Accelerating ZOZOTOWN Modernization with Istio

Yoichi Kawasaki Tech Lead, ZOZOTOWN Platform SRE @ ZOZO, inc

Agenda

- Introduction
- ZOZOTOWN application modernization
  - Migration strategy
  - Istio adoption
- Recap

ZOZOTOWN

https://zozo.jp/

- The largest fashion online shopping website in Japan.
- Over 1,500 stores offering more than 8,400 brands.
- At any given time, more than 830,000 items are available for purchase, in addition of more than 2,900 new items (average) per day.(As of Dec. 31st, 2021)
- Operates ZOZOCOSME, a specialized cosmetics mall, ZOZOSHOES, a shoe-specialized mall, and ZOZOVILLA, a luxury & designer zone.
- Same day delivery services are available in limited areas within Japan.
- Gift-wrapping services.
- Deferred payment option, “Tsukebarai”.
- Since Dec. 15th, 2004

Download on the App Store　GET IT ON Google Play

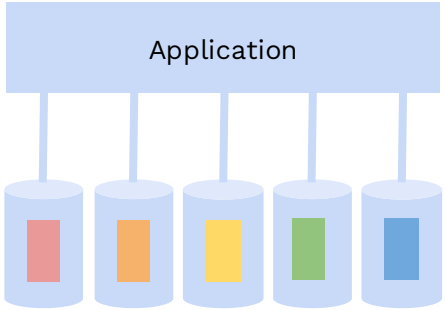
ZOZOTOWN

2004

ZOZOTOWN
was launched

Architecture	Monolith
Infrastructure	On Prem

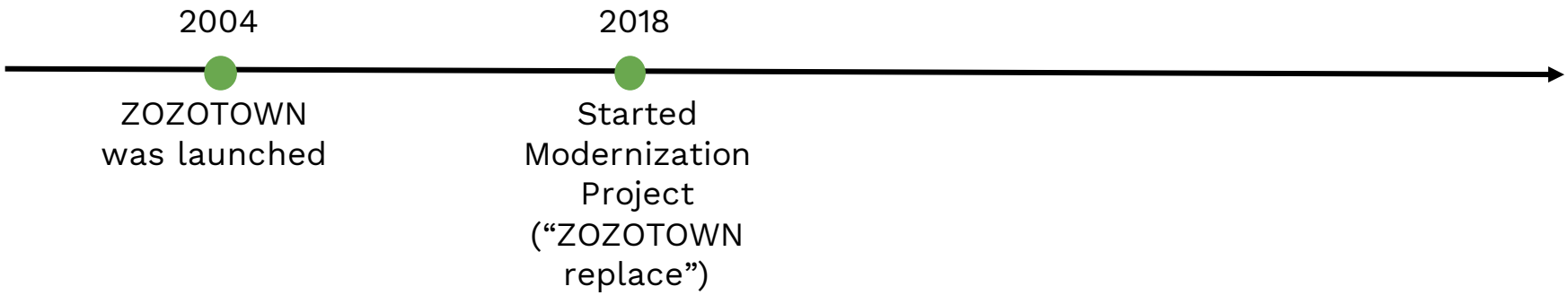
ID	UI	Search
Cart	Products	Session
Payment	Favorites	Membership



ASP Active Server Pages

Microsoft SQL Server

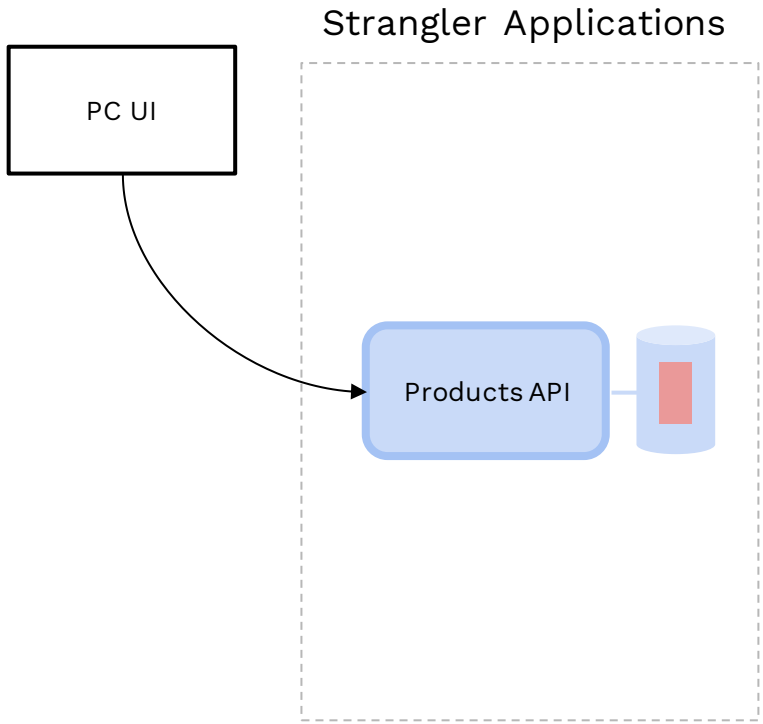
ZOZOTOWN modernization



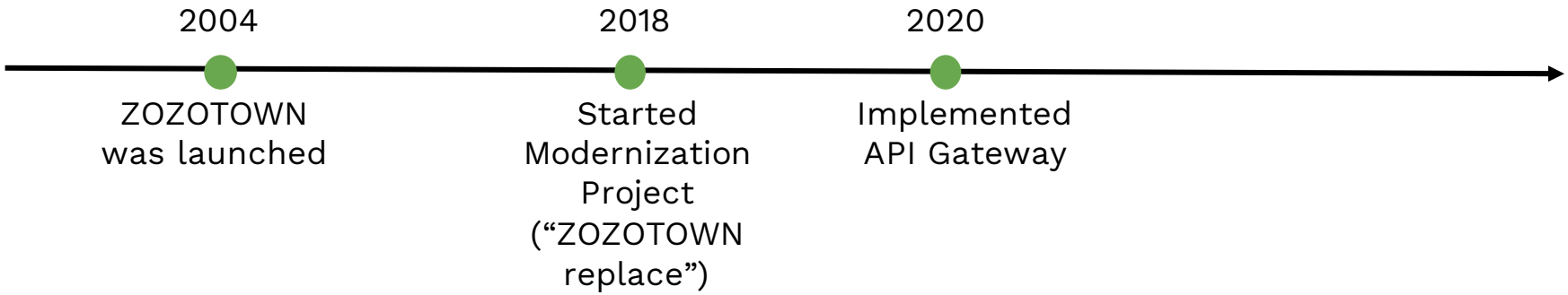
Architecture	Monolith	k8s based microservices
Infrastructure	On Prem	Hybrids (OnPrem + Public Cloud)

Strangling the monolith

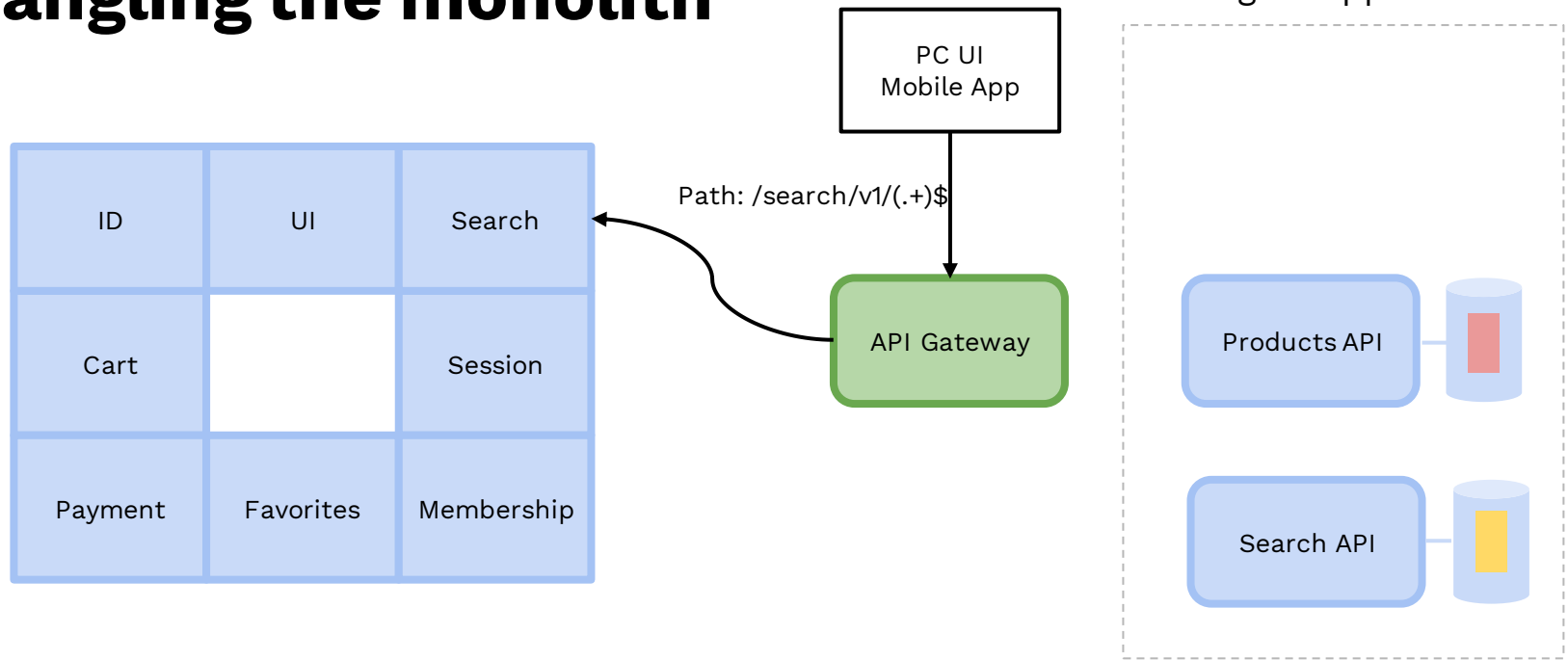
ID	UI	Search
Cart		Session
Payment	Favorites	Membership



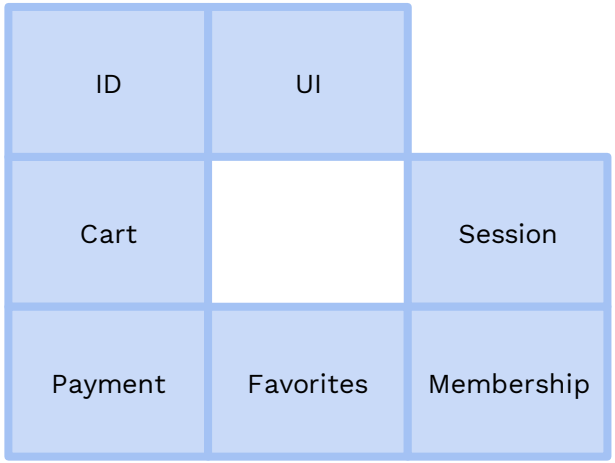
ZOZOTOWN modernization

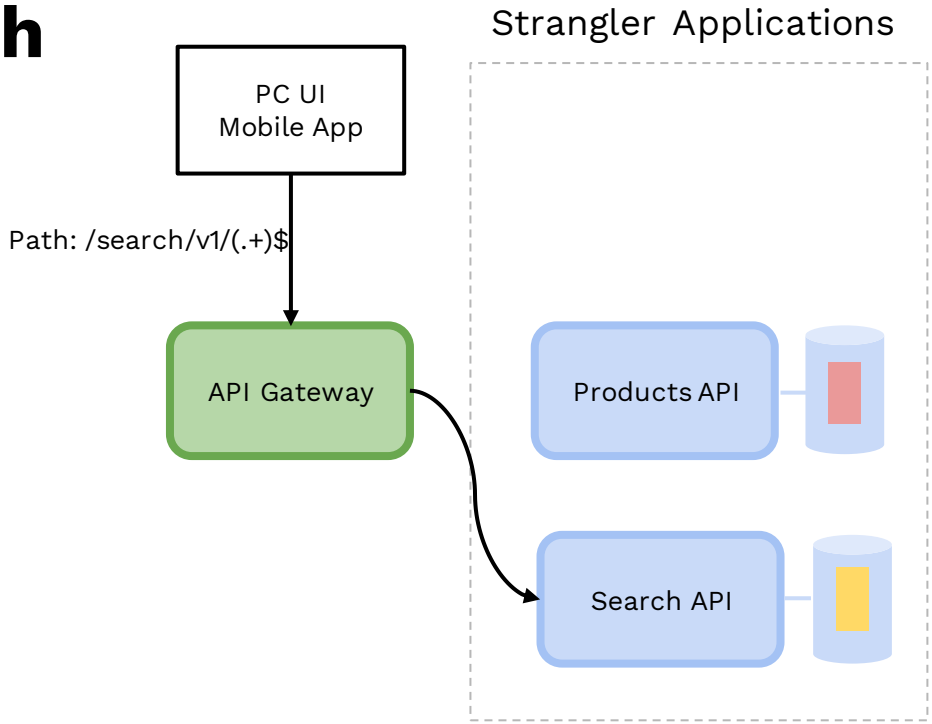


Strangling the monolith

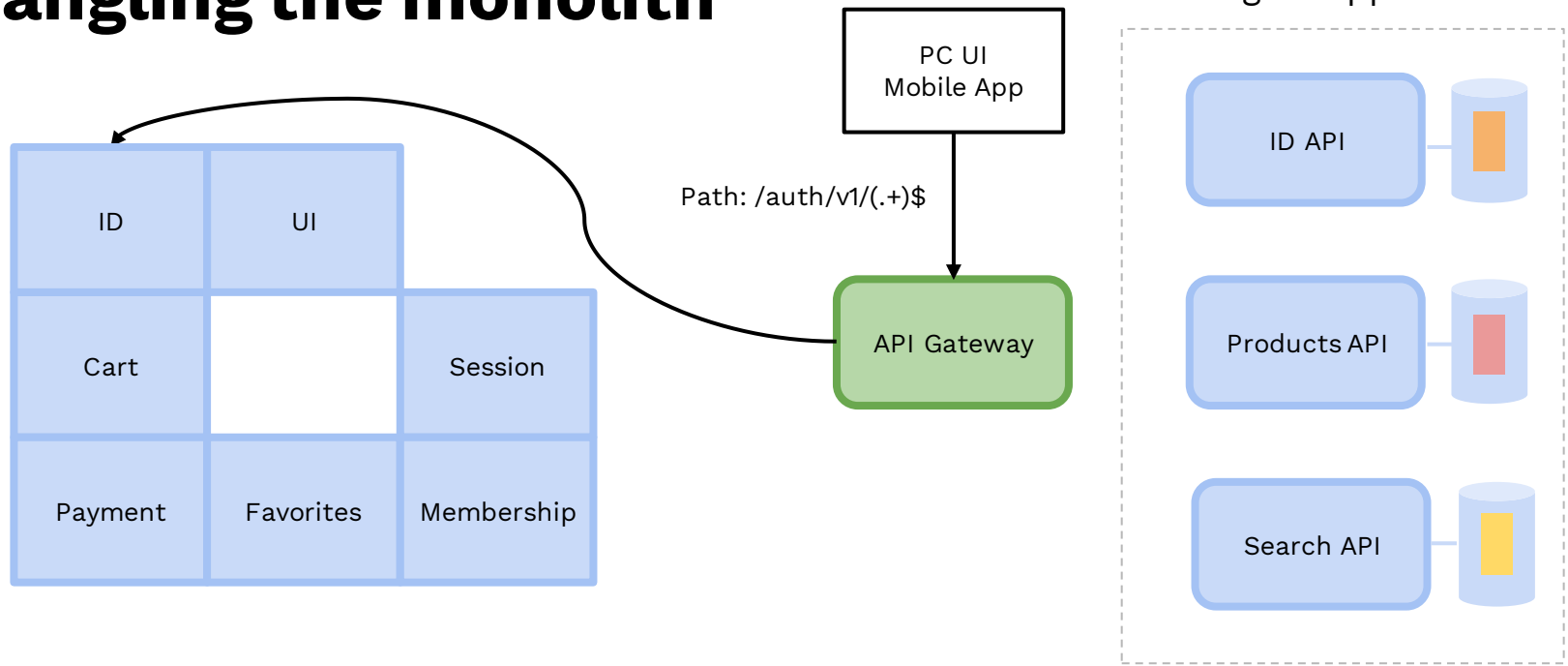


Strangling the monolith

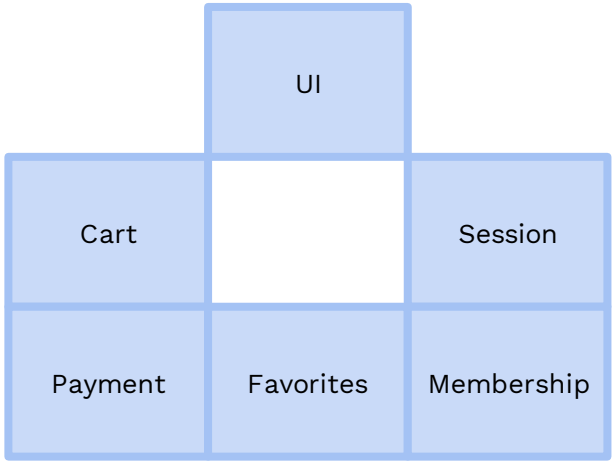


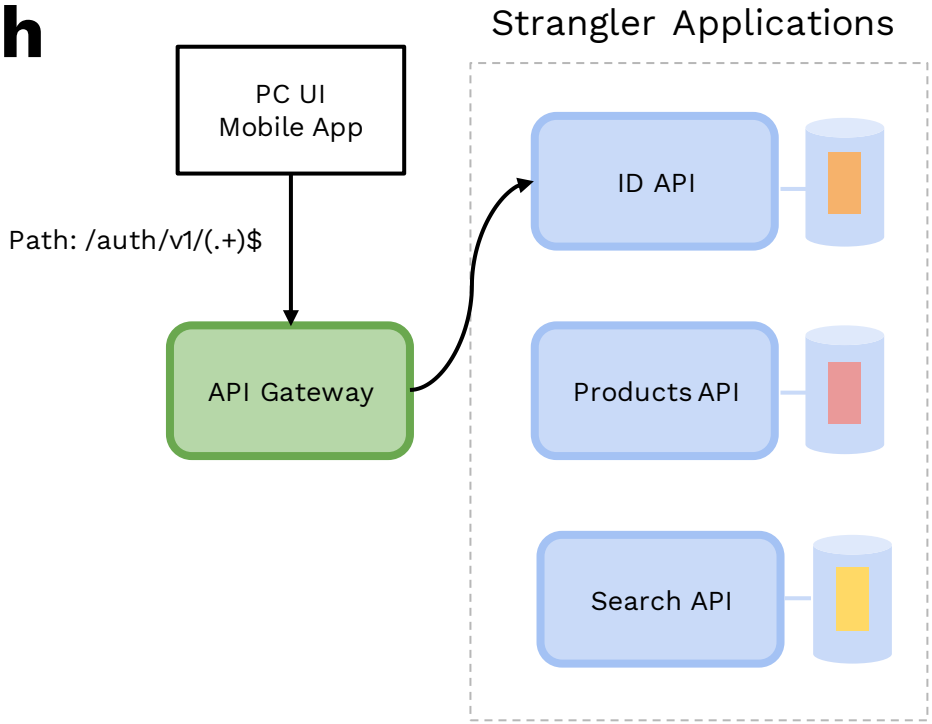


Strangling the monolith



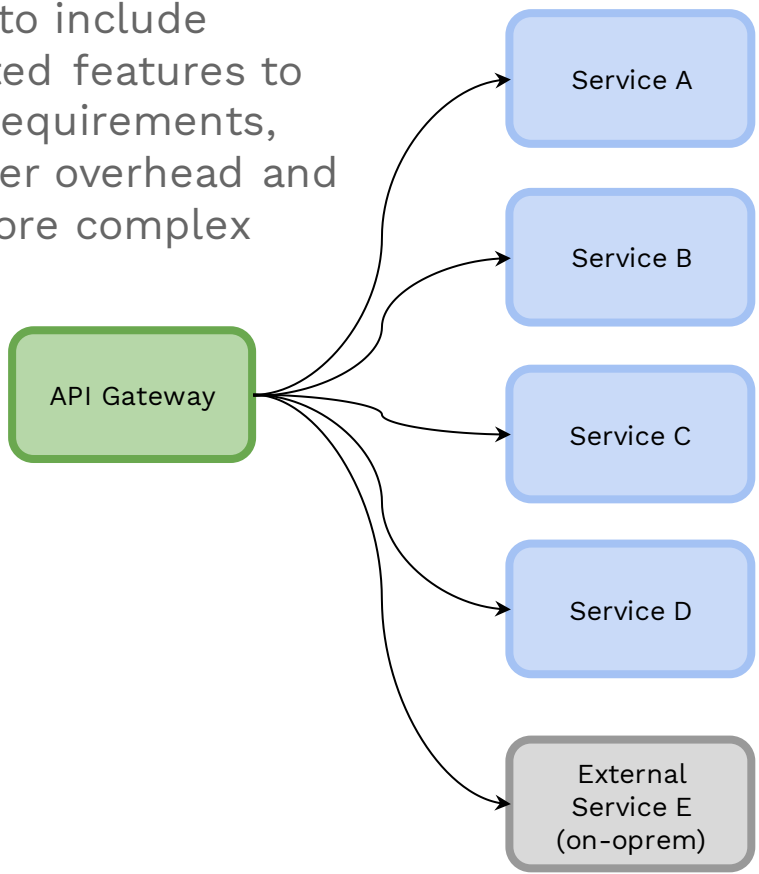
Strangling the monolith



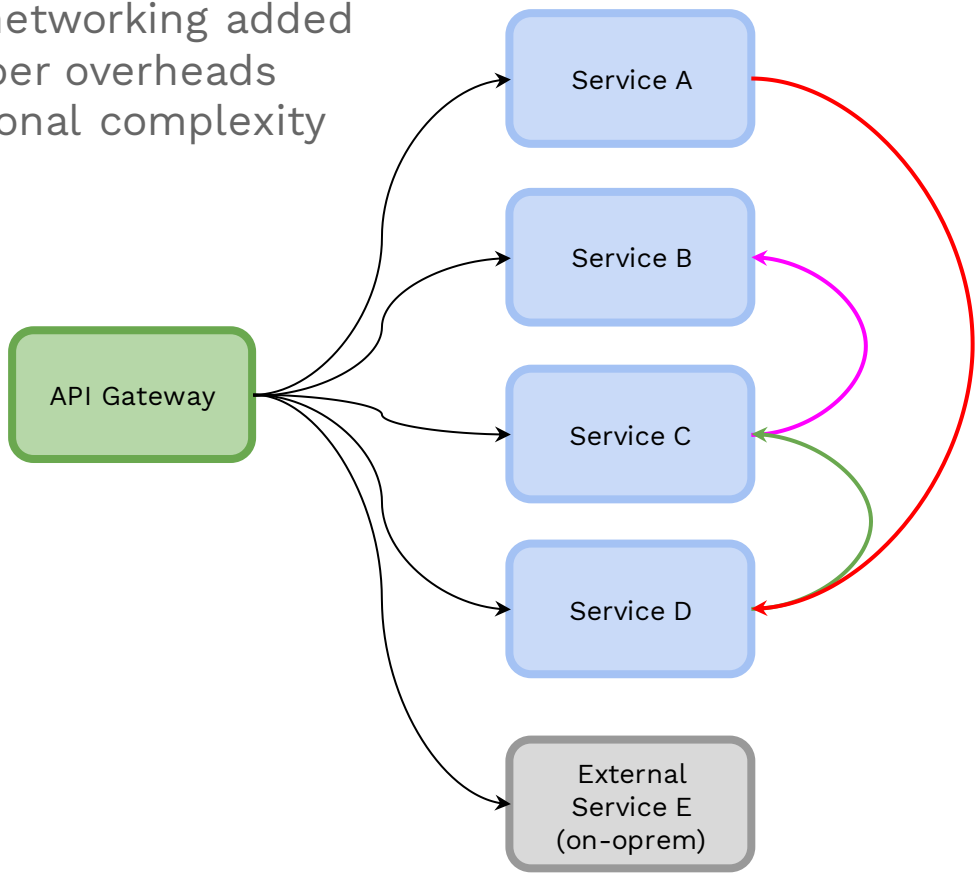


Increased operational complexity and overhead

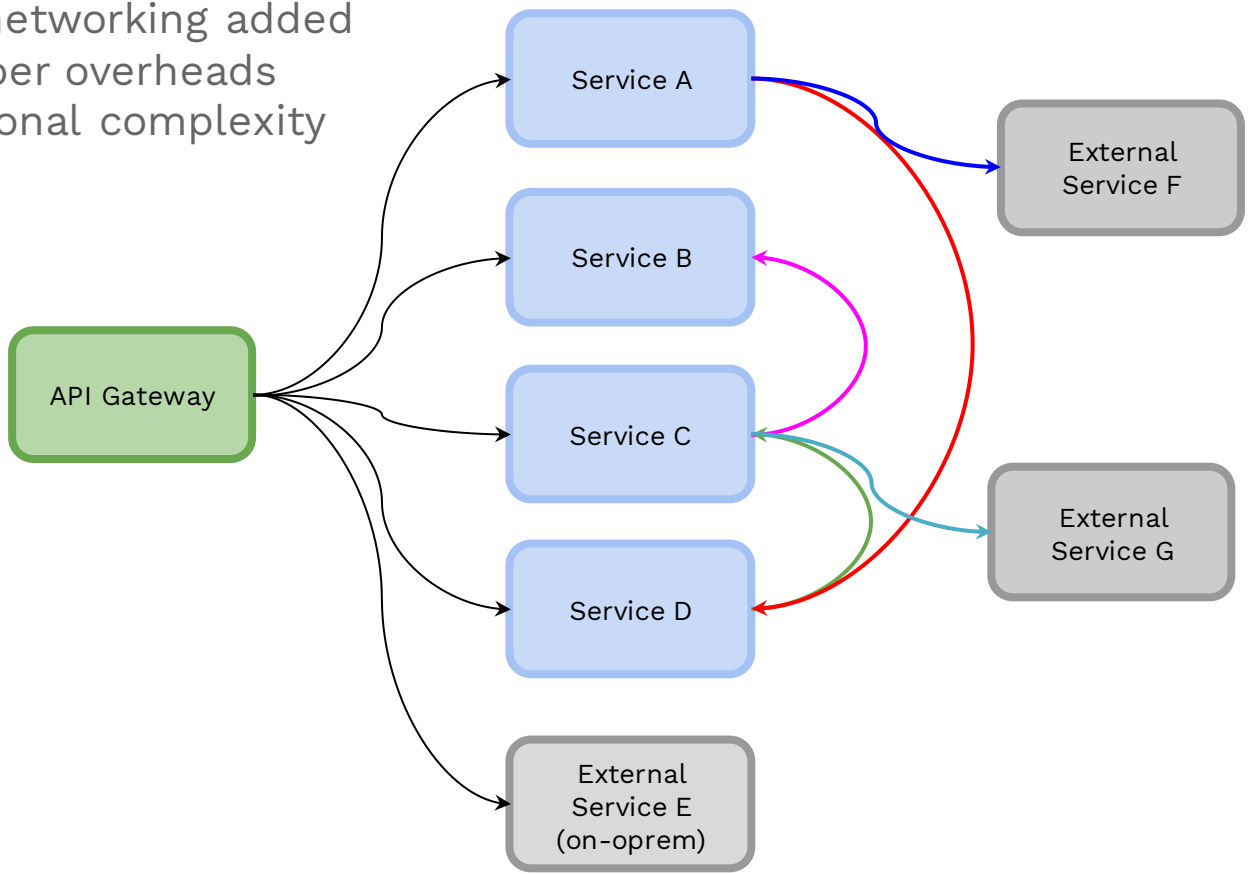
API Gateway needed to include various network-related features to meet each service's requirements, which added developer overhead and made the gateway more complex



Inconsistent service networking added both SRE and developer overheads and increases operational complexity

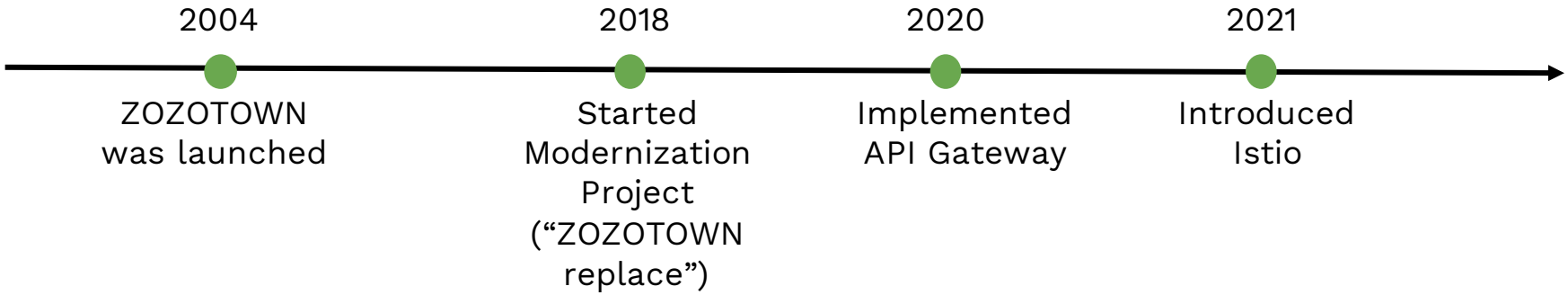


Inconsistent service networking added both SRE and developer overheads and increases operational complexity



Istio adoption

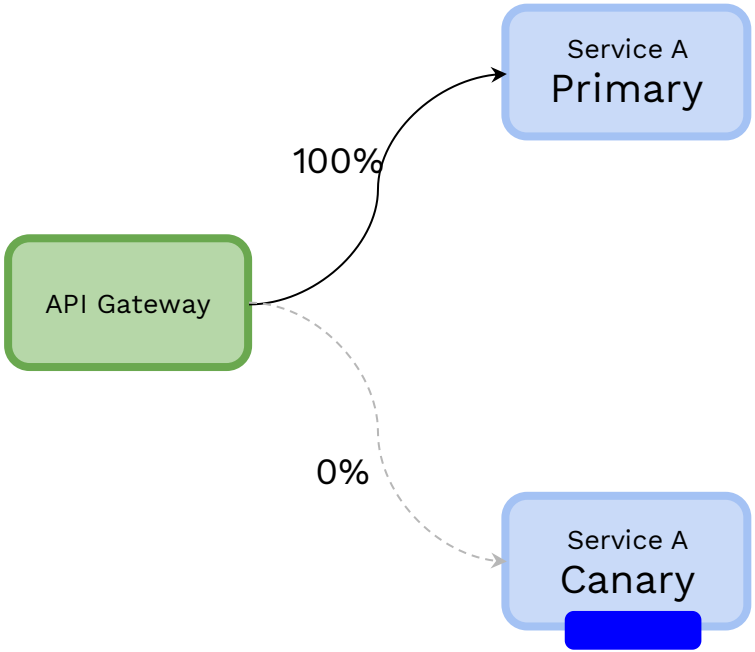
ZOZOTOWN modernization



Gradual introduction of istio

- One microservice at a time
- Zero downtime deployments using Canary deployment strategy
  - ZOZO API Gateway weighted routing for microservice Pods
  - AWS ALB weighted target groups for API Gateway Pods

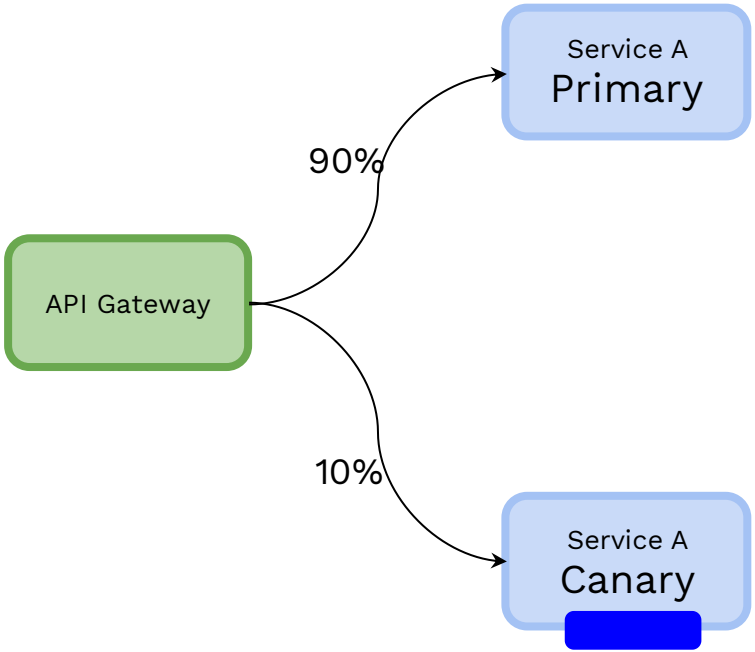
Enabling istio on service Pods



(Static) Canary deployment leveraging
ZOZO API Gateway weighted routing

```
zozo-service-a:
  targets:
    - host: service-a-primary.ns-a.svc.cluster.local
      port: 80
      weight: 100
  connect_timeout: 1000
  max_idle_conns_per_host: 1000
  max_try_count: 2
  retry_base_interval: 1
  retry_max_interval: 1
  read_timeout: 5000
  retry_cases:
    - redirection
    - client_error
    - server_error
    - timeout
  retry_non_idempotent: true
```

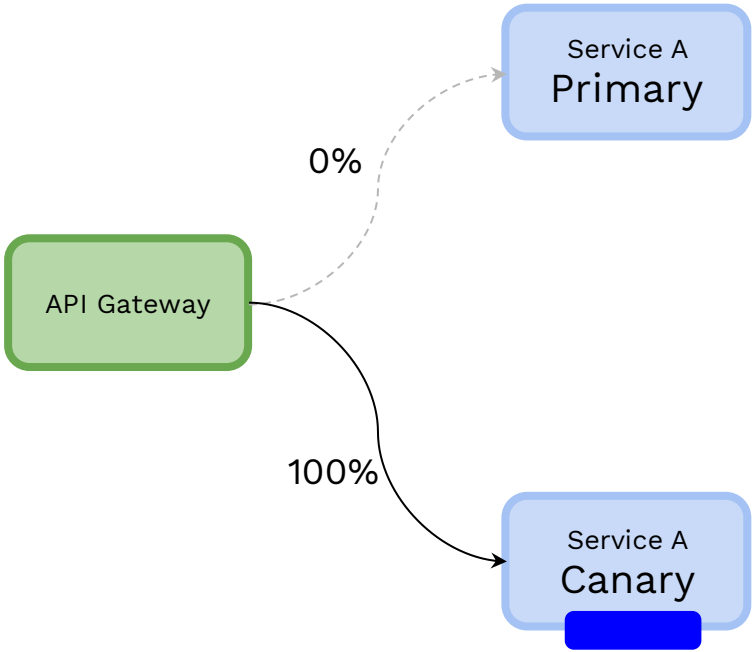
Enabling istio on service Pods



(Static) Canary deployment leveraging
ZOZO API Gateway weighted routing

```
zozo-service-a:
  targets:
    - id: primary
      host: service-a-primary.ns-a.svc.cluster.local
      port: 80
      weight: 90
    - id: canary
      host: service-a-canary.ns-a.svc.cluster.local
      port: 80
      weight: 10
  connect_timeout: 1000
  max_idle_conns_per_host: 1000
  max_try_count: 2
  retry_base_interval: 1
  retry_max_interval: 1
  read_timeout: 5000
  retry_cases:
    - redirection
    - client_error
    - server_error
    - timeout
  retry_non_idempotent: true
```

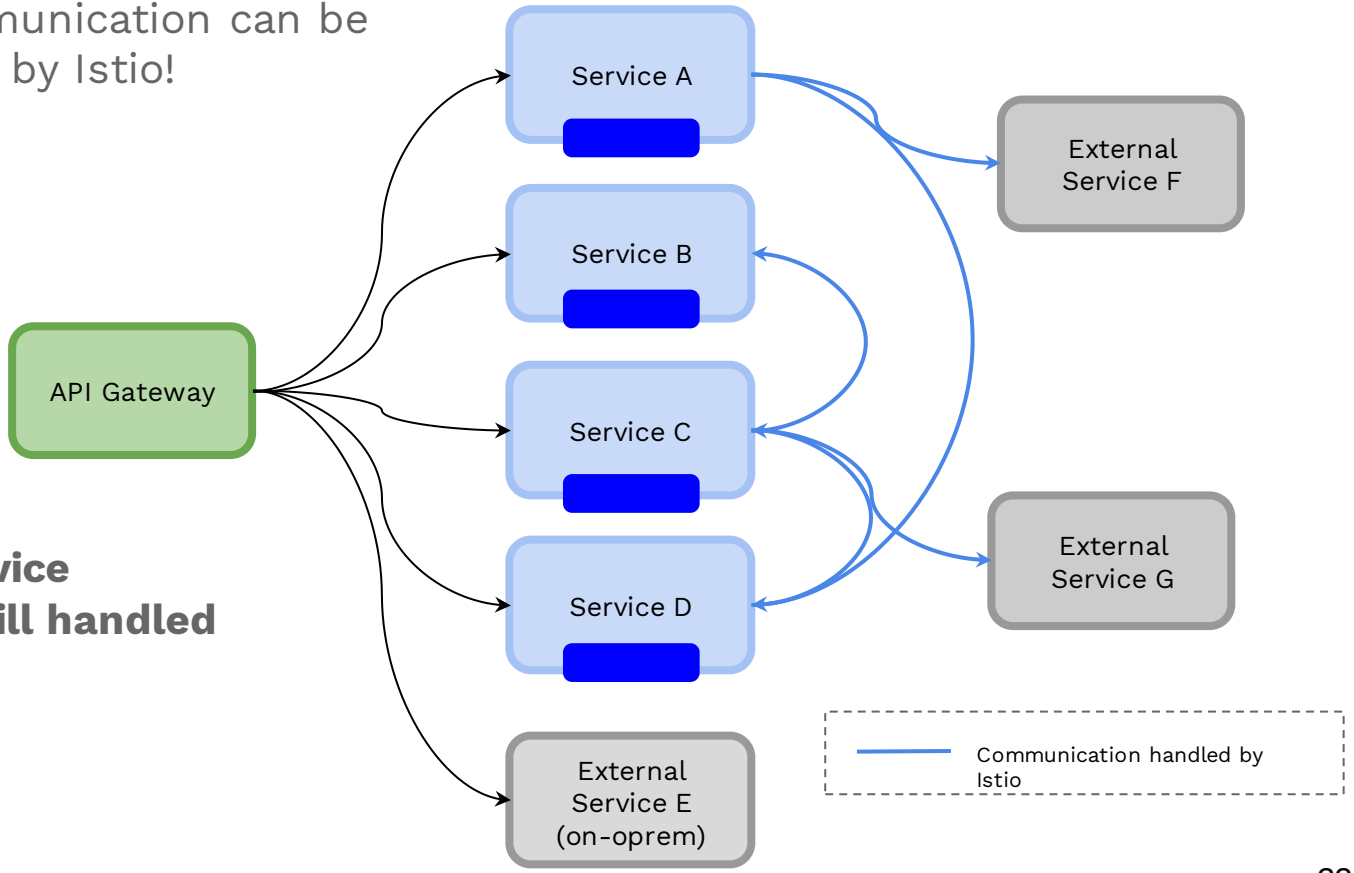
Enabling istio on service Pods



(Static) Canary deployment leveraging
ZOZO API Gateway weighted routing

```
zozo-service-a:
  targets:
    - host: service-a-canary.ns-a.svc.cluster.local
      port: 80
      weight: 100
  connect_timeout: 1000
  max_idle_conns_per_host: 1000
  max_try_count: 2
  retry_base_interval: 1
  retry_max_interval: 1
  read_timeout: 5000
  retry_cases:
    - redirection
    - client_error
    - server_error
    - timeout
  retry_non_idempotent: true
```

Now service-to-service and service-to-external service communication can be consistently handled by Istio!

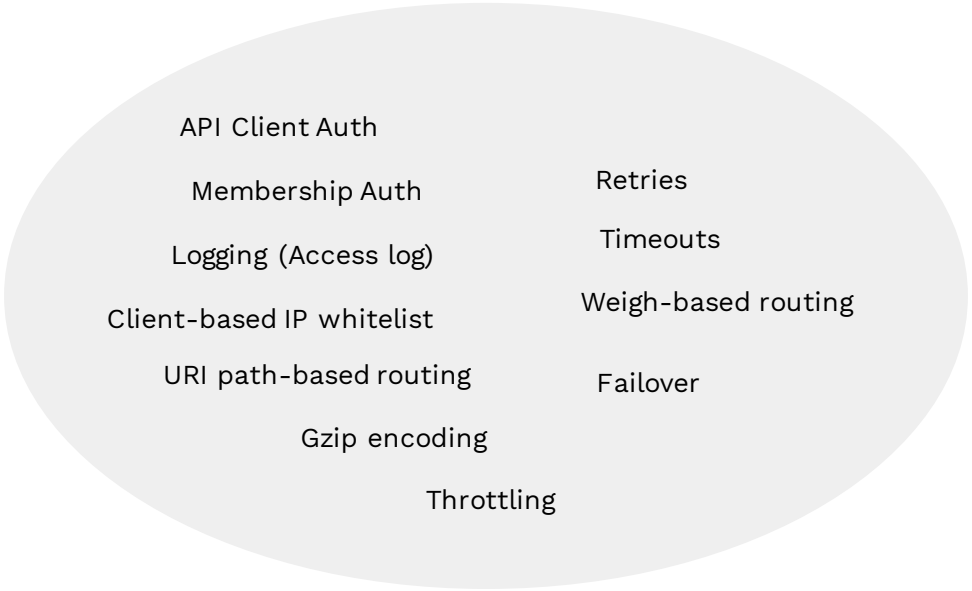


But gateway-to-service communication is still handled differently

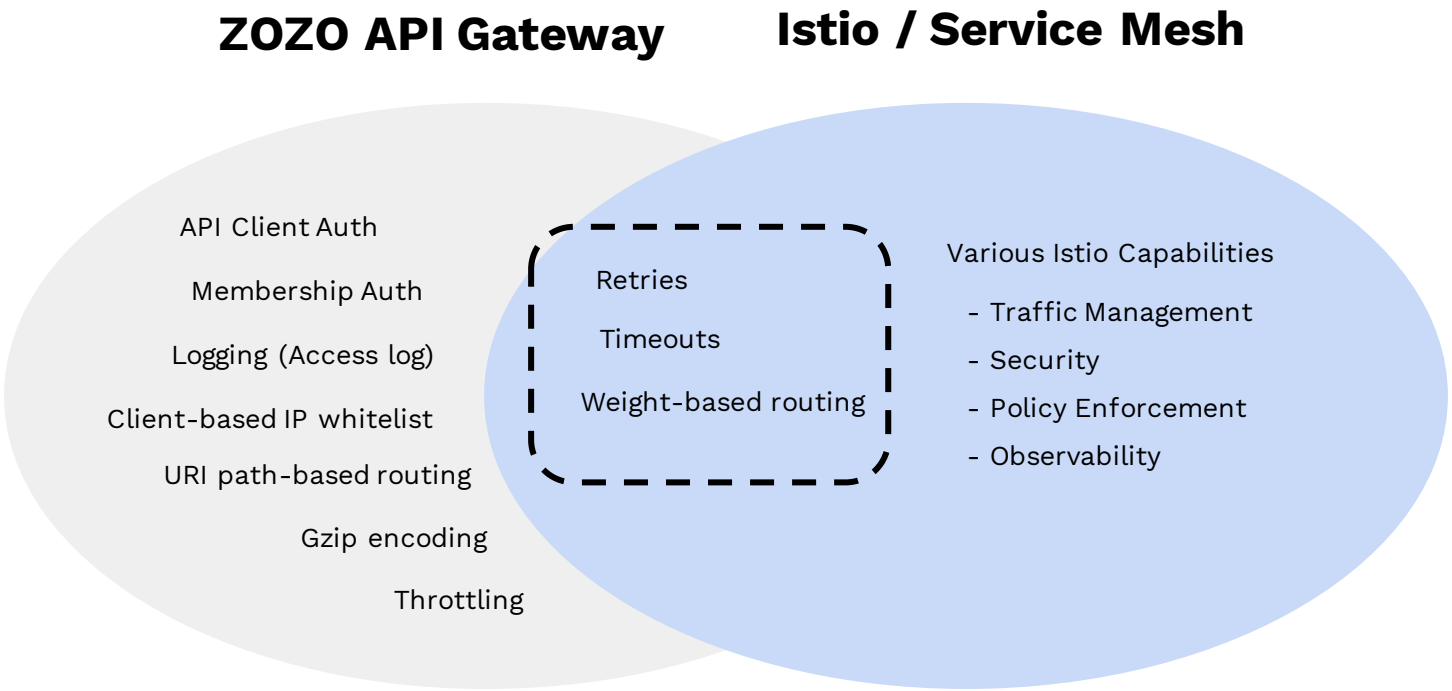
Refactoring of ZOZO API Gateway to fit in with Istio

ZOZO API Gateway's Original Features

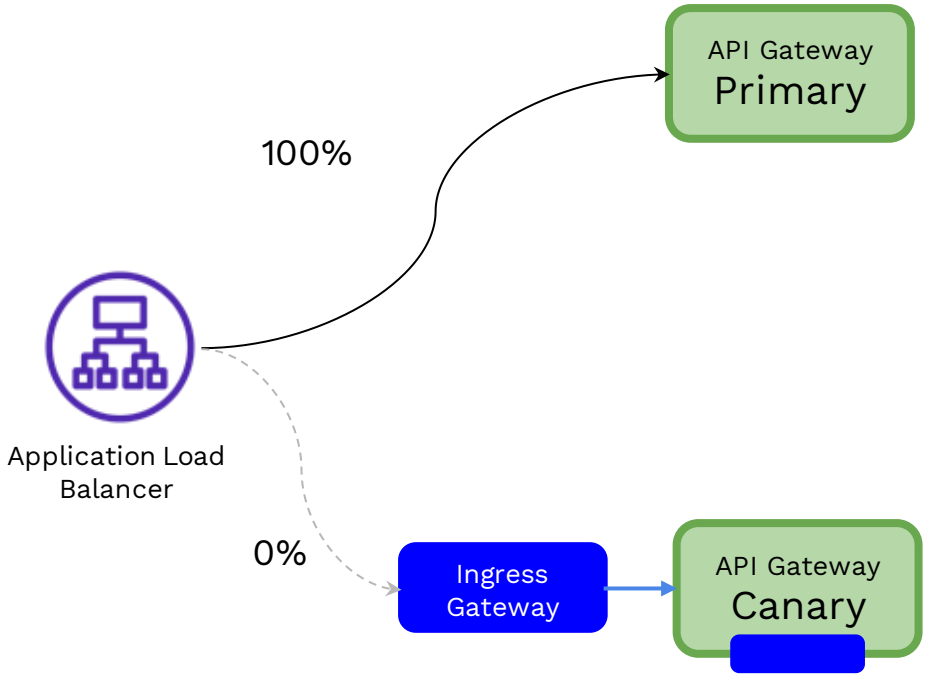
ZOZO API Gateway = ZOZO's Go-based in-house API Gateway



Refactoring ZOZO API Gateway to fit in with Istio



Enabling istio on API Gateway Pods

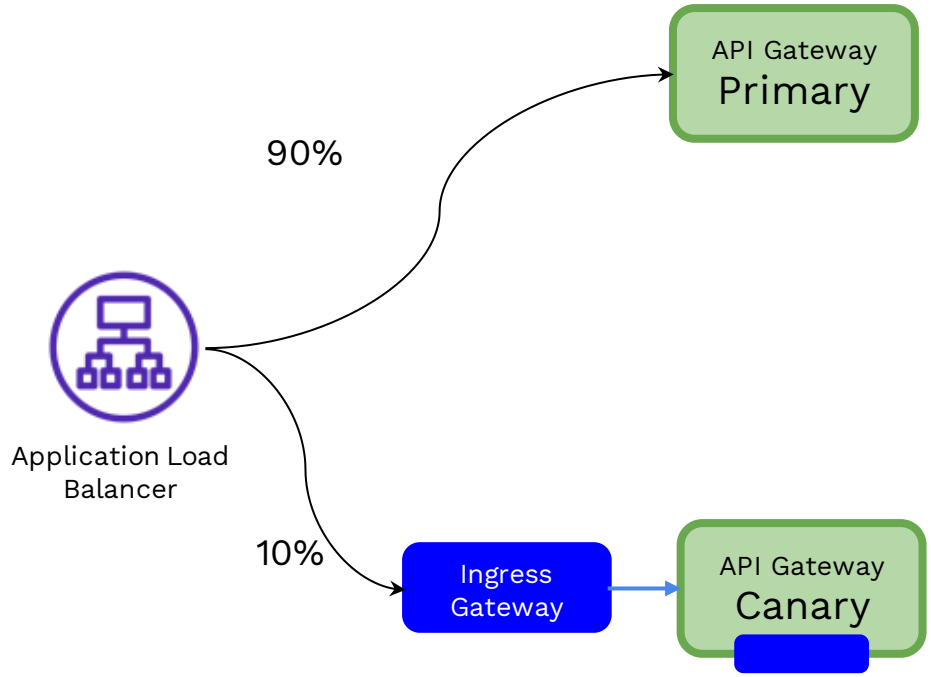


(Static) Canary deployment leveraging
AWS ALB weighted target groups

```yaml
apiVersion: networking.k8s.io/v1beta1
kind: Ingress
metadata:
  name: zozo-api-gateway-ingresss
  annotations:
    kubernetes.io/ingress.class: alb
    alb.ingress.kubernetes.io/target-type: ip
    alb.ingress.kubernetes.io/scheme: internet-facing
    alb.ingress.kubernetes.io/actions.forward-external-traffic: |
      {
        "Type":"forward",
        "ForwardConfig":{
          "TargetGroups":[
            {
                "ServiceName":"zozo-api-gateway-primary",
                "ServicePort":"80",
                "Weight":100
            },
            {
                "ServiceName":"zozo-api-gateway-istio-ingressgateway",
                "ServicePort":"80"
                "Weight":0
            }
          ]
        }
      }
    # ... snip ...
spec:
  rules:
    - http:
        paths:
          - path: /*
            backend:
              serviceName: forward-external-traffic
              servicePort: use-annotation
```

Configure ALB with AWS Load Balancer Controller
https://kubernetes-sigs.github.io/aws-load-balancer-controller/

Enabling istio on API Gateway Pods

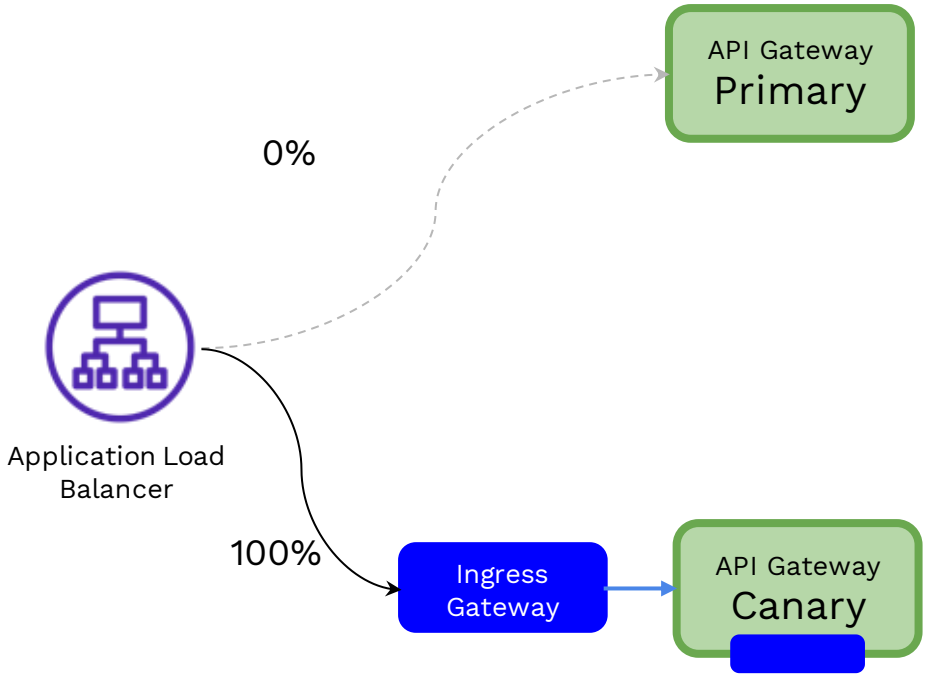


(Static) Canary deployment leveraging
AWS ALB weighted target groups

```yaml
apiVersion: networking.k8s.io/v1beta1
kind: Ingress
metadata:
  name: zozo-api-gateway-ingresss
  annotations:
    kubernetes.io/ingress.class: alb
    alb.ingress.kubernetes.io/target-type: ip
    alb.ingress.kubernetes.io/scheme: internet-facing
    alb.ingress.kubernetes.io/actions.forward-external-traffic: |
      {
        "Type":"forward",
        "ForwardConfig":{
          "TargetGroups":[
            {
                "ServiceName":"zozo-api-gateway-primary",
                "ServicePort":"80",
                "Weight":90
            },
            {
                "ServiceName":"zozo-api-gateway-istio-ingressgateway",
                "ServicePort":"80"
                "Weight":10
            }
          ]
        }
      }
    # ... snip ...
spec:
  rules:
    - http:
        paths:
          - path: /*
            backend:
              serviceName: forward-external-traffic
              servicePort: use-annotation
```

Configure ALB with AWS Load Balancer Controller
https://kubernetes-sigs.github.io/aws-load-balancer-controller/

Enabling istio on API Gateway Pods

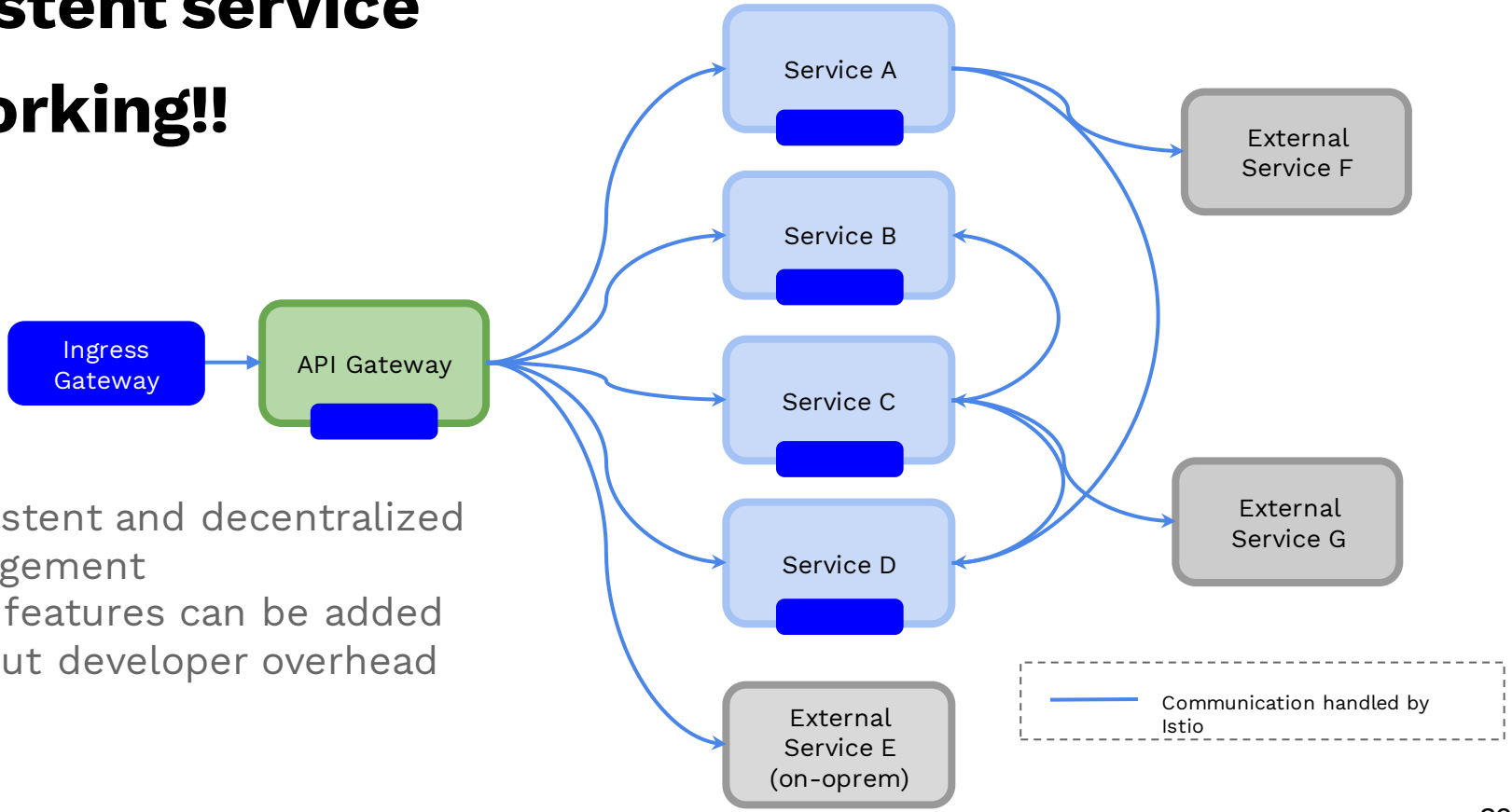


(Static) Canary deployment leveraging
AWS ALB weighted target groups

```yaml
apiVersion: networking.k8s.io/v1beta1
kind: Ingress
metadata:
  name: zozo-api-gateway-ingresss
  annotations:
    kubernetes.io/ingress.class: alb
    alb.ingress.kubernetes.io/target-type: ip
    alb.ingress.kubernetes.io/scheme: internet-facing
    alb.ingress.kubernetes.io/actions.forward-external-traffic: |
      {
        "Type":"forward",
        "ForwardConfig":{
          "TargetGroups":[
            {
                "ServiceName":"zozo-api-gateway-primary",
                "ServicePort":"80",
                "Weight":0
            },
            {
                "ServiceName":"zozo-api-gateway-istio-ingressgateway",
                "ServicePort":"80"
                "Weight":100
            }
          ]
        }
      }
    # ... snip ...
spec:
  rules:
    - http:
        paths:
          - path: /*
            backend:
              serviceName: forward-external-traffic
              servicePort: use-annotation
```

Configure ALB with AWS Load Balancer Controller
https://kubernetes-sigs.github.io/aws-load-balancer-controller/

Consistent service networking!!



- Consistent and decentralized management
- More features can be added without developer overhead

Consistent service networking!!

VirtualService

```yaml
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: zozo-service-api
spec:
  hosts:
  - zozo-service-api.ns-b.svc.cluster.local
  http:
  - name: default
    route:
    - destination:
        host: zozo-service-api.ns-b.svc.cluster.local
        subset: zozo-service-api-primary
      weight: 100
    - destination:
        host: zozo-service-api.ns-b.svc.cluster.local
        subset: zozo-service-api-canary
      weight: 0
    retries:
      attempts: 1
      perTryTimeout: 5.5s
      retryOn: 5xx
    timeout: 11s
```

DestinationRule

```yaml
apiVersion: networking.istio.io/v1alpha3
kind: DestinationRule
metadata:
  name: zozo-service-api
spec:
  host: zozo-service-api.ns-b.svc.cluster.local
  trafficPolicy:
    connectionPool:
      tcp:
        maxConnections: 10
      http:
        http1MaxPendingRequests: 10
        http2MaxRequests: 100
        maxRequestsPerConnection: 2
    outlierDetection:
      consecutiveErrors: 10
      interval: 1m
      baseEjectionTime: 3m
      maxEjectionPercent: 100
  subsets:
  - name: zozo-service-api-primary
    labels:
      version: zozo-service-api-primary
  - name: zozo-service-api-canary
    labels:
      version: zozo-service-api-canary
```

Further Istio adoption for better resiliency and DevOps experiences

- Dynamic canary release deployment (Progressive Delivery)
- Further refactoring of ZOZO API Gateway for better fit in with Istio
- More automated and safer istio upgrade
- Expand service mesh across multiple k8s clusters

Recap

- Gradual migration from monolith to microservices with strangler application pattern
- Increased operational complexity and overhead
- Istio adoption
  - Refactoring of ZOZO API gateway to fit in with Istio
  - Gradual introduction of istio with static canary release deployment
- Further istio adoption for better resiliency and DevOps experiences

Thank you!

Yoichi Kawasaki
@yokawasa

#IstioCon


ZOZO

Template - Title

Template - Add your content

You can use other template/format for inner slides, just please use the cover and closing (thank you) slides that we provide.

Sample slide

Add your content

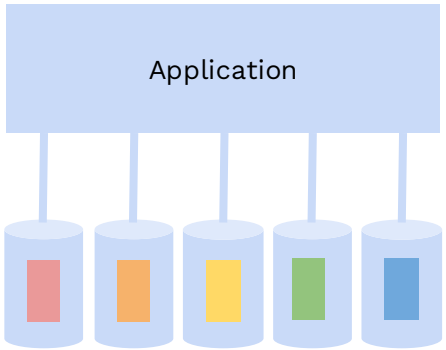
You can use other template/format for inner slides, just please use the cover and closing (thank you) slides that we provide.

icons

https://aws.amazon.com/jp/architecture/icons/

Monolith

ID	UI (pc/mobile)	Search
Cart	Products	Session
Payment	Favorites	Membership



on-premises

ASP Active Server Pages

Microsoft SQL Server