SLX

An Extended SLO Framework to Expedite Incident Recovery

Qian Ding, Xuan Zhang

Ant Group

About Us

- Infra SRE
- Managing 60+ Kubernetes Clusters with 60K+ Nodes
- Other Infra Services (DNS, Gateway, Pubsub)
- Firefighting and 99.999% availability target

Incident recovery is hard.

Definition of incident is often **ambiguous**.

#Motivation 1

Complex interactions between system boundaries

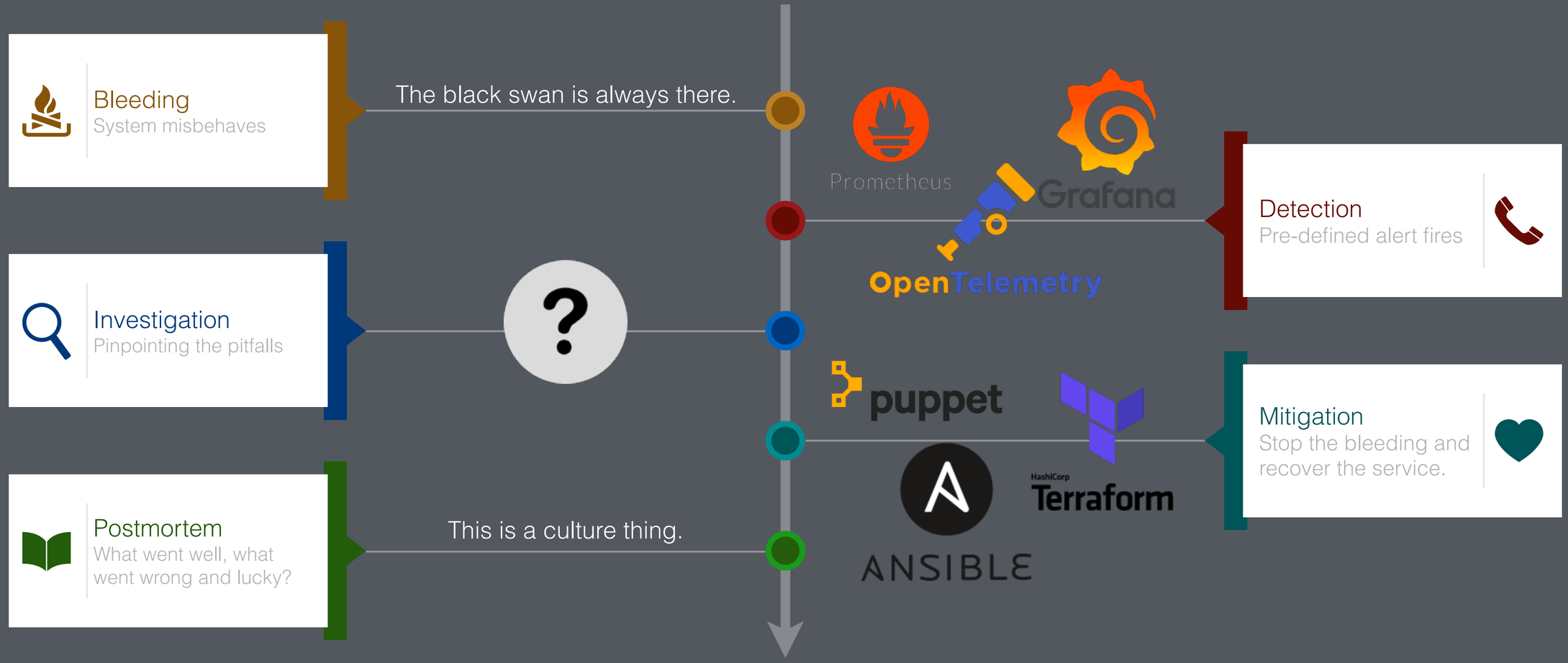
#Motivation 2

Feeling **lost** in Grafana panels & dashboards

#Motivation 3

Intro

Incident Lifecycle



Shortening the investigation time

#Goal

Too many metrics & logs & traces

#Challenge 1

SLO

Metrics building

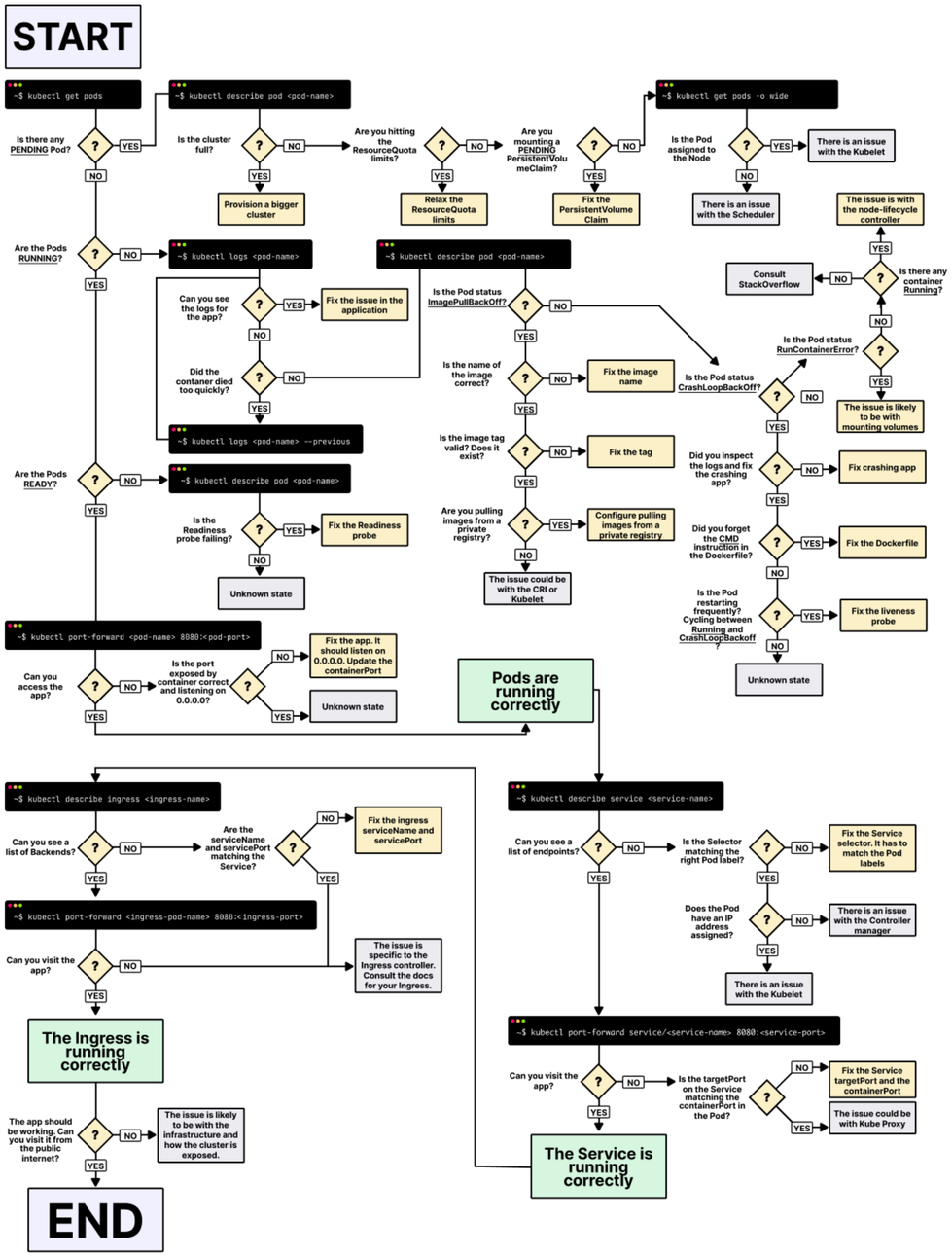
Single-service perspective

SLO reporting

Good for detection

High cognitive load on cross-service investigation

#Challenge 2

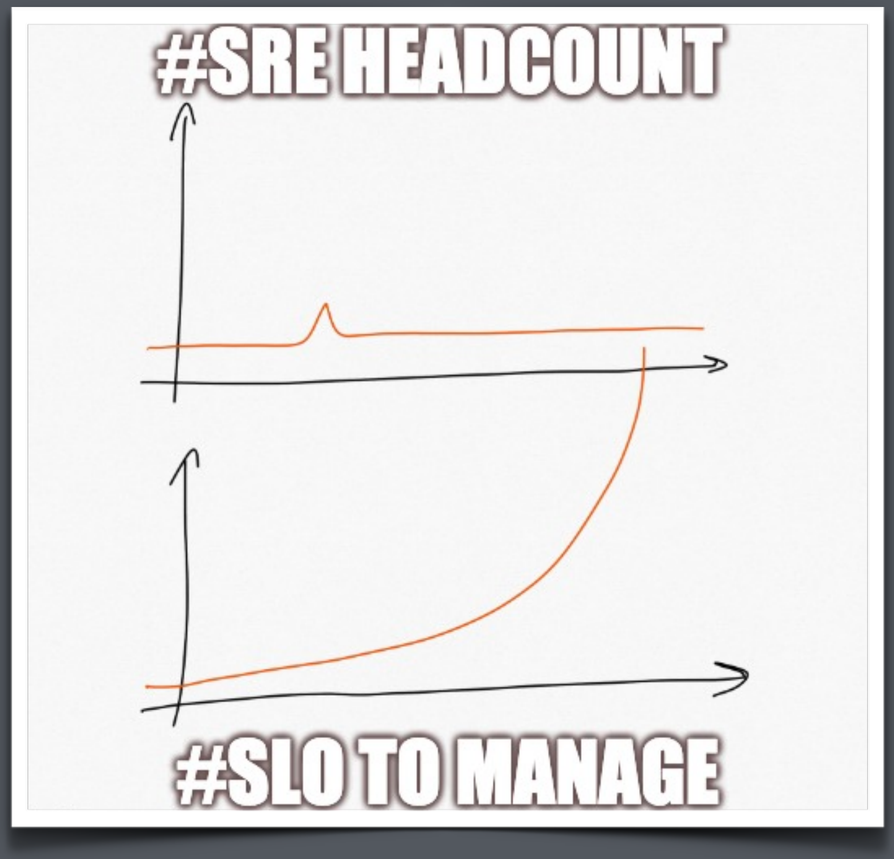


How does an experienced SRE **debug** massive **K8s pod** creation **failures**?

#Apiserver #500 #quota
#network #connectivity
#docker #registry
#DNS #3p-party operator
#ratelimit #release
...

Leveraging on SLO data for investigation

#Challenge 3



How many **SLOs** to pay **attention** to?

50 Components

×

5 Key User Journeys

×

3 Types of basic SLOs

×

3 Different Execution Environment

=

?? SLOs

SLX

Light-weighted SLO Management

+

Automatic Time-series Anomaly Detection and Correlation

SLX Framework

SLX Intro

SLI Service Level Indicator

```
http_request_error_ratio # (dc, api) as tags
```

SLO Service Level Objective

```
max_over_time(http_request_error_ratio{api=~“a|b”}[30d]) < 0.001
```

SLA Service Level Agreement

```
max_over_time(http_request_error_ratio{api=~“a|b”}[30d]) < 0.01
```

SLF Service Level Factor

```
http_request_error_ratio{api=~“a”,dc=“foo”}
```

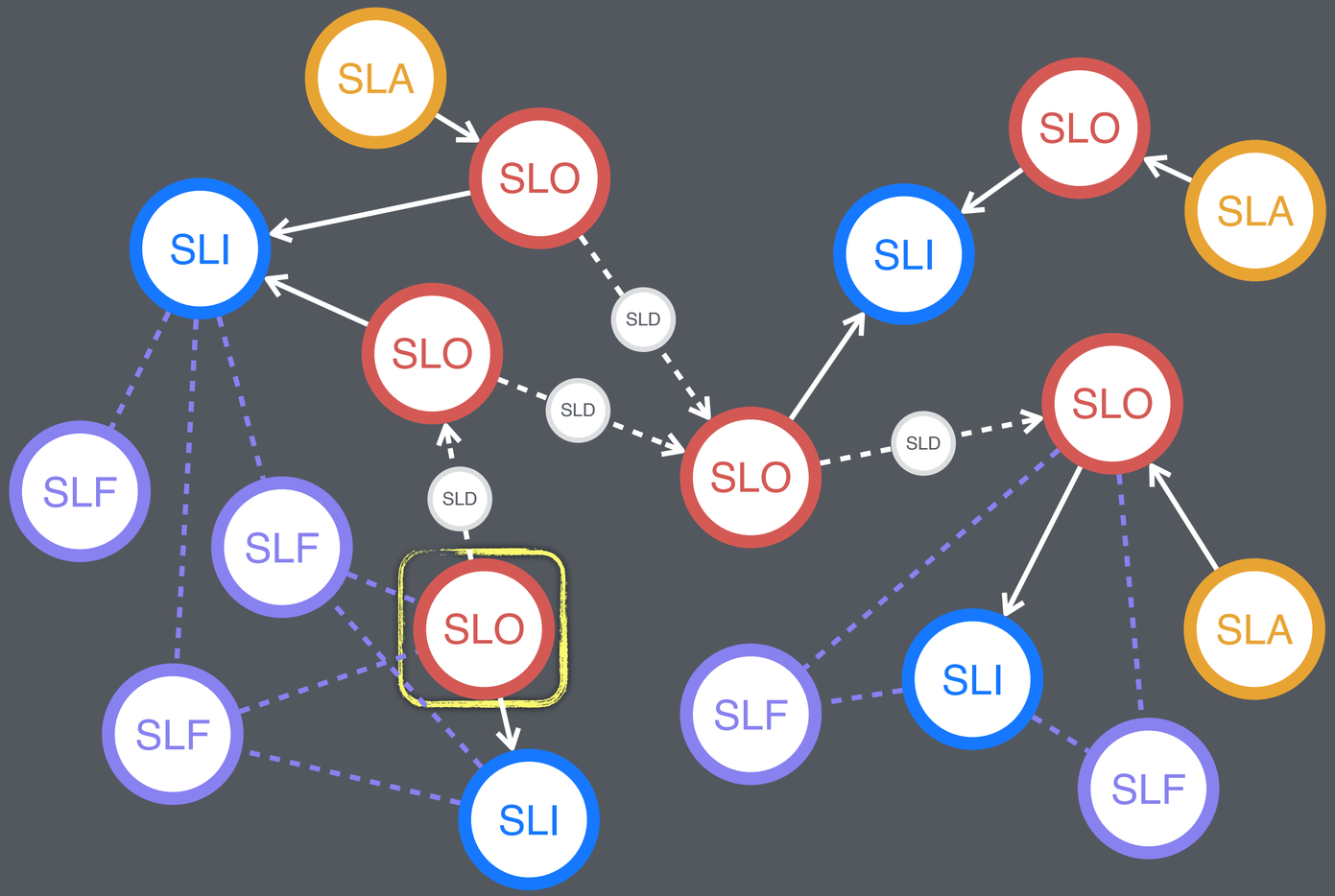
SLD Service Level Dependency

```
processed_cpu_seoncds_count{api=“a”,dc=~“foo”}
rpc_outgoing_request_error_ratio{api=~“a”,rpc=“bar”}
```

SLX Framework

SLX Graph

A **denpendency** graph

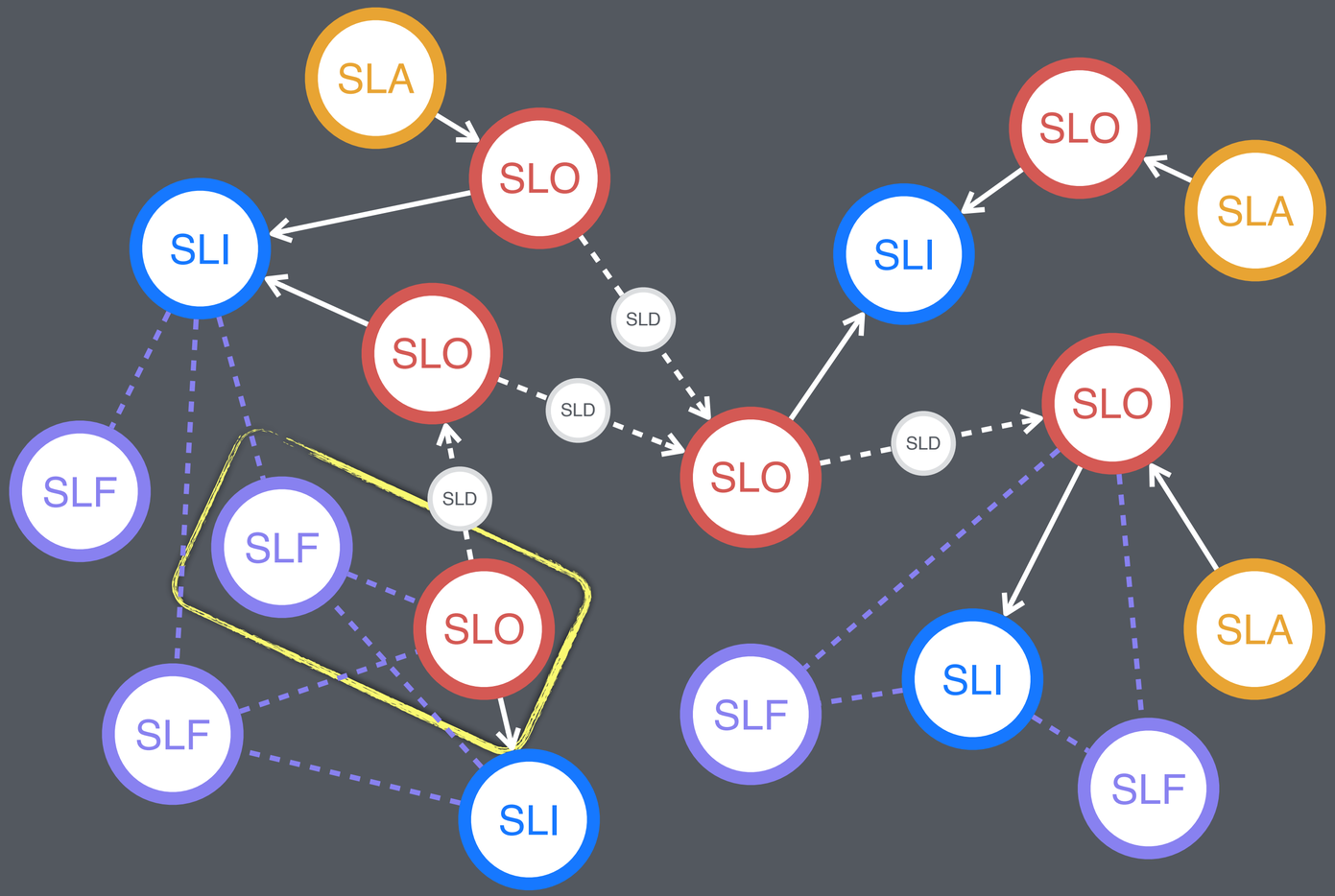


SLX Framework

SLX Graph

Locate the burning **SLO**

with the **abnormal** SLF

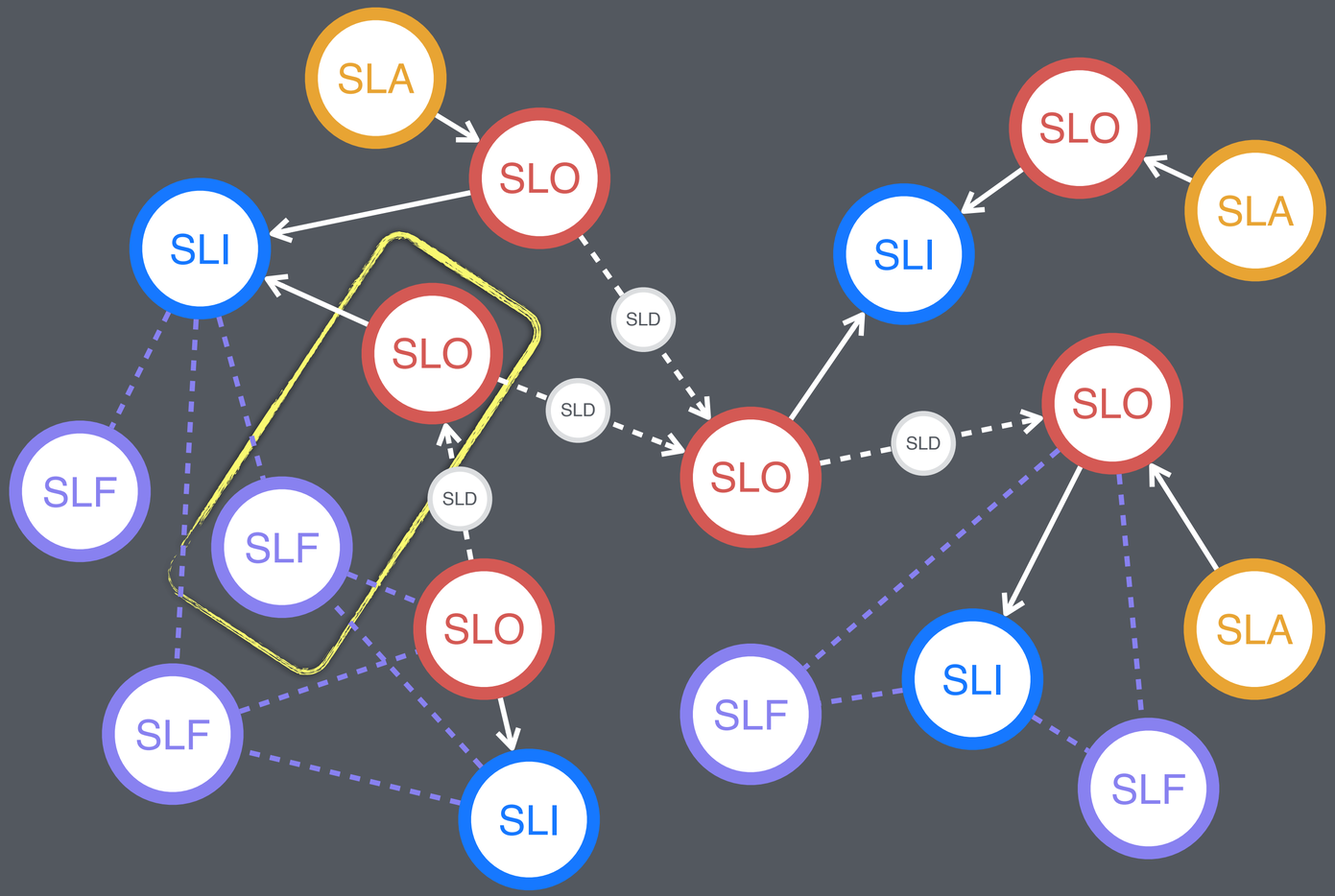


SLX Framework

SLX Graph

Locate the burning **SLO**

with the **abnormal** SLF



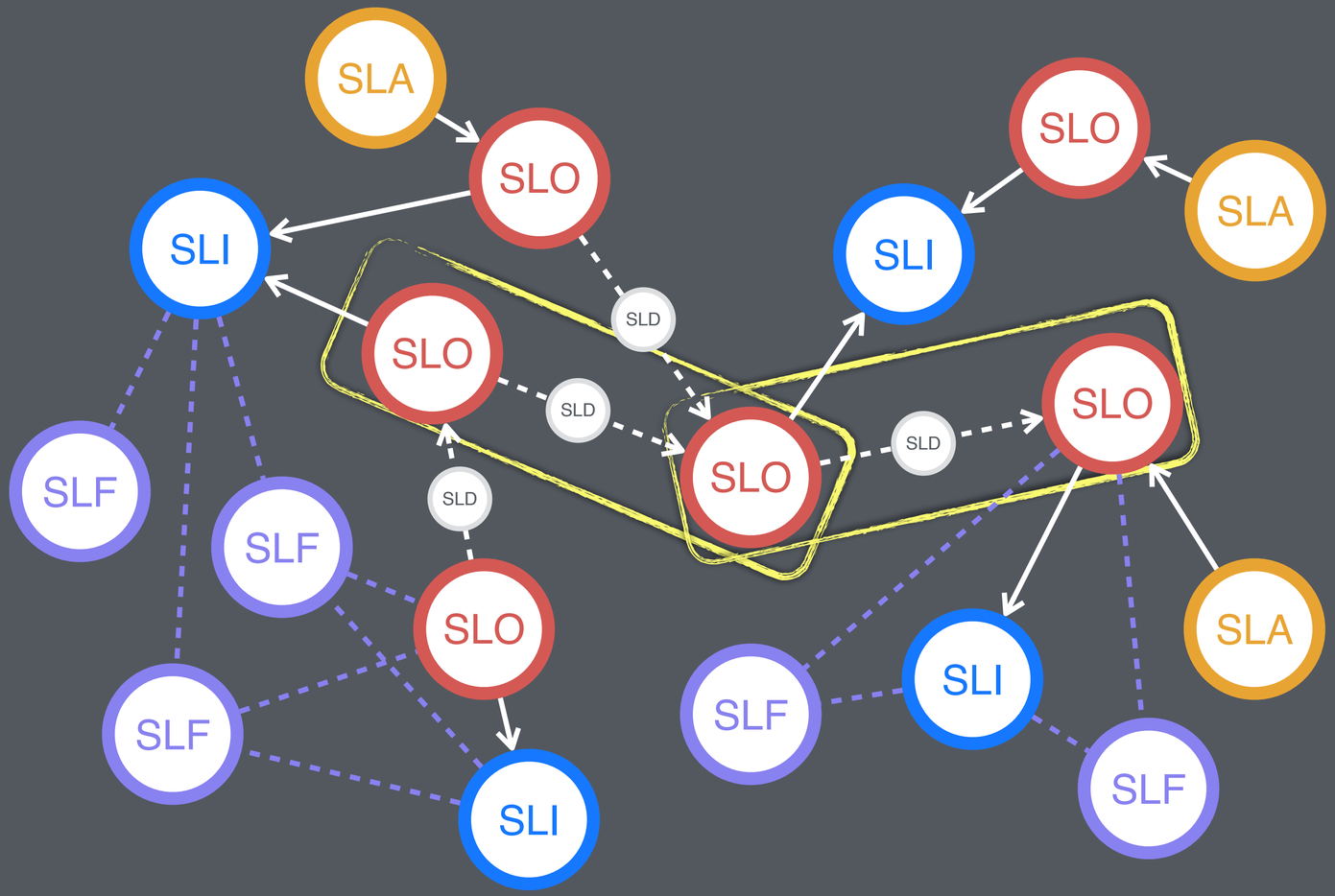
SLX Framework

SLX Graph

Graph traversal

to find the **time-correlated**

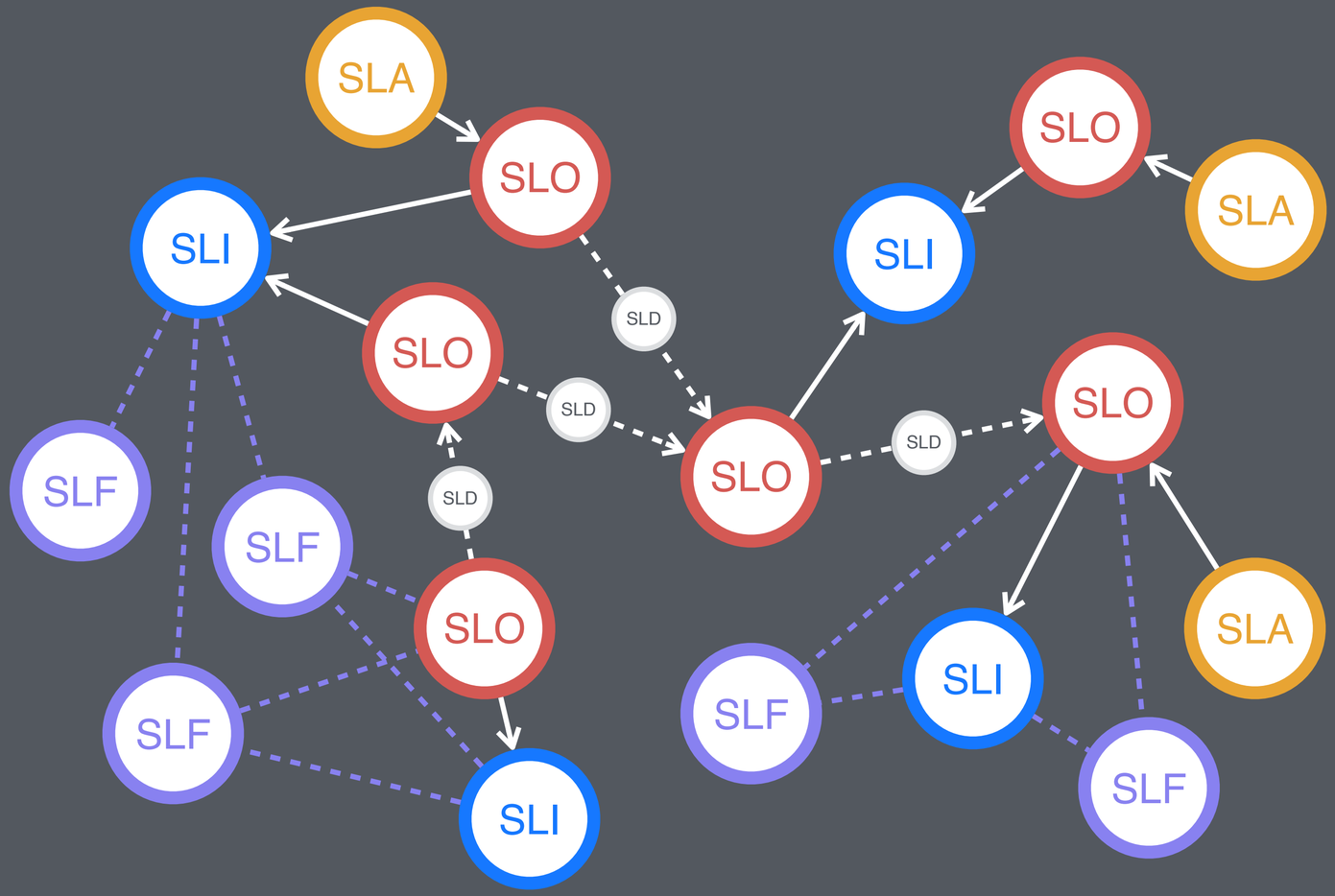
abnormal SLO **dependencies**



Design & Implementation

Design & Implementation

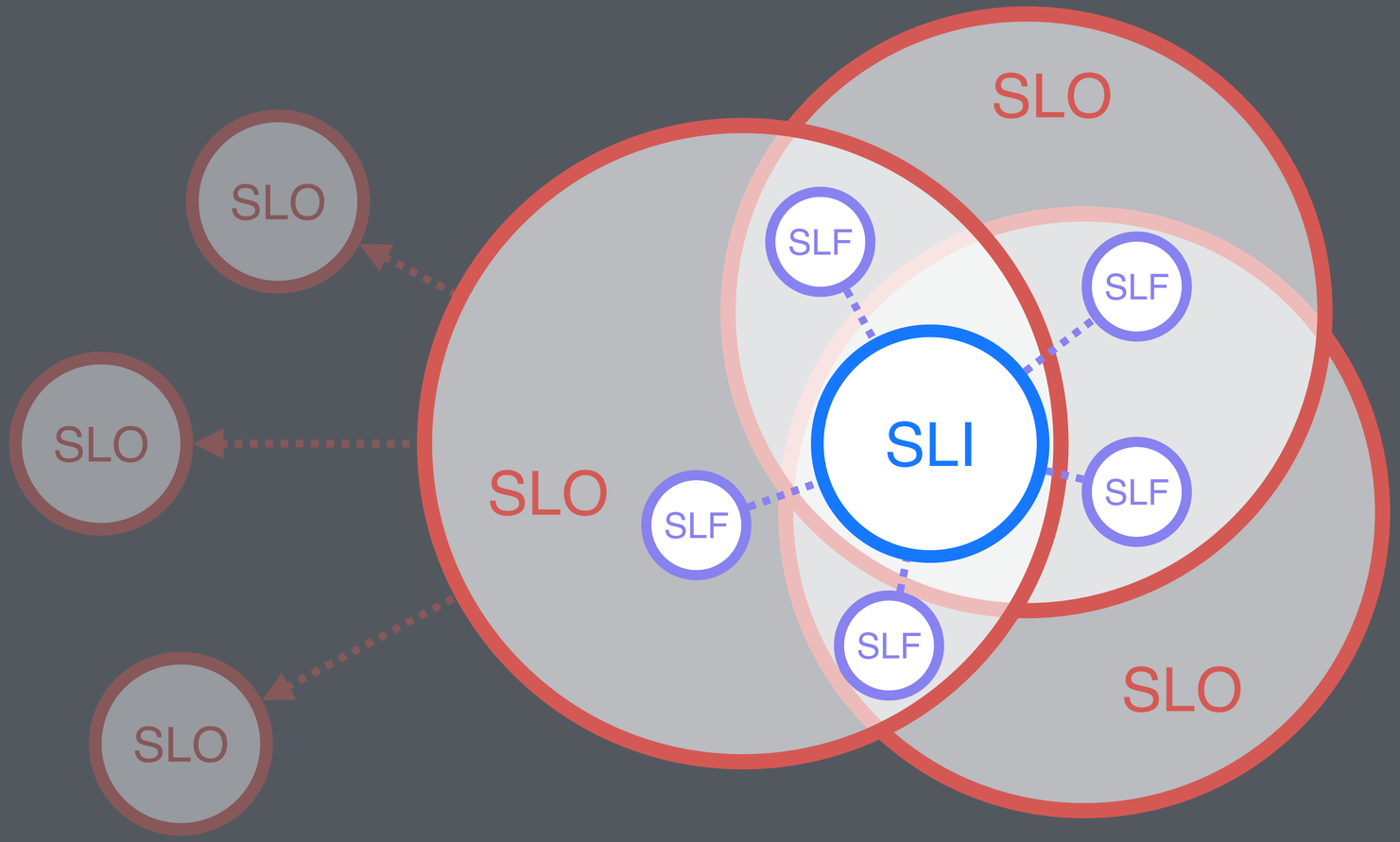
Intro



- Cluster of definitions
- Unmanageable relations
- Constant changes

Design & Implementation

SLX Definitions



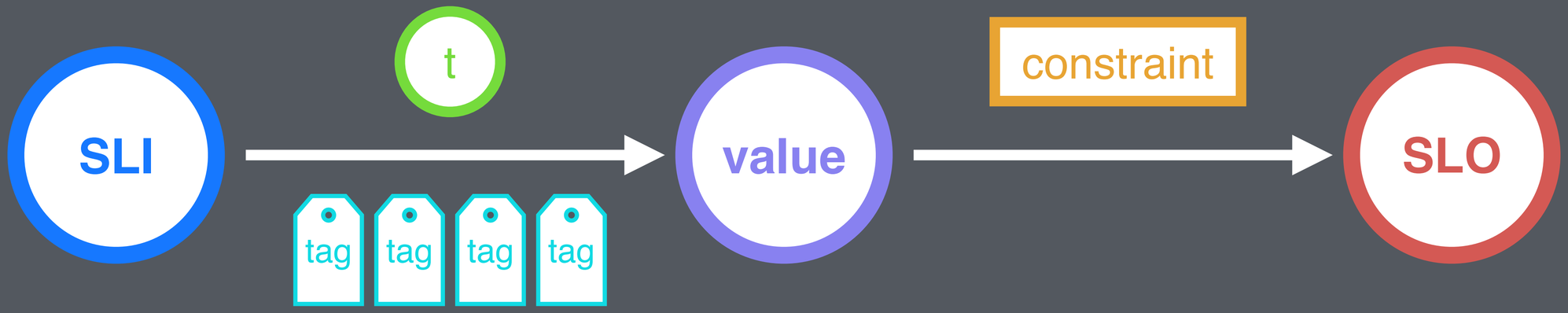
- SLI is the core
- SLI can be reused

Design & Implementation

SLX Definitions

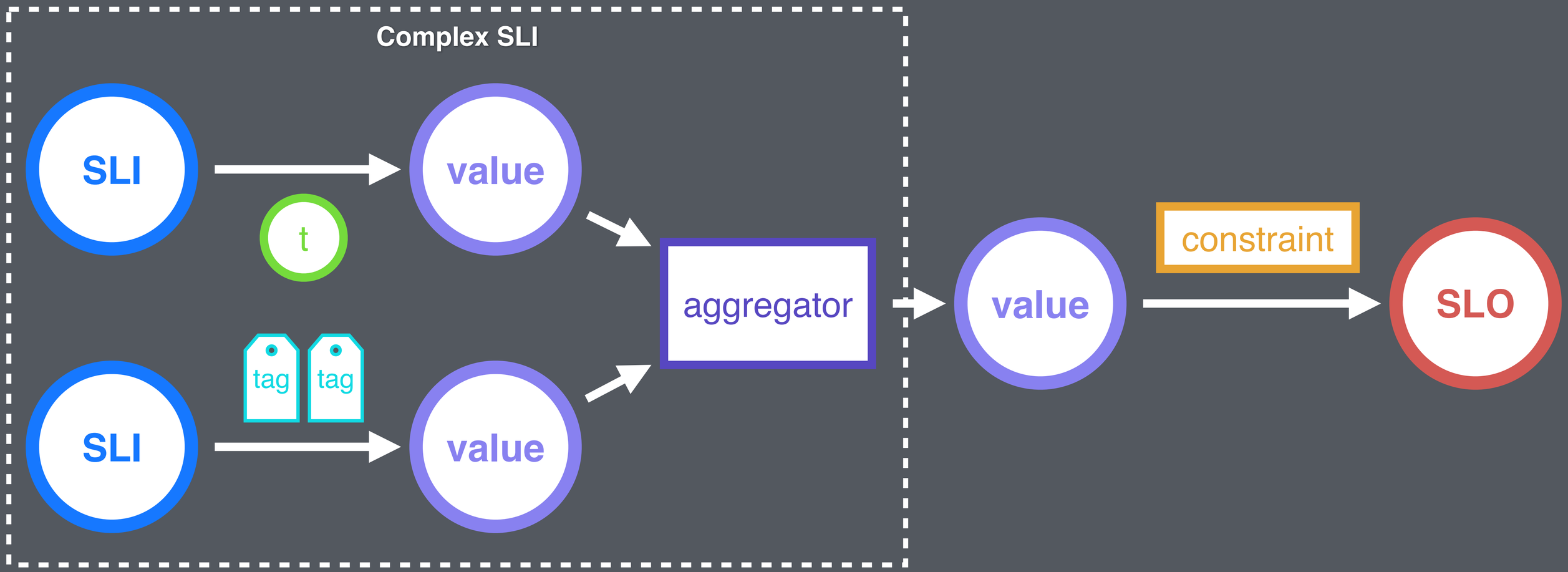
SLI = metric = time series

SLO = SLI + constraint



Design & Implementation

SLX Definitions



Design & Implementation

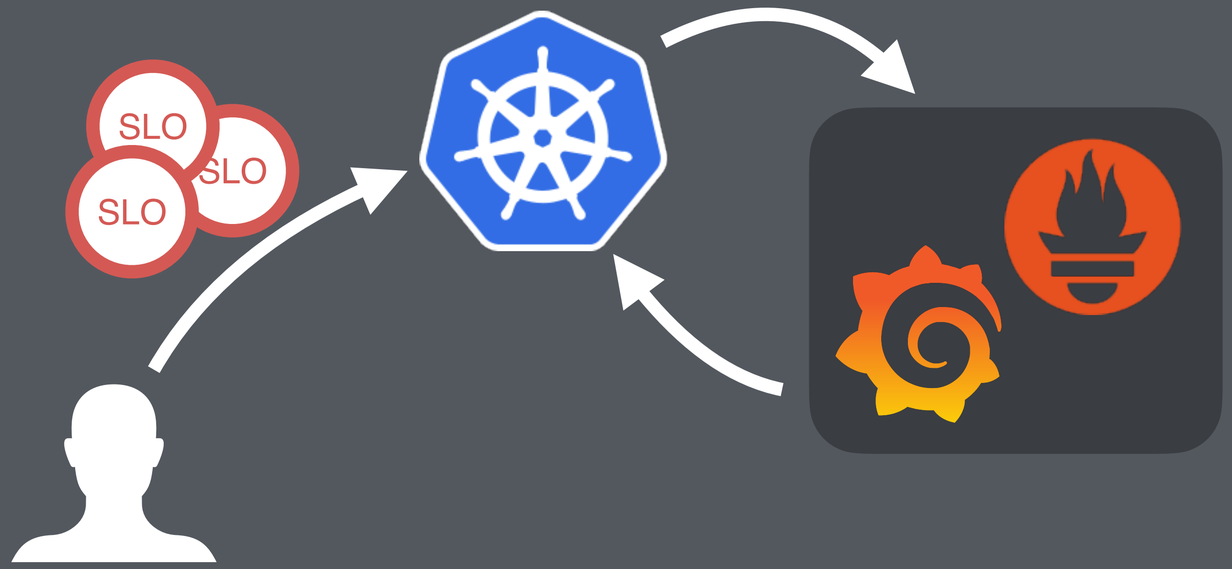
SLX Definitions

```yaml
name: Pod Creation SLO
sli:
  name: Pod Creation Success Ratio
  type: SuccessRatio
  meas:
    error:
      name: Pod Creation Failures
      type: Simple
      meas: promql:increase(slo_pod_startup_result...
    total:
      name: Pod Creations
      type: Simple
      meas: promql:increase(slo_pod_startup_result...
```

```yaml
objectives:
  - name: Pod Creation Success Ratio 1h
    interval: 30d
    type: rolling
    alerts:
      - name: TooManySLOFailuresInAnHour
        active: true
        severity: p0
        trigger:
          type: ErrorBudget
          threshold: 0.99
          window: 1h
          windowBudgetRatio: 0.03
          minWindowBudget: 15
```

Design & Implementation

Automation



- Declarative API
- Automated Reconciliation

Submit our SLOs
Let the Kubernetes take the wheel

Design & Implementation

GitOps

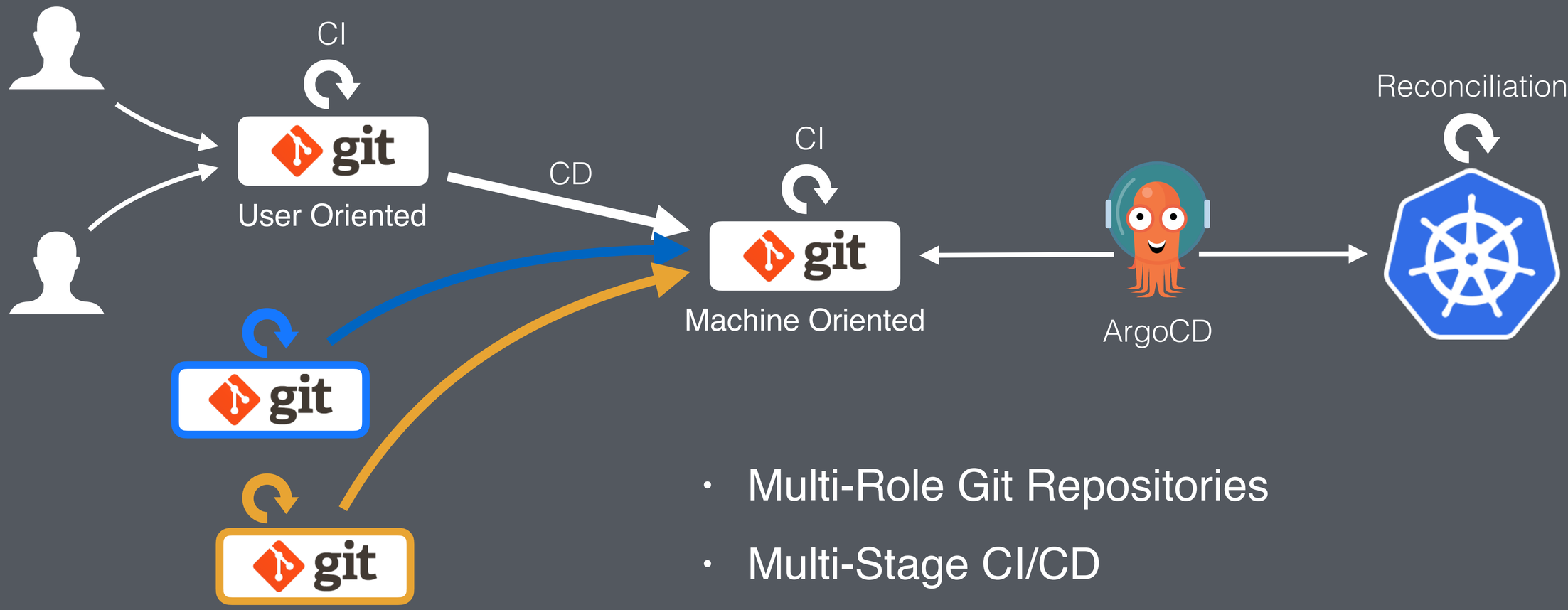
ArgoCD

- Commit history
- Code Review
- Conflict Resolution

- Automated Sync
- Resource Management
- Web App

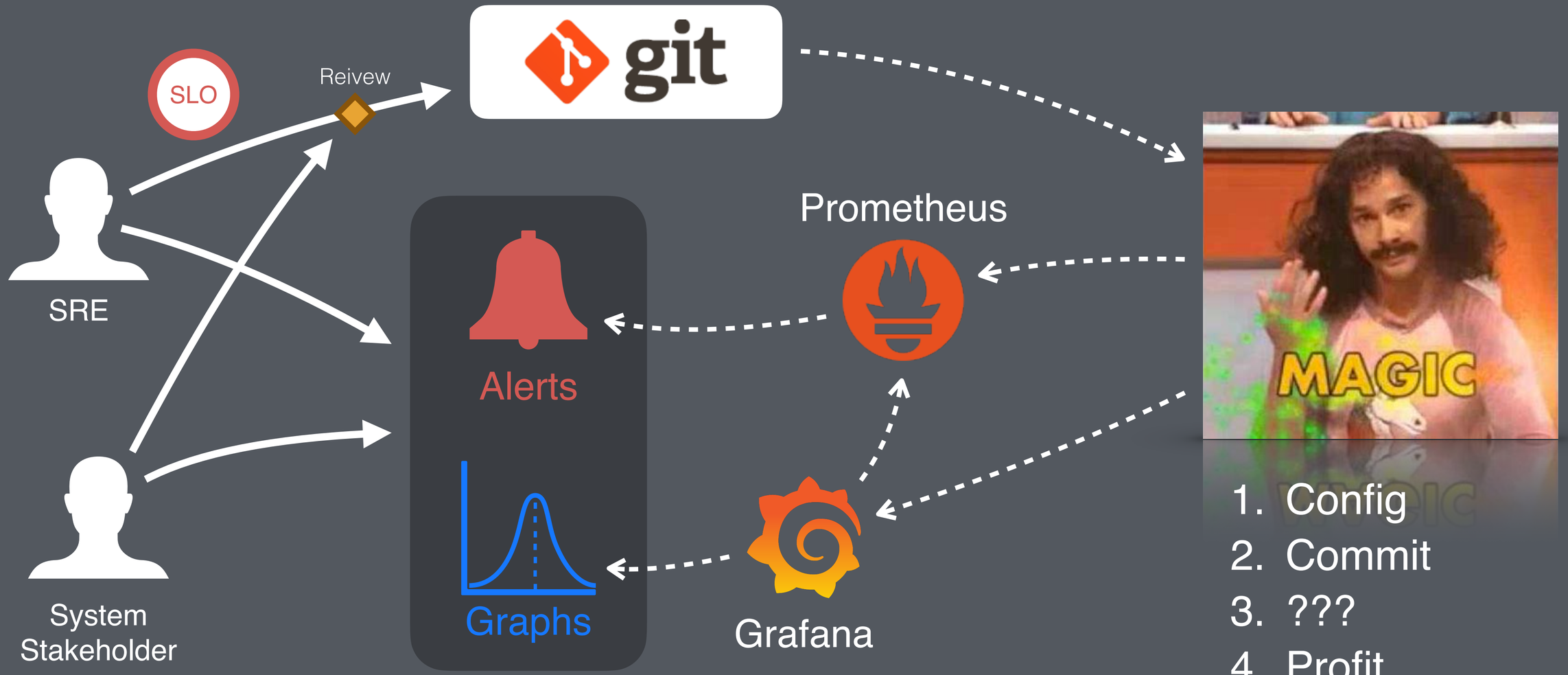
- Reconciliation Loop
- Error Handling
- Deployments

Design & Implementation
GitOps

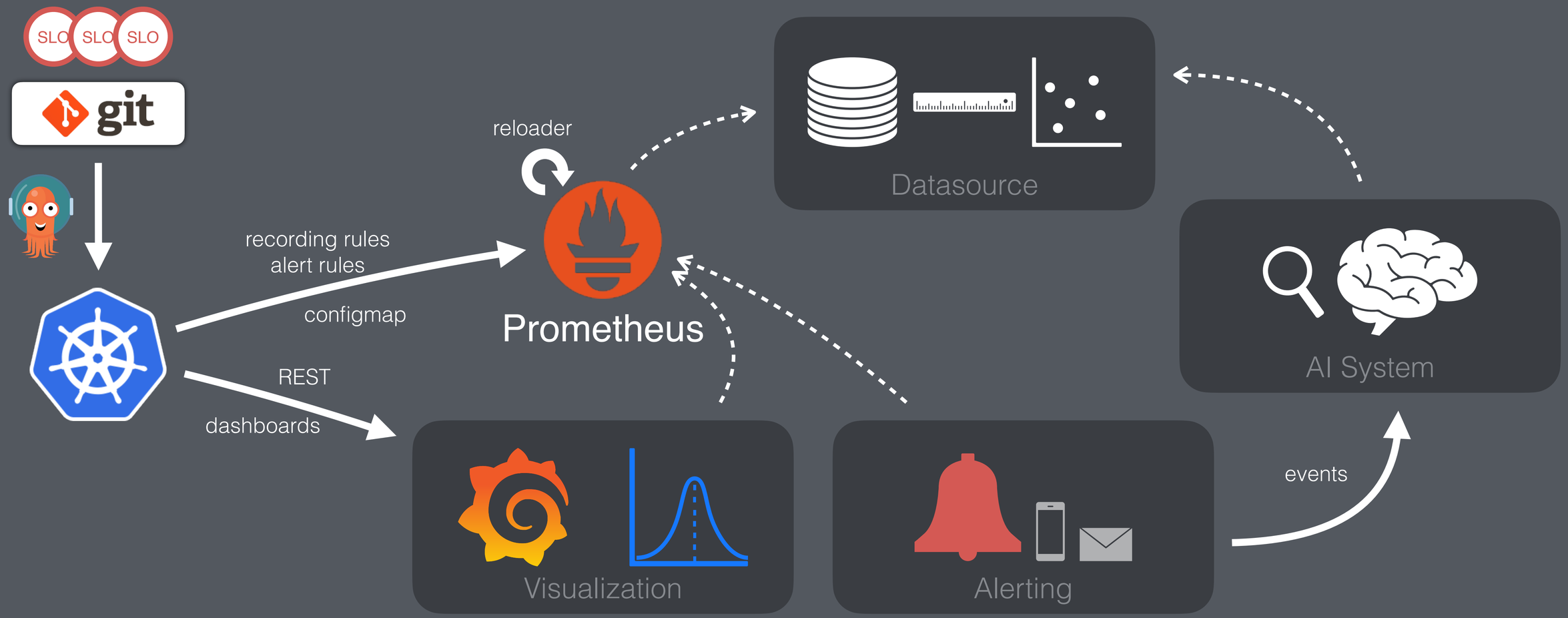


- Multi-Role Git Repositories
- Multi-Stage CI/CD

Design & Implementation
Workflow



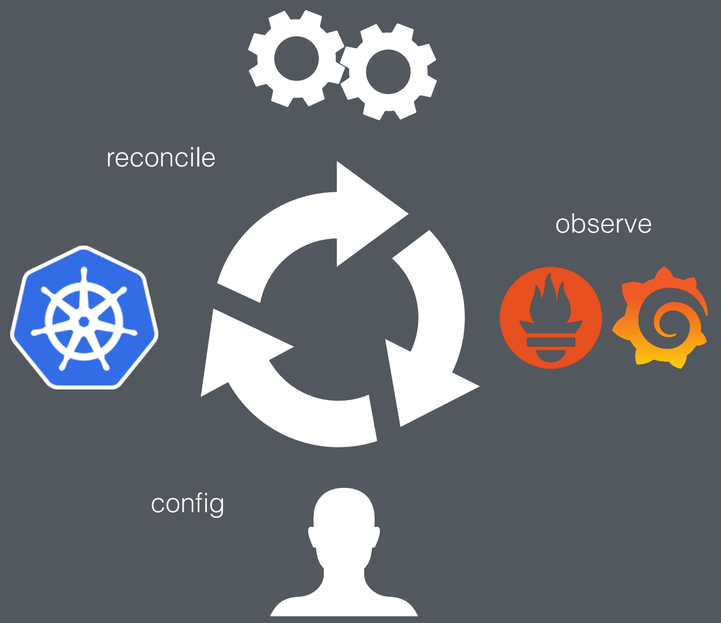
Design & Implementation
System Design



Design & Implementation

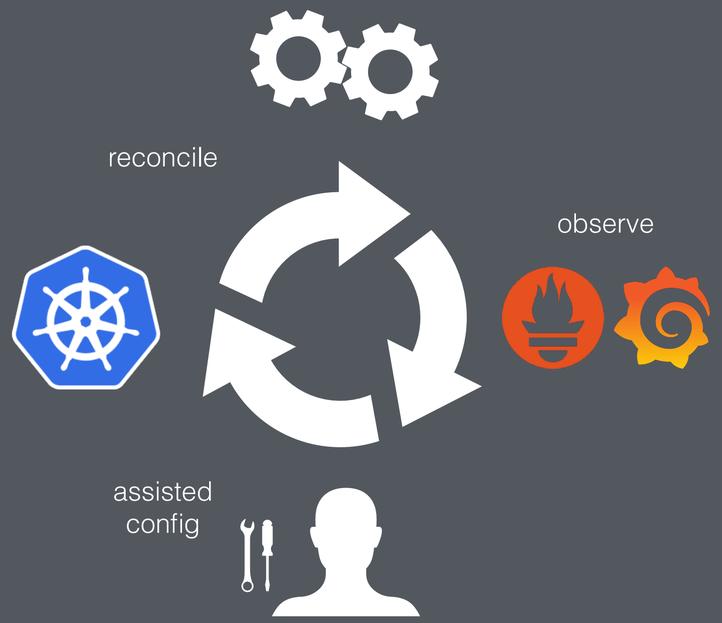
Dashboards

Design & Implementation
What's Next



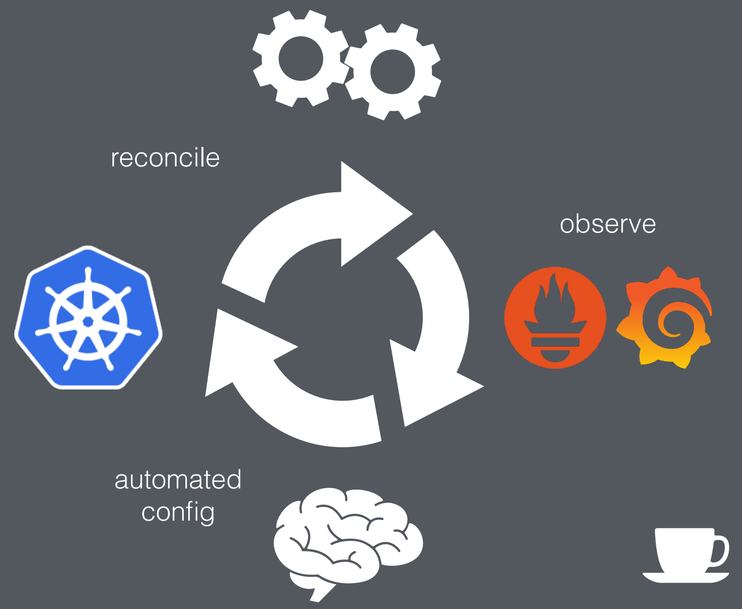
L0 Autopilot

- No driving automation
- Manually controlled
- Alerts before system failure



L1 Autopilot

- Driver Assistance
- System failure analysis
- Response Recommendations

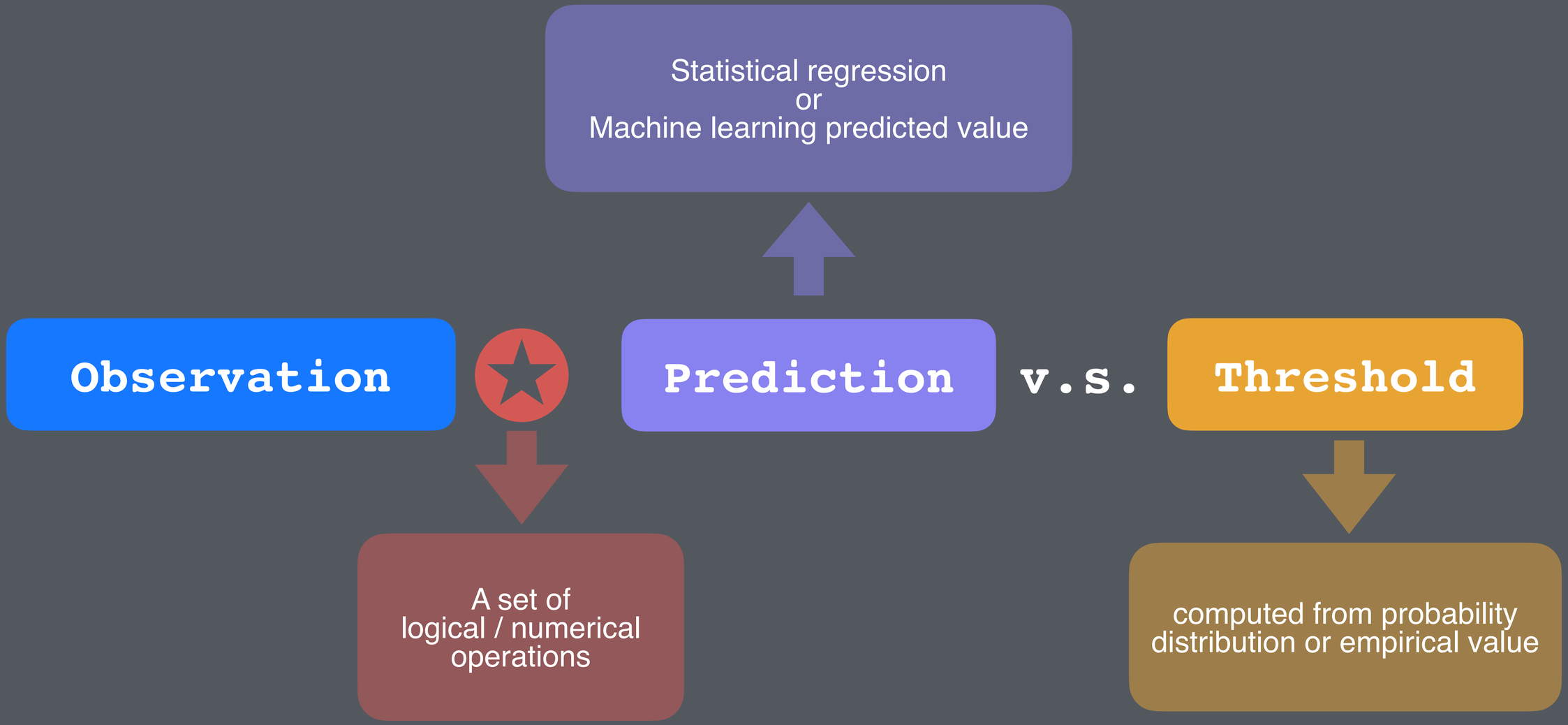


L2+ Autopilot

- Driving automation
- SLO as a reconciliation target
- No hands required

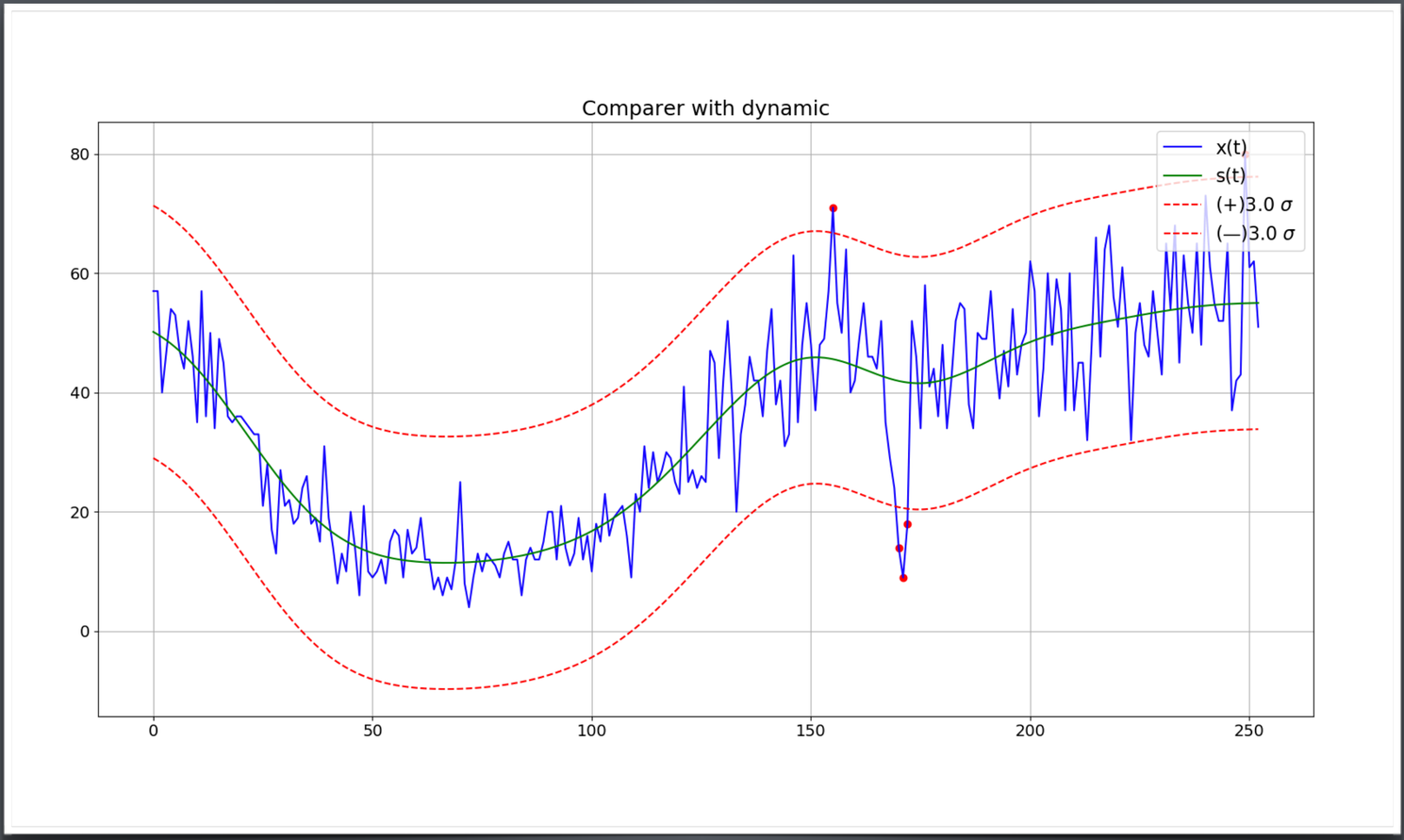
Design & Implementation

Anomaly Detection



Design & Implementation

Anomaly Detection



Underestimate

the importance of

data quality.

#pitfall 1

Highest Cardinality Labels

Name	Count
container_id	4343722
uid	2870959
pod	2460040
key	1325130
pod_ip	860952
pod_name	735653
container_name	604847
container_sn	603127
container_hostname	601593
ip	372114

Watch out

for curse of

dimensionality.

#pitfall 2

Obsessed with (hyper-)parameters in the algorithm

and ignoring

useability.

#pitfall 3

Summary

Key Take-Aways

- SLO is not a silver bullet, it takes time to iterate to become a universal language
- Automation and standardization make things scalable and manageable
- Put the user first: algorithms and machine learning should not steal the spotlight of user-experience

Thank you

#End

Presented By

Qian Ding
sumang.dq@antgroup.com

Xuan Zhang
shenchen.zx@antgroup.com