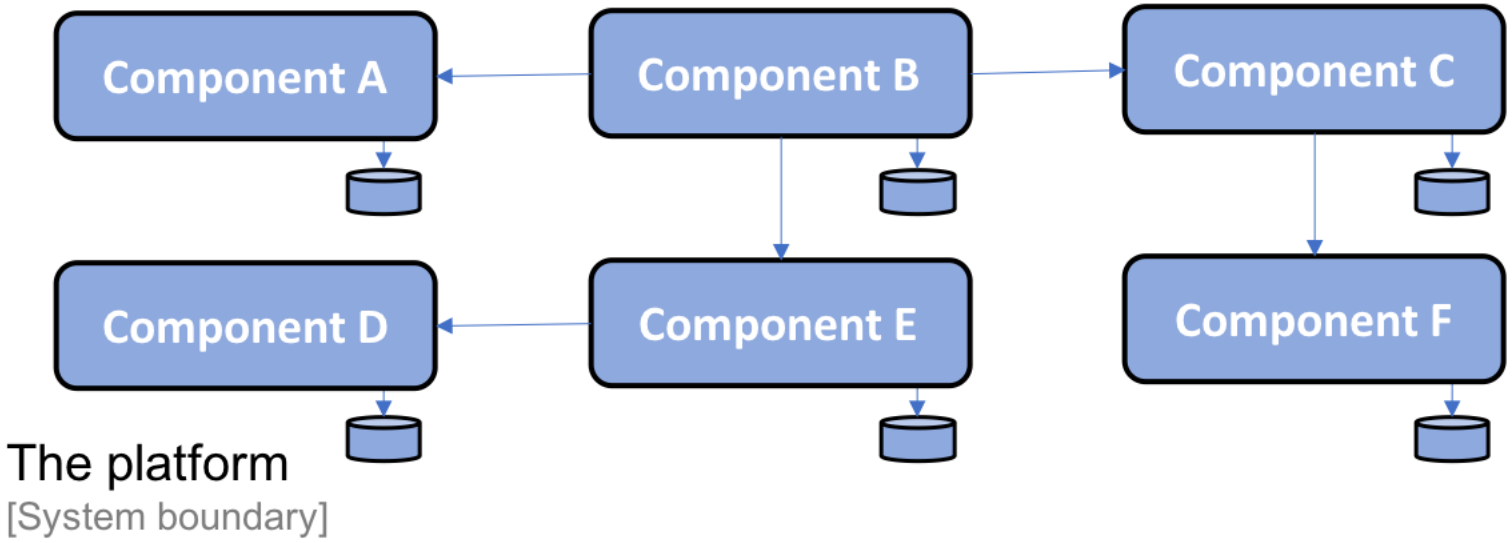
JOEDAYZ academy

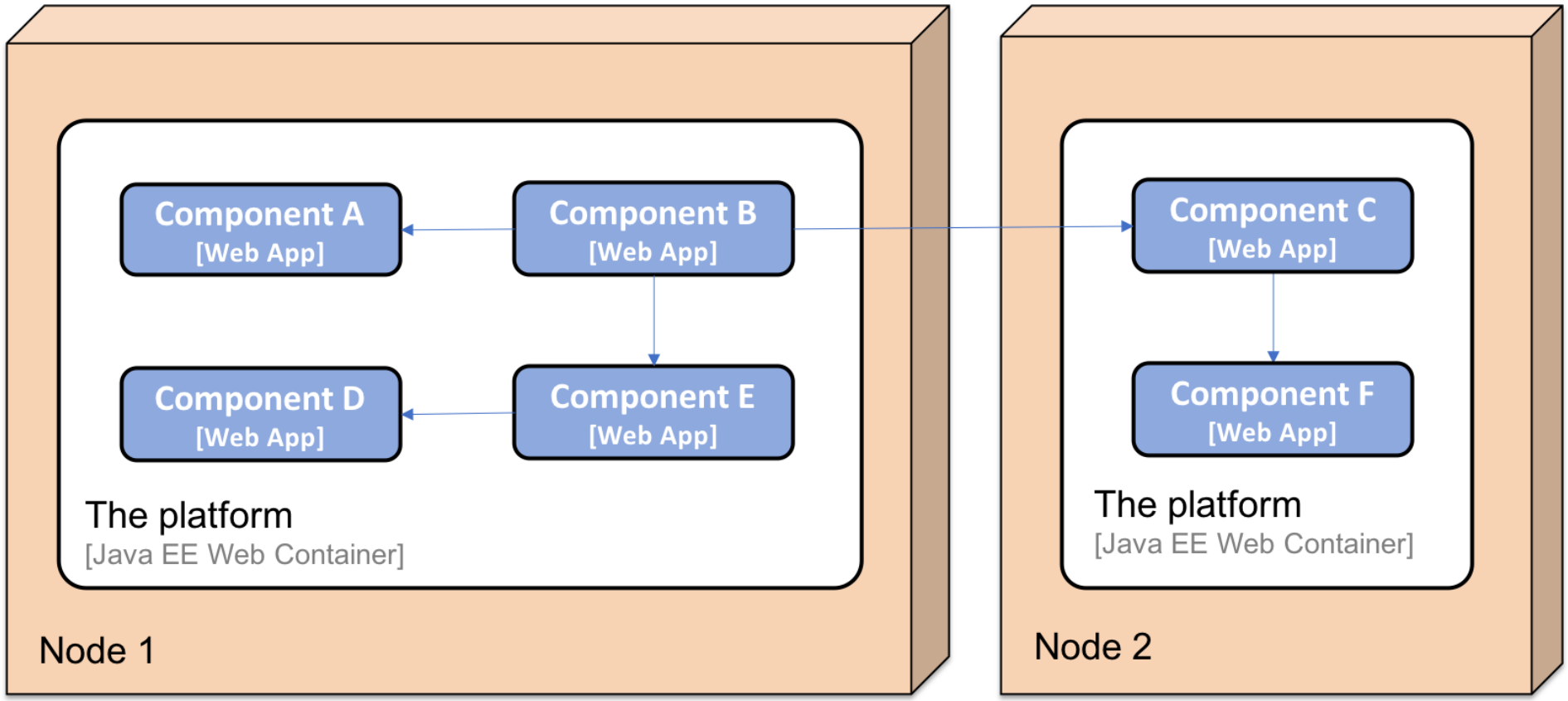
Introducción a Microservicios

@joedayz

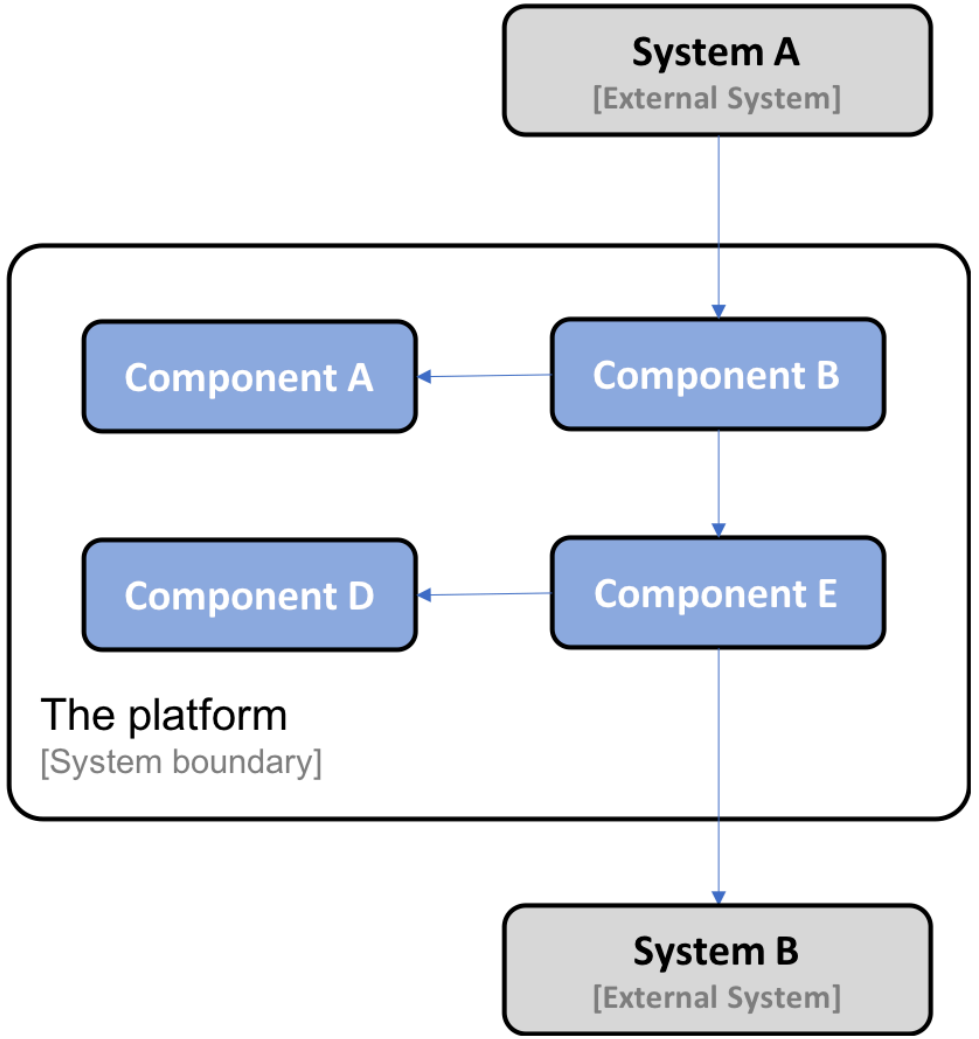
Clase 1

Mi camino a los micro servicios

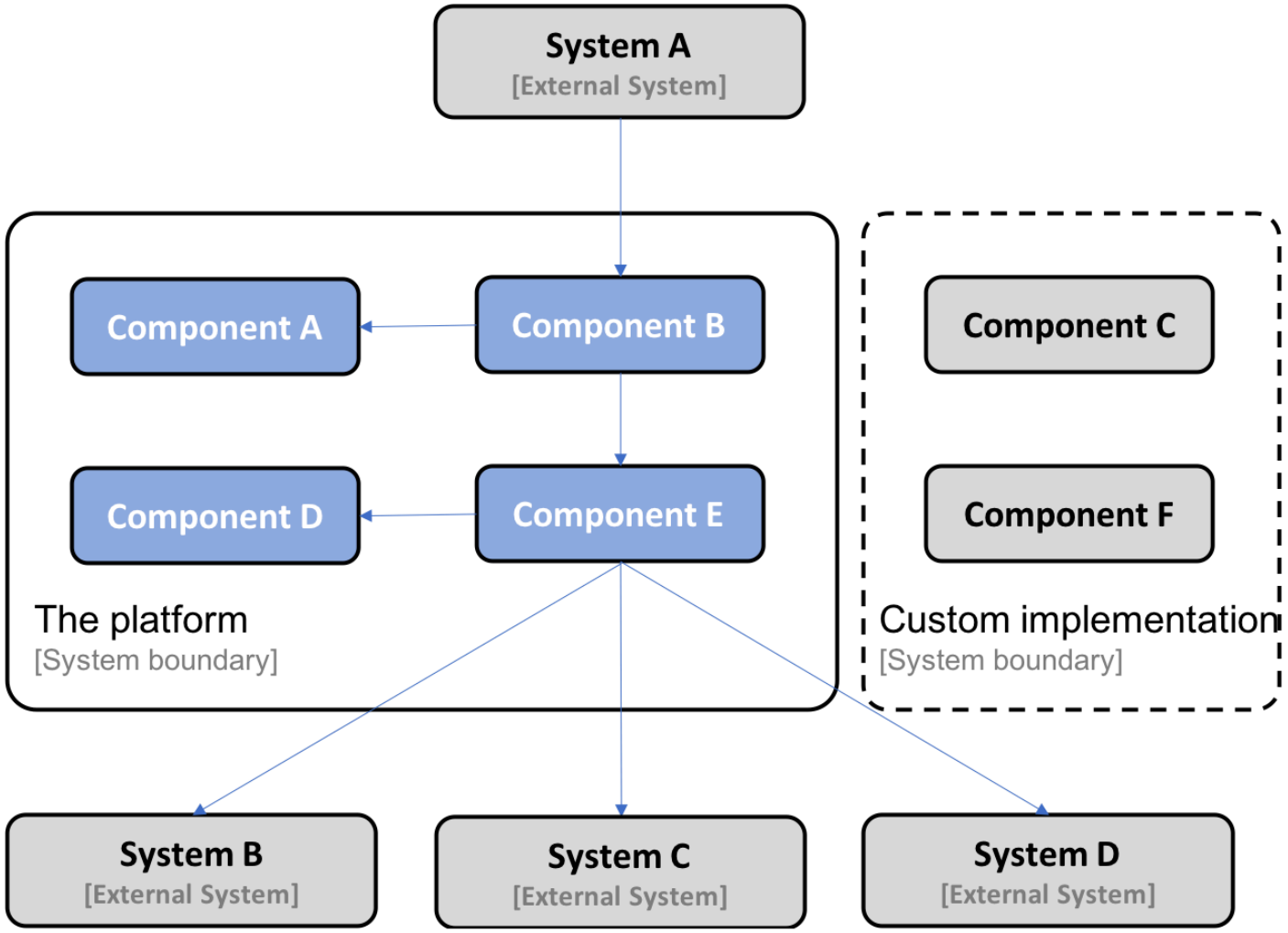



Component A
[Web App]
Component B
[Web App]
Component D
[Web App]
Component E
[Web App]
The platform
[Java EE Web Container]
Node 1
Component C
[Web App]
Component F
[Web App]
The platform
[Java EE Web Container]
Node 2

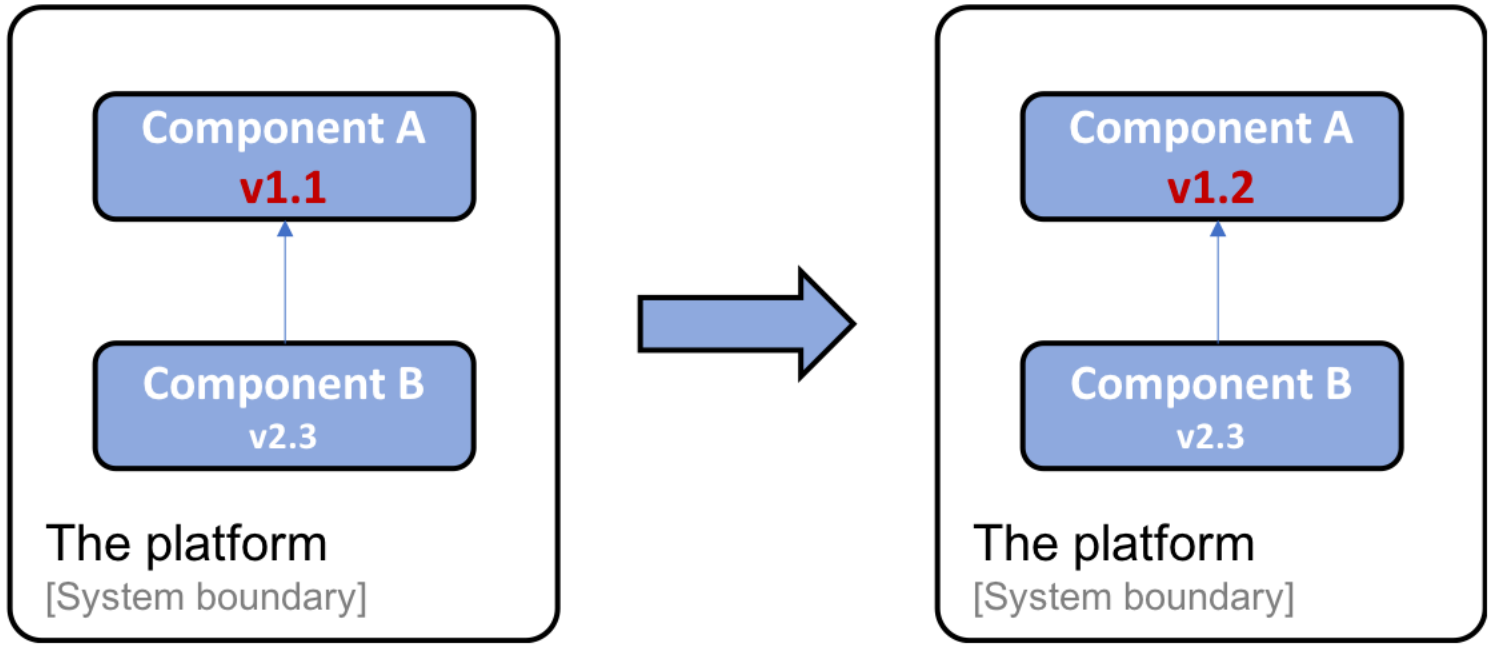
Beneficio de los componentes de software autónomos

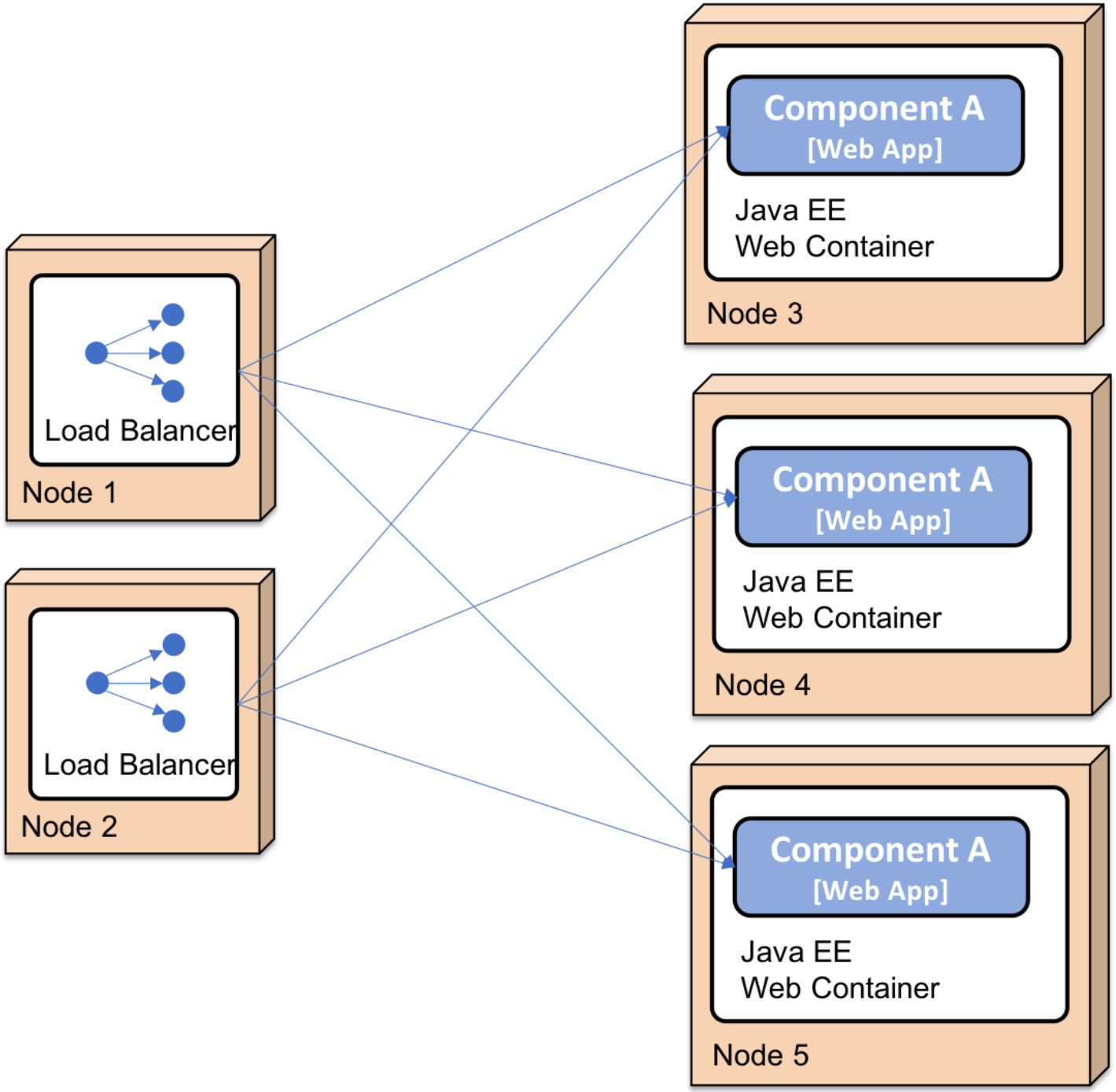


Componentes del cliente



Actualizaciones del componente




Load Balancer
Node 1
Load Balancer
Node 2
Component A
[Web App]
Java EE
Web Container
Node 3
Component A
[Web App]
Java EE
Web Container
Node 4
Component A
[Web App]
Java EE
Web Container
Node 5

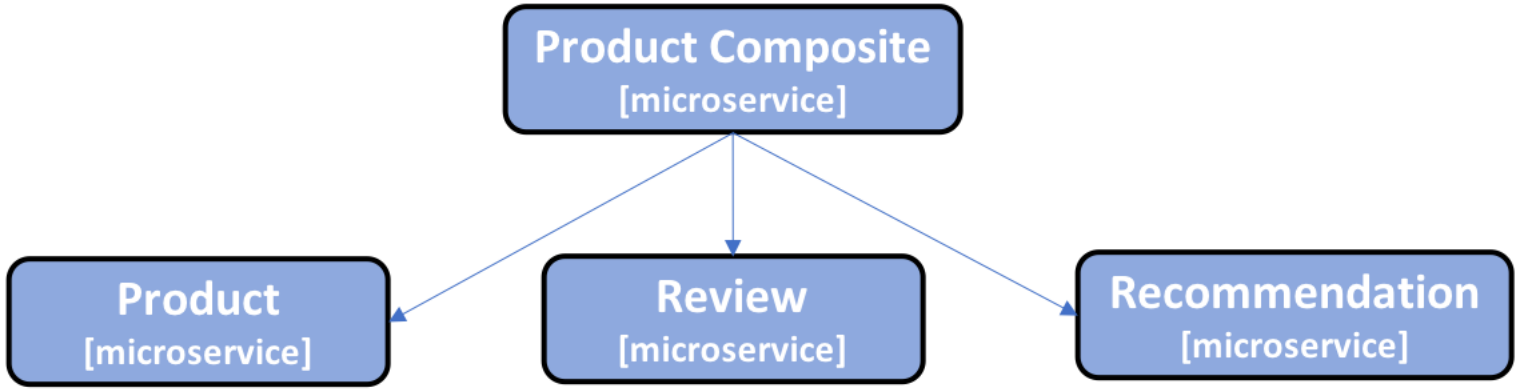
Retos de los componentes de software autónomos

- Agregar nuevas instancias
- Posibilidad alta de cascada de fallas debido a comunicación síncrona entre componentes
- Configuración consistente y actualizado a la fecha de las instancias
- Monitorear el estado de la plataforma
- Coleccionar archivos log de los componentes distribuidos

Aparecieron los micro servicios

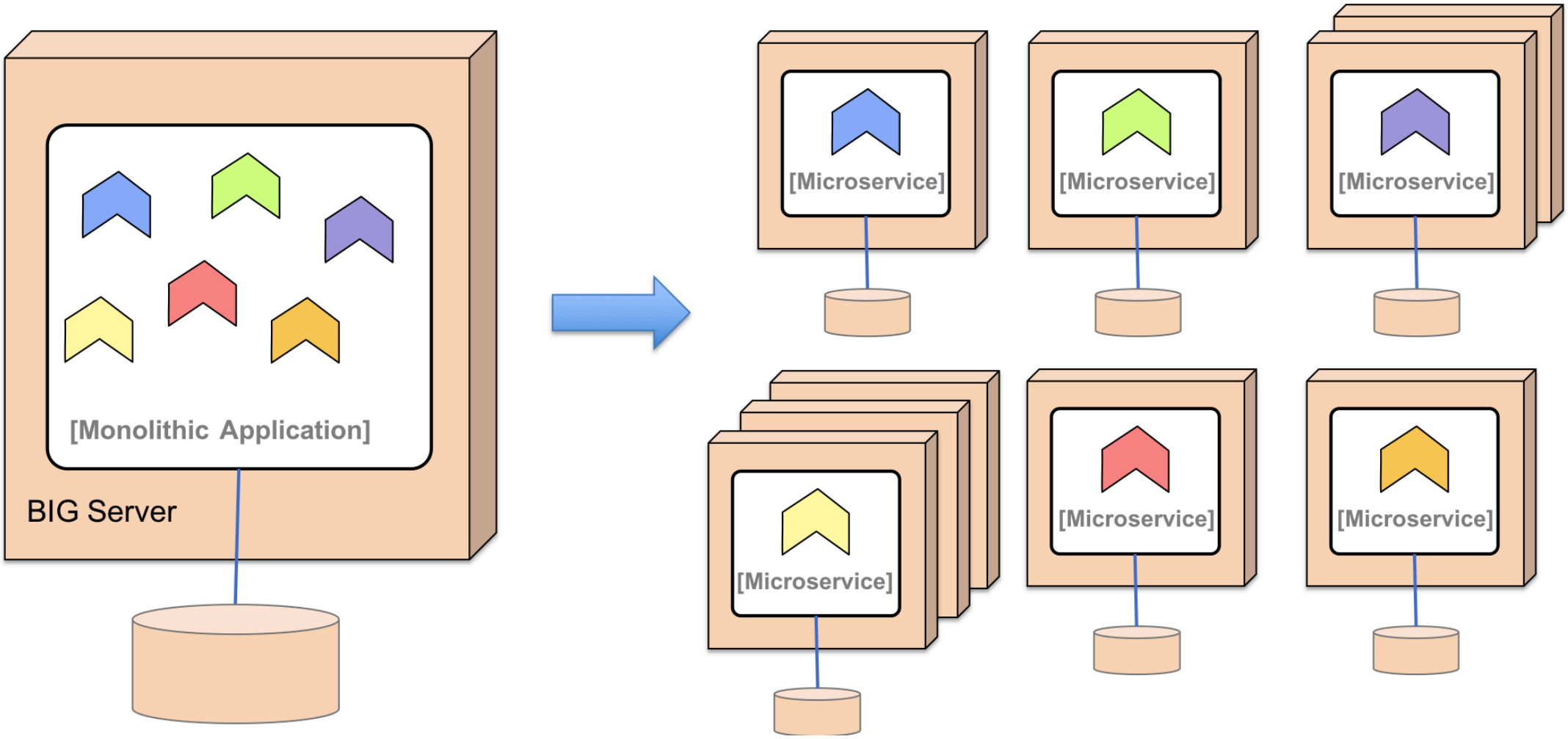
- En el 2014, aparecieron nuevos proyectos open source que brindan herramientas y frameworks que simplifican el desarrollo de microservicios.
- Pivotal libera Spring Cloud, el cual es un wrapper de Netflix OSS brindando service Discovery, configuration management, distributed tracing, circuit breaking, etc.
- Docker revolution.
- Orquestadores de Contenedores como: Apache Mesos, Docker in Swarm mode, Amazon ECS, HashiCorp Nomad, y **Kubernetes.** Google dono Kubernetes a la https://www.cncf.io/
- Service Mesh viene a complementar un orquestador de contenedores

Ejemplo de Micro Servicios



The microservice landscape
[System boundary]

Definir un Micro servicio



Retos con Micro servicios

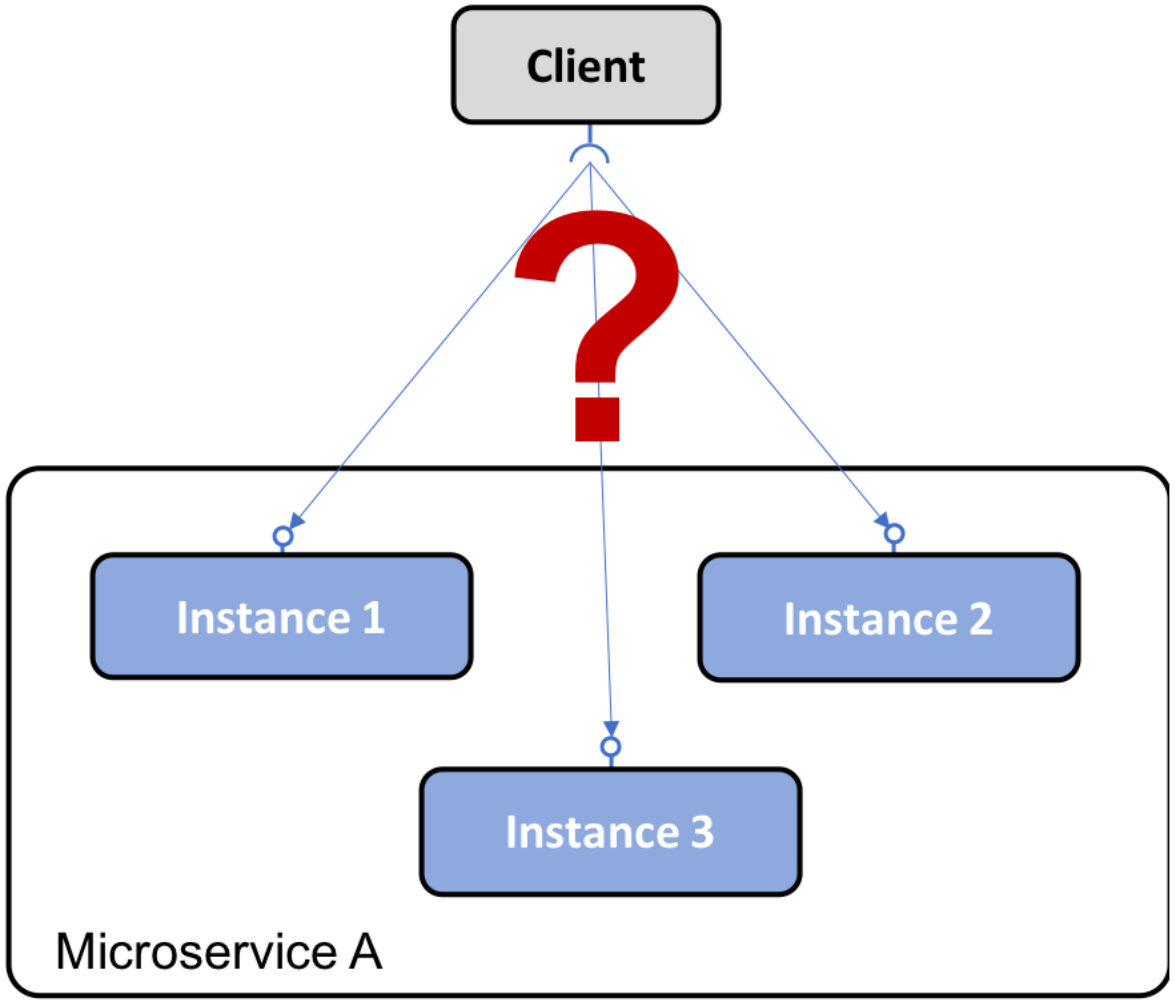
- **The 8 fallacies of distributed computing**
  1. La red es confiable
  2. Latencia es cero
  3. El ancho de banda es infinito
  4. La red es segura
  5. La topología no cambia
  6. No hay administrador
  7. El costo de transporte es cero
  8. La red es homogénea

Peter Deutsch, 1994

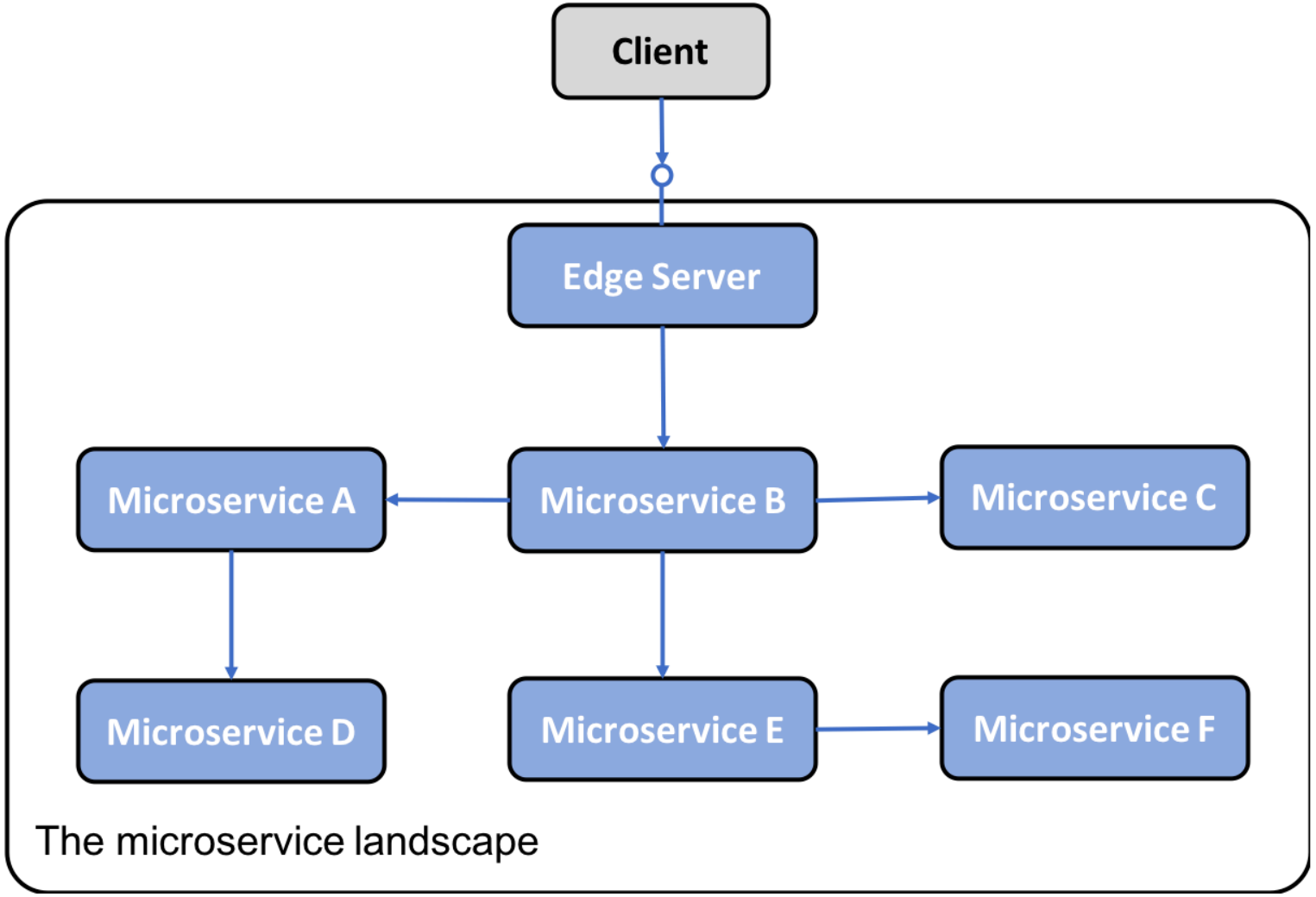
Patrones de Diseño para Micro servicios

- Service Discovery
- Edge server
- Reactive microservices
- Central configuration
- Centralized log analysis
- Distributed tracing
- Circuit breaker
- Control loop
- Centralized monitoring and alarms

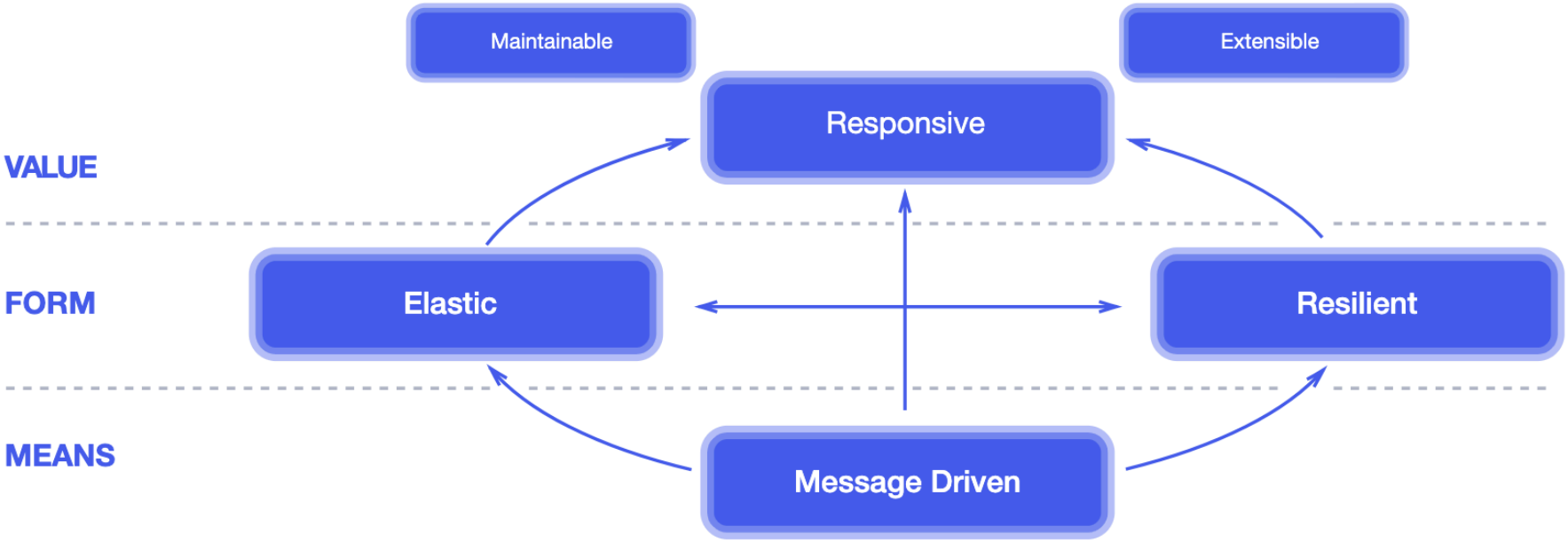
Service Discovery



Edge Server

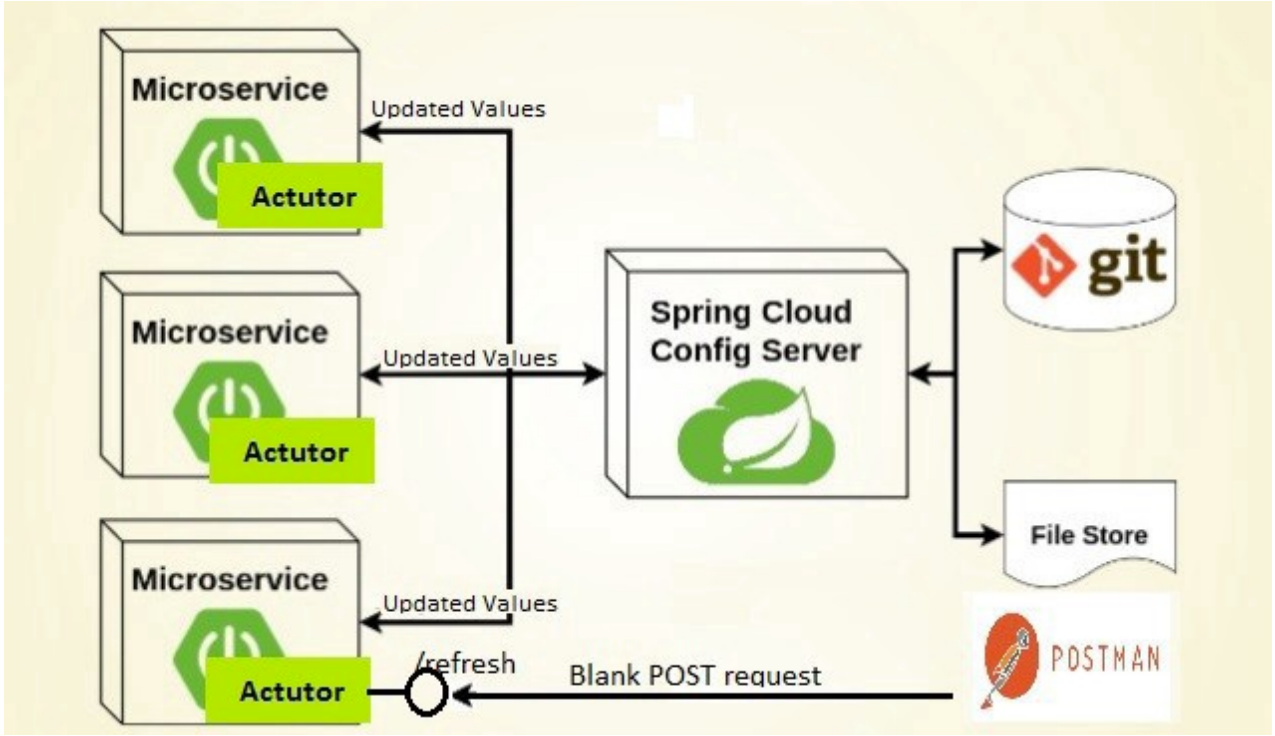

Client
Edge Server
Microservice A
Microservice B
Microservice C
Microservice D
Microservice E
Microservice F
The microservice landscape

Reactive microservice

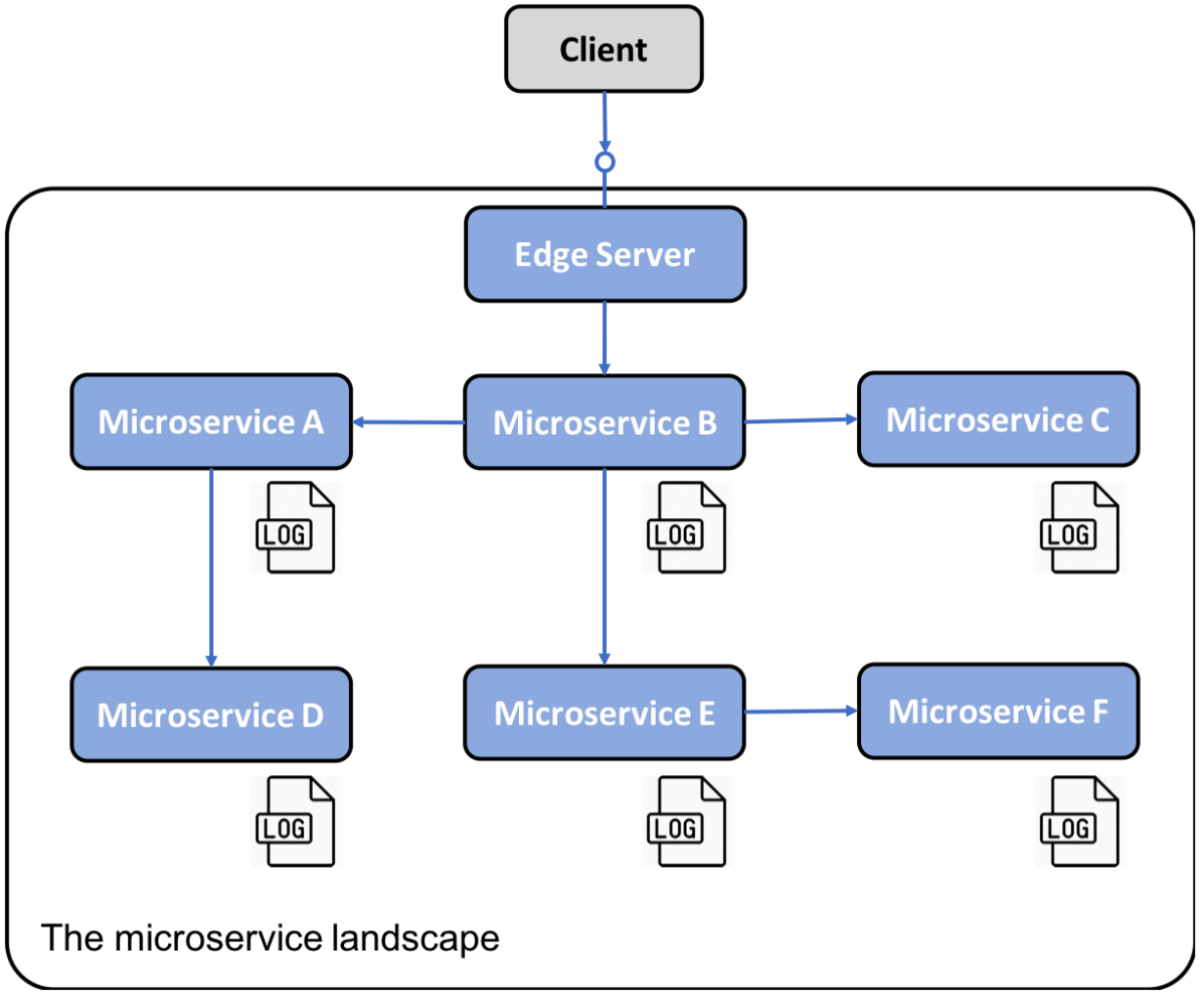


https://www.reactivemanifesto.org/

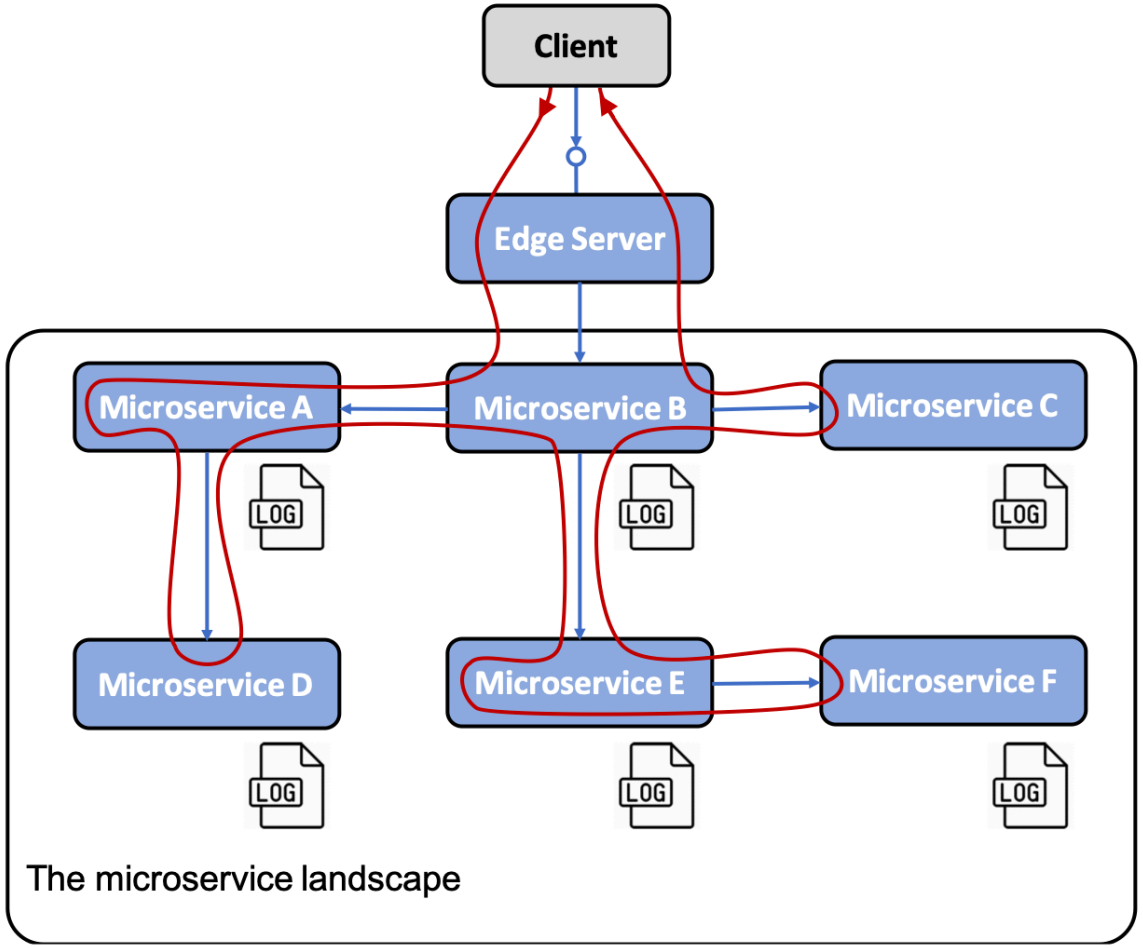
Central configuration



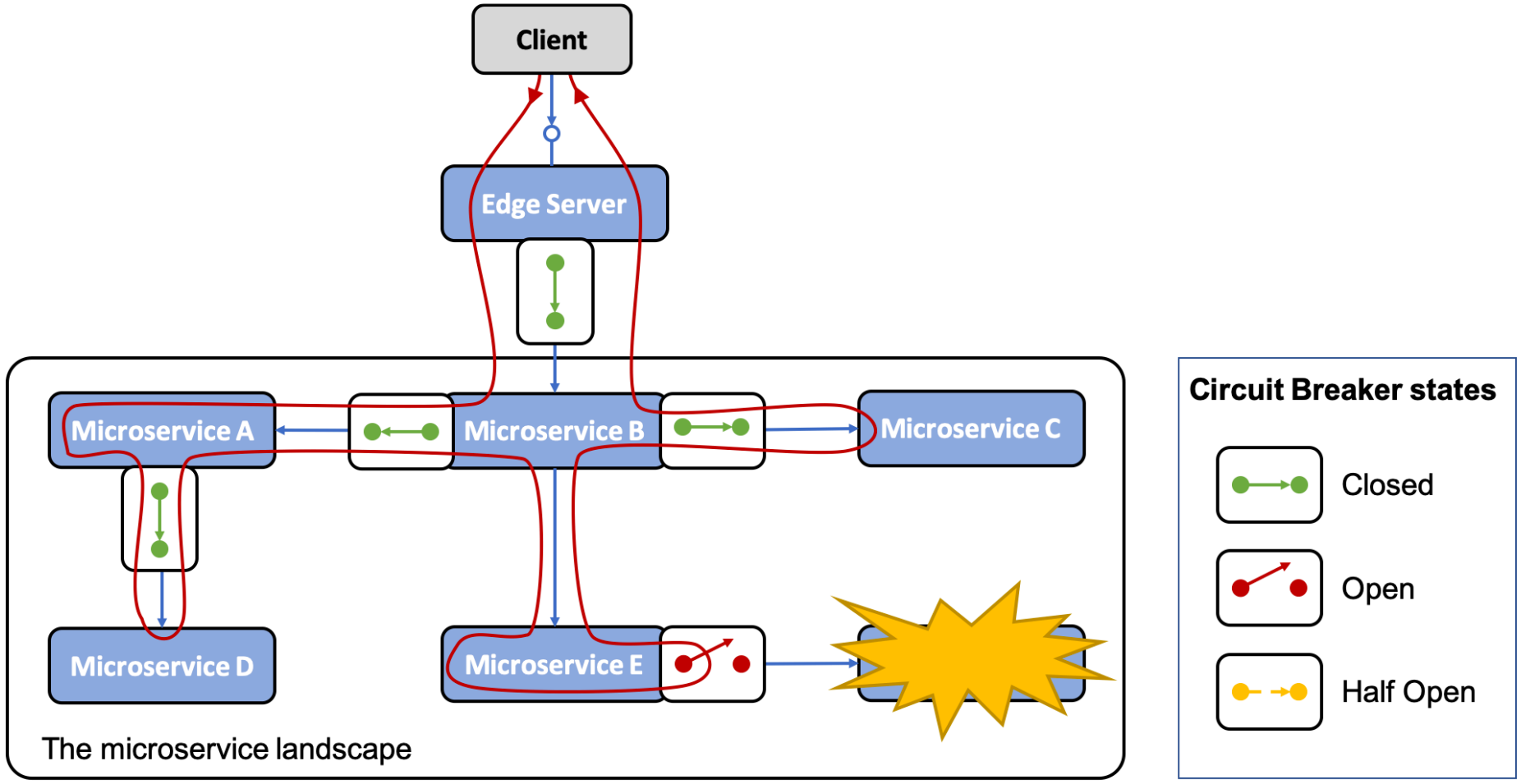
Centralized log analysis



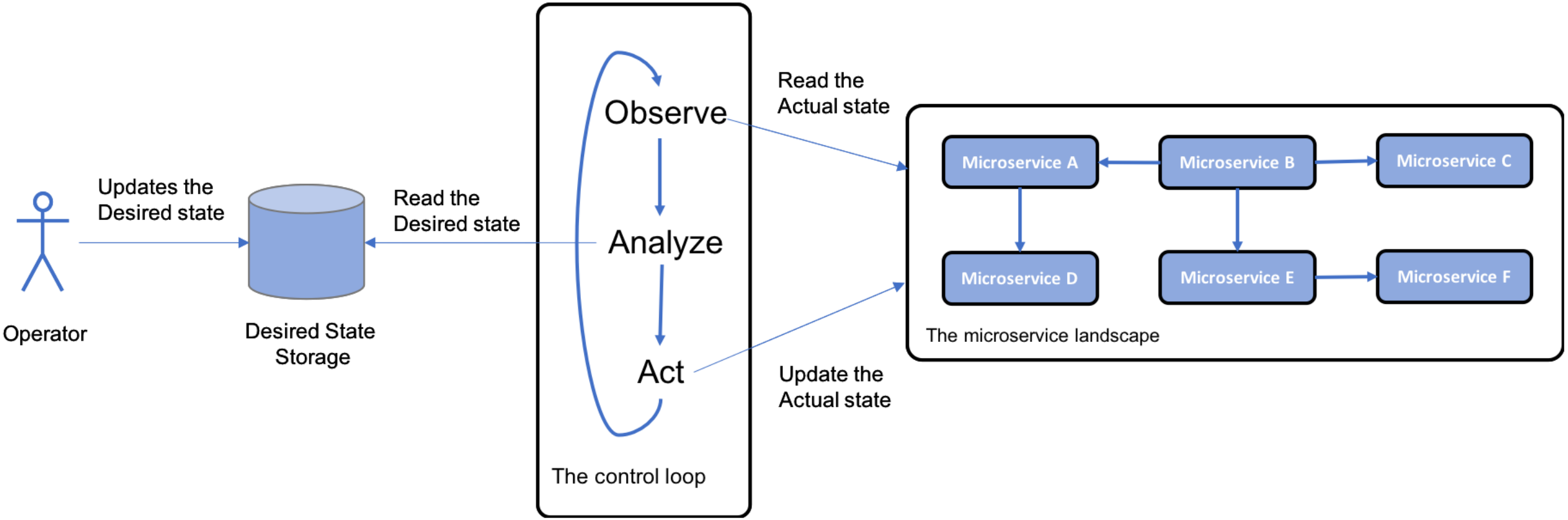
Distributed tracing



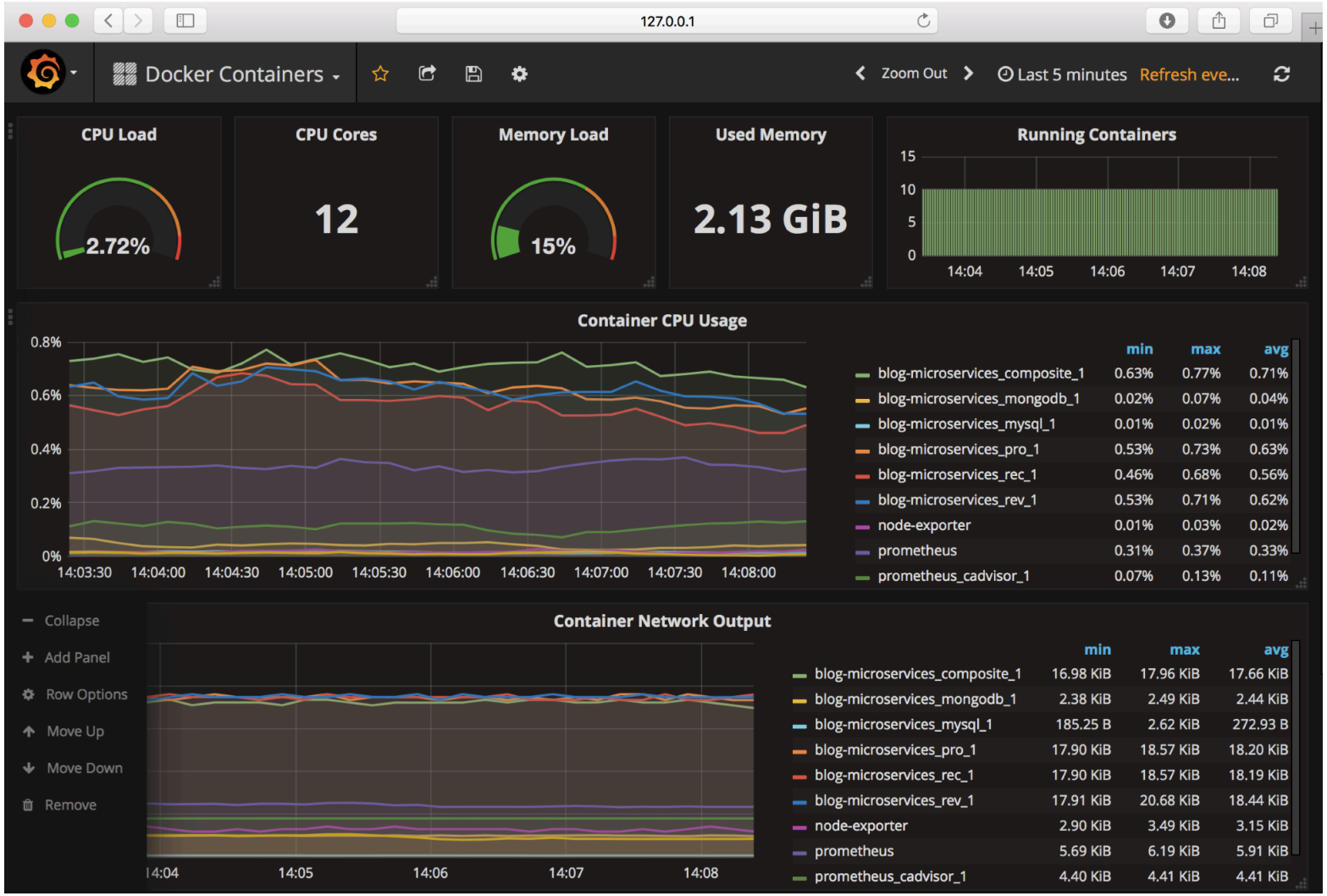
Circuit breaker



Control Loop



Centralized monitoring and alarms



Proveedores de Software

- Spring Boot
- Spring Cloud/Netflix OSS
- Docker
- Kubernetes
- Istio (a service mesh)

Design Pattern	Spring Boot	Spring Cloud	Kubernetes	Istio
Service discovery		Netflix Eureka and Netflix Ribbon	Kubernetes `kube-proxy` and service resources	
Edge server		Spring Cloud and Spring Security OAuth	Kubernetes Ingress controller	Istio ingress gateway
Reactive microservices	Spring Reactor and Spring WebFlux			
Central configuration		Spring Config Server	Kubernetes `ConfigMaps` and Secrets	
Centralized log analysis			Elasticsearch, Fluentd, and Kibana **Note**: Actually not part of Kubernetes but can easily be deployed and configured together with Kubernetes	

Distributed tracing		Spring Cloud Sleuth and Zipkin		Jaeger
Circuit Breaker		Resilience4j		Outlier detection
Control loop			Kubernetes controller manager	
Centralized monitoring and alarms			Grafana and Prometheus **Note:** Actually not part of Kubernetes but can easily be deployed and configured together with Kubernetes	Kiali, Grafana, and Prometheus

Otras importantes consideraciones

- Dev / Ops
- Aspectos Organizaciones y Ley de Conway

"Any organization that designs a system (defined broadly) will produce a design whose structure is a copy of the organization's communication structure."

-- Melvyn Conway, 1967

- Decomponenr una aplicación monolítica a micro servicios
- Importancia de diseño de APIs
- Migración de on-premise a cloud
- Principios de buen diseño de micro servicios, **el 12-factor app**