蘑菇街稳定性实践

吴一帆（无相）

大纲

大促保障

流量评估

依赖梳理

预案评审

工具和平台

全链路监控

全链路压测

限流降级

挑战与规划

数据量

效率&成本

规范保障

一片迷惘

- X 怎么保障不出事故
- X 系统峰值如何评估
- X 依赖关系怎么梳理
- X 系统如何压测
- X 应用如何扩容
- X 出了事故如何应对

大促稳定性保障流程

下单峰值评估

已知

业务目标
pv、uv、gmv
转化率、客单价
品单价

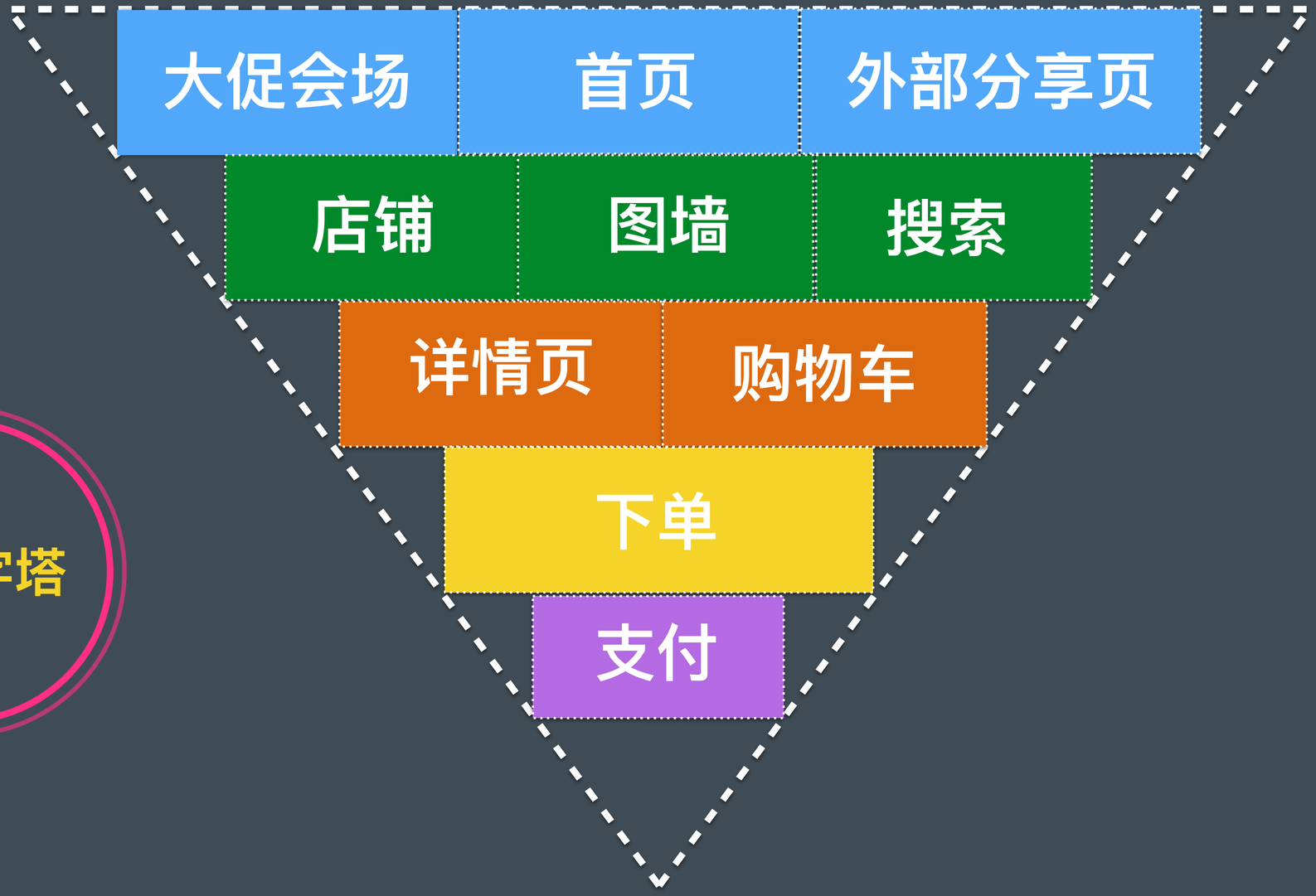
求解

交易系统
下单峰值

推导方

$$\text{预估下单峰值} = \frac{\text{预估大促GMV}}{\text{历史大促GMV}} * \frac{\text{历史客单价}}{\text{预估客单价}} * \text{历史大促峰值}$$

系统峰值评估

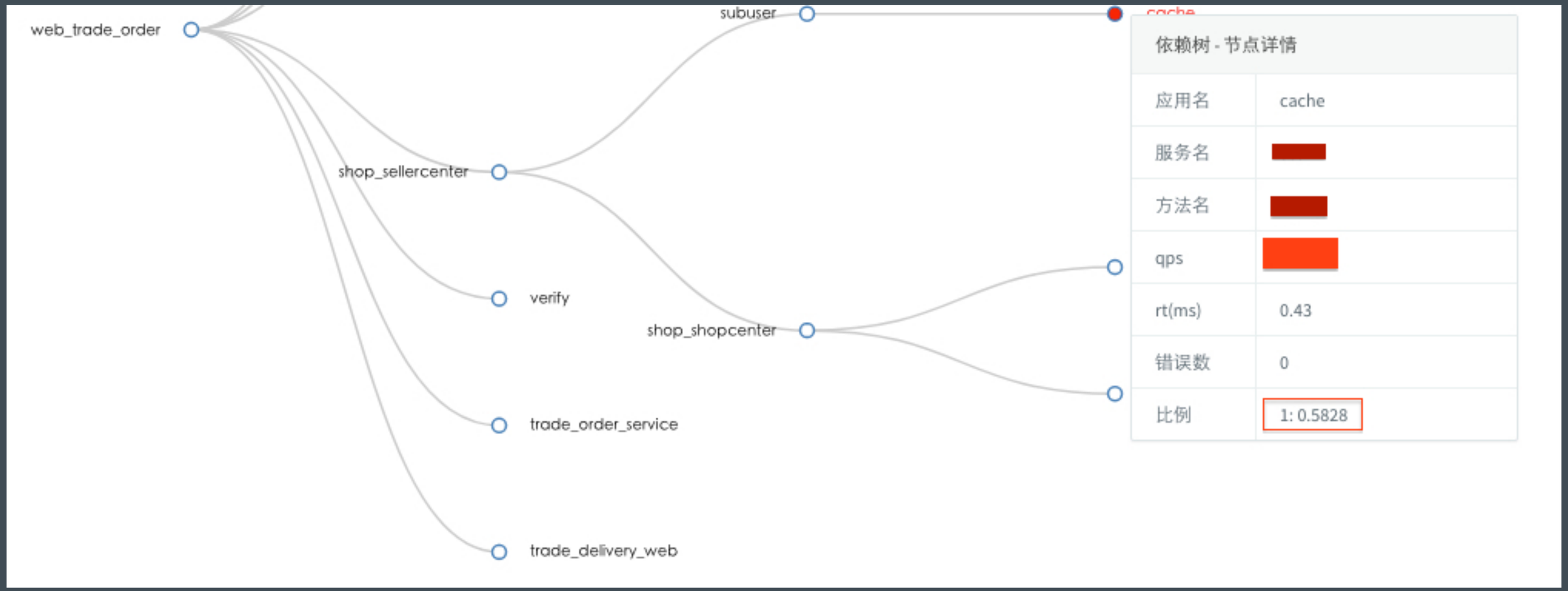


倒金字塔

依赖梳理

以前	• 开发人员查看代码 • 容易遗漏 • 调用比例无法精确计算
现在	• 利用全链路监控系统 • 自动计算调用比例 • 详细到接口级别

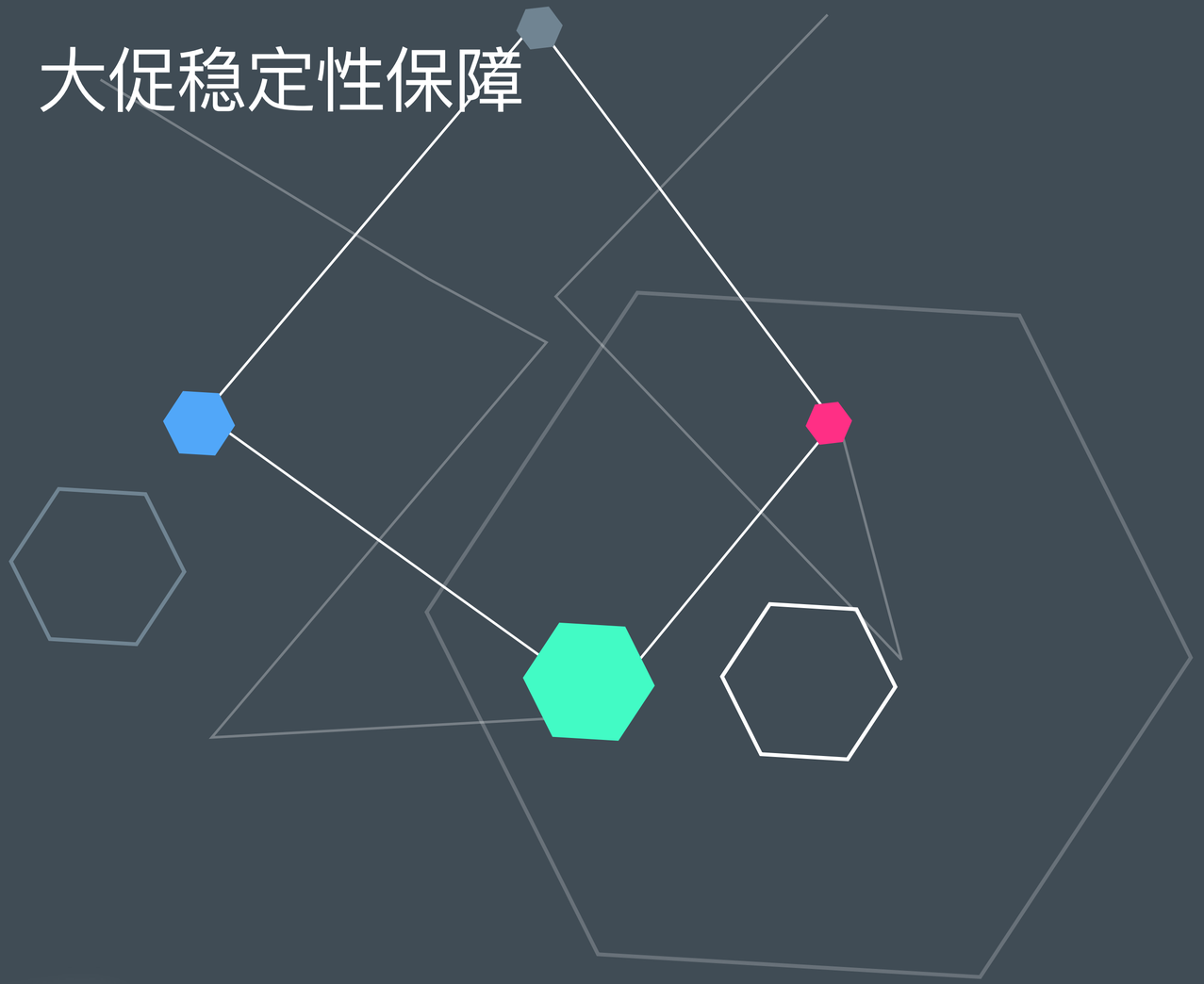
依赖梳理 - 全站应用拓扑


web_trade_order
subuser
cache
shop_sellercenter
verify
shop_shopcenter
trade_order_service
trade_delivery_web
依赖树 - 节点详情
应用名
cache
服务名
方法名
qps
rt(ms)
0.43
错误数
0
比例
1: 0.5828

压力测试

单链路	• 业务方自行组织 • 100%流量，验证限流、预案 • 局部集中
全链路	• 压测数据要尽量真实 • 压测流量识别 • 提高人效、降低成本

三 大促稳定性保障



系统扩容

a 单机压测系统提供水位报告

b 由数据说话、拒绝拍脑袋

c 有效减少了机器、降低了成本

大促稳定性保障经验总结

1

关注玩法 &
架构上的新变动

2

限流降级要
测试充分

3

依赖梳理要细致
标记强弱依赖

4

压测要充分
数据要真实

5

预案要全面
手册要细致

全链路监控lurker第一版

PHP
部分服务化
CURL

xhprof
PHP扩

zend引擎中替换
原有函数执行的
指针，并指向自
己的hook函数

修改curl以
及php-curl
的实现

全链路监控原理

源于google
Dapper论文

spanId
发生顺序
嵌套层次关系

trace context
从前端透传到后端

埋在公共组件
客户端、中间件中

traceId
全局唯一
业务含义

在nginx端
产生trace context

tomcat等容器
通过servlet filter实现

全链路监控系统

链路详情

traceId 006440D0F884110A59104409000B7407 查询

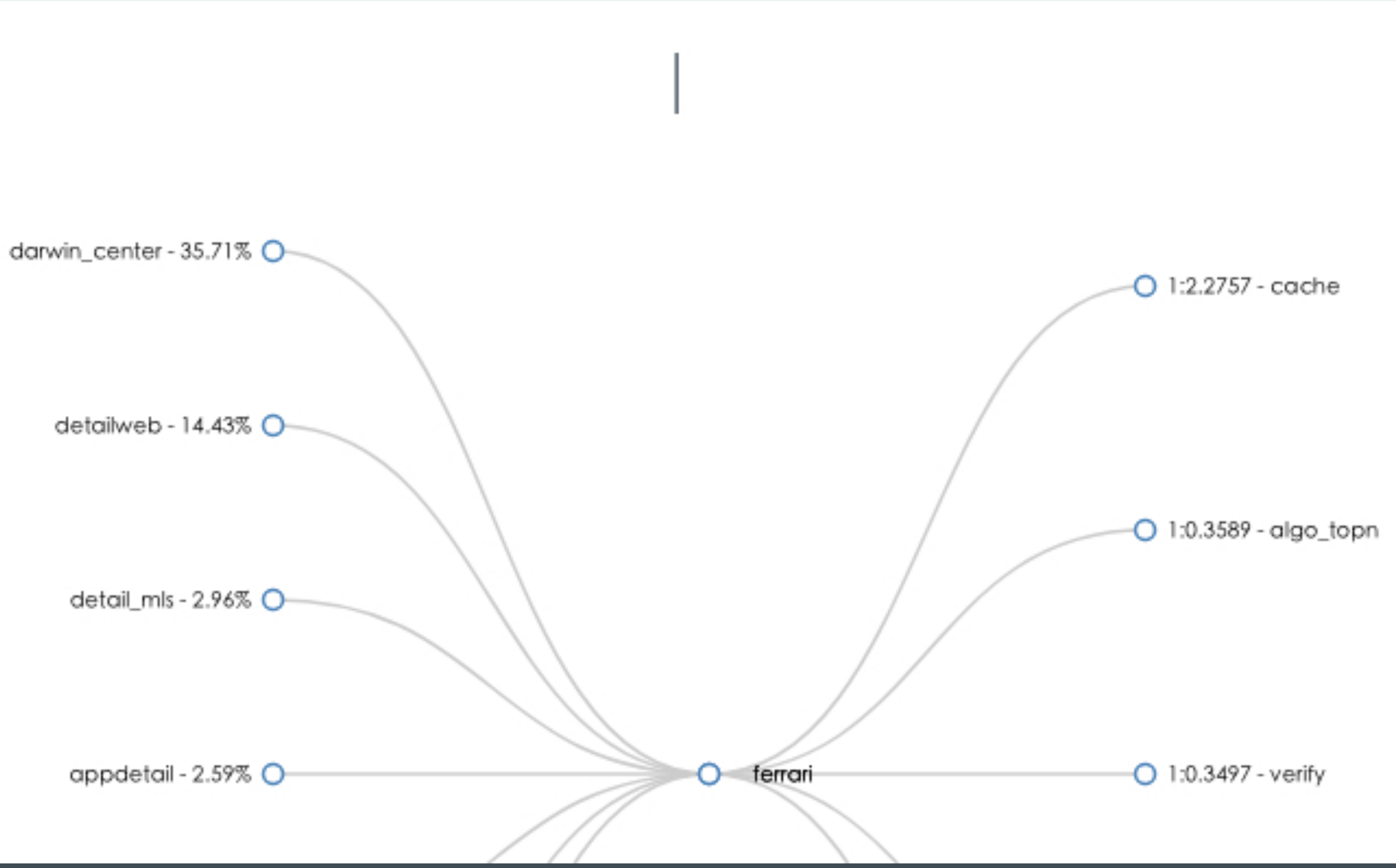
查询结果 入口URL: www.mogujie.com [redacted] 时间: 2017-05-08 18:10:17 采样率: 32

spanId	服务名	方法名	type	IP	耗时(ms)	Timeline	操作
⊟ 1	nmapi_book_welcome_mgjhd_v1	index	PHP		552.05		详情
1.1	PHP-Cache	GET	Cache		1.05		详情
1.2	SessionService	checkSign	Tesla		1.44		详情
1.3	PHP-Cache	GET	Cache		1.10		详情
⊟ 1.4	TopN	rank	HTTP		515.78		详情
1.4.1	solr	q rank	HTTP		513.00		详情
⊟ 1.5	LikesBizService	getMutiLikesCounter	Tesla		5.21		详情
⊟ 1.5.1	RelationReadFacade	getCounterMuliter	Tesla		4.00		详情
1.5.1.1	CounterFacade	mget	Tesla		2.00		详情

应用信息统计


应用统计
时间
2017-05-08 18:48
~
2017-05-08 18:48
应用名称
ferrari
查询
概要
入口来源应用
入口来源URL
我的URL
我的服务
直接来源应用
直接去向应用
访问Mogucache
访问Raptor
访问Corgi
直接来源应用
展示QPS
最近30分钟
趋势图
更新
当前展示数: 9, 总数: 9
darwin_center
查看详情
时间 qps 环比 耗时 错
18:48 ↑7.73% 5.38 0
18:47 ↓-6.51% 5.44 0
18:46 ↑3.98% 5.42 0
18:45 ↓-1.69% 5.44 0
18:44 ↓-0.49% 5.34 0
qps rt err
detailweb
查看详情
时间 qps 环比 耗时 错
18:48 ↓-5.56% 2.15 0
18:47 ↓-10% 1.81 0
18:46 ↑11.11% 2.23 0
18:45 ↑0% 1.97 0
18:44 ↓-2.7% 2 0
qps rt err
appdetail
查看详情
时间 qps 环比 耗时 错
18:48 ↓-14.29% 1.83 0
18:47 ↑16.67% 2.29 0
18:46 ↓-33.33% 2.5 0
18:45 ↑125% 2.22 0
18:44 ↑0% 2 0
qps rt err

应用级依赖

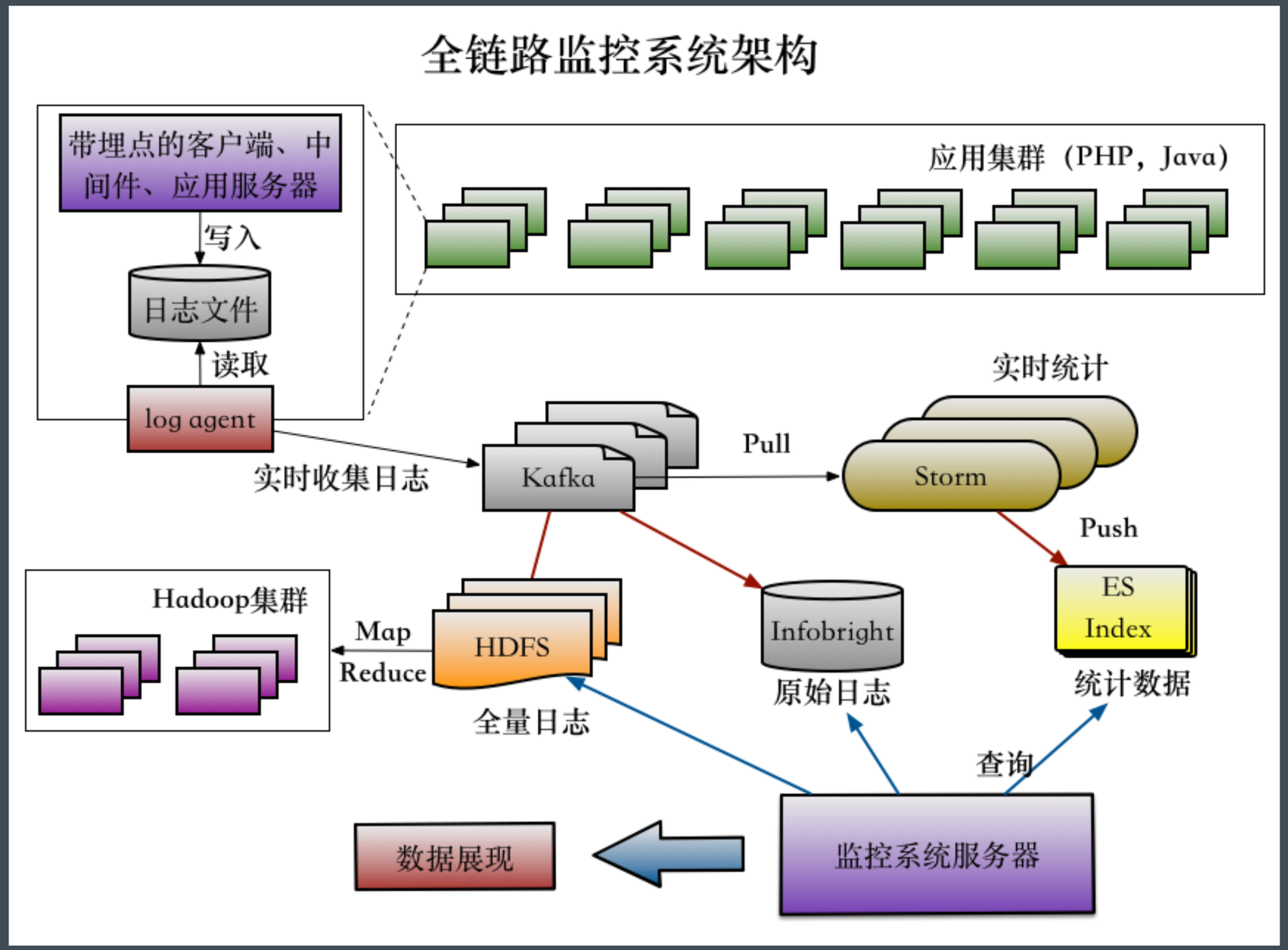

darwin_center - 35.71%
detailweb - 14.43%
detail_mls - 2.96%
appdetail - 2.59%
ferrari
1:2.2757 - cache
1:0.3589 - algo_topn
1:0.3497 - verify

多维查询


起始时间
2017-05-08 18:10
~
2017-05-08 18:12
IP
只支持单个IP
应用名
例如：web_mtalk_android
服务名
例如：nmapi_im_emotion_mgj_v1
方法名
例如：getEmotion
业务响应码
例如:2
耗时(ms)
500
~
300
入口url
例如：www.mogujie.com/book/search
设备ID
例如：A10000490EA6C9
用户ID
例如：10001
HTTP响应码
例如:500
选择链路范围
入口链路
查询

全链路监控系统

- 多维查询
- 链路展示
- 依赖梳理
- 应用信息统计



一些问题

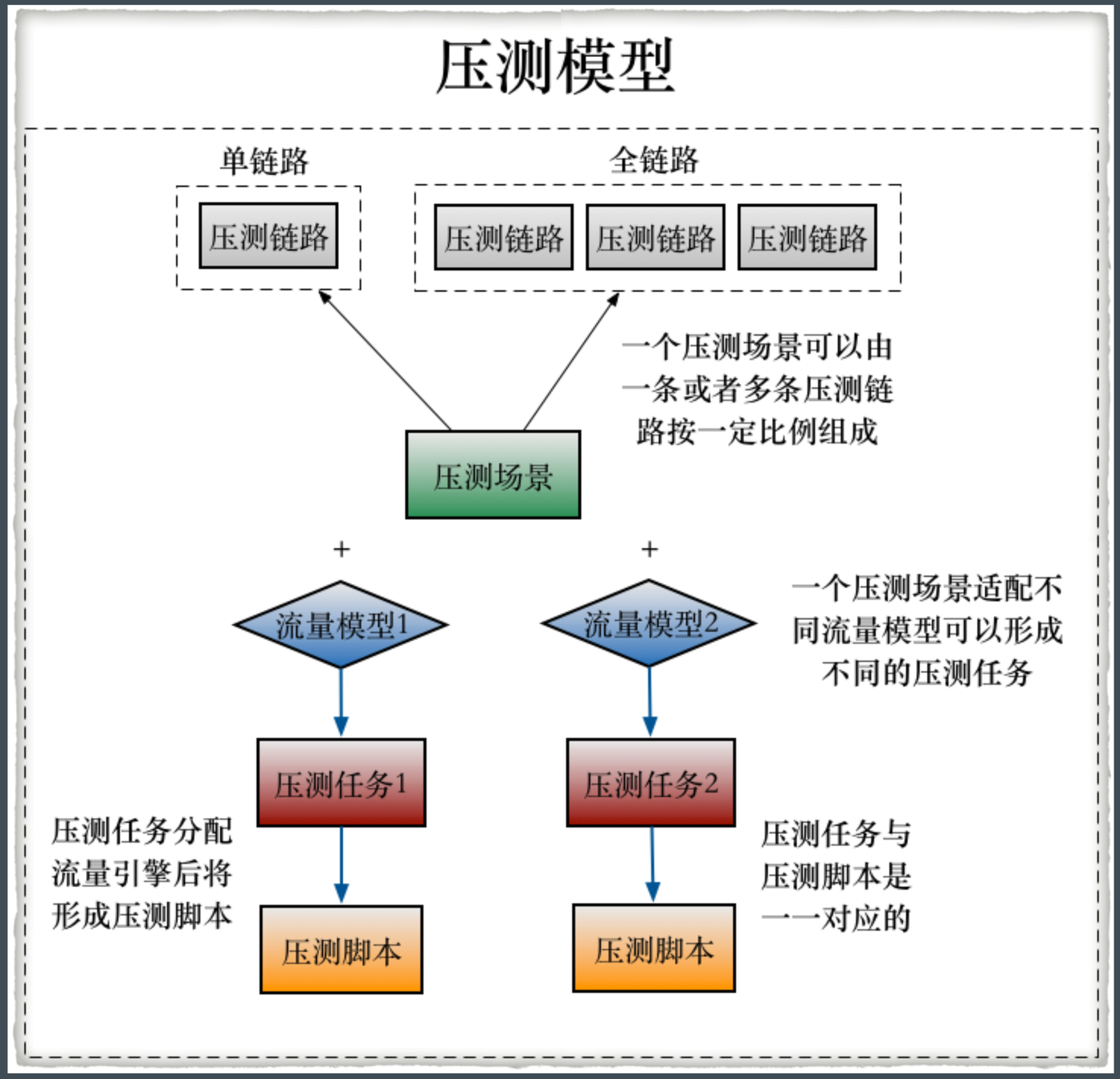
- 数据存储
  hbase、ES、infobright、bigstore
- 跨线程传递trace context
- 前端应用接入不全，导致后端应用qps不准确
- 周期性任务，链路过长，展示有问题

全链路压测

- 对线上真实环境进行大流量演练
- 验证链路在超大流量系统容量和资源分配是否合理
- 找出链路中可能存在的瓶颈
- 验证网络设备、集群容量和预案、限流策略

全链路压测系统

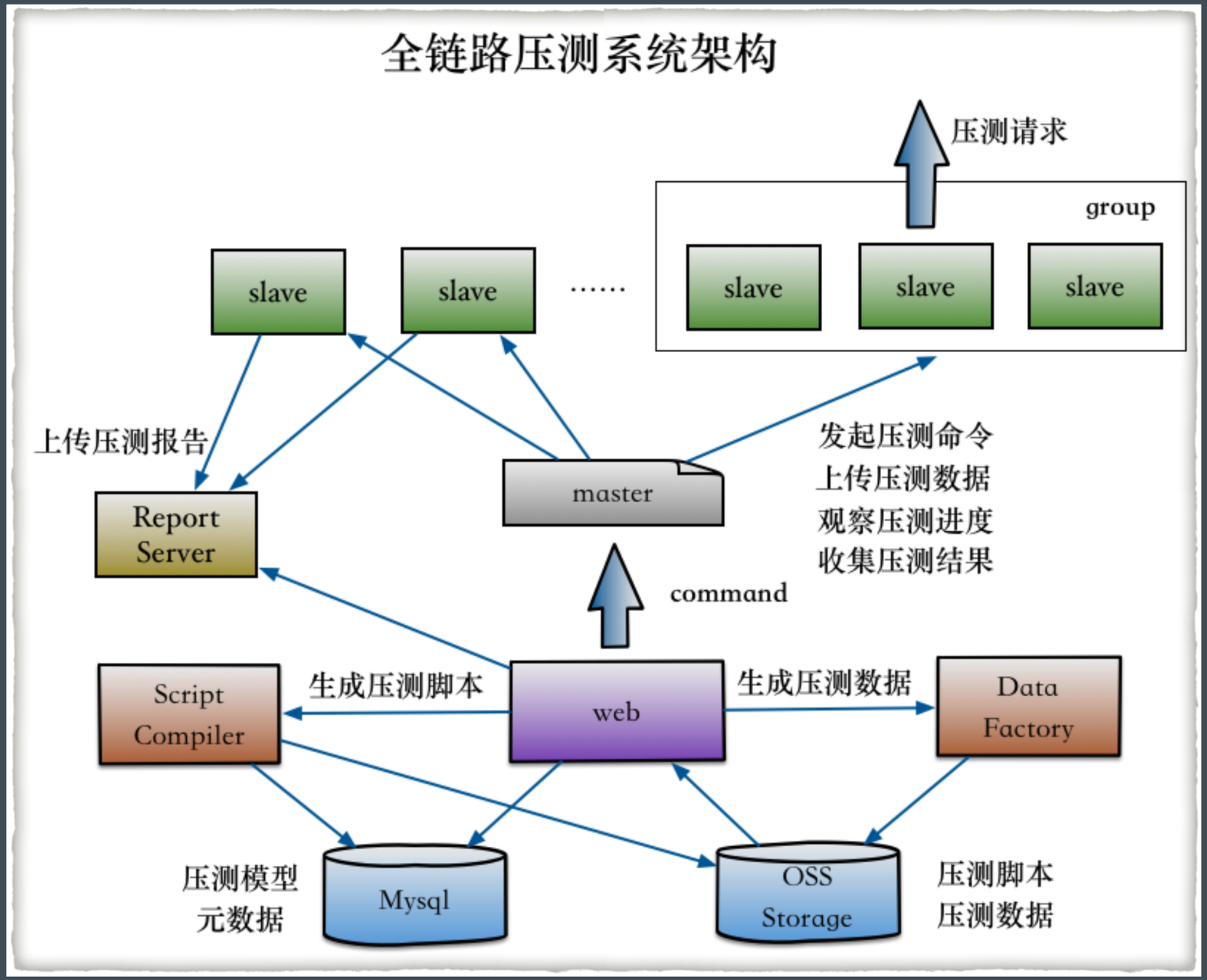

压测链路
压测脚本
流量模型
压测任务
压测场景


压测模型
单链路
全链路
压测链路
压测链路
压测链路
压测链路
一个压测场景可以由
一条或者多条压测链
路按一定比例组成
压测场景
+
+
流量模型1
流量模型2
一个压测场景适配不
同流量模型可以形成
不同的压测任务
压测任务1
压测任务2
压测任务分配
流量引擎后将
形成压测脚本
压测任务与
压测脚本是
一一对应的
压测脚本
压测脚本

全链路压测系统

系统架构

- gatling
- jenkins
- 避免单点
- 脚本引擎
- 支持多种协议：http，mwp，mwcs，tesla，IM



全链路压测系统

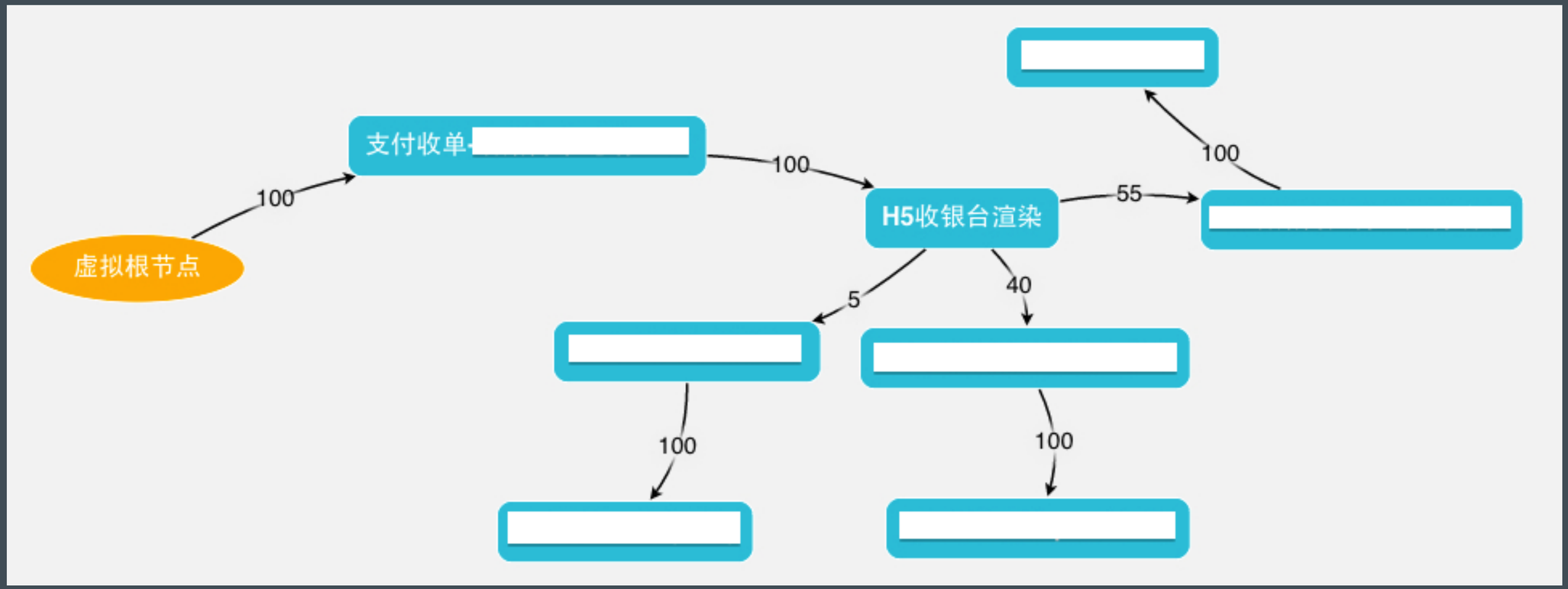
压测场景编辑

- 流量按比例分配
- 场景可以嵌套
- 场景嵌套深度限制

全链路压测系统

压测场景编辑

- 链路可串联
- 链路可复用



全链路压测系统

压测任务执行

- 压测脚本确认
- 压测数据确认
- 压测进度监控

任务id: 1097 , 历史id: 29887 , rps: 303 , rt: 90 , 机器数量: 1 , 持续时间 180 s, 状态: 中止

脚本确认 | 数据确认 | 状态监控&结果

```
package com.mogujie.stress.test

import io.gatling.core.Predef._
import io.gatling.http.Predef._
import io.gatling.http.util.MoguPredef._
import io.gatling.writer.DistributedReportConf
import java.util.concurrent.TimeUnit

class AutoSimulation extends Simulation {

val slaveNum = 1

/*************** http request definition ***************/

val mwpVal_1 = mwp("lifestyle详情dsl接口")
.setIP('[redacted]')
.setAPI('[redacted]")
.setVersion("1")
.setMethod("GET")
.setInclass("java.util.Map")
.setInvalue("{\[redacted]ion
let.1\": {\"typ[redacted]
\"},\"mwp.timel[redacted]":
 ${type},\"iid\": \"${iid}\"}")
.setUid("${uid}")
```

全链路压测系统

- 编译后JVM对方法65535大小的限制
- 避免系统单点的情况出现
- 无法加载太大的压测数据文件

一些问题

一些经验

- 和lurker结合，借trace context由前向后传递，判断是否压测请求
- 和raptor结合，判断是否压测请求，决定是否转向测试库

限流系统

nginx限流

lua实现

cpu限流

qps限流

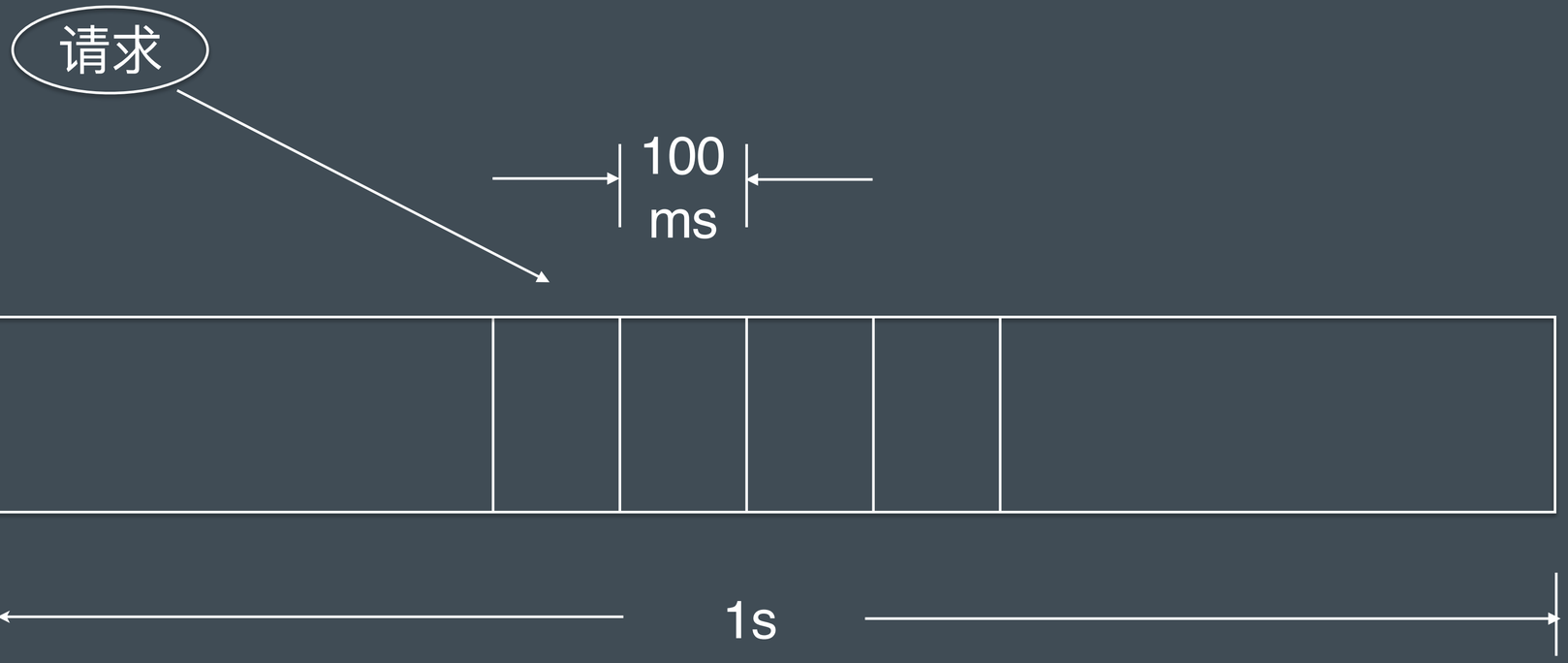
java端限流

并发限流

来源限流

去向限流

qps限流算法



三 思考和规划

思考和规划

- 稳定性工具&平台从0到1
- 贴近业务，从60分到90分
- 大促稳定性工具化&流程化
- 日常稳定性规范和流程的建设
- 稳定性工具横向打通
- 稳定性工具和公司其他工具横向打通
- 性能优化
- case by case，性能优化指导和规范

Thank you