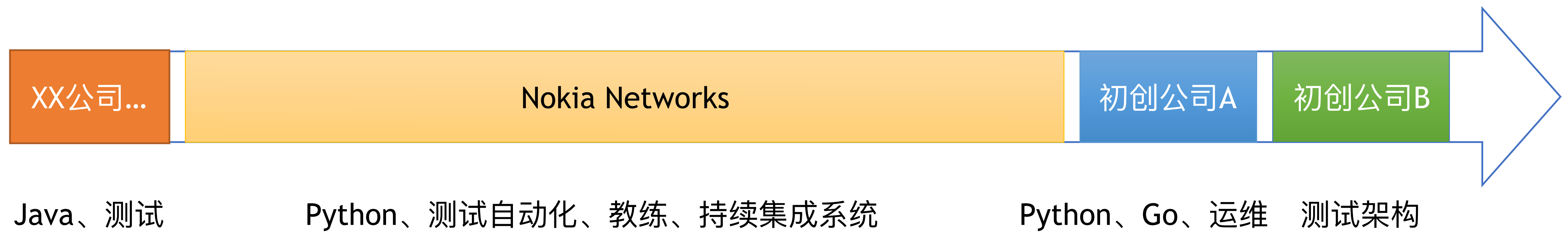
PyConChina
2019

从0开始快速构建DevOps系统

一个小型toB团队的DevOps系统诞生之路

张裕

About me

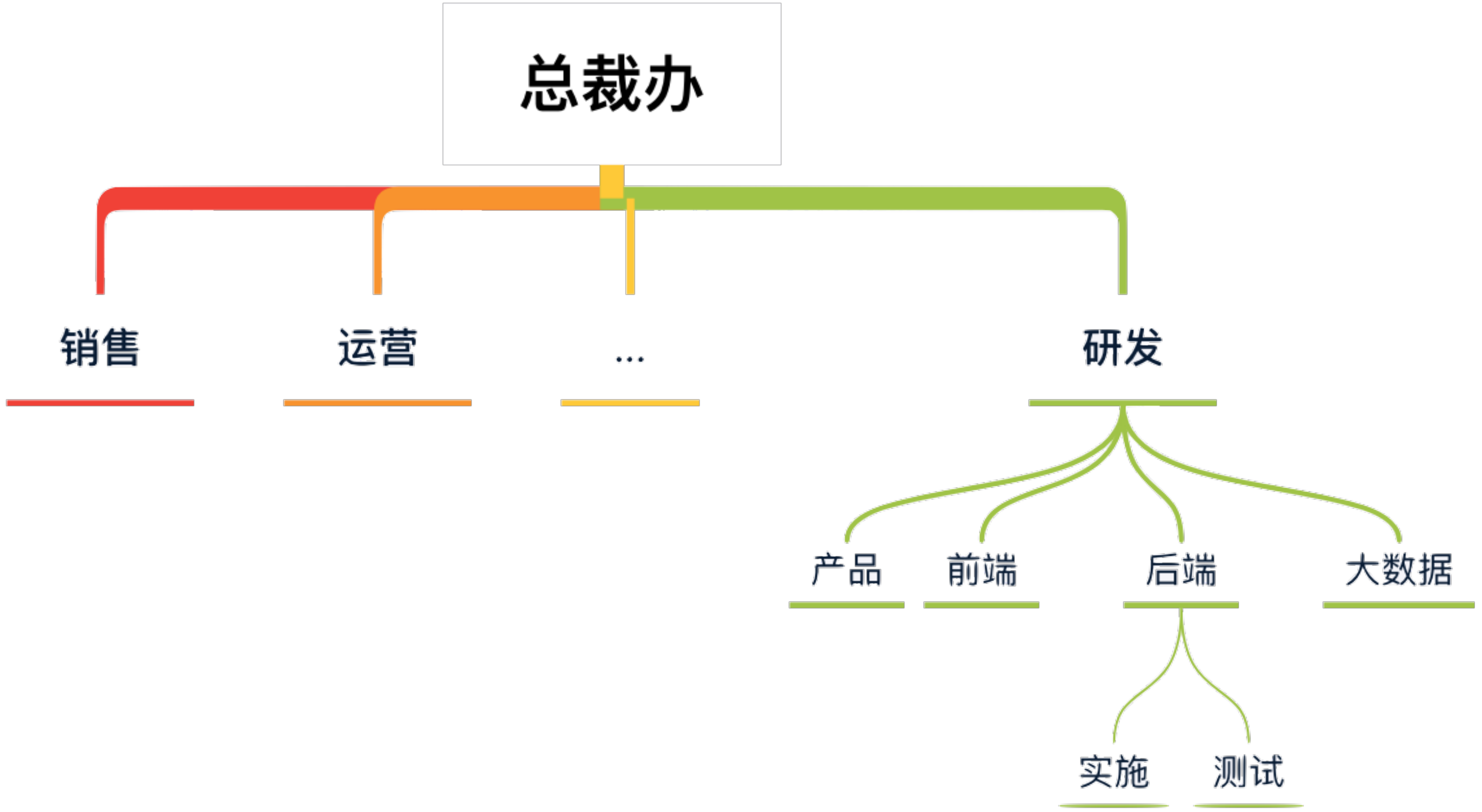

XX公司...
Nokia Networks
初创公司A
初创公司B
Java、测试
Python、测试自动化、教练、持续集成系统
Python、Go、运维
测试架构

目录 CONTENTS

1 问题的由来

- 组织架构
- 产品特点
- 主要问题

组织架构



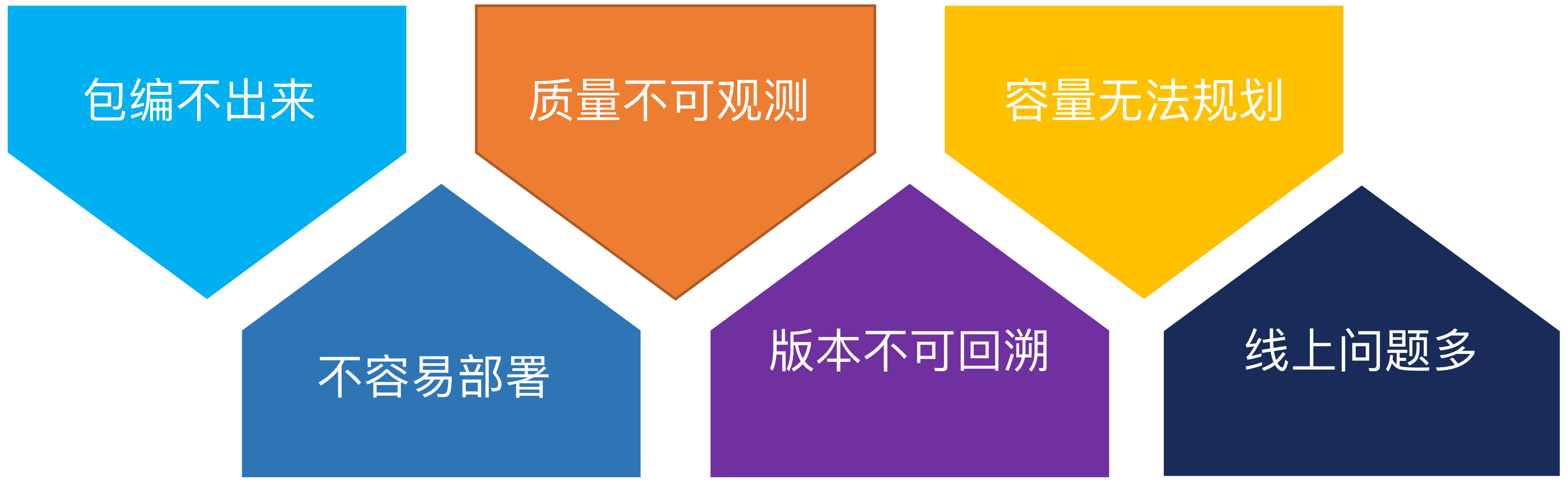
产品特点

PyConChina 2019

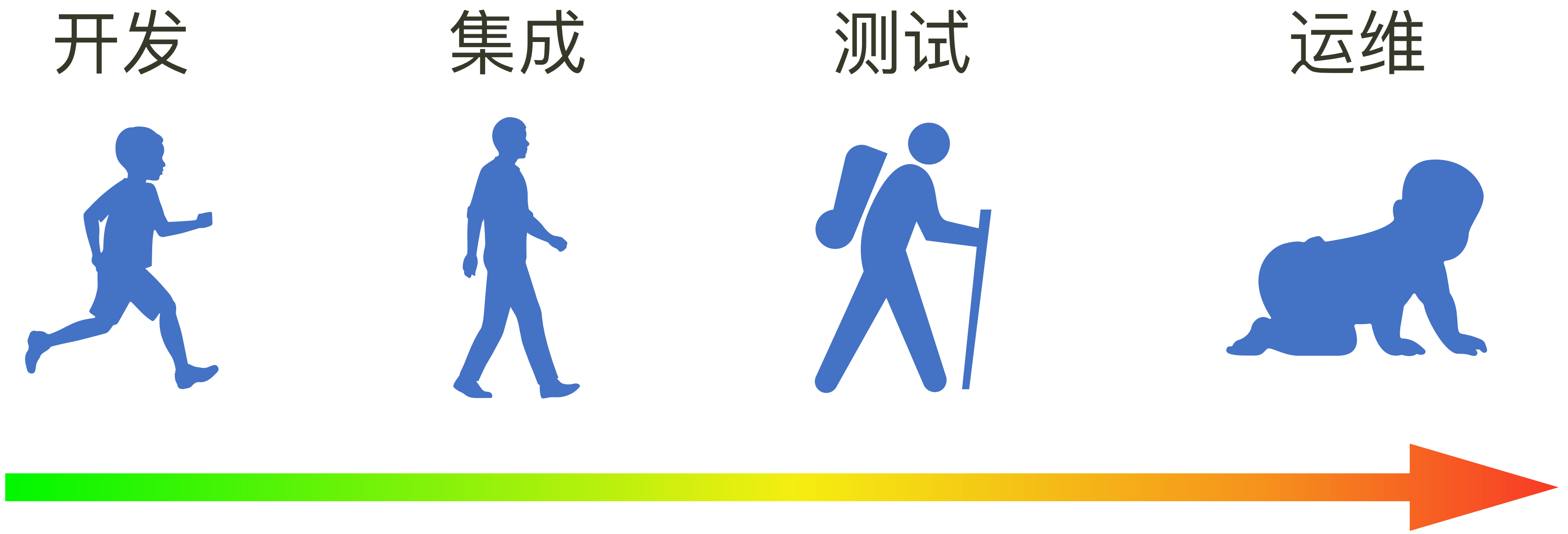
在客户私有环境部署的多服务、
单接口、高可用互联网应用

PYTHON 30th

主要问题



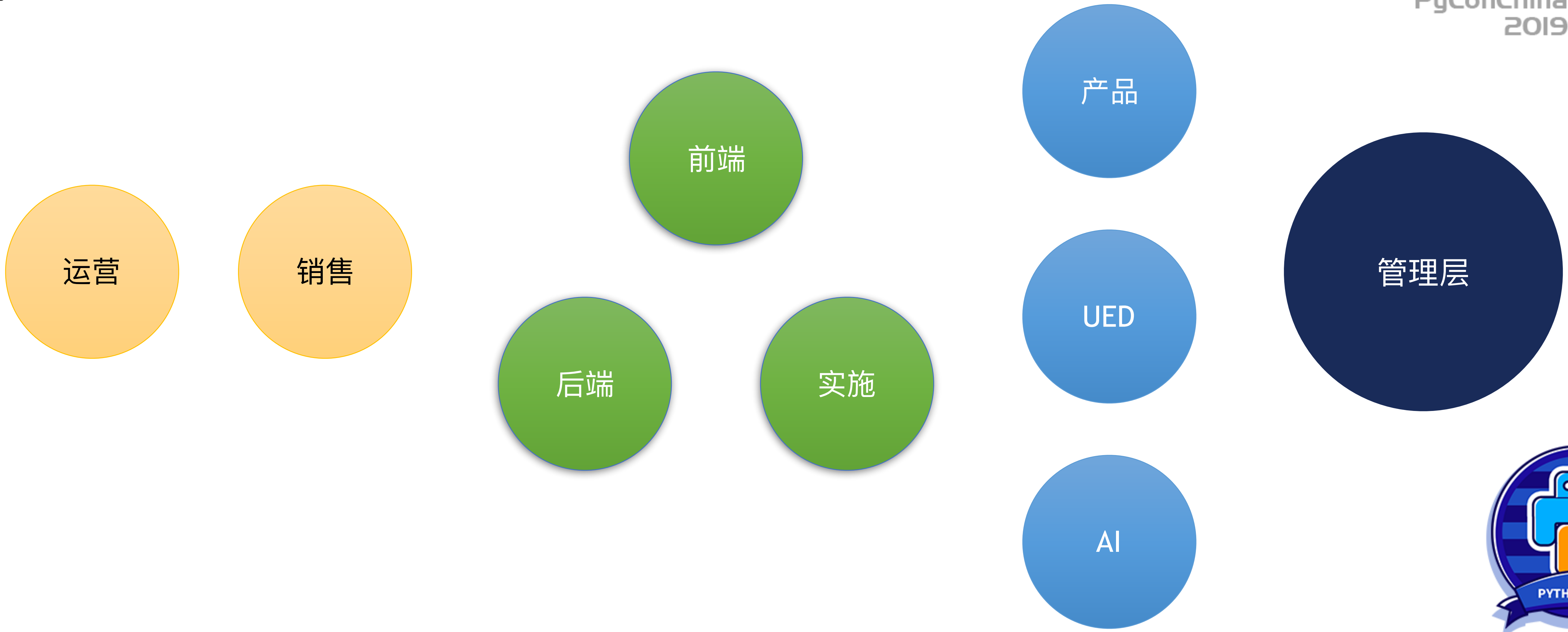
主要问题



2 从部署开始

- 原则：让更多的人更早用起来
- 实现：尽可能简单

原则



实现 - phase 1

从一个脚本开始：

```
python -m tao.tools deploy_docker -h env-1.test.local -u demo -p Demo123 -c harbor/c1:v1 -c harbor/c2:v2
```

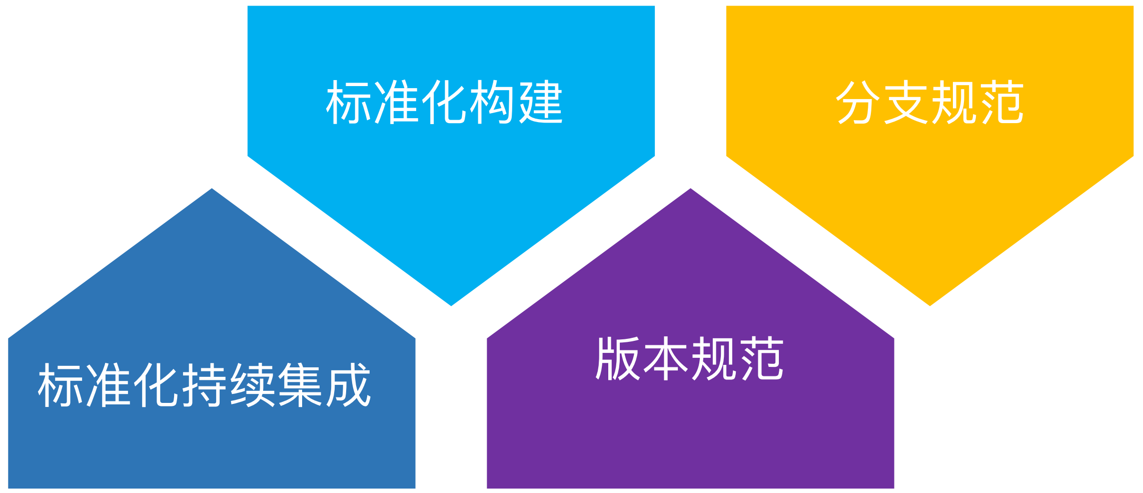
实现 - phase 1

PyConChina 2019

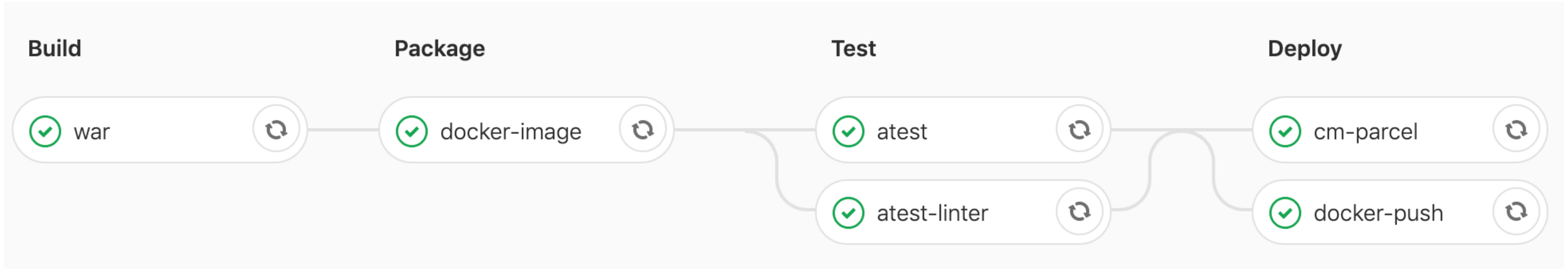
```
import click
from fabric import Connection, Config

@click.command('deploy_docker')
@click.option('--host', '-h', required=True, help='host to deploy')
@click.option('--user', '-u', default='test', help='username of SSH login')
@click.option('--password', '-p', default='test123', help='password of SSH login')
@click.option('--component', '-c', multiple=True, required=True, callback=_validate_app)
def deploy(host, user, password, version, component):
    config = Config(overrides={'sudo': {'password': password}})
    with Connection(host, user, config=config, connect_kwargs={'password': password}) as conn:
        for app_item in component:
            click.echo(f'start to deploy {app_item.image}:{app_item.version}')
            _do_deploy(conn, app_item)
```

PYTHON 30th

实现 - 其他准备



标准化持续集成



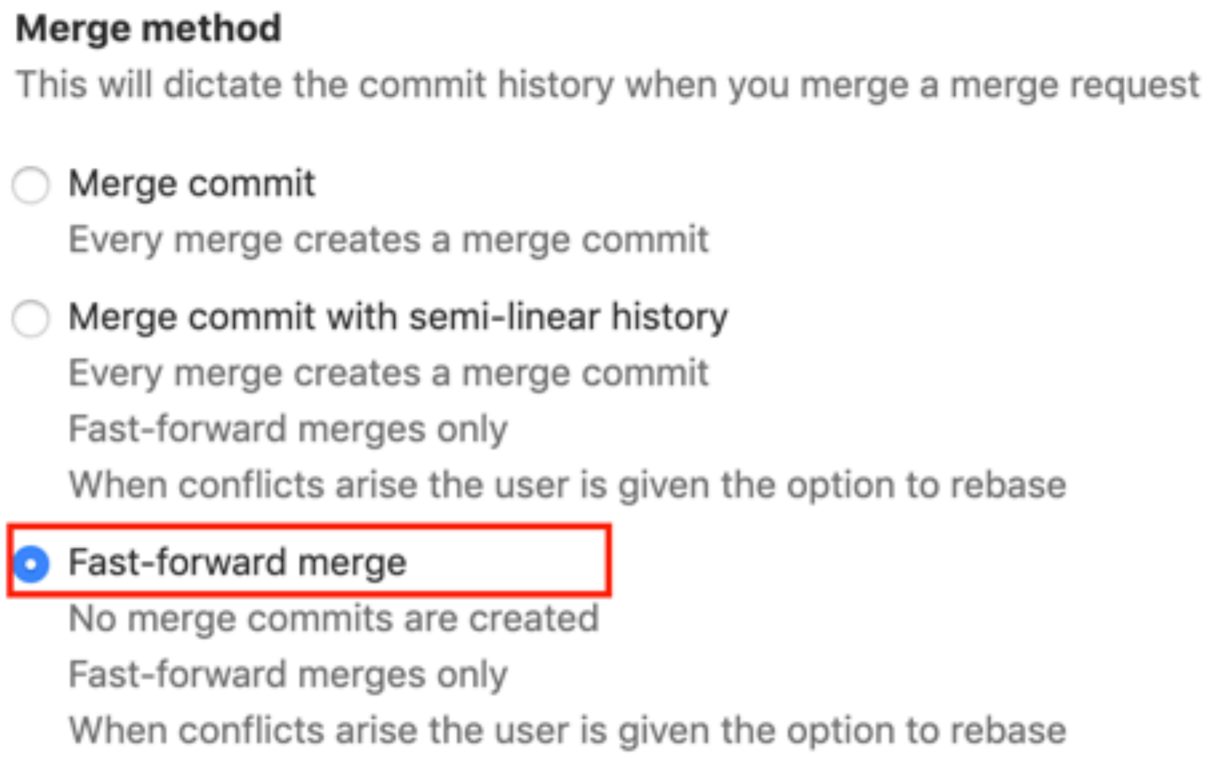
Dockerfile

.gitlab-ci.yml

标准化持续集成

```
# .gitlab-ci.yml

variables:
    QS: "namespace=${CI_PROJECT_NAMESPACE}&project=${CI_PROJECT_NAME}&branch=${CI_COMMIT_REF_NAME}"

before_script:
    - curl -s -o cci.sh --retry 5 http://tao.test.local/api/v1/cci/script\?${QS}
    - source ./cci.sh

stages:
    - build
    - package
    - test
    - deploy
```

标准化持续集成



Merge method

This will dictate the commit history when you merge a merge request

○ Merge commit

Every merge creates a merge commit

○ Merge commit with semi-linear history

Every merge creates a merge commit

Fast-forward merges only

When conflicts arise the user is given the option to rebase

● Fast-forward merge

No merge commits are created

Fast-forward merges only

When conflicts arise the user is given the option to rebase

标准化构建

PyConChina 2019

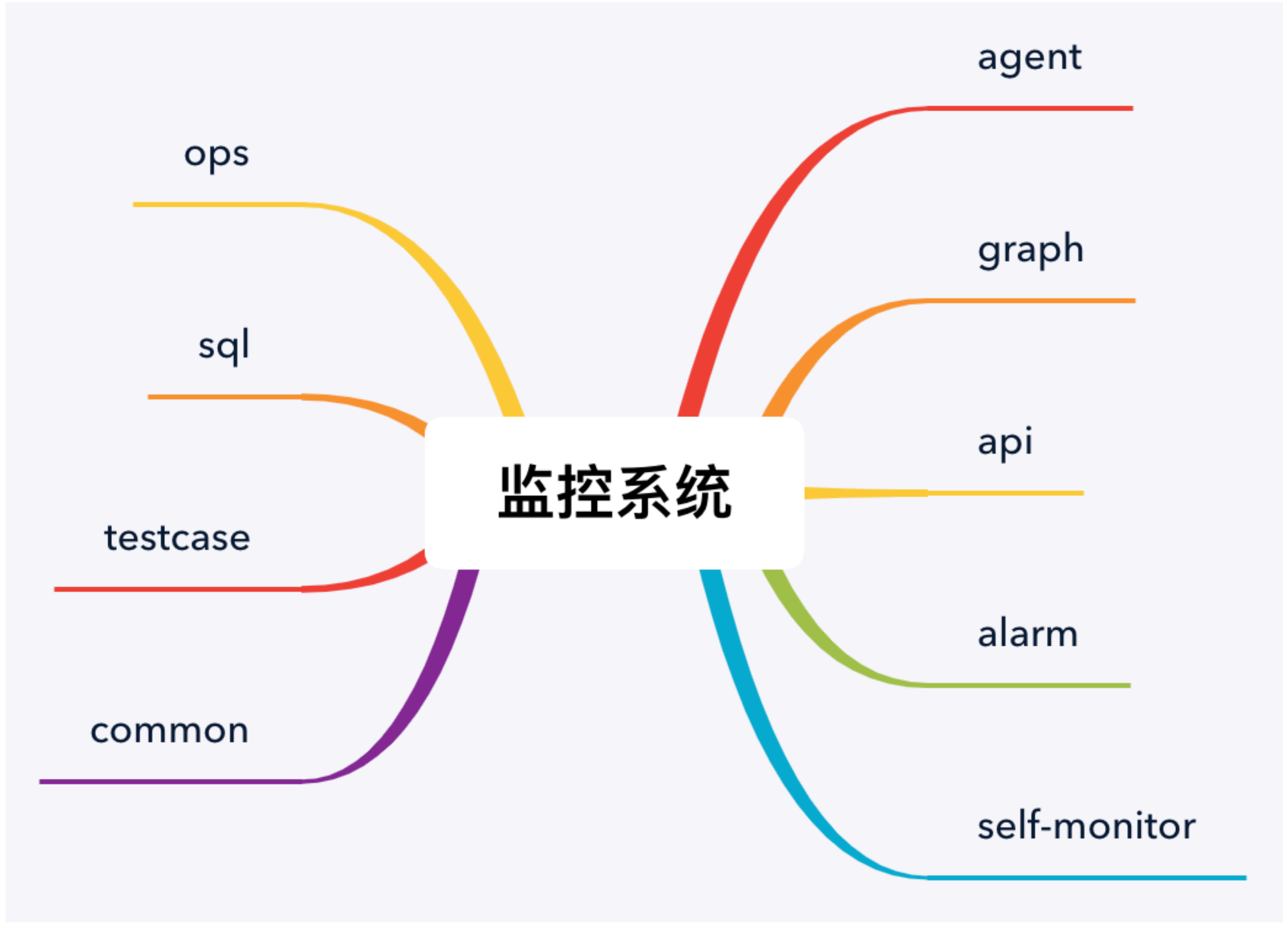
配置 DNS 化　　通过本地DNS服务来统一各个环境的配置

应用部署分层

```
                应用:    component-ui   |   component-a  | ...
——————————————————————————————————————————————
环境独立中间件:    zookeeper  |  ...
——————————————————————————————————————————————
环境共享中间件:    MySQL | Hive | Hadoop | ...
```

PYTHON 30th

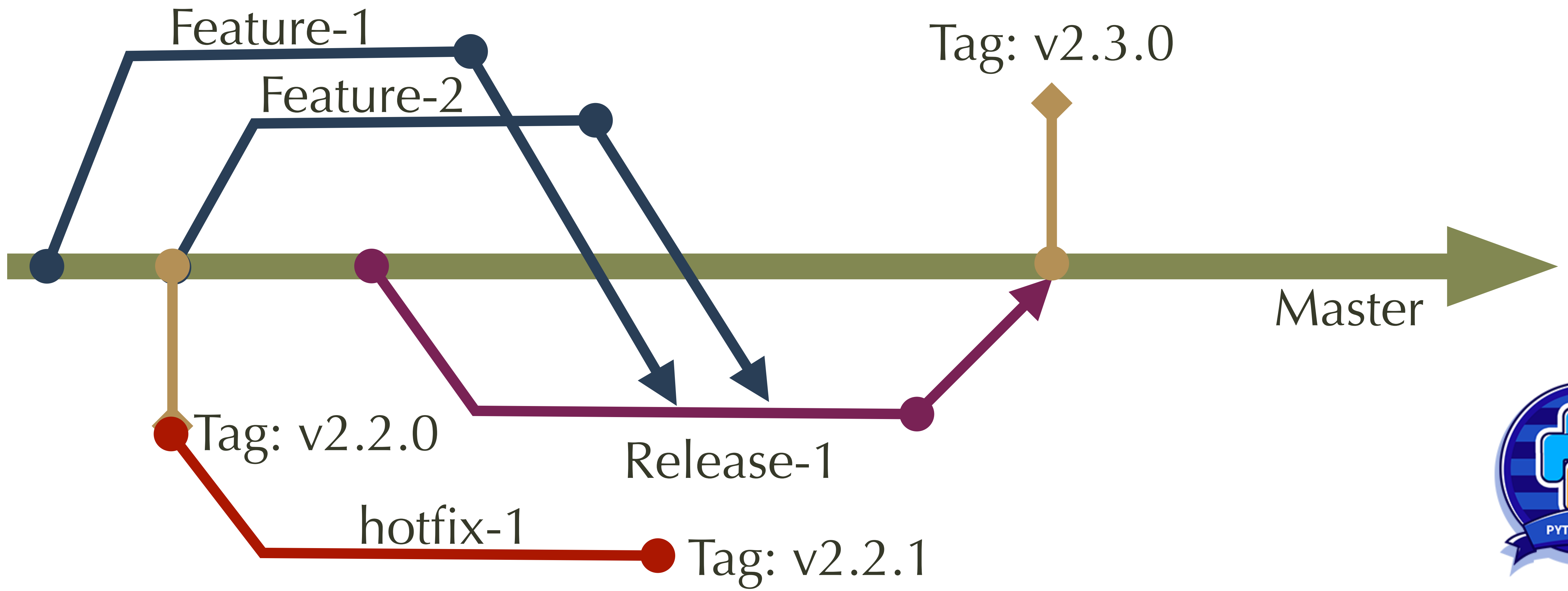
版本规范



监控系统 v1.1.0

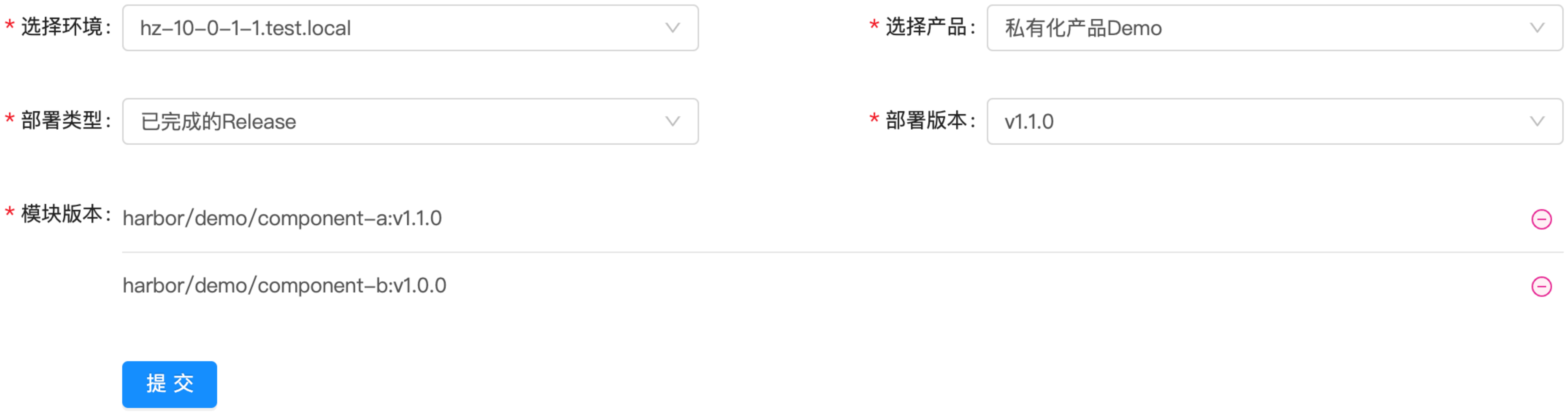
- ops v1.1.0
- sql v1.0.0
- testcase v1.1.0
- common v1.0.1
- agent v1.1.0
- graph v1.0.2
- api v1.1.0
- alarm v1.1.0
- self-monitor v1.1.0

分支规范


Feature-1
Feature-2
Tag: v2.3.0
Master
Tag: v2.2.0
Release-1
hotfix-1
Tag: v2.2.1

实现 - phase 2

一键部署特定版本到某个环境上



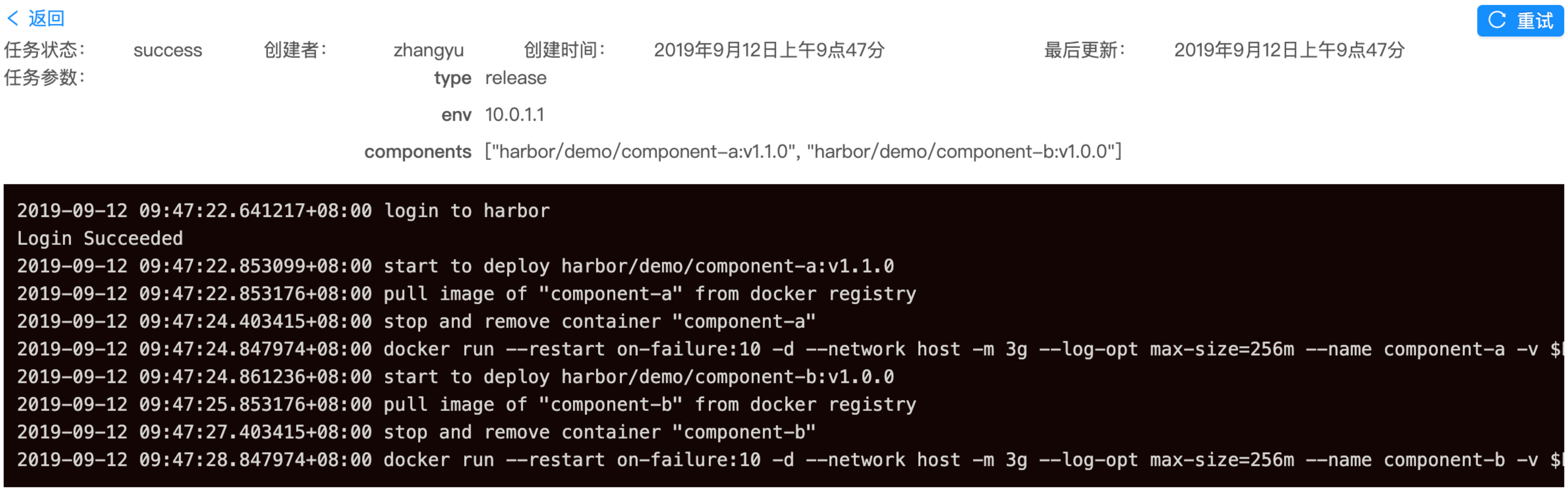
* 选择环境：	hz-10-0-1-1.test.local	* 选择产品：	私有化产品Demo
* 部署类型：	已完成的Release	* 部署版本：	v1.1.0

* 模块版本： harbor/demo/component-a:v1.1.0

harbor/demo/component-b:v1.0.0

实现 - phase 2

PyConChina 2019



< 返回

重试

任务状态: success 创建者: zhangyu 创建时间: 2019年9月12日上午9点47分 最后更新: 2019年9月12日上午9点47分

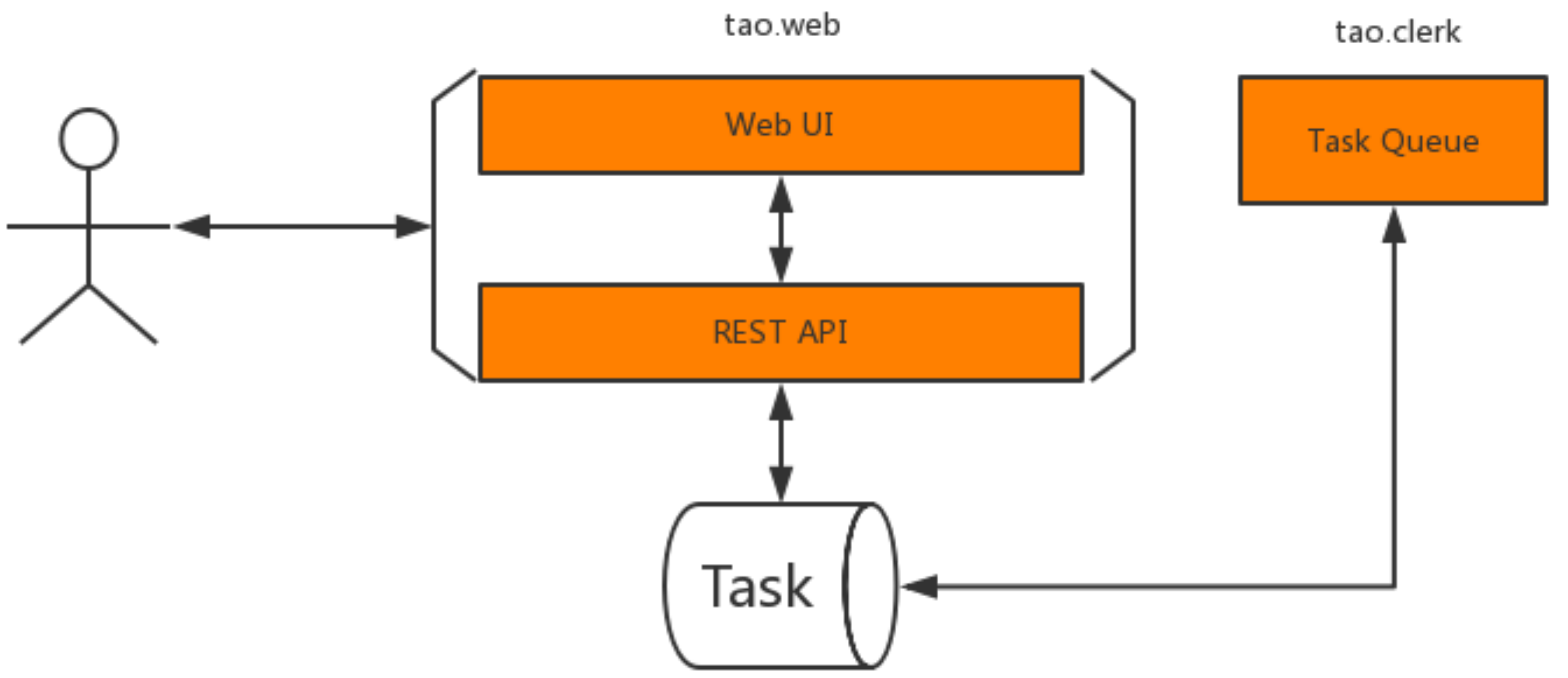
任务参数:

type	release
env	10.0.1.1
components	["harbor/demo/component-a:v1.1.0", "harbor/demo/component-b:v1.0.0"]

```
2019-09-12 09:47:22.641217+08:00 login to harbor
Login Succeeded
2019-09-12 09:47:22.853099+08:00 start to deploy harbor/demo/component-a:v1.1.0
2019-09-12 09:47:22.853176+08:00 pull image of "component-a" from docker registry
2019-09-12 09:47:24.403415+08:00 stop and remove container "component-a"
2019-09-12 09:47:24.847974+08:00 docker run --restart on-failure:10 -d --network host -m 3g --log-opt max-size=256m --name component-a -v $
2019-09-12 09:47:24.861236+08:00 start to deploy harbor/demo/component-b:v1.0.0
2019-09-12 09:47:25.853176+08:00 pull image of "component-b" from docker registry
2019-09-12 09:47:27.403415+08:00 stop and remove container "component-b"
2019-09-12 09:47:28.847974+08:00 docker run --restart on-failure:10 -d --network host -m 3g --log-opt max-size=256m --name component-b -v $
```

实现 - phase 2

PyConChina 2019



实现 - phase 2

```
Supervisor
    |- tao.web
    |      |- sanic + uvloop + motor
    |- tao.clerk
    |      |- asyncio + motor
```

```
import asyncio
from tao.models import Task
from .runner import TaskRunner


_available_workers = asyncio.Semaphore(5)  # max 5 concurrent tasks

async def load_task_queue():
    while True:
        task = await Task.find_one_and_update({'status': Task.WAITING}, {
            '$set': {'status': Task.RUNNING}})
        if not task:
            await asyncio.sleep(2)
            continue
        asyncio.get_event_loop().create_task(_run_task(task))


async def _run_task(task):
    async with _available_workers:
        logging.debug(f'schedule task "{task}"')
        await TaskRunner.run(task)
```

3 让数据互通

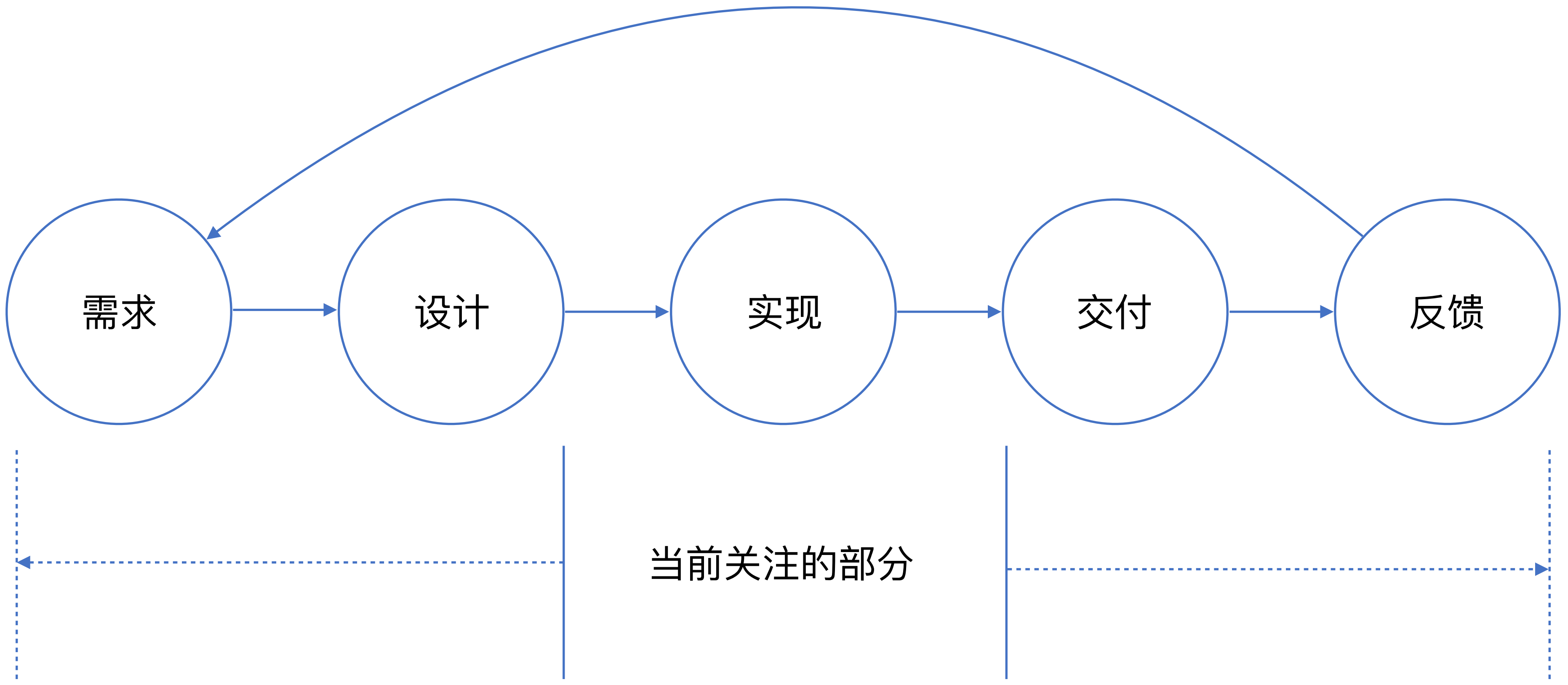
- 私有化的应用怎么做好OPS
- 研发过程数据
- 需求、缺陷数据

问题

私有化的应用怎么做好OPS

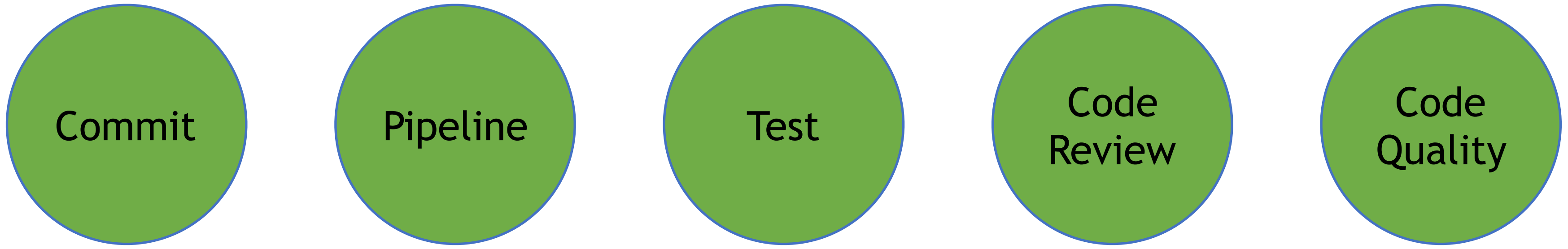
- 假装自己是用户
- 让需求和反馈尽快流动起来

问题



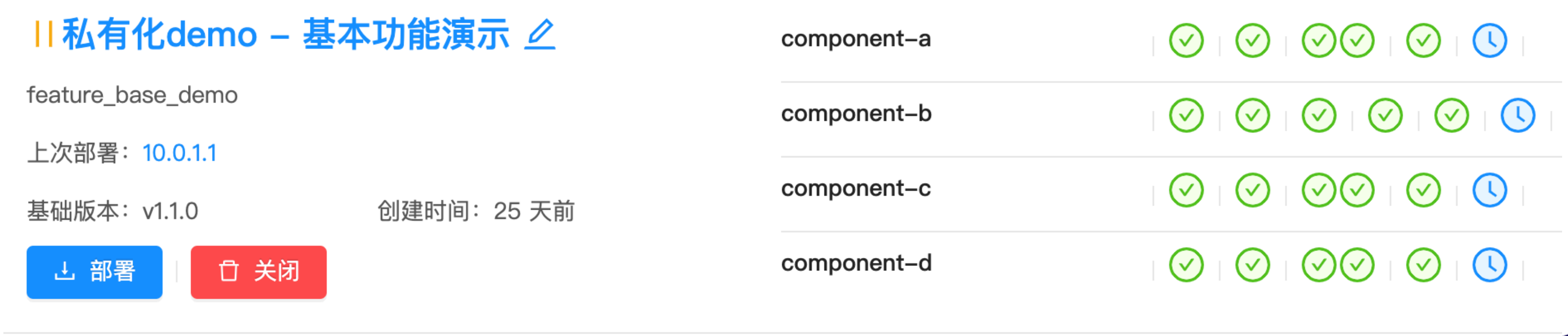
研发过程数据

GitLab Webhook



```python
@gitlab_bp.post('/api/v1/gitlab/webhook')
async def gitlab_webhook(request):
    event_name = request.headers.get('X-Gitlab-Event')
    token = request.headers.get('X-Gitlab-Token')
    await GitlabEvent.create(request.json)
    await WebHookHandlers.on_event(event_name, event)
```

研发过程数据



私有化demo – 基本功能演示

feature_base_demo

上次部署：10.0.1.1

基础版本：v1.1.0 创建时间：25 天前

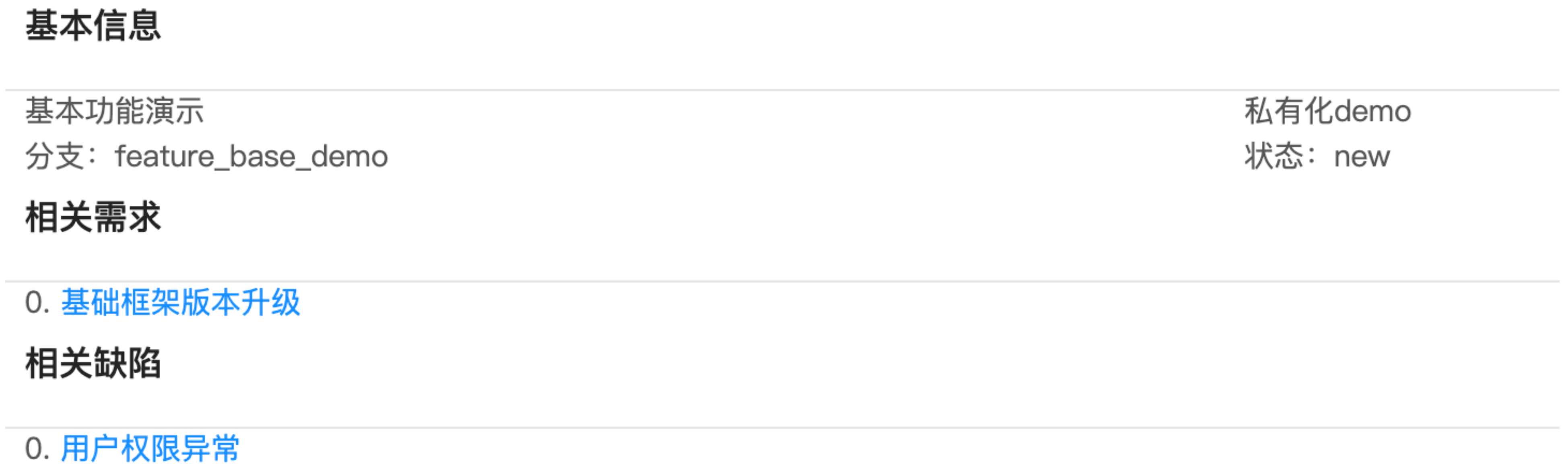
部署 | 关闭

component–a	✓ | ✓ | ✓✓ | ✓ | 🕒
component–b	✓ | ✓ | ✓ | ✓ | ✓ | 🕒
component–c	✓ | ✓ | ✓✓ | ✓ | 🕒
component–d	✓ | ✓ | ✓✓ | ✓ | 🕒

需求、缺陷数据

PyConChina 2019

TAPD
- 项目
- 迭代
- 需求
- 缺陷


基本信息

基本功能演示　私有化demo
分支：feature_base_demo　状态：new

相关需求

0. 基础框架版本升级

相关缺陷

0. 用户权限异常

apscheduler + aiohttp

几个假设

PyConChina 2019

产品 = 项目

发布 = 迭代

版本号 = 迭代名

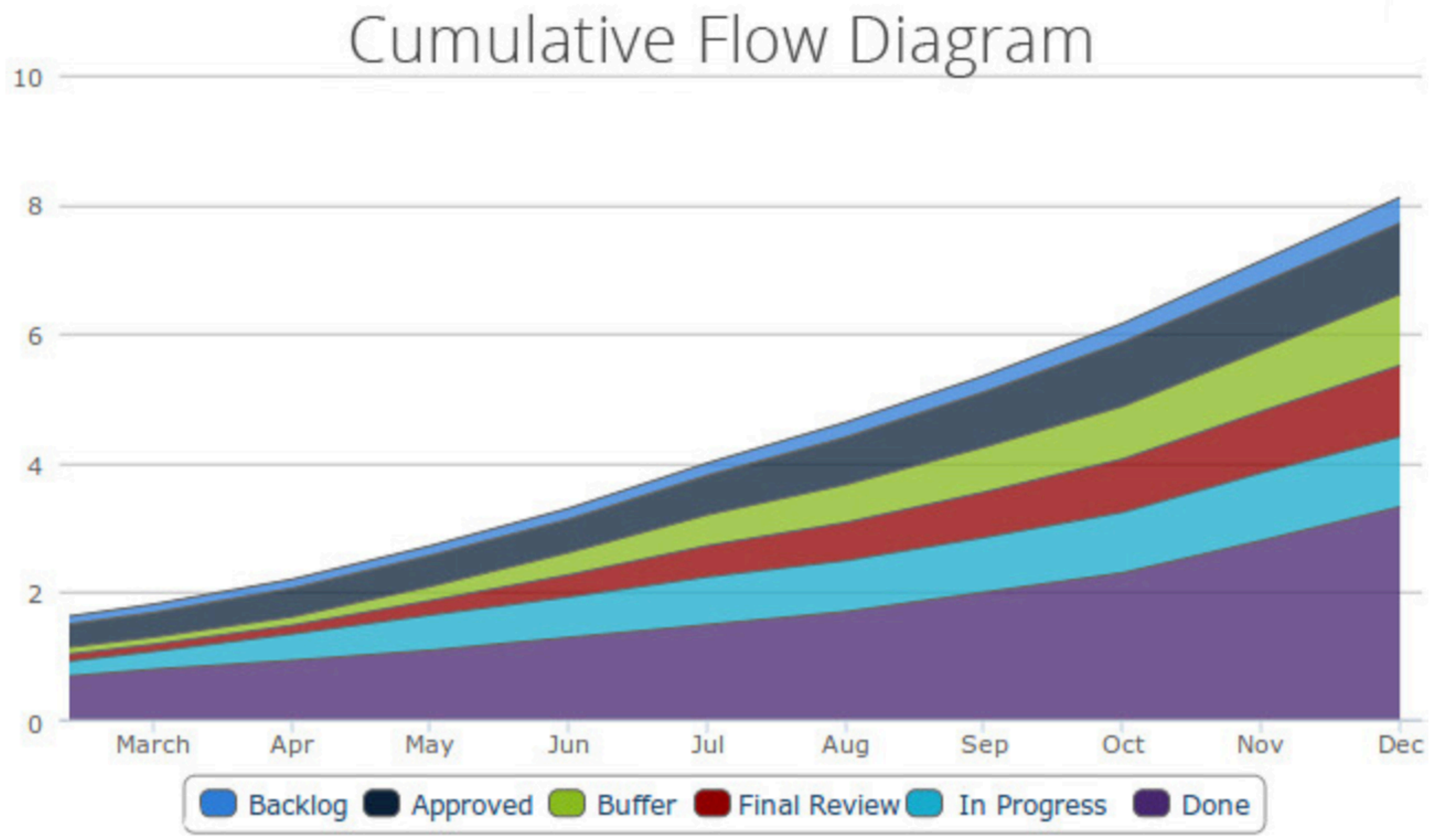
销售、产品… 研发、实施…

PYTHON 30th

4 让数据可见

- 原则：关注整体，远离KPI
- 实现：关注数据接口，展示交给grafana

原则 - 关注整体



PyConChina 2019

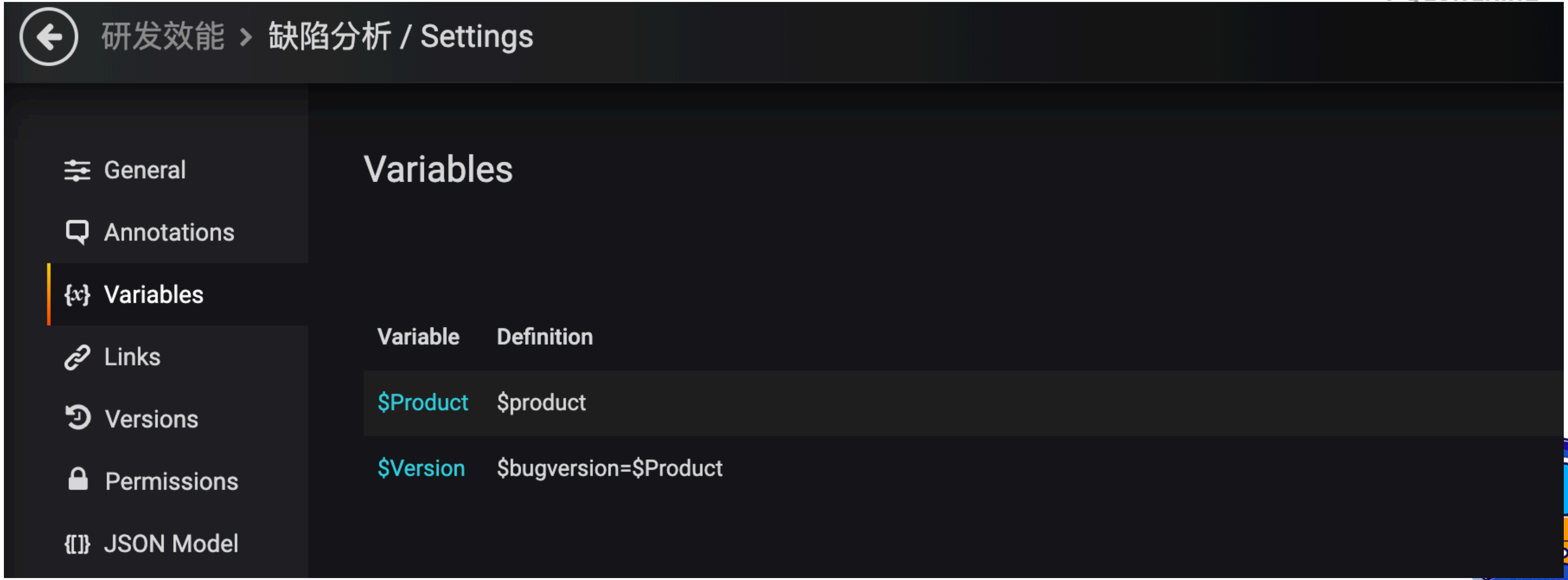
PYTHON 30th

实现

以falcon-plus数据源的形式对接grafana

- GET /api/v1/grafana/metrics/find （交互）
- POST /api/v1/grafana/render （展示）

实现



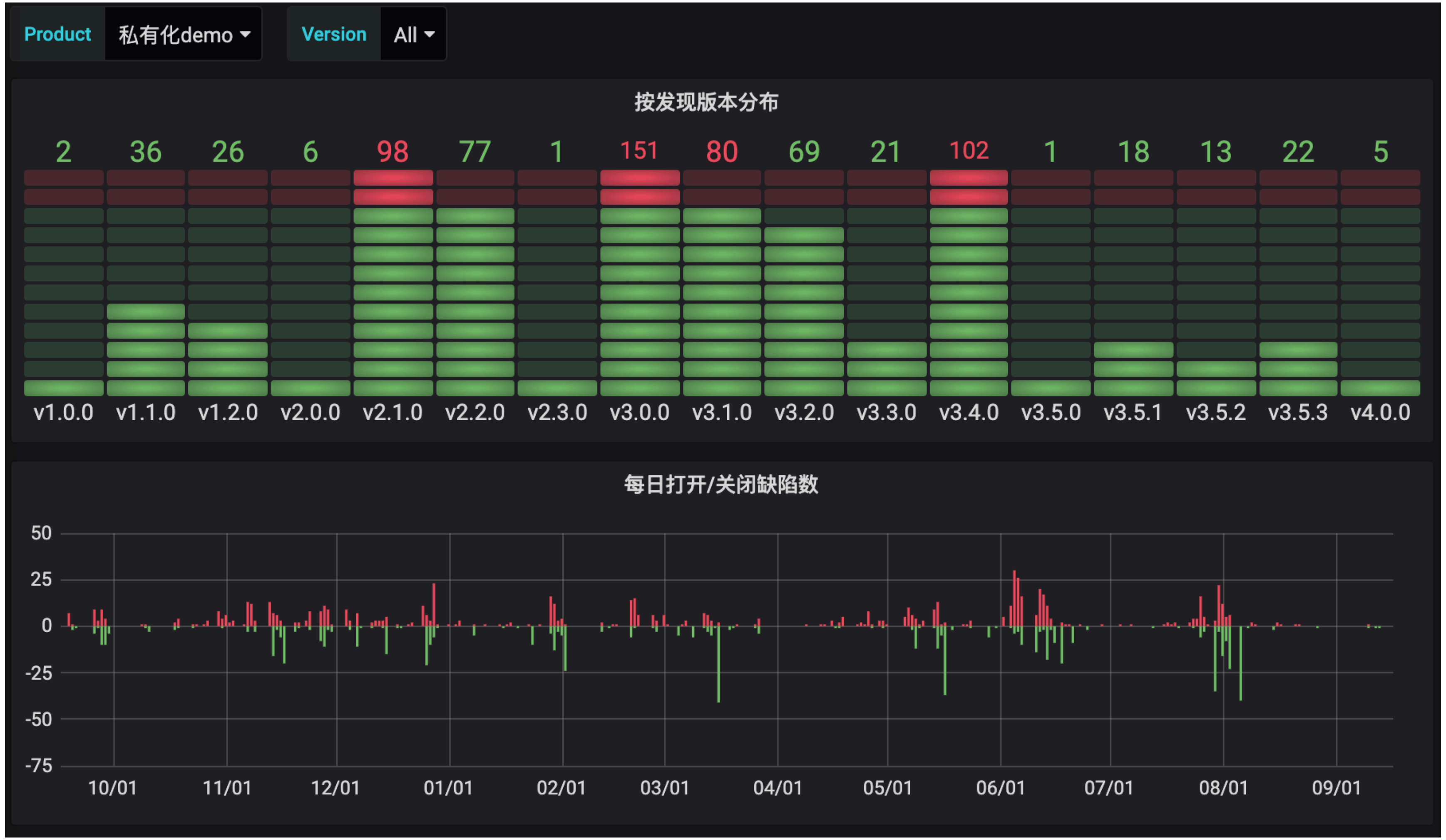
研发效能 > 缺陷分析 / Settings

Variables

Variable	Definition
$Product	$product
$Version	$bugversion=$Product

实现

```python
class _TagQuery(object):
    # tag query functions
    @_tag('product')
    async def query_products(self, q):
        return await Product.distinct('name')

    @_tag('component')
    async def query_components(self, q):  # q is product name
        if q and q != '*':
            return [c['components']['name'] async for c in Product.aggregate([
                {'$match': {'name': q}},
                {'$lookup': {'from': 'component', 'localField': 'components', 'foreignField': '_id', 'as':
'components'}},
                {'$project': {'components.name': True, '_id': False}},
                {'$unwind': '$components'},
            ])]
        return await Component.distinct('name')
```



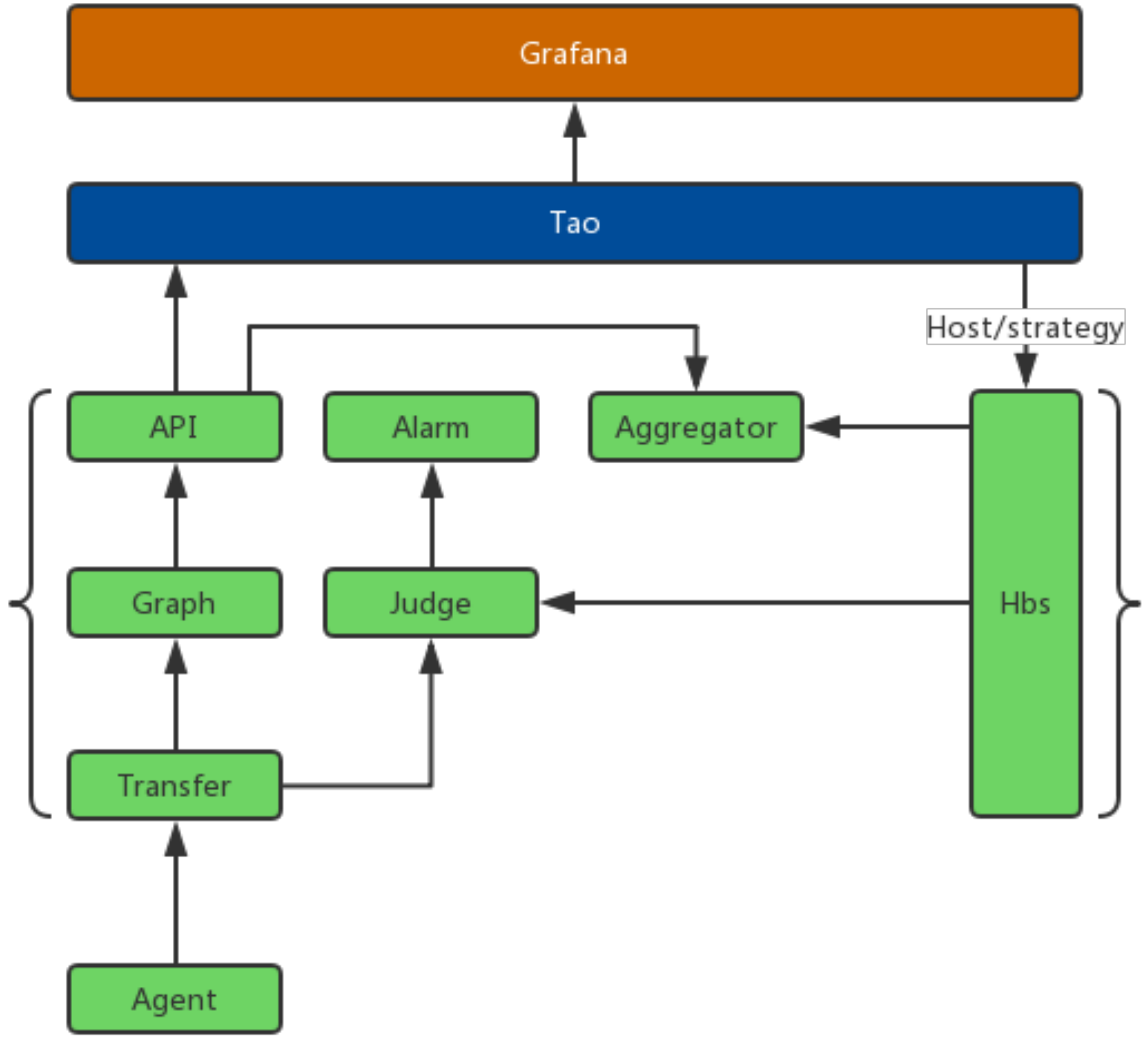
@tapd#product#version#bugs/product=$Product

@tapd#bugs#change#daily/product=$Product,version=$Version

实现

```
class TapdMetrics(AbstractMetrics):
    ENDPOINT = '@tapd'

    @metric('product.version.bugs')
    async def bugs_count_by_version(self, params):
        filter_ = await self._parse_common_params(params)
        if filter_ is None:
            return []

        return [
            self.build_falcon_record(
                [{'value': item['count']}],
                endpoint=item['version'] or 'N/A',
                step=_1DAY
            ) async for item in TAPDBug.aggregate([
                {'$match': filter_},
                {'$group': {'_id': '$version_report', 'count': {'$sum': 1}}},
                {'$project': {'_id': False, 'version': '$_id', 'count': True}}
            ])
        ]
```

实现 - 对falcon-plus的改造



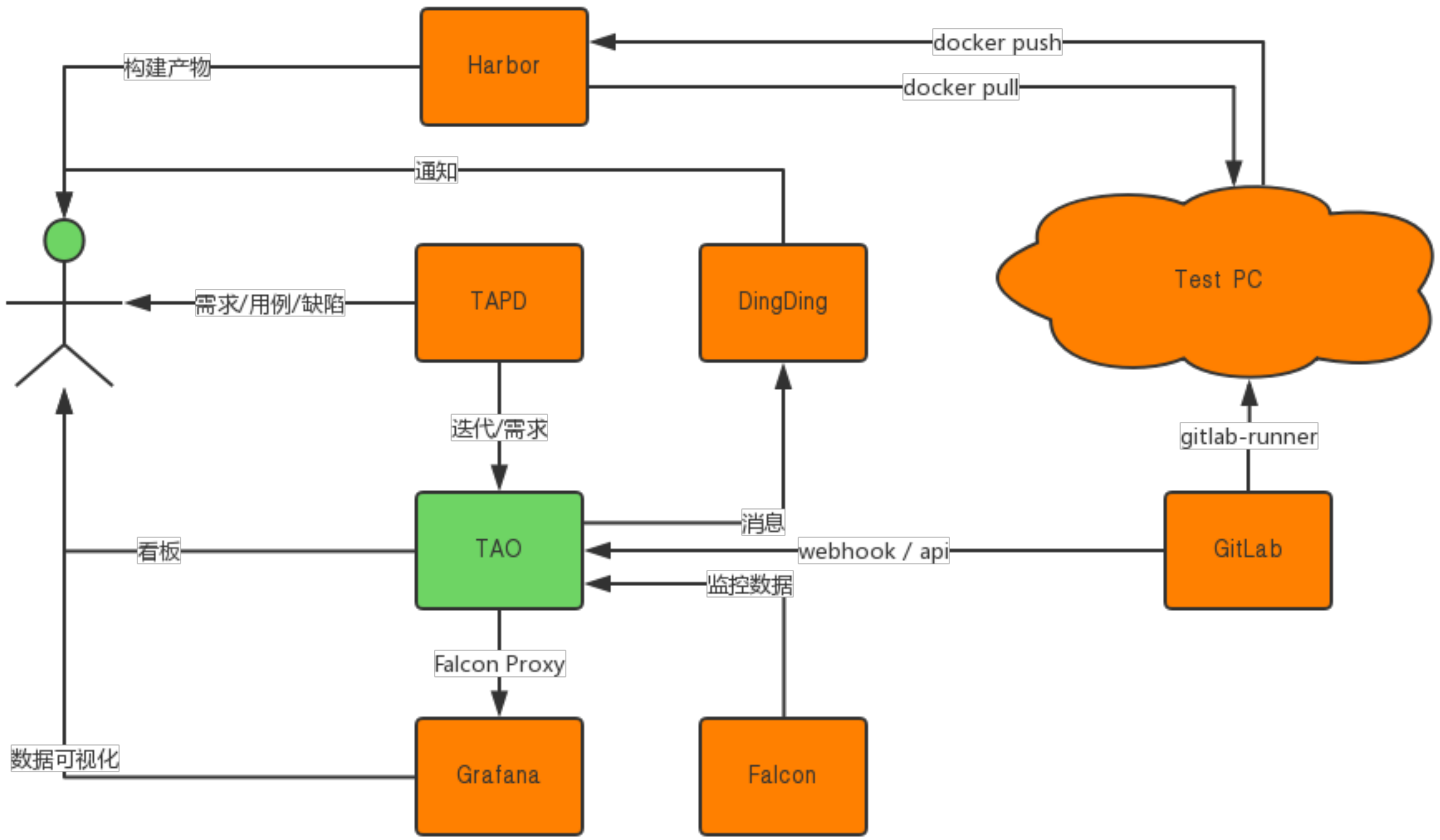
PyConChina 2019

实现 - pandas友好的数据查询接口

GET /api/v1/tapd/bugs?workspace_id=12345678&fields=id,status

```
{"id":"10001","status":"closed"}
{"id":"10002","status":"closed"}
{"id":"10003","status":"closed"}
{"id":"10004","status":"closed"}
{"id":"10005","status":"closed"}
{"id":"10006","status":"closed"}
{“id":"10007","status":"closed"}
```

```python
import ujson
import requests
import pandas as pd


def _read_bugs(workspace_id):
    return requests.get(
        'http://tao.local/api/v1/tapd/bugs',
        {'workspace_id': workspace_id, 'fields': 'id,status'}
    ).iter_lines()

bugs = pd.DataFrame((ujson.loads(line) for line in _read_bugs('12345678')))
```



PyConChina 2019

PYTHON 30th

PyConChina 2019

6340 python + 6386 javascript

PYTHON 30th

PyConChina 2019

PYTHON 30th

PyConChina
2019

THANK YOU

PYTHON 30th

@feiyuw