



ORACLE

Siebel CRM Cloud Native Architecture

Chandan DasGupta

Senior Development Director

Azahar Uddin

Senior Principal Software Engineer

September, 2020



Safe harbor statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, timing, and pricing of any features or functionality described for Oracle's products may change and remains at the sole discretion of Oracle Corporation.



Cloud Native Siebel

- 1 What is Cloud Native?
- 2 Why should you care?
- 3 Siebel Cloud Native Implementation
- 4 What about Microservices?
- 5 Siebel Cloud Native Architecture vis-à-vis Siebel Cloud Enablement (IP17+)
- 6 Migration Effort
- 7 What's next?



What is Cloud Native?

Cloud Native Siebel CRM Unleashes transformational opportunities

top-line

bottom-line

increase agility and speed for innovation



react to market needs



lower cost of innovation

increase revenue with lower cost of downtime



improve customer sat



lower downtime cost

reduce platform/skills risk and cost of IT



highly adaptable infra



reduce OpEx costs

modern platform to embrace change



simplicity of change



no greenfield CRM

Cloud Native Definition

CNCF (Cloud Native Computing Foundation) Cloud Native Definition v1.0 -

<https://github.com/cncf/toc/blob/master/DEFINITION.md>

Cloud native technologies empower organizations to build and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds. Containers, service meshes, microservices, immutable infrastructure, and declarative APIs exemplify this approach.

These techniques enable loosely coupled systems that are resilient, manageable, and observable. Combined with robust automation, they allow engineers to make high-impact changes frequently and predictably with minimal toil.

Cloud Native Principles - <https://cloud-native-principles.org>

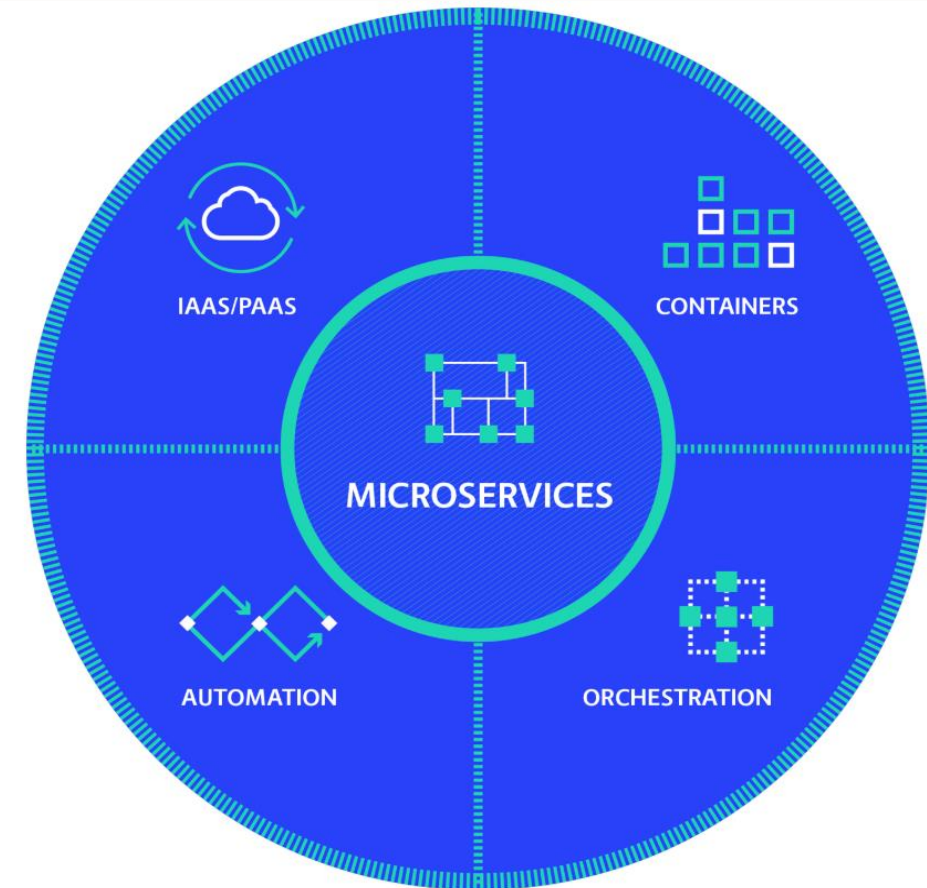
Cloud Native Principles - Beyond The Twelve-Factor App -

<https://tanzu.vmware.com/content/ebooks/beyond-the-12-factor-app>

“A cloud-native application is an application that has been designed and implemented to run on a Platform-as-a-Service installation and to embrace horizontal elastic scaling.”

Cloud Native Primer - <https://docs.microsoft.com/en-us/dotnet/architecture/cloud-native/definition>

Excellent primer on Cloud Native and Microservices



CLOUD NATIVE TRAIL MAP

The Cloud Native Landscape [Landscape](https://landscape.cncf.io) has a large number of options. This Cloud Native Trail Map is a recommended process for leveraging open source, cloud native technologies. At each step, you can choose a vendor-supported offering or do it yourself, and everything after step #3 is optional based on your circumstances.

HELP ALONG THE WAY

A. Training and Certification

Consider training offerings from CNCF and then take the exam to become a Certified Kubernetes Administrator or a Certified Kubernetes Application Developer
cncf.io/training

B. Consulting Help

If you want assistance with Kubernetes and the surrounding ecosystem, consider leveraging a Kubernetes Certified Service Provider
cncf.io/kscsp

C. Join CNCF's End User Community

For companies that don't offer cloud native services externally
cncf.io/enduser

WHAT IS CLOUD NATIVE?

Cloud native technologies empower organizations to build and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds. Containers, service meshes, microservices, immutable infrastructure, and declarative APIs exemplify this approach.

These techniques enable loosely coupled systems that are resilient, manageable, and observable. Combined with robust automation, they allow engineers to make high-impact changes frequently and predictably with minimal toil.

The Cloud Native Computing Foundation seeks to drive adoption of this paradigm by fostering and sustaining an ecosystem of open source, vendor-neutral projects. We democratize state-of-the-art patterns to make these innovations accessible for everyone.

l.cncf.io

v20200501



1. CONTAINERIZATION

- Commonly done with Docker containers
- Any size application and dependencies (even PDP-11 code running on an emulator) can be containerized
- Over time, you should aspire towards splitting suitable applications and writing future functionality as microservices



3. ORCHESTRATION & APPLICATION DEFINITION

- Kubernetes is the market-leading orchestration solution
- You should select a Certified Kubernetes Distribution, Hosted Platform, or Installer: cncf.io/ckd
- Helm Charts help you define, install, and upgrade even the most complex Kubernetes application



5. SERVICE PROXY, DISCOVERY, & MESH

- CoreDNS is a fast and flexible tool that is useful for service discovery
- Envoy and Linkerd each enable service mesh architectures
- They offer health checking, routing, and load balancing



7. DISTRIBUTED DATABASE & STORAGE

When you need more resiliency and scalability than you can get from a single database, Vitess is a good option for running MySQL at scale through sharding. Rook is a storage orchestrator that integrates a diverse set of storage solutions into Kubernetes. Serving as the "brain" of Kubernetes, etcd provides a reliable way to store data across a cluster of machines. TiKV is a high performance distributed transactional key-value store written in Rust.



9. CONTAINER REGISTRY & RUNTIME

Harbor is a registry that stores, signs, and scans content. You can use alternative container runtimes. The most common, both of which are OCI-compliant, are containerd and CRI-O.



2. CI/CD

- Setup Continuous Integration/Continuous Delivery (CI/CD) so that changes to your source code automatically result in a new container being built, tested, and deployed to staging and eventually, perhaps, to production
- Setup automated rollouts, roll backs and testing
- Argo is a set of Kubernetes-native tools for deploying and running jobs, applications, workflows, and events using GitOps paradigms such as continuous and progressive delivery and MLops



4. OBSERVABILITY & ANALYSIS

- Pick solutions for monitoring, logging and tracing
- Consider CNCF projects Prometheus for monitoring, Fluentd for logging and Jaeger for Tracing
- For tracing, look for an OpenTracing-compatible implementation like Jaeger



6. NETWORKING, POLICY, & SECURITY

To enable more flexible networking, use a CNI-compliant network project like Calico, Flannel, or Weave Net. Open Policy Agent (OPA) is a general-purpose policy engine with uses ranging from authorization and admission control to data filtering. Falco is an anomaly detection engine for cloud native.



8. STREAMING & MESSAGING

When you need higher performance than JSON-REST, consider using gRPC or NATS. gRPC is a universal RPC framework. NATS is a multi-model messaging system that includes request/reply, pub/sub and load balanced queues. CloudEvents is a specification for describing event data in common ways.



10. SOFTWARE DISTRIBUTION

If you need to do secure software distribution, evaluate Notary, an implementation of The Update Framework.



Cloud Native Operational Attributes

Immutability:	Immutable (unable to be changed) services, all changes versioned, all changes via automated pipeline
Automation:	Everything fully automated, no human touch, automated deployment, automated operations
Disposability:	Stateless Services, Load Balancing, Fault Tolerance, Scalability, Repaving tolerance
External Config:	No internal hard coded configuration of services, configuration of services via Kubernetes descriptors
Logged Events:	Centralized logging for audit, training and debugging
Telemetry:	Constant logging and collection of key performance indicator metrics (KPIs) both environmental (CPU, Memory etc.) and functional (order throughput, UX shopping card abandons etc.).
Operations:	Automated, Controlled, No Human access to nodes or services, Agile
Service Oriented:	No monolith, services can deploy, scale and upgrade independently.

Siebel Cloud Native Development Epics

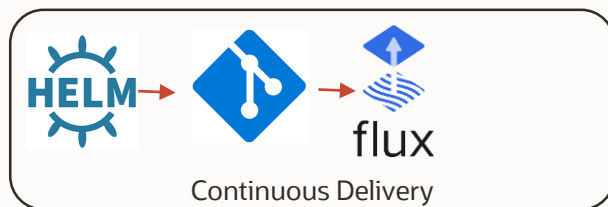
Epic		Scope
Siebel Cloud Native Orchestration CNCF: Containerization CNCF: Orchestration & Application Definition	<ul style="list-style-type: none"> Deploys Components as Kubernetes Services with each MTServe as a Pod Scales each Service automatically based on workload Load Balances Pods per Service - for both external and inter-component requests Gateway and present Siebel Server infrastructure no longer play any role in orchestration of Components 	General*
Siebel Zero Downtime - Rolling/Canary Upgrades	During upgrades, each Service should be uninterrupted while Pods for a given Services are gradually rolled over from one version to another.	General*
Siebel Event Pub Sub Framework CNCF: Streaming & Messaging	New infrastructure for Event Driven communication between Siebel Components and external event driven systems. Adapter based architecture that can be extended to other messaging products, presently working with Kafka.	General*
Siebel CI/CD CNCF: CI/CD	Improvements that facilitate fully automated CI and CD pipelines for Oracle internal development. Benefits will pass on to customers for their pipelines as well.	General*
Siebel Cloud Native Logging-Monitoring-Tracing ** CNCF: Observability & Analysis	Improvements to facilitate Cloud Native Logging-Monitoring-Tracing using products such as fluentd, Prometheus, Jaeger. Benefits will pass on to customers to operate their deployments as well.	General*
Siebel Cloud Native Security CNCF: Networking, Policy & Security	Improvements to facilitate Siebel security features aligning to Cloud Native topologies.	General*
Per Application Component Containerization Completeness	Ensure "Siebel Cloud Native Orchestration" works for each Industry / Application	Specific
Per Application API Completeness	REST API completeness of individual Applications / Industries.	Specific
Siebel Gateway Removal	"Siebel Cloud Native Orchestration" removes the Gateway from the topology orchestration - it only servers to stored system configuration. This epic will remove the dependency of all Components on the Gateway for their configuration.	General*
Siebel PaaS Independence	Provide mechanisms to adapt usage of Siebel with different PaaS services, e.g. adapter based mechanism for different caching services.	General*
Siebel Industry Services CI-CD	Application / Industry Component specific improvements for corresponding to "Siebel CI/CD"	Specific

*General – Not Business/Application Domain specific




** Apart from Application Domain Specific Event Monitoring

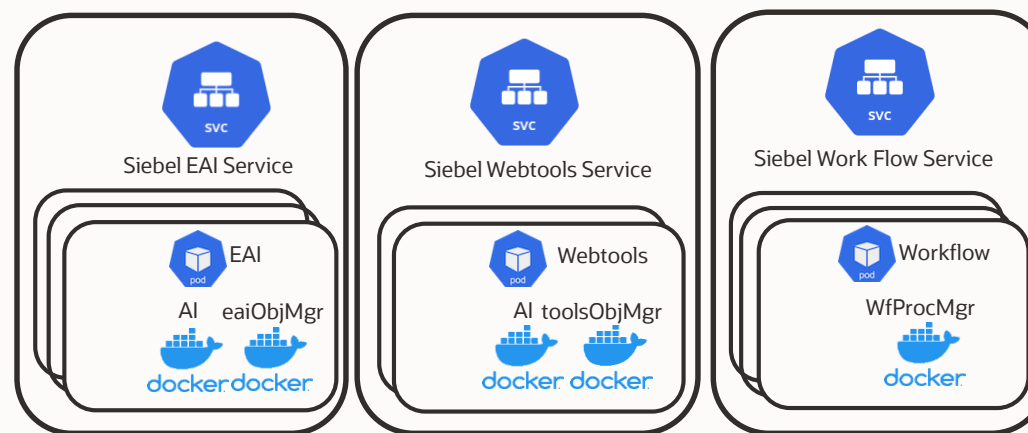
Cloud Native Infrastructure built around independent Siebel Components

CNCF Criteria – Containerization, Orchestration & Application Definition



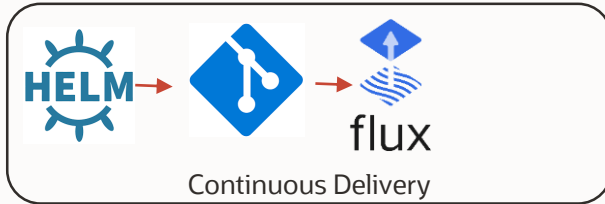
CNCF Criteria

- Containerization: 
- Orchestration & Application Definition:  



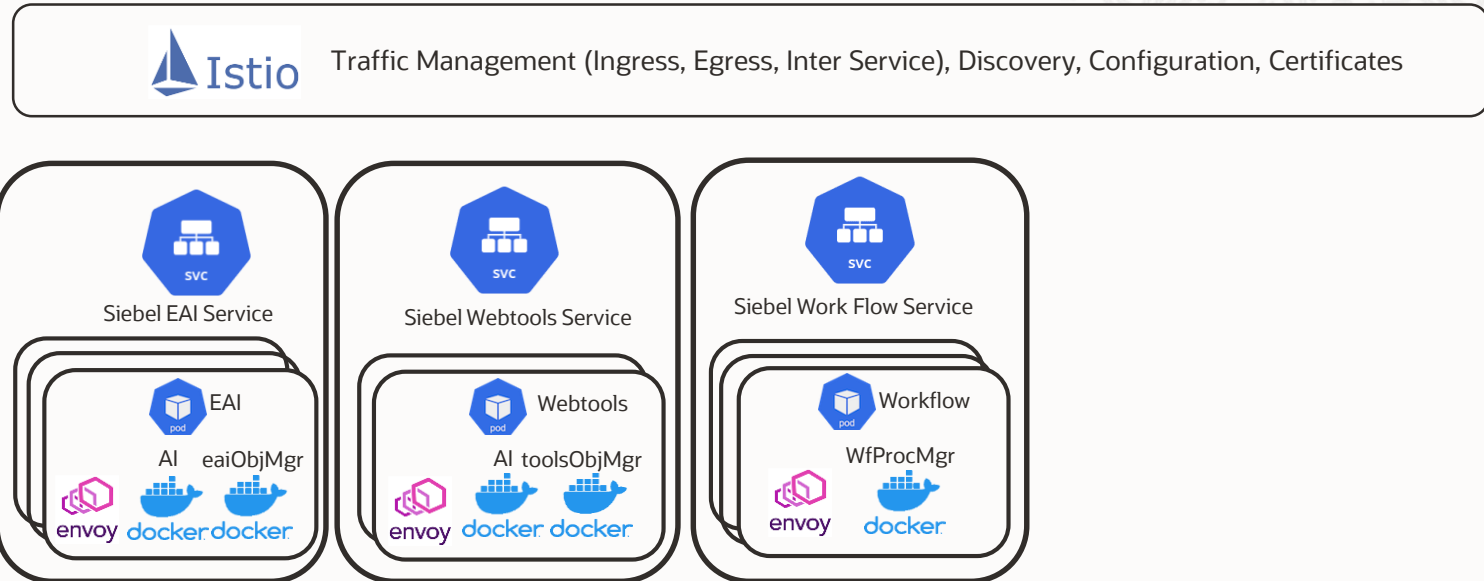
Cloud Native Infrastructure built around independent Siebel Components

CNCF Criteria – Service Proxy, Discovery & Mesh



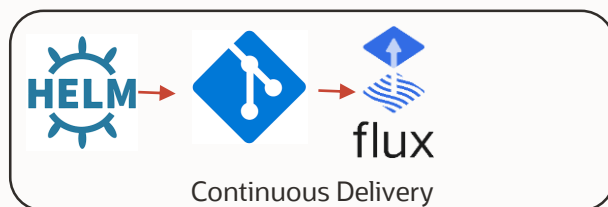
CNCF Criteria

- Containerization:
- Orchestration & Application Definition:
- Service Proxy, Discovery & Mesh:



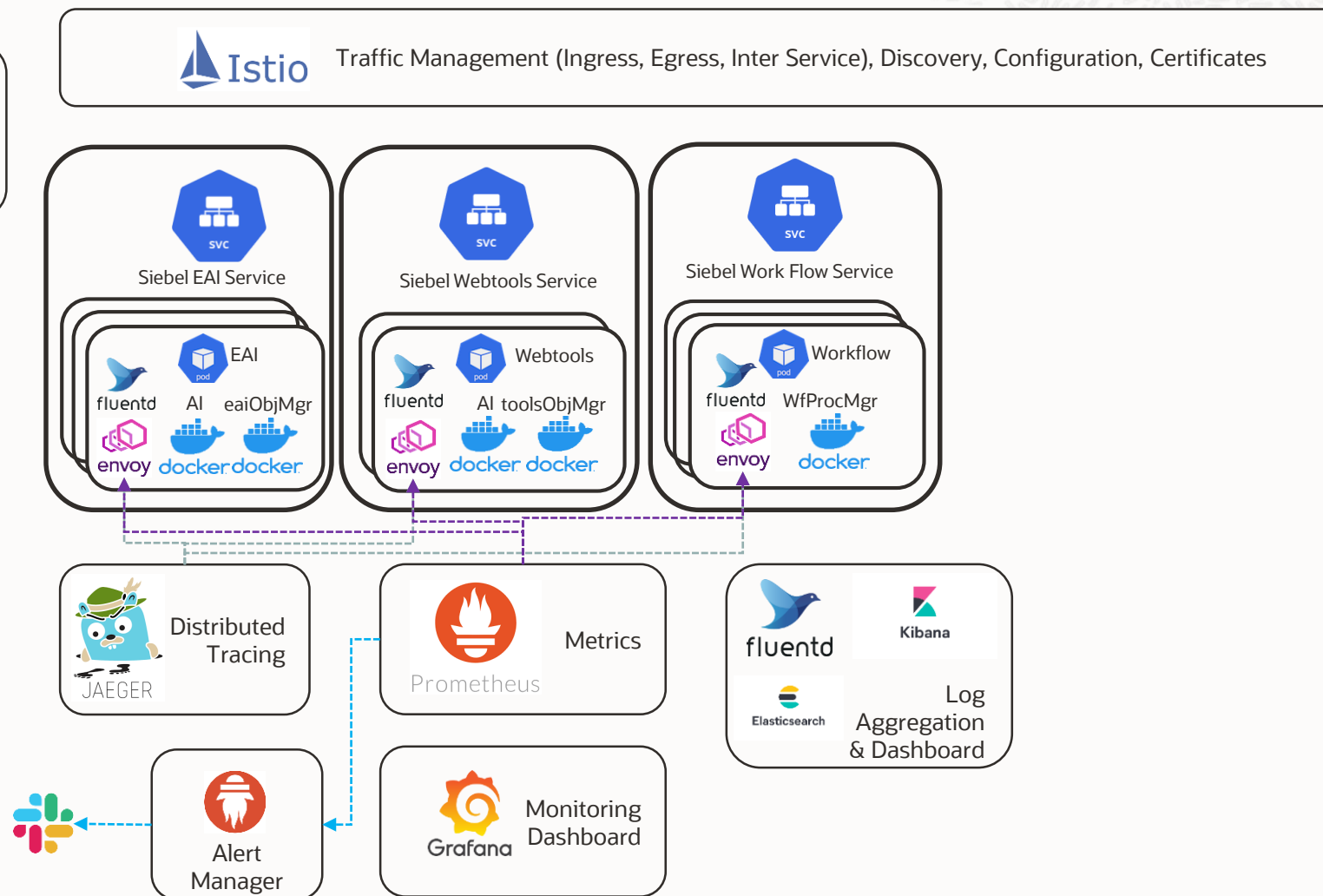
Cloud Native Infrastructure built around independent Siebel Components

CNCF Criteria – Observability & Analysis



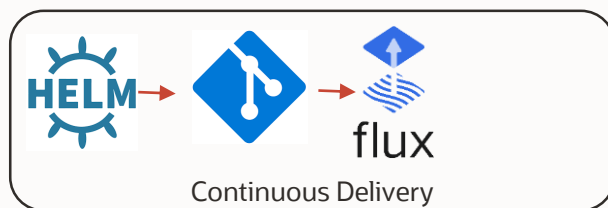
CNCF Criteria

- Containerization:
- Orchestration & Application Definition:
- Service Proxy, Discovery & Mesh:
- Observability & Analysis



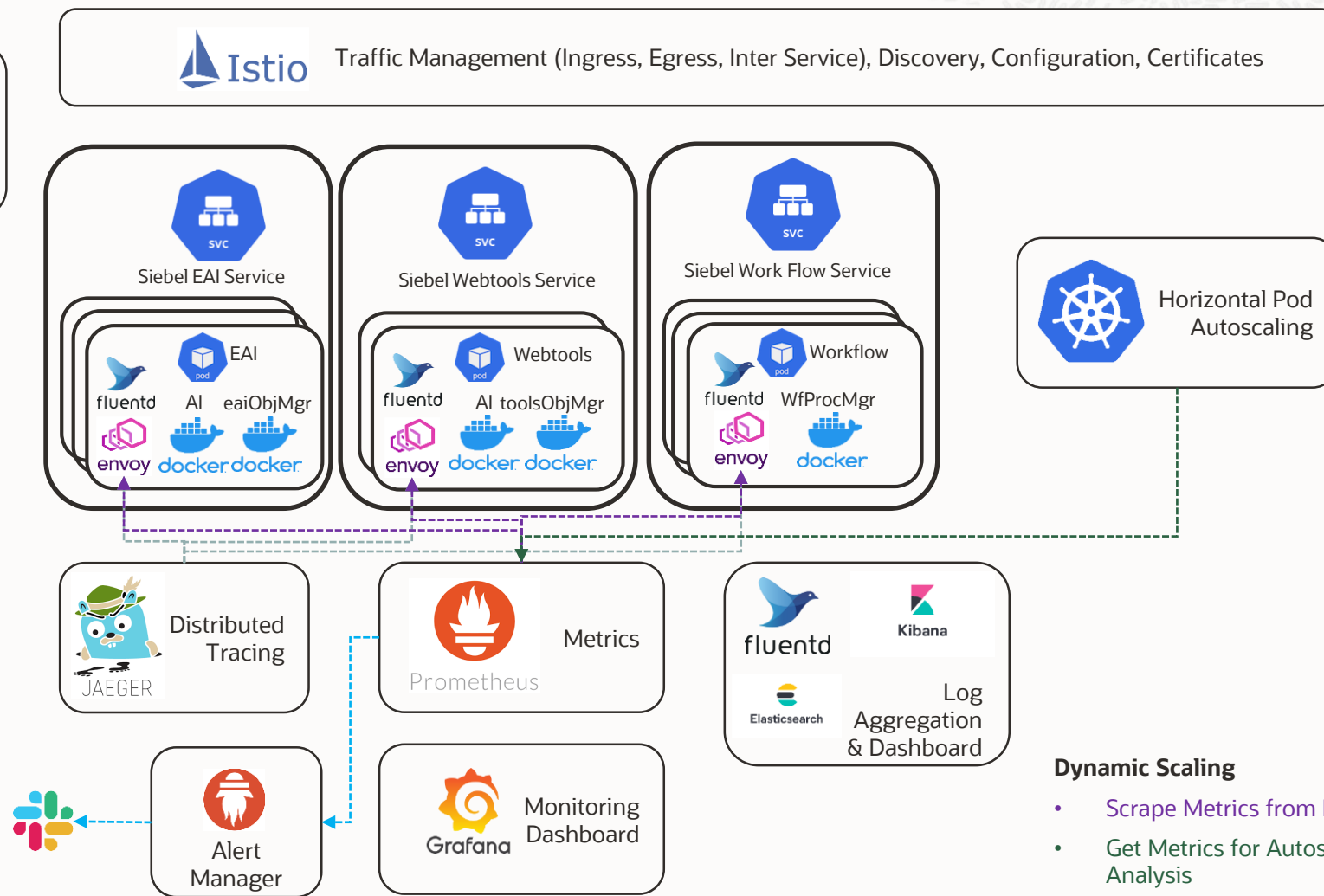
Cloud Native Infrastructure built around independent Siebel Components

Dynamic Scaling



CNCF Criteria

- Containerization:
- Orchestration & Application Definition:
- Service Proxy, Discovery & Mesh:
- Observability & Analysis



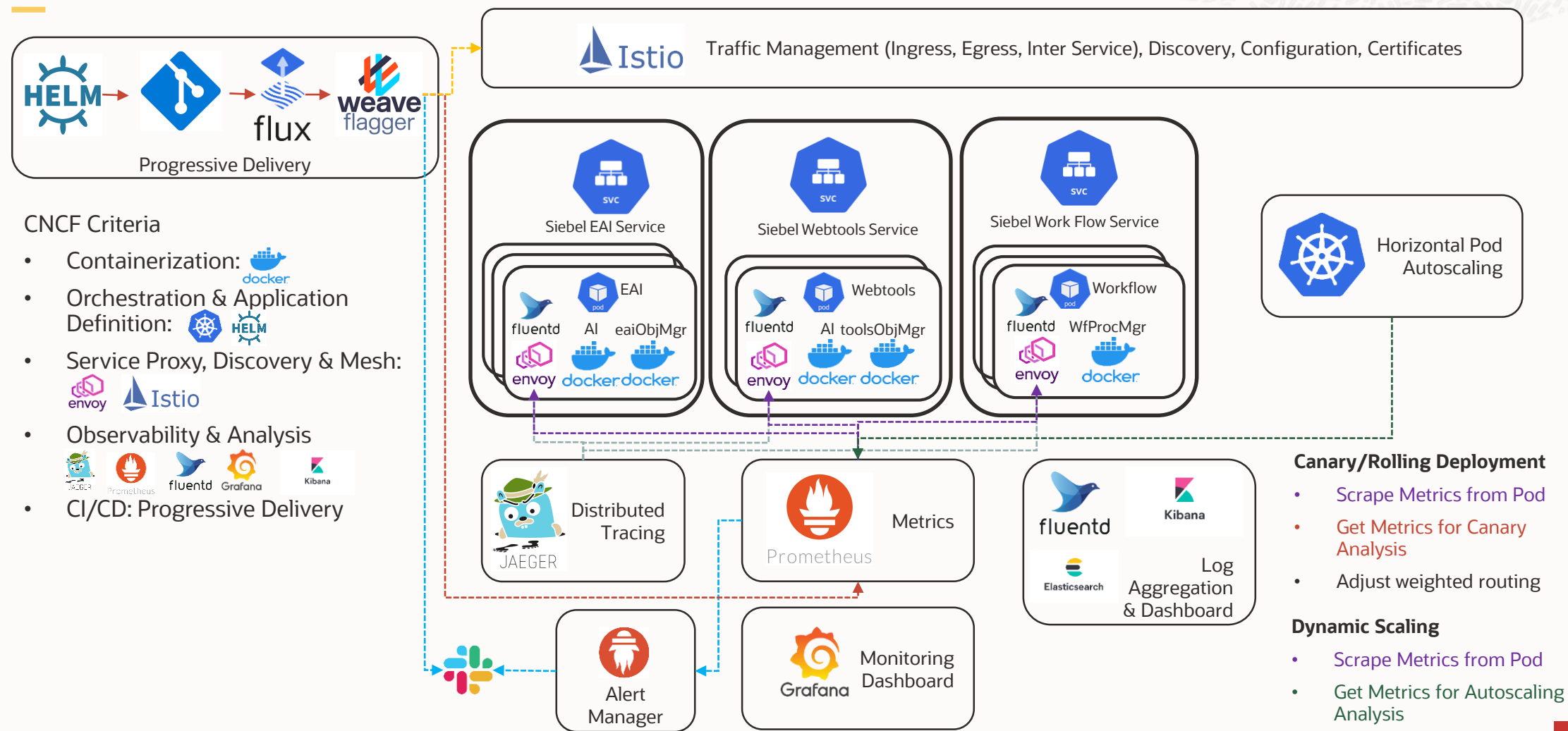
Dynamic Scaling

- Scrape Metrics from Pod
- Get Metrics for Autoscaling Analysis



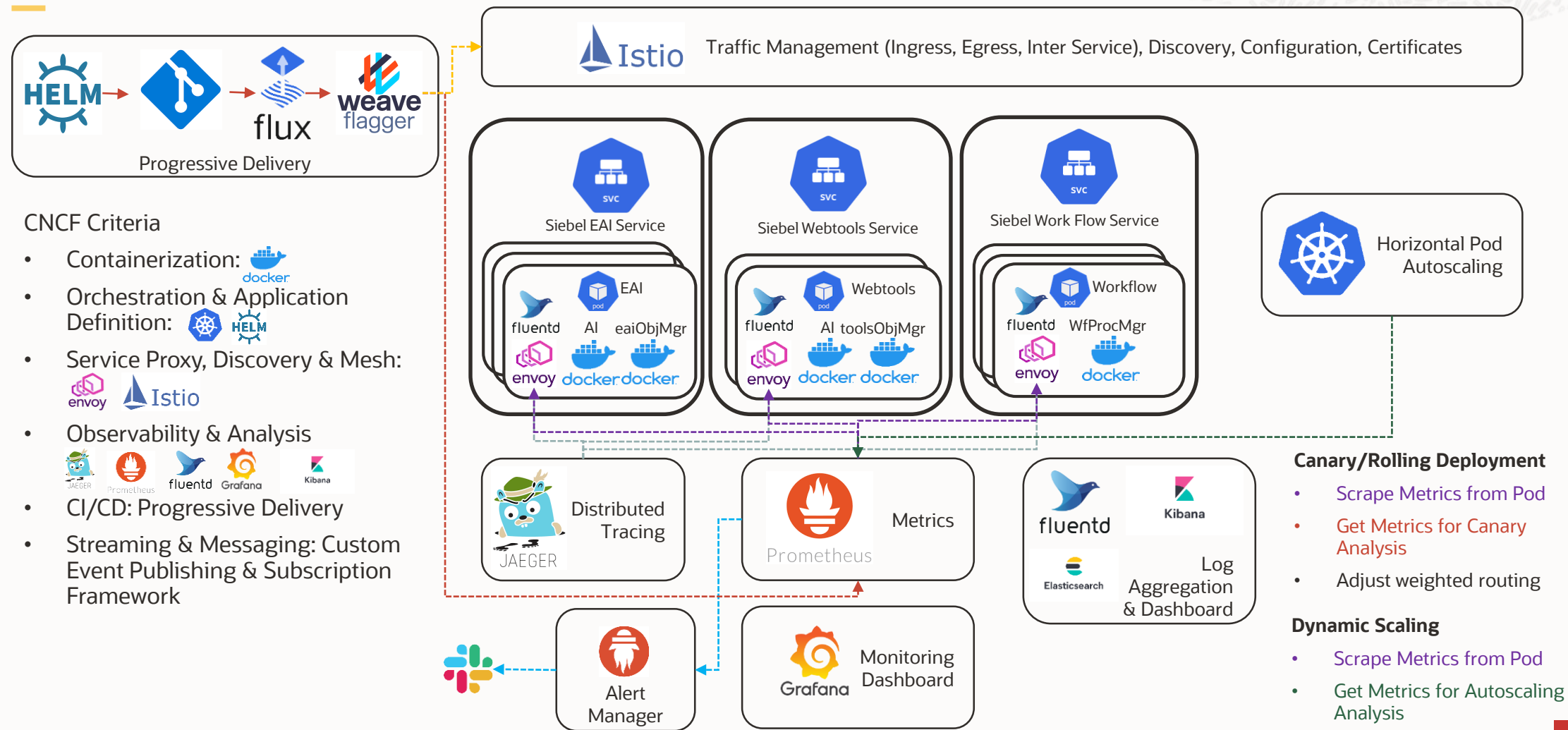
Cloud Native Infrastructure built around independent Siebel Components

CNCF Criterion – CI/CD - Progressive Delivery facilitating Canary / Rolling Deployments



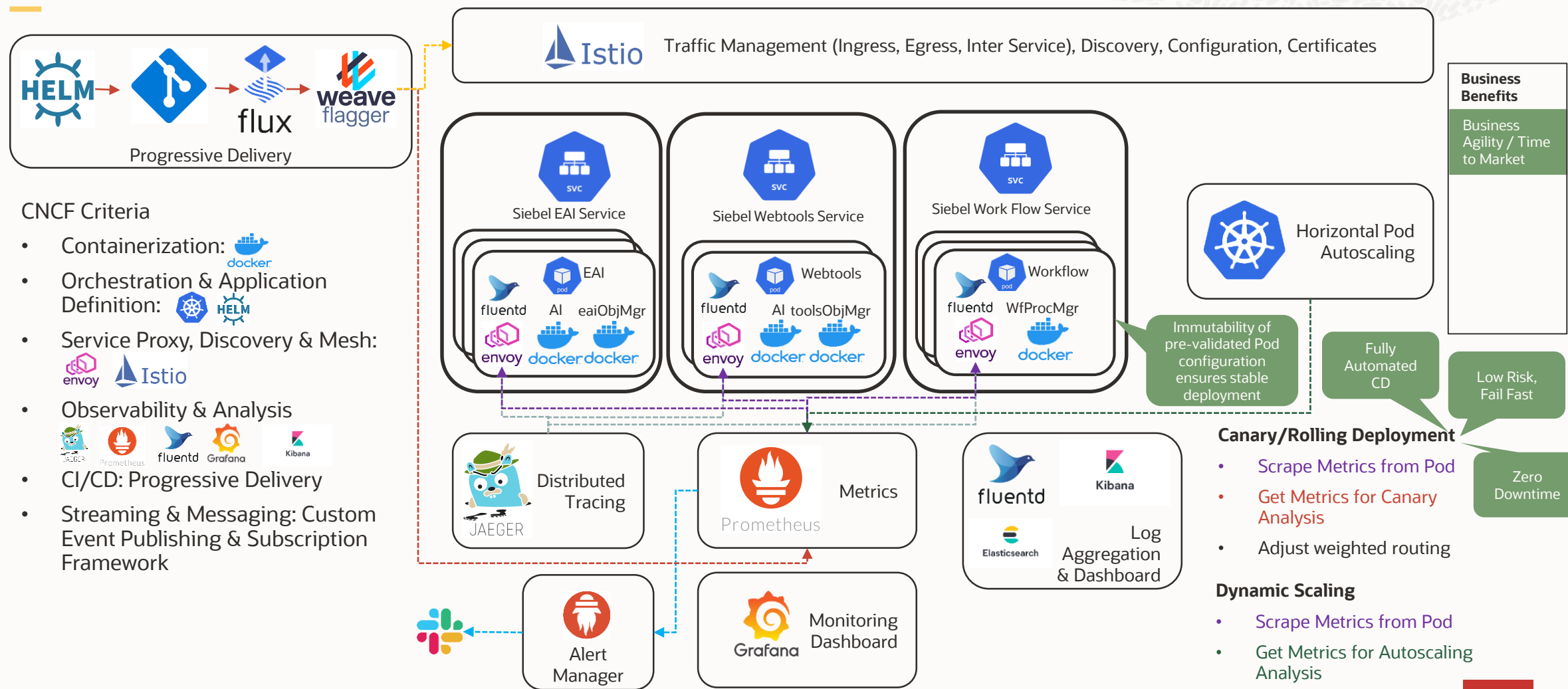
Cloud Native Infrastructure built around independent Siebel Components

CNCF Criterion – Streaming and Messaging



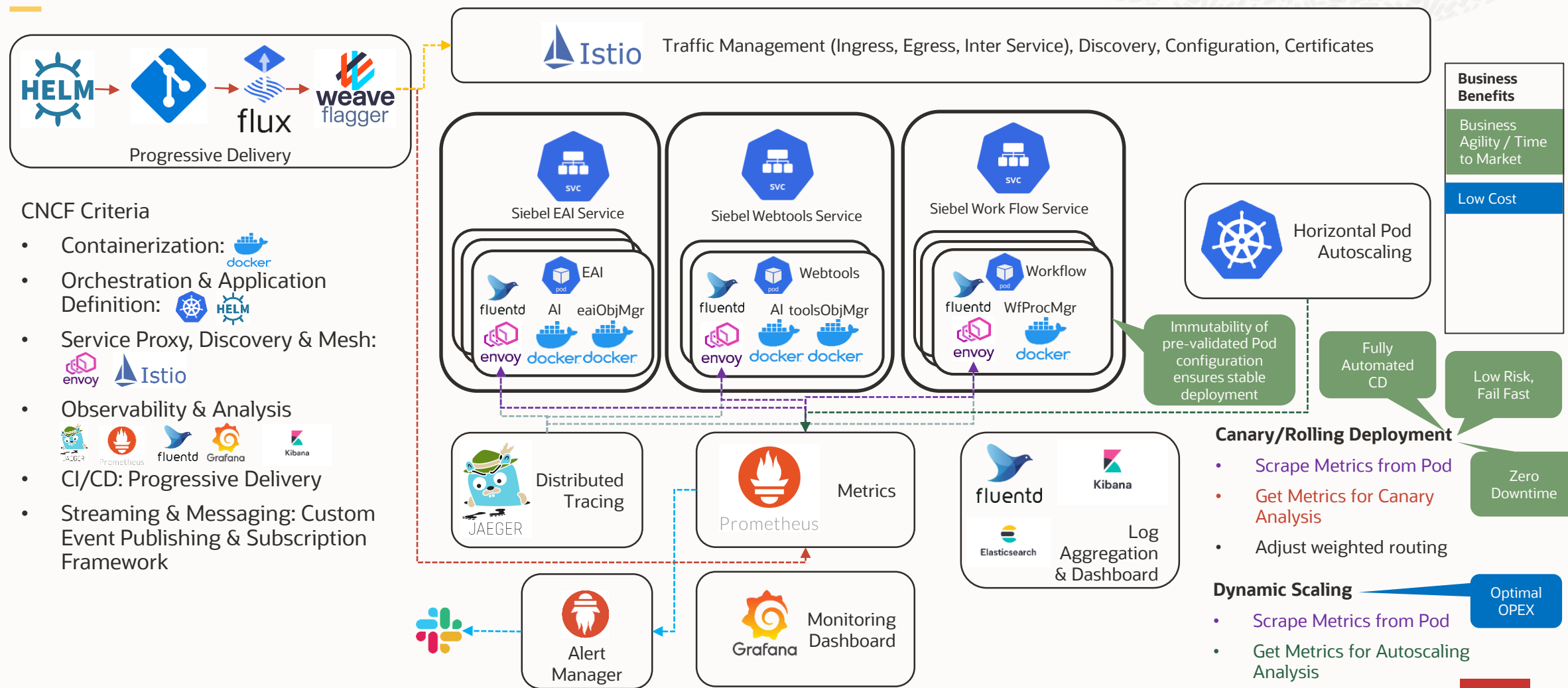
Cloud Native Infrastructure built around independent Siebel Components

Business Benefit - Agility / Time to Market



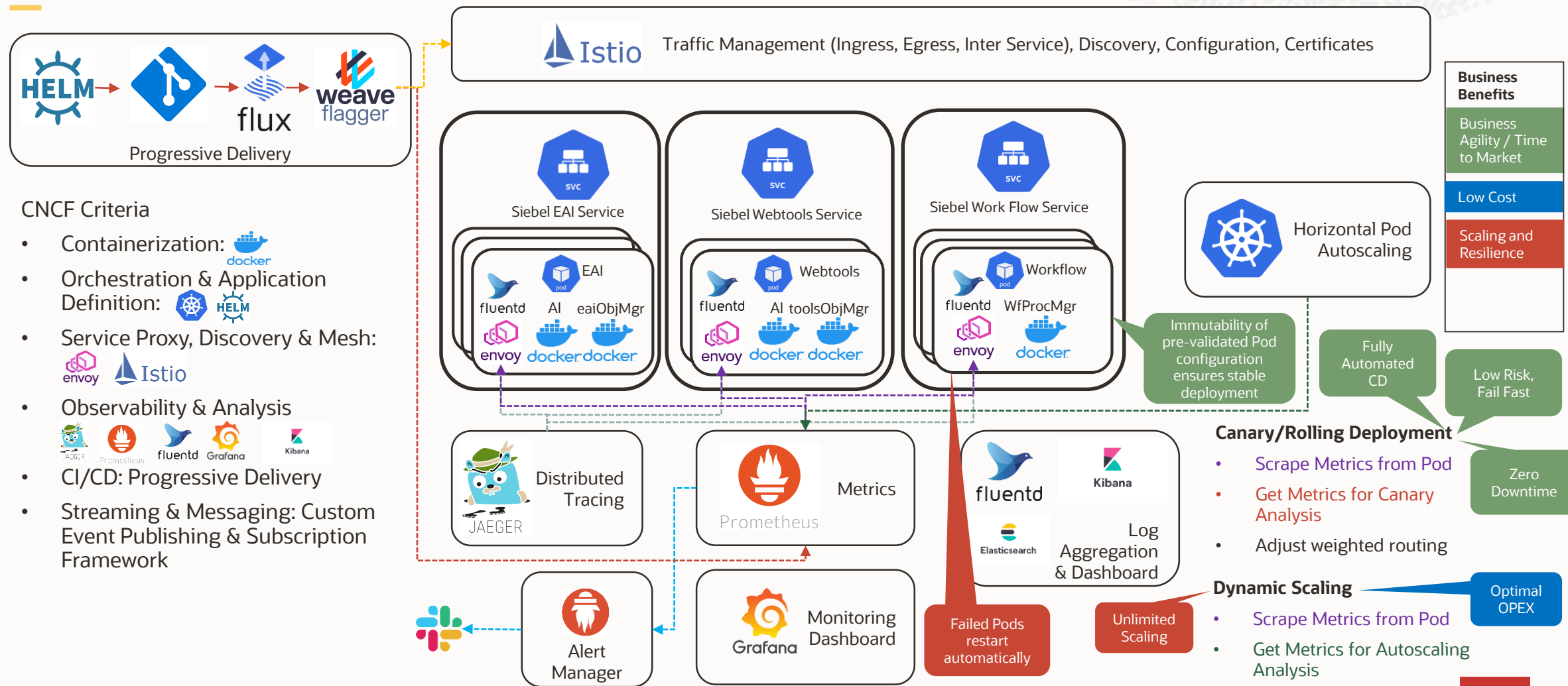
Cloud Native Infrastructure built around independent Siebel Components

Business Benefit – Low Cost



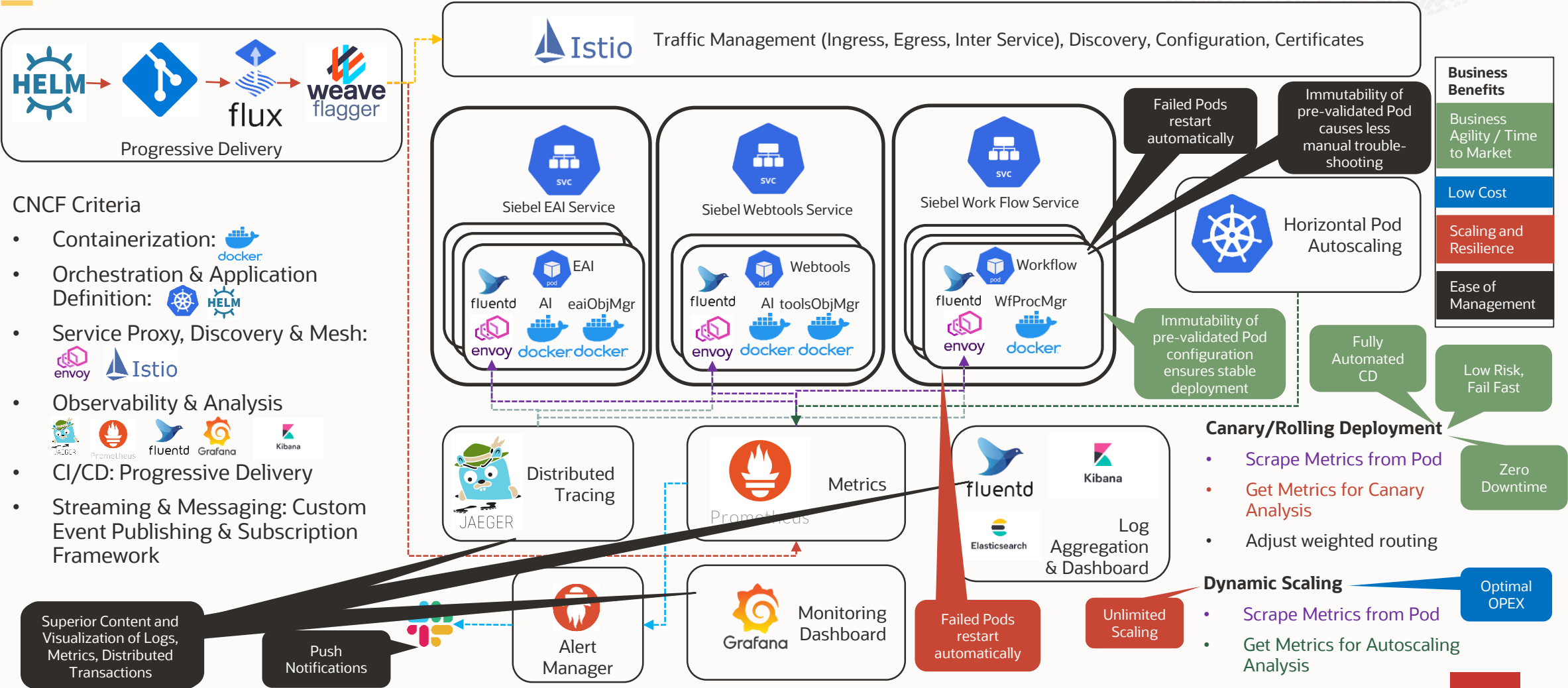
Cloud Native Infrastructure built around independent Siebel Components

Business Benefit – Scaling and Resilience for ML, Edge and other Massive Scale Use Cases



Cloud Native Infrastructure built around independent Siebel Components

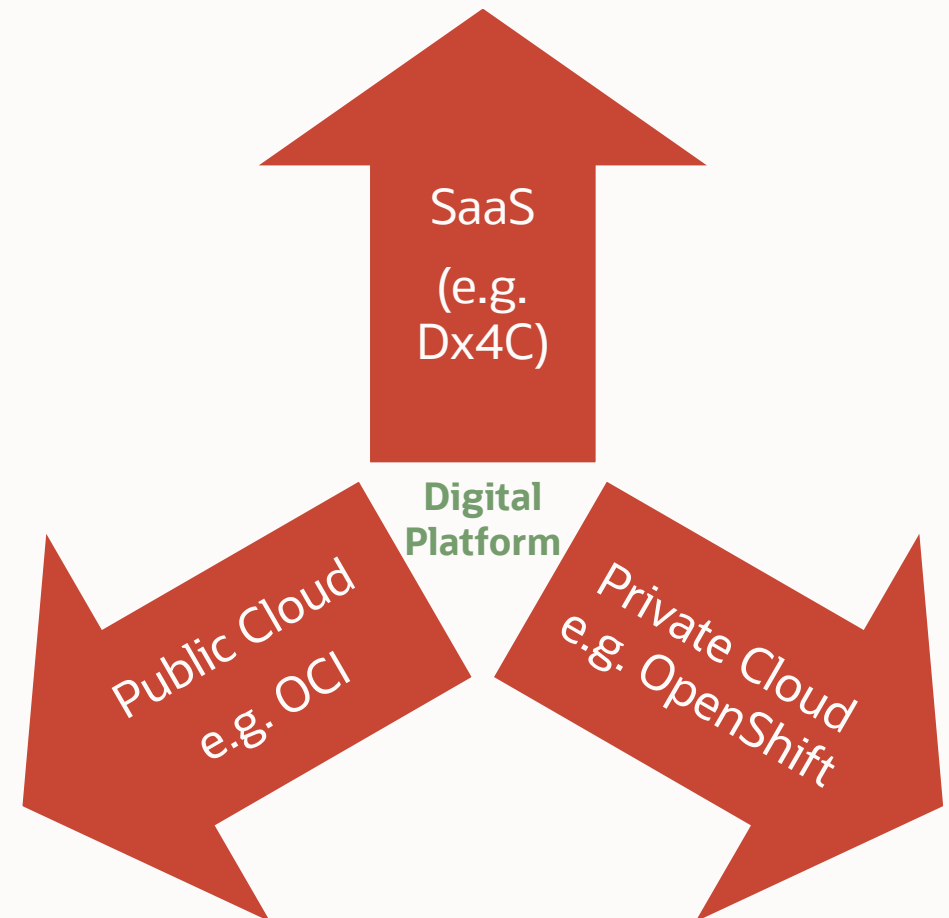
Business Benefit – Ease of Management



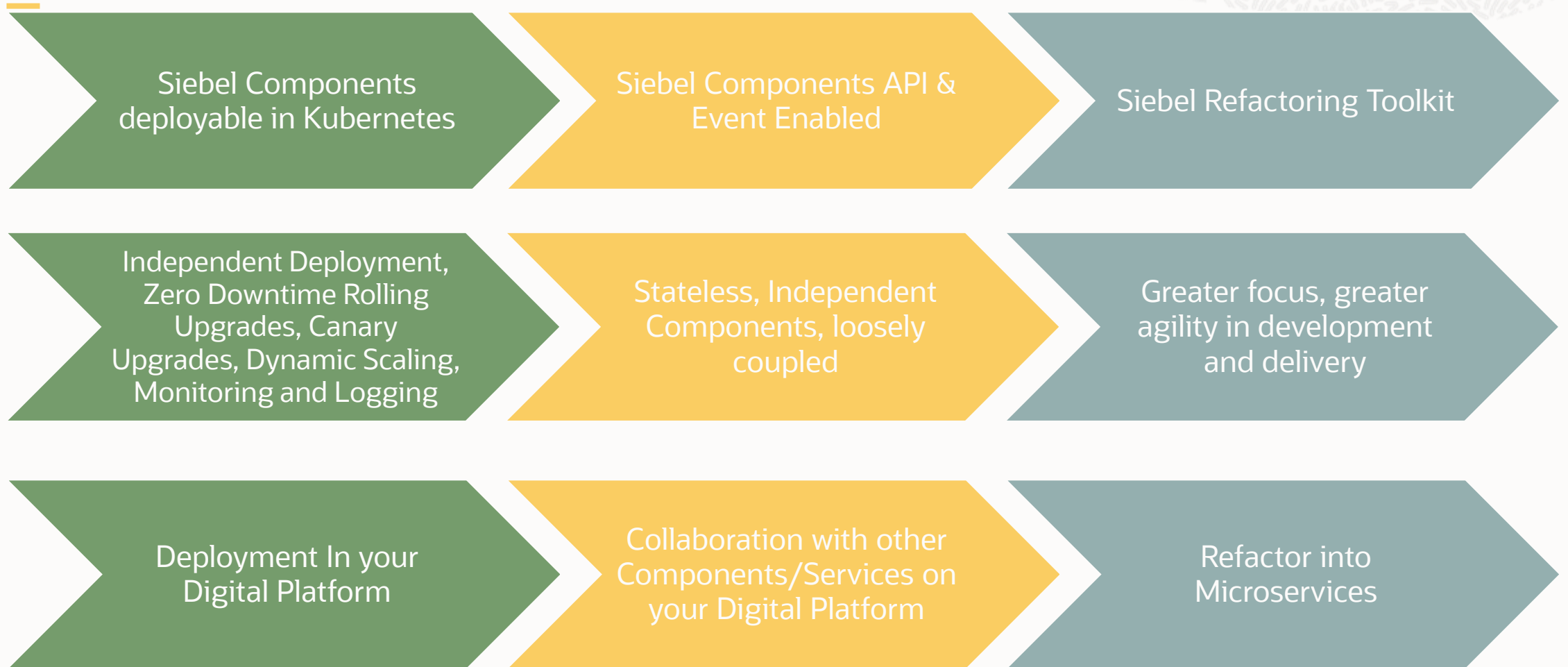
Why should you care?

Retaining Siebel ROI during your Digital Transformation journey

- Platforms underlying the technical side of a Digital Transformation strategy are inherently Cloud Native.
- Participating Applications / Services are expected to play well in a Cloud Native context.
- If using Siebel alongside other old or planned digital assets to drive fast paced generation of value is part of your goal, then Siebel Cloud Native should be something for you to try out.

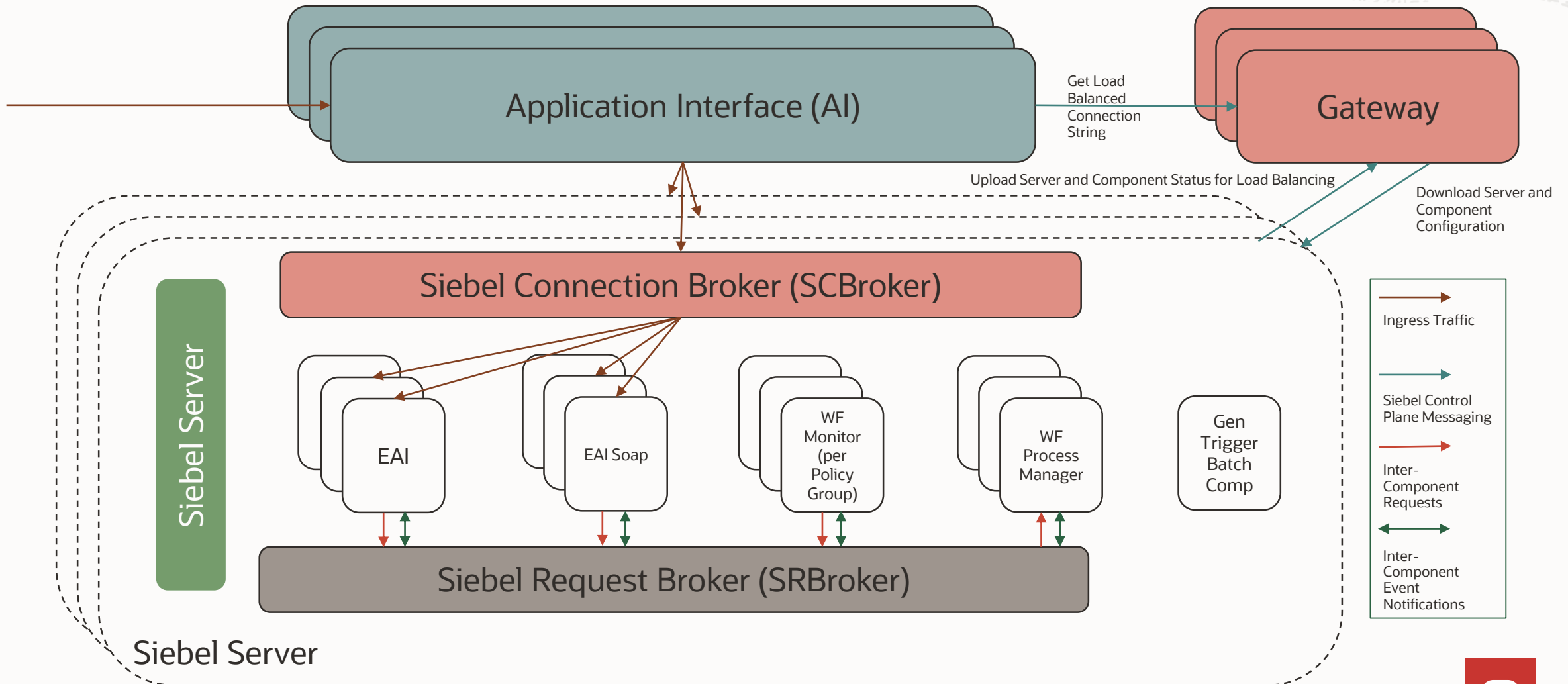


Siebel CRM as part of your Digital Transformation journey

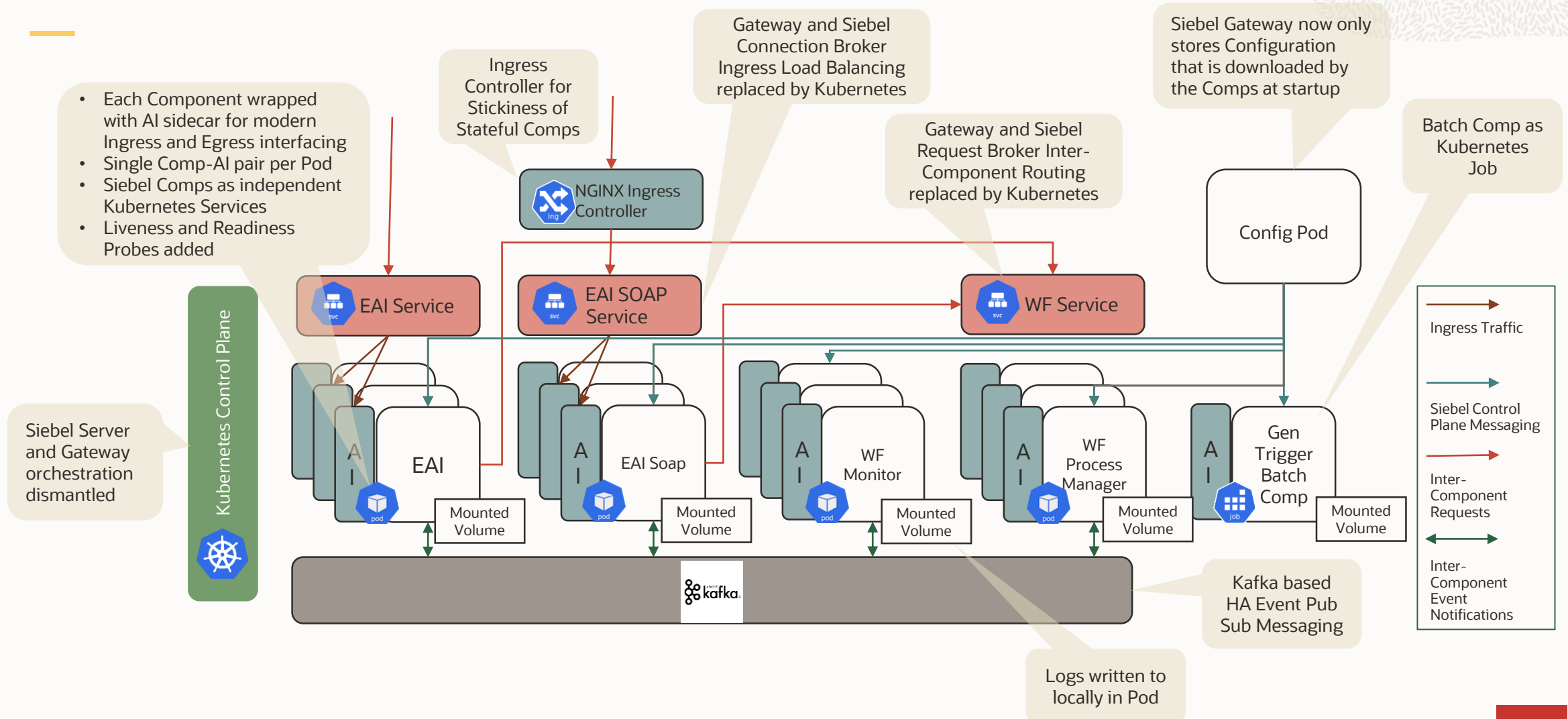


Siebel Cloud Native Implementation

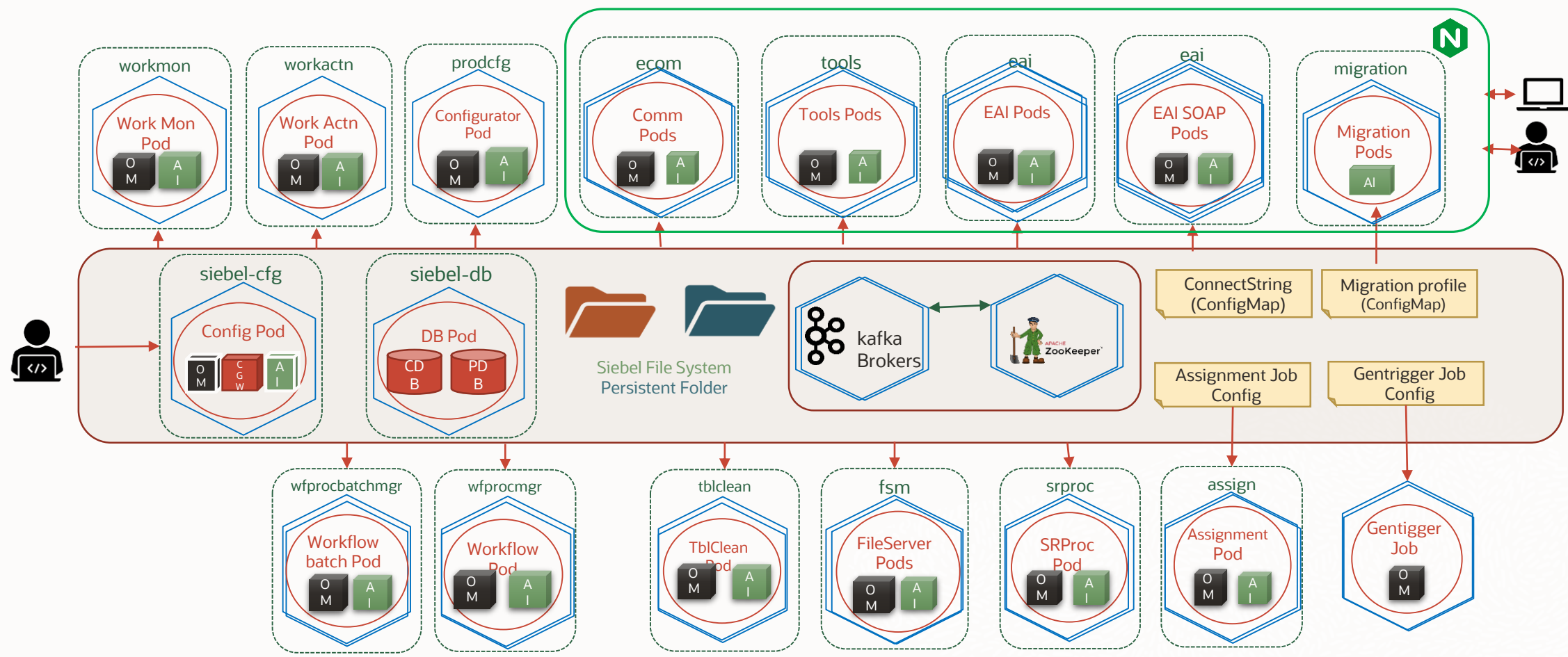
Siebel CRM - Current Architecture

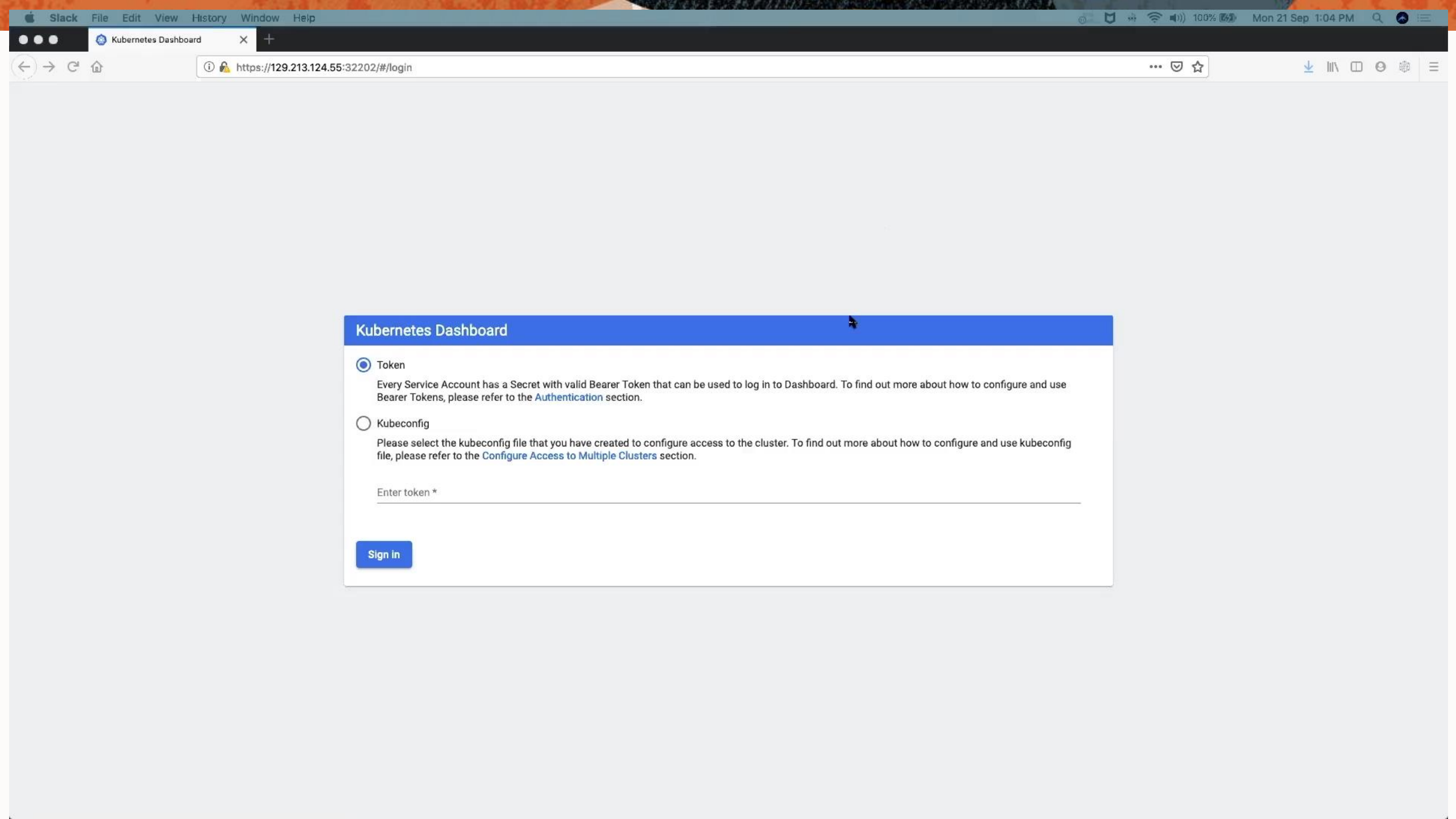


Siebel CRM - Cloud Native Architecture



Deployment Architecture





Kubernetes Dashboard

☒ Token

Every Service Account has a Secret with valid Bearer Token that can be used to log in to Dashboard. To find out more about how to configure and use Bearer Tokens, please refer to the [Authentication](#) section.

☐ Kubeconfig

Please select the kubeconfig file that you have created to configure access to the cluster. To find out more about how to configure and use kubeconfig file, please refer to the [Configure Access to Multiple Clusters](#) section.

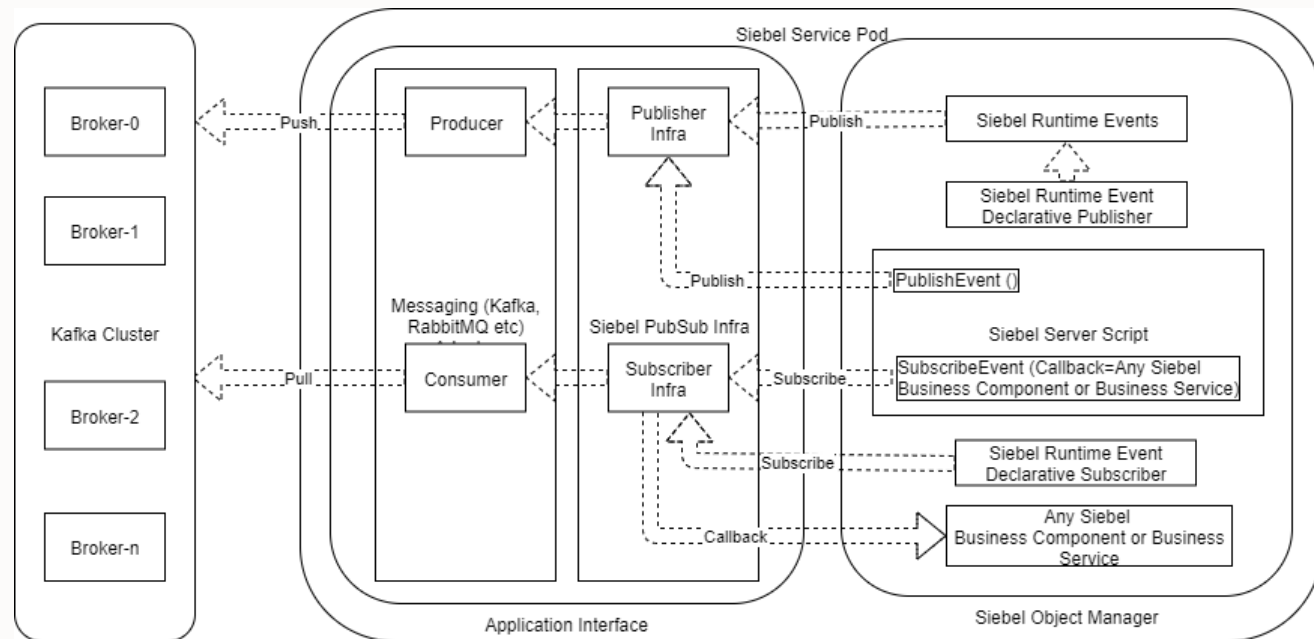
Enter token *

Sign in

Siebel Event Driven Framework

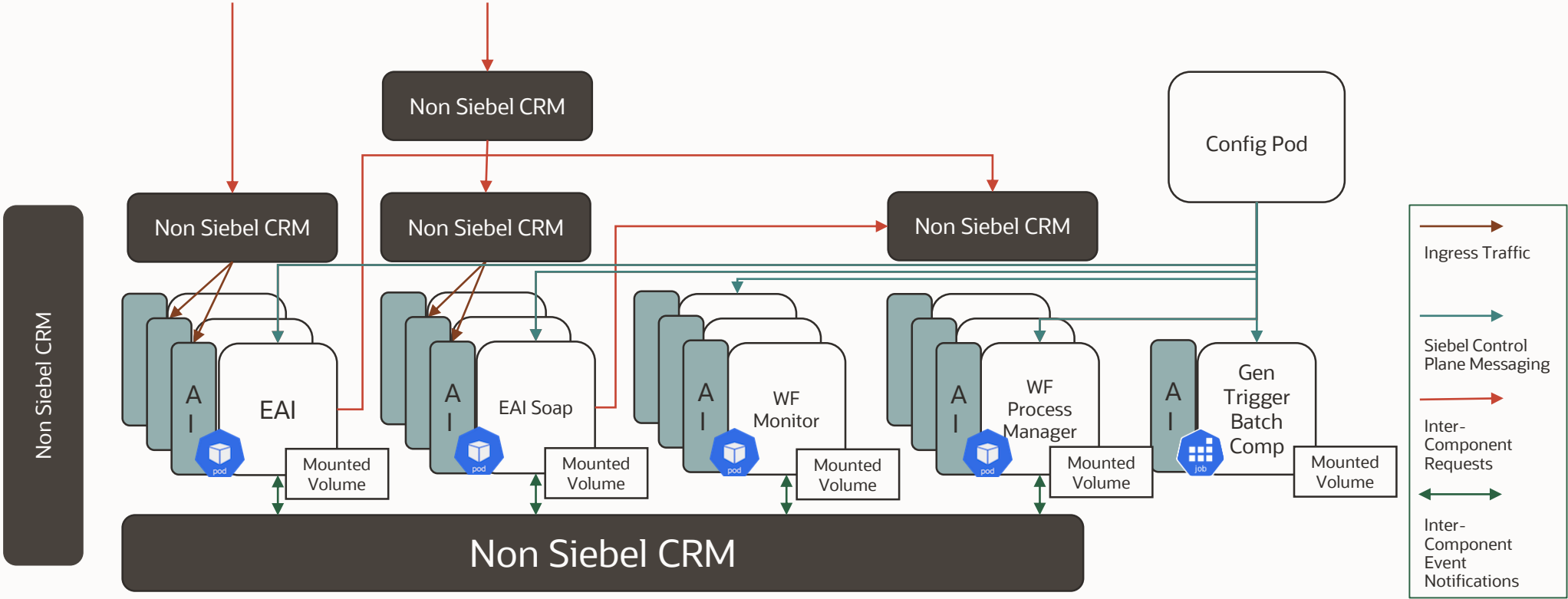
Cloud Native Pub Sub Messaging

- Pub Sub Based Event Messaging Infrastructure – adapter based so can work with Kafka, AMQP etc.
- Kafka cluster fault tolerant, auto scalable and highly available.
- Scripting API to Publish and Subscribe to Events.
- Declarative Configuration to Publish External Events on Siebel Runtime Events
- Declarative Configuration to Subscribe to External Events with Callback to any valid Siebel Invocation Endpoint (i.e. Business Component, Business Service etc.)
- Other internal touchpoints for PubSub of External Events can be added depending on requirements



What about Microservices ?

Siebel CRM – Deployment in Digital Platform



API and Event enablement of Siebel Components



- It is important to distinguish between splitting Siebel CRM into actual fine grained Microservices vs present Siebel Components as Services that are able to operate within a Cloud Native Digital Platform, collaborating with other external Microservices.
- This enables our existing customers to continue to extract value from Siebel CRM as part of their Digital Transformation journey.
- This is an identical strategy for our own SaaS platform - Digital Transformation for Communications (Dx4C)

Siebel Microservices

While we can do infrastructural changes to allow for Services (Siebel Components) with independent Deployment cycles, Services can only achieve truly independent life cycles across Dev and Ops when they are as independent as possible from other Services (Siebel or non Siebel) around them, by

- Addressing a **cohesive well bounded** Functional/Business context
- Have **low coupling** (implementation, temporal, deployment, domain) with other Services
- Have **stable Interfaces** (via an API First approach) that insulate other collaborating Services from change
- Have completely **independent stacks** i.e. do not share any layers from UI/API down to DB

The above are the typical expectations from a true Microservice.

Microservices are a natural progression to Cloud Native deployment and collaboration with other Cloud Native assets.

Siebel Cloud Native Architecture vis-à-vis Siebel Cloud Enablement (IP 17+)

Additional Cloud Native Deployment Option

Zero Downtime



- Parallel Development
- Siebel Composer
- Outbound REST, Open API
- Event Based Messaging

- Docker Containers including Repository



- Kubernetes Orchestration
- Rolling or Canary Zero Downtime Progressive Deployment



- Kubernetes and Service Mesh based Orchestration
- Automated Scaling



- Logs, Metrics, Distributed Requests Capture and Analysis using Best of Breed Tooling
- Usage-Pattern Tracking

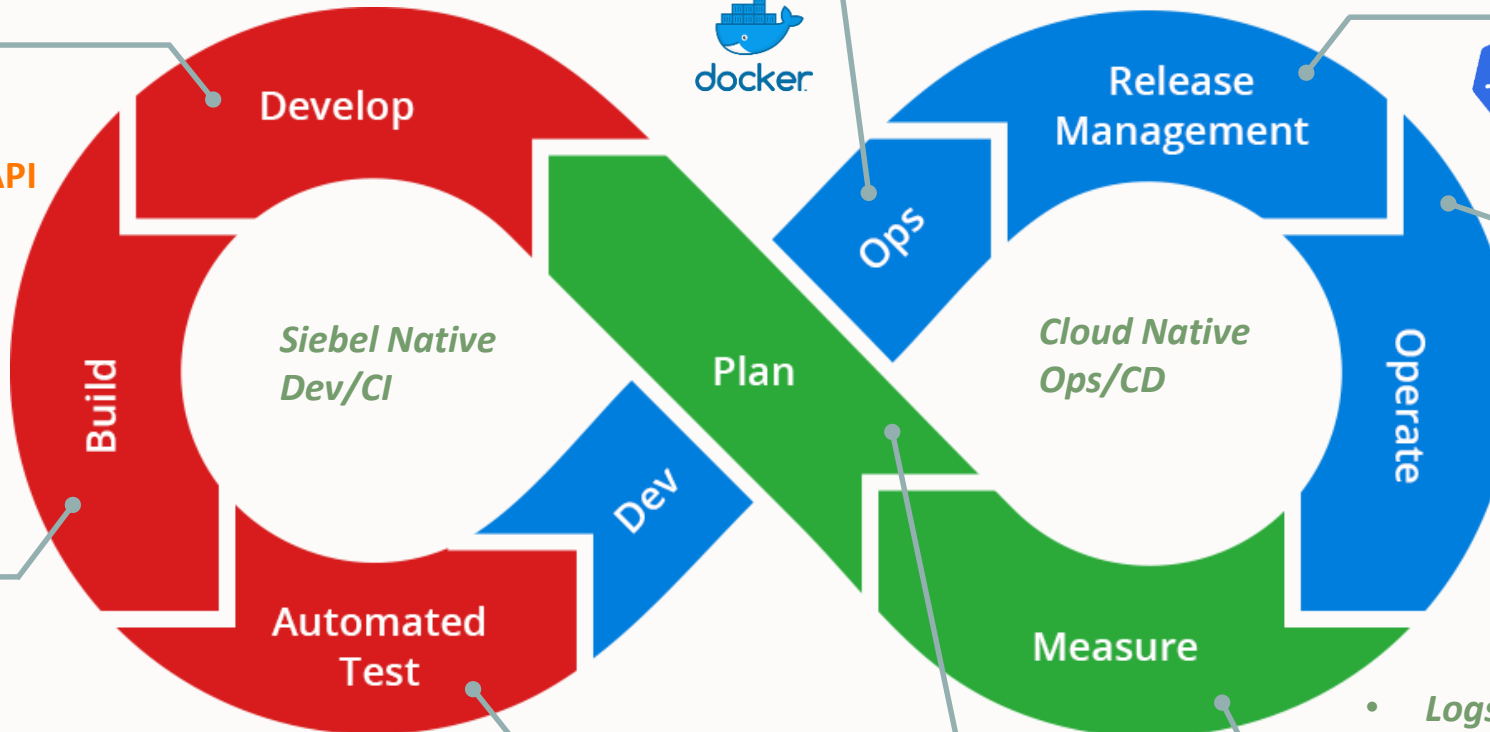


- Roadmap & Statement of Direction (SOD)
- Based on CAB input

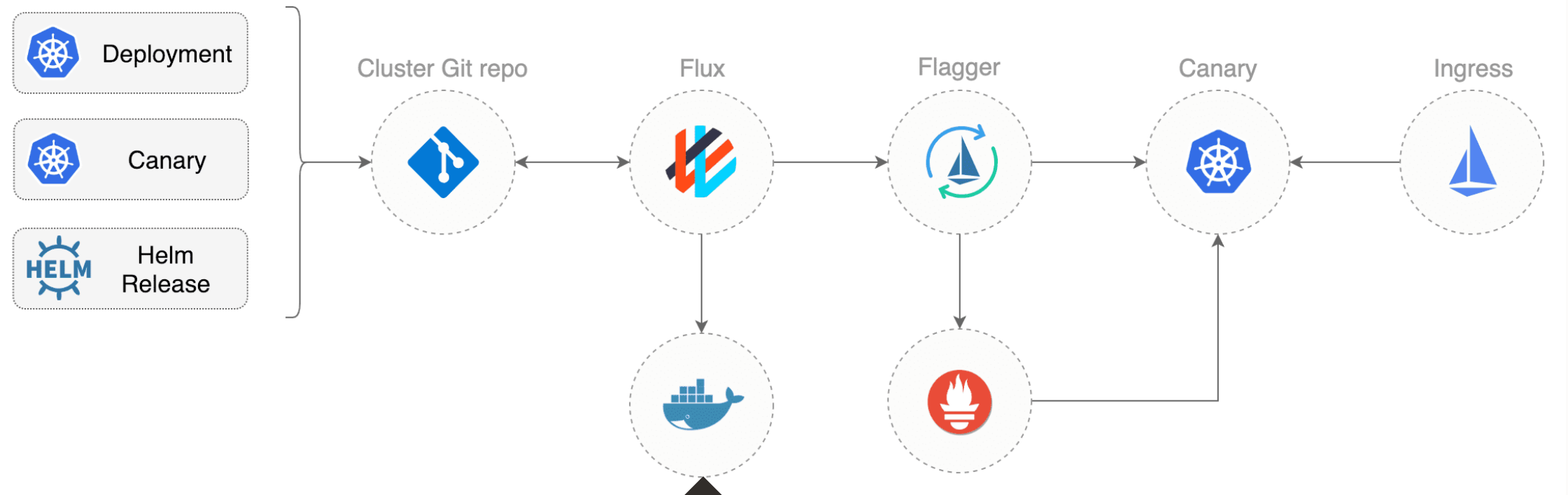
- Built-in Test Automation
- Siebel Test Execution Jenkins Plugin



- Docker Containers
- Jenkins DevOps pipeline



Siebel DevOps features upstream from Cloud Native Deployment



Siebel CI Process leveraging technologies such Workspaces & Parallel Development

Migration Effort

Key Operational Journeys **today** - Classic vs Cloud Native Siebel

Journey	Classic	Cloud Native	Comments
Installation	Installer	Docker Images	
Deployment	Gateway API / SMC	Gateway REST API for Components, Local Configuration for AI and other nodes	<p>Moving completely away from configuration in the Gateway is the goal, for the following reasons (at the least)–</p> <ul style="list-style-type: none"> Pods should ideally be completely independent of each other (including Gateway) Downloading Configuration from the Gateway impacts Component startup time
Server Administration - Configuration	Server Manager (UI/CLI) / Gateway API / SMC	NA	Immutability dictates that every change must be a complete repave of relevant pods.
Server Administration - Monitoring	Server Manager (UI/CLI)	<p>Metrics: Istio-Prometheus-Grafana Logs: Fluentd-ElasticSearch-Kibana</p> <p>We have used these, and they are among the most popular, but Siebel Components will be ensured / enhanced to provide whatever information is needed, so other tools that can source this information from the Pods should also work.</p>	<ul style="list-style-type: none"> With the Gateway now only being a configuration store with a one time pull by each Component pod at launch, Gateway based Monitoring via Server Manager or solutions such as Oracle Enterprise Manager sitting on top of them will not work. More modern Cloud Native tools that Scrape and Visualize Request details, Metrics or Logs from individual independent Pods will be used
Server Administration - Jobs	Server Manager (UI/CLI)	Kubernetes Jobs	<p>Again a natural consequence of</p> <ul style="list-style-type: none"> Kubernetes deployment Gateway no longer orchestrating the topology
Migration	Siebel Migration for Application Data & Seed as well as other artifacts such Repository, Web Files etc.	<ul style="list-style-type: none"> Siebel Migration for Application Data & Seed No Migration for Repository, Web Files etc. 	Immutability dictates that things like Repository or Web Files which are part of the base implementation of the deployed Application must be maintained in a version store (ideally versioned images), and only deployed via a complete repaving of Pods.

Business Logic

A decorative graphic in the top right corner consisting of a light gray fingerprint pattern.

- Until now nothing has changed in the Siebel stack. So Business Logic should remain untouched.

What's Next?

To Do ...



- Reduce Component Startup Time
- Retire Gateway for Cloud Native Architecture
- Localize Repository to allow for Rolling upgrade
- Enrich telemetry and logging to allow for real world Canary, A/B, Rolling updates
- Add process and framework for Schema and Seed upgrade to work with present and target Service versions

Thanks!

Chandan Dasgupta

✉ Chandan.dasgupta@oracle.com

📄 blogs.oracle.com/siebelcrm



ORACLE

Take the Siebel CRM Innovation Survey



Let us help you kickstart your
Siebel CRM transformation

<https://go.oracle.com/siebelcrm-innovation> 





Stay Connected
blogs.oracle.com/siebelcrm

Useful Resources



[Siebel CRM Blog](#)

[Siebel CRM YouTube](#)

[Siebel CRM Sales Team](#) ✉

[Siebel CRM ACS Services](#) ✉

[Oracle Support Value](#)

[Partner Spotlights](#)



[Siebel CRM Learning Subscription](#)
(Free content, click Preview)

[Siebel CRM Bookshelf](#)

[Siebel CRM Github](#)

[Siebel CRM Advisor Webcasts](#)

[My Oracle Support Community](#)



[Siebel CRM Statement of Direction](#)

[Siebel CRM Release Updates](#)

[Siebel CRM Premier Support](#)

[Datasheets – Features by Release](#)

[Siebel CRM Ideas](#) (Collaboration)



[Siebel CRM Customer Connect](#)
[CAB portal](#)

[LinkedIn Customer Connect](#)

Newsletter Email Distribution list
([Customer](#)) & ([Partner](#))

[Virtual CAB replays](#)