饿了么性能测试平台实践与原理

吴骏龙 饿了么测试基础设施部

个人与团队介绍

个人

吴骏龙

曾就职于eBay从事测试基础设施的研发和创新工作多年，2017年3月加入饿了么，负责测试基础设施的研发以及全链路压测等工作。

团队

饿了么测试基础设施部

承担饿了么全链路压测、多活验证及相关自动化平台研发等重要工作。我们的目标是推动测试自动化进程，推行先进的测试理念，惠及整个公司技术团队。

基础工具

- 容易上手，学习曲线平缓
- 开源，免费
- 生态圈完善，社区支持强
- 饿了么技术团队广泛使用

JMETER实践中的痛点

信息共享难

测试脚本、测试数据、测试结果等

配置工作复杂

脚本上传，数据文件切分，可能还要涉及多台压测机

测试结果不直观

机械信息较多，缺乏图形化展示手段

不安全

对测试过程基本不做监管

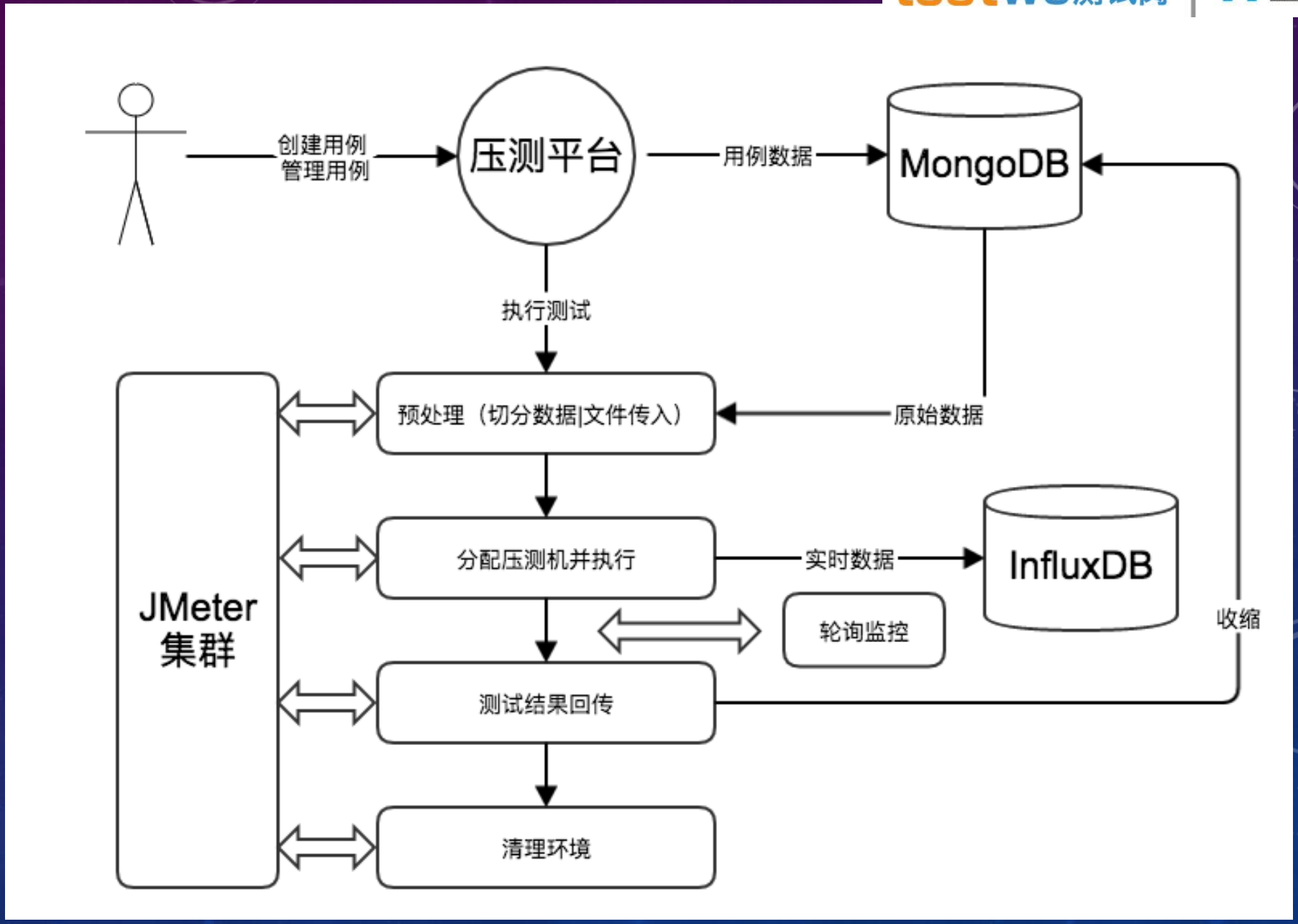
我们需要的是

能快速上手的 信息互通的 安全的 一站式平台

功能需求

- 用例管理：用户建立测试用例，上传资源文件，系统进行分类管理。
- 压测执行：一键触发，可指定各项参数，自动切分数据，分布式执行。
- 实时结果(热数据)：响应时间、吞吐量、错误率等数据以图表形式实时显示。
- 测试结果(冷数据)：QPS，90线等数据以图表形式在测试结束后显示。
- 测试机集群监控：监控测试集群的使用状态，提示用户可用的测试机。
- 安全保障：平台应该对用户操作进行适当限制，并能自我应对一些异常情况。

总体架构


创建用例
管理用例
压测平台
用例数据
MongoDB
执行测试
预处理（切分数据|文件传入）
原始数据
分配压测机并执行
实时数据
InfluxDB
JMeter
集群
轮询监控
测试结果回传
收缩
清理环境

分布式测试

JMeter分布式测试的缺点：

●JMeter的分布式测试执行和单机执行方式的差异较大，这会导致平台架构不必要的复杂度，实际用户只感知测试机的数量区别。

●JMeter分布式执行的方式，master机通常不参与测试，而是收集slave信息，但这会造成一定程度上的资源浪费。

●Jmeter分布式执行时，slave不间断传输数据给master，会一定程度上影响性能（尤其是带宽）。

压测平台的分布式实现

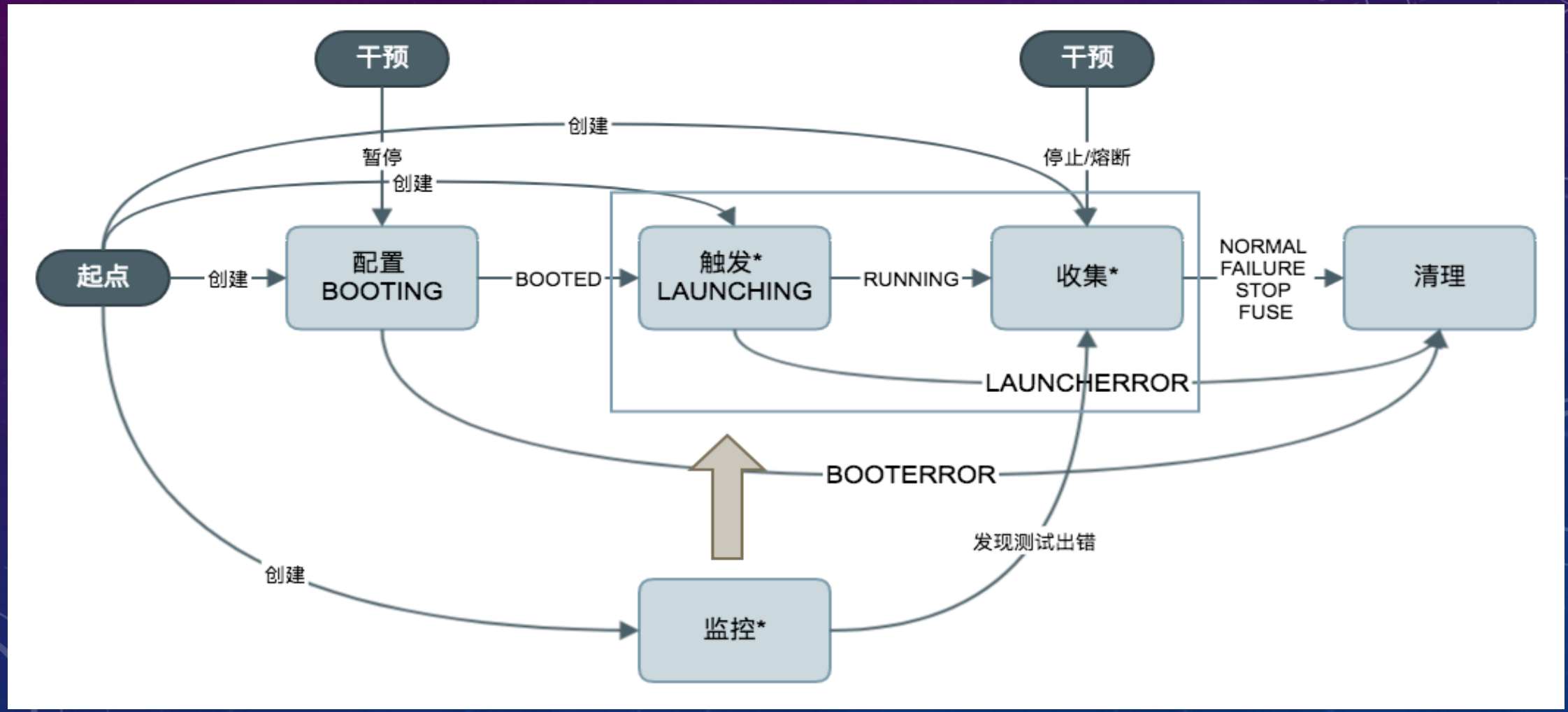
- 服务器统一调度测试过程
- 冷热数据分离

功能	JMeter实现	压测平台实现
脚本分发	master进行分发	服务器统一分发，可预配置
数据文件分发	用户自行上传至每台压测机	服务器统一分发，可根据压测机数量切分
结果回传	各slave回传至master	冷热分离回传至服务器，统一聚合入库
节点失败重试	统一配置重试	自动重试 + 人工重试

测试状态流转

配置 -> 触发 -> 运行 -> 结果收集 -> 清理

测试状态流转


干预
干预
创建
暂停
停止/熔断
创建
起点
创建
配置
BOOTING
BOOTED
触发*
LAUNCHING
RUNNING
收集*
NORMAL
FAILURE
STOP
FUSE
清理
LAUNCHERROR
BOOTERROR
发现测试出错
创建
监控*

实时数据

JMeter backend listener – UDP to InfluxDB (Jmeter 3.3 Core Improvement)

Backend Listener

Name: Backend Listener

Comments:

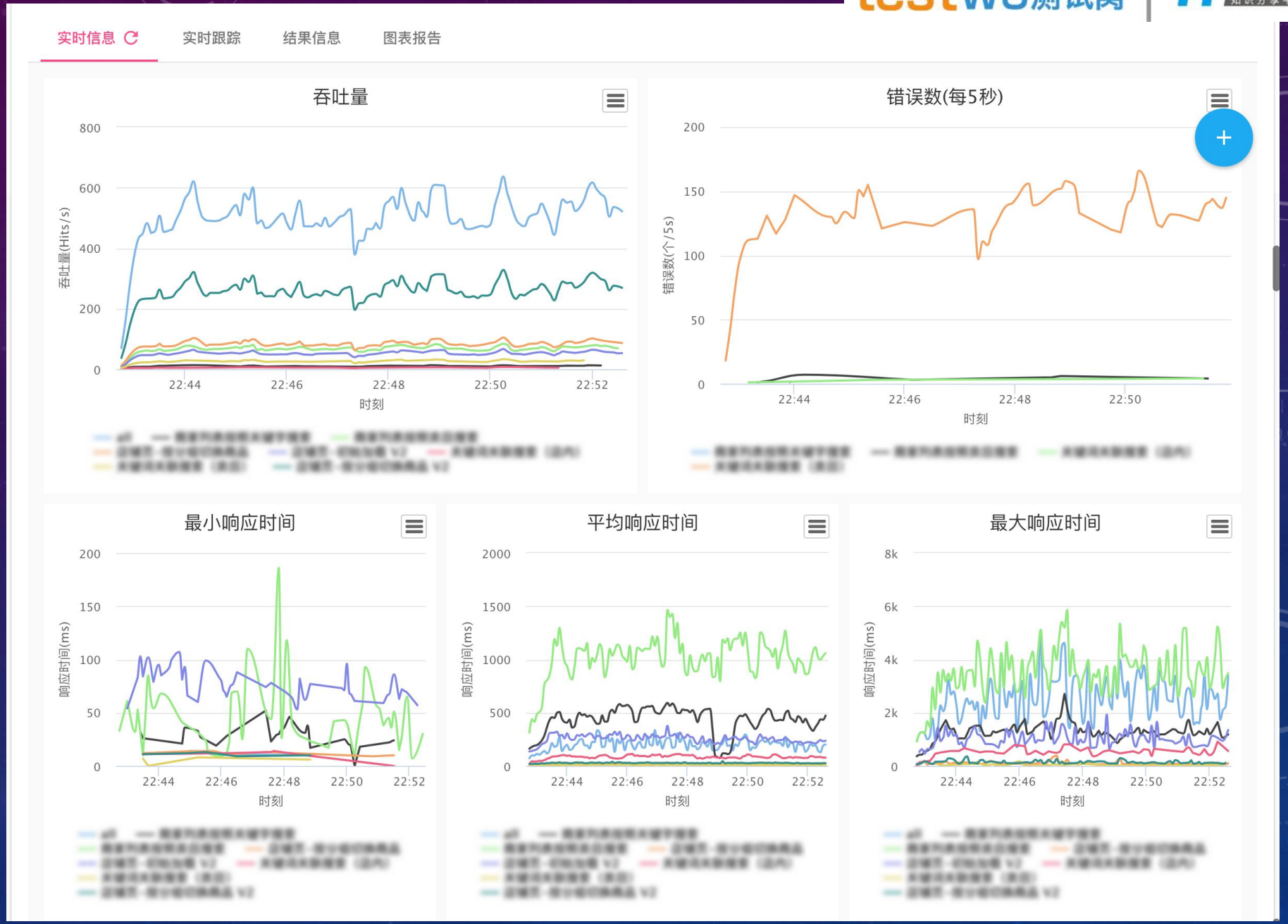
Backend Listener implementation: org.apache.jmeter.visualizers.backend.influxdb.InfluxdbBackendListenerClient

Async Queue size: 5000

Parameters

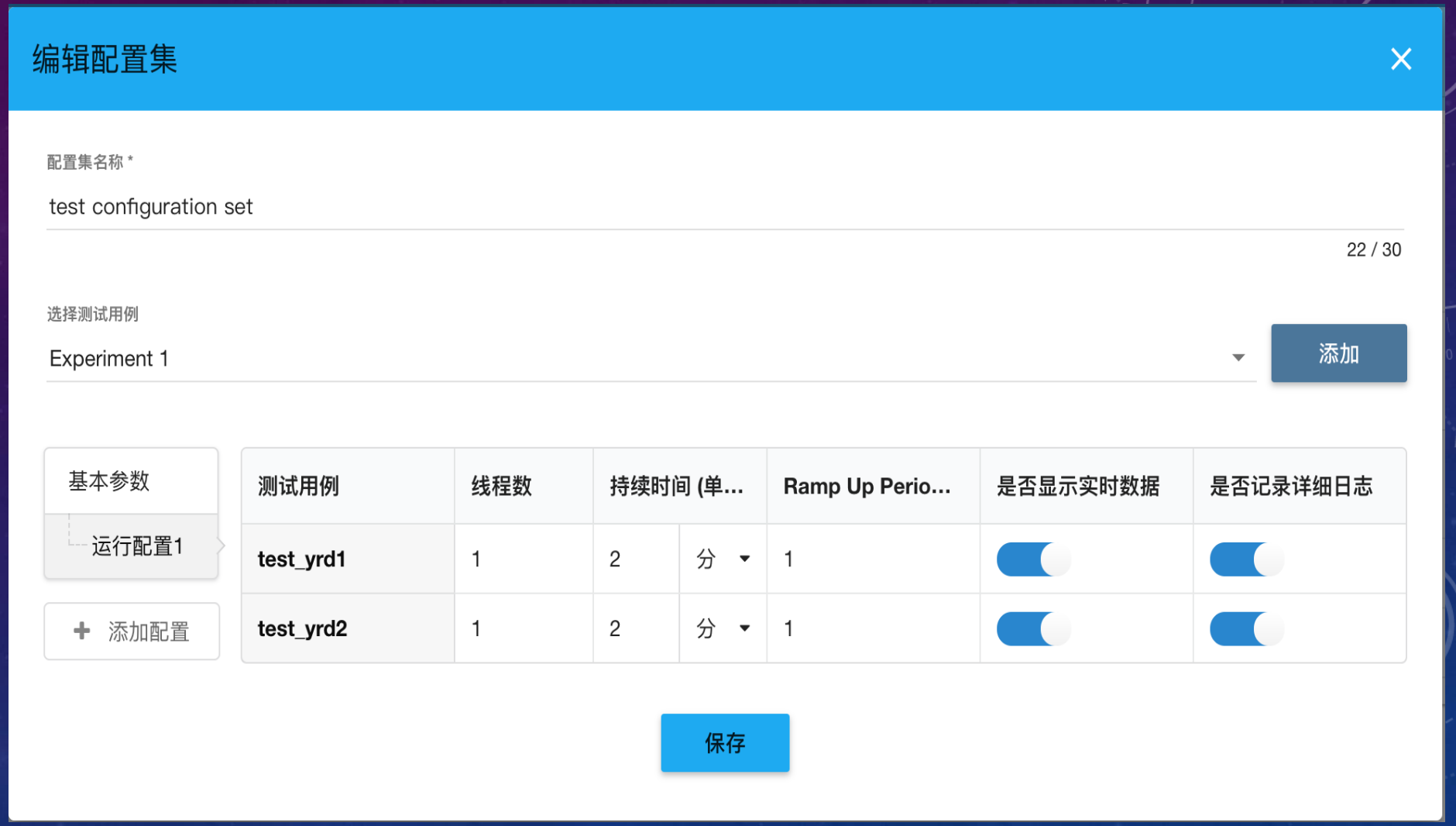
Name:	Value
influxdbMetricsSender	org.apache.jmeter.visualizers.backend.influxdb.UdpMetricsSender
influxdbUrl	influxDbHost:8089
application	application name
measurement	jmeter
summaryOnly	true
samplersRegex	.*
percentiles	90;95;99
testTitle	Test name
eventTags	

实时数据


实时信息
实时跟踪
结果信息
图表报告
吞吐量
吞吐量(Hits/s)
800
600
400
200
0
22:44
22:46
22:48
22:50
22:52
时刻
错误数(每5秒)
错误数(个/5s)
200
150
100
50
0
22:44
22:46
22:48
22:50
时刻
最小响应时间
响应时间(ms)
200
150
100
50
0
22:44
22:46
22:48
22:50
22:52
时刻
平均响应时间
响应时间(ms)
2000
1500
1000
500
0
22:44
22:46
22:48
22:50
22:52
时刻
最大响应时间
响应时间(ms)
8k
6k
4k
2k
0
22:44
22:46
22:48
22:50
22:52
时刻

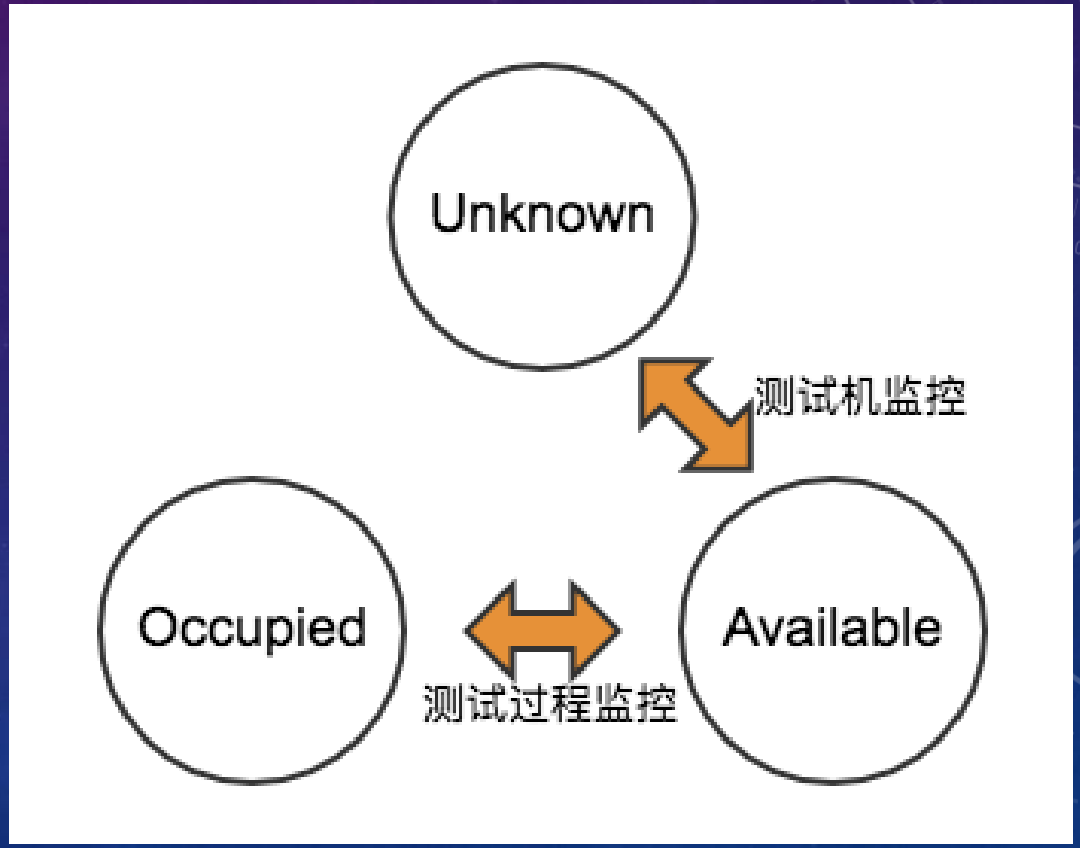
配置集

- 来源于实践中的痛点需求
- 建立测试用例间的纽带
- 一键触发多个测试用例



监控

- 测试机监控 – 类心跳机制
- 测试过程监控 – 异步Job/类状态机



监控

压测机管理与监控

共46台 空闲 占用 锁定 未知

压测机	压测机	压测机	压测机
机器名称 *	机器名称 *	机器名称 *	机器名称 *
al-app-test-96	al-app-test-95	al-app-test-94	al-app-test-93
14 / 20	14 / 20	14 / 20	14 / 20
Host地址 * al-app-test-96	Host地址 * al-app-test-95	Host地址 * al-app-test-94	Host地址 * al-app-test-93
SSH端口 * 0	SSH端口 * 0	SSH端口 * 0	SSH端口 * 0

压测机	压测机	压测机	压测机
机器名称 *	机器名称 *	机器名称 *	机器名称 *

+

安全保障

权限管理：用户权限分级管理，不能随意触发他人的测试用例，同时高峰期和禁止发布期，不允许执行任何测试。

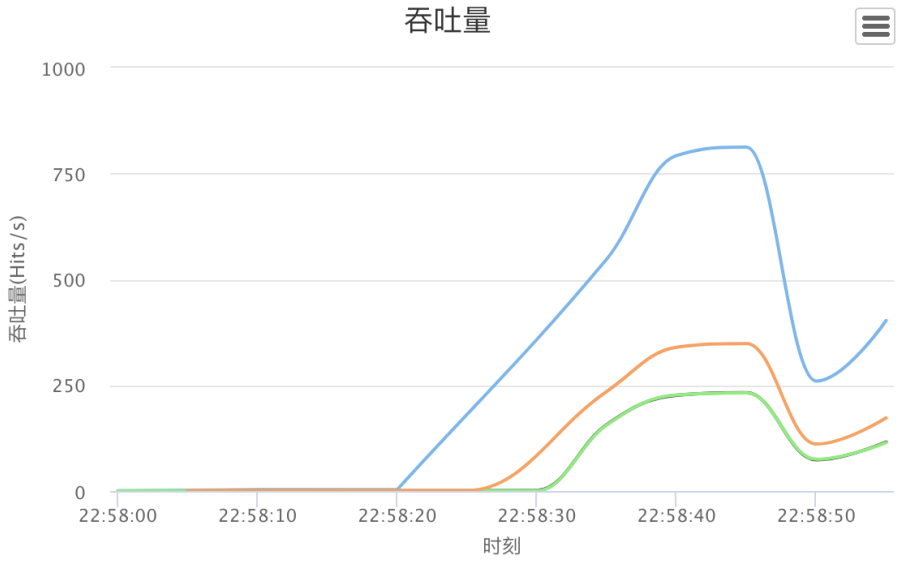
停止功能：面向用户的手动停止功能，用户可以随时点击运行状态下测试用例上的停止按钮，后台会直接kill掉所有运行该测试用例的测试机上的JMeter进程。

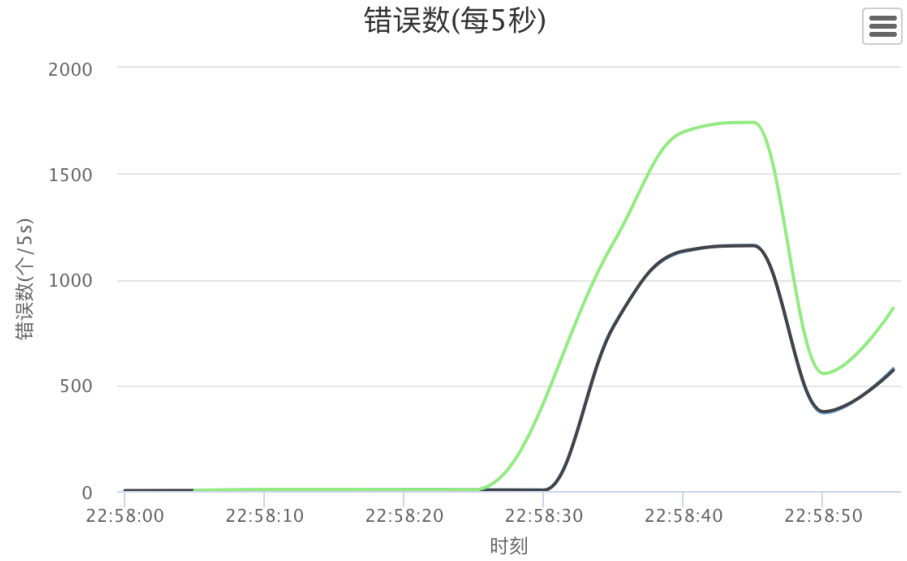
熔断功能：系统会根据实时信息中的错误率进行判断，当一定时间内的实时错误率达到或超过某个阈值时，该次测试将被自动熔断，无需用户干预。

兜底脚本：最极端的情况，当整个系统不可用，而此时需要停止测试时，我们提供了一份外部脚本直接进行停止。

熔断实例

实时信息 | 实时跟踪 | 结果信息 | 图表报告





出错信息

⚠ 实时错误率达到100.00%，已超过错误率上限15.0%,测试用例被熔断

14496 个	240.10 Hits/s	27.00 ms	0.00 ms	3,058.00 ms	14496(100.00%)
样本总数	平均吞吐量	平均响应时间	最小响应时间	最大响应时间	错误总数(错误率)

结果收集

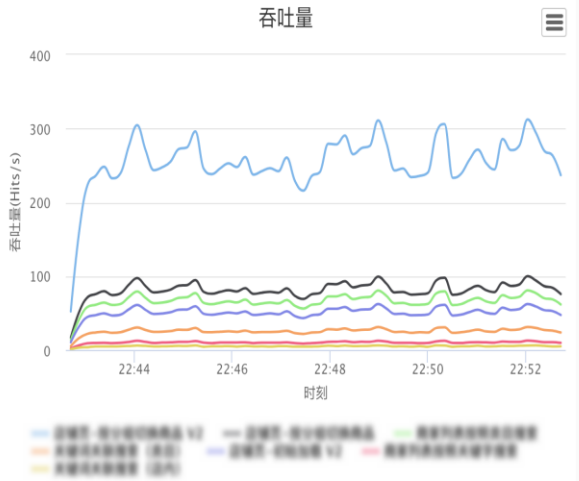
冷数据收集 - 测试结束后基于JTL进行数据合并和聚合，数据精度需要考虑

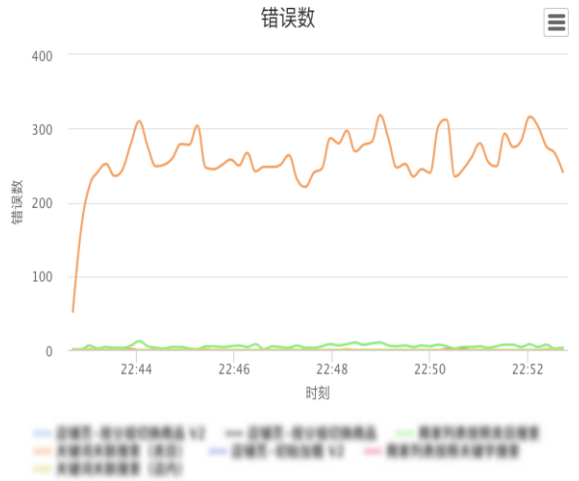
- 趋势类曲线(如平均响应时间)：不需要太精确，以固定60个时间片聚合
- 特征点曲线(如最大响应时间)：需要精确，以固定60个时间片聚合取特征值
- 特征点单值(如90/95/99线)：需要精确

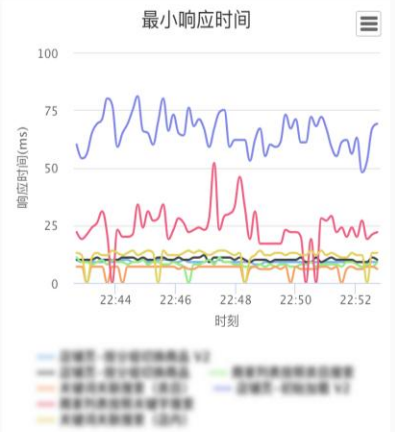
实时信息 | 实时跟踪 | 结果信息 | 图表报告

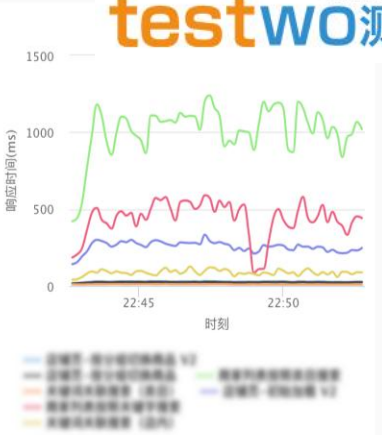
详细信息 @ 20171018 22:42

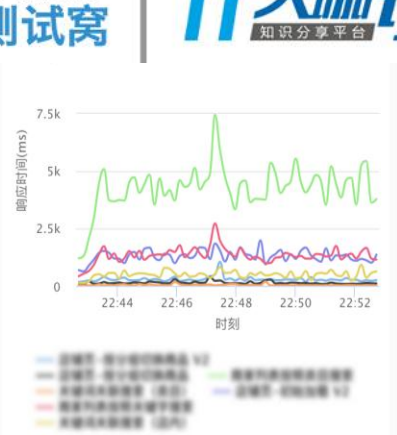
样本	S(满意阈值)	T(容忍阈值)	F(失望阈值)	APDEX
[redacted]	0.5s	2.0s	8.0s	1.000
[redacted]	0.5s	2.0s	8.0s	1.000
[redacted]	0.5s	2.0s	8.0s	0.517
[redacted]	0.5s	2.0s	8.0s	1.000
[redacted]	0.5s	2.0s	8.0s	1.000
[redacted]	0.5s	2.0s	8.0s	0.842
[redacted]	0.5s	2.0s	8.0s	1.000

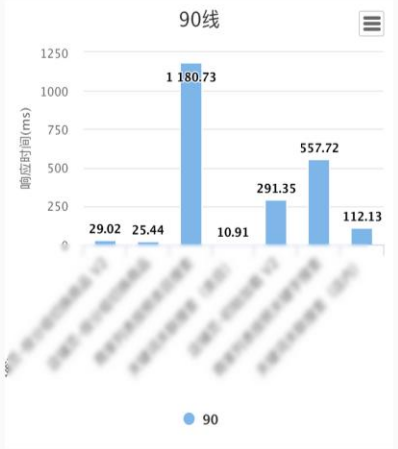


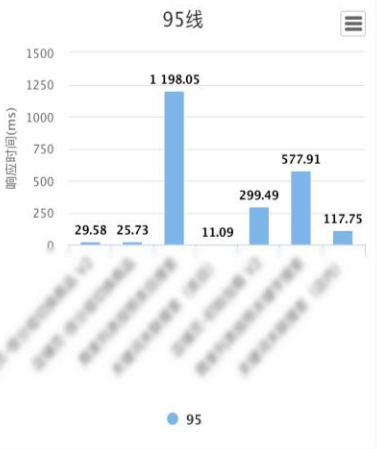


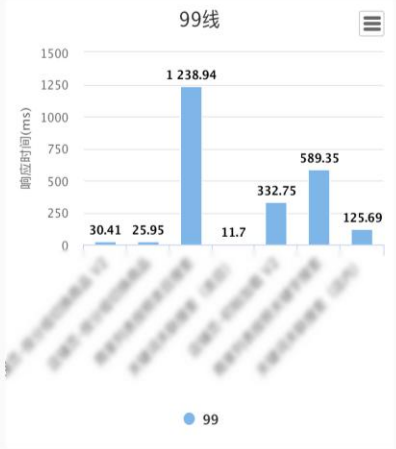


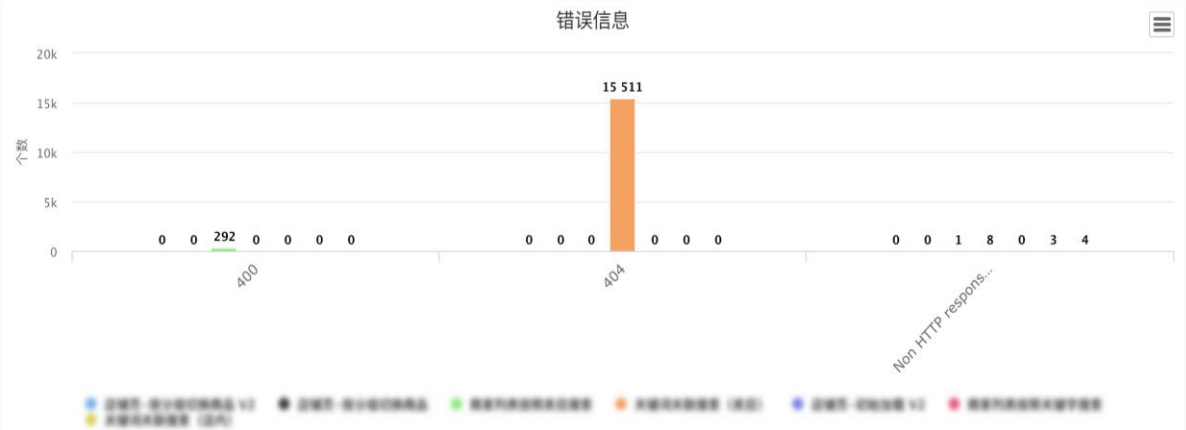












成果

全链路压测平台自今年7月上线以来，为超过5个部门累计提供了上千次测试服务。按照每一类测试配置和执行的人力成本为15分钟计算，大约节省了1000个小时的工作量。

后期构想

- 更好上手：用户无须掌握JMeter知识
- 平台联动：各平台打通，真正建立性能测试生态圈体系
- 更智能：让平台更“聪明”