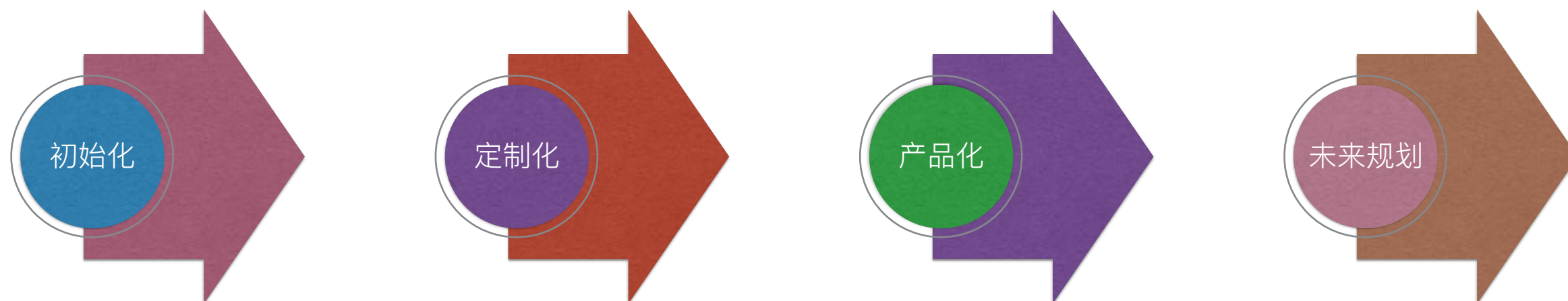


美团点评监控系统Falcon发展历程

SRE 殷大闪

大纲



初始化阶段

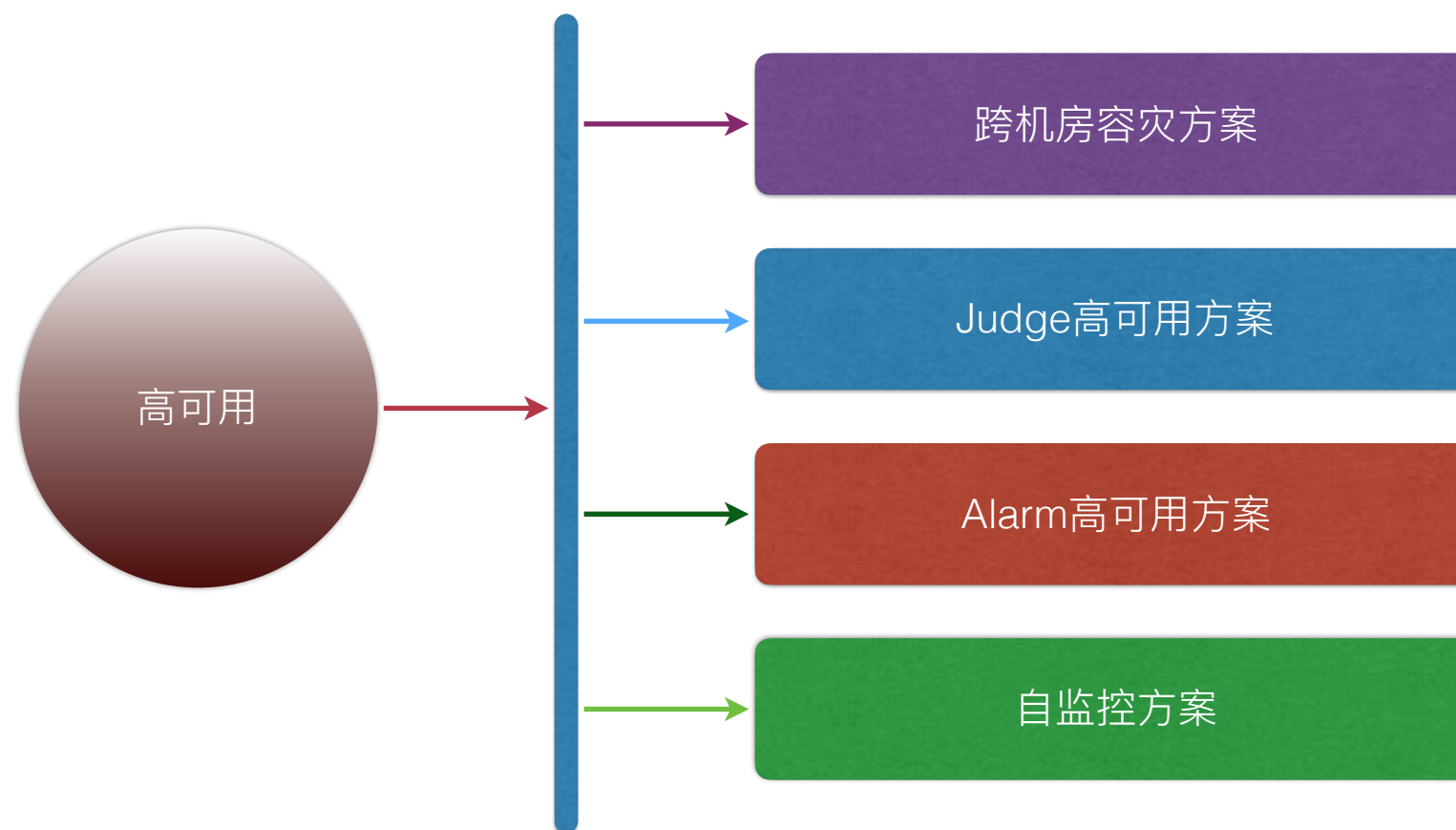


- 2015年5月份Open-Falcon开源，开始引入Open-Falcon
- 2015年9月份在公司正式上线Falcon监控系统
- 2015年底完成Zabbix到Falcon的迁移

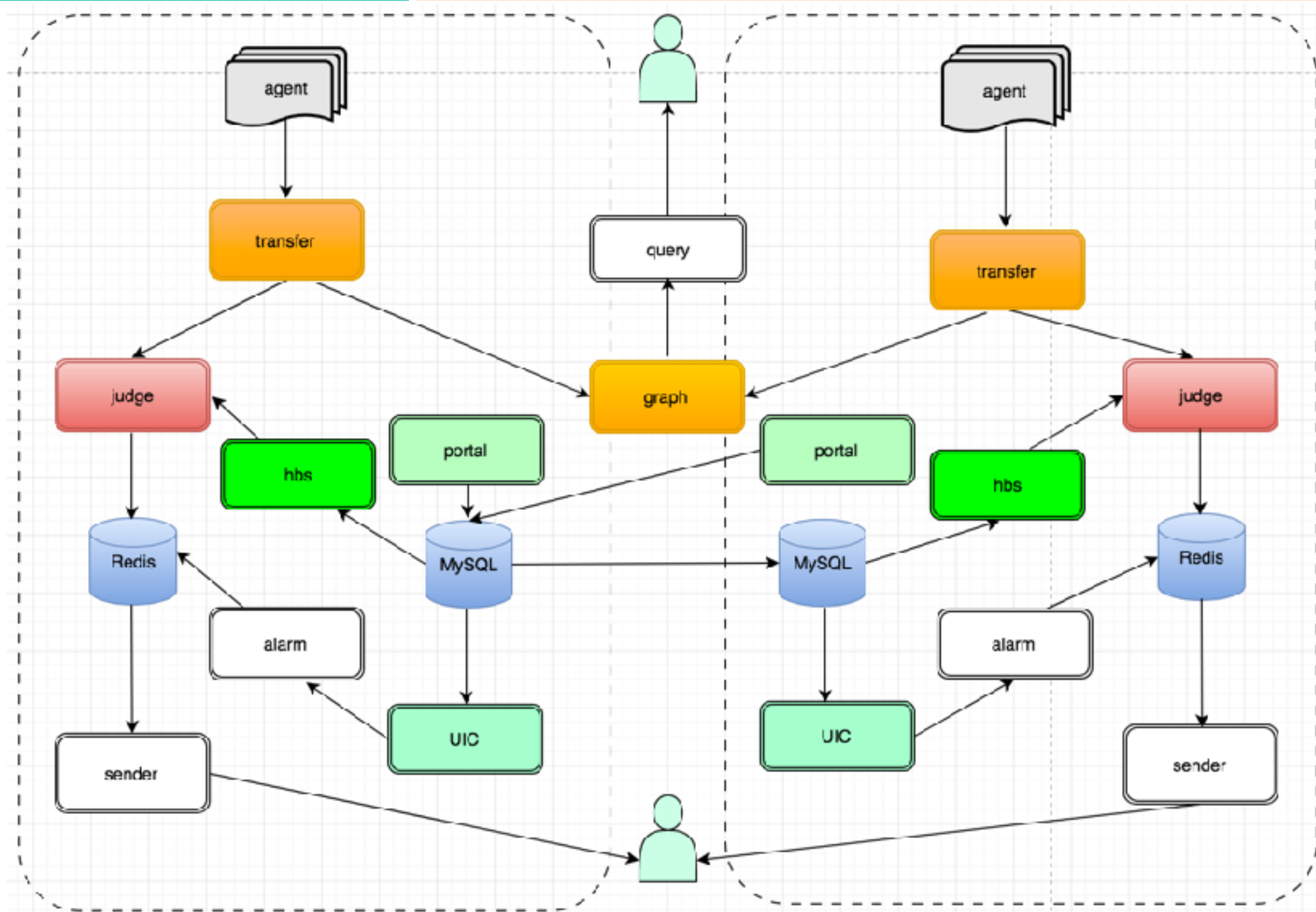
定制化阶段



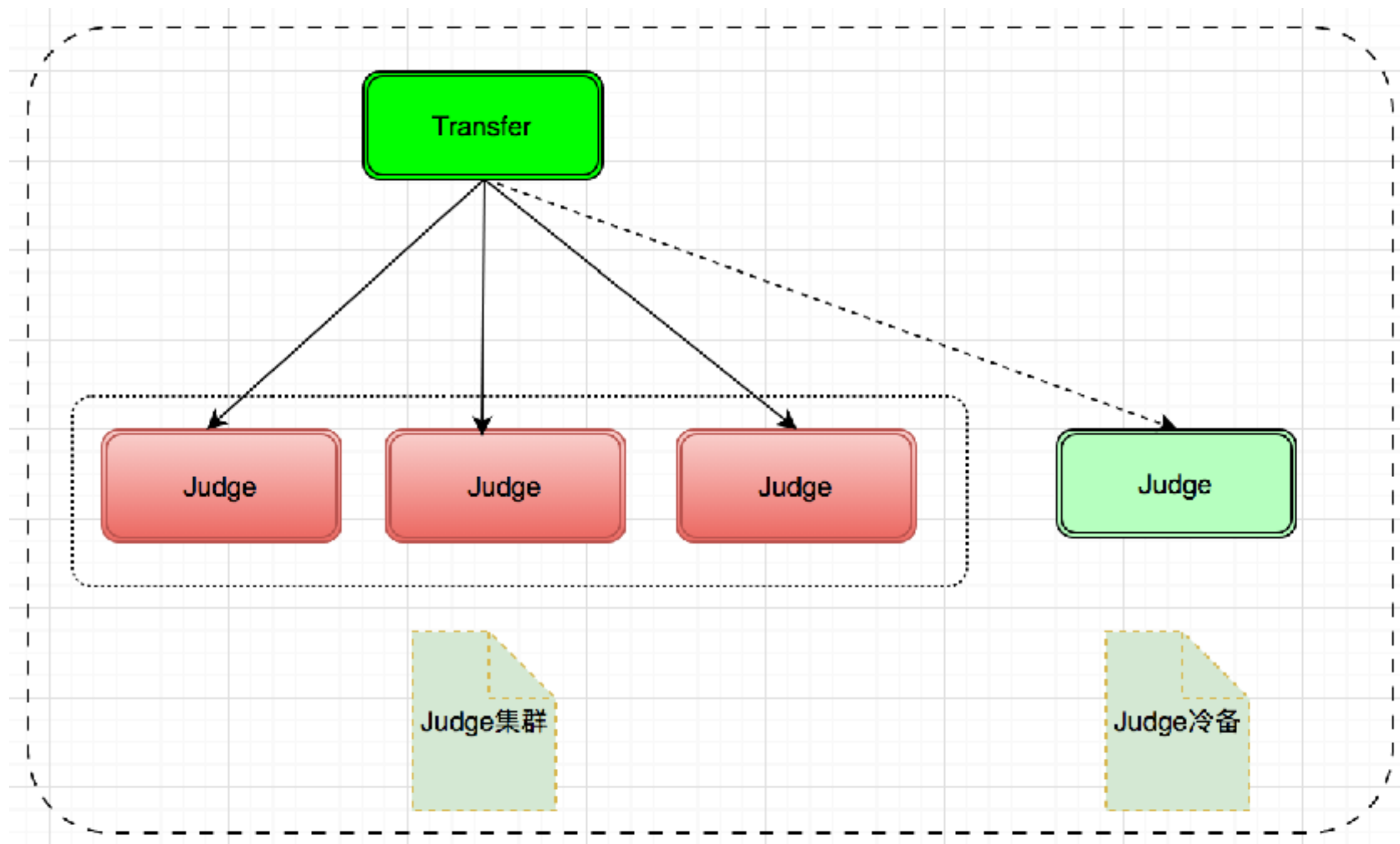
定制化阶段 — 可用性



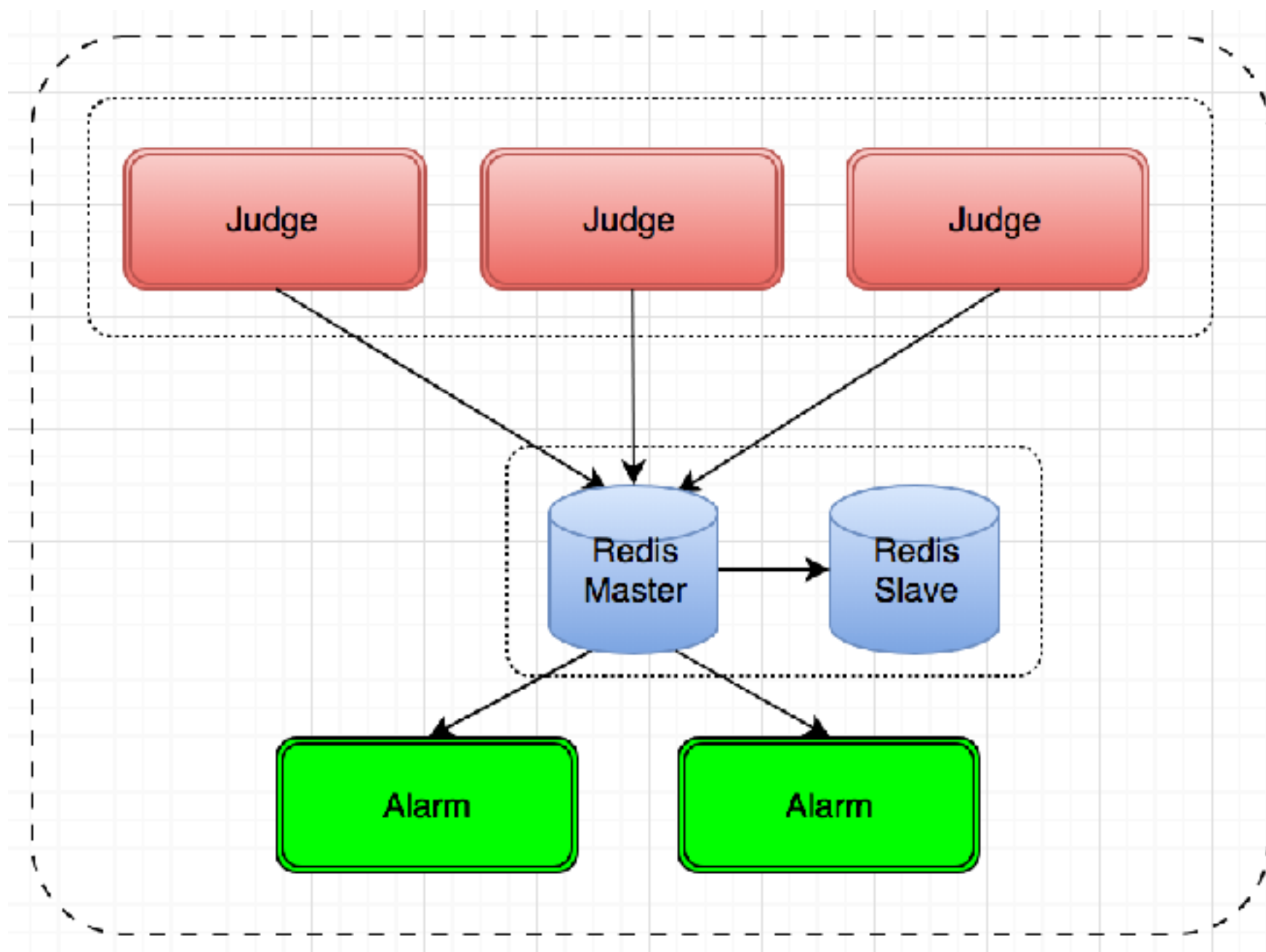
可用性—跨机房容灾方案



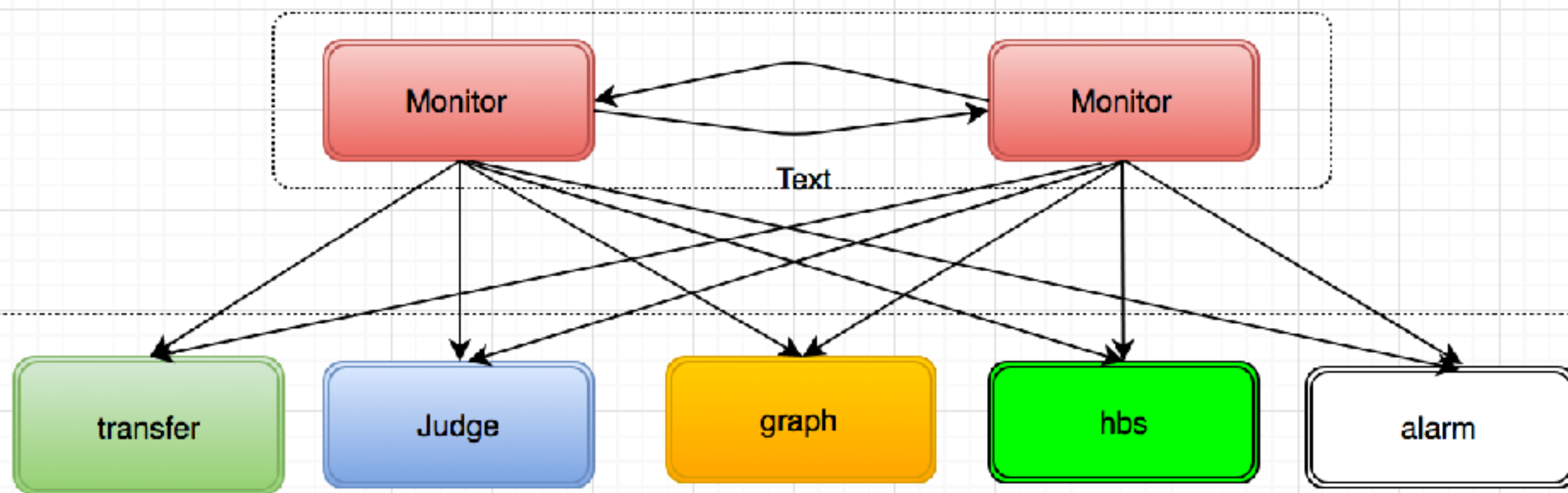
可用性 — Judge高可用方案



可用性 — Alarm高可用方案



可用性—自监控方案



定制化阶段 — 可扩展性



01

索引存储改造，过期索引自动删除和重建，手动维护



02

指定监控项存入OpenTSDB



03

告警升级、ACK、禁用、发散、合并、自动处理



04

Ping监控、字符串监控、多条件监控

可扩展性 — 多条件监控



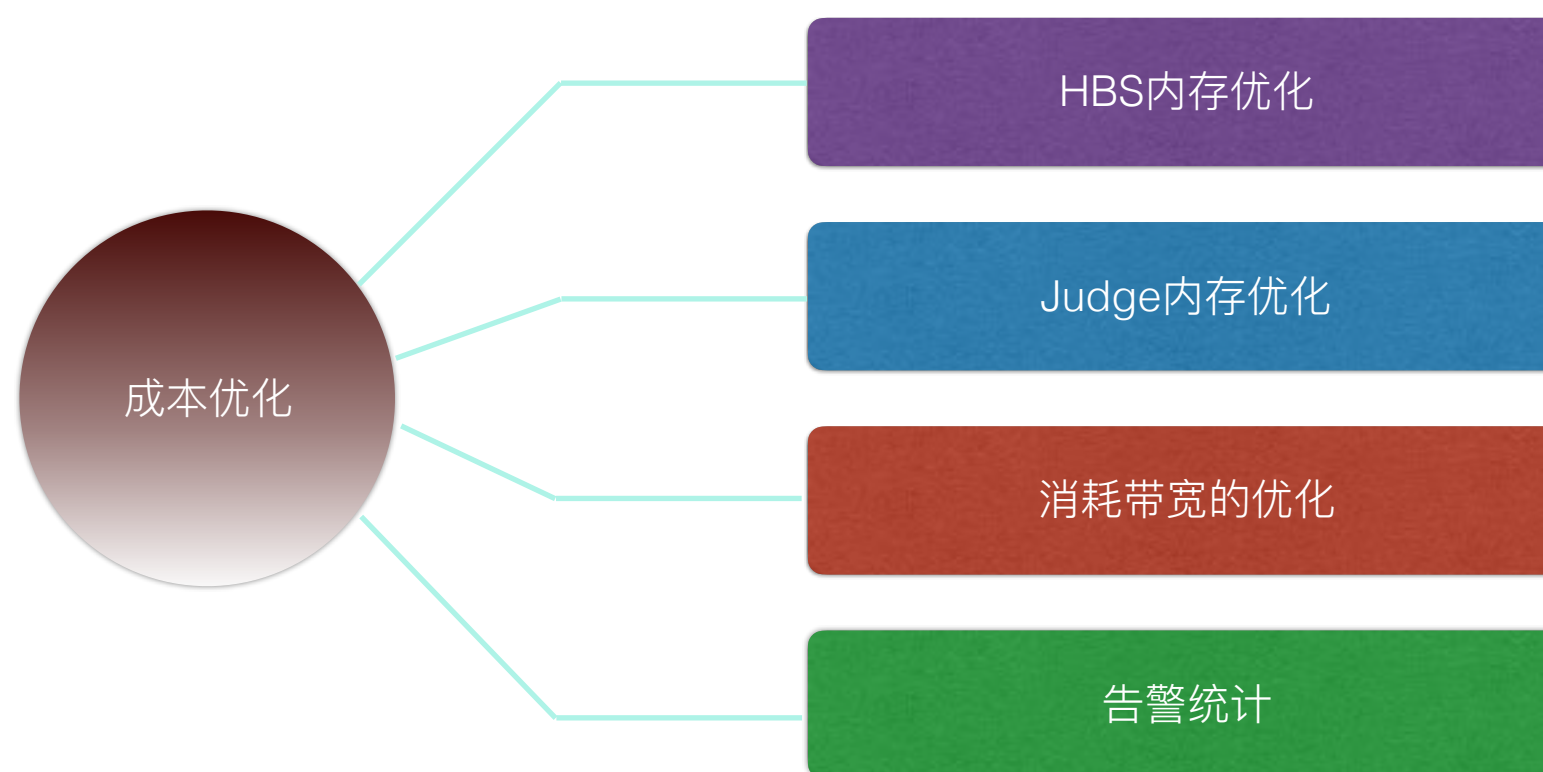
cpu.idle [cpu空闲率]	all(#1)>0	10	2	大象+邮件
cpu.system [cpu系统使用率]	all(#1)>0			
cpu.user [cpu用户使用率]	all(#1)>0			



美团点评P2告警

[P2][故障][cq-inf-falcon-test02][{cpu空闲率 all(#1) cpu.idle > 0 当前值:99.24433}; {cpu系统使用率 all(#1) cpu.system > 0 当前值:0.25189}; {cpu用户使用率 all(#1) cpu.user > 0 当前值:0.50378}][报警模版:test_falcon_tpl][第2次 2017-03-14 10:12:00][持续时间:5 minutes][[服务树](#)][[dashboard](#)][[ACK](#)]

定制化阶段 — 成本优化



定制化阶段 — 易用性



易用性——一键创建Screen



Mt-Falcon-Dashboard

Screen 报警红盘 使用手册 常用监控项 索引自维护 链接 联系我们

搜索Endpoints

根据Endpoint/主机名筛选 使用说明

dx-inf-falcon-judge

根据Endpoint/主机名筛选第二条件 使用说明

可以用空格分割多个搜索关键字，不同关键字取交集，不同条件取并集

根据标签(eg: job=appstore-web)筛选

根据服务节点查找

根据VM/宿主机(主机名完全匹配)查找

全局搜索 Limit 50

支持shift多选

刷新counter列表

新增screen

一级screen_name

Falcon监控系统

二级screen_name

输入二级screen名

falcon

hbase-falcon

falcon_api

falcon-tair基础监控

falcon使用的hbase监控

falcon测试环境

falcon-agent灰度测试

falcon

删除counter 添加counter 新增screen 网卡更改

查看

	类型	频率
	原始值	86400s
	原始值	60s
	原始值	60s
	原始值	60s
	原始值	60s
	原始值	60s
	原始值	60s
	原始值	60s
	原始值	60s
	计数器	60s
	原始值	60s

dx-inf-falcon-judge01

dx-inf-falcon-judge02

cpu

Counter

cpu

Counter

搜索

Counter

cpu.9

cpu.bus

cpu.quest

cpu.idle

cpu.lowait

cpu.inq

cpu.nice

cpu.softirq

cpu.steal

cpu.switches

cpu.system

易用性 — Screen收藏夹



收藏夹

♥ / [性能监控](#)

♥ / [Network](#)

♥ / [graph异常检测](#)

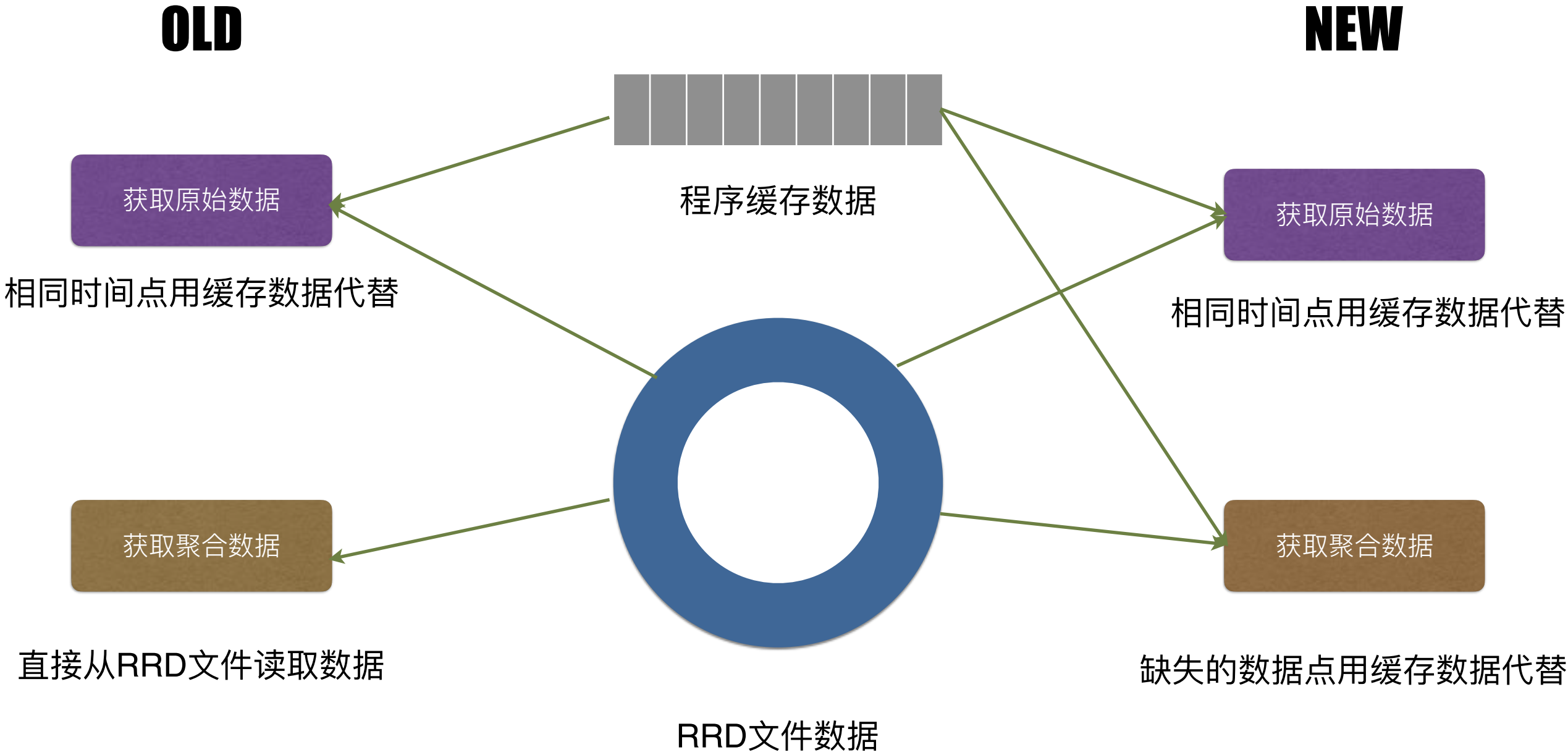
♥ / [wiki](#)

♥ / [Test](#)

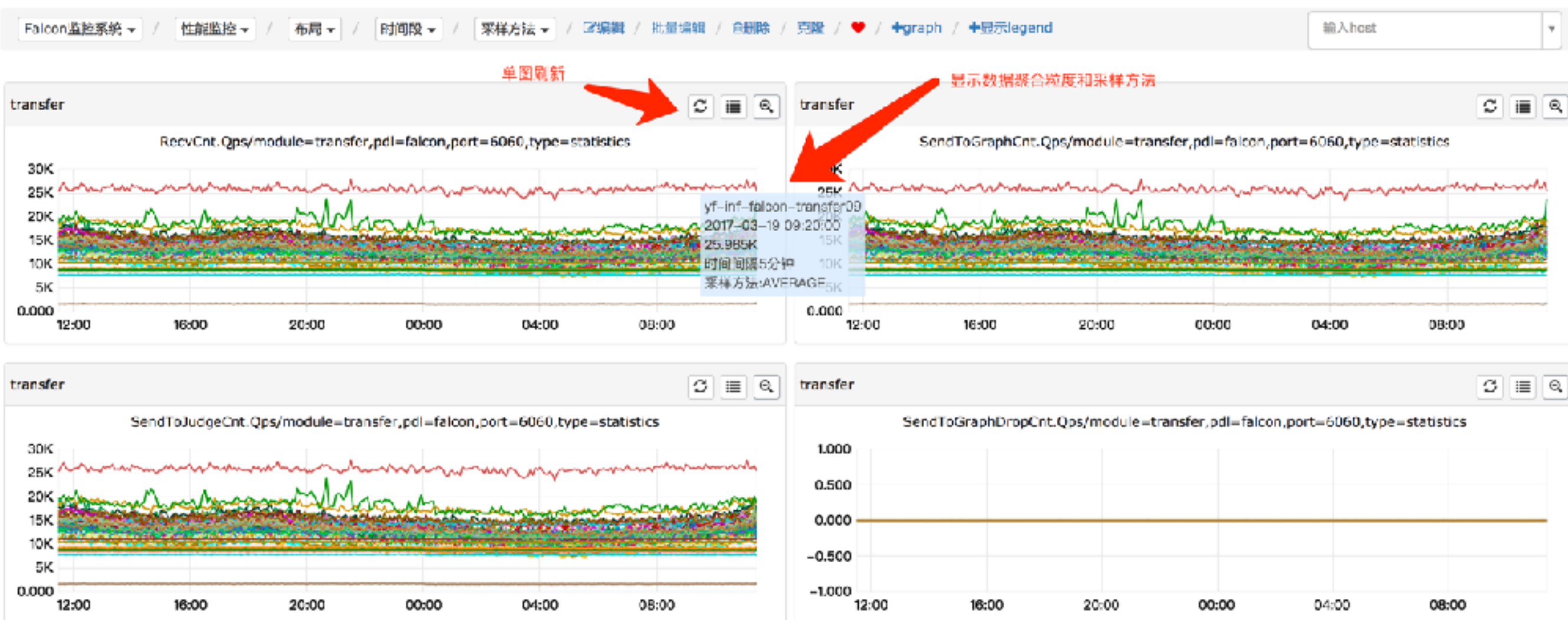
♥ / [基础性能](#)

♥ / [falcon使用的hbase监控](#)

易用性 — 解决查询历史数据时最新数据点丢失的问题



易用性——单图刷新并显示数据聚合粒度和采样方法



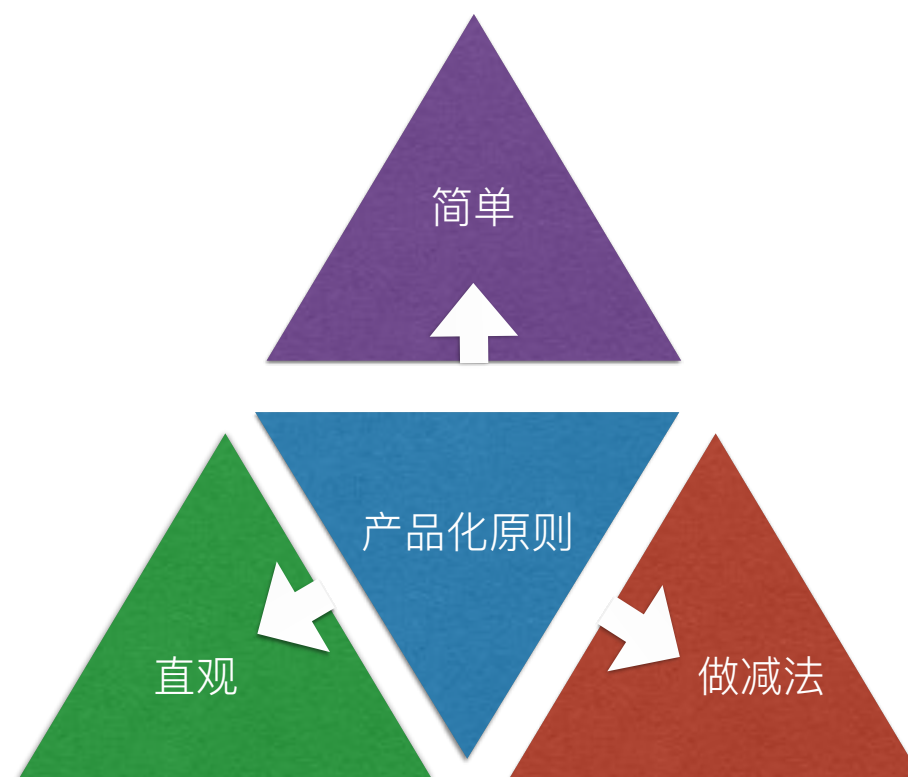
易用性 — Tag反选



- 监控所有的磁盘，只需要配置一条监控策略就可以 (metric为disk.io.util)
- 需要过滤掉某几块盘，一条策略也可以搞定(metric为disk.io.util, tags为^device=sda|sdb)
- 自行上报数据的时候也尽量利用tags的这个特性，简化监控配置

- ☐ disk.io.util/device=sda
- ☐ disk.io.util/device=sdb
- ☐ disk.io.util/device=sdc
- ☐ disk.io.util/device=sdd
- ☐ disk.io.util/device=sde
- ☐ disk.io.util/device=sdf
- ☐ disk.io.util/device=sdg
- ☐ disk.io.util/device=sdh
- ☐ disk.io.util/device=sdi
- ☐ disk.io.util/device=sdj
- ☐ disk.io.util/device=sdk
- ☐ disk.io.util/device=sdl
- ☐ disk.io.util/device=sdm

产品化阶段



产品化阶段 — 节点直接和策略绑定



corp=meituan&owt=sre&pdl=falcon 监控策略

+ 新增策略

+ 新增组合策略

<input type="checkbox"/>		指标(metric/tag/note)	报警条件(condition)	HostGroup	最大报警次数	报警级别
<input type="checkbox"/>		cpu.idle	all(#3)<10	corp=meituan&owt=sre&pdl=falcon	3	P2
<input type="checkbox"/>		cpu.busy	all(#3)==0	corp=meituan&owt=sre&pdl=falcon	0	P2
<input type="checkbox"/>		cpu.system	all(#3)==0	corp=meituan&owt=sre&pdl=falcon	0	P0
<input type="checkbox"/>		cpu.guest	all(#3)==0	corp=meituan&owt=sre&pdl=falcon	0	P0
<input type="checkbox"/>		cpu.idle	all(#3)<50	corp=meituan&owt=sre&pdl=falcon	3	P0
		mem.memfree.percent	all(#3)<10	corp=meituan&owt=sre&pdl=falcon		

报警接收人

第二报警接收人

☐ 发给SRE/RD/测试负责人 ?

☒ 发给值班人

callback地址

产品化阶段 — 编辑策略



策略配置

基础配置

指标	cpu.idle	×	▼	标签	例如: a=b,c=d	备注						
表达式	all(#3)	<	↕	10	最大报警次数	3	报警级别	P2	▼	报警方式	大象+邮件	▼
过滤子节点	输入子节点的tag			开始生效时间				结束生效时间				

报警配置

第一报警接收组	× yindashan		
第二报警接收组	input username		
<input type="checkbox"/> 发给SRE/RD/测试负责人	<input checked="" type="checkbox"/> 发给值班人	callback地址	

保存

关闭

产品化阶段 — 要解决的痛点



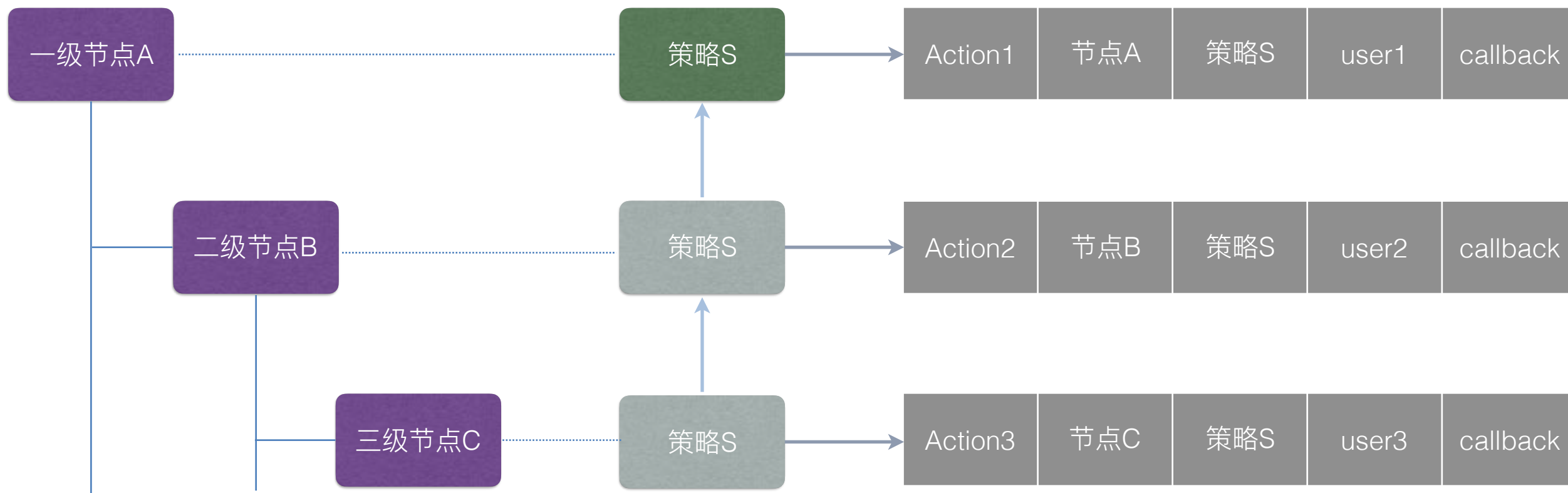
- 监控权限控制比较麻烦
- 多模板情况下配置策略比较复杂
- 看不到节点最终生效的策略列表
- 模板跟告警组关联，往告警组里面添加、删除用户的时候还要去告警组管理模块单独维护
- 分环境监控和按机器维度监控实现方式太复杂

产品化阶段 — 解决方案



- 去掉模板的概念，节点直接跟策略一对多关联
- 策略支持过滤子节点
- 节点+策略的组合跟Action一对一关联，实现告警接收人可继承
- 去掉告警组的概念，Action直接与用户多对多关联
- 用打Tag的方式实现分环境监控和机器维度的监控

产品化阶段 — 告警接收人设计



未来规划



- 故障定位
- 数据分析
- 自动处理
- 深度挖掘

谢谢

Q&A

@殷大闪