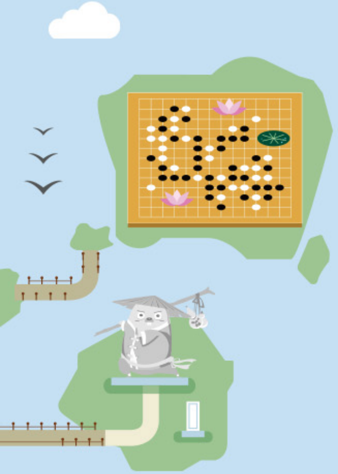
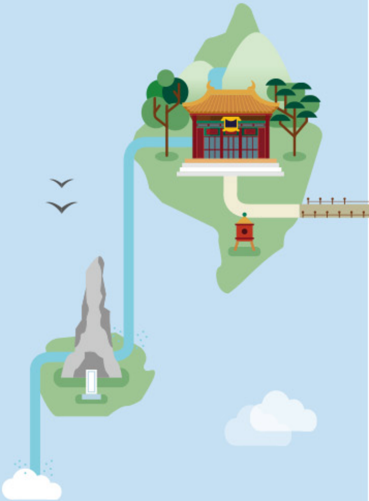


ContainerOps – DevOps Orchestration



GopherChina 2017

Quanyi Ma

DevOps & Open Source Expert
Senior Architect & Full Stack Developer

Email: maquanyi@huawei.com
Twitter: @genedna
Github: https://github.com/genedna

Agenda

1. The story of DevOps
2. What’ s the DevOps
3. The ContainerOps Concept – Defining/Drawing/Running
4. What’ s the DevOps Component
5. The ContainerOps core – DevOps Workflow Engine
6. A user case of ContainerOps

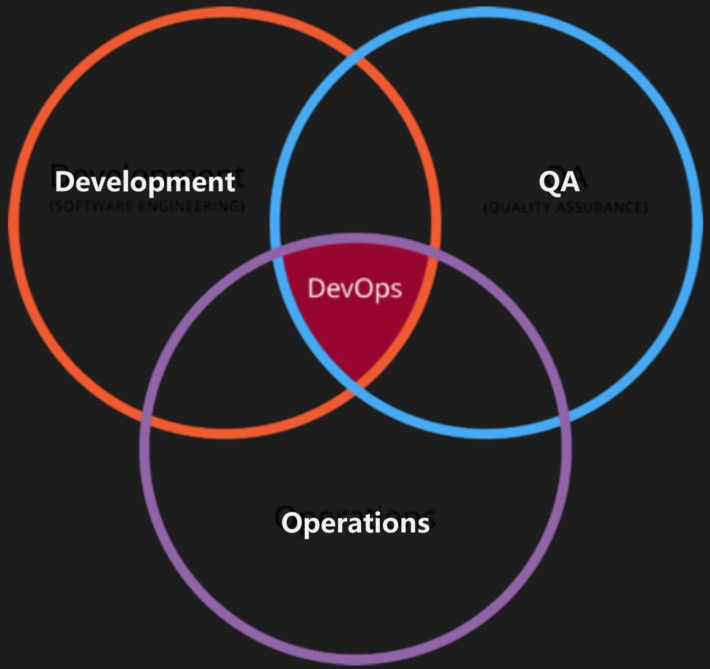
DevOps Story

- **2007** - While consulting on a data center migration for the Belgium government, system administrator Patrick Debois becomes frustrated by conflicts between developers and system admins. He ponders solutions.
- **Agile Conference 2008 in Toronto** - Andrew Clay Shafer’s “***birds of a feather***” ad hoc session called Agile Infrastructure. The only person who showed up was Patrick Debois. Shafer and Debois started a Google group called “***Agile System Administration***”
- **O’Reilly Velocity 2009 Conference** - Presentation at Velocity of ***10+ Deploys per Day: Dev and Ops Cooperation at Flickr*** by John Allspaw and Paul Hammond – Debois watched by streaming video, tweeted.
- **October 2009** - Organized through Twitter. Conversation continued on Twitter and the #DevOps hashtag was born, dropping “***Days***” for brevity.
- **2010 Mountain View, CA** - ***DevOpsDays***
- **Mar. 2011** - Gartner’s first notes about DevOps
- **April 2012** - In an InfoQ video interview, Debois admitted that naming the movement was not as intentional as it might seem: “I picked ‘***DevOpsDays***’ as Dev and Ops working together because ‘***Agile System Administration***’ was too long,” he said. “***There never was a grand plan for DevOps as a word.***”

What’ s the DevOps?

DevOps is "a portmanteau of 'development' and 'operations'" and is "a software development method that stresses communications, collaboration, integration, automation and measurement of cooperation between software developers and other IT professionals".

-From Wikipedia



DevOps is an operational philosophy that promotes better communication between development and operations as more elements of operations become programmable.

What’ s the DevOps ultimate AIM?

Goal ->

The ultimate is break down barriers between developer, QAs and operators.

How ->

- Define the operation environment at development stage.
- Define the process from development to the production.
- Automate everything.

Why improve so hard?

- Don’ t break the original DevOps workflow.
- Add DevOps orchestration tool adaptive the workflow.
- Improve the process with customize DevOps task.
- Add DevOps service like Travis CI.
- Everyone is happy!!!

Håkan Forss @hakanforss http://hakanforss.wordpress.com
This illustration is inspired by and in part derived from the work by Scott Simmerman, “The Square Wheels Guy” http://www.performancemanagementcompany.com/

ContainerOps - DevOps Orchestration

Defining -> Component

Drawing -> Workflow

Running -> Container Orchestration

Open Source @ ->
https://github.com/Huawei/containerops

Defining Component - Container Image For DevOps

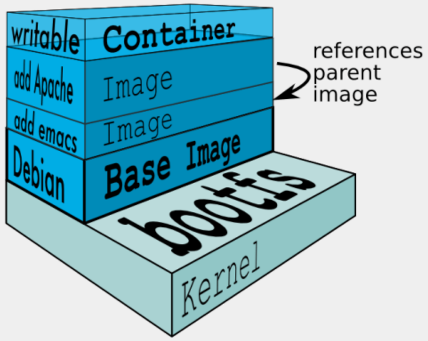
1. Encapsulating your DevOps task in a container image.
2. Defining the input & output data.
    a. Data type is Key/Value
    b. Input data with environment variables. Reserved environment variable name is [CO_DATA].
    c. Output data in the stdout/stderr, the reserved environment variable name is [CO_RESULT].
    d. Output data format [COUT] CO_RESULT = true/false .
    e. Base image is phusion/baseimage .
3. Management the component lifecycle in the ContainerOps system.

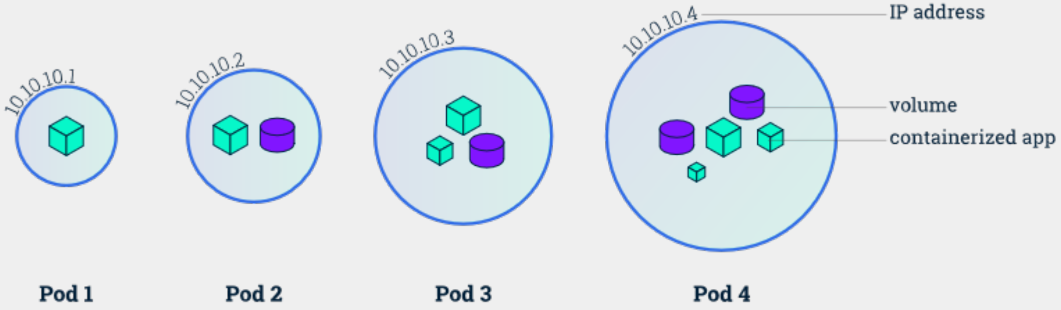
Why Use DevOps Component?

	Search	Download	Environment	Run
Plugin	Jenkins Wiki	Wget https://...	Follow README, prepare virtual machine and initialize the environments	Scheduling the VM to run the plugin
Component	Component Repository	Copy URL	N/A	N/A

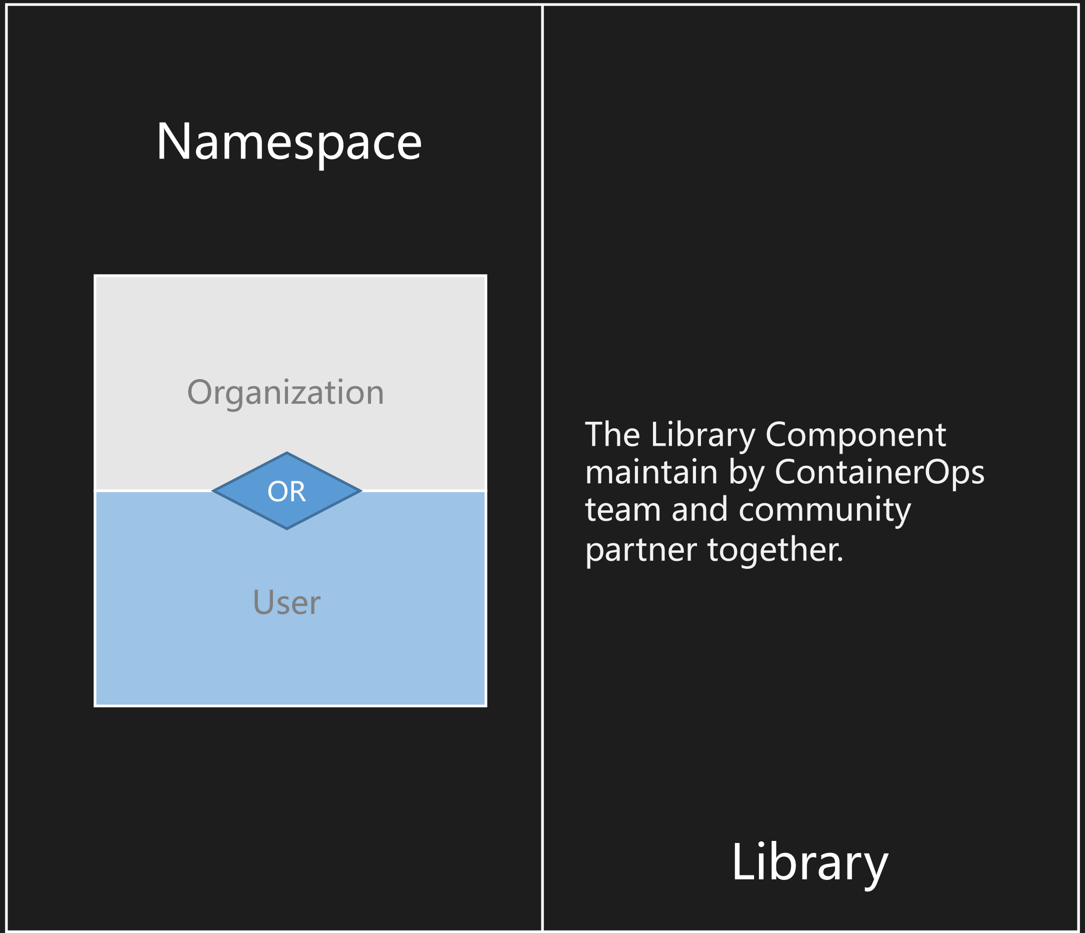
Why A Component Not A Pod?

1. Adaptive other container orchestration system like Docker Swarm.
2. Only one task in the component to easily maintain and share.



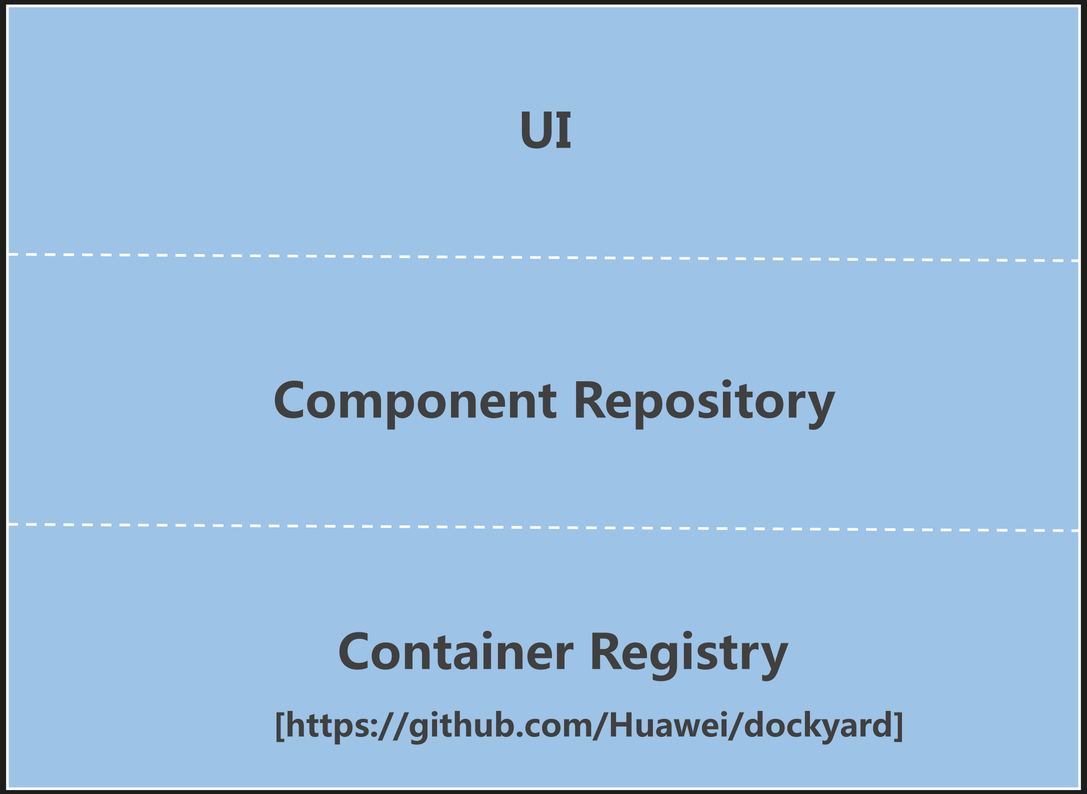


Component Namespace



The Library Component maintain by ContainerOps team and community partner together.

Component Repository



UI

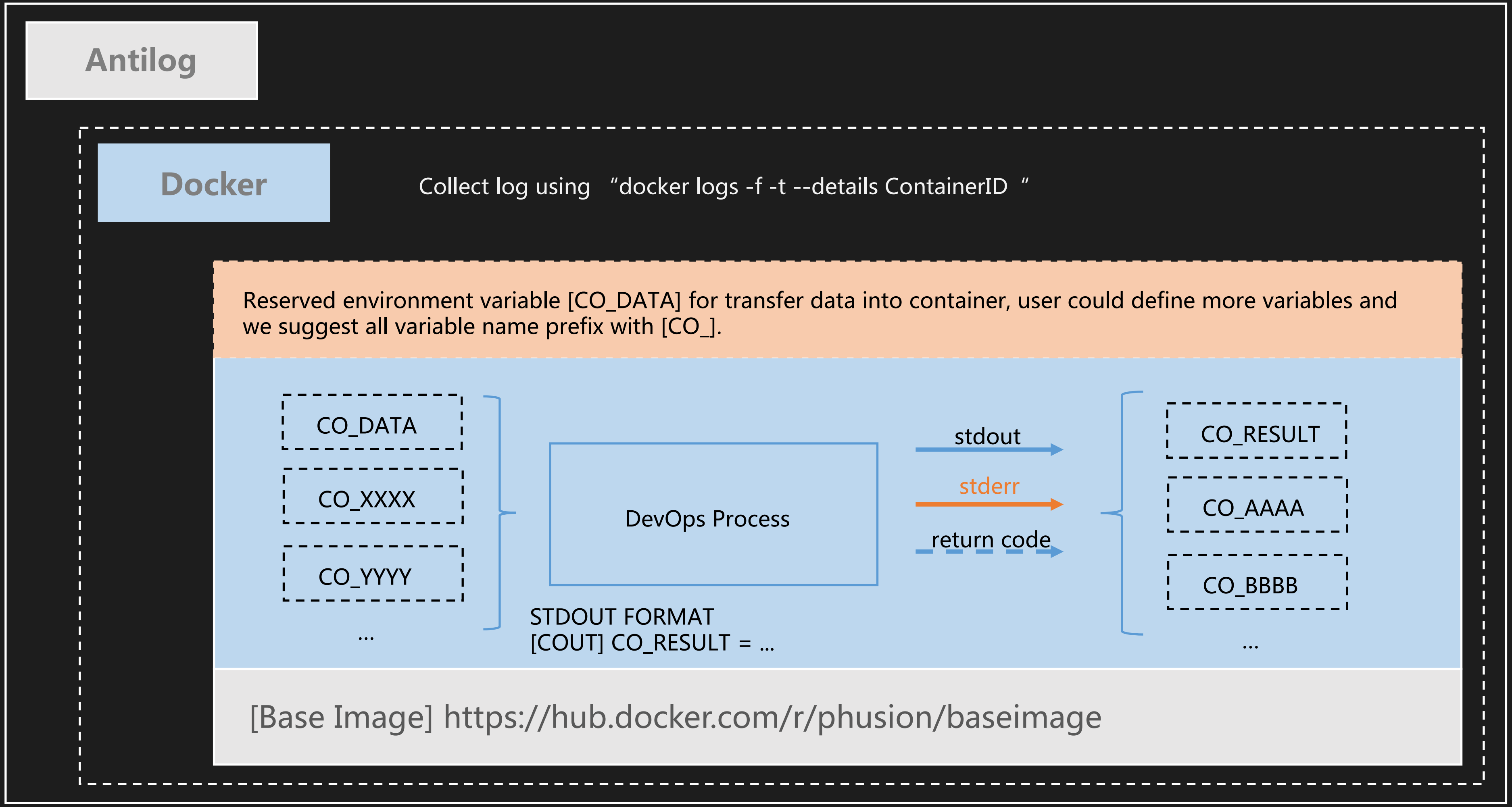
Component Repository

Container Registry

[https://github.com/Huawei/dockyard]

Component Architecture

Resource	CPU/Memory
Timeout	
Manifest	
K8S-config	
Repository	
...	



Antilog

Docker

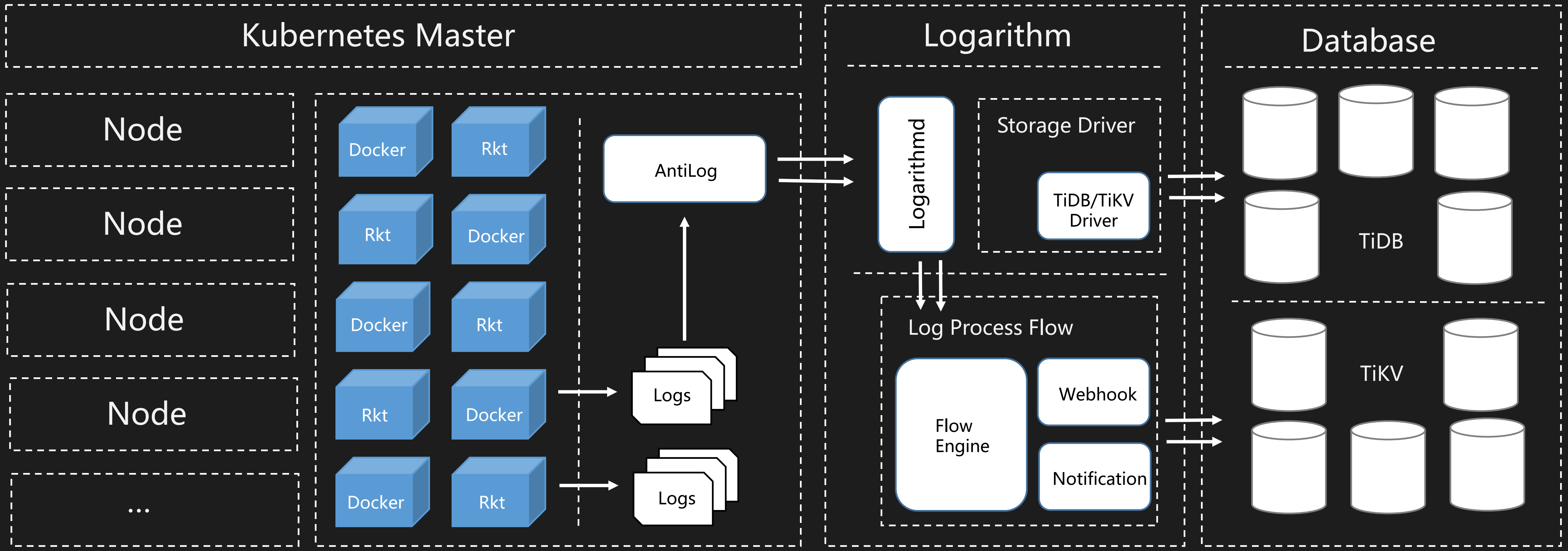
Collect log using “docker logs -f -t --details ContainerID ”

Reserved environment variable [CO_DATA] for transfer data into container, user could define more variables and we suggest all variable name prefix with [CO_].

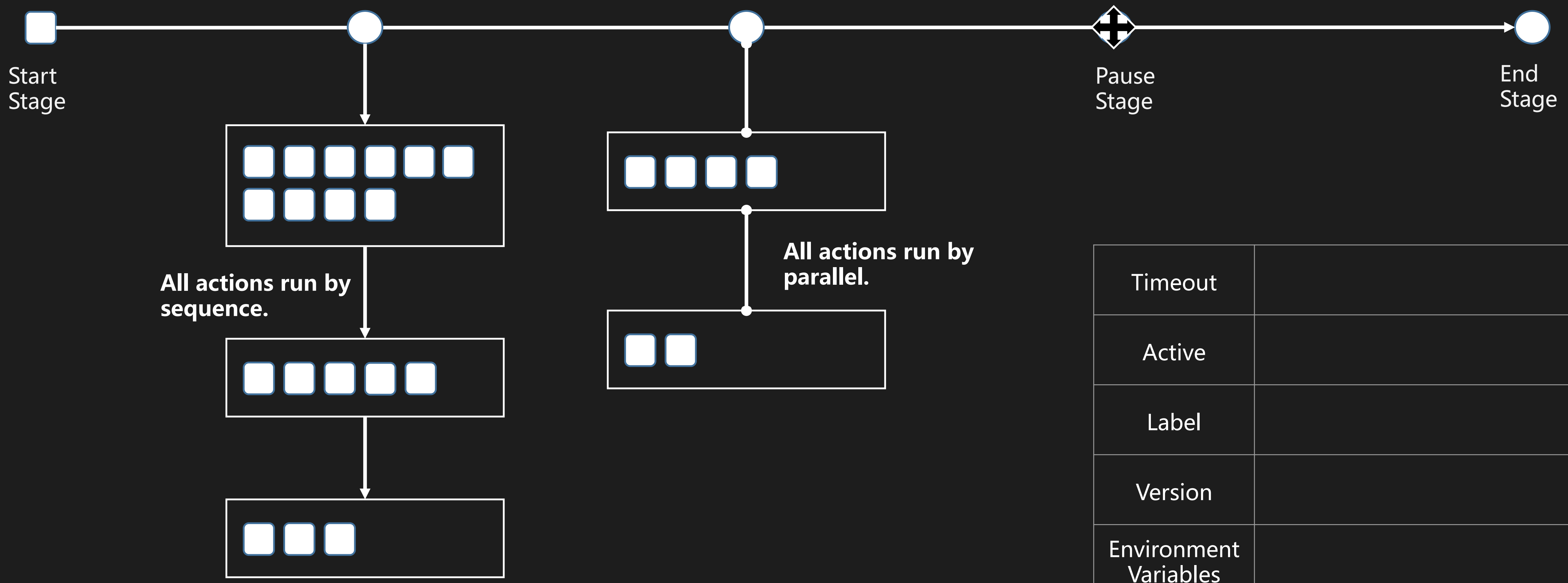
STDOUT FORMAT
[COUT] CO_RESULT = ...

[Base Image] https://hub.docker.com/r/phusion/baseimage

Logarithm Architecture



Workflow Engine – Stage/Action/Job



Start Stage

Pause Stage

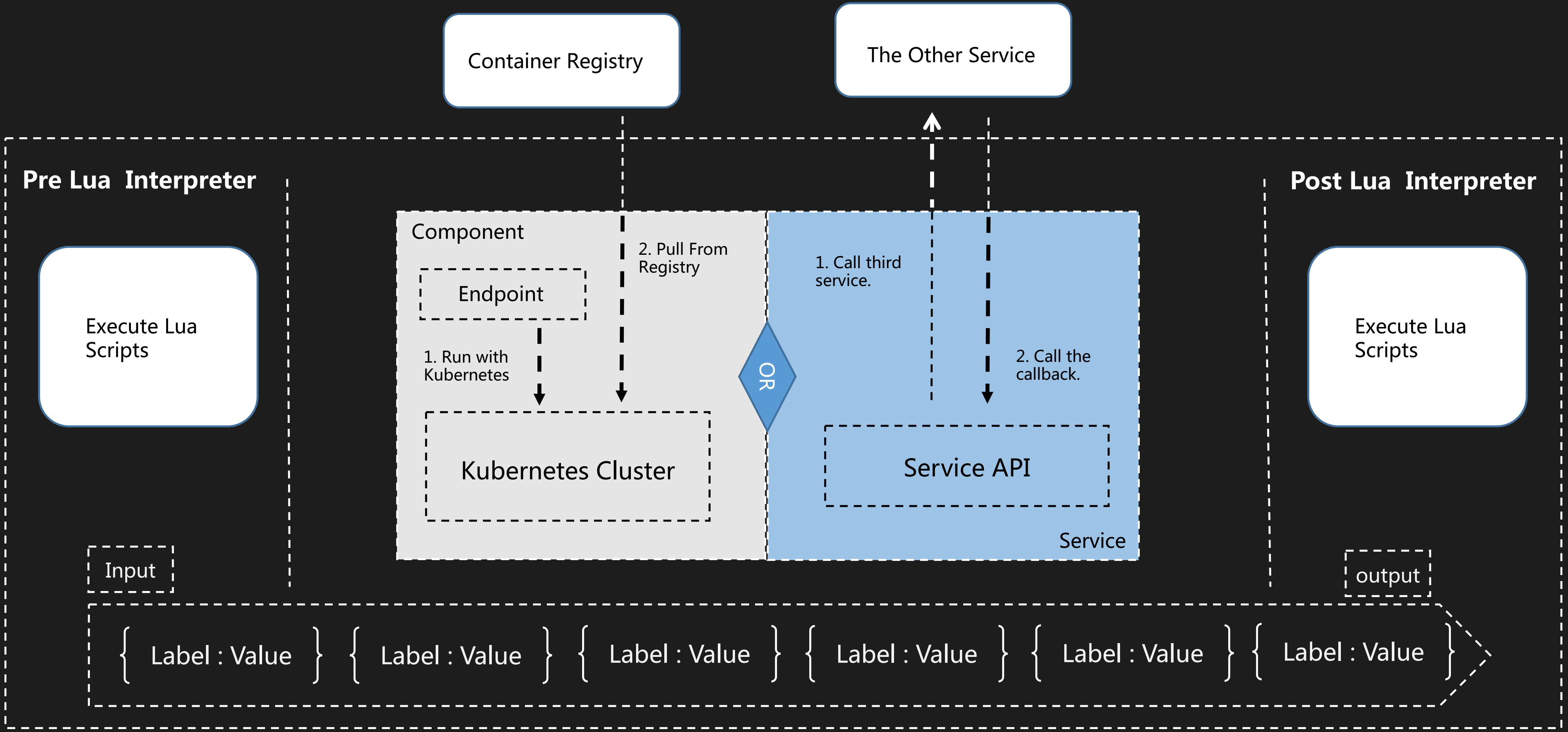
End Stage

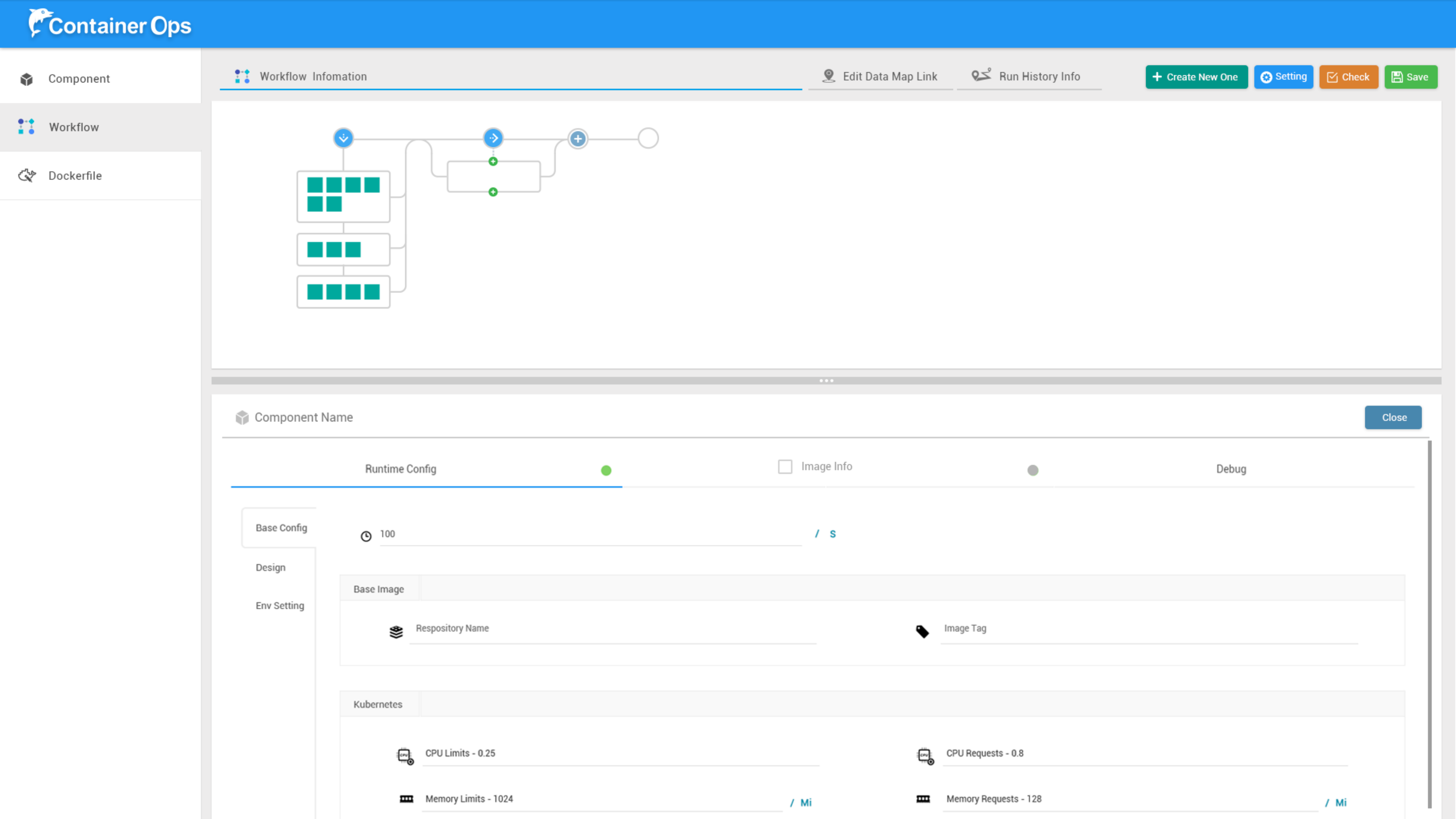
All actions run by sequence.

All actions run by parallel.

Timeout	
Active	
Label	
Version	
Environment Variables	
Resource	
Trigger	

Workflow Engine - Job





Component Name

Runtime Config ● | □ Image Info | ● | Debug

Base Config

Design

Env Setting

100 / S

Base Image

Respository Name

Image Tag

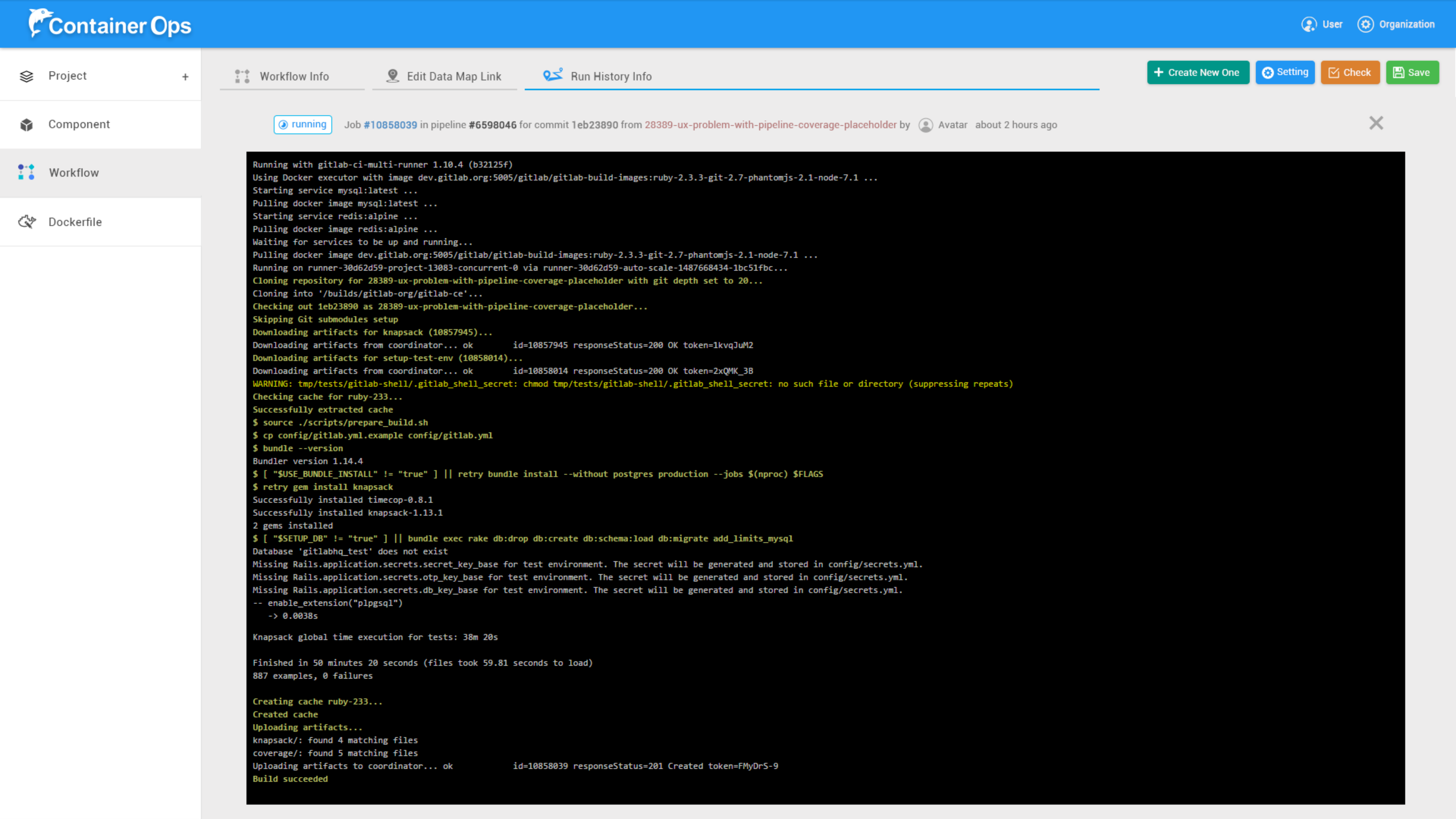
Kubernetes

CPU Limits - 0.25

CPU Requests - 0.8

Memory Limits - 1024 / Mi

Memory Requests - 128 / Mi



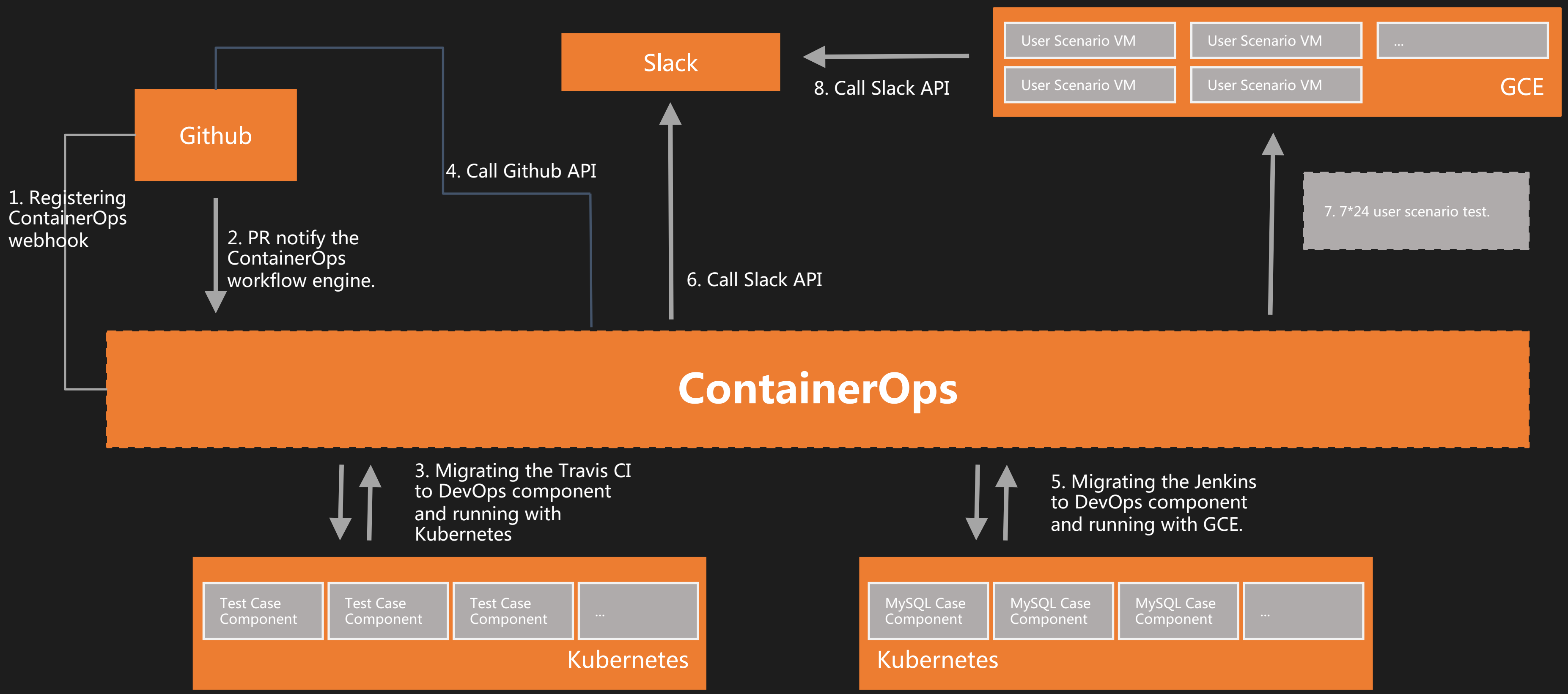
running Job #10858039 in pipeline #6598046 for commit 1eb23890 from 28389-ux-problem-with-pipeline-coverage-placeholder by Avatar about 2 hours ago

```
Running with gitlab-ci-multi-runner 1.10.4 (b32125f)
Using Docker executor with image dev.gitlab.org:5005/gitlab/gitlab-build-images:ruby-2.3.3-git-2.7-phantomjs-2.1-node-7.1 ...
Starting service mysql:latest ...
Pulling docker image mysql:latest ...
Starting service redis:alpine ...
Pulling docker image redis:alpine ...
Waiting for services to be up and running...
Pulling docker image dev.gitlab.org:5005/gitlab/gitlab-build-images:ruby-2.3.3-git-2.7-phantomjs-2.1-node-7.1 ...
Running on runner-30d62d59-project-13083-concurrent-0 via runner-30d62d59-auto-scale-1487668434-1bc51fbc...
Cloning repository for 28389-ux-problem-with-pipeline-coverage-placeholder with git depth set to 20...
Cloning into '/builds/gitlab-org/gitlab-ce'...
Checking out 1eb23890 as 28389-ux-problem-with-pipeline-coverage-placeholder...
Skipping Git submodules setup
Downloading artifacts for knapsack (10857945)...
Downloading artifacts from coordinator... ok        id=10857945 responseStatus=200 OK token=1kvqJuM2
Downloading artifacts for setup-test-env (10858014)...
Downloading artifacts from coordinator... ok        id=10858014 responseStatus=200 OK token=2xQMK_3B
WARNING: tmp/tests/gitlab-shell/.gitlab_shell_secret: chmod tmp/tests/gitlab-shell/.gitlab_shell_secret: no such file or directory (suppressing repeats)
Checking cache for ruby-233...
Successfully extracted cache
$ source ./scripts/prepare_build.sh
$ cp config/gitlab.yml.example config/gitlab.yml
$ bundle --version
Bundler version 1.14.4
$ [ "$USE_BUNDLE_INSTALL" != "true" ] || retry bundle install --without postgres production --jobs $(nproc) $FLAGS
$ retry gem install knapsack
Successfully installed timecop-0.8.1
Successfully installed knapsack-1.13.1
2 gems installed
$ [ "$SETUP_DB" != "true" ] || bundle exec rake db:drop db:create db:schema:load db:migrate add_limits_mysql
Database 'gitlabhq_test' does not exist
Missing Rails.application.secrets.secret_key_base for test environment. The secret will be generated and stored in config/secrets.yml.
Missing Rails.application.secrets.otp_key_base for test environment. The secret will be generated and stored in config/secrets.yml.
Missing Rails.application.secrets.db_key_base for test environment. The secret will be generated and stored in config/secrets.yml.
-- enable_extension("plpgsql")
   -> 0.0038s

Knapsack global time execution for tests: 38m 20s

Finished in 50 minutes 20 seconds (files took 59.81 seconds to load)
887 examples, 0 failures

Creating cache ruby-233...
Created cache
Uploading artifacts...
knapsack/: found 4 matching files
coverage/: found 5 matching files
Uploading artifacts to coordinator... ok            id=10858039 responseStatus=201 Created token=FMyDrS-9
Build succeeded
```

TiDB/TiKV/PB Case

3 DevOps Stage with different systems:

1. Test case Within Travis CI
2. Merge Stage: 10000000+ MySQL test case with Jenkins.
3. Release Stage: 7*24 hours user scenario with manual.



Github

Slack

User Scenario VM

User Scenario VM

...

User Scenario VM

User Scenario VM

GCE

1. Registering ContainerOps webhook

2. PR notify the ContainerOps workflow engine.

3. Migrating the Travis CI to DevOps component and running with Kubernetes

4. Call Github API

5. Migrating the Jenkins to DevOps component and running with GCE.

6. Call Slack API

7. 7*24 user scenario test.

8. Call Slack API

ContainerOps

Test Case Component

Test Case Component

Test Case Component

...

Kubernetes

MySQL Case Component

MySQL Case Component

MySQL Case Component

...

Kubernetes

Thanks & End