WSO2CON 2025

Unified Management of Ingress and Egress Across Multiple API Gateways

WSO2CON 2025

Sanjeewa Malalgoda
Director, Engineering
WSO2

Arshardh Ifthikar
Technical Lead
WSO2

Agenda

- The Growing Complexity of API Ecosystems
- Complexities with Centralized and Federated API Management
- The Need for a Single Control Plane with Gateway Federation
- Demonstration 1: Behavior of the Unified API Control Plane (ACP) with Multiple Gateways
- Ingress/Egress API Management
- Demonstration 2: AI Gateway principles and the functionality of AI/LLM APIs
- Future Roadmap
- Key Takeaways & Next Steps

Introduction & Lab Objectives

Introduction & Lab Objectives

- **Objective of the Lab**
  - Understand the need for unified API management by exploring the challenges of managing ingress and egress across multiple API gateways.
  - Demonstrate features in WSO2 API Manager to achieve a Unified API Management Experience.
  - Learn how to integrate decentralized API gateways into a single control plane for consistent security, governance, and observability.
  - Gain hands-on experience in configuring and managing API traffic across different gateways, ensuring seamless connectivity and policy enforcement.

The Growing Complexity of API Ecosystems

APIs are everywhere. As organizations scale, they deploy **multiple API gateways**

- Different security requirements of different teams.
- For different cloud environments
- Now, even AI-specific gateways
- To comply with various protocols and platforms

But with this growth comes **complexity**: inconsistent policies, fragmented security, and siloed observability. How can we bring order to this chaos? That's where unified ingress and egress management comes in.

Decentralized API Management

- Independent Control: Each team manages its own APIs and related assets.
- Team-Specific Policies: Security, access, and governance vary across teams.
- Distributed Traffic Routing: APIs use different gateways, portals, etc.
- Custom Infrastructure: Teams provision their own hosting, databases, and networking.
- Flexible Processes: API design and standards differ based on team preferences.
- Quick Deployments: New APIs and gateways can be added without central approval.

Concerns with Decentralized API Management

Lack of Standardization – Inconsistent security, governance, and documentation across teams.

Security Risks – Increased exposure to vulnerabilities without centralized control.

Compliance Challenges – Difficult to enforce regulatory and policy adherence.

Scalability Issues – Managing multiple APIs across different teams can create inefficiencies.

Observability & Monitoring – Harder to track usage, errors, and performance metrics.

Versioning & Dependency Conflicts – Risk of breaking changes across interconnected services.

Increased Operational Overhead – More effort required for coordination and maintenance.


This is too much trouble. I am going to Centralize all my APIs to a single platform!

Centralized API Management

- Single Control: One team manages all APIs from a central platform.
- Policy & Security: Rules, access, and security are set and enforced centrally.
- Traffic Routing: All API requests go through a main gateway.
- Infrastructure Setup: The central team provides hosting, databases, and networking.
- Standardized Process: APIs follow common design and governance rules.
- Approval Needed: Adding new APIs or gateways requires central team approval.

Concerns with Centralized API Management

Slow onboarding – New APIs, gateways, and policies take time to approve.

Limited team flexibility – Teams must wait for central decisions.

Generic solutions – One-size-fits-all approach may not suit all teams.

Single point of failure – Central issues can disrupt all APIs.

Innovation bottlenecks – Strict approvals slow experimentation.

Scalability issues – Central team struggles as APIs grow.

High operational overhead – Heavy workload on the central team.

Lack of team autonomy – Limited control over API development.

Slow adoption of new tech – Hard to integrate emerging API tools.

What if there was a middle path?

This is where the requirement for a Unified API Control Plane comes in

The Need for a Central Control Plane

Why Do We Need Central Control Plane?

Challenges with API Sprawl:

- Organizations use multiple API gateways across different environments (AWS, Kubernetes, Solace, etc.)
- Lack of a single control plane leads to inconsistency in policy enforcement, security, and monitoring

Key Questions to Answer:

- How can we centrally manage different gateways?
- How do we ensure security, governance, and operational efficiency at scale across multiple gateways?

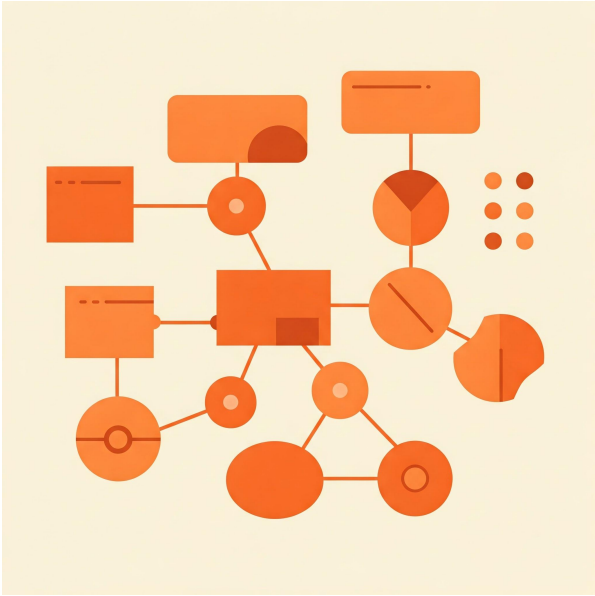
A Single Control Plane for API Gateways

What is an API Control Plane (ACP)?

A centralized system to manage multiple API gateways, gateway deployments and enforce policies

Key Capabilities of ACP:

- Unified ingress and egress management
- Policy governance across all environments
- Observability and analytics



Capabilities Of API Control Plane

- Policy Management – Security, rate limits, authentication, governance.
- Multi-Gateway Support – Manage WSO2, AWS, Apigee, Kong etc.
- API Discovery – Centralized registry for all APIs.
- Observability – Monitoring, logging, tracing, analytics.
- Access Control – Multi-tenant RBAC for teams and users.
- Automation – CI/CD, policy enforcement, gateway provisioning.
- Hybrid & Multi-Cloud – Sync across on-prem, cloud, multi-cloud.
- Self-Service – Teams manage APIs within governance rules.

API Management Software

WSO2 API Control Plane

WSO2 Kubernetes Gateway

K8s native design, Lightweight, Envoy based

WSO2 Immutable Gateway

Offline mode, Immutability, Edge Gateway

WSO2 Universal Gateway

Inbuilt mediation, Range of protocols

Federated Gateways

AWS API Gateway

Gateway Federation

Gateway Federation and Multi-Gateway Management

The Challenge

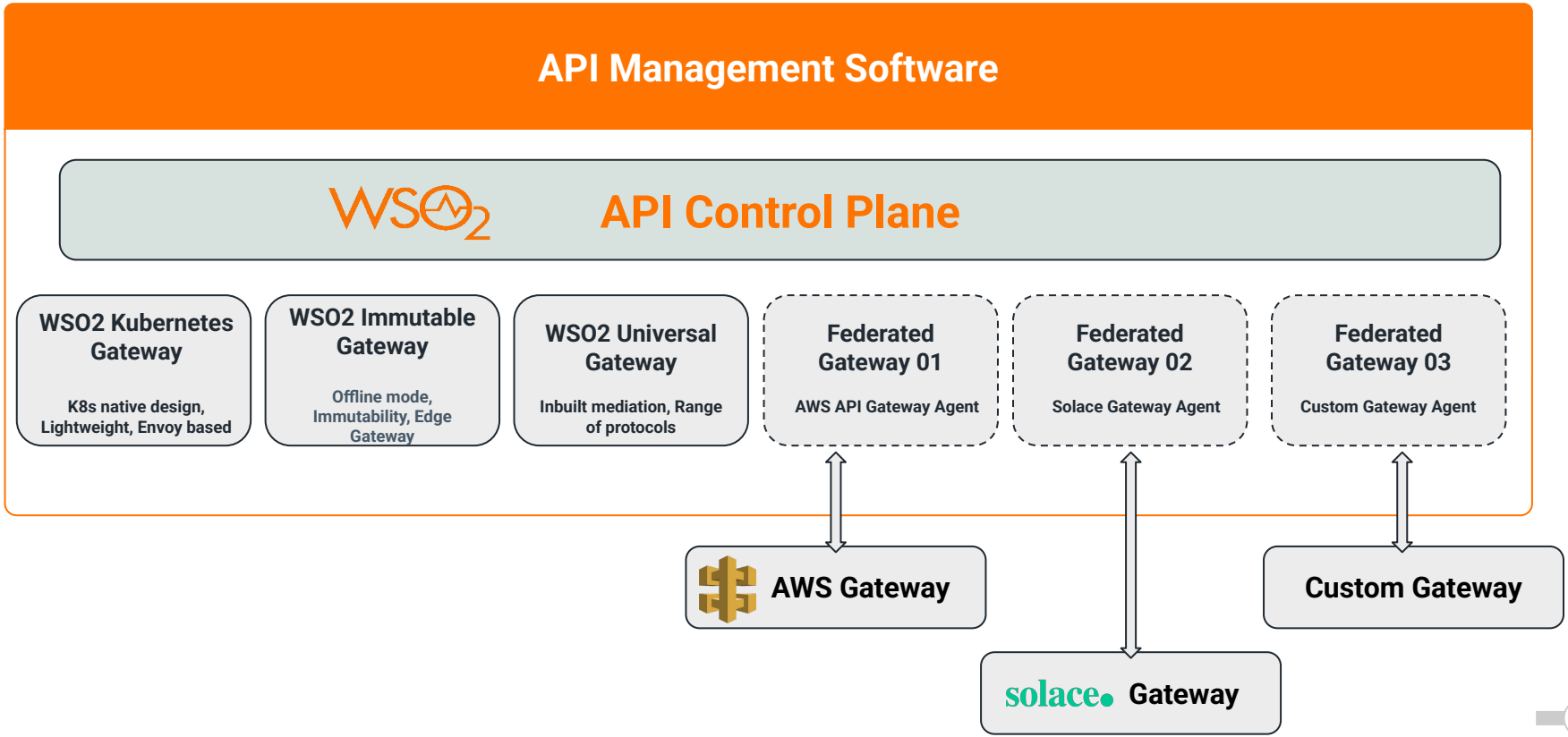
- No centralized management, requiring separate configurations and monitoring
- Inconsistent security policies across multiple gateways
- Fragmented traffic control causing inefficiencies
- Varied configurations and UI increasing complexity

Gateway Federation and Multi-Gateway Management

How Gateway Federation Solves It

- Single control plane for centralized management
- Unified security and policy enforcement
- Multi-gateway support across WSO2 and third-party gateways
- Federated traffic management for routing and security
- Standardized configuration and UI for consistency

How We Extend Gateway Support: Agent Framework



Demo 01

Behavior of the Unified API Control Plane (ACP) with Multiple Gateways

The Scenario

Company: ForbizLogic, a fast-growing enterprise with services running across multiple clouds and regions.

Challenge: They have different API gateways across AWS, and on-prem data centers. Managing policies, security, and traffic is fragmented.

ForbizLogic

Eventing APIs Team	Financial Services Team	Other IT Teams	
Requirement to use Solace Broker to apply QoS for eventing APIs	Services are in k8s. Looking for k8s native gateway for their APIs	Some cloud services are running on AWS. Need a GW closer to services	Need a wide variety of protocols and mediation capabilities, 2 regions
solace. Gateway	WSO2 Kubernetes Gateway	AWS Gateway	WSO2 Universal Gateway Europe APAC

Unified Control Plane

Ingress/Egress API Management

Securing API Traffic Coming In

What is an Ingress API?

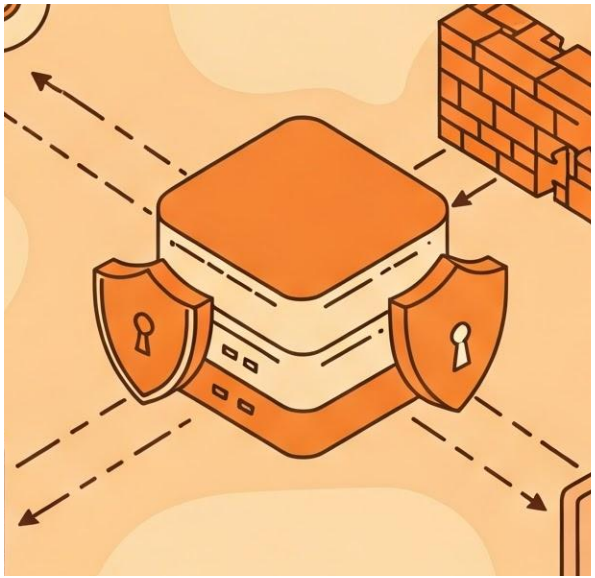
Controls API traffic entering the organization

Ensures security, authentication, and access control

Why Organizations Need It?

Standardized API exposure

Secure and compliant access control



Governing Outbound API Consumption

What is an Egress API?

Controls API traffic leaving the organization

Enforces governance on external API consumption

Why It's Critical for Organizations?

Secure outbound API calls

Cost management and compliance

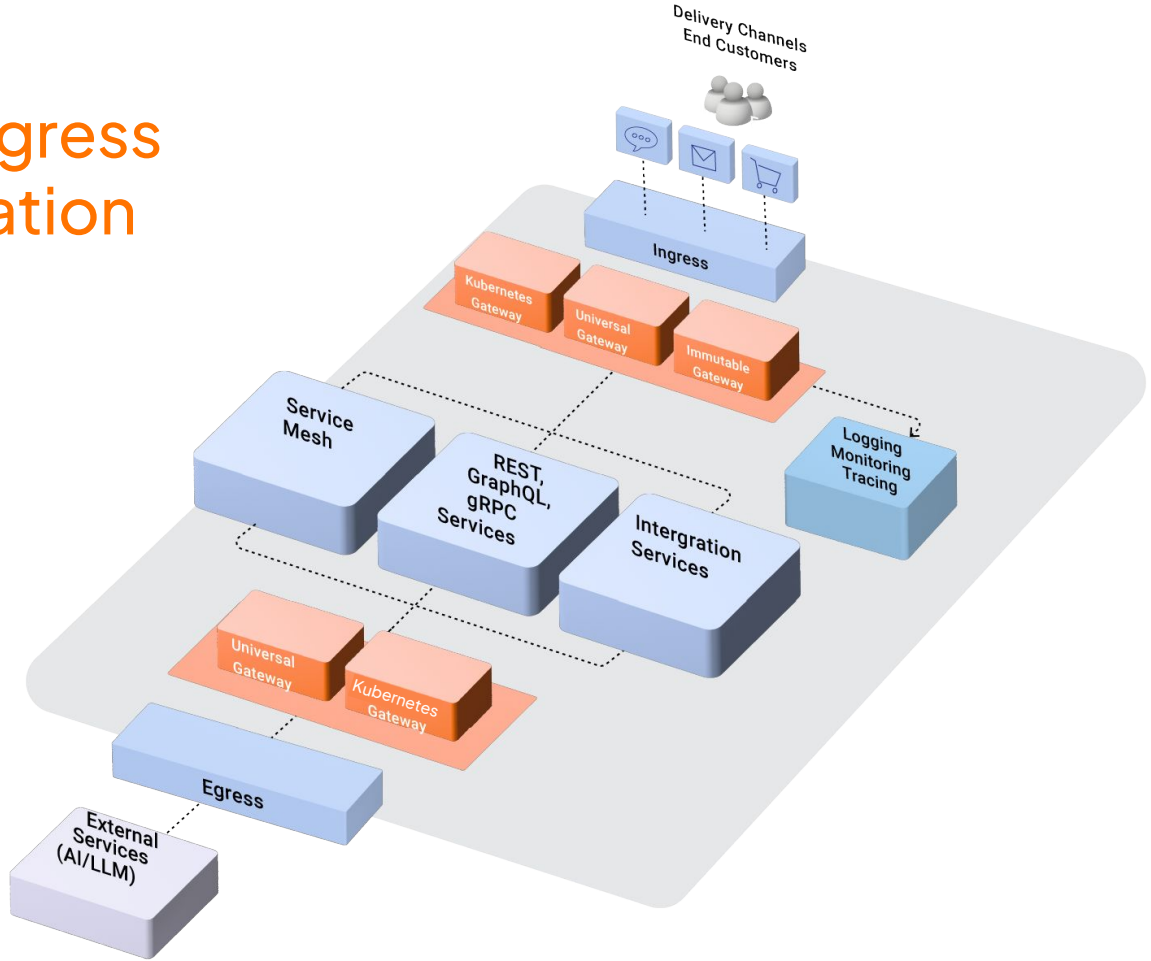
Use Cases:

AI APIs (Azure, OpenAI, Mistral)

Third-party CRM & messaging APIs

- Salesforce
- Twilio
- Slack

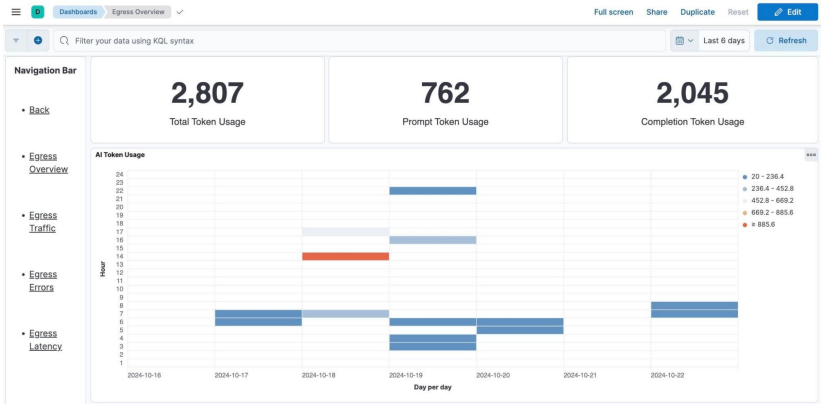
Ingress/Egress API Invocation Flow

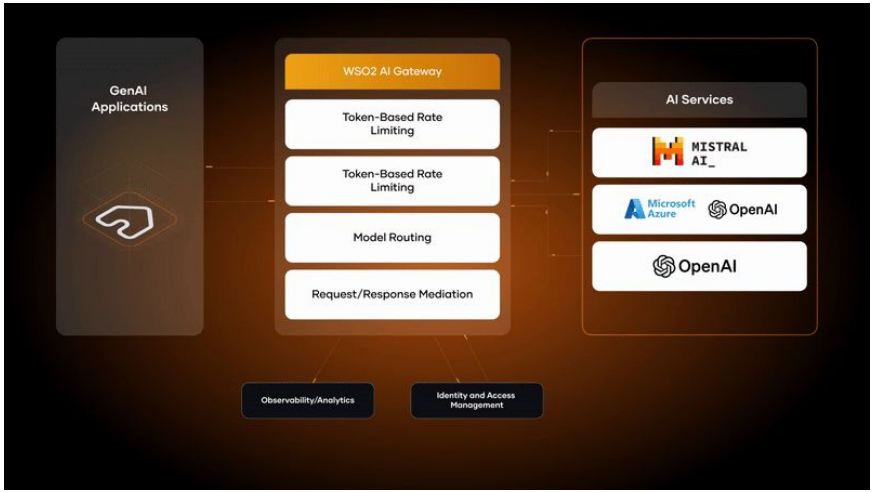


Egress API Management: AI Gateway

WSO2 AI Gateway governs, optimizes, and secures Generative AI API usage with flexible deployment and multi-provider support for seamless AI integration.

- Token-based rate limiting
- AI-specific analytics
- Multi-provider support
- Adaptive request handling
- AI guardrails with request/response mediation





Demo 02

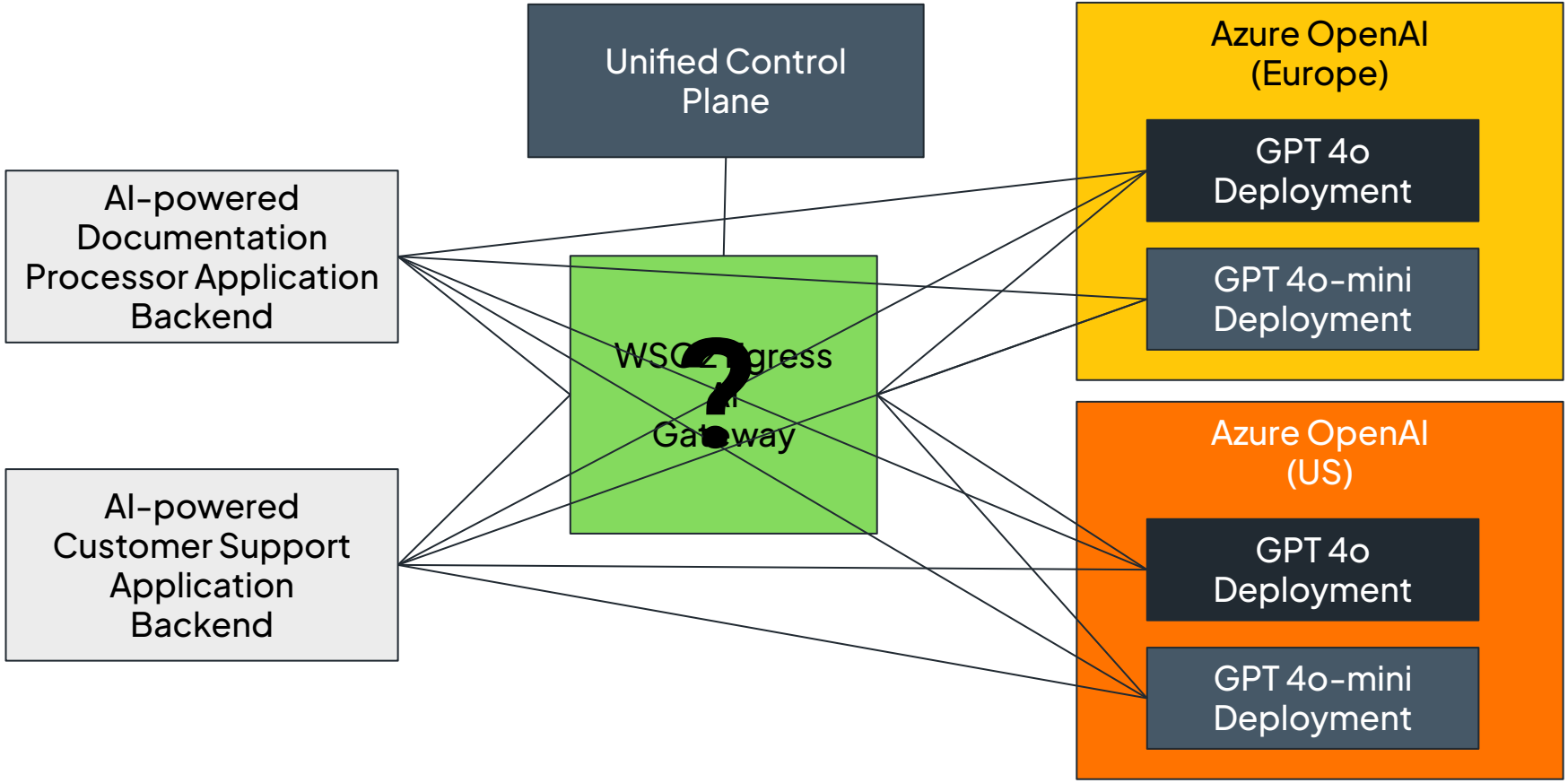
AI Gateway principles and the functionality of AI/LLM APIs

The Scenario

Company: DeepCrestAI, a SaaS provider offering AI-powered document processing and customer support.

Challenge: They rely on Azure OpenAI but face issues with:

- Overloaded models – GPT-4o deployment gets overwhelmed, leading to higher latencies.
- Failover risk – If GPT-4o goes down, there's no fallback.
- Cost inefficiency – All requests go to GPT-4o, even when GPT-4o-mini suffices for some cases.
- Rate limits – The app exceeds Azure's request quotas. Certain applications utilize more tokens than others.
- No Observability as to what is happening
- In case one Azure region fails, there is no backup to a different region


Unified Control Plane
AI-powered Documentation Processor Application Backend
AI-powered Customer Support Application Backend
Gateway
Azure OpenAI (Europe)
GPT 4o Deployment
GPT 4o-mini Deployment
Azure OpenAI (US)
GPT 4o Deployment
GPT 4o-mini Deployment

Future Roadmap

Future Roadmap

- Multi Vendor Routing Support for AI Gateway
- Additional Egress API types will be added in the future
- Additional Federated Types Support (eg: Kong)
- In case there are changes in External Gateways our Agents will be versioned to bridge them.

Key Takeaways & Next Steps

Key Takeaways

- **Unified API management simplifies operations** – Centralizing ingress and egress governance reduces complexity across multiple API gateways.
- **Consistent security and policies** – A unified control plane ensures uniform enforcement of authentication, rate limiting, and compliance rules.
- **Enhanced observability and analytics** – Real-time monitoring across all API gateways provides better insights for performance and security.
- **Scalability & future-proofing** – A unified approach makes it easier to integrate new gateways, cloud services, and AI-based APIs.

Next Steps

- **Evaluate your current API gateway landscape** – Identify gaps in ingress and egress management.
- **Explore control plane solutions** – Assess tools that can unify policies and monitoring across multiple gateways.
- **Start with a phased implementation** – Test unification on a subset of gateways before scaling up.
- **Monitor & optimize** – Continuously refine policies based on usage patterns and performance data.

Upcoming API Management Sessions

Upcoming API Management Sessions

- **Conference Day 3 (Thursday, March 20th)**

API-First Architectures for Modernization →

Thilini Shanika
Associate Director and Head of Engineering, APIM BU, WSO2

Managing Egress APIs for AI and SaaS Applications →

Arshardh Ifthikar
Technical Lead, WSO2

Advancing API Governance for Scalable Systems →

Vidura Gamini Abhaya
Vice President, Solutions Architecture, WSO2

Federated API Management for Enterprise Agility →

Sanjeewa Malalgoda
Director of Engineering, WSO2

Bijira: API PaaS for the Cloud and AI-Native Era →

Pubudu Gunatilaka
Associate Director & Architect, WSO2

Customer Panel: Journeys in Enterprise Modernization Through API Management →

Question Time!


Thank you!
WSO2CON2025