Argo Rollouts

How Intuit Does Canary and Blue Green Deployments with a k8s Controller

Overview

- Intuit background
- Problem
- Options
- Final solution
- Demo!
- What's next

Who are we

1983	1993	4,000	21	$6B FY18	50M
Founded	IPO	Developers	Locations	Revenue	Customers

Kubernetes at Intuit (1 year)

Cumulative Applications Over Time
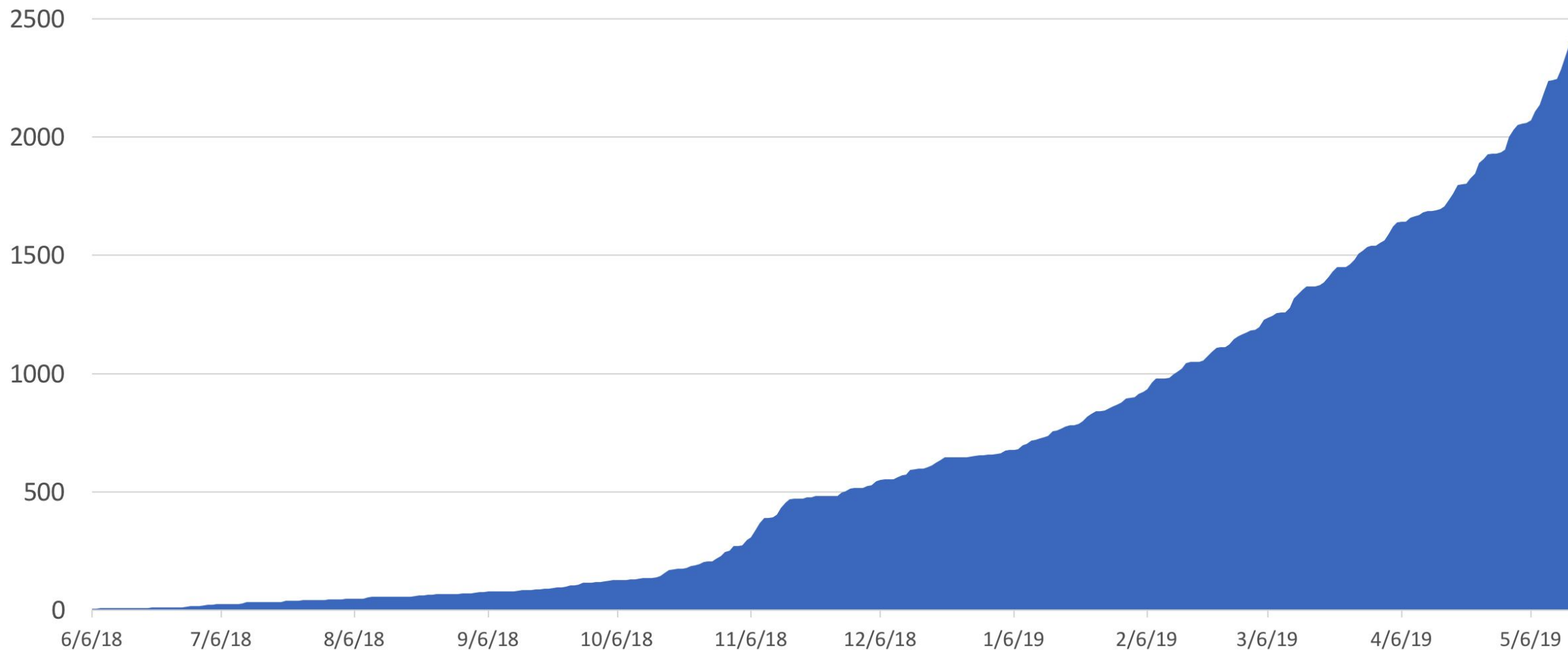
Intuit Developer Platform (Modern SaaS)

Application Services

| Intuit API (v4) | JSK + Config + Experimentation | Dev Patterns | Argo workflows | Streaming/ Messaging | Serverless Framework | UX Fabric |

Developer and Operations Experience

| Discover | Learn/Play | Onboarding | Monitoring | Management | Multi-Cluster Mgmt (IKSM) |

| GitHub (Apps as Code) | IBP 2.0 Jenkins (Build & Test - CI/d) | Quality Frameworks (TDS, Overwatch, TrinityJS, Hubble…) | JFrog Artifactory (CDP) | Argo CD (GitOps) | Metrics/Analytics (Team Speed Dashboards) |

| Multi-Cluster Service Mesh and Gateway — envoy, AWS App Mesh | Service Catalog — aws |

Intuit Kubernetes Service (IKS)
(Core Kubernetes with Intuit Network & Security policies & best practices)

- Continuous Operations (Monitoring, Analytics, Remediation)
- Security & Compliance — Signal Sciences
- Kops
- EKS

AWS Infrastructure — VPC, ALB/NLB, S3, RDS, DynamoDB, Elasticache, ... — aws

Sidebar

- **Olympus** (SSO & AWS Roles)
- **IDPS** (Secrets)
- **Splunk** (Logging)
- **Wavefront** (Monitoring)
- **Appdynamics** (Monitoring)
- **PagerDuty** (Alerts)
- **NetGenie** (Certs))
- **ServiceNow** (CM)

intuit

Some Intuit Statistics

Intuit

- 4 business units
- 30 business segments
- 1,200+ developers using Kubernetes

Kubernetes

- 120+ clusters (Intuit managed)
- 3,200 nodes
- 25,000 cpu cores
- 2,400 namespaces
- 27,000 pods
- **1,300 deploys a day**

K8S Deployment Strategies

✓ ## Recreate
Version A is terminated then version B is rolled out.

✓ ## Rolling Update
Version B is slowly rolled out and replacing version A.

? ## Blue/Green
Version B is released alongside version A, then the traffic is switched to version B.

? ## Canary
Version B is released to a subset of users, then proceed to a full rollout.

? ## A/B testing
Version B is released to a subset of users under specific condition.

GitOps

What:
- Git as single source of truth of the desired state of your infrastructure and apps

Why:
- Observability
- Auditability & compliance
- Consistent rollback process
- Facilitates disaster recovery
- Developer-centric
- Perfect fit for Kubernetes declarative manifests

How:
- Argo CD

Problem

How to implement Blue-Green/Canary using GitOps?

- No built-in support for Blue-Green/Canary in K8S
- GitOps is declarative, Blue-Green/Canary are imperative
- Integration with CI tools

Attempt 1: Jenkins scripting

Except...

- Did not fit our GitOps model
- Not idempotent
- Extremely brittle (lots of assumptions and edge cases)
- Jenkins requires K8S credentials to deploy (risky!)
- Painful to set up
- and more...

Attempt 2: Deployment Hooks

Except...

- Still not idempotent and not transparent
- Requires a lot of work to start using it
- Still not following GitOps

Attempt 3: Custom Controller

Argo Rollouts Design Considerations

- Codifies the deployment orchestration in the controller
- GitOps-friendly (idempotent)
- Runs inside the K8S cluster
- Easy adoption and migration from deployments
- Feature parity with deployments

Introducing Argo Rollouts

- An advanced open-source K8S deployment controller
- Kubernetes-native
- Supports Blue-Green & Canary deployments

https://github.com/argoproj/argo-rollouts

How it works

- Handles ReplicaSet creation, scaling, and deletion
- Single Desired State as a Pod Spec
- Support manual and automated promotions
- Integrates with HPA

From Deployment to Argo Rollout

```yaml
apiVersion: apps/v1 argoproj.io/v1alpha1
kind: Deployment Rollout
metadata:
  name: exciting-app
spec:
  replicas: 10
  selector:
    app:  exciting-app
  template:
    metadata:
      labels:
        app: exciting-app
    spec:
      containers:
        - name: exciting-app
          image: example/hello-world
  strategy:
    ...
```

Blue-Green Strategy

```yaml
apiVersion: apps/v1 argoproj.io/v1alpha1
kind: Deployment Rollout
metadata:
  name: exciting-app
spec:
  replicas: 10
  selector:
    app:  exciting-app
  template:
    metadata:
      labels:
        app: exciting-app
    spec:
      containers:
        - name: exciting-app
          image: example/hello-world
  strategy:
    blueGreen:
      activeService: active-svc
      previewService: preview-svc # optional
      previewReplicaCount: 1 # optional
      autoPromotionSeconds: 30 # optional
      scaleDownDelaySeconds: 30 # optional
```

- Manages an active and preview Services' selector to provide a service-level cutover
- Sizing control over preview environment
- Manual or automatic promotion

Canary Strategy

```yaml
apiVersion: apps/v1 argoproj.io/v1alpha1
kind: Deployment Rollout
metadata:
  name: exciting-app
spec:
  replicas: 10
  selector:
    app:  exciting-app
  template:
    metadata:
      labels:
        app: exciting-app
    spec:
      containers:
      - name: exciting-app
        image: example/hello-world
  strategy:
    canary:
      maxSurge: 10%
      maxUnavailable: 1
      steps:
      - setWeight: 10 #percentage
      - pause:
          duration: 60 # seconds
```

- Declarative Promotion
- No Service modification
- Traffic Split based on replica ratio between versions of an application
- Superset of Rolling Update Strategy
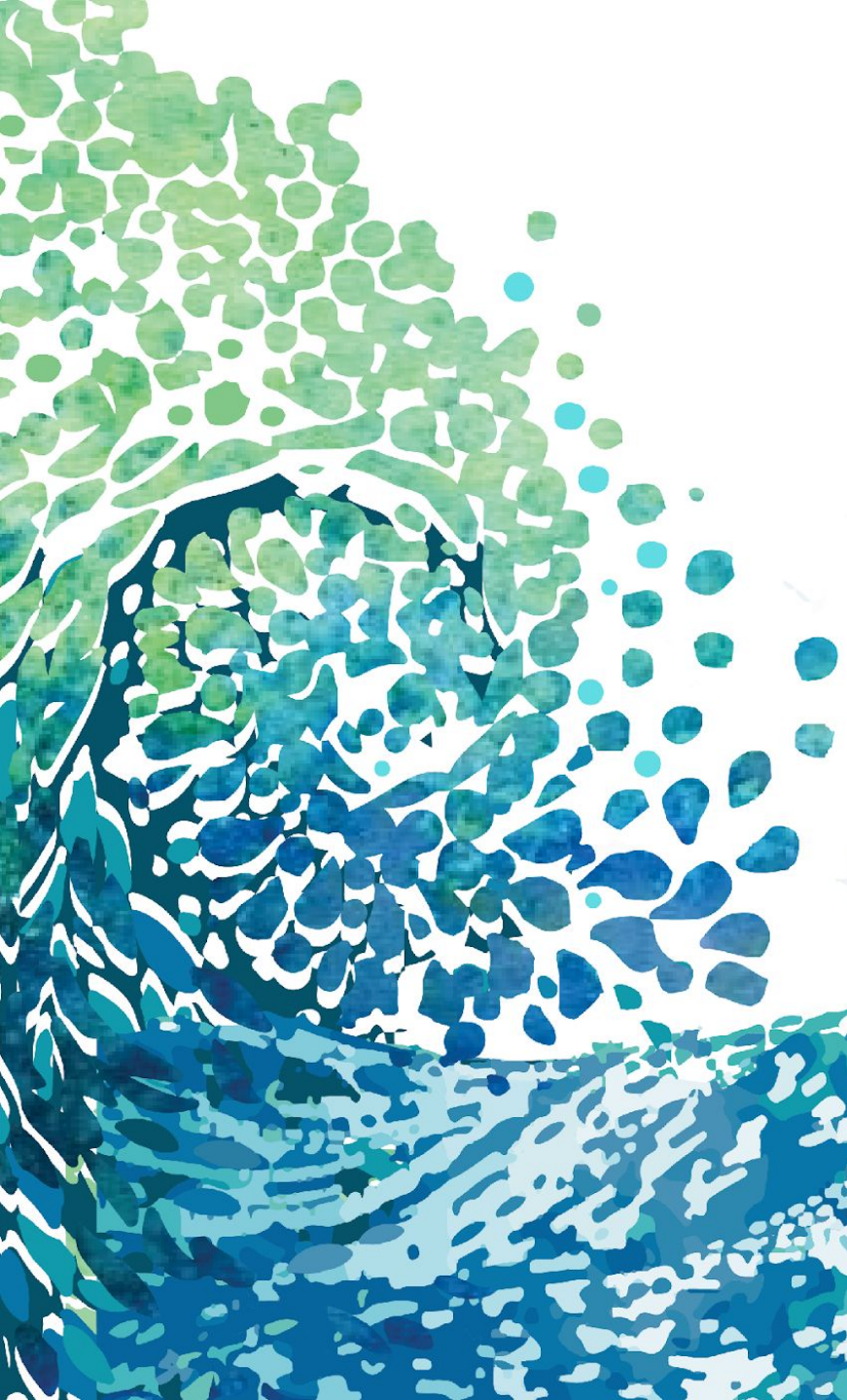
DEMO

What's next?

- Decision-based Promotion CRD
- Service Mesh integration
- A/B testing, experimentation strategies

Resources

- Argo Rollouts: https://github.com/argoproj/argo-rollouts
- Argo CD: https://argoproj.github.io/argo-cd/
- Intuit Careers: https://careers.intuit.com/

Thank you!