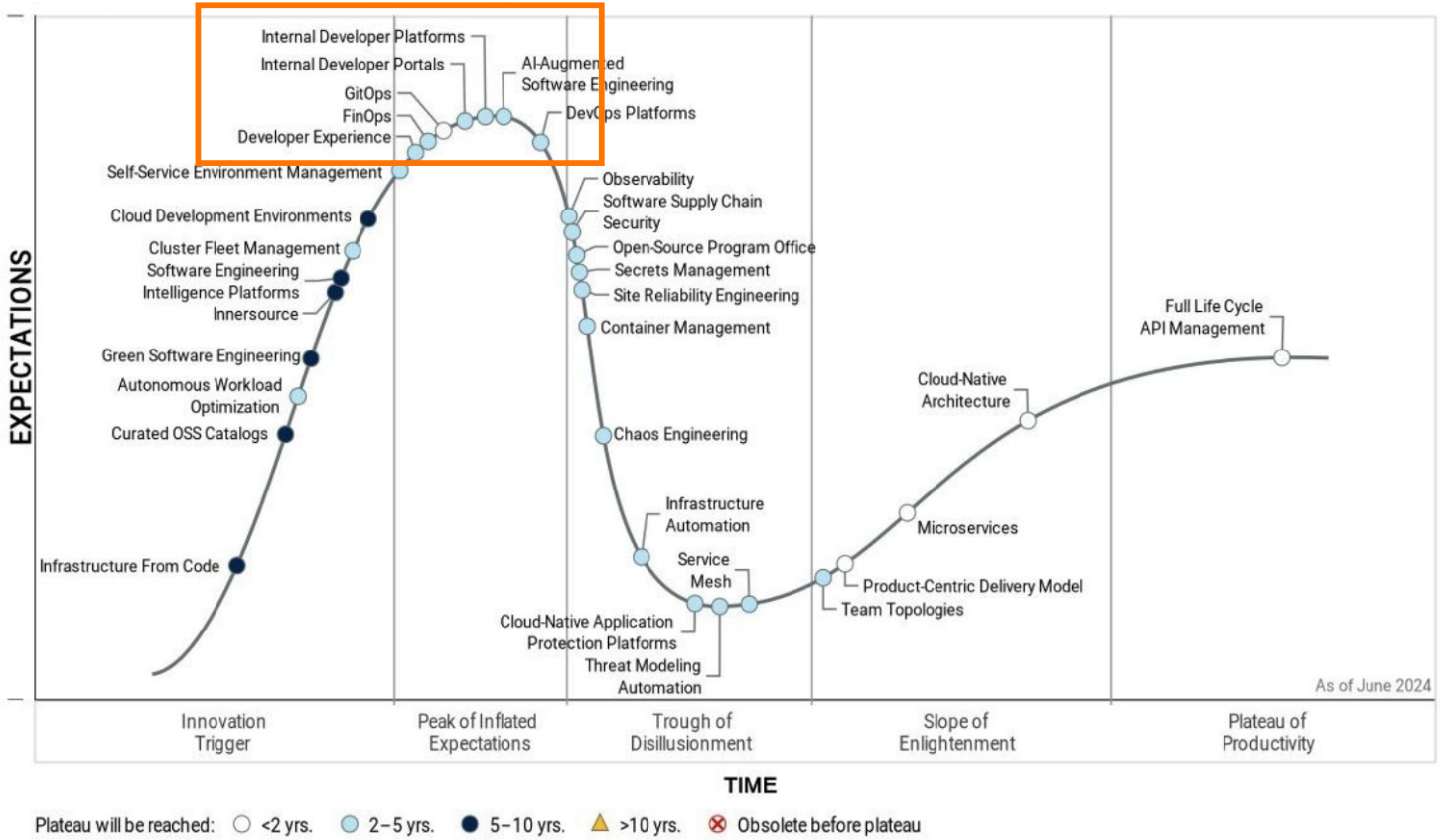
CNCF Platform Engineering Maturity Levels

- Level 1, Provisional
- Level 2, Operationalized
- Level 3, Scalable
- Level 4, Optimizing

ASPECT		PROVISIONAL	OPERATIONAL	SCALABLE	OPTIMIZING
Investment	*How are staff and funds allocated to platform capabilities?*	Voluntary or temporary	Dedicated team	As product	Enabled ecosystem
Adoption	*Why and how do users discover and use internal platforms and platform capabilities?*	Erratic	Extrinsic push	Intrinsic pull	Participatory
Interfaces	*How do users interact with and consume platform capabilities?*	Custom processes	Standard tooling	Self-service solutions	Integrated services
Operations	*How are platforms and their capabilities planned, prioritized, developed and maintained?*	By request	Centrally tracked	Centrally enabled	Managed services
Measurement	*What is the process for gathering and incorporating feedback and learning?*	Ad hoc	Consistent collection	Insights	Quantitative and qualitative

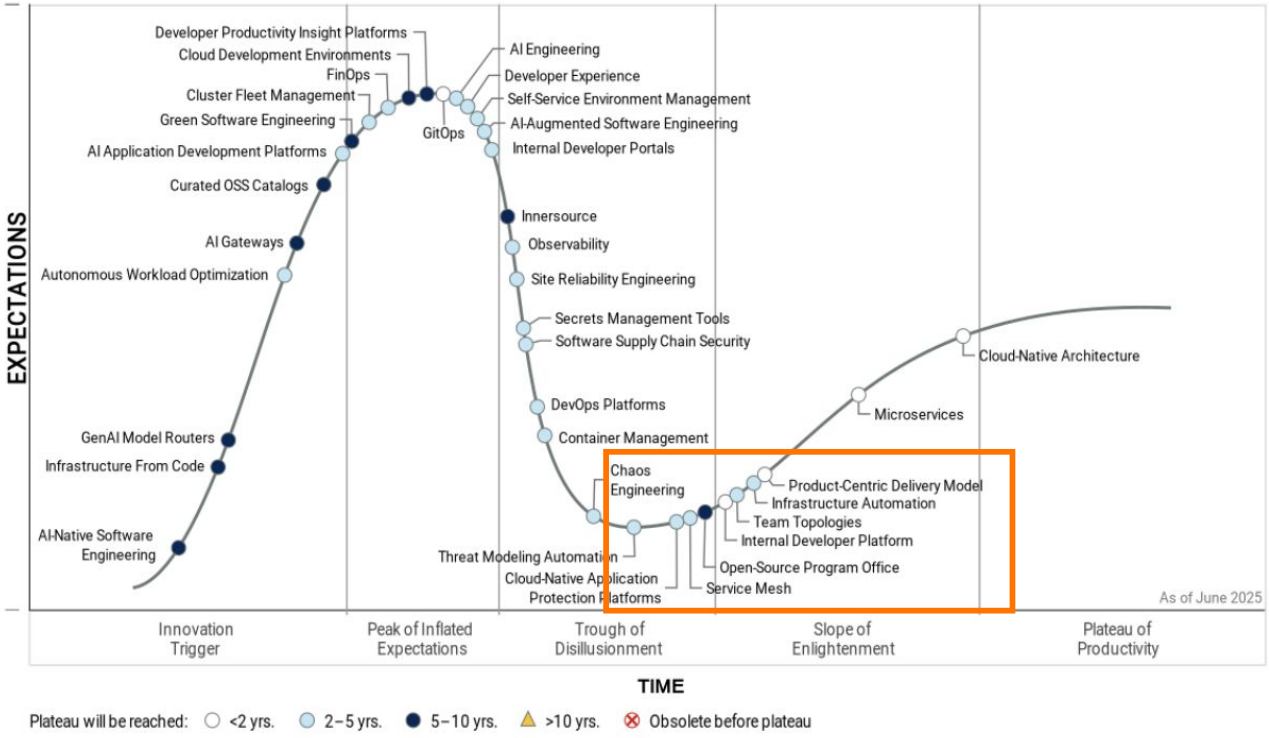
Reference : https://tag-app-delivery.cncf.io/whitepapers/platform-eng-maturity-model/

Gartner® Hype Cycle™ for Platform Engineering 2024



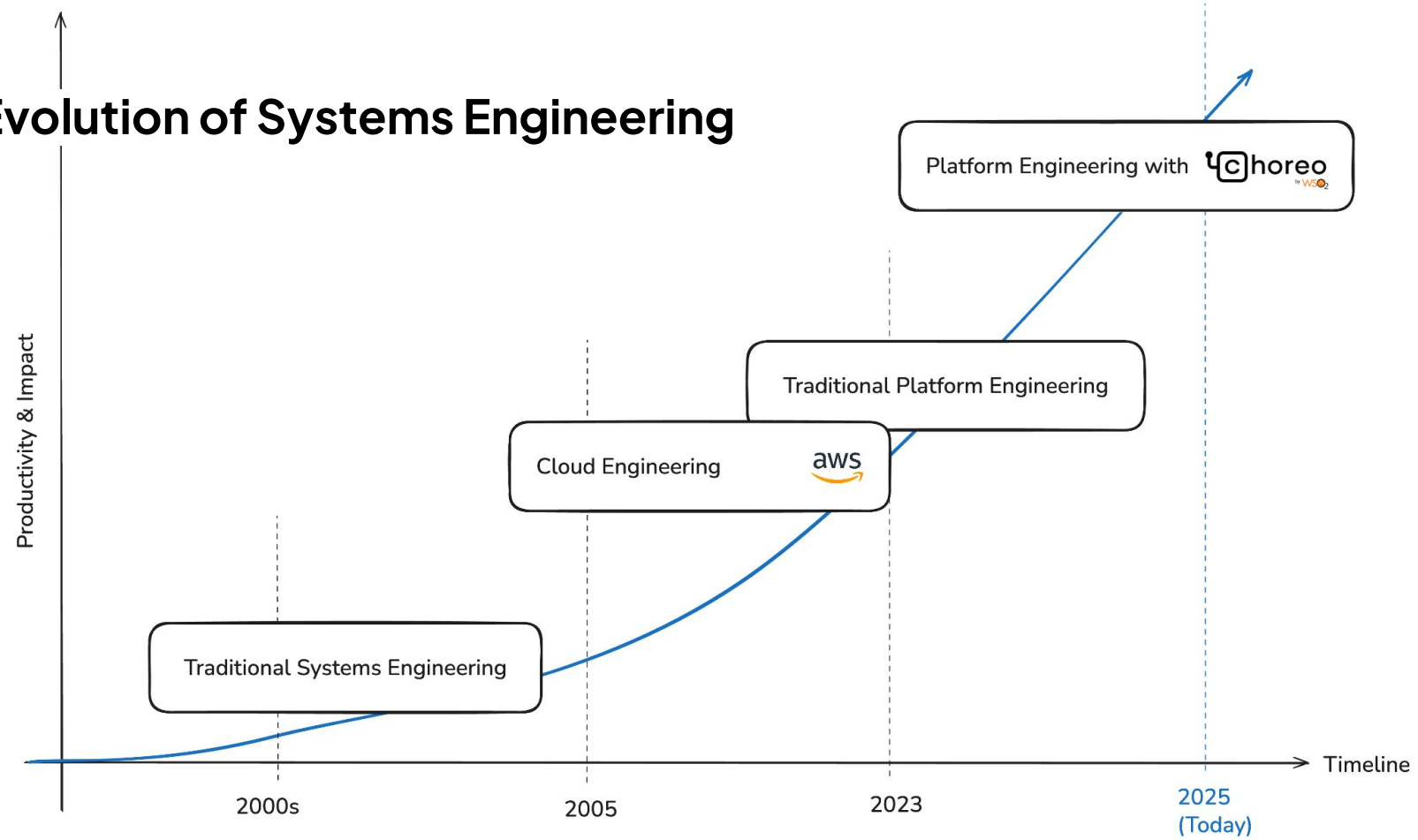
Source: https://www.linkedin.com/posts/activity-7209558179402170368-gLxp

Gartner® Hype Cycle™ for Platform Engineering 2025



Source: https://www.gartner.com/en/documents/6586902

Evolution of Systems Engineering



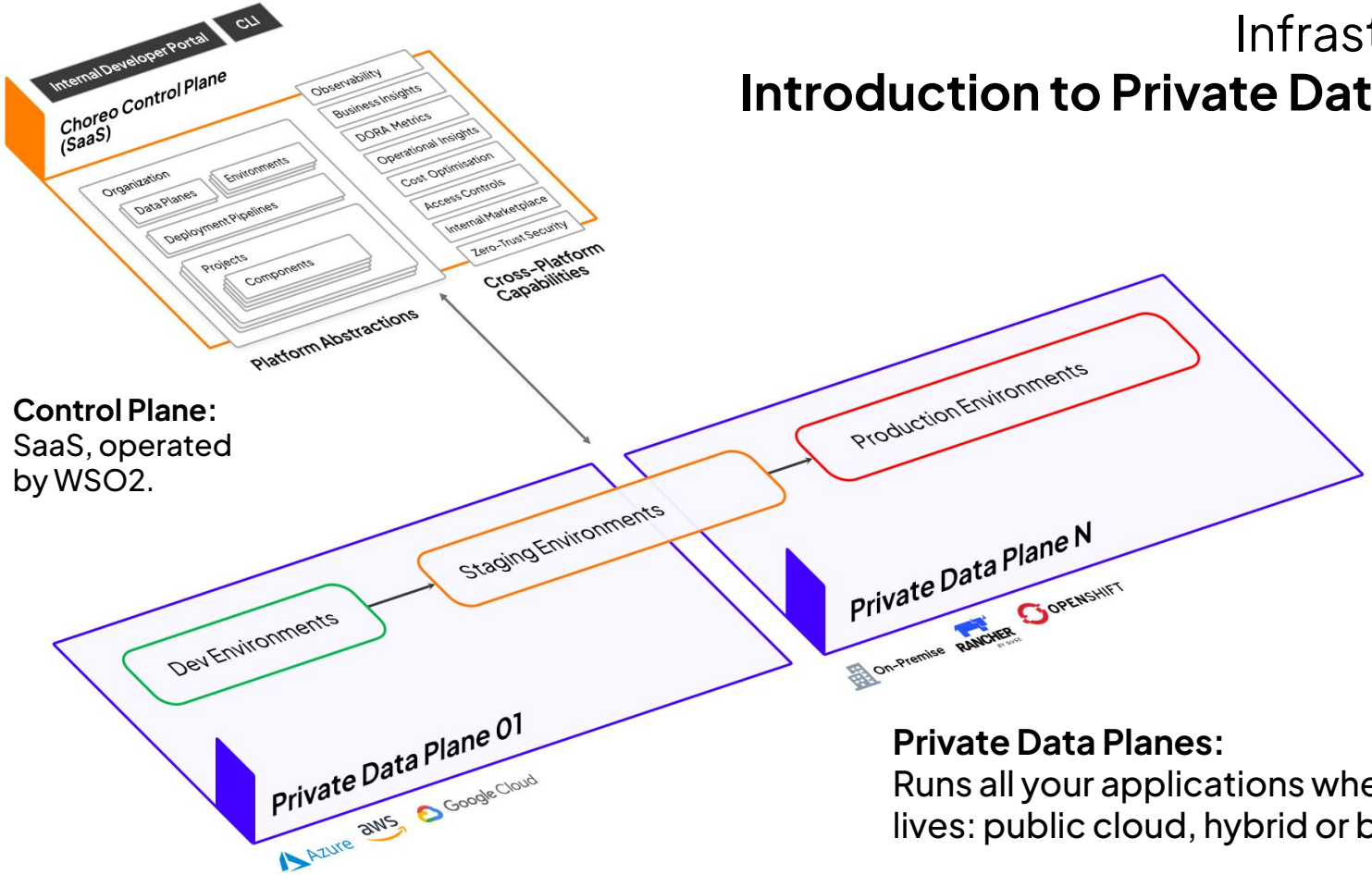
Lab: Introducing the Platform Engineering Perspective (UI)

1. **Infrastructure:** Building a Foundation for Scale

Topics:

- Intro to **Private Data Planes (PDPs)** on Choreo.
  - PDP Management Models and Types
  - PDP Architecture.
- Working with **Platform Services.**
  - Choreo Automation Pipelines
  - Choreo-managed Postgres, MySQL, Valley and Kafka as a Service.
- Lab:
  - Provisioning a One-Click PDP on Vultr.
  - Set up an automation pipeline with Terraform.

Infrastructure:
Introduction to Private Data Planes



Control Plane:
SaaS, operated by WSO2.

Private Data Planes:
Runs all your applications where your data lives: public cloud, hybrid or bare-metal k8s.

A Bird's Eye View

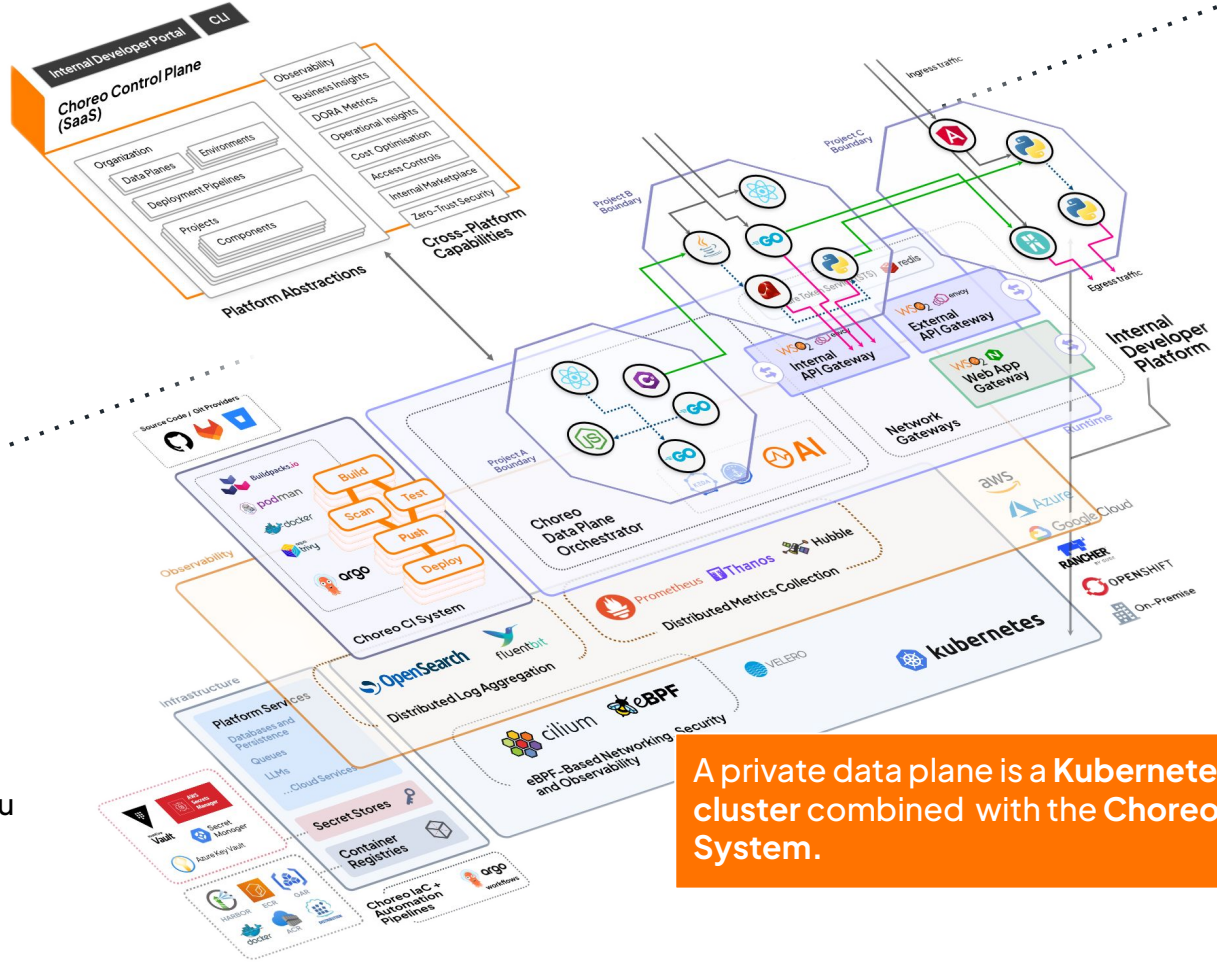
1. The SaaS Control Plane (CP)

Provides platform capabilities via the Choreo Console UI and CLI.

Managed and operated by WSO2 as a SaaS.

2. A Private Data Plane (PDP)

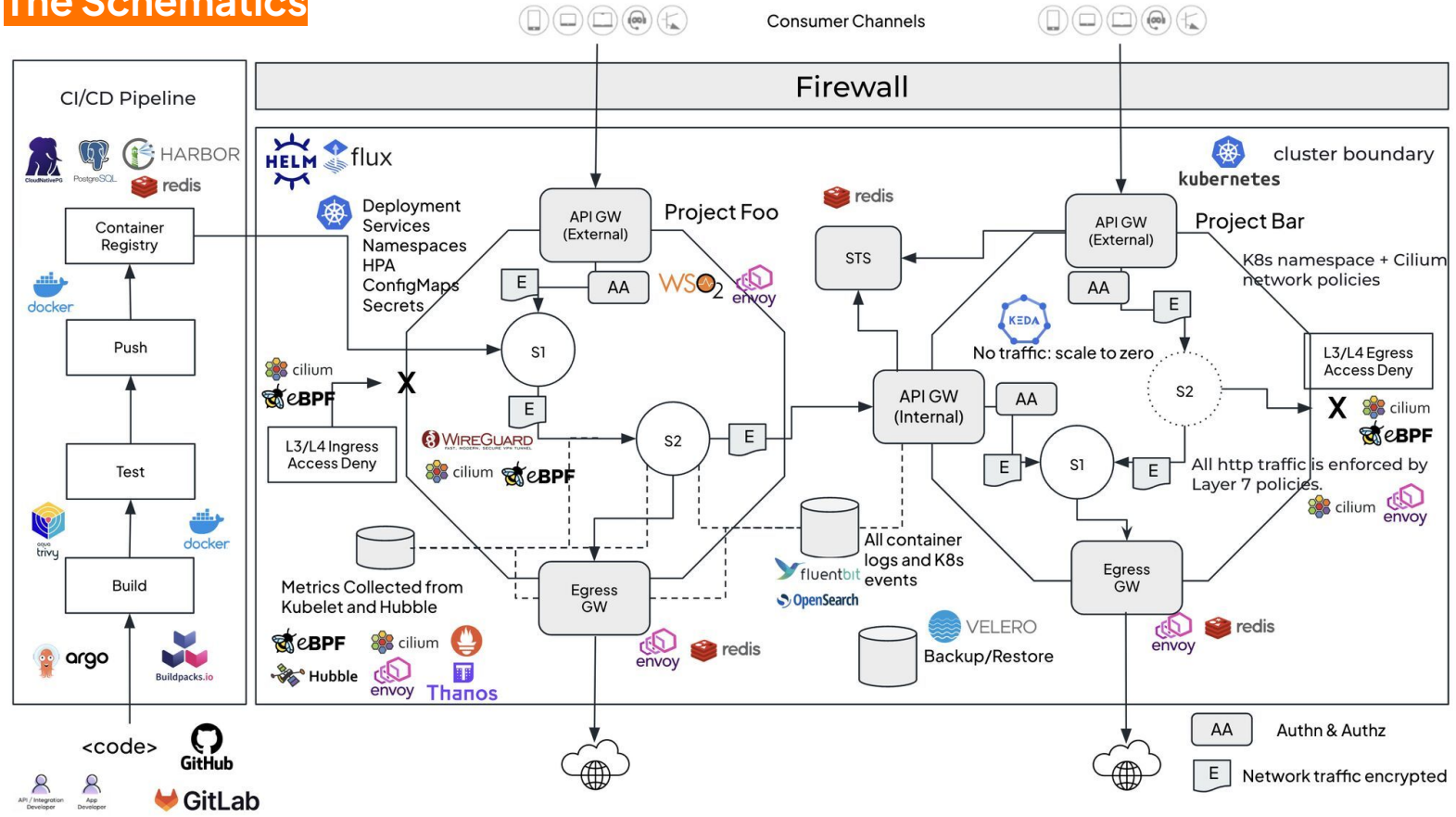
This runs your applications and workloads–on the cloud or even on you on-prem k8s.



A private data plane is a **Kubernetes cluster** combined with the **Choreo System.**

Private Data Plane Architecture

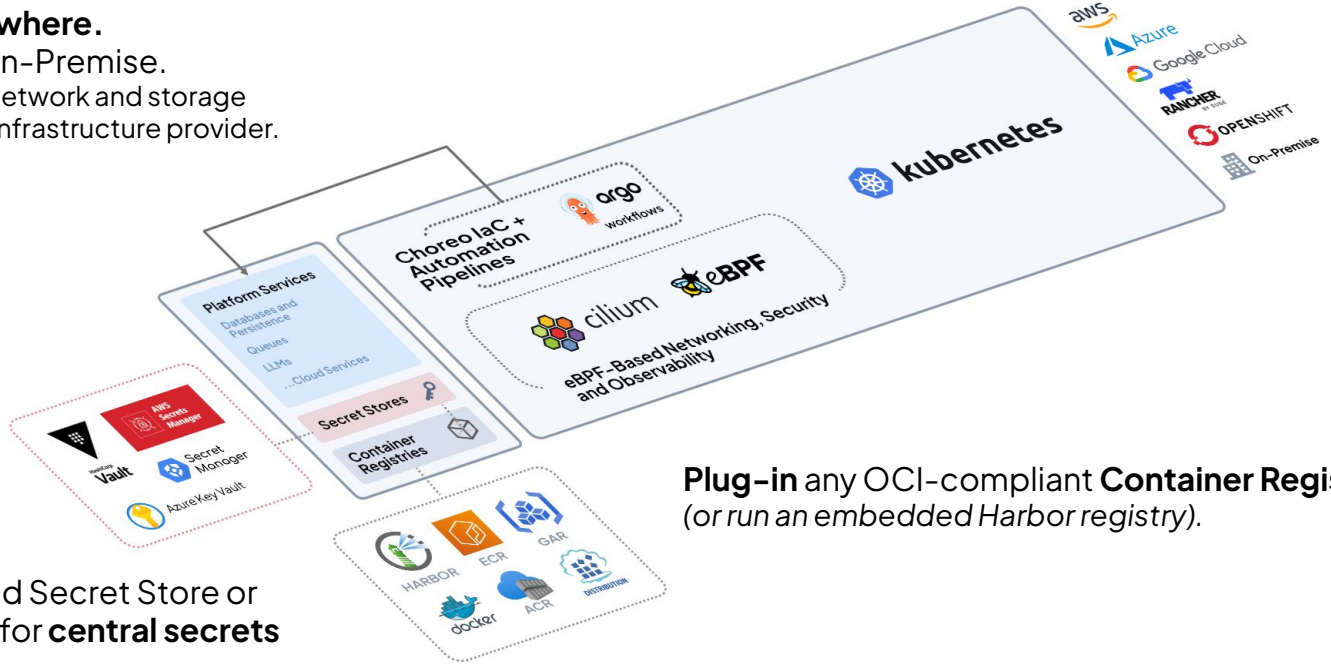
The Schematics



PDP Architecture: Layer 1
Kubernetes & Infrastructure

Built-in DevOps Automation Pipelines for **IaC** and **provisioning platform services.**

Runs on k8s anywhere.
Cloud/Hybrid/On-Premise.
Requires minimum network and storage primitives from the infrastructure provider.



Plug-in any OCI-compliant **Container Registry** *(or run an embedded Harbor registry).*

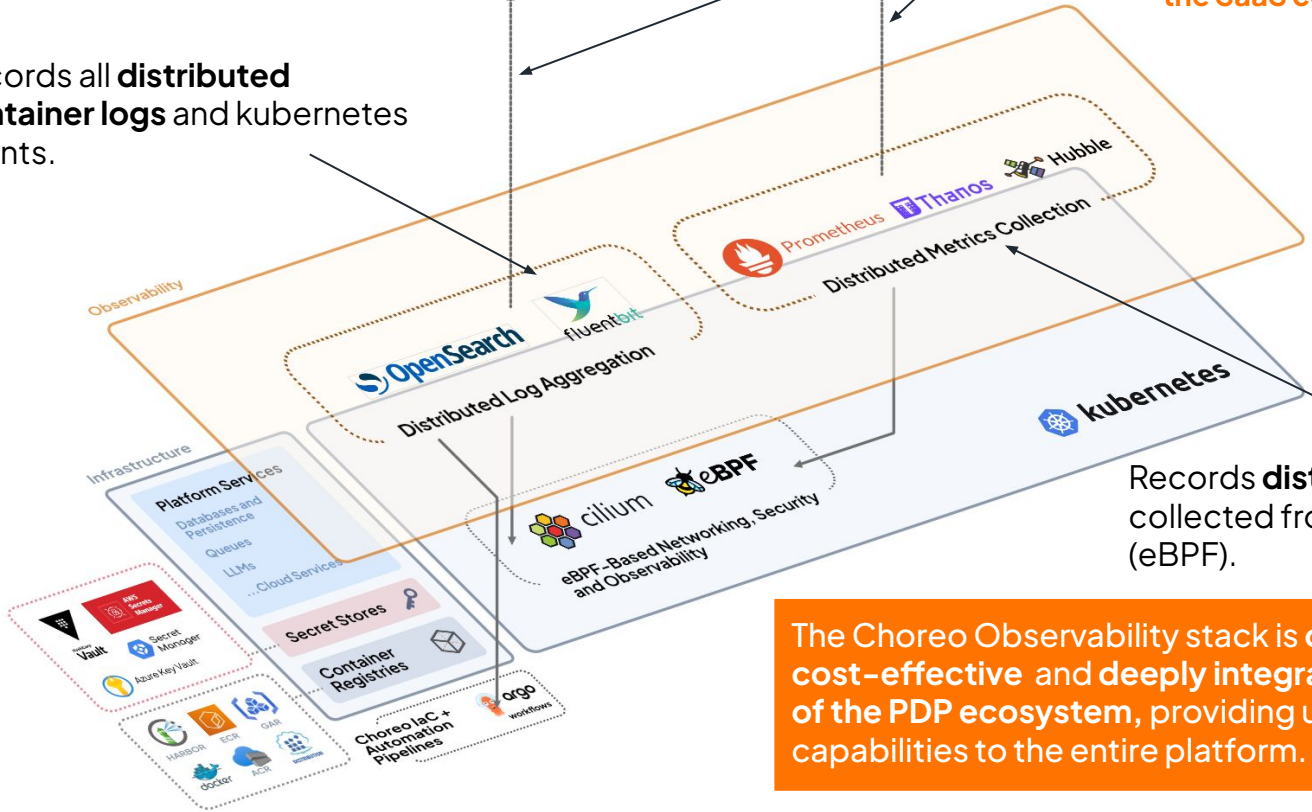
Plug-in any Cloud Secret Store or HashiCorp Vault for **central secrets management.**

PDP Architecture: Layer 2
Embedded Observability Stack

Exposes hooks and APIs for upper layers to integrate with.

PII-data is never sent through the SaaS control plane.

Records all **distributed container logs** and kubernetes events.



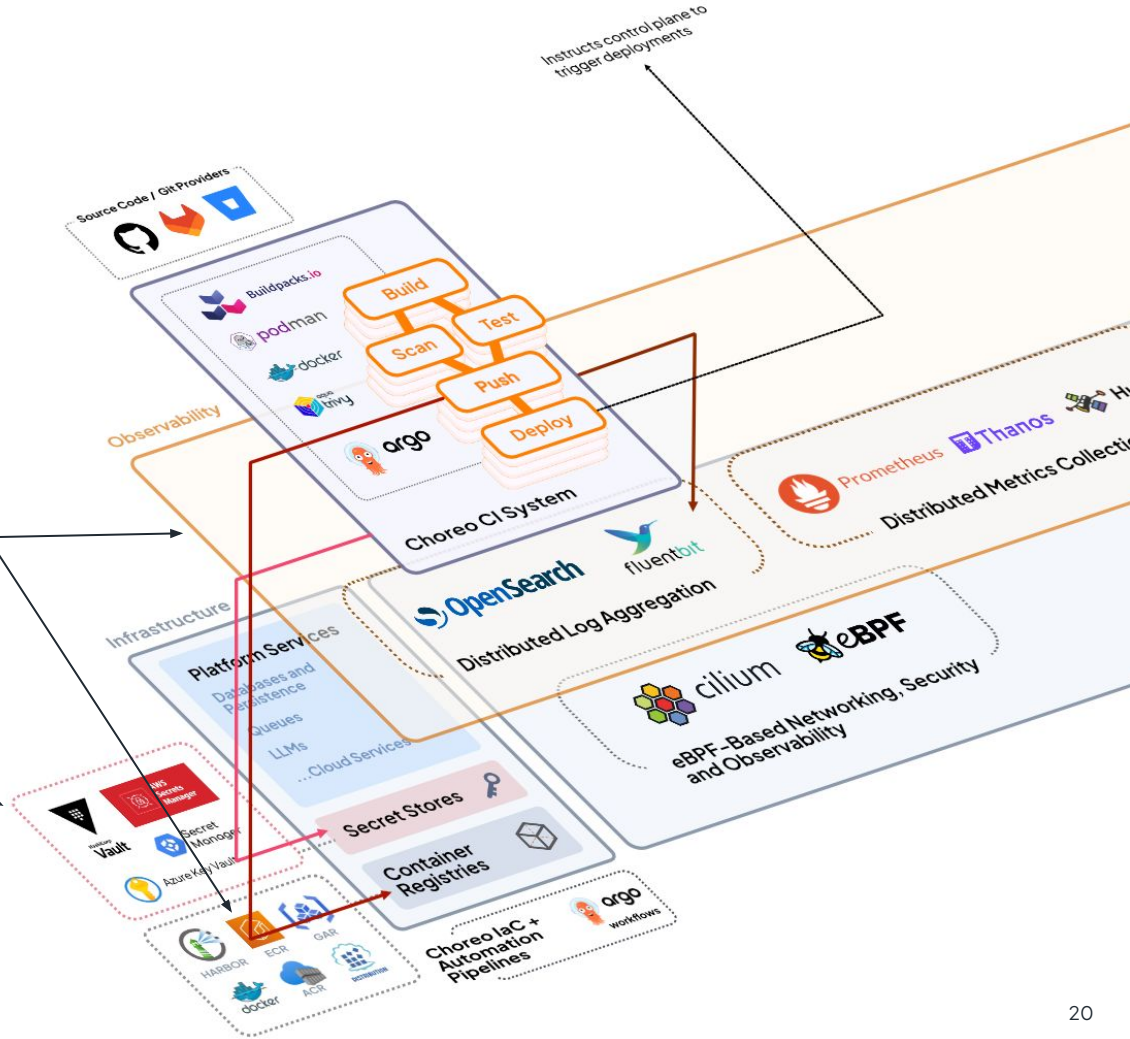
Records **distributed metrics** collected from Kubelet and Hubble (eBPF).

The Choreo Observability stack is **cloud-agnostic**, **cost-effective** and **deeply integrated into all other parts of the PDP ecosystem**, providing unified observability capabilities to the entire platform.

PDP Architecture: Layer 3
The Choreo Build (CI) System

Enables zero-touch CI pipelines for developers with great flexibility.

- Automatically pushes and caches build artifacts to connected container registries.
- Build logs collected and queried via the observability layer.
- Build-time secret and config management via connected secret store/s.



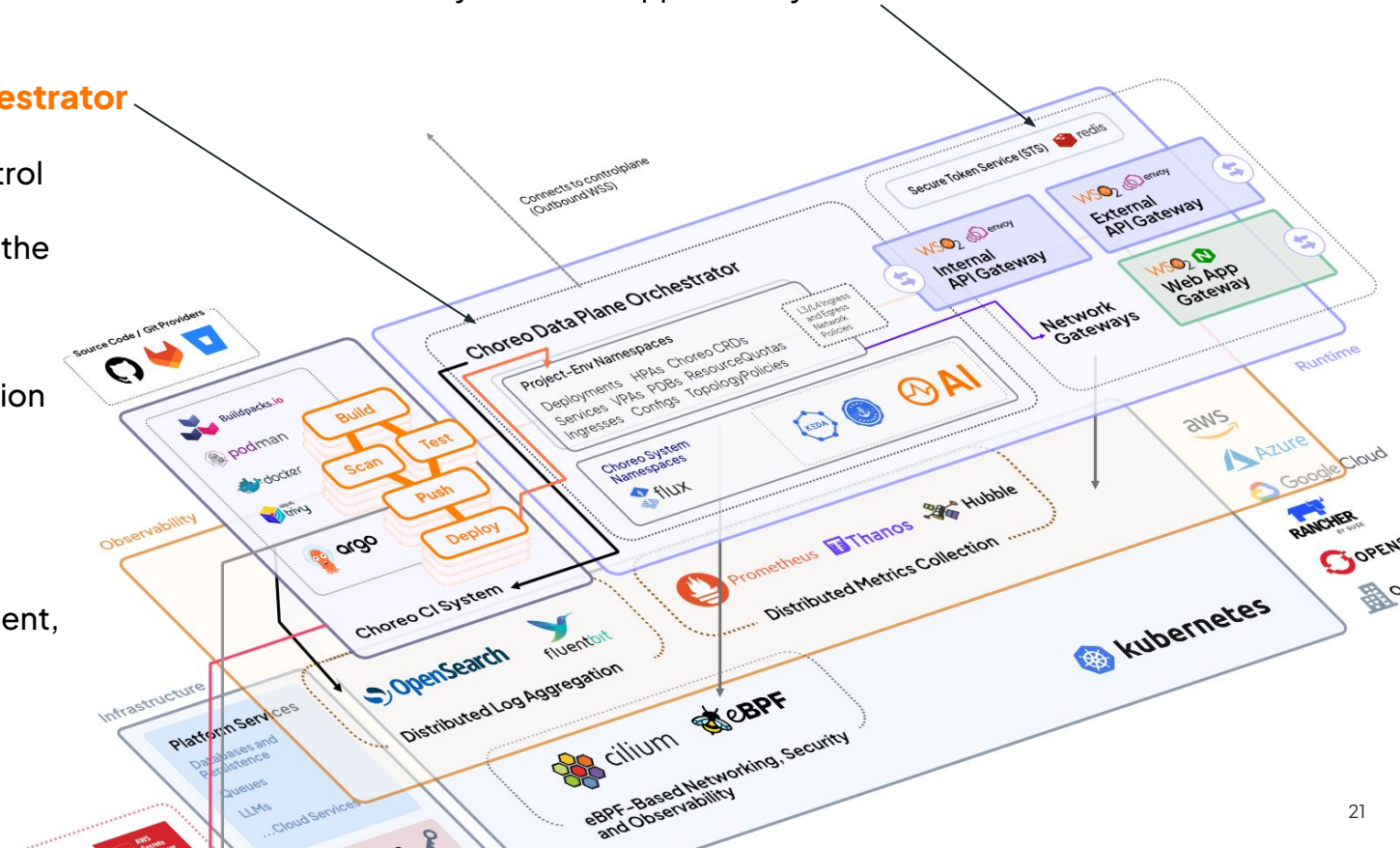
PDP Architecture: Layer 4
The Choreo Runtime

1. The Data Plane Orchestrator

- Connects to the control plane to orchestrate deployments across the PDP.
- Creates logical separation and isolation for multi-tenancy.
- Manages ecosystem toolchains for scale-to-zero, certificate management, AI capabilities, etc.

2. Network Gateways

An Embedded Internal and External API Gateway and a Web App Gateway.

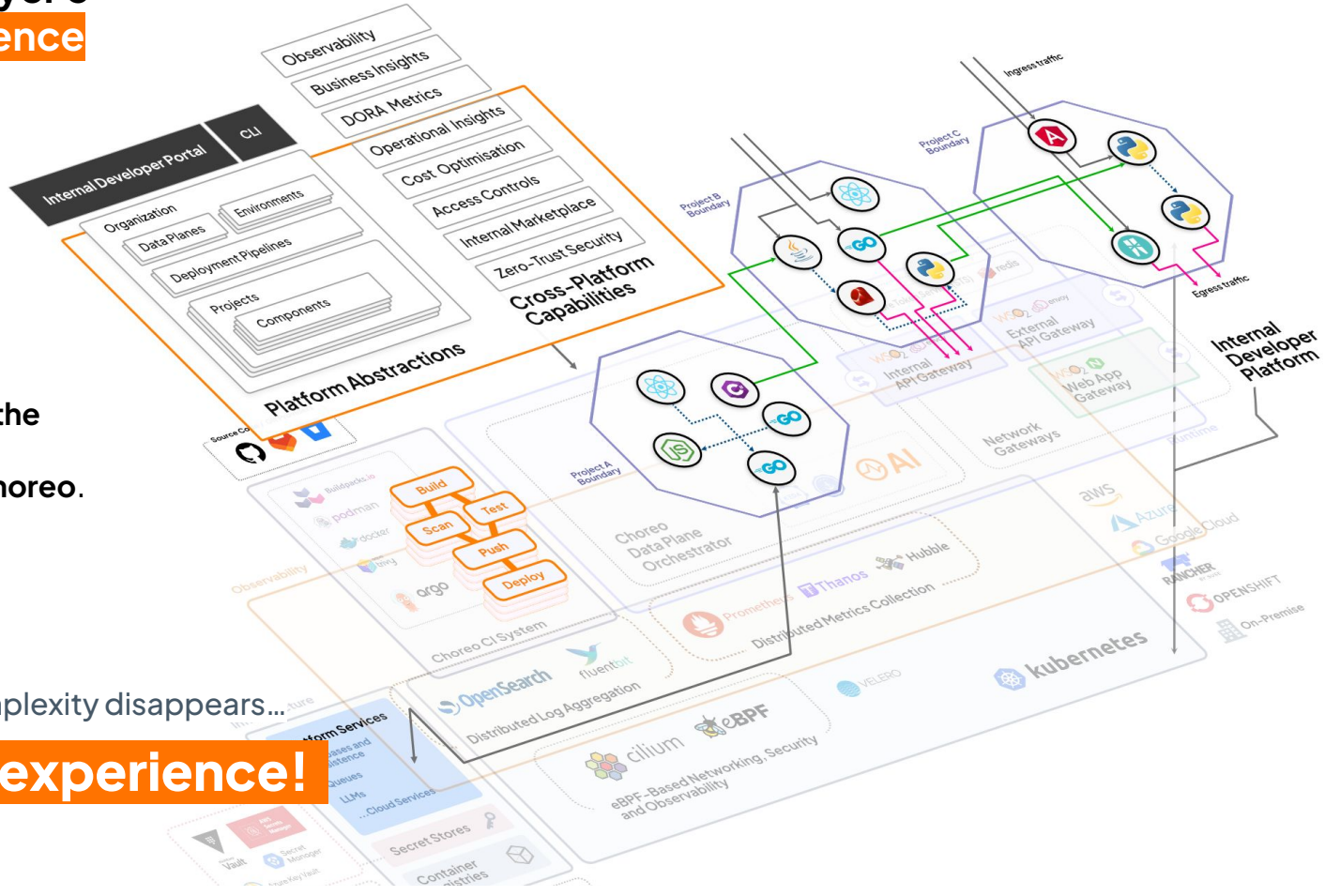


PDP Architecture: Layer 5
The End-User Experience

1. Platform Operators configure their IDP.

Choreo PDPs auto-adjust to these settings.

2. Developers interact with the platform capabilities and abstractions provided by Choreo.



The underlying platform complexity disappears...

A *Platformless* experience!

PDPs: Modes of Operation

1. Fully-managed PDPs as a Service by WSO2
   - Infrastructure and subscription owned by WSO2.
   - One-click PDPs.
2. (Customer) Self-managed PDPs
   - “Bring your own Kubernetes”
   - WSO2 manages only the SaaS control plane, and provides OTA upgrades and patches to the Choreo System.
3. Managed Services - Fully-managed PDPs on Customer’s cloud subscriptions/infrastructure.

Lab: Provisioning a PDP
(One-Click PDPs)

Infrastructure: **Provisioning Platform Services**

Platform Services: Databases, Caches, Queues, and other cloud services that are are part of your solution's architecture.

Choreo provides two ways to provision and manage platform services:

- Fully-managed (*including upgrades, backups, high-availability*) databases (MySQL and PostgreSQL), caches (Valkey) and Queues (Kafka).

- Provision using your own Infrastructure-as-Code (IaC) with Choreo Automation Pipelines.

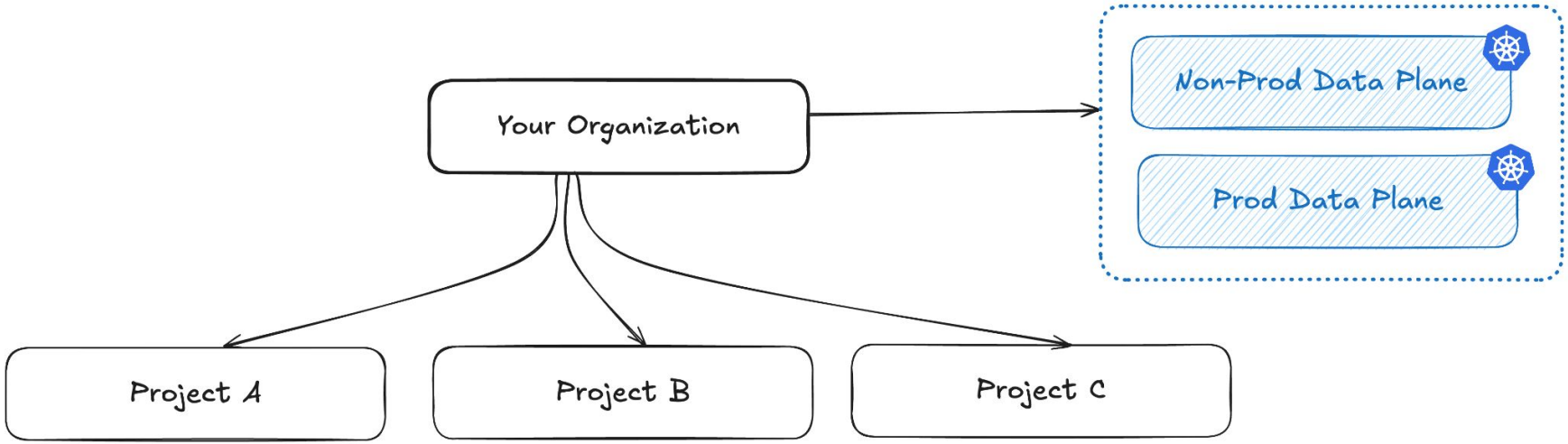
Lab: Platform Services and Automation Pipelines

2. Environments & CI/CD Pipelines: Reliable Releases at Scale

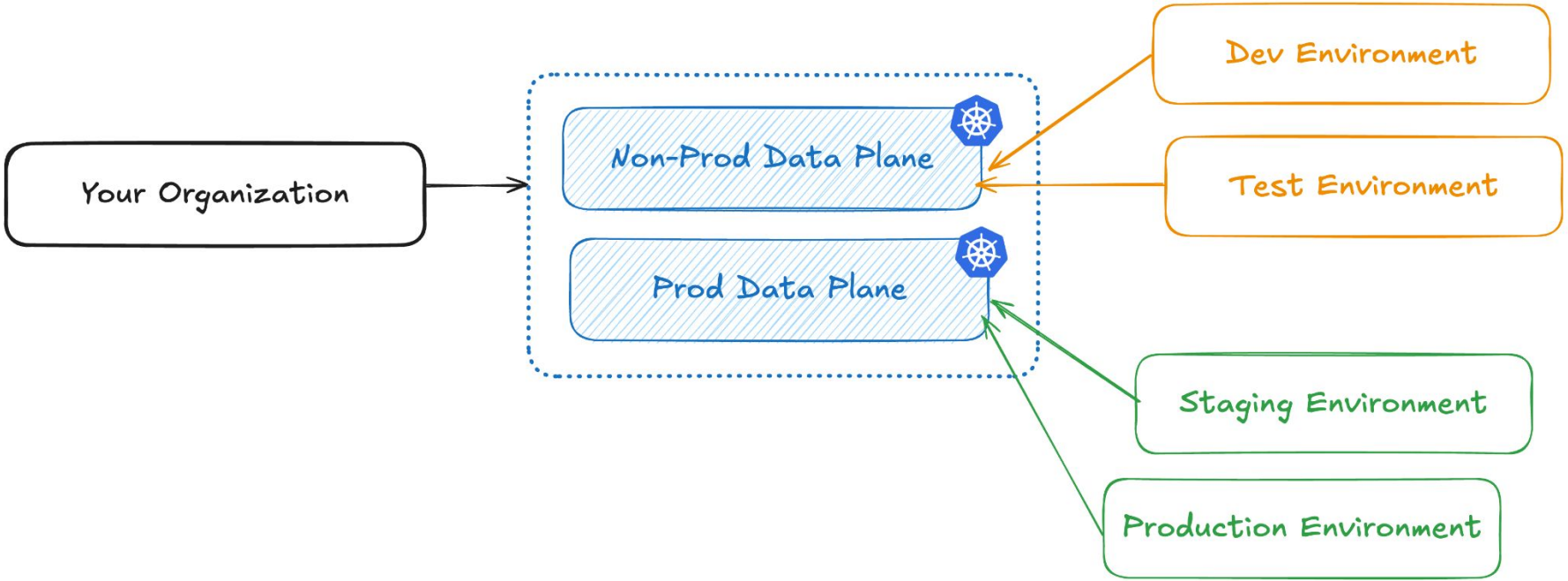
Topics:

- Recap on Platform Concepts in Choreo:
  - Data Planes, Environments, CD Pipelines, CI Pipelines
- Lab:
  - Create new staging environments on the new PDP.
  - Create new delivery pipelines for the new environments.
  - Create hotfix pipelines.
  - Set up workflow approvals for environment promotions.
  - Set up CI Pipelines

Working with Platform Abstractions



Logical and/or physically isolated deployment environments can be created on connected data planes.



Choreo follows a "**build-once-and-promote**" container strategy.**CD Pipelines** can be designed to promote application builds between environments.

- Even across clouds and different clusters – ie. *multi-cloud* and *hybrid cloud* deployments are possible with Choreo.

Example 01: Default CD Pipeline



Example 02: Production-Hotfix Pipeline



Lab:

- Creating Environments
- Creating CD Pipelines

Working with CI Pipelines

- Choreo can build applications written in any programming language.
  - Use Buildpacks that can automagically build a container from source code.
  - Or bring your own Dockerfile.
  - Or deploy an image from an external container registry (using an external CI system like Jenkins, Github Actions, etc).

- This native build system is designed for developer productivity
  - It “just works” out-of-the-box (with best practises like container vulnerability scanning enabled by default)
  - But it gives platform operators full control of their organization’s CI pipelines.


Demo: Customizable CI Pipelines

3. Zero-Trust & Governance: Securing Your Growing Platform

Topics:

- Cell-based Architecture for Software Engineering.
  - Domain driven design
  - Network components
  - Logical isolation and multi-tenancy
- The Zero-Trust Architecture.
- Sidecar-less service mesh
- End-to-end network encryption
- Egress Control
- Lab:
  - Governance: Enforce container vulnerability scans, Egress Control

4. Observability & Enabling Site Reliability Engineering (SRE)

Topics:

- Intro to Choreo's In-Data Plane Observability Architecture: Compliant and Cost Effective.
- Lab:
  - Deeply Integrated Observability:
    - Runtime logs
    - Metrics
    - Service Map - Dynamic view vs Static view
  - Audit logs
  - Monitoring and Alerting

6. Platform Insights: Measuring to Improve.

- Lab:
  - Operational dashboards
  - Delivery dashboards
  - Business dashboards

Question Time!




Thank you!
20TH ANNIVERSARY EDITION
WSO2CONASIA
PLATFORMLESS MODERNIZATION