keptn

Overcoming (organizational) scalability issues in your Prometheus ecosystem

Jürgen Etzlstorfer

DevOps Activist @ Dynatrace

@jetzlstorfer

https://www.linkedin.com/in/juergenetzlstorfer

Hands-On @ https://tutorials.keptn.sh

Follow us @keptnProject

Star us @ https://github.com/keptn/keptn

Visit us @ https://keptn.sh

Join the Keptn Community via
https://github.com/keptn/community

Abstract

keptn

Prometheus is considered a foundational building block when running applications on Kubernetes and has become the de-facto open-source standard for visibility and monitoring in Kubernetes environments.

Your first starting points when operating Prometheus are most probably configuring scraping to pull your metrics from your services, building dashboards on top of your data with Grafana, or defining alerts for important metrics breaching thresholds in your production environment. in your production environment.

As soon as you are comfortable with Prometheus as your weapon of choice, your next challenges will be scaling and managing Prometheus for your whole fleet of applications and environments. As the journey “From Zero to Prometheus Hero” is not trivial you will find obstacles on the way. In this presentation we are highlighting the most common challenges we have seen and provide guidance on how to overcome them. Finally, we are discussing a solution to get you there more quickly to build automated, future-proof observability with Prometheus showing Keptn as one possible implementation.

https://medium.com/keptn/overcoming-scalability-issues-in-your-prometheus-ecosystem-4430cea6472f?source=friends_link&sk=edf9a96ff81721c290e005fc7b4b9bfb

My mission today

keptn

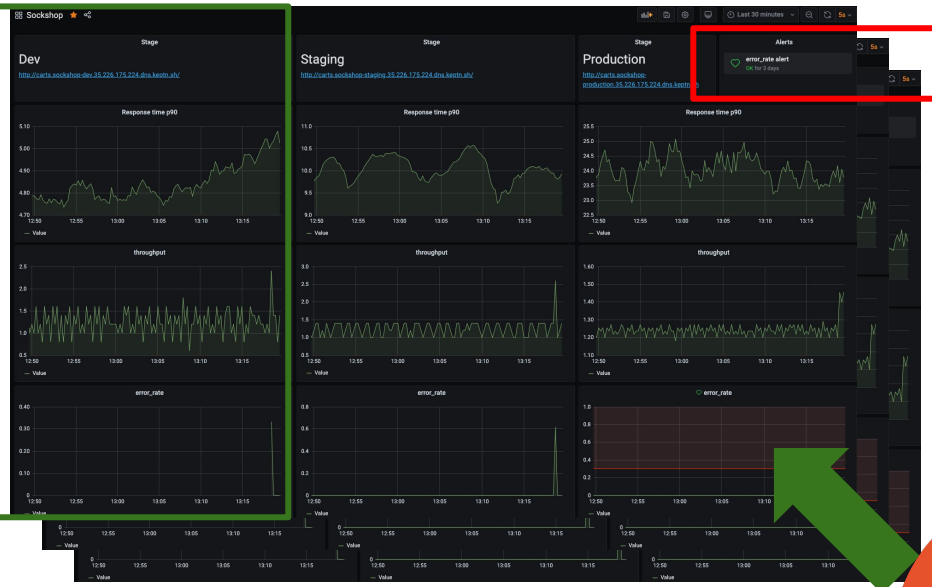
- **Discuss** prevalent **challenges** we have discovered
- **Propose solutions** to overcome these challenges
- **Learn from you** what needs to be done to finally solve them
- **Please add your comments**, suggestions, questions in the chat or **reach out** to me via Twitter, LinkedIn, Slack, ...

What I want

keptn

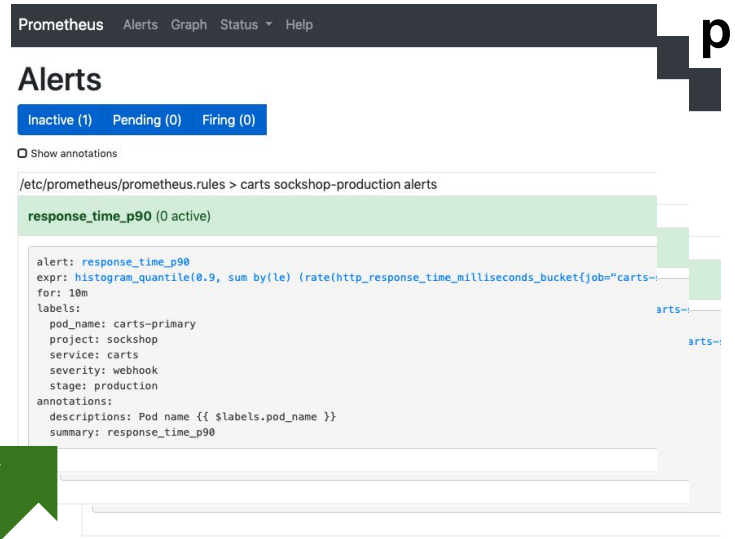
✔ **Automated dashboards**

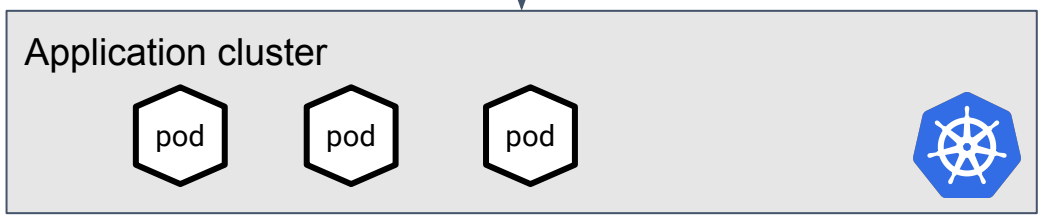
RED Metrics



Alerts

✔ **Alerts set up for apps in production**





✔ **Automated monitoring of my applications**

Agenda

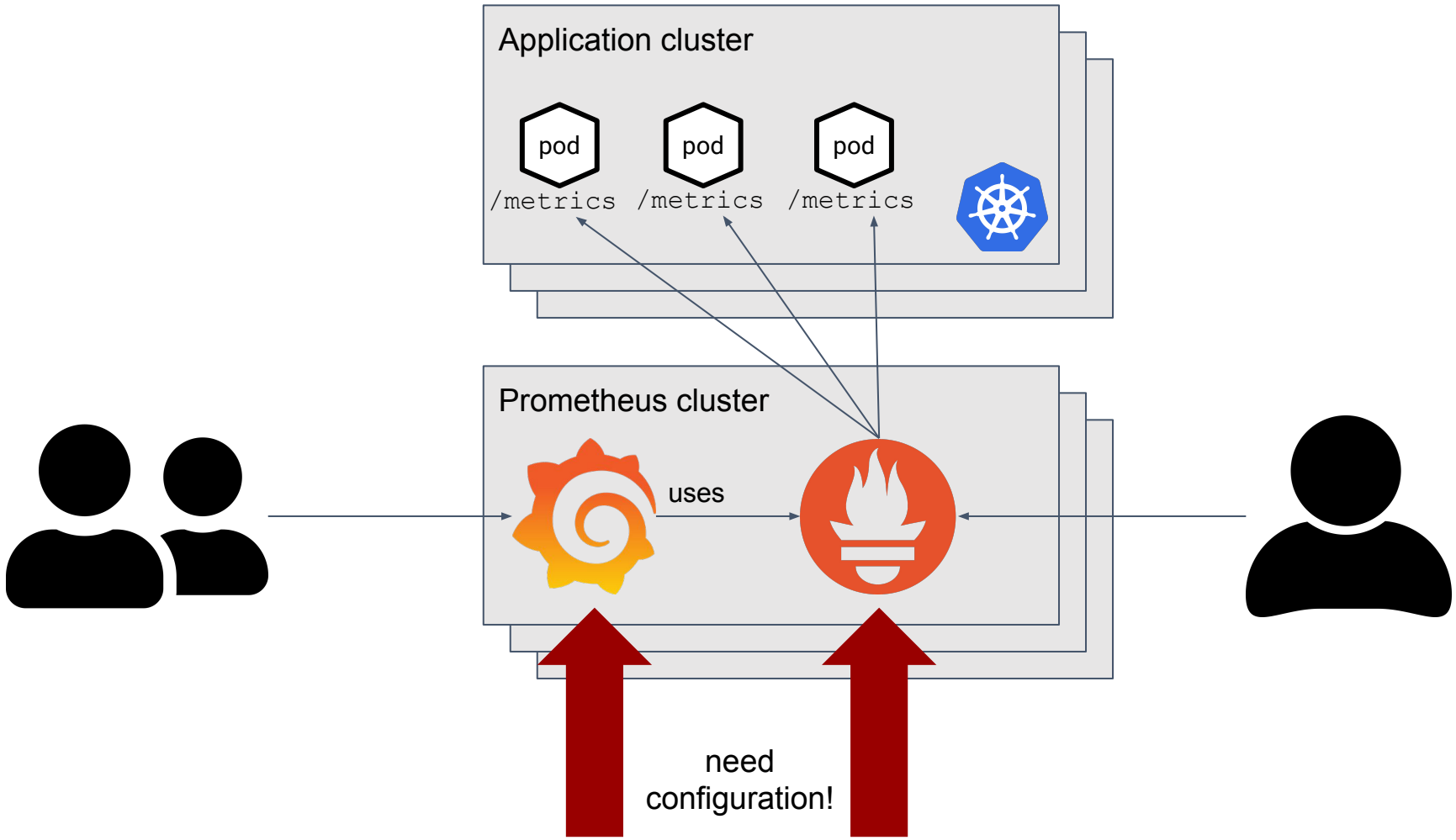
keptn

1. Where to **start**: basic Kubernetes building blocks
2. Challenge 1: No central **configuration management**
3. Challenge 2: **Writing** and **maintaining** of configurations
4. Challenge 3: **Code duplication**
5. **Keptn** to the rescue

Read the blog:
https://medium.com/keptn/overcoming-scalability-issues-in-your-prometheus-ecosystem-4430cea6472f?source=friends_link&sk=edf9a96ff81721c290e005fc7b4b9bfb

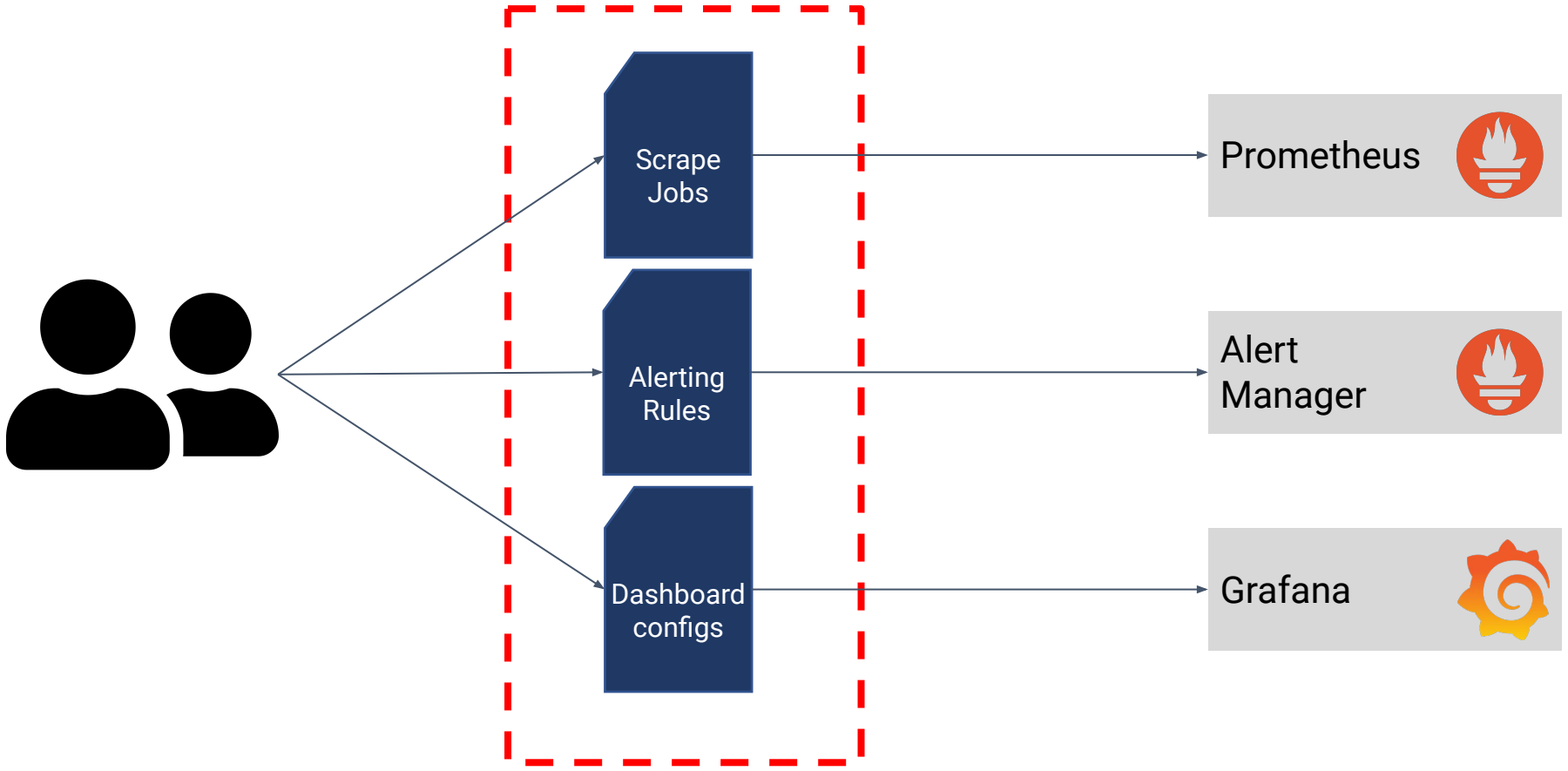
Kubernetes monitoring building blocks (simplified!)

keptn



Challenge 1: No central configuration management

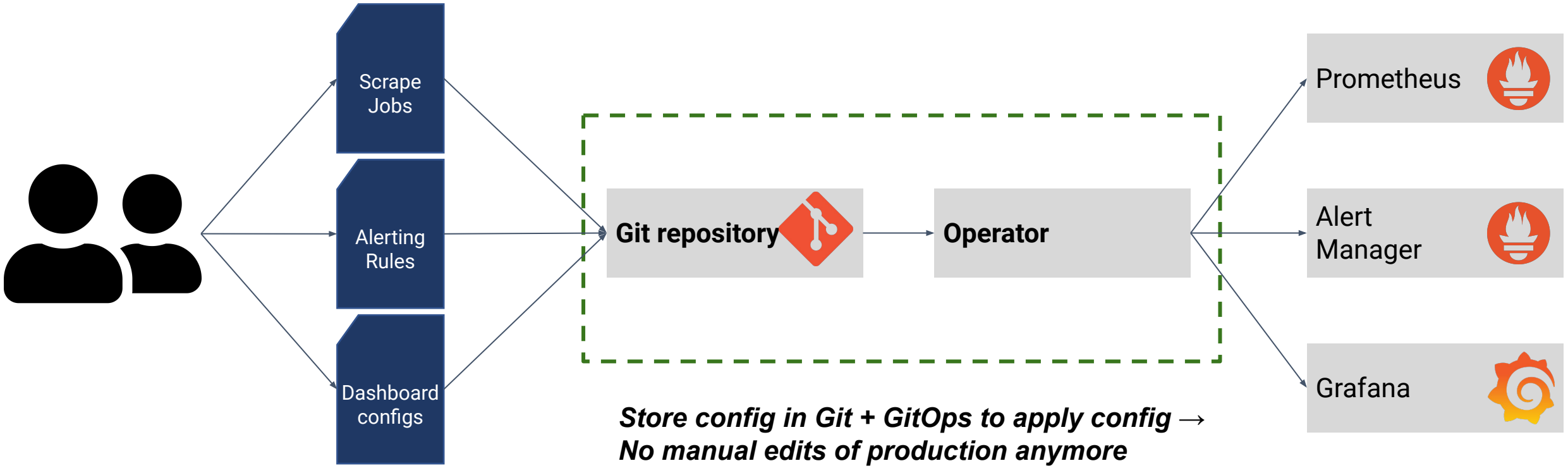
keptn



How to version and keep in sync?

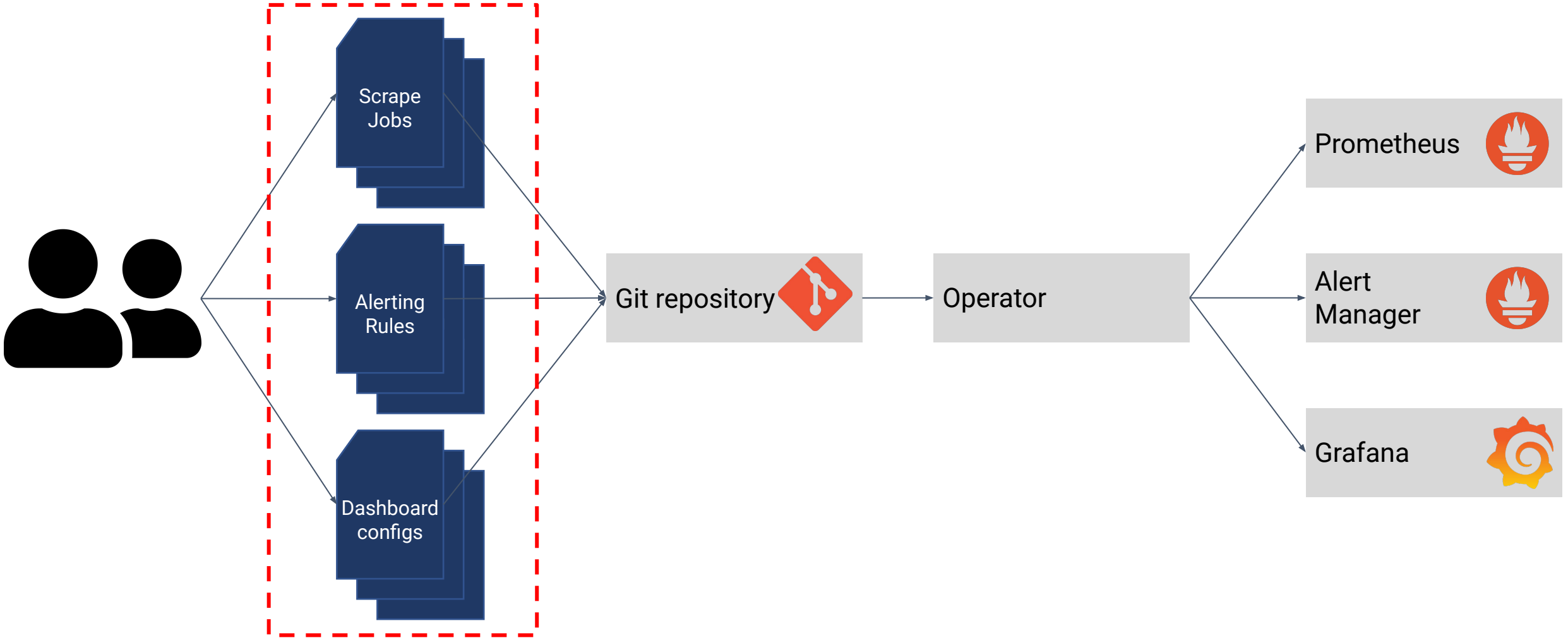
Potential solution: GitOps

keptn



Challenge 2: Writing all the configurations

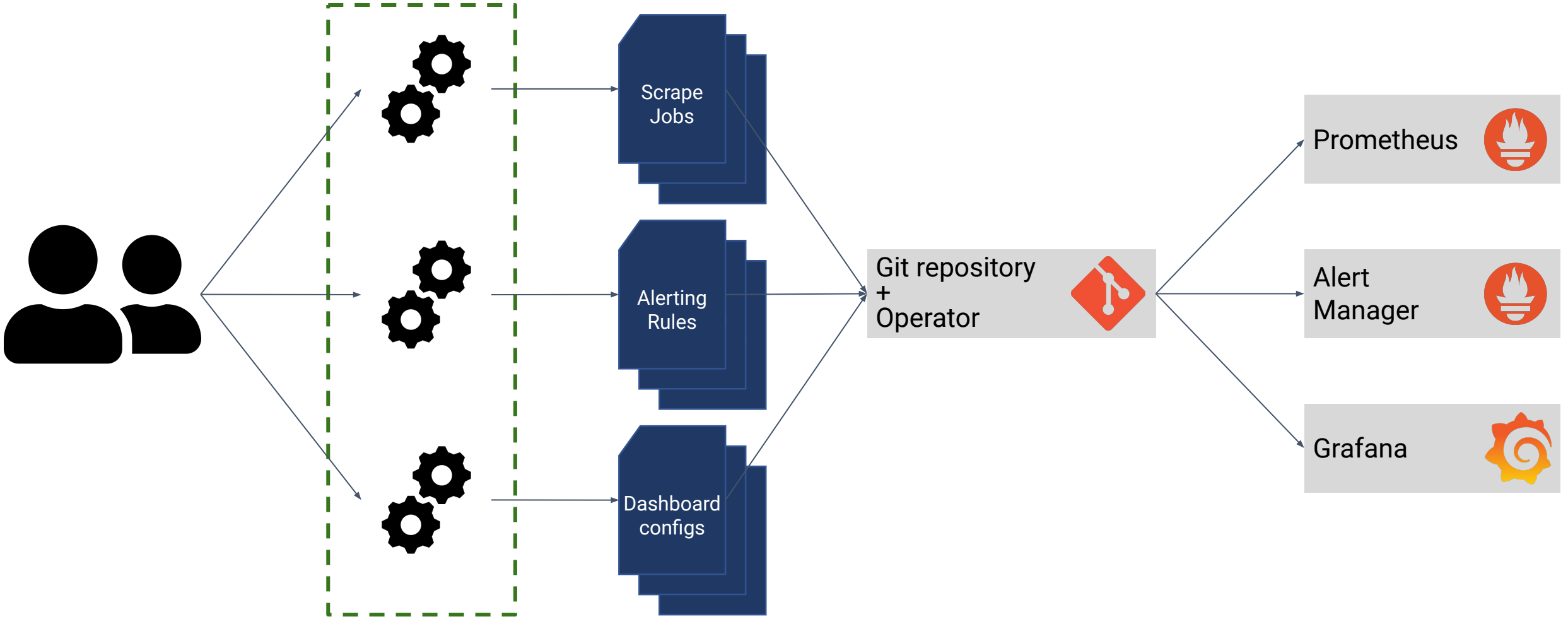
keptn



Not one, but many configurations!

Potential solution: Use code generators

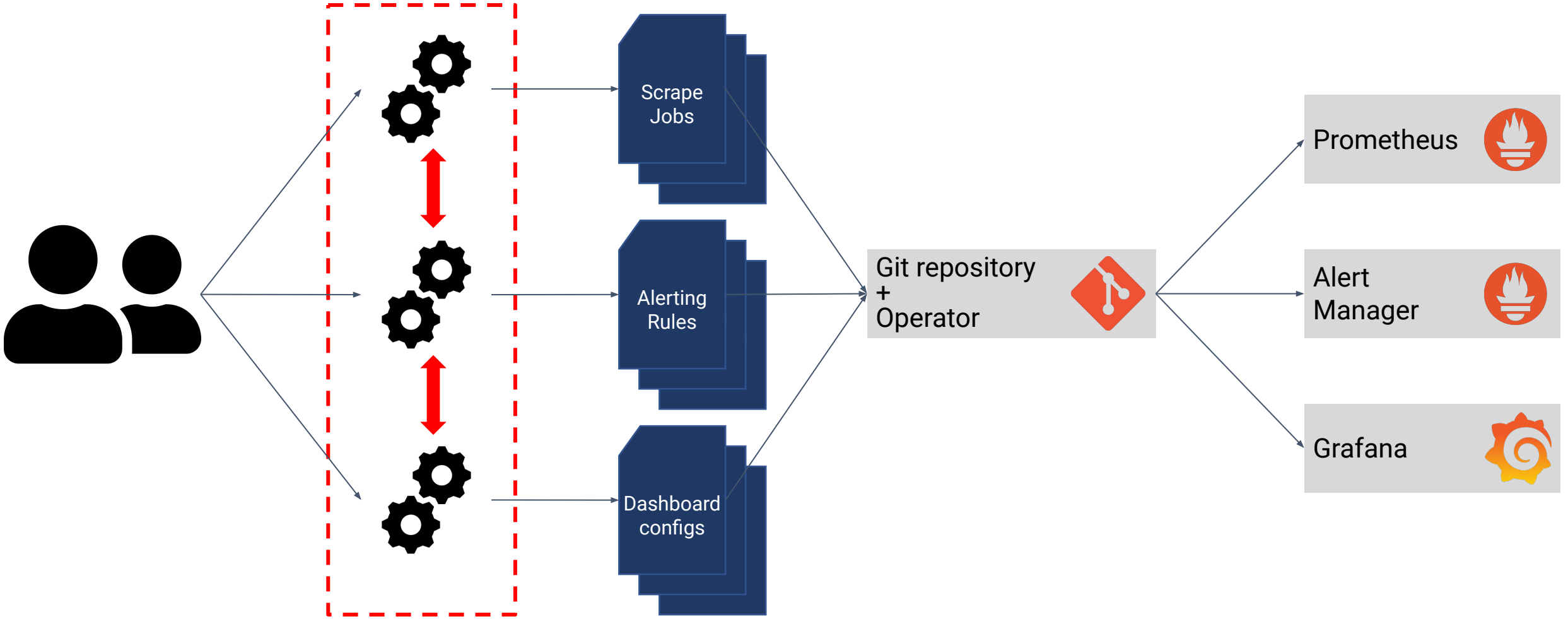
keptn



(Potential generators in reference section)

Challenge 3: No synchronization between configs

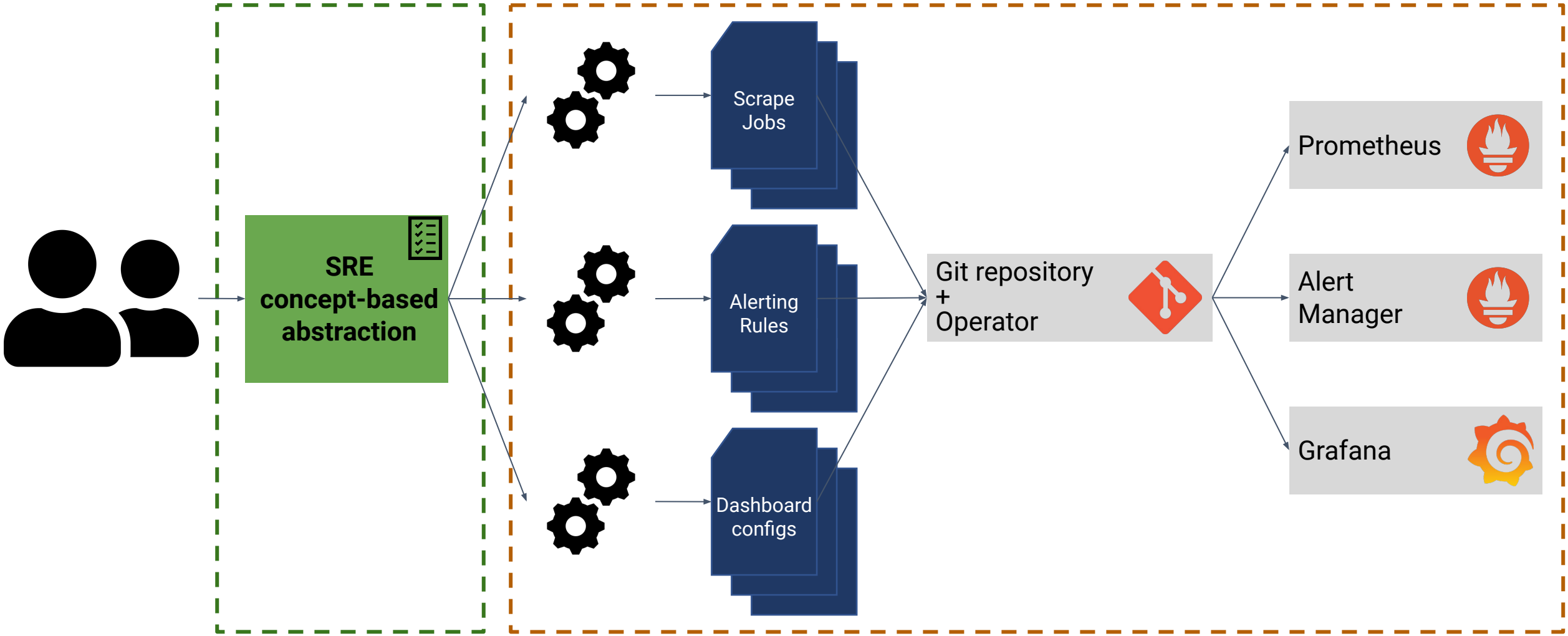
keptn



Different configs drift out-of-sync after time

Potential solution: Abstraction of implementation details

keptn



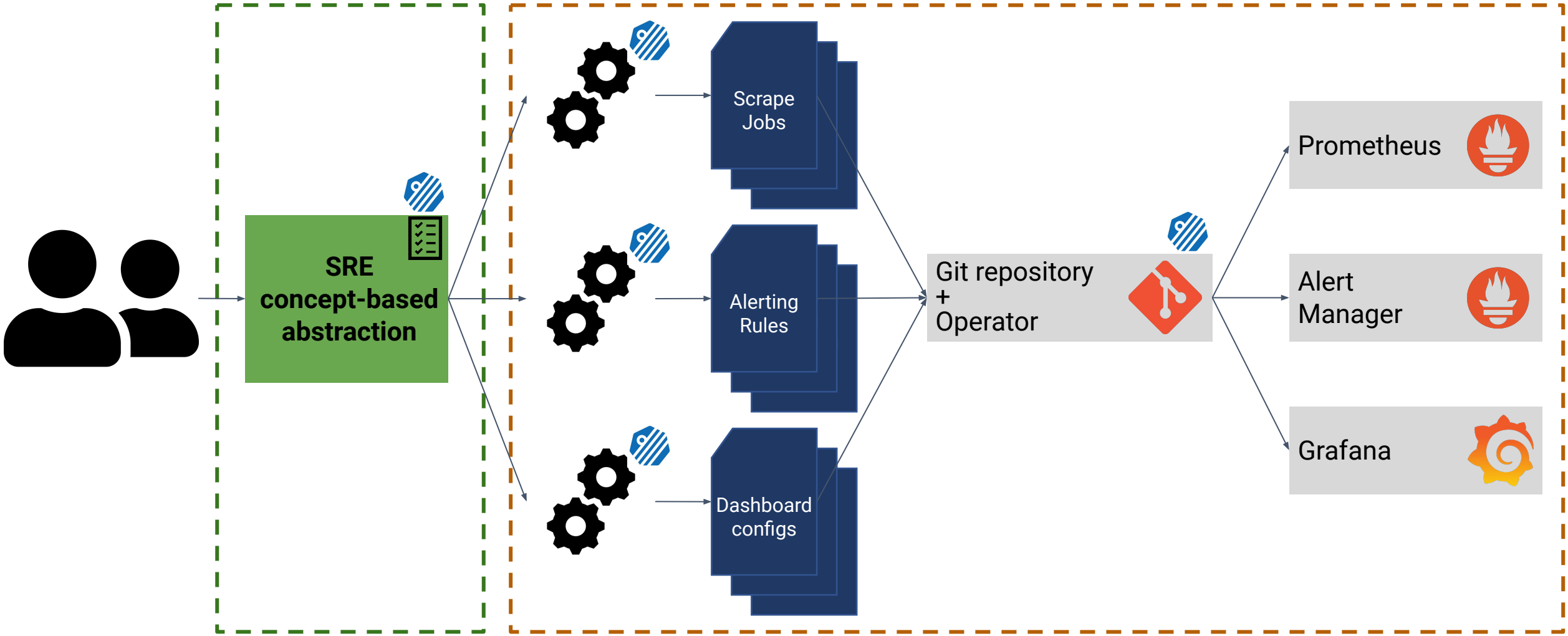
✔ ***Define WHAT you want!***

vs. HOW it is done! ✘

Introducing  keptn

How to **Automate Configuration of Prometheus & Grafana**
+
Automated **Quality Gates** for Continuous Delivery
+
Automated Operations

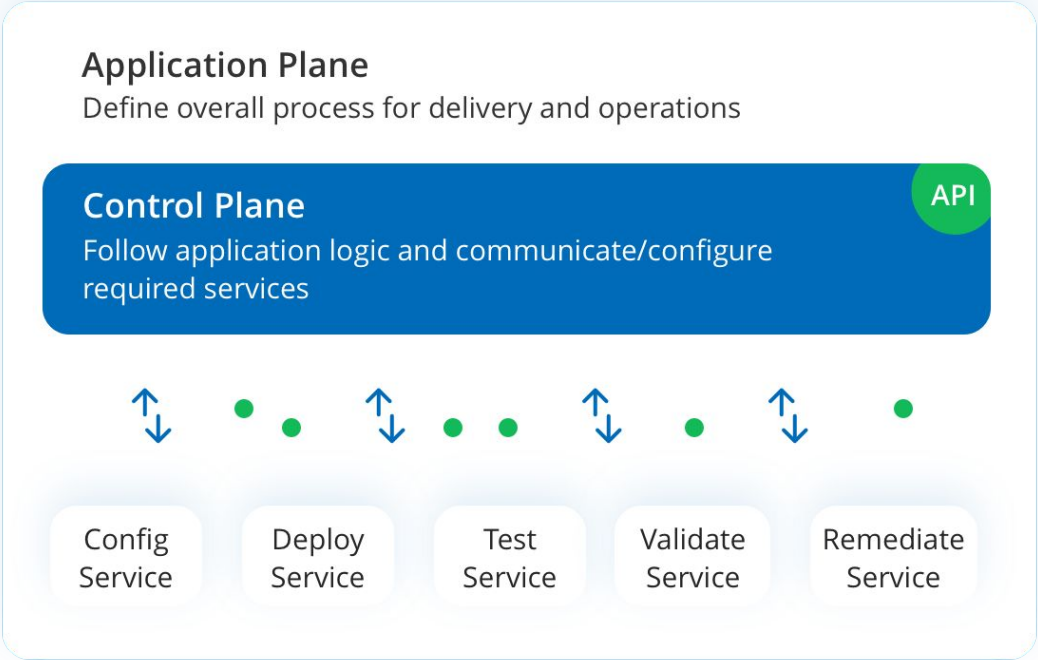
The Keptn way: Abstraction of implementation details



keptn Demo

Keptn in a nutshell

Keptn is an event-based **control plane** for **continuous delivery** and **automated operations** for cloud-native applications.



keptn – **conceptual architecture**

Environment Definition (shipyard file)

Core

keptn Autonomous Cloud Control Plane

Services

GitOps

Container Registry

Continuous Delivery

Test Automation

ChatOps

Observabi-lity

Dash-boarding

Platform

Dev Namespace

Staging Namespace

Production Namespace

Incorporating SRE Best Practices

keptn

SLIs drive **SLOs** which inform **SLAs**

- **Service Level Indicators (SLIs)**
  - Definition: Measurable Metrics as the base for evaluation
  - Example: Error Rate of Login Requests

RED metrics already included in Keptn

- **Service Level Objectives (SLOs)**
  - Definition: Binding targets for Service Level Indicators
  - Example: Login Error Rate must be less than 2%
- **Service Level Agreements (SLAs)**
  - Definition: Business Agreement between consumer and provider typically based on SLO
  - Example: Logins must be reliable & fast (Error Rate, Response Time, Throughput) 99% within a 30 day window
- Google Cloud YouTube Video
  - SLIs, SLOs, SLAs, oh my! (class SRE implements DevOps)

SLI/SLO-based creation on alerts and dashboards

keptn

SLIs defined per SLI Provider as YAML

SLI Provider specific queries, e.g: Prometheus Metrics Query

```yaml
indicators:
  error_rate:          "sum(rate(http_requests_total{job='...'}"
  count_dbcalls:       "sum(rate(http_requests_total{job='...'})""
  response_time_p90:   "histogram_quantile(0.90, sum(rate("
```

SLOs defined on Keptn Service Level as YAML

List of objectives with fixed or relative pass & warn criteria

```yaml
objectives:
  - sli: error_rate
    pass:
    - criteria:
      - "<=1“  # We expect a max error rate of 1%
  - sli: response_time_p90
  - sli: count_dbcalls
    pass:
    - criteria:
      - "=+2%"  # We allow a 2% increase in DB Calls to previous runs
    warning:
    - criteria:
      - "<=10"  # We expect no more than 10 DB Calls per TX
total_score:
  pass: "90%"
  warning: "75%"
```

1 `$ keptn configure monitoring <tool>`

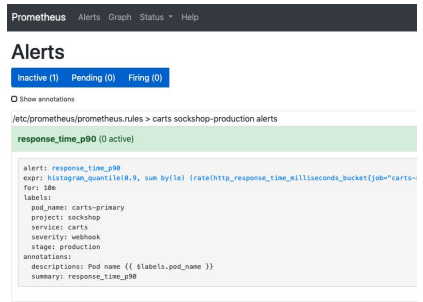
keptn Control Plane

Core — keptn Autonomous Cloud Control Plane

2 Distribution of cloud events

3 Tool specific interpretation and execution

Grafana | Prometheus | Dynatrace | Tool X ...



SLI/SLO-based evaluation implementation in Keptn

keptn

SLIs defined per SLI Provider as YAML

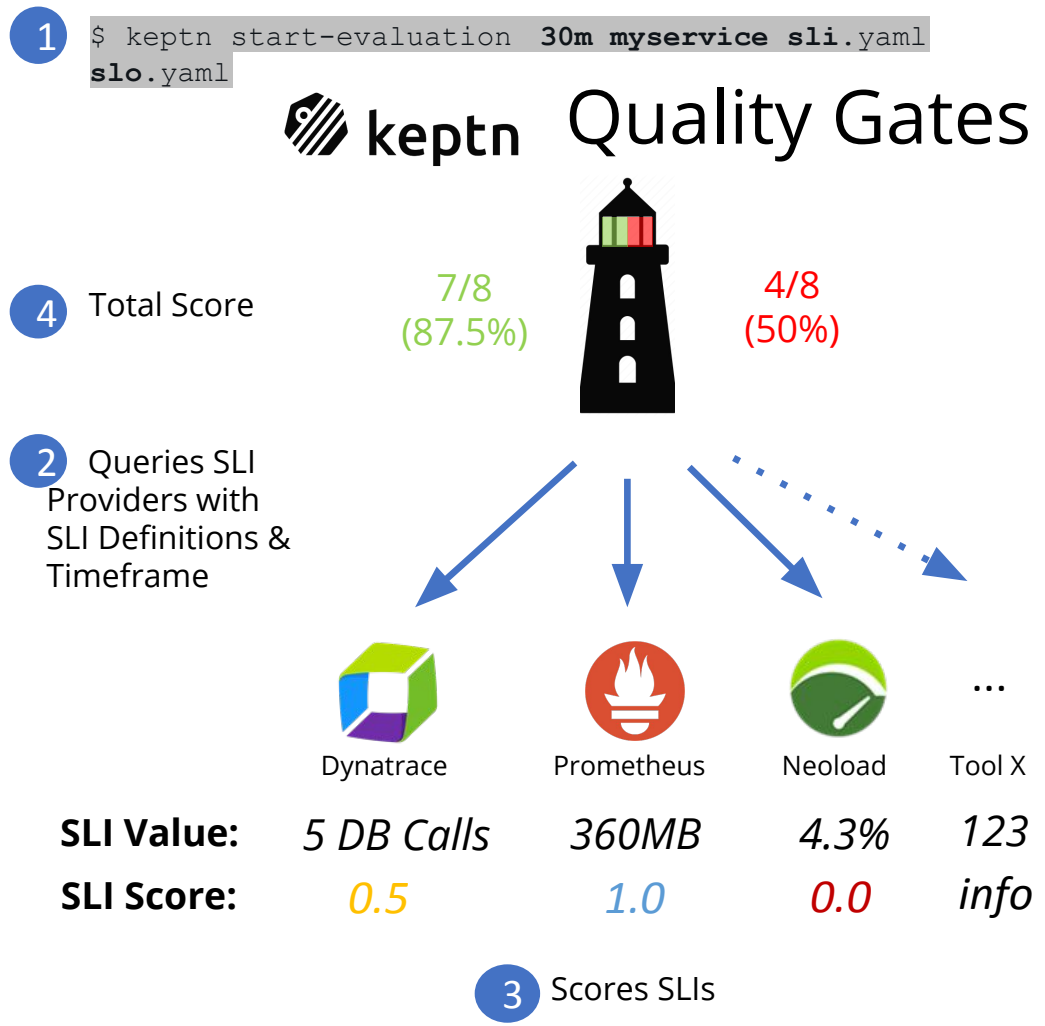
SLI Provider specific queries, e.g: Prometheus Metrics Query

```
indicators:
  error_rate:          "sum(rate(http_requests_total{job='...'}"
  count_dbcalls:       "sum(rate(http_requests_total{job='...'}""
  response_time_p90:   "histogram_quantile(0.90, sum(rate("
```

SLOs defined on Keptn Service Level as YAML

List of objectives with fixed or relative pass & warn criteria

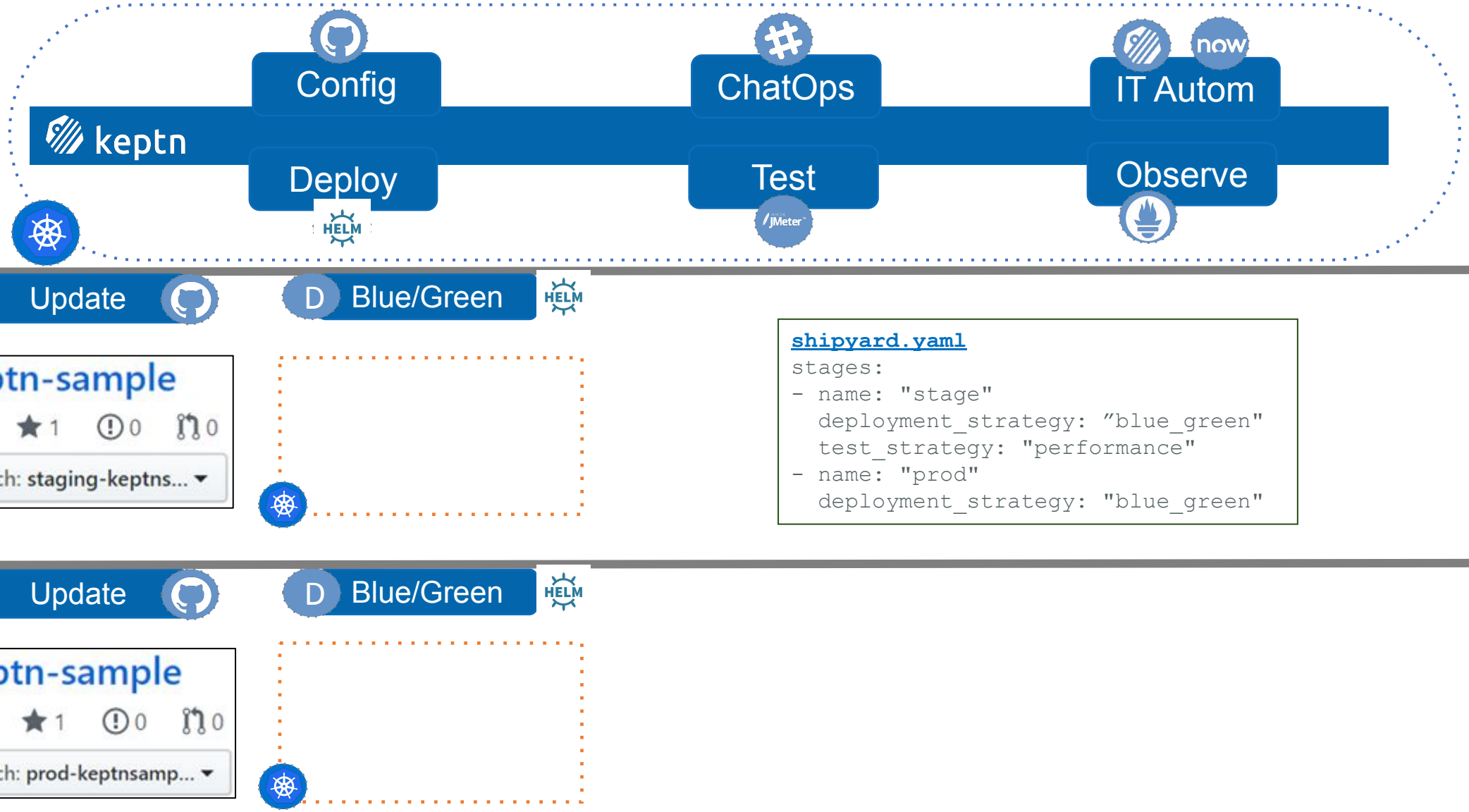
```
objectives:
  - sli: error_rate
    pass:
    - criteria:
      - "<=1“  # We expect a max error rate of 1%
  - sli: response_time_p90
  - sli: count_dbcalls
    pass:
    - criteria:
      - "=+2%"  # We allow a 2% increase in DB Calls to previous runs
    warning:
    - criteria:
      - "<=10"  # We expect no more than 10 DB Calls per TX
total_score:
  pass: "90%"
  warning: "75%"
```



Re-Think Pipelines:

```
$ keptn create project keptn-sample shipyard.yaml
```

keptn



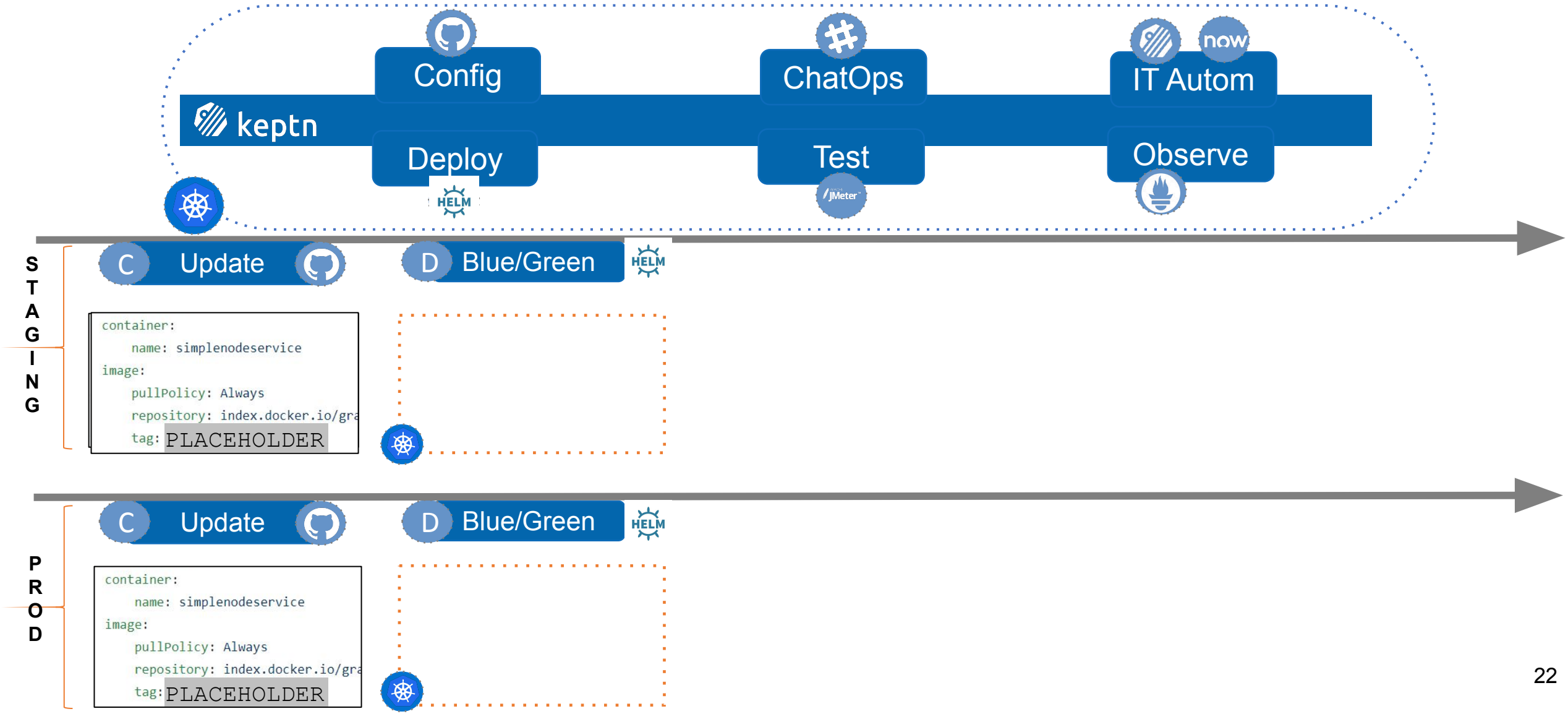
shipyard.yaml

```
stages:
- name: "stage"
  deployment_strategy: ”blue_green"
  test_strategy: "performance"
- name: "prod"
  deployment_strategy: "blue_green"
```

Zero-Touch Cloud Native Services:

```
$ keptn onboard service carts ... --chart=...
```

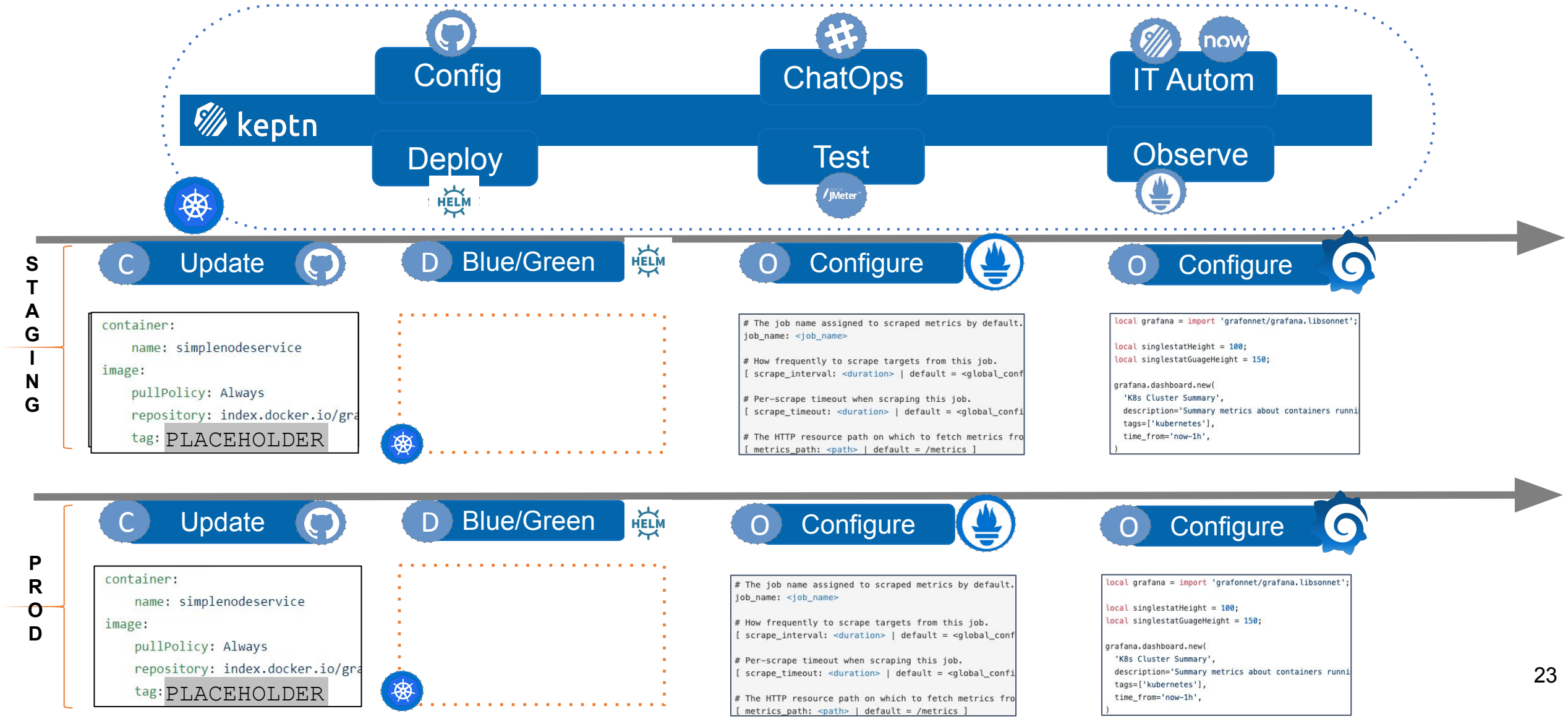
keptn



Automated Configuration of Monitoring & Dashboarding

```
$ keptn configure monitoring prometheus --service= --project=
```

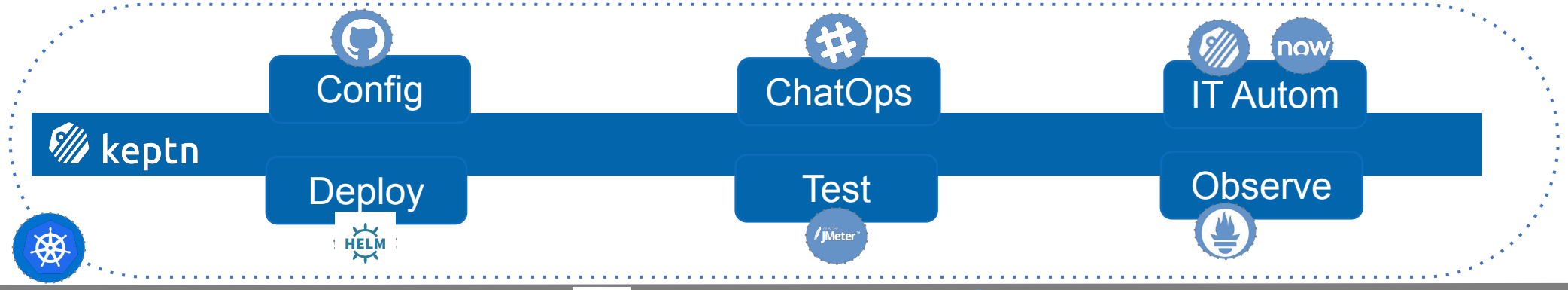
keptn



Cloud Native Delivery & Automated Quality Gates

```
$ keptn send event new-artifact --service= --project= --image=carts:0.10.2
```

keptn

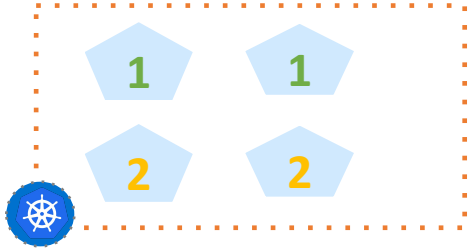


STAGING

C Update

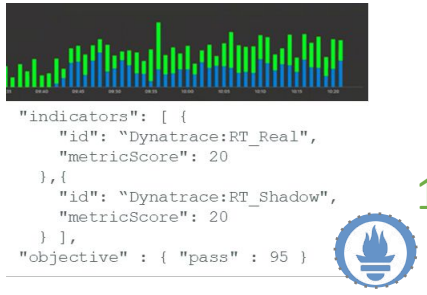
```
container:
    name: simplenodeservice
image:
    pullPolicy: Always
    repository: index.docker.io/gra
    tag: 0.10.2
```

D Blue/Green



T Performance

O Score



89 / 100

Promote?

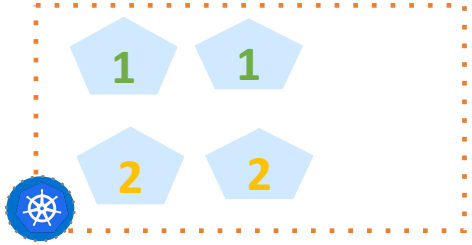
PROMOTE

PROD

C Update

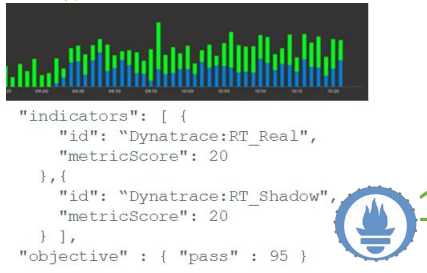
```
container:
    name: simplenodeservice
image:
    pullPolicy: Always
    repository: index.docker.io/gra
    tag: 0.10.2
```

D Blue/Green



T

O Score



95 / 100

Keep?

KEEP

Prometheus Integration in Keptn

https://github.com/keptn-contrib/prometheus-service

Prometheus Service

release v0.3.4 | build passing | go report A

The *prometheus-service* is a Keptn service that is responsible for

1. configuring Prometheus for monitoring services managed by Keptn, and
2. receiving alerts from Prometheus Alertmanager and translating the alert payload to a cloud event that is sent to the Keptn eventbroker.

Compatibility Matrix

Please always double check the version of Keptn you are using compared to the version of this service, and follow the compatibility matrix below.

Keptn Version	Prometheus Service Image
0.5.x	keptn/prometheus-service:0.2.0
0.6.x	keptn/prometheus-service:0.3.0
develop	keptn/prometheus-service:latest

https://github.com/keptn-contrib/prometheus-sli-service

Prometheus SLI Service

release v0.2.2 | build passing | go report A+

This service is used for retrieving Service Level Indicators (SLIs) from a Prometheus API endpoint. Per default, it fetches metrics from the prometheus instance set up by Keptn (`prometheus-service.monitoring.svc.cluster.local:8080`), but it can also be configures to use any reachable Prometheus endpoint using basic authentication by providing the credentials via a secret in the `keptn` namespace of the cluster.

The supported default SLIs are:

- throughput
- error_rate
- response_time_p50
- response_time_p90
- response_time_p95

The provided SLIs are based on the RED metrics

https://github.com/keptn-sandbox/grafana-service

Grafana Service

release no releases or repo not found | go report A

Example of a dashboard automatically generated by the Keptn Grafana Service.

Resources

keptn

- Blog: Overcoming scalability issues in your Prometheus ecosystem
- Code generators (examples):
  - https://github.com/metalmatze/slo-libsonnet
  - https://github.com/grafana/grafonnet-lib
  - https://promtools.matthiasloibl.com/
  - https://gitlab.com/gitlab-com/runbooks/-/tree/master/metrics-catalog
  - https://github.com/line/promgen
  - https://monitoring.mixins.dev/
- Presentation on the RED method

Join the **Keptn community**

keptn

- Overview https://github.com/keptn/community
- Github https://github.com/keptn/keptn
- Website https://keptn.sh
- Google Group https://groups.google.com/forum/#!forum/keptn
- Twitter https://twitter.com/keptnProject

Thank you!

Jürgen Etzlstorfer

DevOps Activist @ Dynatrace

@jetzlstorfer

https://www.linkedin.com/in/juergenetzlstorfer

Hands-On @ https://tutorials.keptn.sh

Follow us @keptnProject

Star us @ https://github.com/keptn/keptn

Visit us @ https://keptn.sh

Join the Keptn Community via
https://github.com/keptn/community