AI Nativity

The path forward

Pre-AI World

Microservices Today

- **Key components**
- **Key Workflows**
- **Key Solutions**
  - **Routing**
  - **Policy Enforcement**
  - **Resiliency**
  - **Observability**
  - **More Complex Solutions**
    - WCP
    - API Gateway, Bubble

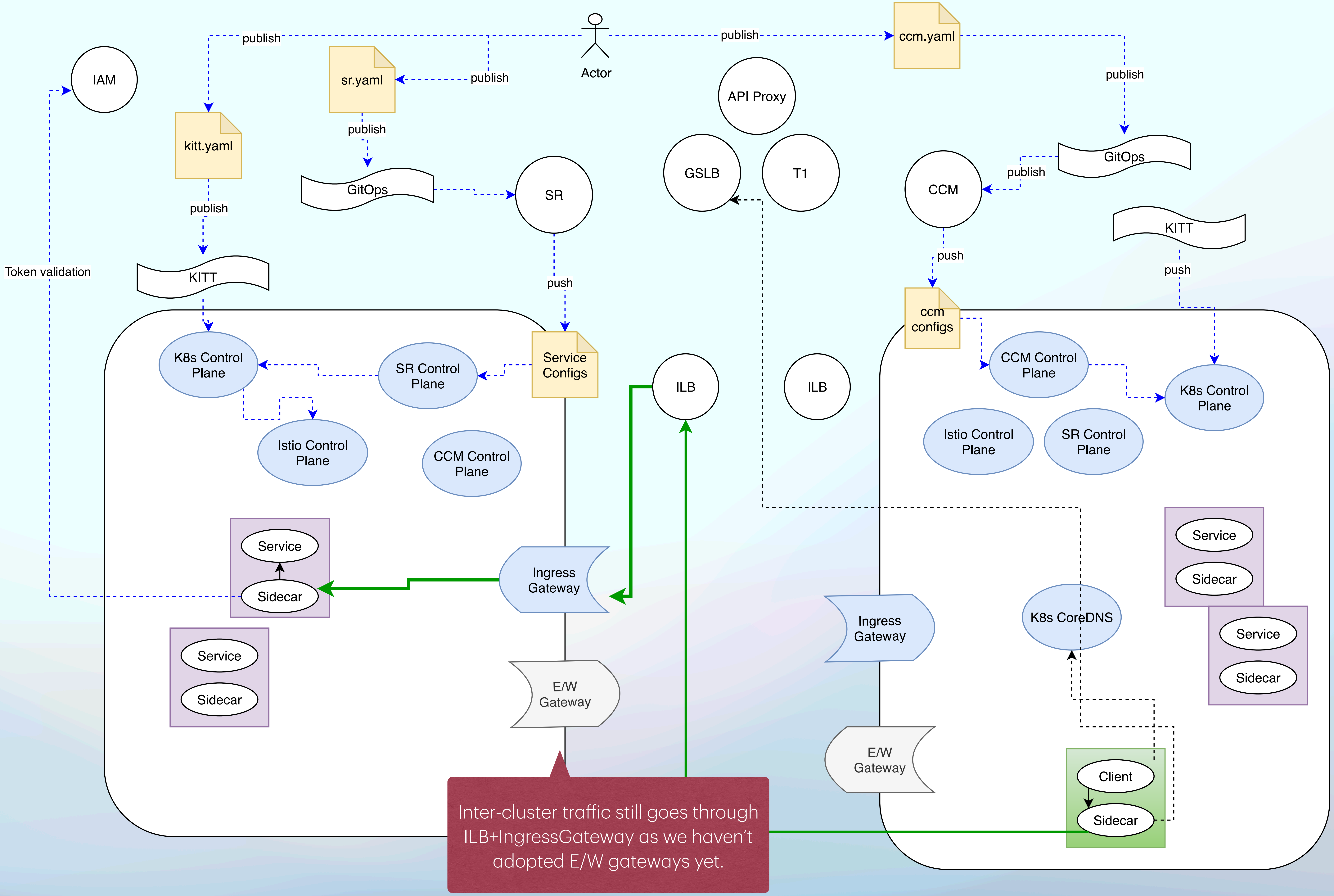
AI World

Where we want to be tomorrow

- **The Journey**
- **New players**
- **New/Adjusted Workflows**
- **Routing**
- **Policies**
- **Resiliency**

Microservices World Today

- **Key components**
  - Services
  - WCNP
  - GSLB, ILB
  - Ingress Gateway (Envoy)
  - Sidecar (Envoy)
  - Istio Control Plane (IstioD)
  - K8s Control Plane (API Server, CoreDNS)
  - SR + Controller



Inter-cluster traffic still goes through ILB+IngressGateway as we haven't adopted E/W gateways yet.

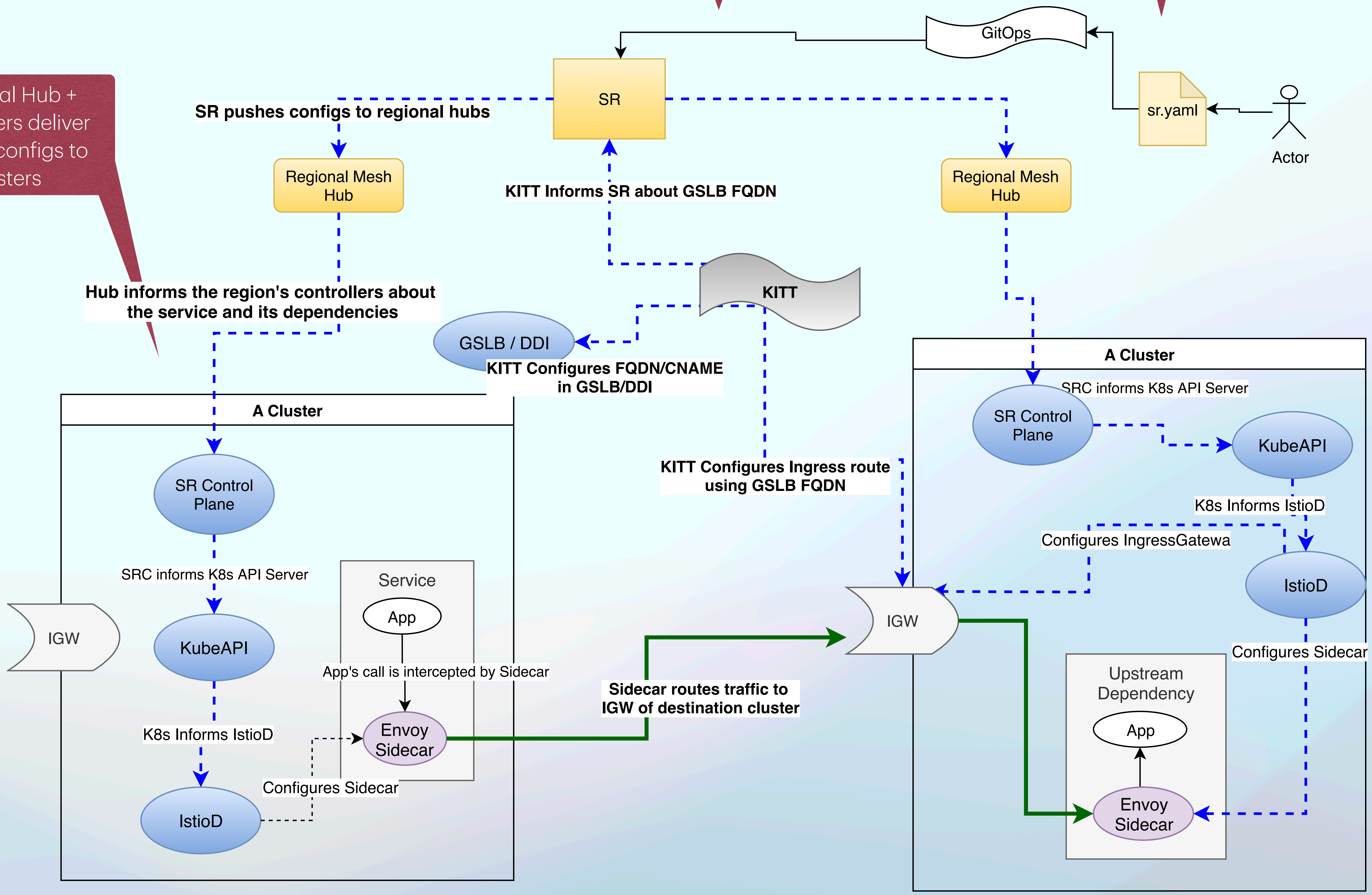
Microservices World Today

- **Key components**
- **Key Workflows**
  - CI/CI: build application artifact
  - KITT: deploy application to K8s
  - SR Gitops: Platform confis: service dependencies, routing, policies, resiliency
  - CCM Gitops: App's configs

Regional Hub + Controllers deliver service configs to clusters

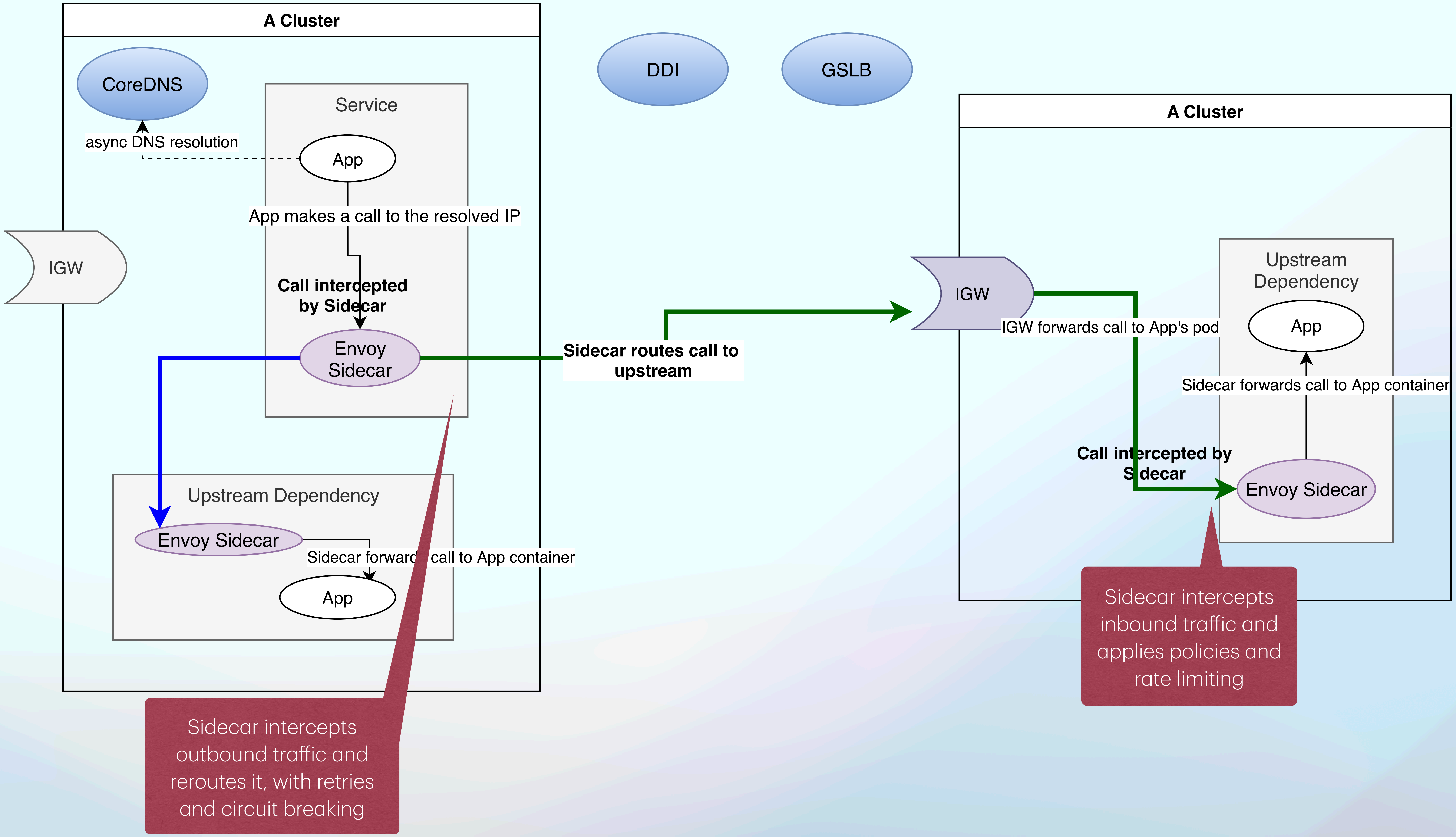
KITT pipeline helps connect the dots when deploying applications,

Services are registered for discovery and deployed through gitops


GitOps
SR
sr.yaml
Actor
SR pushes configs to regional hubs
Regional Mesh Hub
KITT Informs SR about GSLB FQDN
Regional Mesh Hub
Hub informs the region's controllers about the service and its dependencies
KITT
GSLB / DDI
KITT Configures FQDN/CNAME in GSLB/DDI
A Cluster
SR Control Plane
SRC informs K8s API Server
IGW
KubeAPI
Service
App
App's call is intercepted by Sidecar
K8s Informs IstioD
Envoy Sidecar
Configures Sidecar
IstioD
KITT Configures Ingress route using GSLB FQDN
Sidecar routes traffic to IGW of destination cluster
A Cluster
SR Control Plane
SRC informs K8s API Server
KubeAPI
K8s Informs IstioD
Configures IngressGatewa
IstioD
IGW
Configures Sidecar
Upstream Dependency
App
Envoy Sidecar

Microservices World Today

- **Key components**
- **Key Workflows**
- **Routing, Policies, Resiliency, Observability**
  - Let’s zoom-in Into Service Mesh routing.

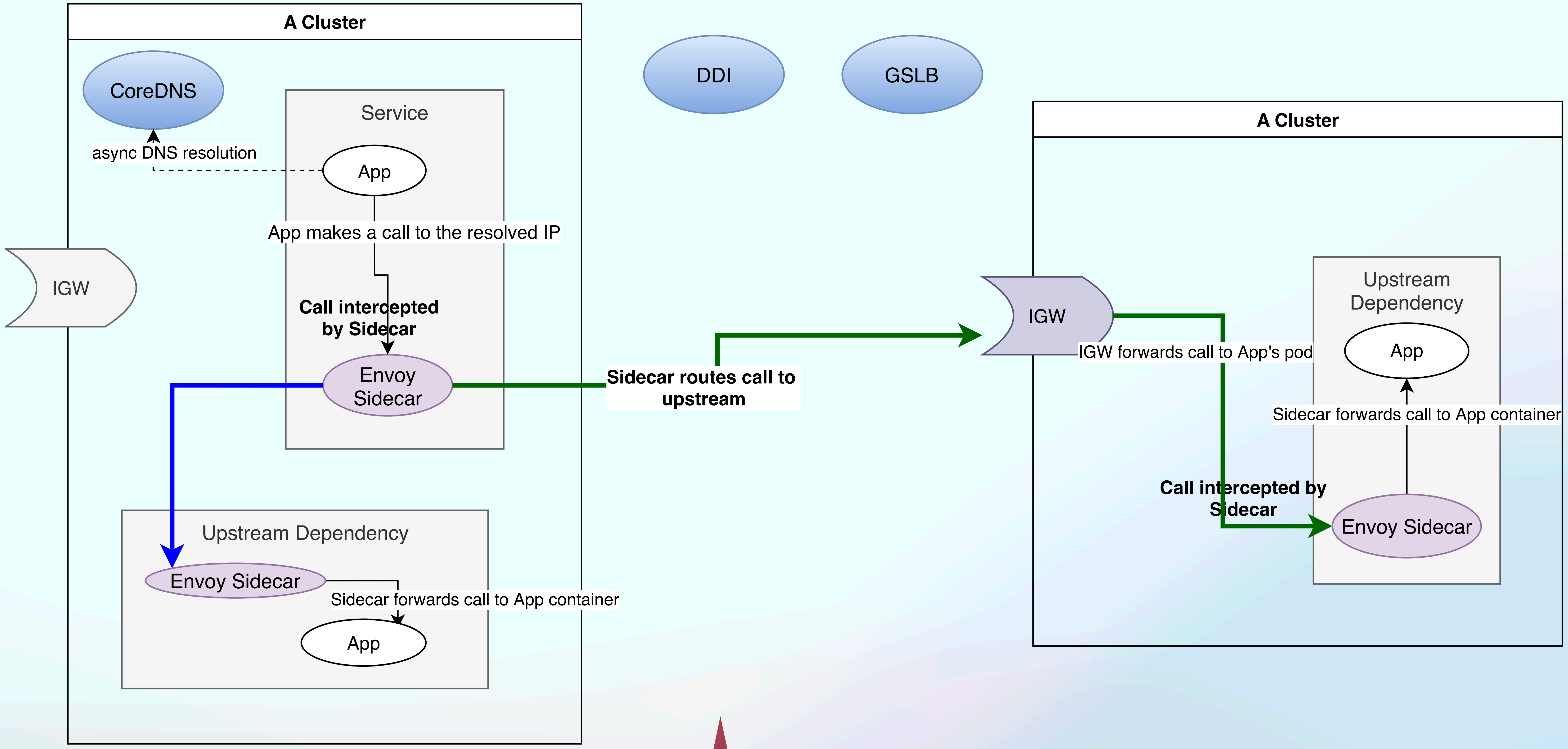

A Cluster
CoreDNS
async DNS resolution
Service
App
App makes a call to the resolved IP
Call intercepted by Sidecar
Envoy Sidecar
IGW
Upstream Dependency
Envoy Sidecar
Sidecar forward call to App container
App
Sidecar routes call to upstream
DDI
GSLB
A Cluster
IGW
IGW forwards call to App's pod
Upstream Dependency
App
Sidecar forwards call to App container
Call intercepted by Sidecar
Envoy Sidecar

Sidecar intercepts outbound traffic and reroutes it, with retries and circuit breaking

Sidecar intercepts inbound traffic and applies policies and rate limiting

Microservices World Today

- **Key components**
- **Key Workflows**
- **Routing, Policies, Resiliency, Observability**
  - Let's zoom-in Into Service Mesh routing.
  - The hops penalty question
  - Before that, let's look at the future of Service Mesh



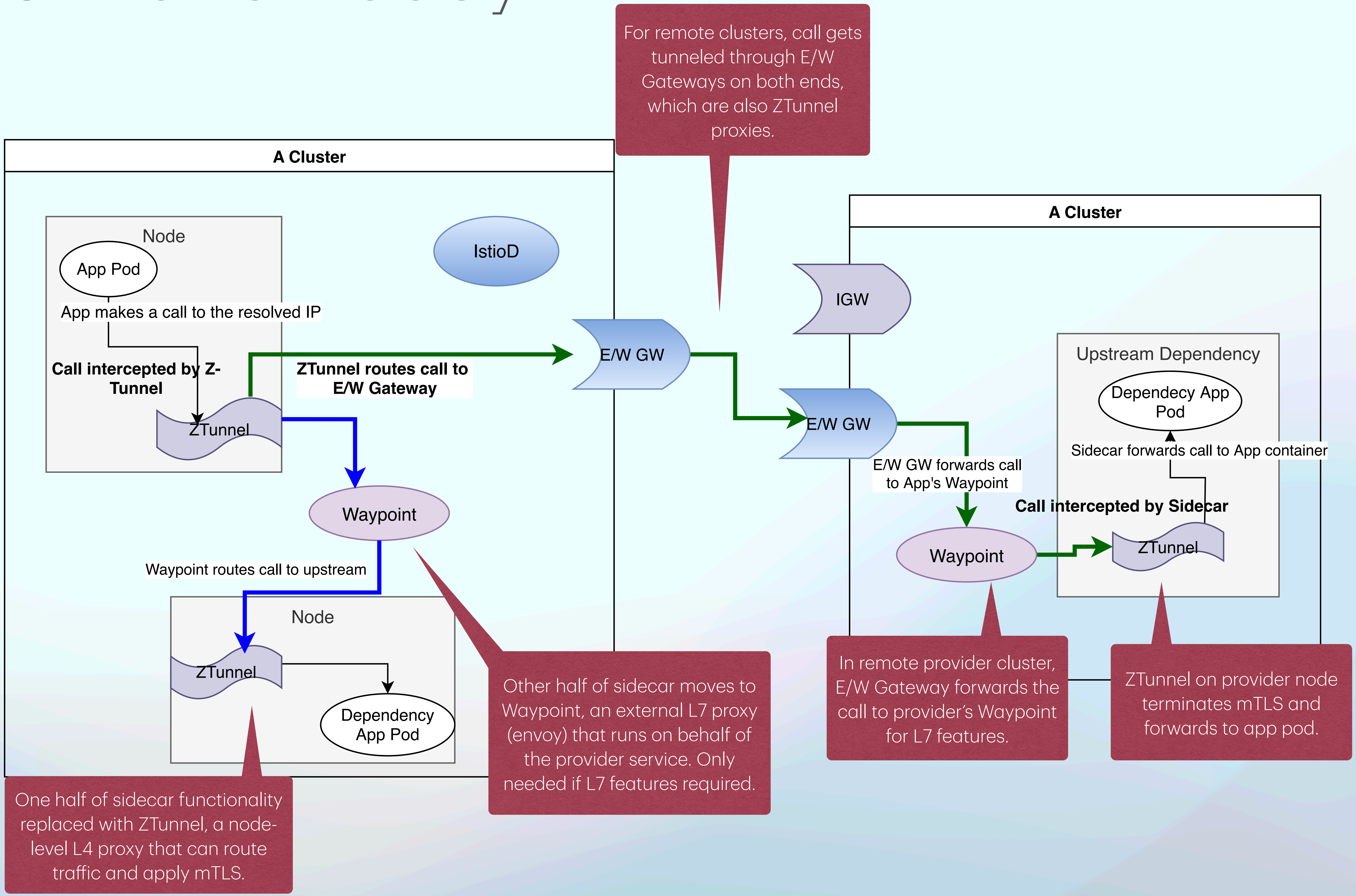
A Cluster
CoreDNS
Service
async DNS resolution
App
App makes a call to the resolved IP
IGW
Call intercepted by Sidecar
Envoy Sidecar
Upstream Dependency
Envoy Sidecar
Sidecar forwards call to App container
App
DDI
GSLB
Sidecar routes call to upstream
A Cluster
IGW
IGW forwards call to App's pod
Upstream Dependency
App
Sidecar forwards call to App container
Call intercepted by Sidecar
Envoy Sidecar

How many hops are too many?

Microservices World Today

Ambient Mesh

- The logical evolution of Service Mesh.
- Why do we care? It helps lower the operational cost by cutting down the number of sidecar cores/memory needed (likely 1/10th).



A Cluster
Node
App Pod
App makes a call to the resolved IP
Call intercepted by Z-Tunnel
ZTunnel
ZTunnel routes call to E/W Gateway
IstioD
E/W GW
Waypoint
Waypoint routes call to upstream
Node
ZTunnel
Dependency App Pod
A Cluster
IGW
E/W GW
E/W GW forwards call to App's Waypoint
Waypoint
Upstream Dependency
Dependecy App Pod
Sidecar forwards call to App container
Call intercepted by Sidecar
ZTunnel

For remote clusters, call gets tunneled through E/W Gateways on both ends, which are also ZTunnel proxies.

One half of sidecar functionality replaced with ZTunnel, a node-level L4 proxy that can route traffic and apply mTLS.

Other half of sidecar moves to Waypoint, an external L7 proxy (envoy) that runs on behalf of the provider service. Only needed if L7 features required.

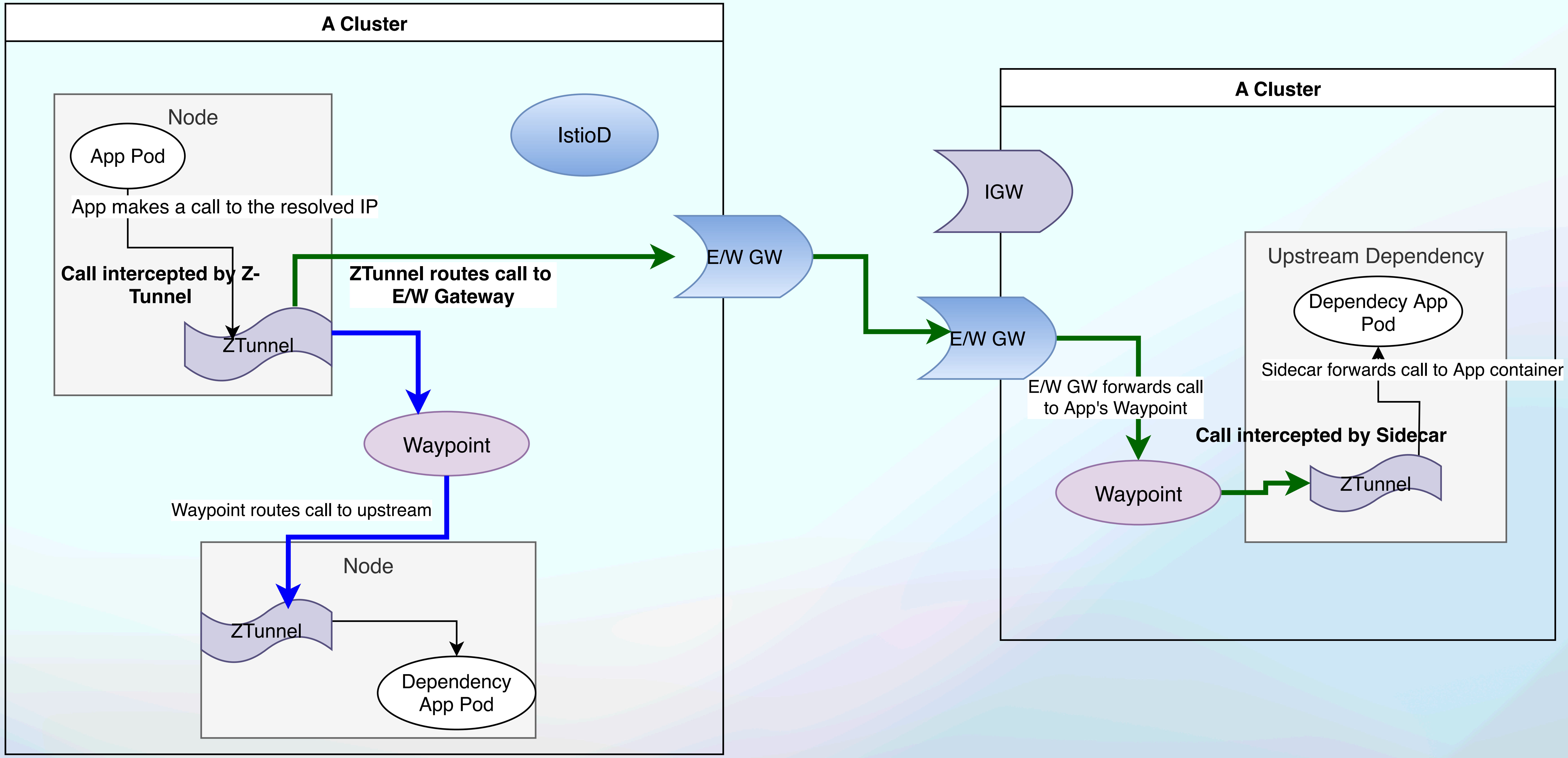
In remote provider cluster, E/W Gateway forwards the call to provider's Waypoint for L7 features.

ZTunnel on provider node terminates mTLS and forwards to app pod.

Microservices World Today

Ambient Mesh

- The logical evolution of Service Mesh.
- Why do we care? It helps lower the operational cost by cutting down the number of sidecar cores/memory needed (likely 1/10th).
- But the hop count increased!
- The hops equation is tricky. Let's visit it.
- And then we'll see how Ambient plays a role in the future AI world.

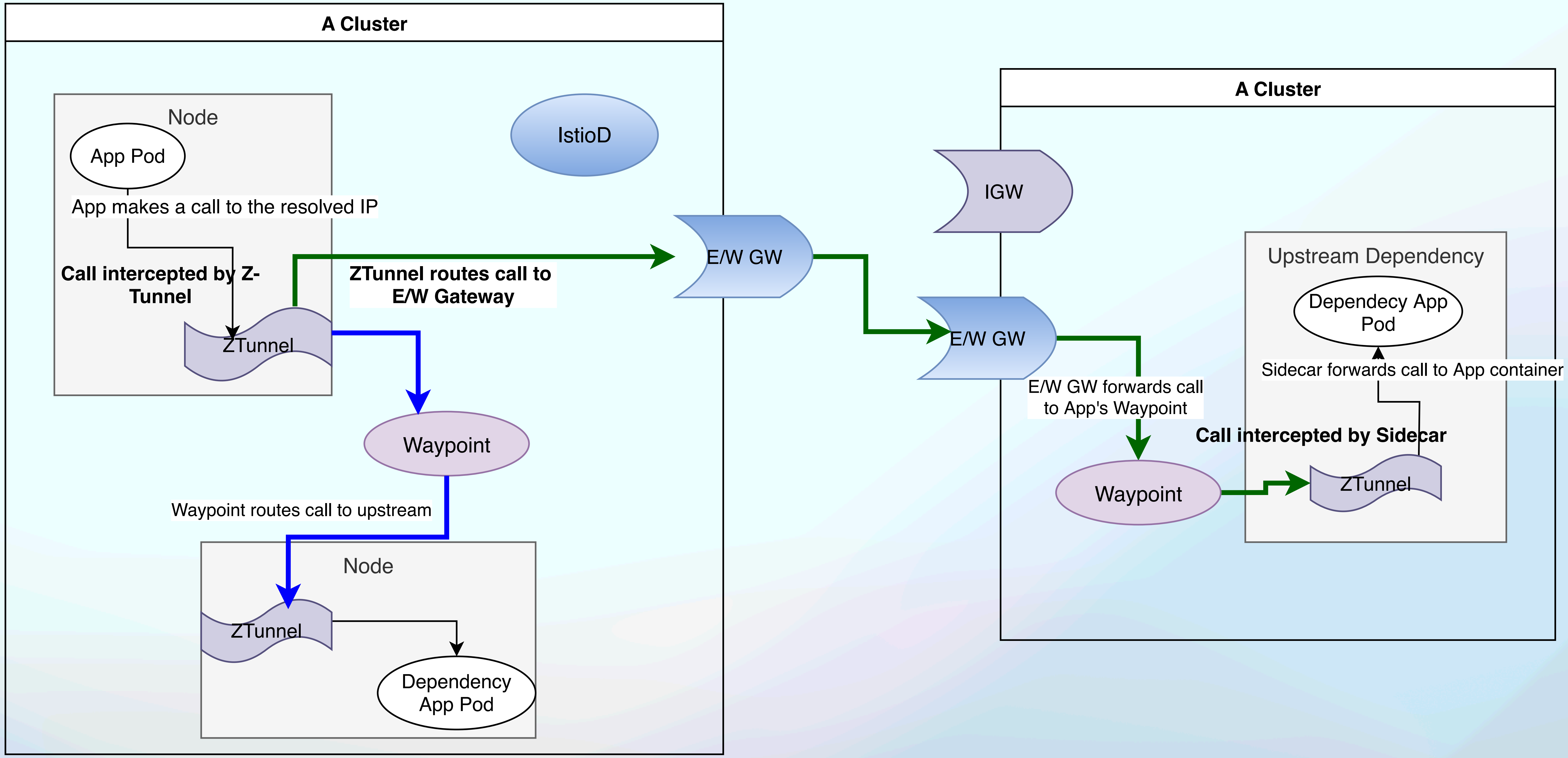

A Cluster
Node
App Pod
App makes a call to the resolved IP
Call intercepted by Z-Tunnel
ZTunnel
ZTunnel routes call to E/W Gateway
IstioD
E/W GW
Waypoint
Waypoint routes call to upstream
Node
ZTunnel
Dependency App Pod
A Cluster
IGW
E/W GW
Upstream Dependency
Dependecy App Pod
Sidecar forwards call to App container
E/W GW forwards call to App's Waypoint
Call intercepted by Sidecar
Waypoint
ZTunnel

Microservices World Today

The Hops Equation

- Network call is much cheaper than L7 proxy processing overhead.
- Assume net cost=**1**, L7 cost=**2**
- Local Cluster Traffic:
  - Sidecar Local: (1x net) + (2x L7) = (1 + 4) = **5**
  - Ambient Local: (2x net) + (1x L7) = (2 + 2) = **4**
- Remote Cluster Traffic:
  - Sidecar Remote: (2x net) + (3x L7) = (2+6) = **8**
  - Ambient Remote: (4x net + 1x L7) = (4+2) = **6**
- The higher the L7 processing cost goes, the lesser the relative impact of network hops.

- We're almost ready to look at the AI space now...


A Cluster
Node
App Pod
App makes a call to the resolved IP
Call intercepted by Z-Tunnel
ZTunnel
IstioD
ZTunnel routes call to E/W Gateway
E/W GW
Waypoint
Waypoint routes call to upstream
Node
ZTunnel
Dependency App Pod
A Cluster
IGW
E/W GW
E/W GW forwards call to App's Waypoint
Waypoint
Upstream Dependency
Dependecy App Pod
Sidecar forwards call to App container
Call intercepted by Sidecar
ZTunnel

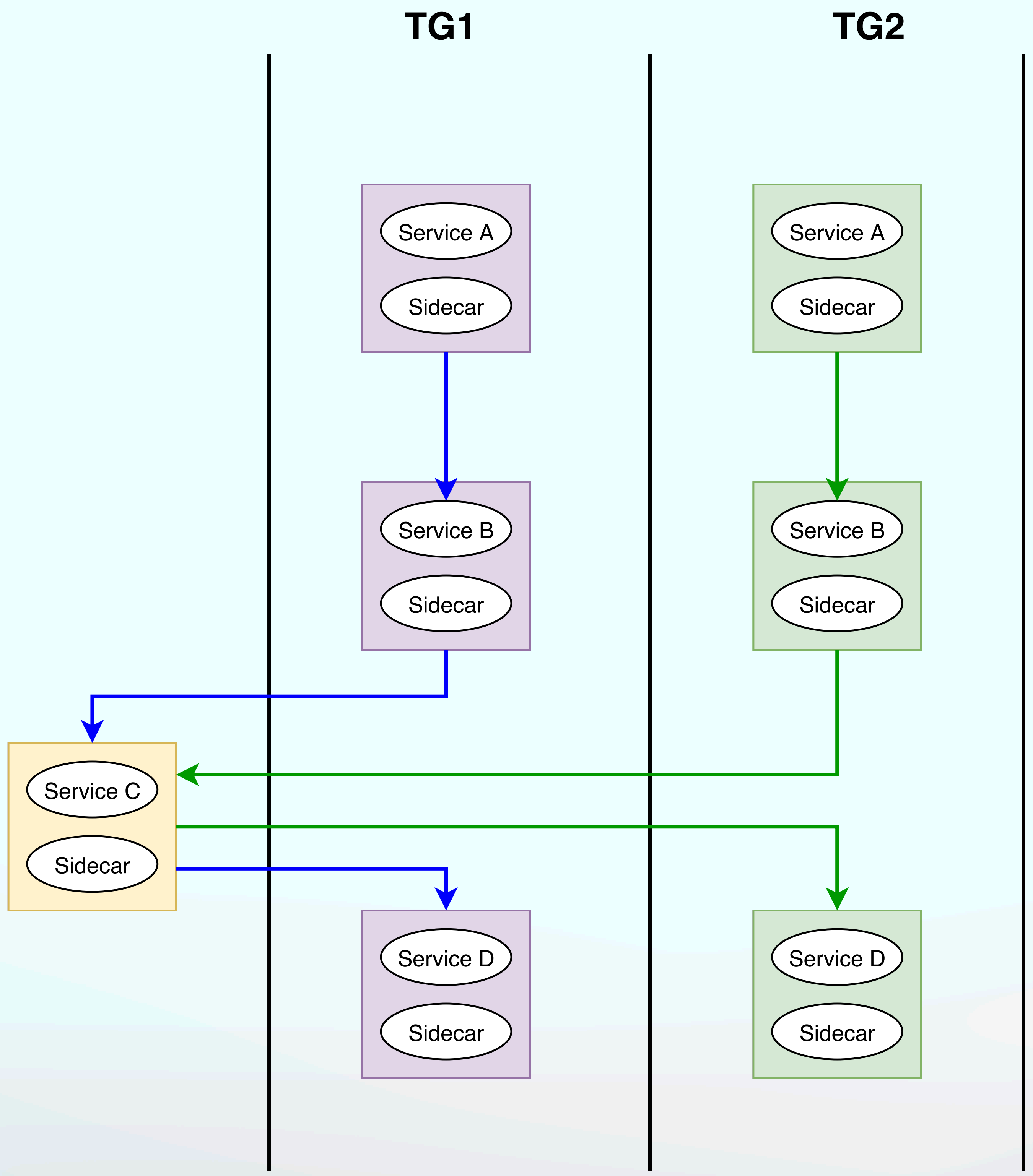
Microservices World Today

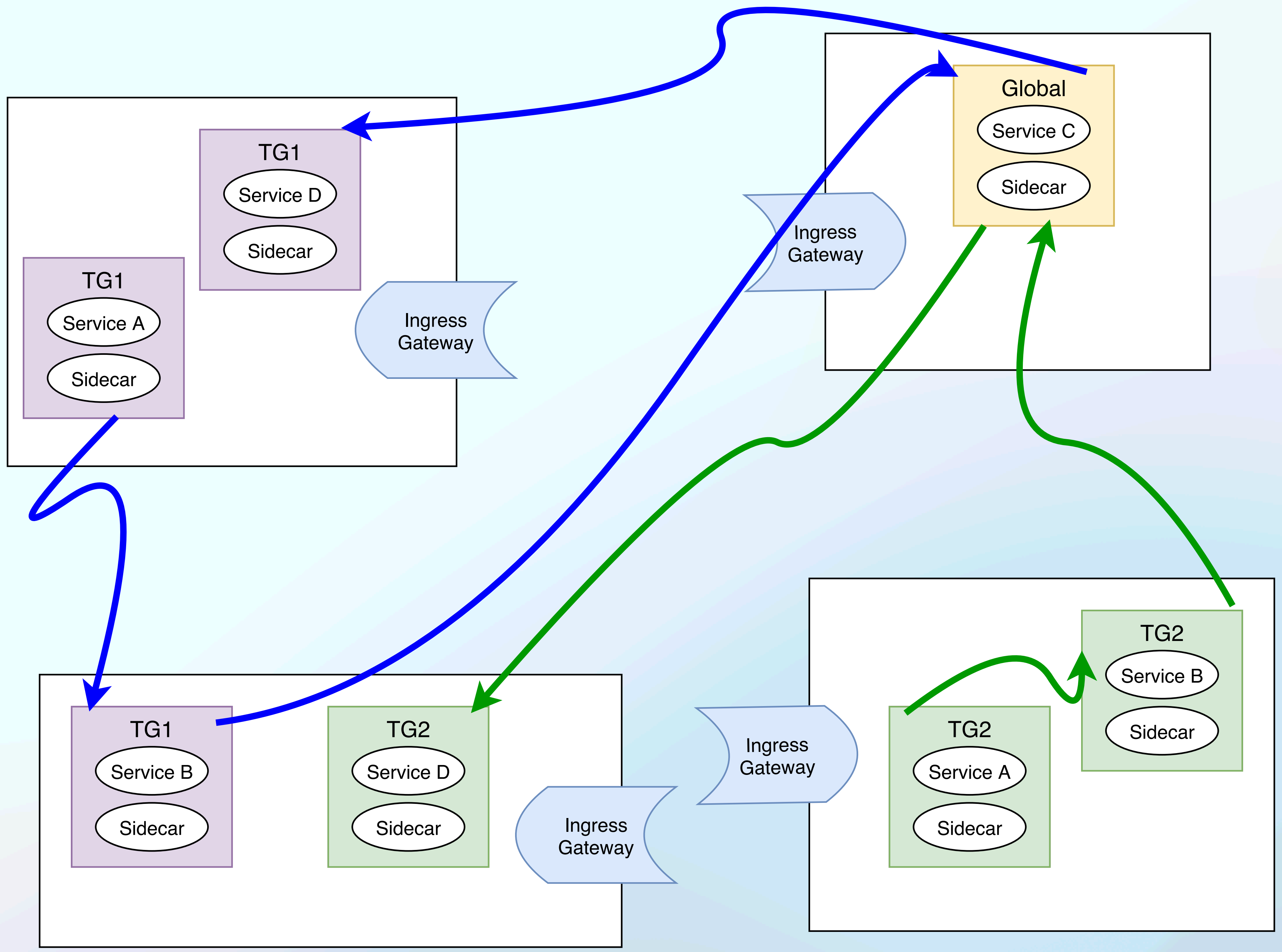
Advanced Use Cases

- WCP
- API Gateway, Bubble

Opportunities

- **Proximity based routing**
  - Solved at GSLB via regional proximity. Service Mesh has a “WCNP Proximity” feature too.
  - Still, we could do more: capacity-aware, AZ-aware, load-aware, weight-based, and other advanced routing
- **API-based routing**
  - Mesh supports URI and header based routing, but we don’t support a formal notion of APIs. If we did, we could make mesh aware of API-based capacity, weights, topology. proximity/colocation, availability, failovers.
  - We’ll be circling back to this one in the AI world.



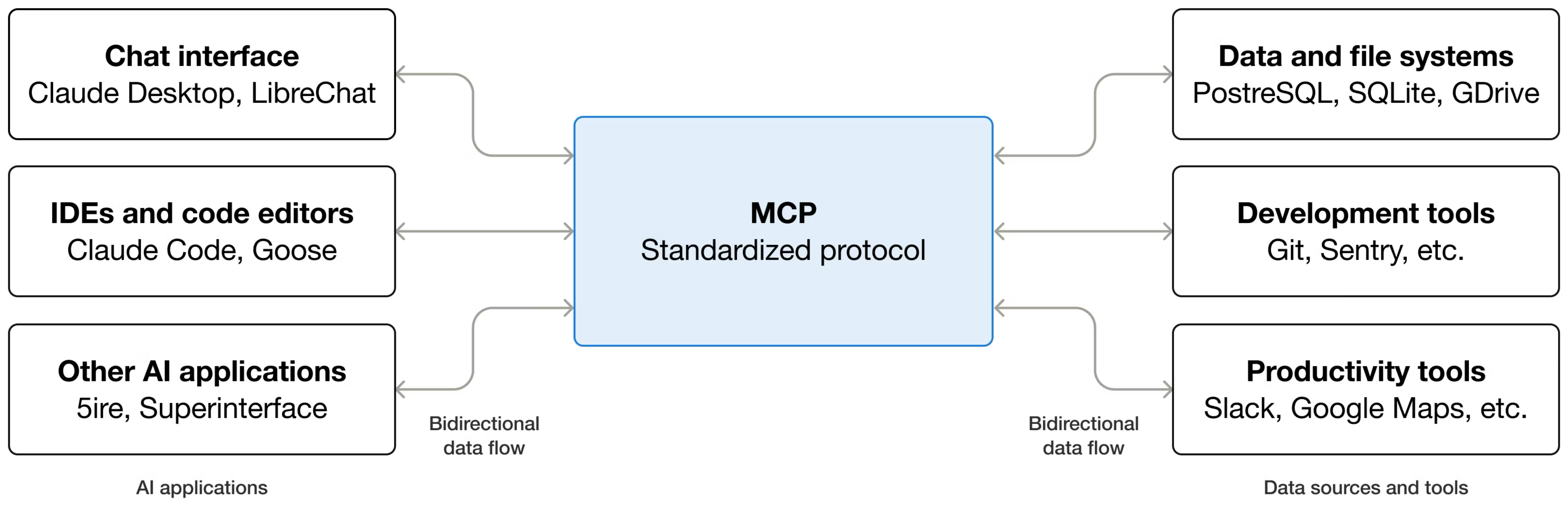


AI World

The Journey

- LLMs
- LLM Client Apps (e.g. ChatGPT, CoPilot)
- LLM Apps in enterprises: Need for integration with enterprise data
- **Enter MCP**
- The need for smarter clients: Agents
- The need for agents to converse
- **Enter A2A**
- LLM <- Agent <-> [A2A] <-> Agent -> [MCP] -> Services

MCP



Chat interface
Claude Desktop, LibreChat

IDEs and code editors
Claude Code, Goose

Other AI applications
5ire, Superinterface

AI applications

Bidirectional data flow

MCP
Standardized protocol

Bidirectional data flow

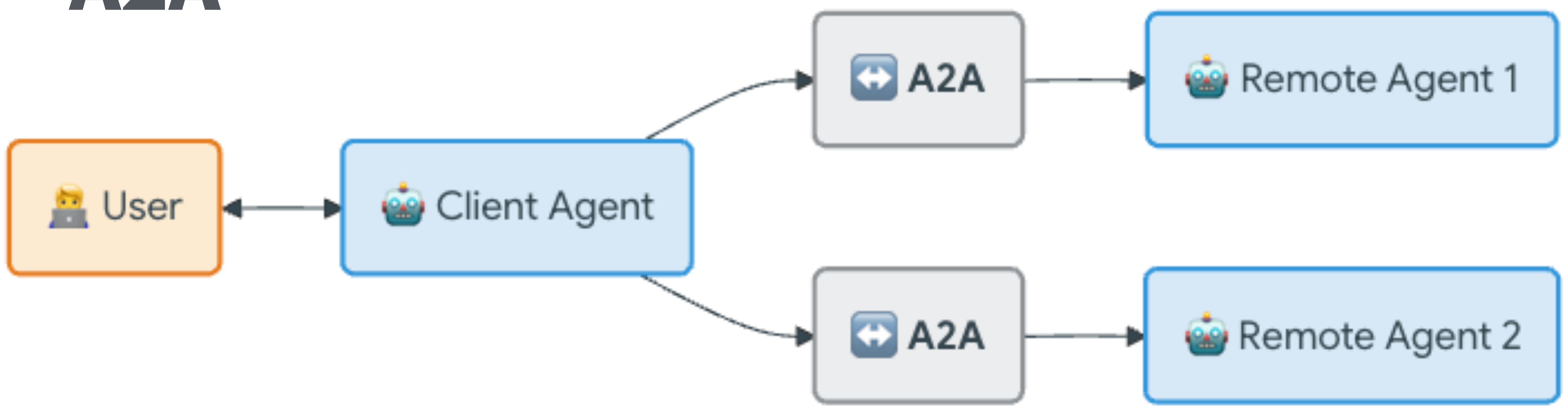
Data and file systems
PostreSQL, SQLite, GDrive

Development tools
Git, Sentry, etc.

Productivity tools
Slack, Google Maps, etc.

Data sources and tools

A2A



User

Client Agent

A2A

Remote Agent 1

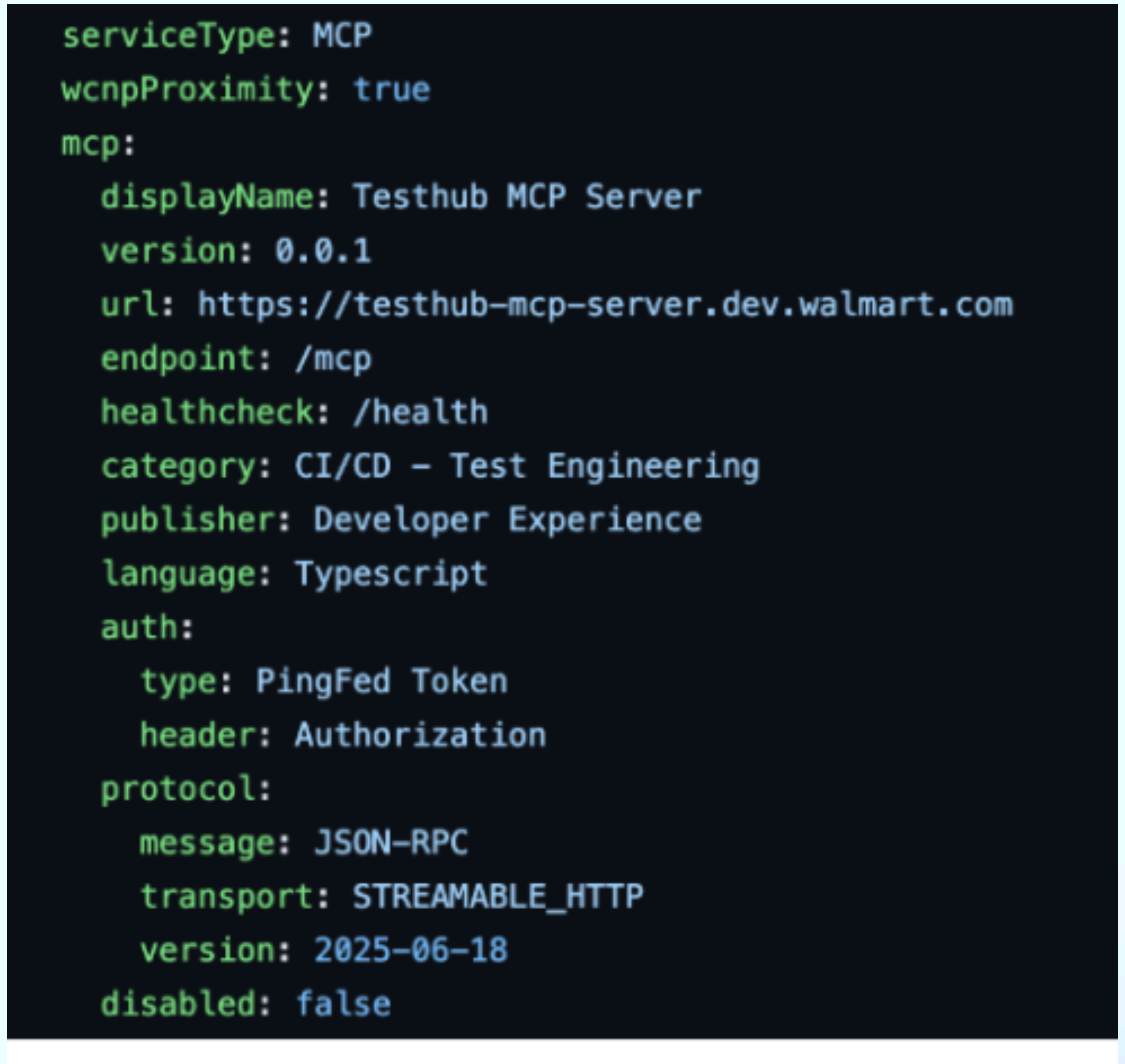
A2A

Remote Agent 2

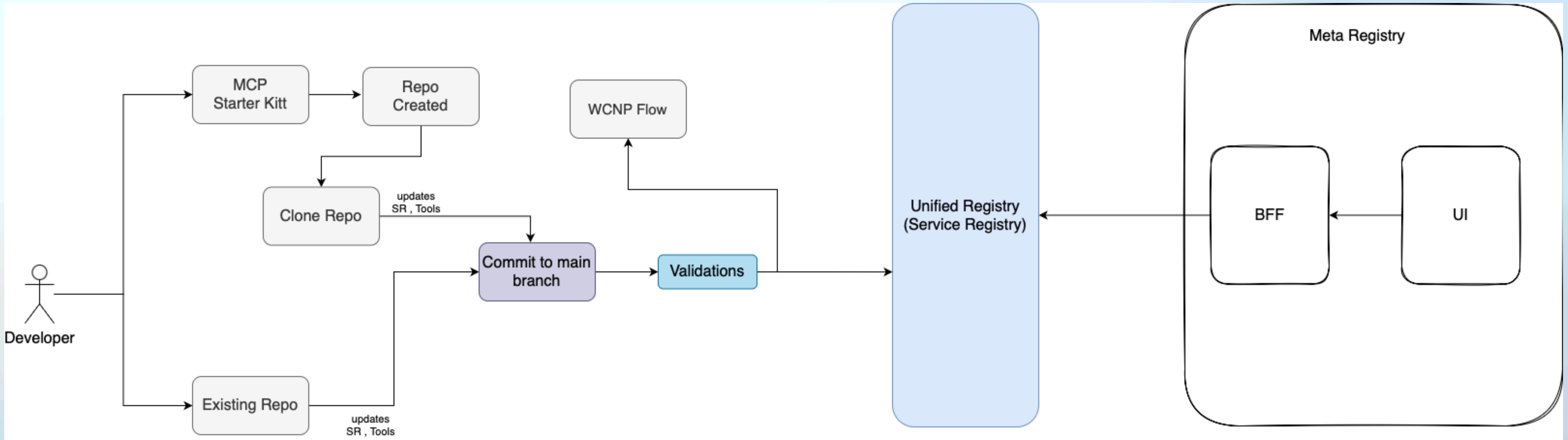
AI World

MCP Workflow

- MCP servers will mostly be the existing microservices in new clothing.
  - Some may be deployed as external servers that access the micro service via API to deliver tool functionality.
  - Others may be built embedded within the microservice as a new API/URI/Port.
- As such, it only follows that the MCP servers are registered into SR as microservices using the existing workflow.
- Gitops will register the server details in SR under a newly introduced "MCP" capability.
- Service Mesh will pick up the SR configs and enable routing/policies/resiliency/ metrics for the MCP servers automatically.



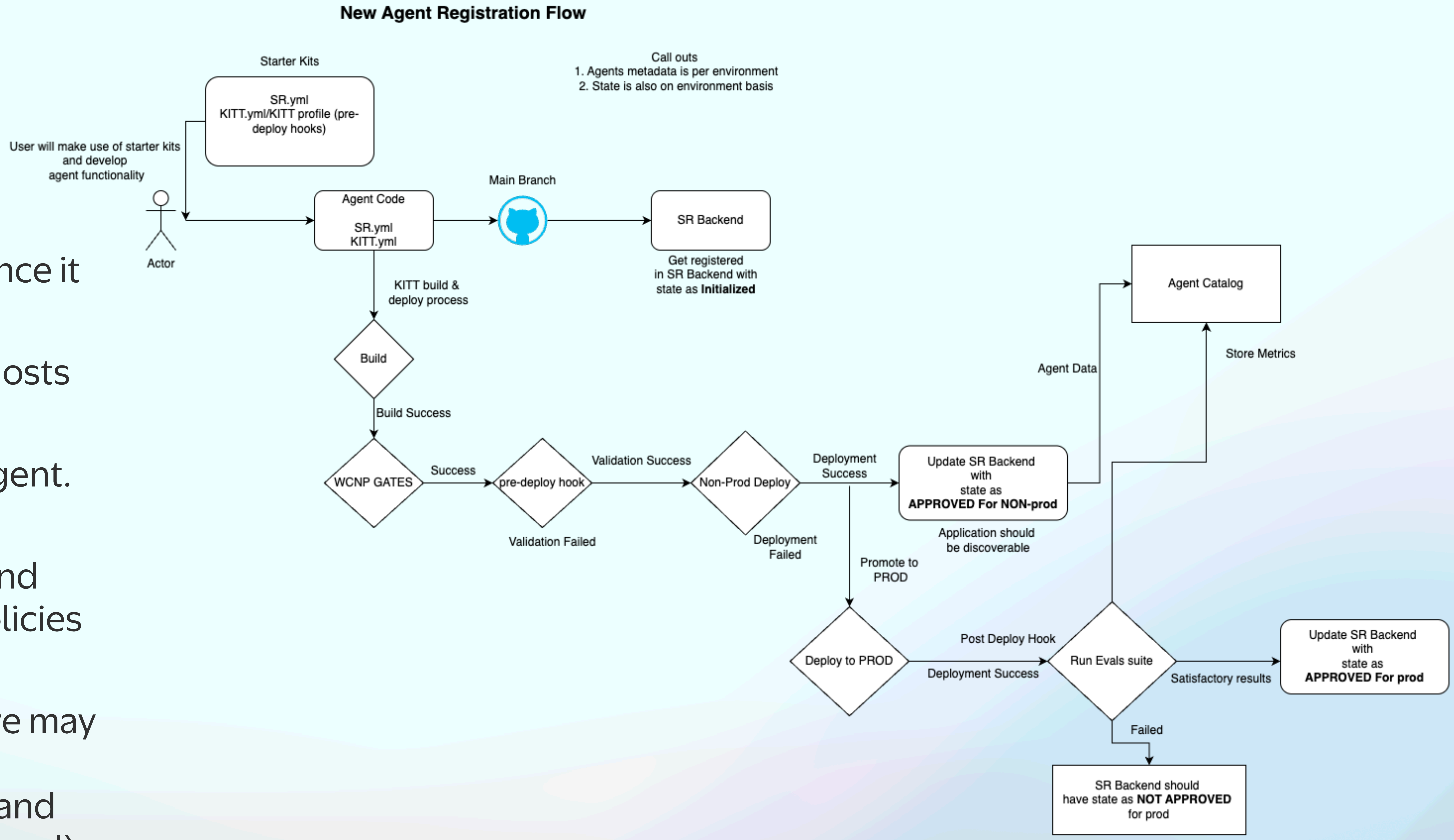
```
serviceType: MCP
wcnpProximity: true
mcp:
  displayName: Testhub MCP Server
  version: 0.0.1
  url: https://testhub-mcp-server.dev.walmart.com
  endpoint: /mcp
  healthcheck: /health
  category: CI/CD - Test Engineering
  publisher: Developer Experience
  language: Typescript
  auth:
    type: PingFed Token
    header: Authorization
  protocol:
    message: JSON-RPC
    transport: STREAMABLE_HTTP
    version: 2025-06-18
  disabled: false
```



AI World

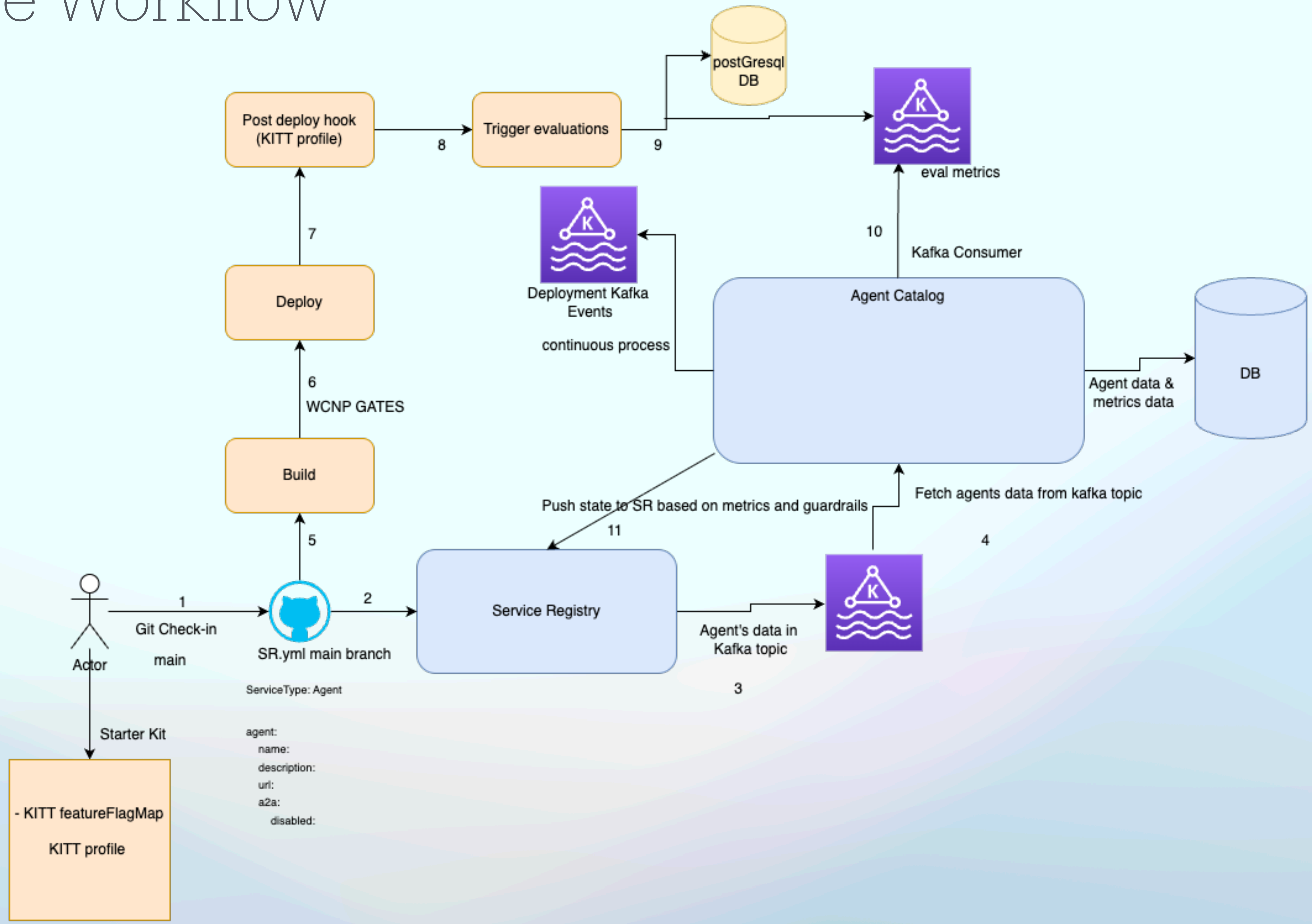
Agents Workflow

- Agent definition is more nuanced since it may or may not be a "service".
- A2A has a notion of a "Server" that hosts the agents.
- A server may not be 1-to-1 with an agent.
- It's the container server that will be registered in SR as a microservice, and service mesh will enable routing/policies for it.
- Since the space is still evolving, there may be additional needs in the future to perform more fine-grained routing and policy enforcement (at tools/tasks level).


New Agent Registration Flow
Starter Kits
SR.yml
KITT.yml/KITT profile (pre-deploy hooks)
Call outs
1. Agents metadata is per environment
2. State is also on environment basis
User will make use of starter kits and develop agent functionality
Actor
Agent Code
SR.yml
KITT.yml
Main Branch
SR Backend
Get registered in SR Backend with state as Initialized
KITT build & deploy process
Build
Build Success
WCNP GATES
Success
pre-deploy hook
Validation Success
Validation Failed
Non-Prod Deploy
Deployment Success
Deployment Failed
Update SR Backend with state as APPROVED For NON-prod
Application should be discoverable
Agent Catalog
Agent Data
Store Metrics
Promote to PROD
Deploy to PROD
Post Deploy Hook
Deployment Success
Run Evals suite
Satisfactory results
Update SR Backend with state as APPROVED For prod
Failed
SR Backend should have state as NOT APPROVED for prod

AI World

A2A Governance Workflow



Post deploy hook (KITT profile)

8

Trigger evaluations

9

postGresql DB

eval metrics

10

Kafka Consumer

7

Deployment Kafka Events

continuous process

Deploy

Agent Catalog

Agent data & metrics data

DB

6

WCNP GATES

Build

Push state to SR based on metrics and guardrails

11

Fetch agents data from kafka topic

4

5

Actor

1

Git Check-in

main

2

SR.yml main branch

Service Registry

Agent's data in Kafka topic

3

```
ServiceType: Agent


agent:
  name:
  description:
  url:
  a2a:
    disabled:
```

Starter Kit

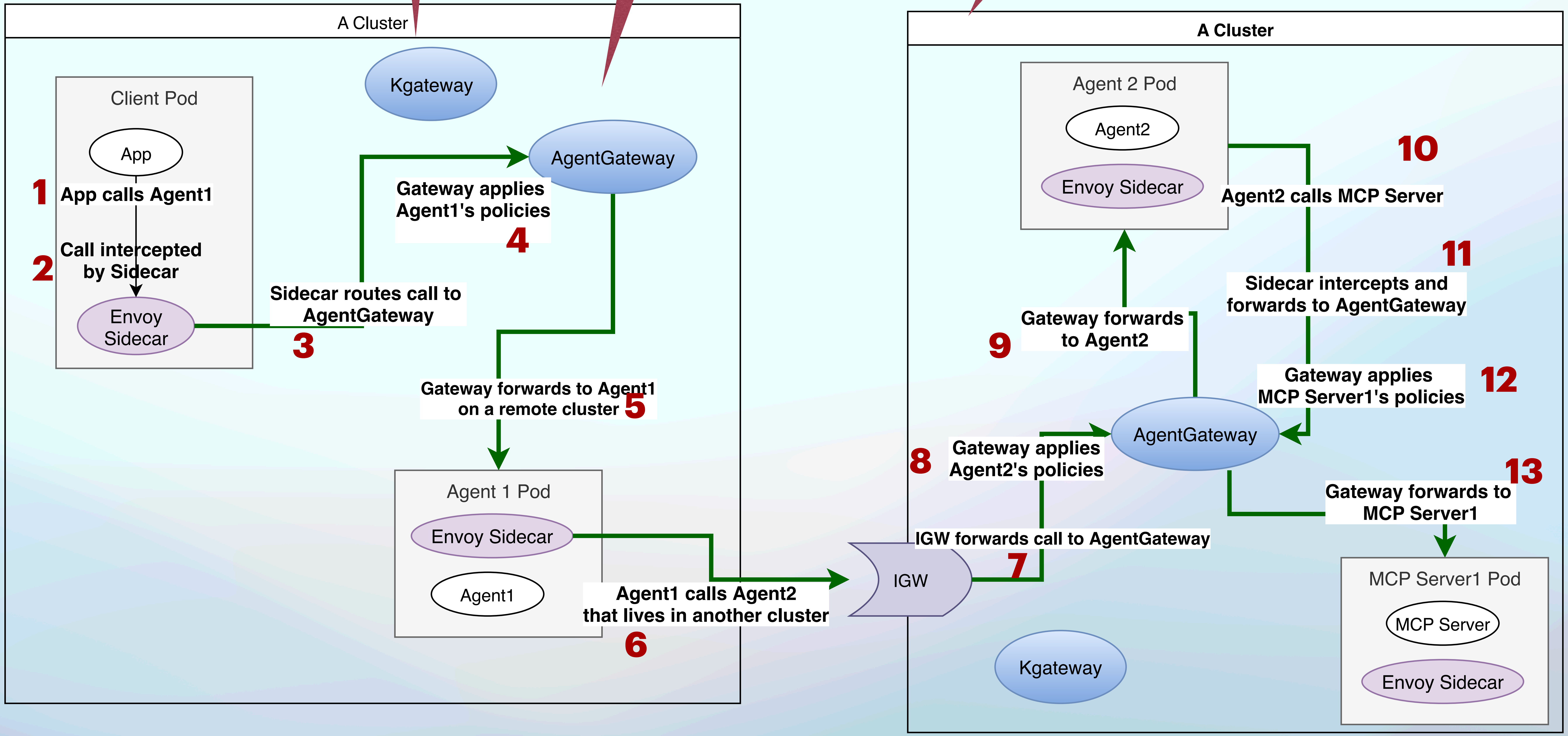
- KITT featureFlagMap

KITT profile

AI World

MCP and A2A Routing

- Fits into current service mesh routing
- Let's see this in action in the Demo!



A Cluster
Kgateway
AgentGateway
Client Pod
App
1 App calls Agent1
2 Call intercepted by Sidecar
Envoy Sidecar
Sidecar routes call to AgentGateway
3
Gateway applies Agent1's policies
4
Gateway forwards to Agent1 on a remote cluster
5
Agent 1 Pod
Envoy Sidecar
Agent1
Agent1 calls Agent2 that lives in another cluster
6
IGW
IGW forwards call to AgentGateway
7
A Cluster
Agent 2 Pod
Agent2
Envoy Sidecar
8 Gateway applies Agent2's policies
9 Gateway forwards to Agent2
10
Agent2 calls MCP Server
11
Sidecar intercepts and forwards to AgentGateway
Gateway applies MCP Server1's policies
12
AgentGateway
13
Gateway forwards to MCP Server1
MCP Server1 Pod
MCP Server
Envoy Sidecar
Kgateway

Kgateway is the controller that configures AgentGateway

AgentGateway gets deployed as Waypoint in Ambient mesh.

Policies enforced by AgentGateway for Agents and MCP Servers

AI World

MCP and A2A Routing

- Demo
- QnA