

GIAC | BEIJING  
Dec.12.16-17

架構  
ARCHNOTES  
高可用架構

技术架构未来

thegiach.com

# 知乎的微服务架构实践

彭晟杰@Team Arches

# Agenda

---

- 背景和动机
- 服务是什么
- 服务通信
- 拆分单体应用
- 服务的粒度
- 服务间共享数据
- 开发、测试、交付、运维
- 团队和文化

# 背景和动机

# 2012 年的知乎网站

---

MVC 经典架构  
前后端未分离

一个 Codebase  
10+ 开发者

一周发布一次  
风险和成本都高

作为一家正常的创业公司，也想把握移动互联网的机会

# 2012 年的知乎移动端 API

---

- 快速启动：新的独立项目，打包主站，依赖之
- 快速迭代：单独的更高的部署频率
- 共享业务代码 😞

# Problems

---

- 主站和 API 发布节奏不同：重复代码或者逻辑不一致
- API 定期同步主站代码：风险极高
- 主站业务逻辑越来越复杂，模块之间耦合严重，很脆弱
- Codebase 越来越大：开发、测试、部署都慢



怎么办?

# 微服务!

# 微服务的好处

---

# 微服务的好处

---

按需异构

减少耦合

错误隔离

扩展灵活

快速迭代

.....

然而怎样从单体应用转化成微服务架构？





# 有关服务



那么究竟什么是服务？

# 服务的直观印象

---

# 服务的直观印象

---

- 单独运行的进程
- 提供 API
- 用于复用逻辑

# 服务的直观印象

---

- 单独运行的进程
  - 提供 API
  - 用于复用逻辑
- } 库

# 服务还是库

---

## 库的问题

- 库的本质是分发代码
- 库的变更成本大，升级困难，控制力弱

## 几个事实

- 业务逻辑涉及存储，存储要有约束地访问
- 业务逻辑需要频繁变更和交付
- 我们对业务逻辑要有强的控制力

# 什么适合做成服务

---

- 变更频繁、有存储的业务逻辑

# 什么适合做成服务

---

- 变更频繁、有存储的业务逻辑
- 节点有状态，需要控制节点规模（发号器）

有了服务，它们如何通信？



# 同步通信

---

复杂度

多语言支持

性能

约束力

---

RESTful API

小

超级好

一般

弱

RPC

大

看情况

一般更好

强

静态和强类型语言的公司可以选择 RESTful  
我们用 Python 的公司还是乖乖用 RPC

# 异步通信

---

- 需要消息中间件
- 保证幂等（如果 At Least Once）

# 同步还是异步

复杂度

耦合

逻辑集中程度

同步

小

强

强

异步

大

松

弱

# 同步还是异步？

---

- 不是关键逻辑（出错不会造成业务失败）
- 对时效性要求不高
- 执行时间相对长

可以选择异步，松耦合。

那怎么拆成微服务呢？

# 知乎主站的拆分

Templates

Business Logic

# 知乎主站的拆分

Templates

Business Logic



# 知乎主站的拆分

Templates



Business Logic Service

# 知乎主站的拆分

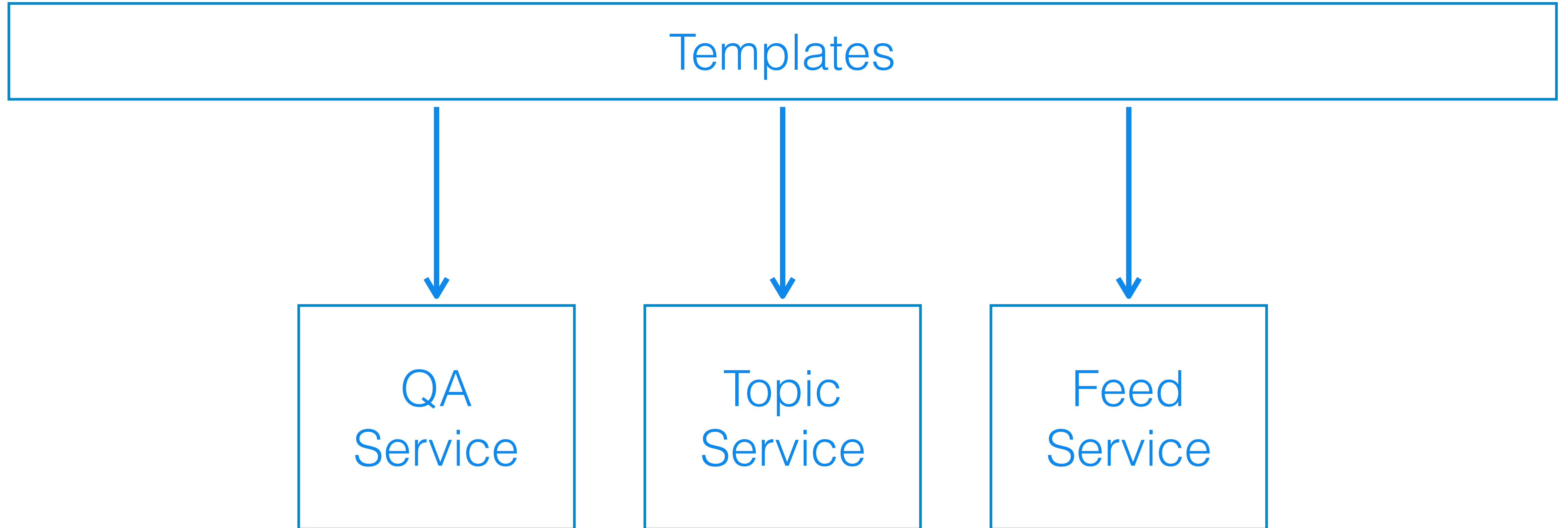
Templates

QA  
Service

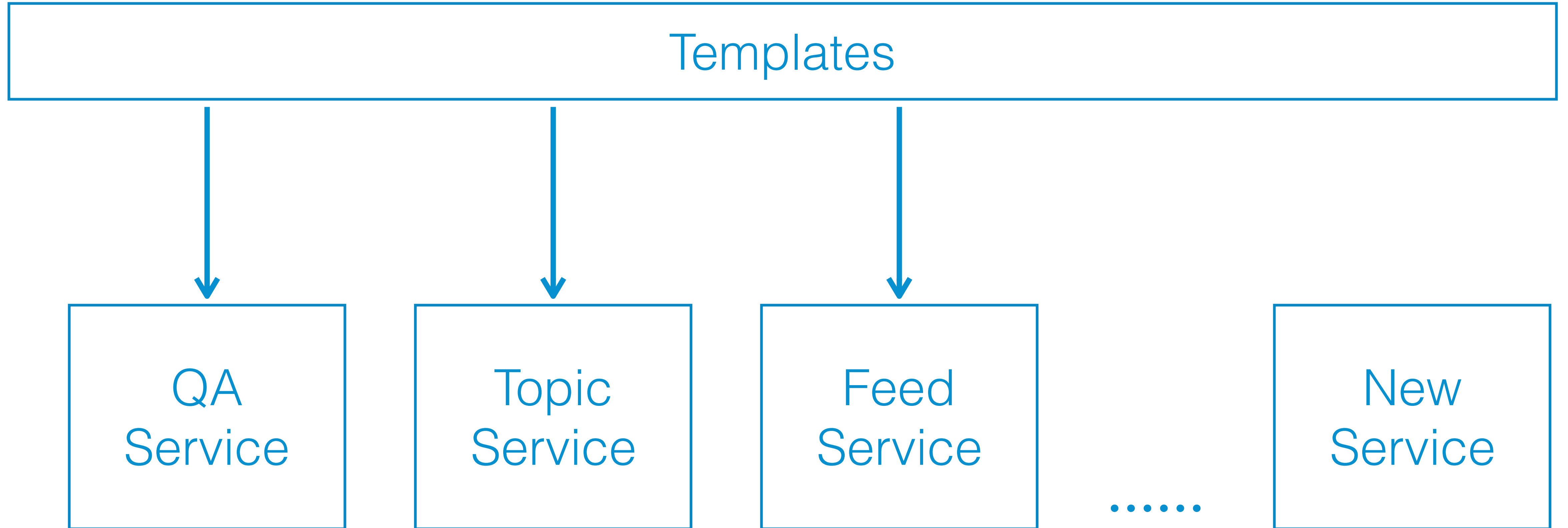
Topic  
Service

Feed  
Service

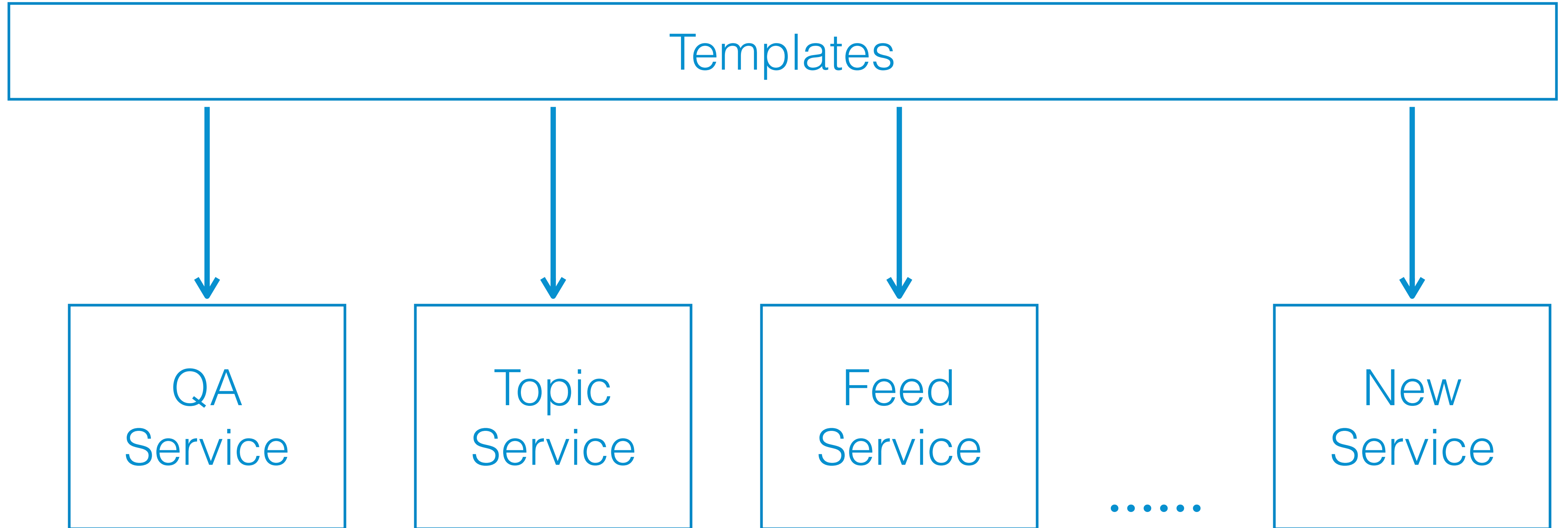
# 知乎主站的拆分



# 知乎主站的拆分



# 知乎主站的拆分



# 知乎主站的拆分的问题

---

# 知乎主站的拆分的问题

---

- RPC 服务接口非常多开发成本高
- 主站和 RPC 服务其实是强耦合：必须两边一起修改

# 知乎主站的拆分的经验

---

- 按照水平方向拆分虽符合经验，但是带来的问题比较多
- 根据业务从前到后，按照垂直方向拆分
- 再抽象出公共的逻辑，作为基础服务
- 然后为了特定的目的（比如为了稳定性和性能）进行水平拆分

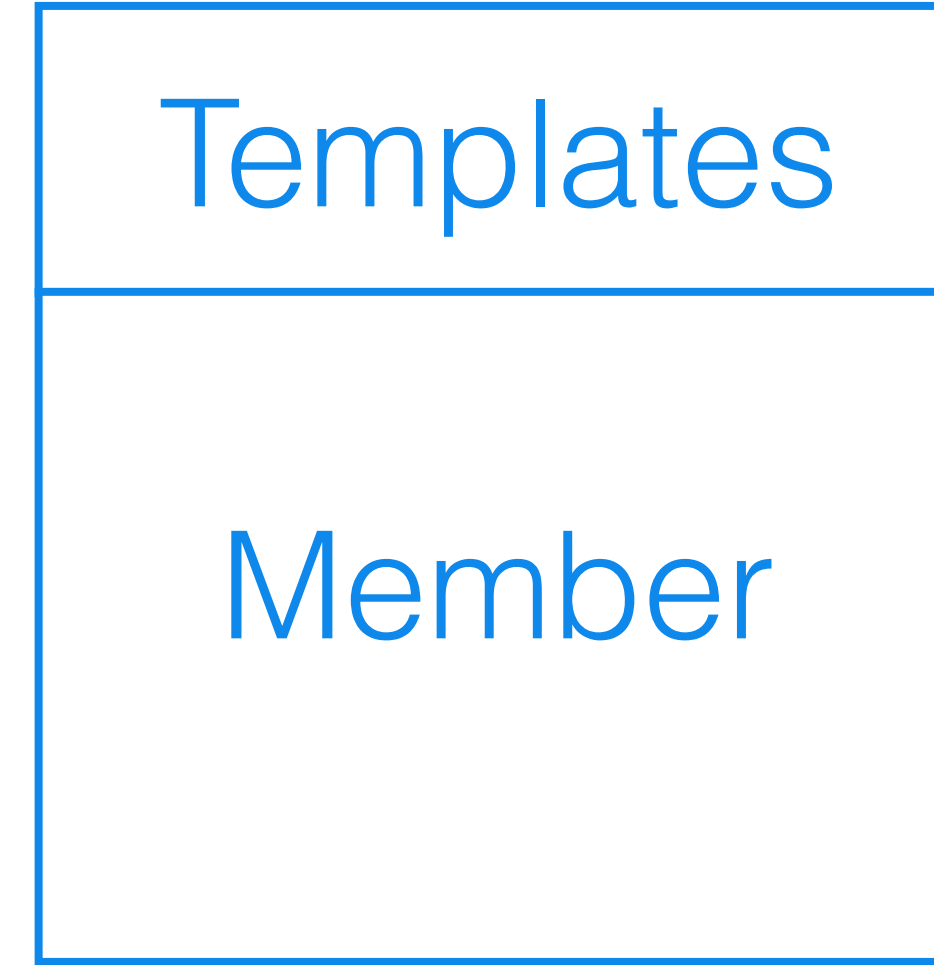
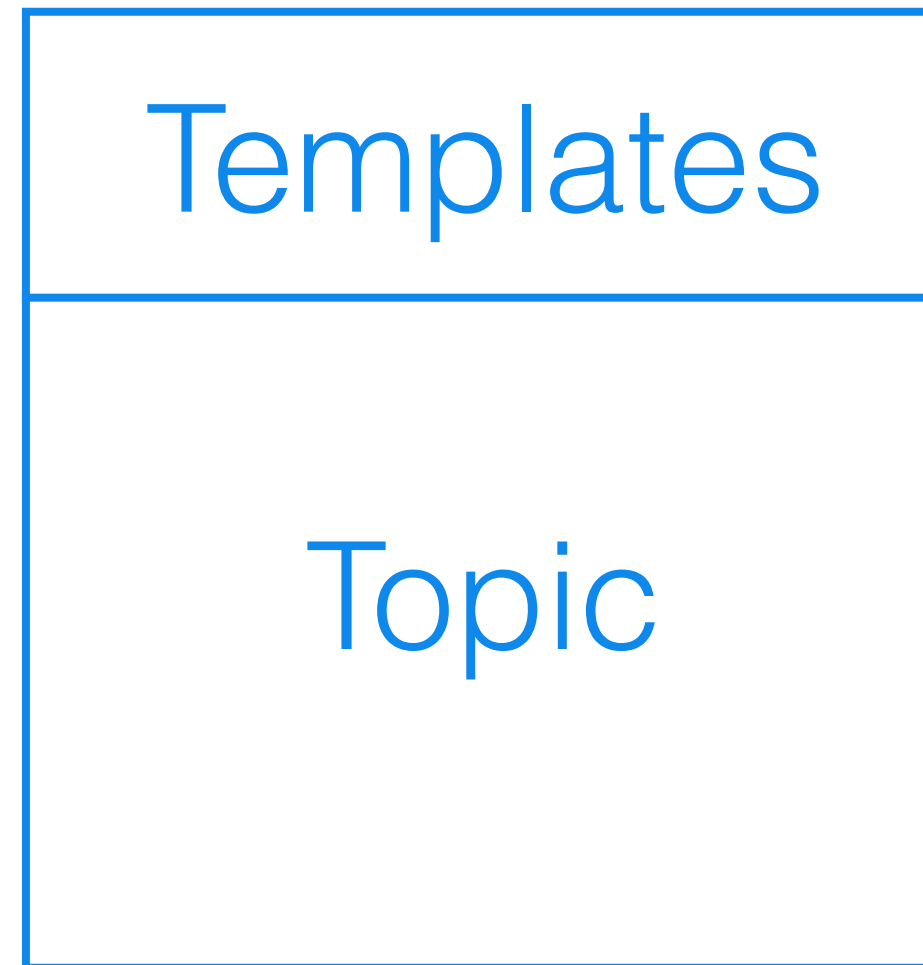
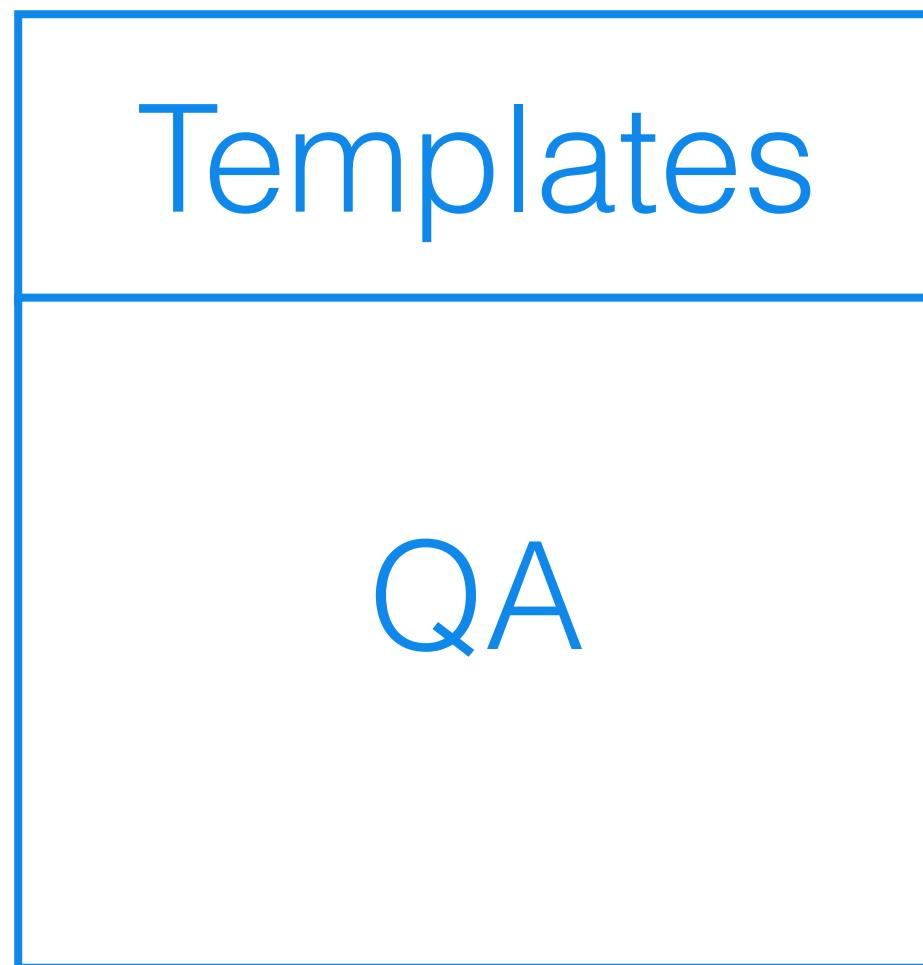


# 理想的拆分

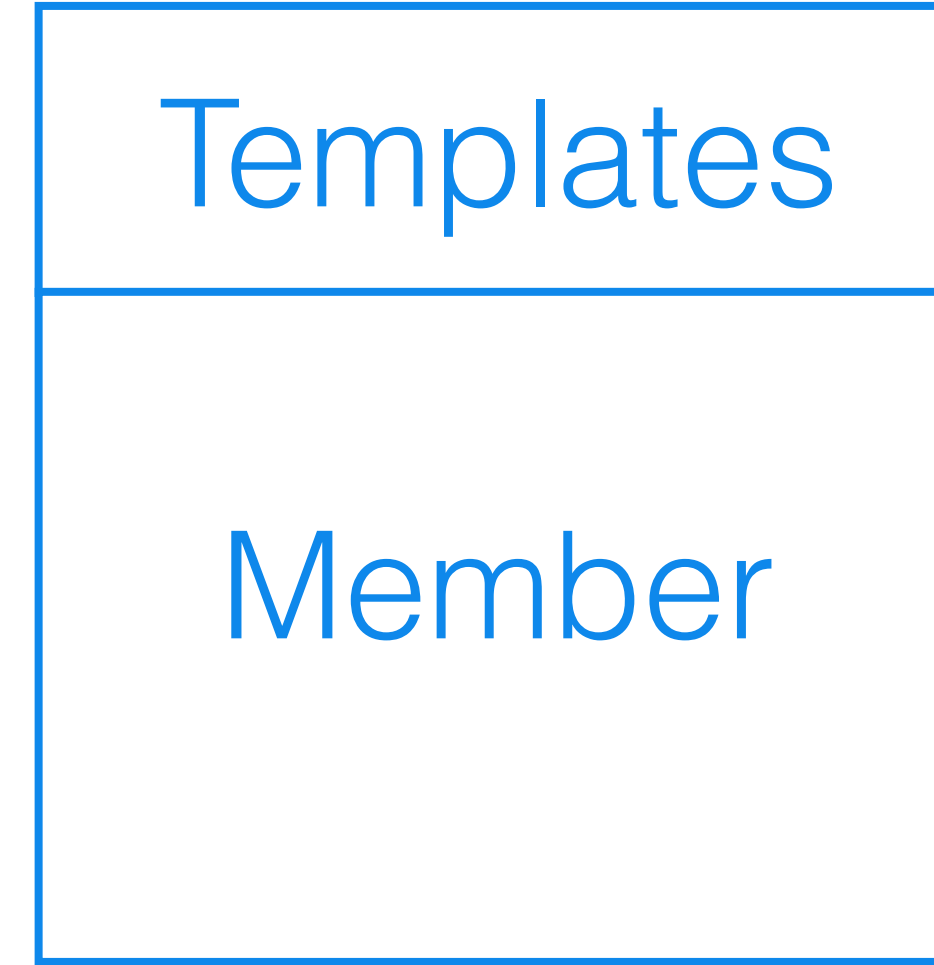
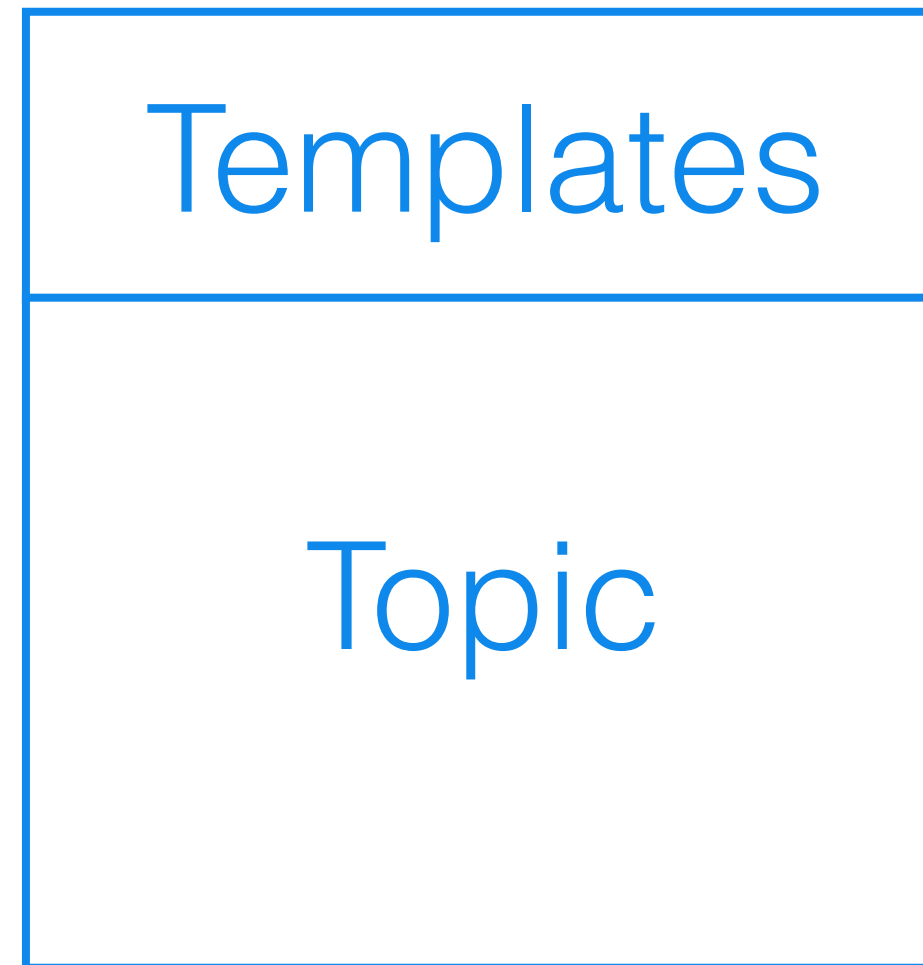
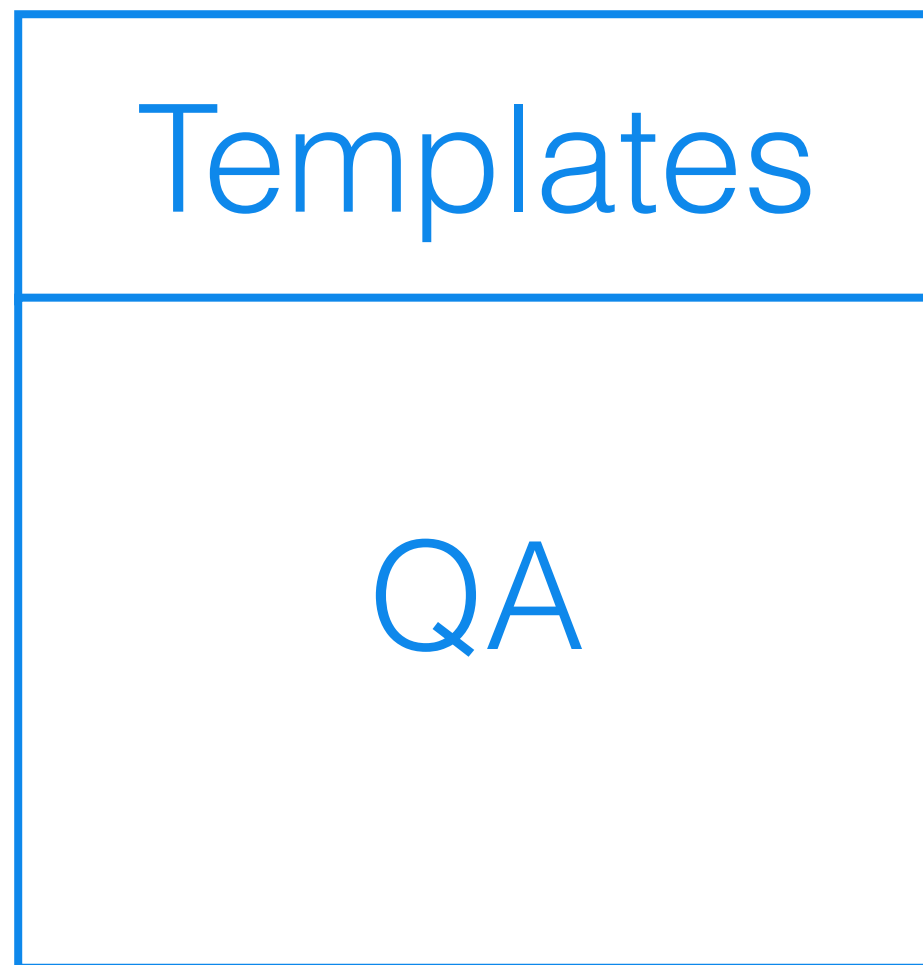
Templates

Business Logic

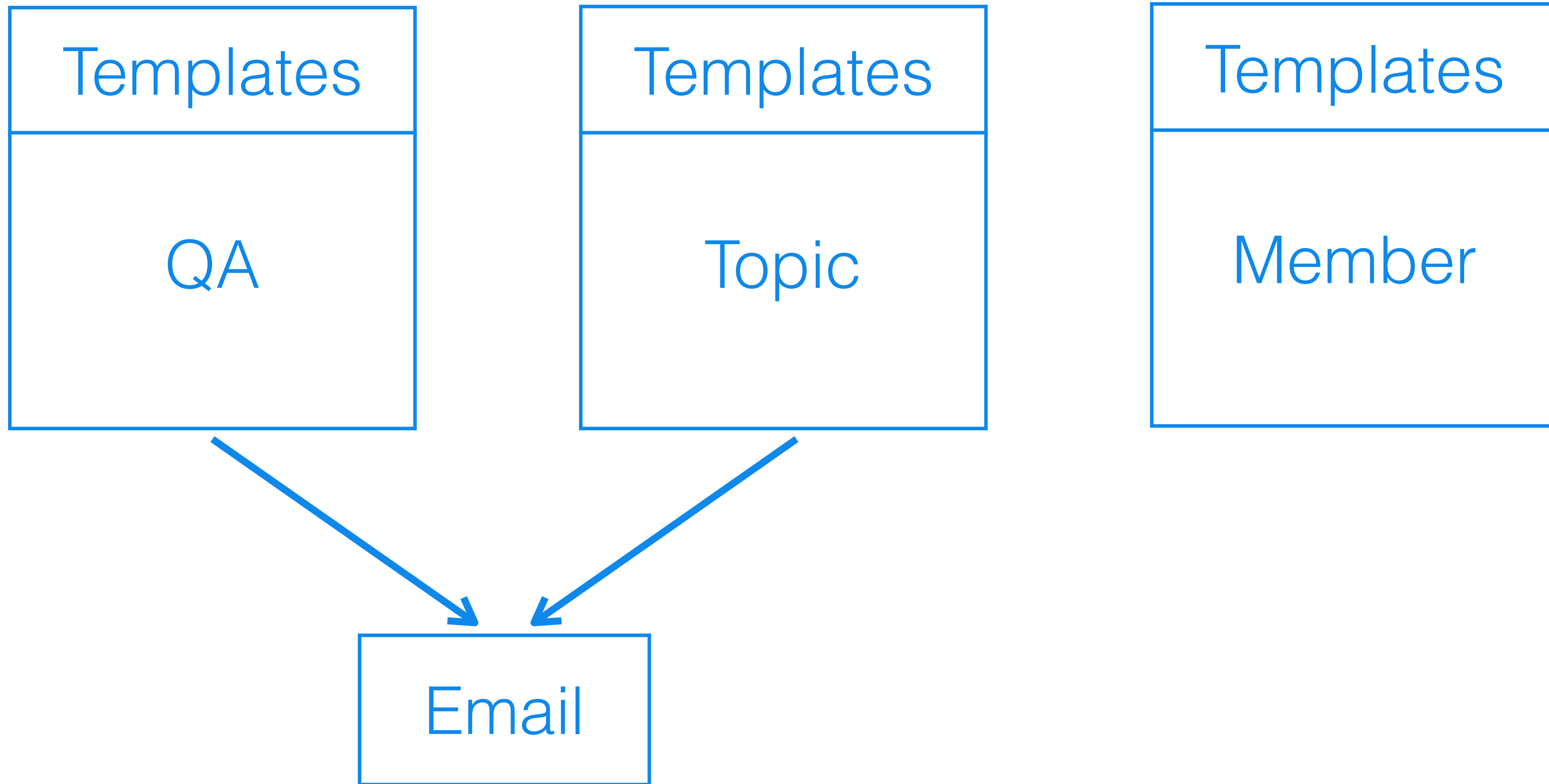
# 理想的拆分



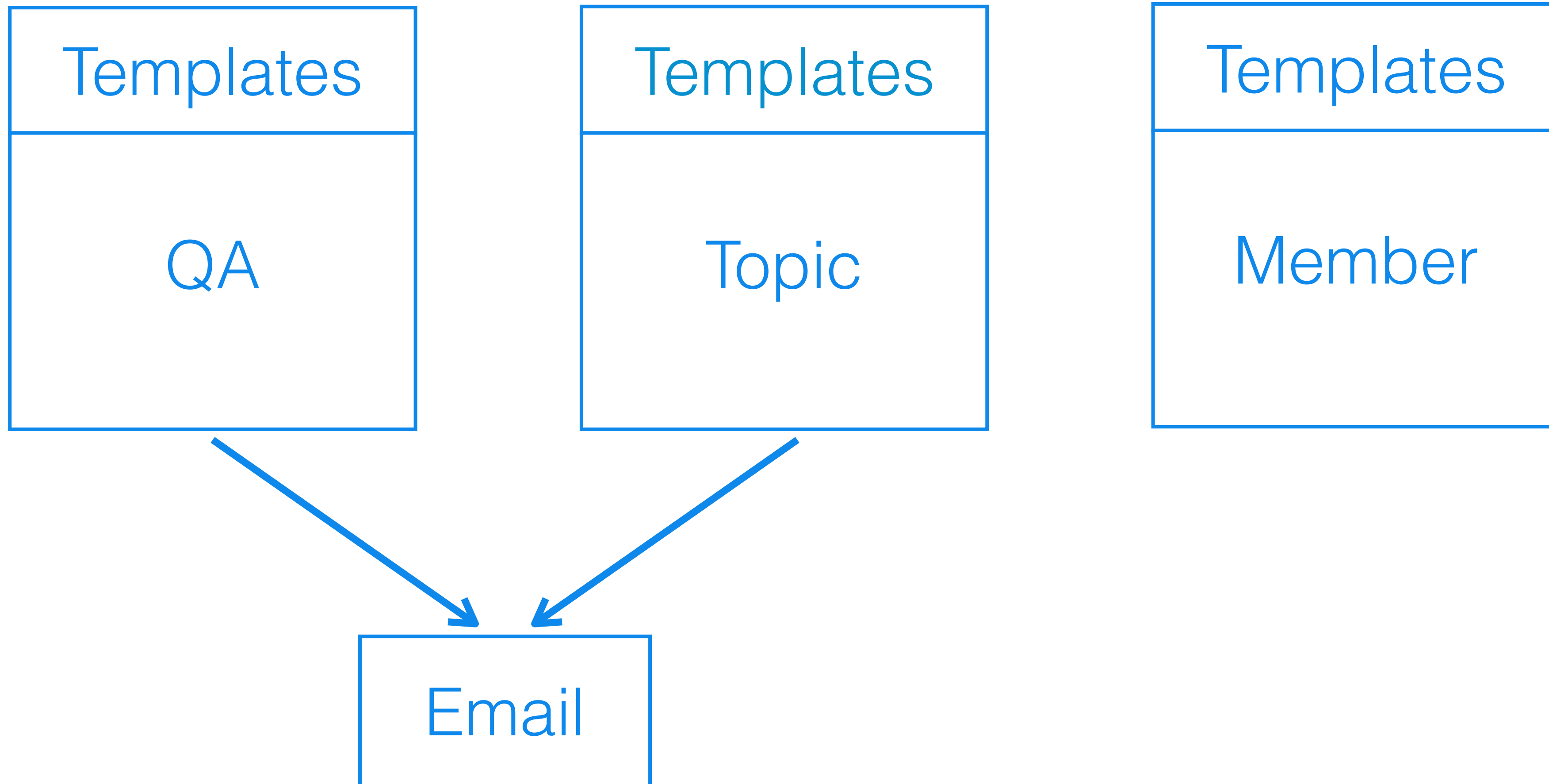
# 理想的拆分



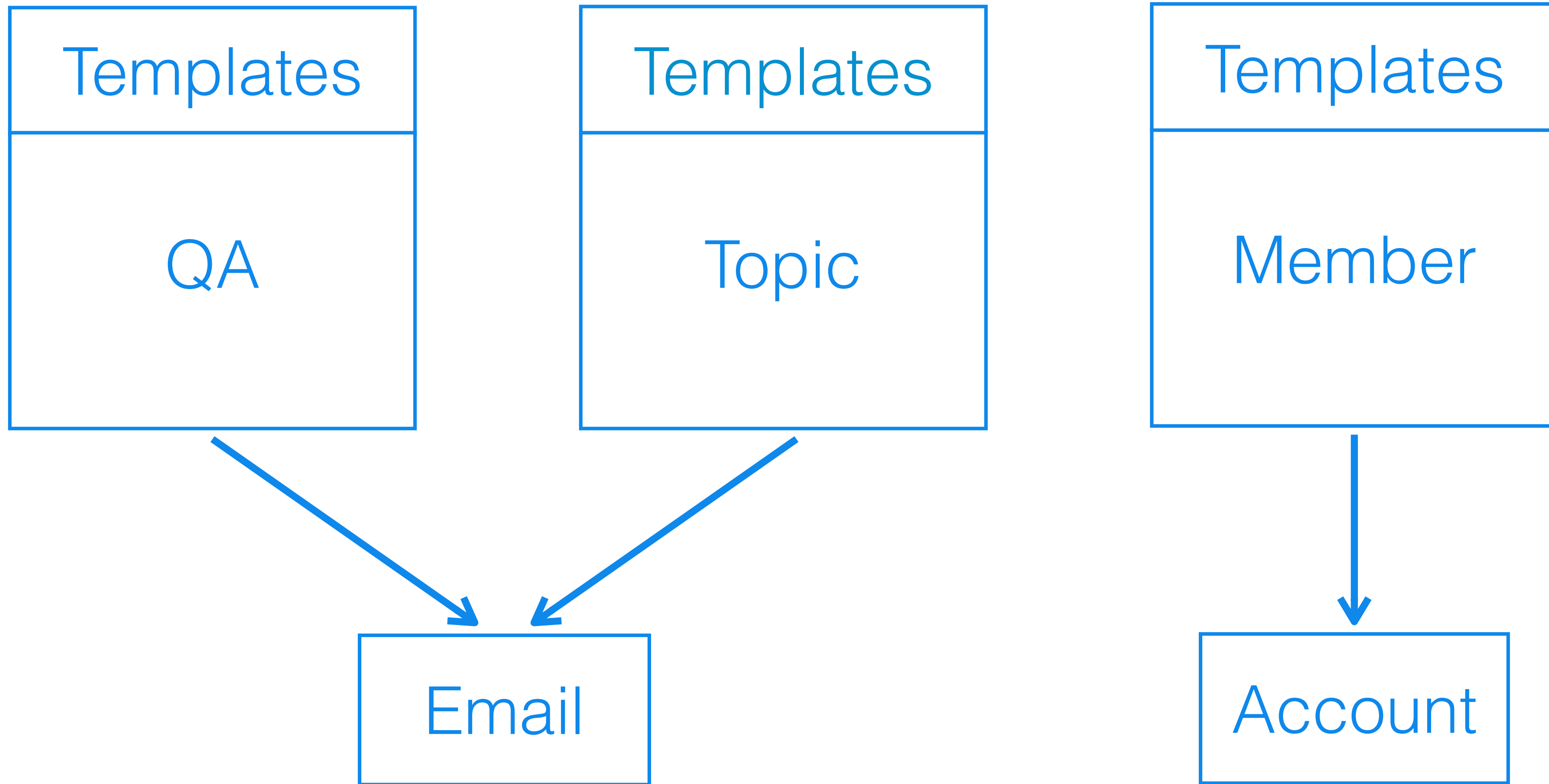
# 理想的拆分



# 理想的拆分



# 理想的拆分



服务的粒度应该如何把握？

# 微服务



# 微服务



# 微服务

# 粒度过细的问题

---

- 工程师要同时维护很多服务，维护和迁移工作繁重
- 服务的维护者只有一个人，明显的人力单点
- 耦合强，服务之间必须相互配合才能完成业务

合理的粒度应该是「小且自治」的，  
本质上是：「低耦合，高内聚」

# 合理粒度的特征

---

# 合理粒度的特征

---

- 日常开发在服务内部修改完成，不牵扯多个服务
- 一个服务仅被一个团队负责
- 一个服务的维护者有多个人，且都理解细节
- 负责服务的团队，有独立交付的能力

服务之间如何共享数据？





**不要共享存储！**

# 正确共享数据的方式

---

接口调用

意味着耦合

数据冗余

松耦合  
但是一致性难以保证

# 实施微服务的过程

首先是开发

# 开发微服务

---

# 开发微服务

---

- 不可能运行整个系统
- 维护一个公共环境
- 公共环境一定要有数据恢复的能力
- 定时备份，脏数据清理和 restore

开发完了需要测试

# 好的测试

---



# 好的测试

---

- 覆盖全的
- 确定的
- 不依赖特定环境的
- 运行快的

# Boring framework

---

- Service Stub (Record Support)
- In-memory Database
- Test Fixture
- Bored Infrastructures (Message Bus, Configurations, ...)

如何优雅地交付？



标准化

自动化

平台化

# 交付微服务

---

- 统一的开发框架
- 统一的工具库
- CI / CD
- 自动化部署：服务注册
- 私有云平台：容器、存储、MQ、LB 等资源
- 自动监控和报警

一些截图

## Apps / New

为规范 Service 相关管理，请大家创建新 Service 前需要 RFC 讨论通过，也请同时阅读 [服务设计的原则](#)。

New App

Owner Name \*

Name \*

Team \*

Arches



Description \*

Gitlab URL \*

RFC URL

 已阅读服务设计的原则 已通过 RFC 讨论

Submit

## Create Unit



**Name \***

将会被用作服务注册和发现，全局唯一且不能修改

**运行环境 \***

容器

支持物理机和容器

**类型**

✓ 未定义

Zone

HTTP

Wish

Thrift

Miller Worker

Miller Router

**端口号 \***

**Discovery Name**

Same as Unit Name by default...

服务注册发现的名字，默认和 Unit Name 相同，请勿随意修改

Save



## Apps / topstory2 / Resource

资料

成员

资源

部署

Units

LifeCycle

部署历史

操作记录

dependencies out of date

build passing

coverage 69%

## 容器 (28)

member\_worker 740-7fdf6f09

/data/apps/topstory2/bin/topstory\_miller worker -c  
/data/apps/topstory2/etc/miller/production.ini --tube=topstory2\_member

topic\_worker 740-7fdf6f09

/data/apps/topstory2/bin/topstory\_miller worker -c  
/data/apps/topstory2/etc/miller/production.ini --tube=topstory2\_topic

column\_worker 740-7fdf6f09

/data/apps/topstory2/bin/topstory\_miller worker -c  
/data/apps/topstory2/etc/miller/production.ini --tube=topstory2\_column

roundtable\_worker 740-7fdf6f09

/data/apps/topstory2/bin/topstory\_miller worker -c  
/data/apps/topstory2/etc/miller/production.ini --tube=topstory2\_roundtable

promotion\_worker 740-7fdf6f09

/data/apps/topstory2/bin/topstory\_miller worker -c  
/data/apps/topstory2/etc/miller/production.ini --  
tube=topstory2\_shuffle\_promotion\_content

restore\_worker 740-7fdf6f09

/data/apps/topstory2/bin/topstory\_miller worker -c  
/data/apps/topstory2/etc/miller/production.ini --tube=topstory2\_restore

roundtable\_router 740-7fdf6f09

## Kafka (5)

tj-offline:topstory2.log

tj-offline:topstory2.data

tj-offline:topstory2.log.offline

tj-offline:topstory2.data.filtered

tj-offline:topstory2.log.web

## Redis (11)

storage623 (high-available)  
qps: 0, 命中率: 0.0, 连接数: 2cluster718 (cluster)  
qps: 0, 命中率: 0.0, 连接数: 28cluster719 (cluster)  
qps: 0, 命中率: 0.0, 连接数: 61storage822 (redis)  
qps: None, 命中率: 0.0, 连接数: Nonecluster1146 (cluster)  
qps: 0, 命中率: 0.0, 连接数: 204



jenkins @jenkins commented 25 days ago

😞 Jenkins Build Failed

Results available at: [Jenkins](#)



彭晟杰 @sheng started a discussion on an outdated diff 25 days ago

taozle @taozle 25 days ago

Added 1 commit:

- [994ce5ad](#) - rename Censor to CensorStatus.



jenkins @jenkins commented 25 days ago

❤️ This branch passed all test, [click here](#) to deploy to unstable environment



jenkins @jenkins commented 25 days ago

✅ Jenkins Build Success

Results available at: [Jenkins](#)

彭晟杰 @sheng 23 days ago

Status changed to merged

# Apps / topstory2 / Deploy

- 资料
- 成员
- 资源
- 部署
- Units
- LifeCycle
- 部署历史
- 操作记录

## Candidates

机房设置 锁定部署

	Build	Version	Author	Build Time	State	
testing office	39020	752-34ce9d81	fanliwen	2016-12-03 00:17:53	B > T > O > C > C > TC > P	禁止部署 部署
	38999	751-f853fd9d	zheng	2016-12-02 19:16:15	B > T > O > C > C > TC > P	禁止部署 部署
	38927	750-fab0f5c1	huang	2016-12-02 16:03:00	B > T > O > C > C > TC > P	禁止部署 部署
	38914	749-5151b2a5	lyz	2016-12-02 15:34:58	B > T > O > C > C > TC > P	禁止部署 部署
	38911	748-f8490724	wyx	2016-12-02 15:29:13	B > T > O > C > C > TC > P	禁止部署 部署
	38903	747-b1defcd9	huang	2016-12-02 15:09:46	B > T > O > C > C > TC > P	禁止部署 部署
	38885	746-4d6e0342	huang	2016-12-02 14:14:56	B > T > O > C > C > TC > P	禁止部署 部署
	38881	745-b6756bbf	fanliwen	2016-12-02 14:09:51	B > T > O > C > C > TC > P	禁止部署 部署
	38856	744-a676171c	wvx	2016-12-02 13:29:01	B > T > O > C > C > TC > P	禁止部署 部署

dependencies out of date

build passing

coverage 69%

以及交付之后的错误、性能和稳定性

# 微服务的错误诊断

---

# 微服务的错误诊断

---

- 没有完整的错误栈：跨项目甚至跨语言
- 错误路径深

时间	Type	name	status	Duration & Attributes
- S22:00:39.045	zhihu-api	exploredcollectionshandler_get	zone.exc:ServerTimeout	<p>3024.944ms ca=10.3.2.68:0 &amp; http.code=500 &amp; http.path=/explore/collections &amp; http.uri=ht plore/collections &amp; sc=zhihu-api-explore &amp; span.server.error=zone.exc:ServerTimeout</p> <p>Traceback (most recent call last):</p> <p>File "/data/apps/zhihu-web/apps/api_v3/allin/exception_handler.py", line 36, in inner info = func(self, *args, **kwargs)</p> <p>File "/data/apps/zhihu-web/apps/api_v3/views/__init__.py", line 300, in get return self.real_get(*args, **kwargs)</p> <p>File "/data/apps/zhihu-web/apps/api_v3/views/misc.py", line 102, in real_get info = self.get_hot_collections(after_id, limit)</p> <p>File "/data/apps/zhihu-web/apps/api_v3/mixins/explore.py", line 268, in get_hot_collectior limit=limit)</p> <p>File "/data/apps/zhihu-web/apps/api_v3/rpcs/explore_rpc.py", line 61, in get_hot_favlists return zhexplore.Explore.get_hot_favlists(offset=offset, limit=limit)</p> <p>File "/data/apps/zhihu-web/eggs/Zone-2.0.0-py2.7.egg/zone/util.py", line 63, in __call__ return self._client.call(self._proto_name, self._method_name, **kwargs)</p> <p>File "/data/apps/zhihu-web/eggs/tracing-1.13.5-py2.7.egg/tracing/hooks/rpc_zone.py", lir ret = wrapped(*args, **kwargs)</p> <p>File "/data/apps/zhihu-web/eggs/Zone-2.0.0-py2.7.egg/zone/client.py", line 302, in call kwargs)</p> <p>File "/data/apps/zhihu-web/eggs/Zone-2.0.0-py2.7.egg/zone/client.py", line 457, in _call self._format_target(host, port), str(e)))</p> <p>ServerTimeout: Timeout while receiving from explore (10.3.2.71, 10080): timed out</p>
- S22:00:39.059	explore	explore_get_hot_favlists	zone.exc:ServerTimeout	<p>3001.881ms cc=zhihu-api-explore &amp; sc=explore &amp; span.client.error=zone.exc:ServerTimeou</p> <p>Client Send</p>
E22:00:39.059				
A22:00:39.061	redis	zrangebyscore		<p>0.524ms cc=explore &amp; redis.key=fav_list &amp; sa=hredis-explore-storage2471.tc.rack.zhihu.co</p>
E22:00:39.061				<p>Server Receive</p>

- S22:00:39.045	zhihu-api	exploredcollectionshandler_get	zone.exc:ServerTimeout	<pre> File "/data/apps/zhihu-web/apps/api_v3/admin/exception_handler.py", line 30, in inner     info = func(self, *args, **kwargs) File "/data/apps/zhihu-web/apps/api_v3/views/__init__.py", line 300, in get     return self.real_get(*args, **kwargs) File "/data/apps/zhihu-web/apps/api_v3/views/misc.py", line 102, in real_get     info = self.get_hot_collections(after_id, limit) File "/data/apps/zhihu-web/apps/api_v3/mixins/explore.py", line 268, in get_hot_collections     limit=limit) File "/data/apps/zhihu-web/apps/api_v3/rpcs/explore_rpc.py", line 61, in get_hot_favlists     return zhexplore.Explore.get_hot_favlists(offset=offset, limit=limit) File "/data/apps/zhihu-web/eggs/Zone-2.0.0-py2.7.egg/zone/util.py", line 63, in __call__     return self._client.call(self._proto_name, self._method_name, **kwargs) File "/data/apps/zhihu-web/eggs/tracing-1.13.5-py2.7.egg/tracing/hooks/rpc_zone.py", line 11, in __call__     ret = wrapped(*args, **kwargs) File "/data/apps/zhihu-web/eggs/Zone-2.0.0-py2.7.egg/zone/client.py", line 302, in call     kwargs) File "/data/apps/zhihu-web/eggs/Zone-2.0.0-py2.7.egg/zone/client.py", line 457, in _call     self._format_target(host, port), str(e))) ServerTimeout: Timeout while receiving from explore (10.3.2.71, 10080): timed out </pre>
- S22:00:39.059	explore	explore_get_hot_favlists	zone.exc:ServerTimeout	3001.881ms cc=zhihu-api-explore & sc=explore & span.client.error=zone.exc:ServerTimeout
E22:00:39.059				Client Send
A22:00:39.061	redis	zrangebyscore		0.524ms cc=explore & redis.key=fav_list & sa=hredis-explore-storage2471.tc.rack.zhihu.co
E22:00:39.061				Server Receive
E22:00:39.062				Server Send
E22:00:42.061				Client Receive
E22:00:42.070				Server Send



# 性能问题

---

RPC 不是再是简单的 Procedure Call

# 性能问题

---

- 网络、序列化开销
- 请求量放大严重 (N+1 Query)

# Leviathan Framework

---

# Leviathan Framework

---

- 自动批量调用
- 尽可能地并发
- Session 级别的缓存
- 面向对象
- Pythonic

# Leviathan Framework

---

```
question = Question(id=1)
question.answers[:10].include('author')
print question.answers[0].author.name
```

# 稳定性

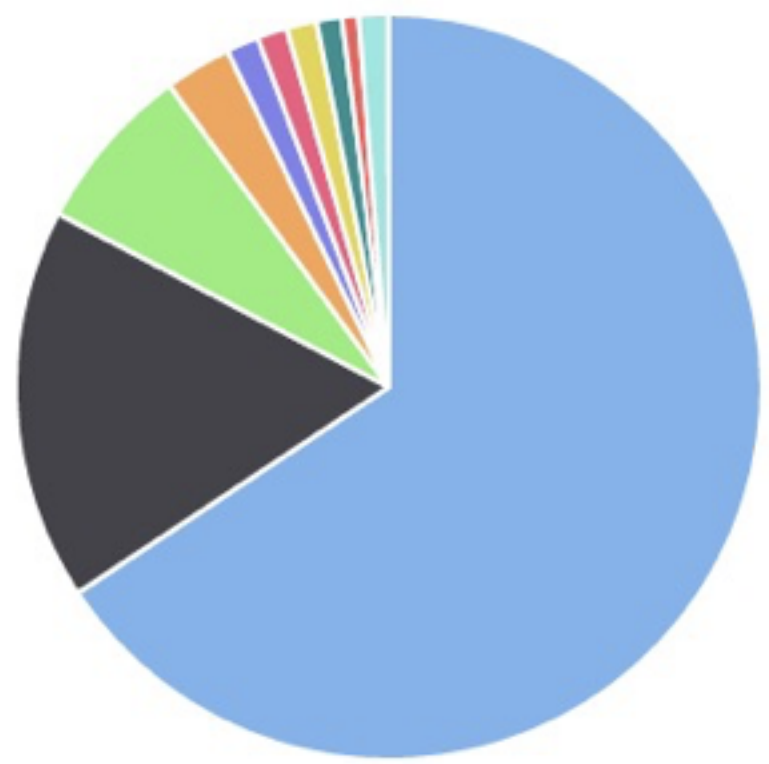
---

- 监控
- 限流
- 分组部署
- 降级与熔断
- Canary 发布、容量预估
- 自动伸缩
- SLA

另一些截图

# 应用接口分析: comment

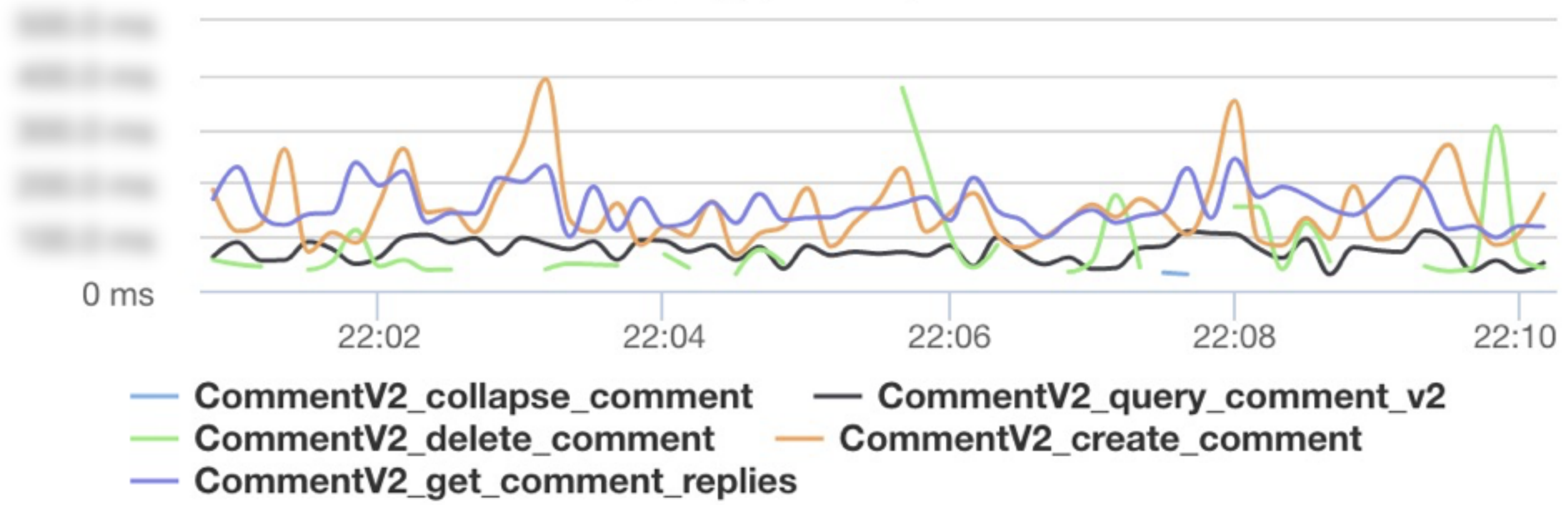
各接口请求量占比



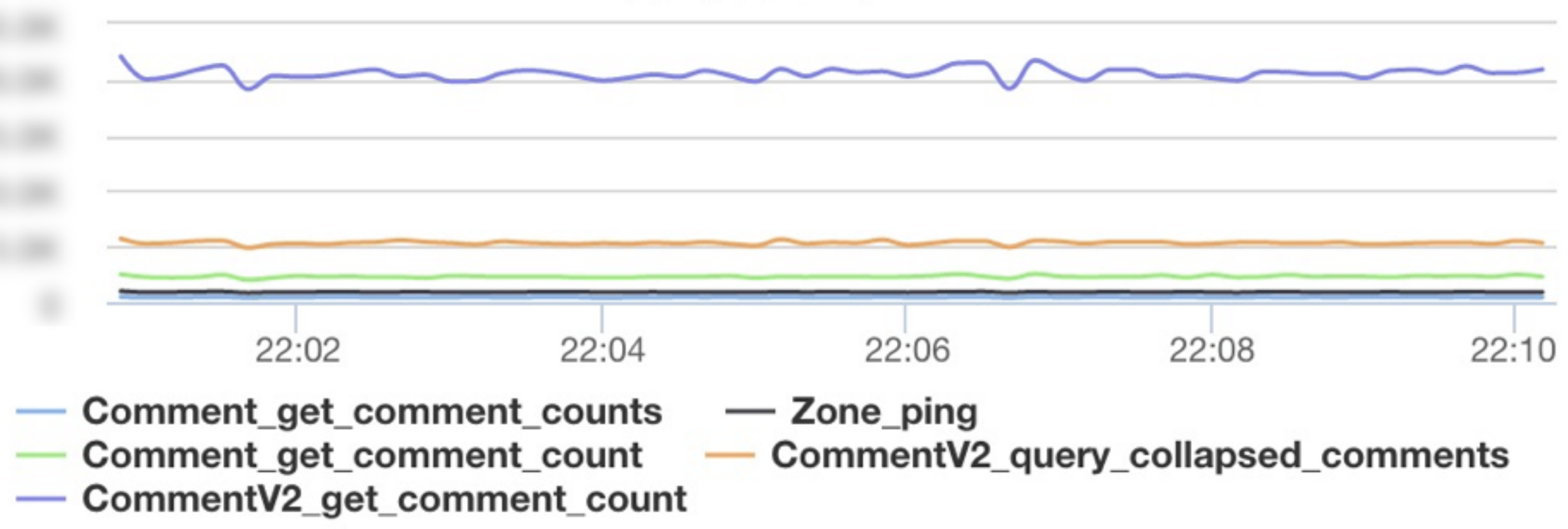
- CommentV2\_get\_comment\_count: 65.7%
- CommentV2\_query\_collapsed\_comments: 17.0%
- Comment\_get\_comment\_count: 7.3%
- Zone\_ping: 2.9%
- Comment\_get\_comment\_counts: 1.4%
- CommentV2\_get\_comment\_counts: 1.3%
- CommentV2\_get\_comment\_ids: 1.3%
- CommentV2\_query\_featured\_comments: 1.1%
- CommentV2\_get\_comments\_by\_ids\_v2: 0.7%
- 其他: 1.3%

服务端

响应时间 P95 Top5



每秒请求量 Top 5



每分钟错误数 Top5



Halo

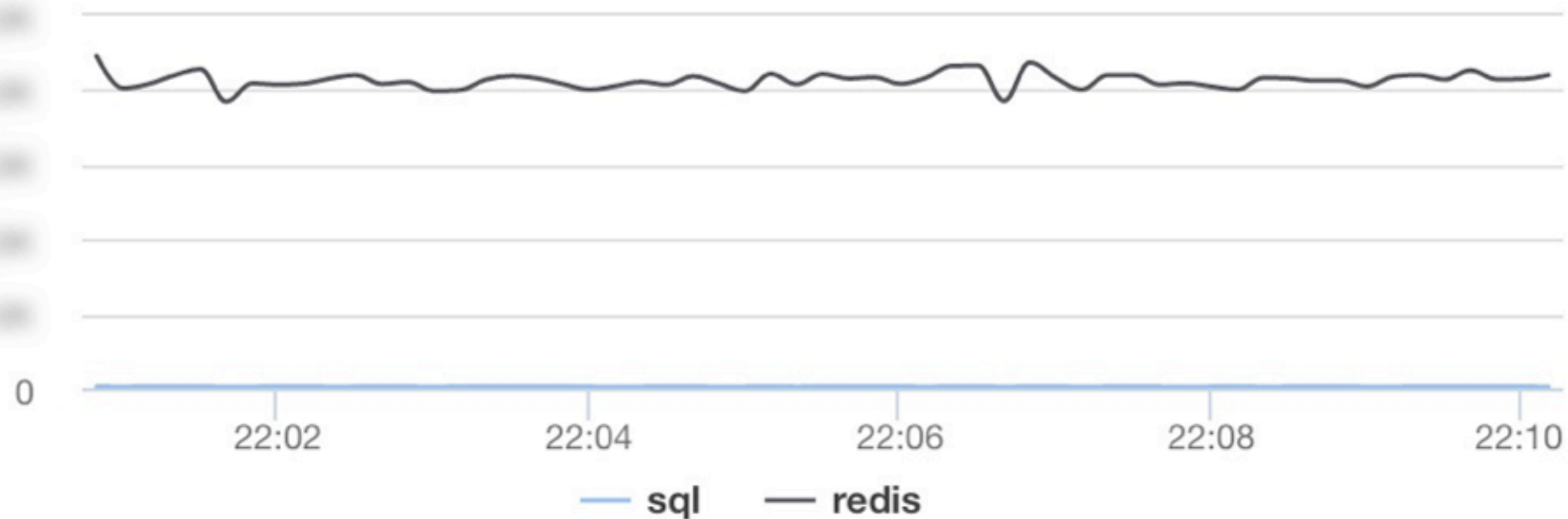
Halo

Halo

Halo



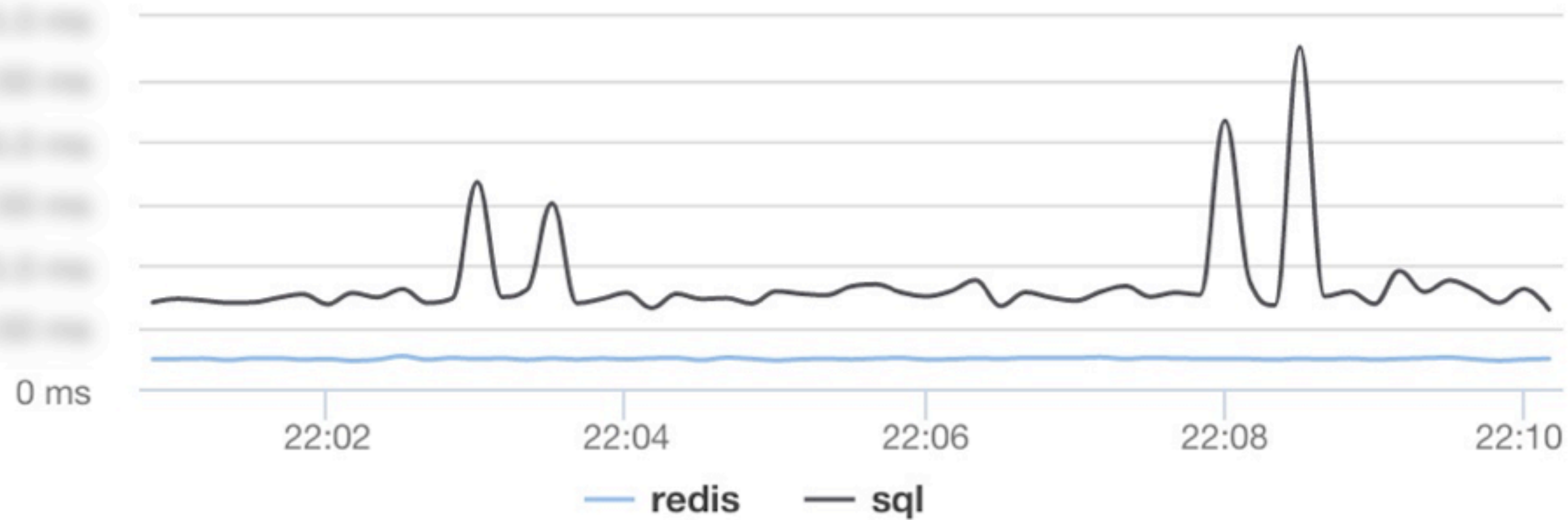
每秒请求量 Top5



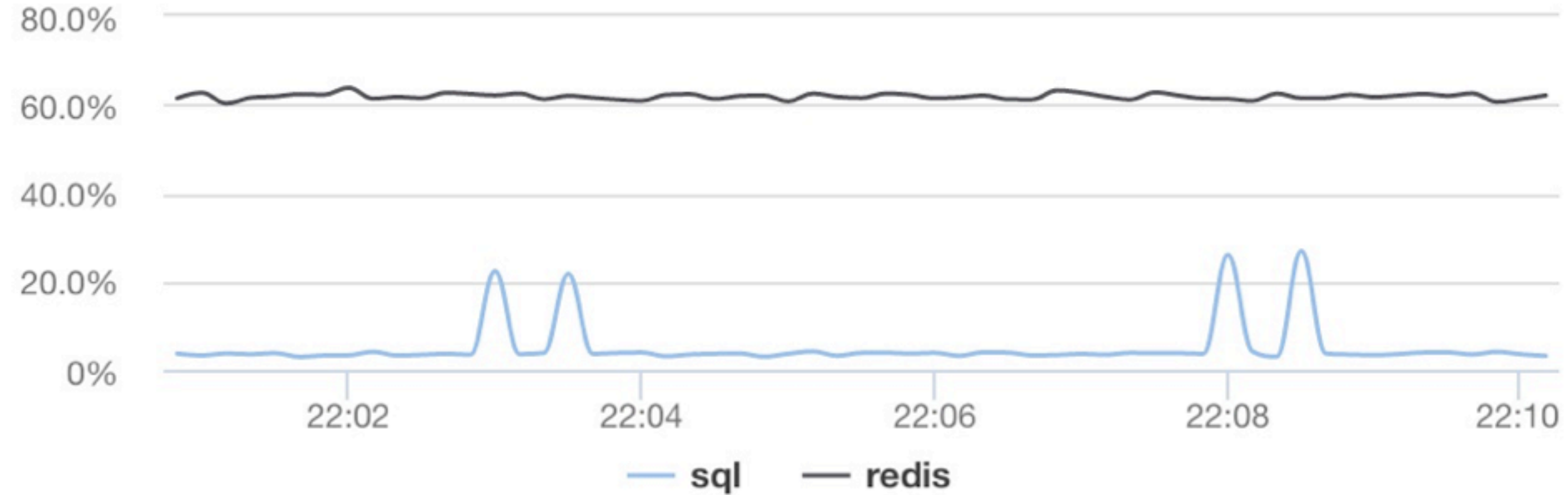
每分钟错误数 Top5



响应时间 P95 Top5



响应时间占比 Top5



被依赖分析

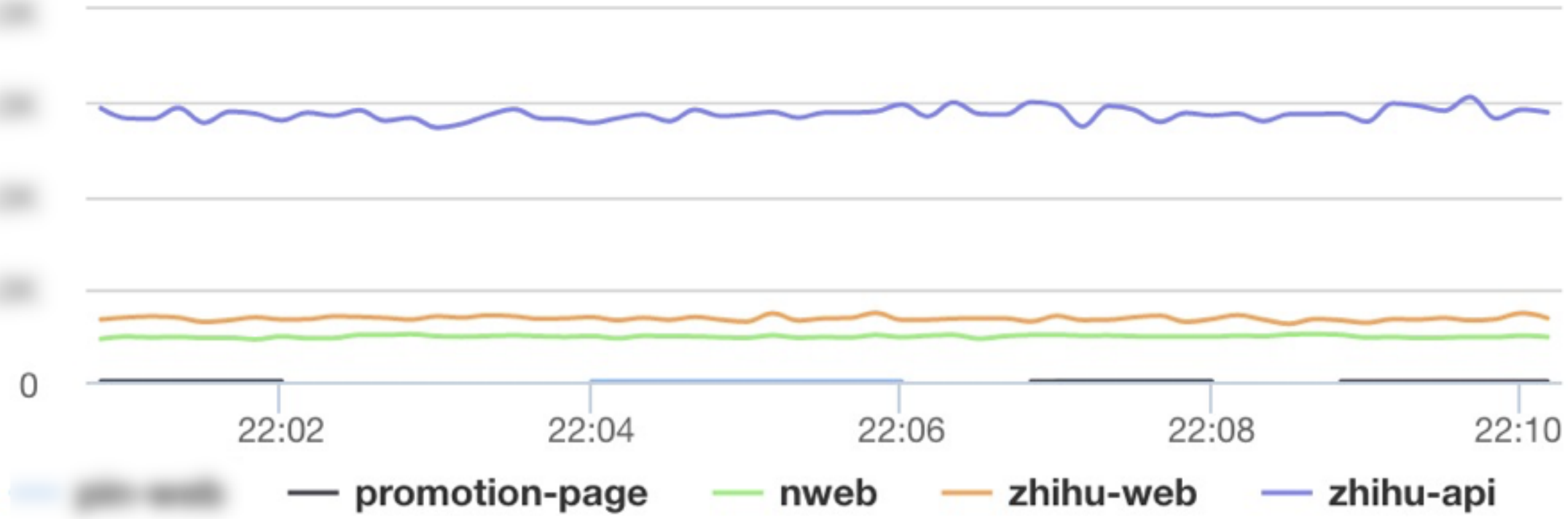
comment: CommentV2\_get\_comment\_count

使用 p999

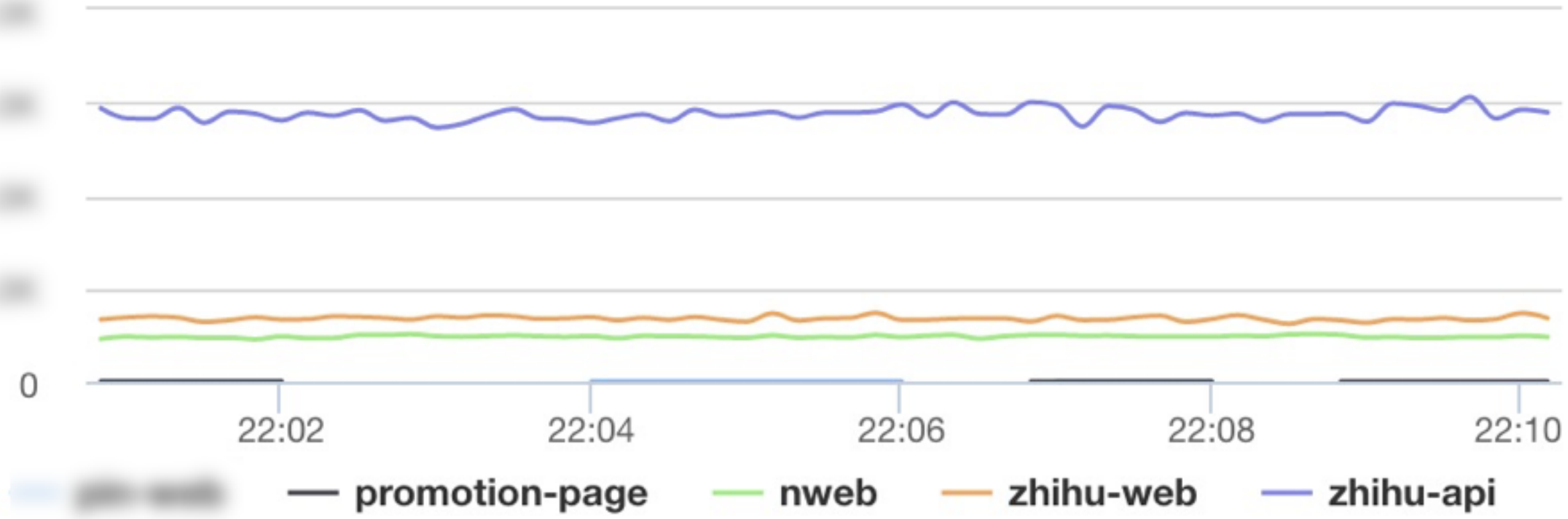
自动刷新

12月5日 22:00:45 - 12月5日 22:10:13

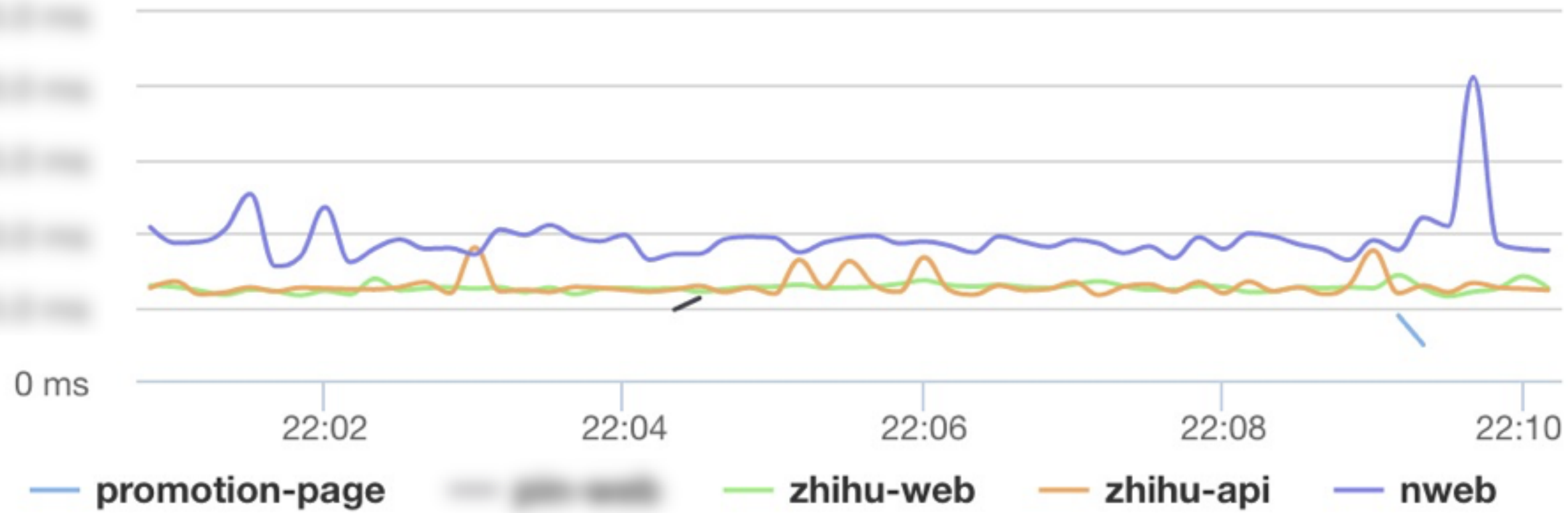
每秒请求量 Top5



每分钟错误数 Top5



响应时间 P95 Top5



Halo

Halo

Unit

comment



## Limited API

column-web:CommentV2.comment\_like  
adding (QPS: 100)

pin-

service:CommentV2.get\_comment\_count  
adding (QPS: 128)

Commit

Cancel

## column-web

CommentV2.query\_collapsed\_comments +

Comment.get\_user\_censored\_cids +

Comment.query\_comment +

CommentV2.get\_comment\_conversation +

CommentV2.delete\_comment +

CommentV2.get\_comment +

CommentV2.get\_comment\_ids +

CommentV2.get\_comments\_by\_ids +

CommentV2.query\_featured\_comments +

CommentV2.create\_comment +

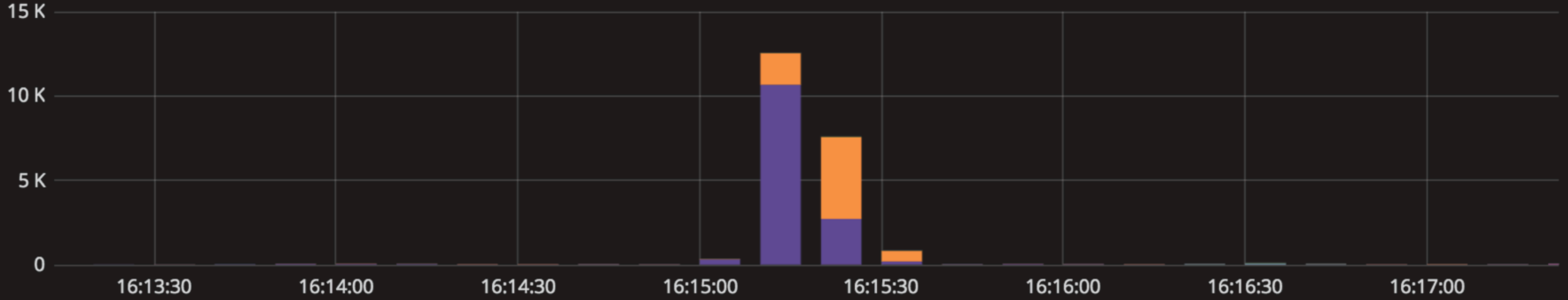
CommentV2.comment\_like +

## publications-api

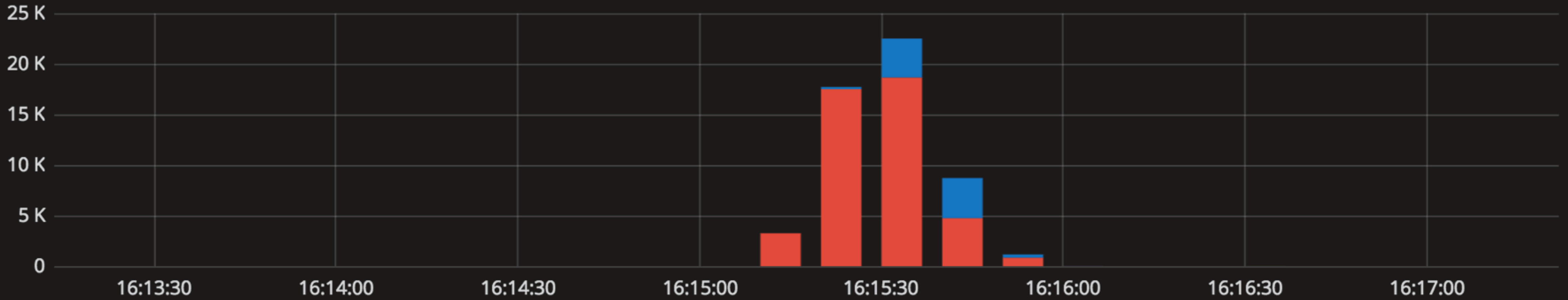
Comment.get\_comment\_count +

```
@Fail-safe(RatioFail, default=0,  
           exceptions=(TimeoutException,))  
def get_comments_count(question_id):  
    # Call RPC here...  
    pass
```

Failed call count



Masked call count



## Scale group

**Scale**  fixed  auto-scale

**Low \***

150

**High \***

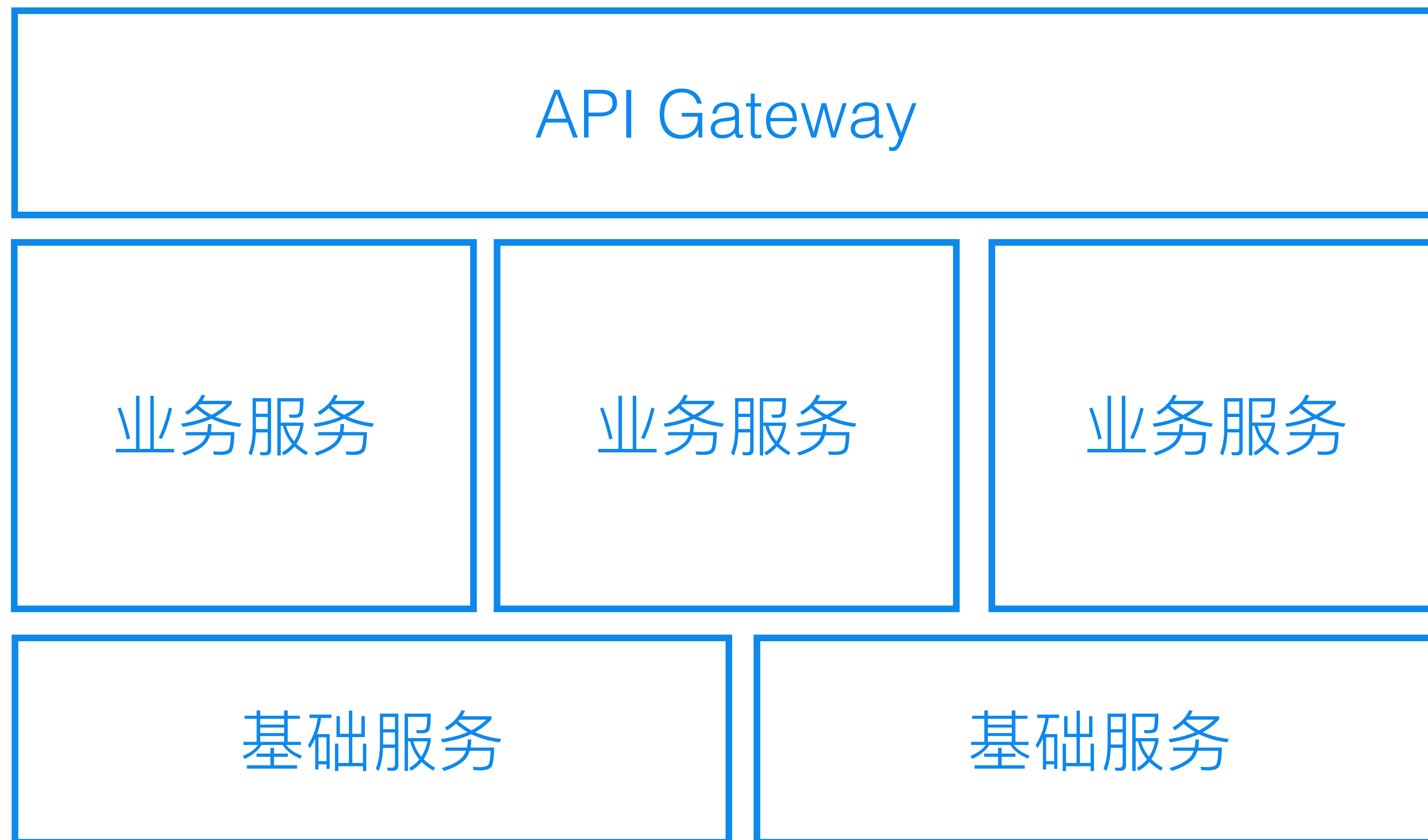
300

**Metric**

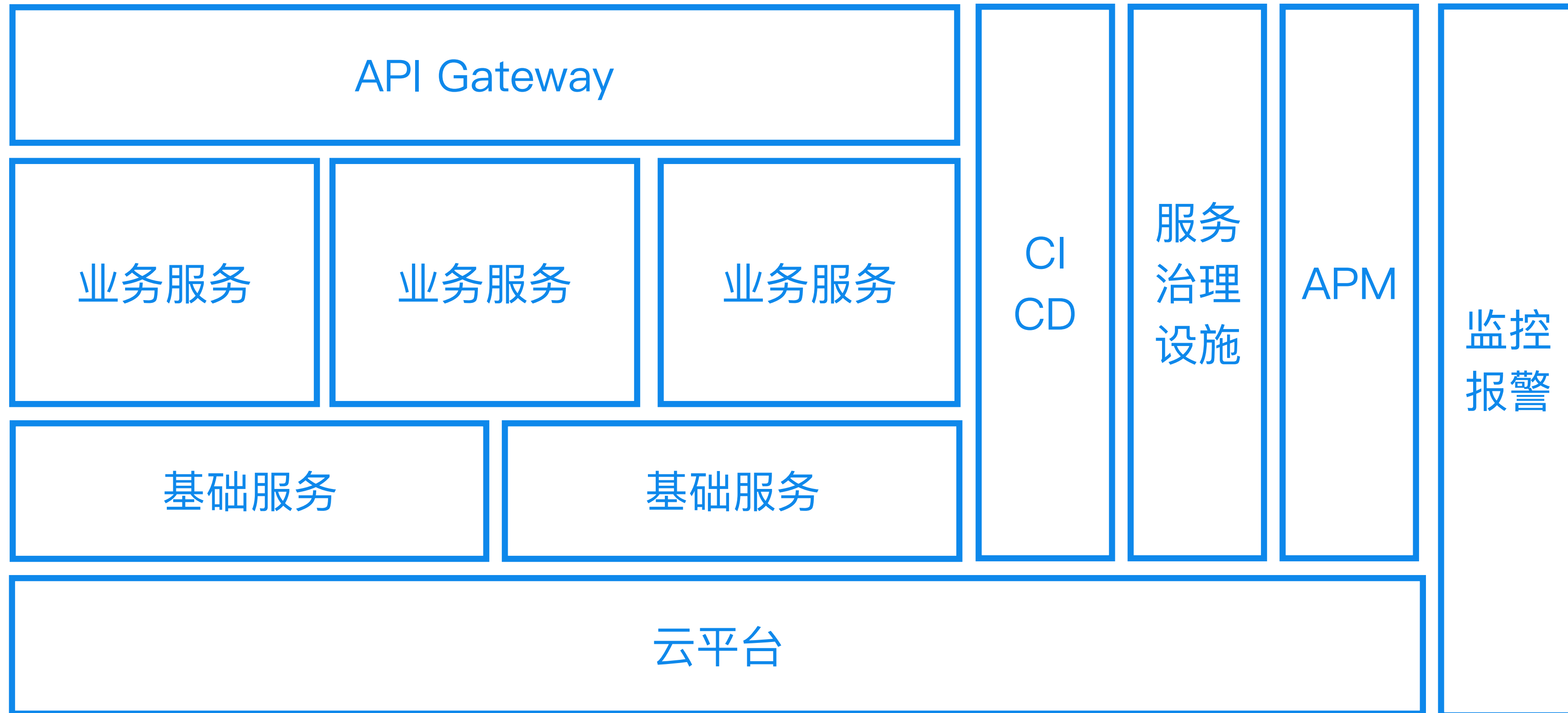
OK

# 微服务 The Big Picture

---



# 微服务 The Big Picture





这不仅仅是一个技术问题

# 团队与文化

---

DevOps

Feature Team

主人翁意识

契约精神

# 现状

---

# 现状

---

120

服务数量

70

部署频率  
(次/天)


10,000

容器数量



Thank You!

 @psjcarlos

 @psjay



Hacker's log

知乎技术日志

关注专栏

