How We Build Production-Grade HPA: From Effective Algorithm to Risk-Free Autoscaling

Ziqiu Zhu & Yiru Guo, Ant Group

Overview

1. *Take a brief look at K8s HPA (what's its problem?)*
2. *How we build production-grade HPA at Ant Group*
3. *How Kapacity practically applies the above methodology to any of your Kubernetes environments*

About Us

朱子秋
Ziqiu Zhu

GitHub: zqzten

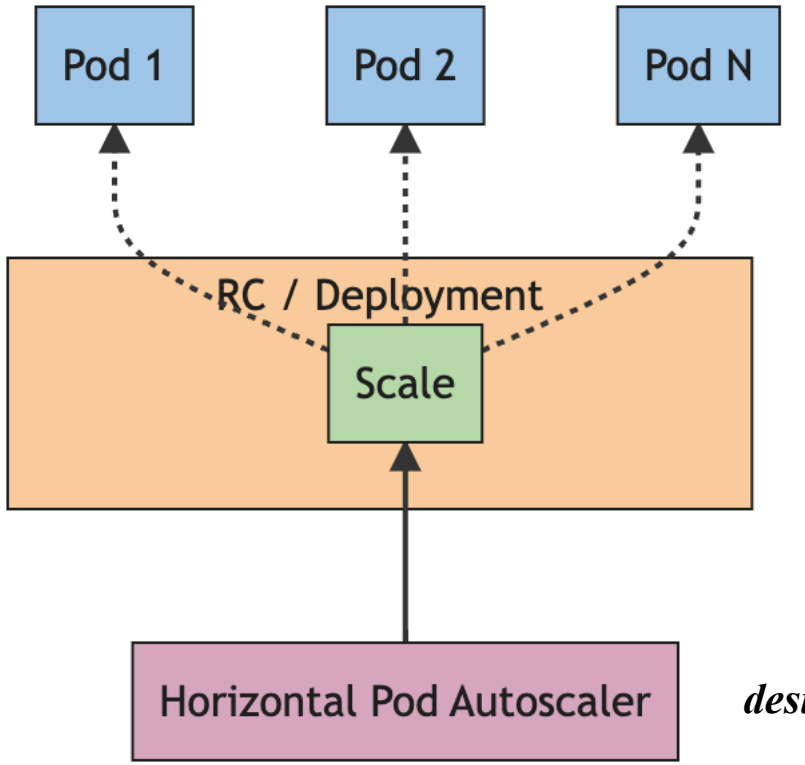
- Engineer at Cloud Native Technology, Ant Group
- Experienced in building cloud native platforms and products
- Focus on autoscaling, scheduling, multi-cluster
- Kubernetes Member
- Contributor of multiple CNCF projects (Kubernetes and its sub-projects, Koordinator, etc.)
- Co-Founder of Kapacity project

郭钇汝
Yiru Guo

GitHub: dayko2019

- Engineer at Infrastructure Reliability, Ant Group
- Working in Intelligent Capacity Team
- Deeply involved in the construction of various production-grade capacity technologies for large-scale production systems at Ant Group from the very beginning
- Co-Founder of Kapacity project

Brief look at K8s HPA

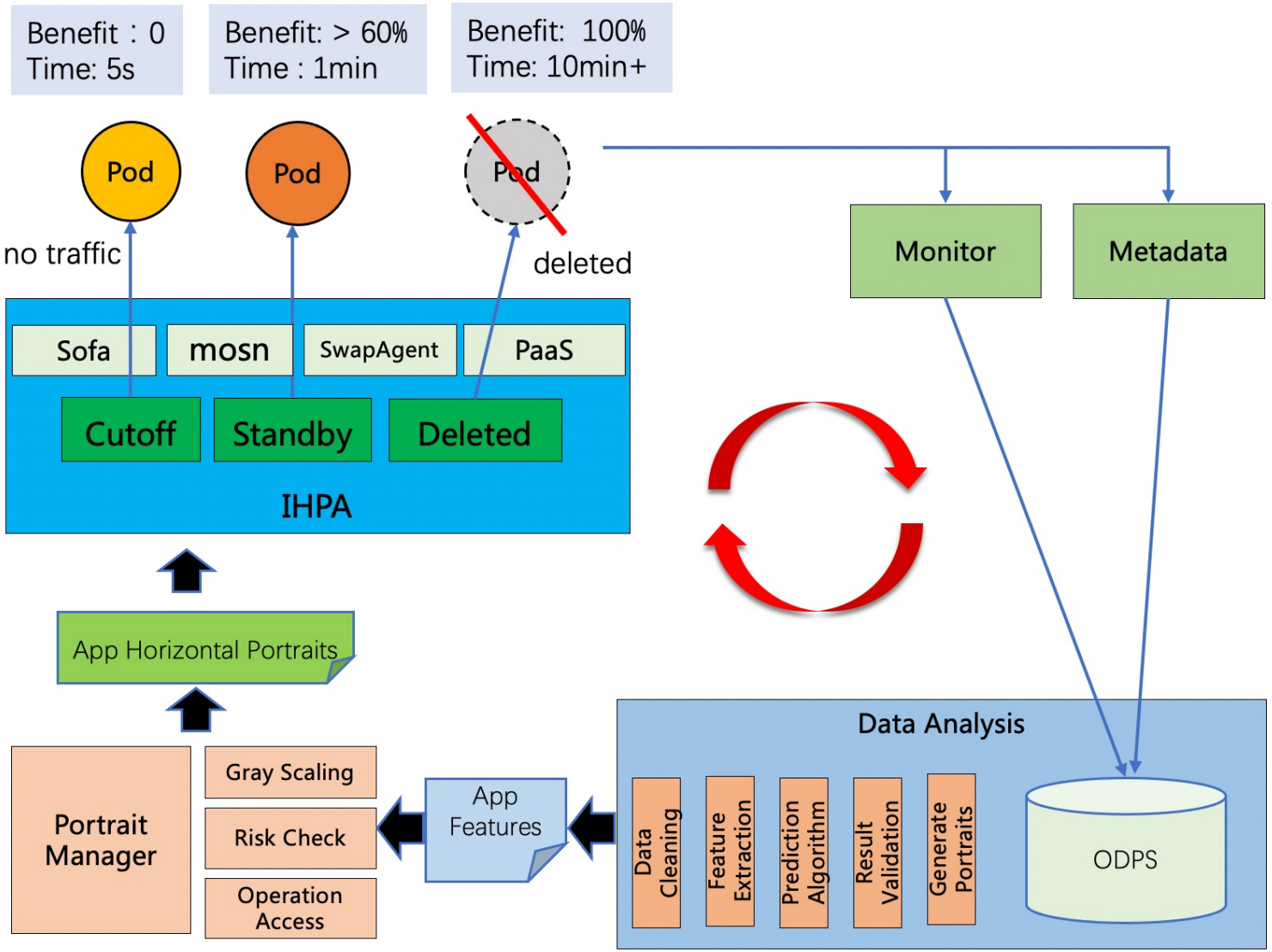


Non Production-Grade Sides:

- Works only in reactive way
- Simple ratio algorithm
- Limited risk mitigation ability
- Kubernetes built-in, hard to customize

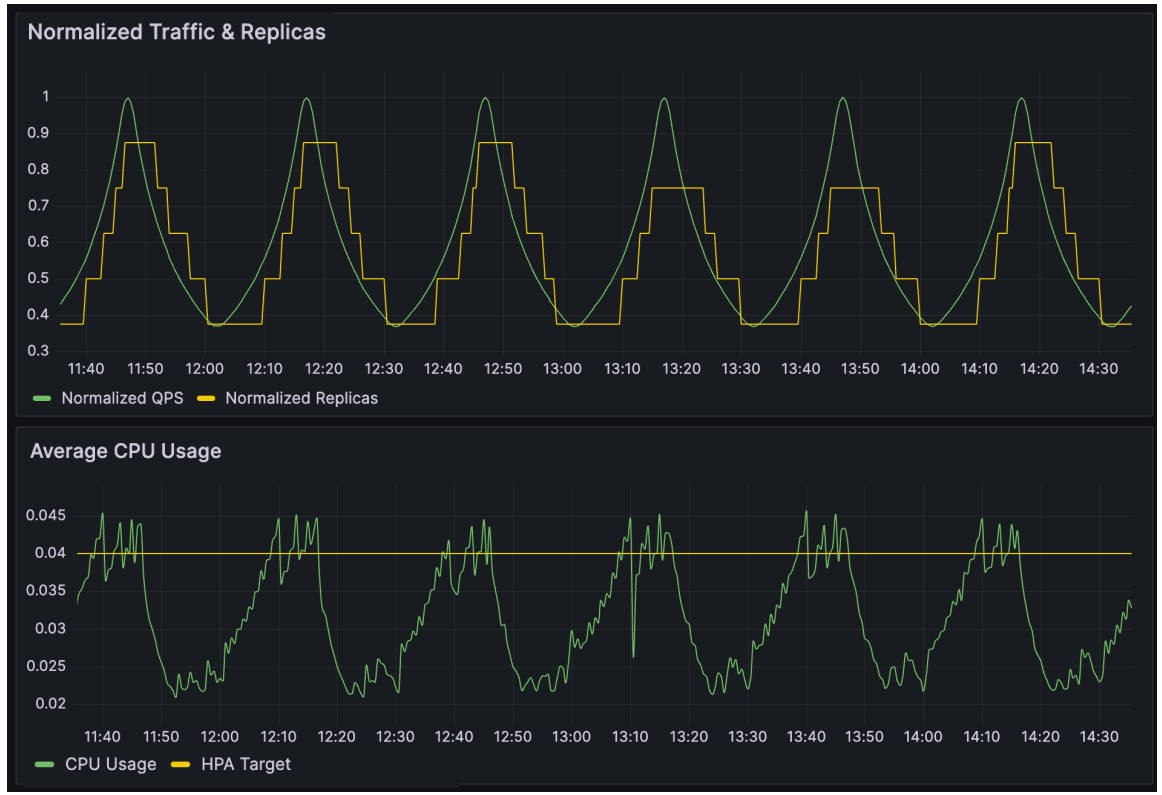
$$desiredReplicas = ceil[currentReplicas * (currentMetricValue / desiredMetricValue)]$$

Production-Grade HPA at Ant Group



- Introduce **multiple intelligent algorithms** (including prediction, burst detection, etc.) based on historical and real-time metrics to improve the effect and stability of autoscaling
- Introduce **multi-stage scaling** to improve the efficiency and mitigates risk of autoscaling
- Introduce **gray scaling** to minimize risk of autoscaling
- Saving ~100k CPU cores yearly with high stability

Why Prediction – A Simple Sample



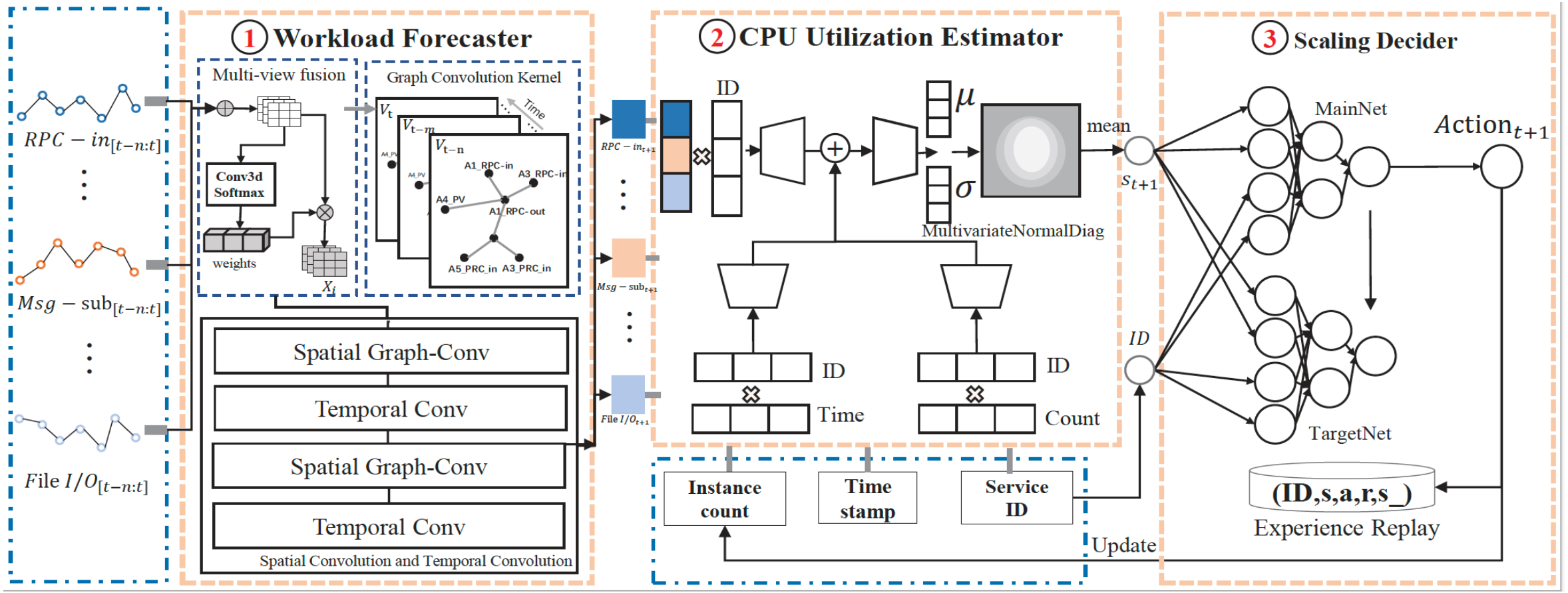
Reactive

- Scaling **after** traffic fluctuation
- **Hard to achieve** desired resource usage
- Scaling with **low precision**

Predictive

- Scaling **before** traffic fluctuation
- **Easy to achieve** desired resource usage
- Scaling with **high precision**

Our “Traffic-Driven” Prediction Model



Traffic Forecasting

Time series forecasting of traffics.
e.g. Pyraformer, ICLR 2022

Resource Estimating

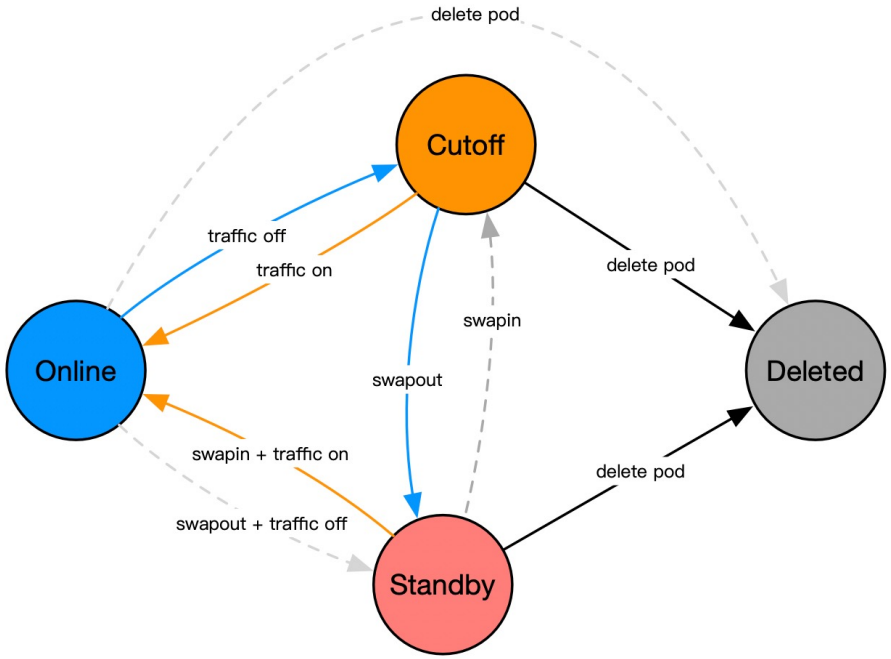
Find the relationship between resource usage, traffics and replicas.

Scaling Decision

Get optimal replica count.
e.g. Improved DQN Model, SoCC 2022

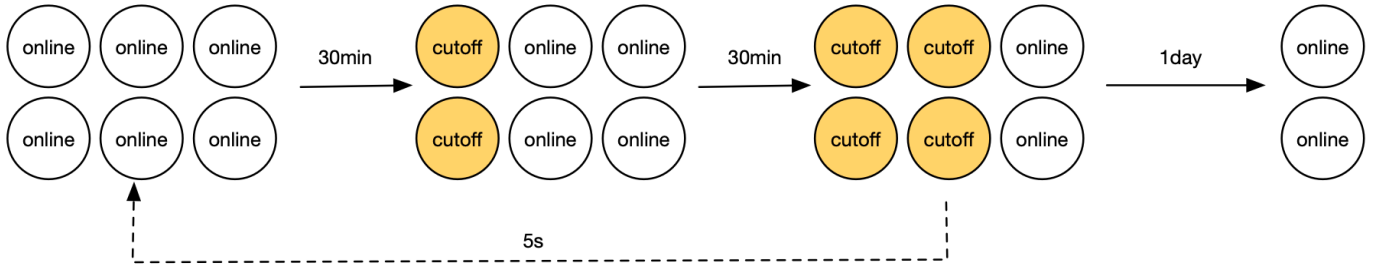
Mitigate Risk of Autoscaling

1. *Multi-Stage Scaling*
   - Fine-grained Pod state control: Online/Cutoff/Standby
   - Faster scaling
   - Can be utilized by higher-level resource orchestration to further increase resource utilization

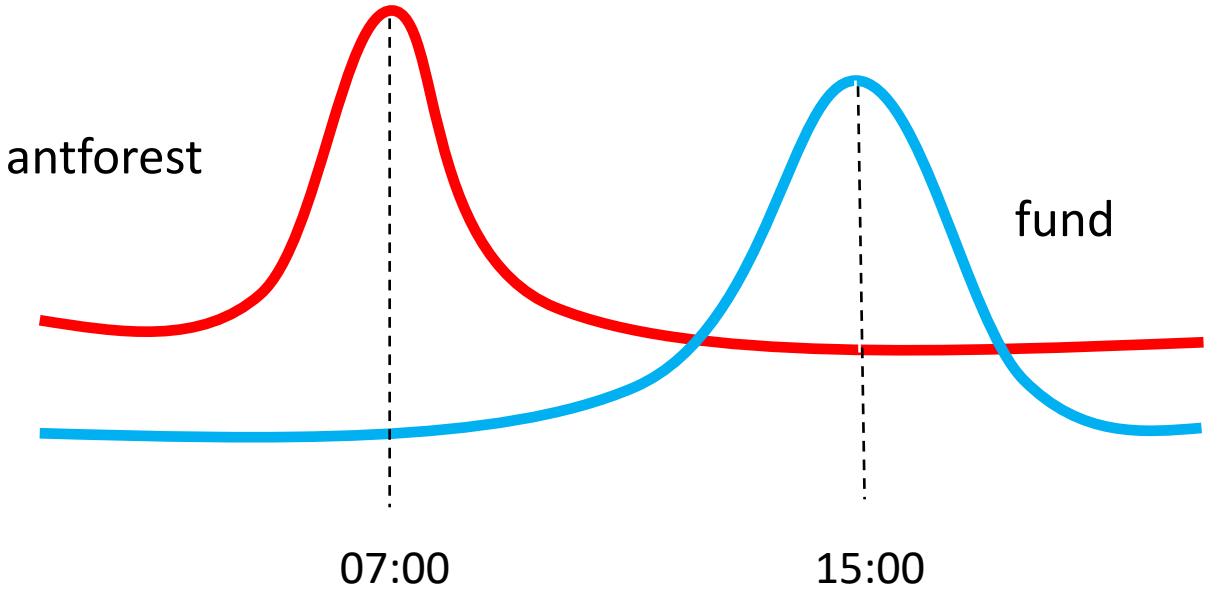


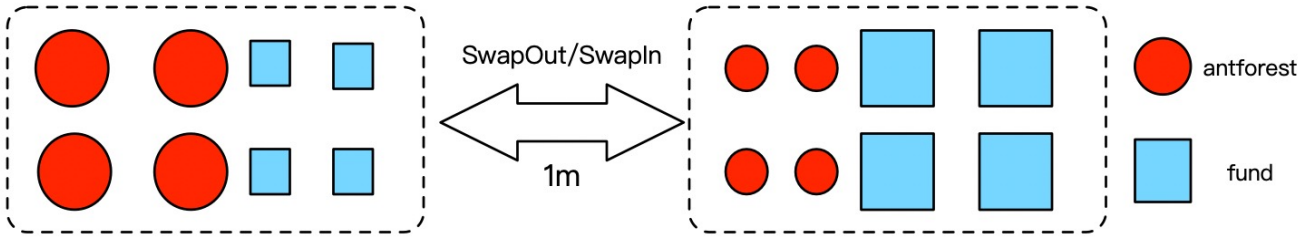
2. *Gray Scaling*
   - Reduce “explosion radius” of breakdown caused by autoscaling
   - Can be combined with multi-stage scaling to reduce rollback time

3. *Automatic Risk Detection and Mitigation during Autoscaling*
   - Multi-dimensional anomaly detection, not limited to scaling metrics
   - Automatic risk mitigation



One Step Further: Time-Sharing Scheduling





Resource Sharing

peer pods can share resources: CPUs are dynamically shared, memory can be swapped

State Switching

peer pods can switch to opposite state (online/standby) quickly

Introduce Kapacity's IHPA

Kapacity

https://github.com/traas-stack/kapacity

Kapacity is an open **cloud native** capacity solution which helps you achieve ultimate resource utilization in an **intelligent** and **risk-free** way.

- *2023.6 OSS v0.1*
- *2023.9 v0.2 final testing*
- *2023.10 v0.2 milestone release*

Intelligent HPA (IHPA) - An intelligent, risk-defensive, highly adaptive and customizable substitution for HPA.

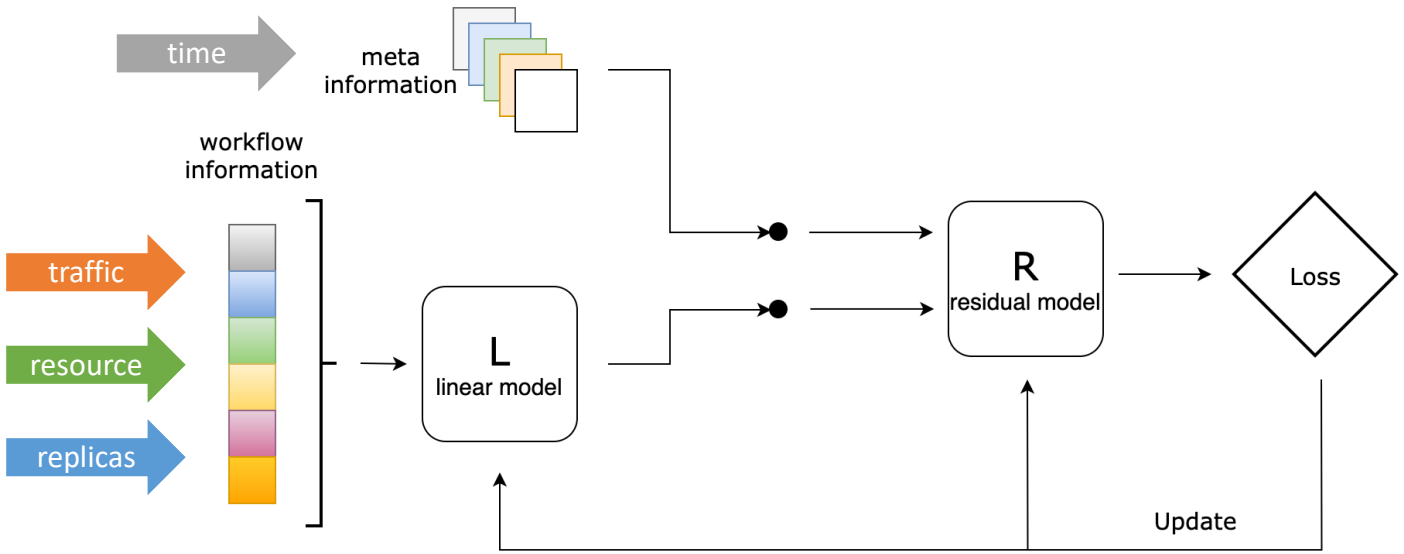
Autoscaling powered by intelligent algorithms

Scaling with multiple risk defense means

Open and highly extensible architecture

Traffic-Driven Replicas Prediction

Predicting replicas based on predicted traffics.

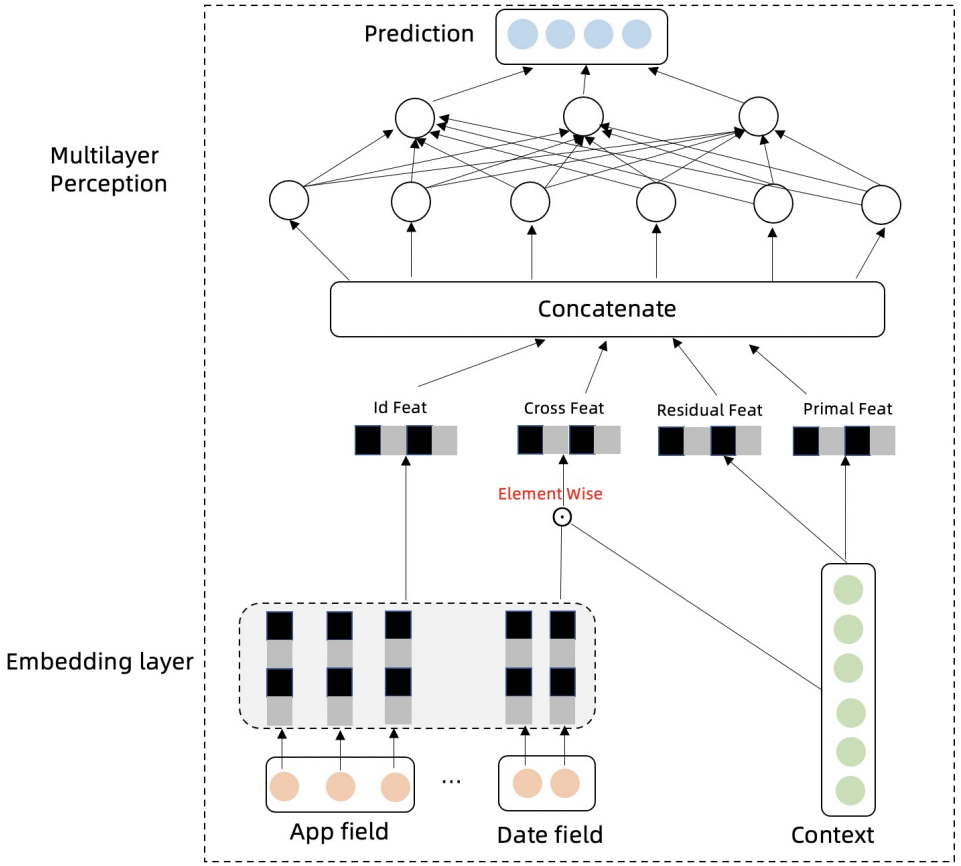


f(traffic/replicas) = resource util

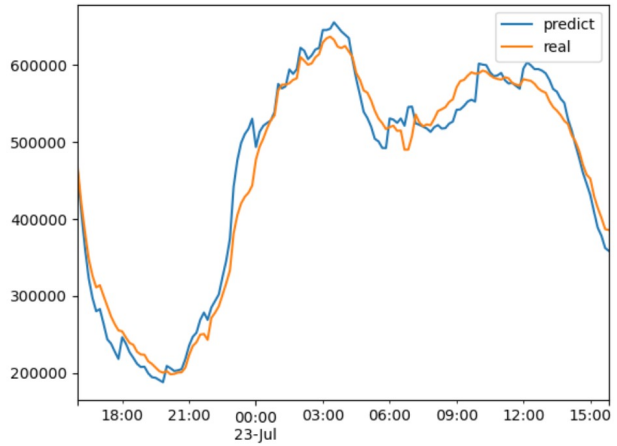
- *Metric Target*
  - Resource Utilization
- *Metric Source*
  - Traffic History
  - Resource Util History
  - Replicas History
- *Model Impl*
  - Linear: ElasticNet
  - Residual: LightGBM

Time Series Forecasting of Traffic

Predicting traffics by time series forecasting.



Model Structure



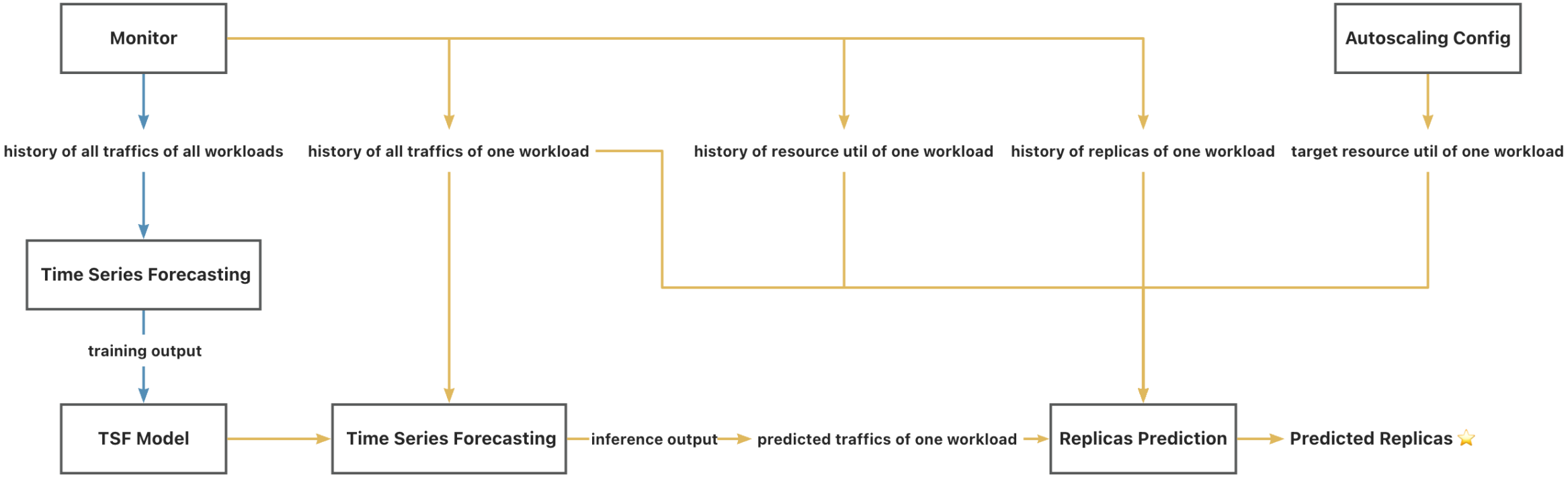
Forecasting Result of Real Production Traffic with 10min Precision

- *Lightweight*
  - **model size < 1MB** when forecasting 12 points (2h with 10min precision) of 1 traffic with history of 12 points
  - **costs 1min/epoch** when training with laptop CPU
- *Good Performance*
  - **better performance** compared to other popular models on real production traffic dataset

	MAE	RMSE
DeepAR	1.734	31.315
N-BEATS	1.851	41.681
ours	1.597	28.732

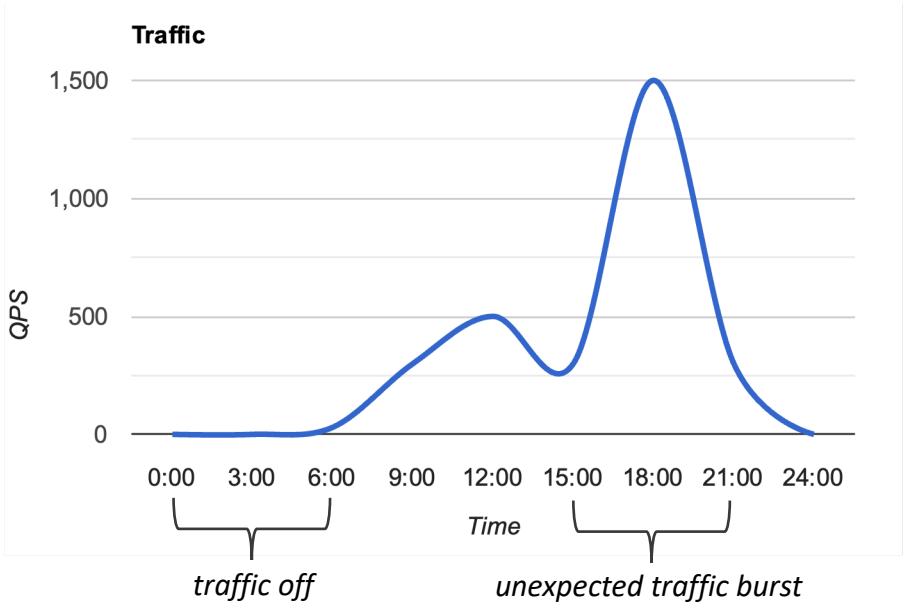
Models Comparison on Real Production Traffic Dataset

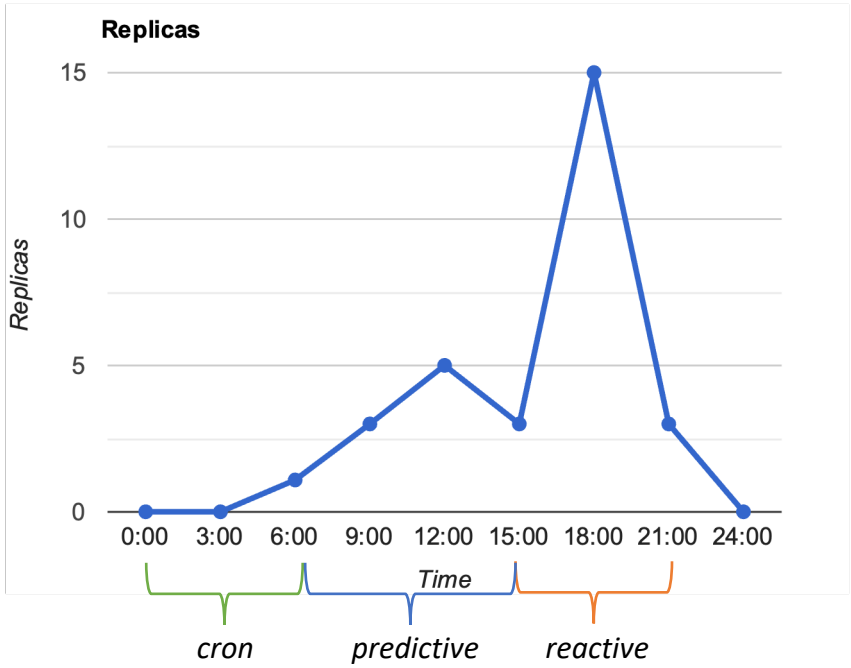
Wrap Up: Replicas Prediction Workflow



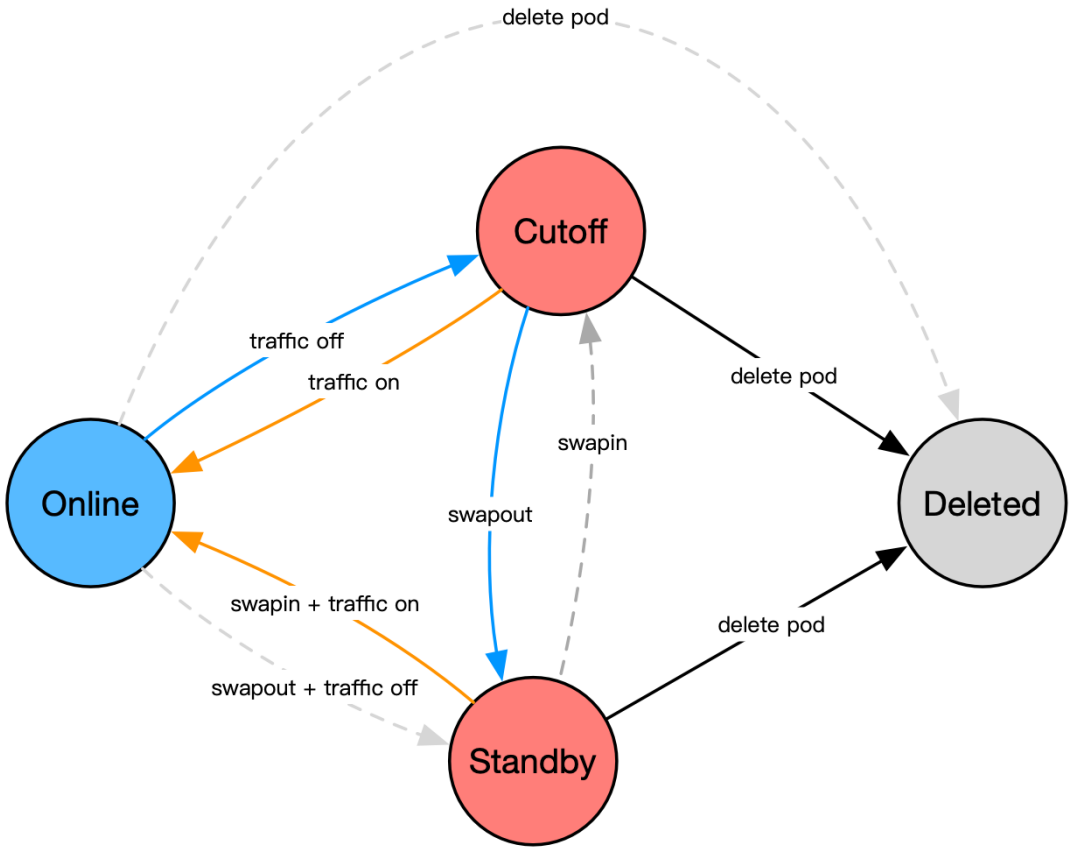
Autoscaling with Multiple Prioritized Rules


```yaml
portraitProviders:
- type: Dynamic
  priority: 10
  dynamic:
    portraitType: Predictive
    metrics:
    - type: Resource
      resource:
        target:
          type: Utilization
          averageUtilization: 45
      # ...
    algorithm:
      type: ExternalJob
      # ...
- type: Dynamic
  priority: 10
  dynamic:
    portraitType: Reactive
    metrics:
    - type: Resource
      resource:
        target:
          type: Utilization
          averageUtilization: 45
    algorithm:
      type: KubeHPA
- type: Cron
  priority: 20
  cron:
    crons:
    - name: offline at night
      start: 0 0 * * *
      end: 0 6 * * *
      replicas: 0
```





Multi-Stage Gray Scaling



Sort

- decide the scale down order of Pods
- workloads have their default orders
- able to customize order if workload supports selecting Pods to scale down

Cutoff

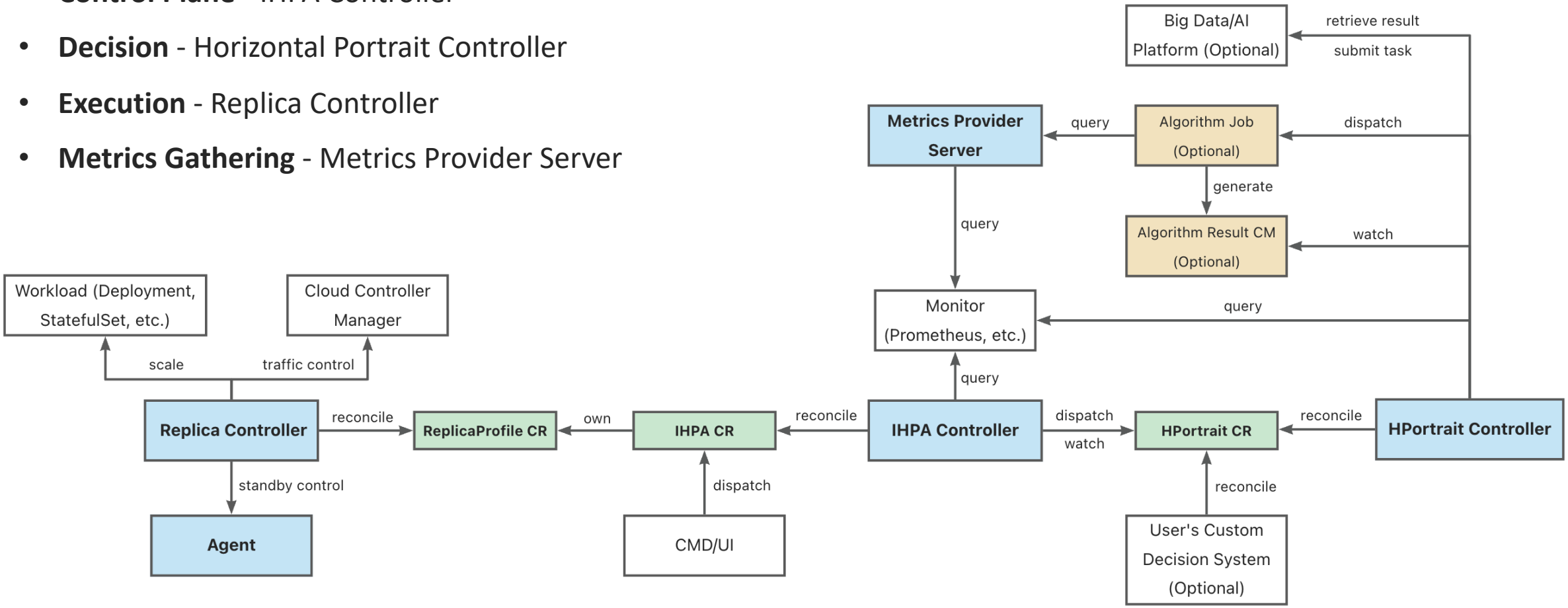
- cutoff Pod's traffic
- utilize readiness gate by default
- support custom traffic controllers

Scale

- scale down (delete) the Pod
- utilize Kubernetes scale API

All in One, with Everything Customizable

- **Control Plane** - IHPA Controller
- **Decision** - Horizontal Portrait Controller
- **Execution** - Replica Controller
- **Metrics Gathering** - Metrics Provider Server



What's Next

- **Intelligence**
  - More algorithms (burst detection, etc.)
  - Further automated algorithm workflows
- **Risk-Mitigation**
  - Anomaly detection during autoscaling
  - Automatic risk mitigation (pause, rollback, etc.)
- **Visualization**
  - Dashboard
  - Visualized resource utilization, costs and carbon emission
- **More than HPA...**

Thanks! Q & A

Kapacity

https://github.com/traas-stack/kapacity

群聊: Kapacity 开源交流群

该二维码7天内(10月5日前)有效，重新进入将更新

微信交流群

钉钉交流群

slack

Slack Workspace (for English Speakers)