性能驱动
正确打开姿势

聂永 @ 新浪微博

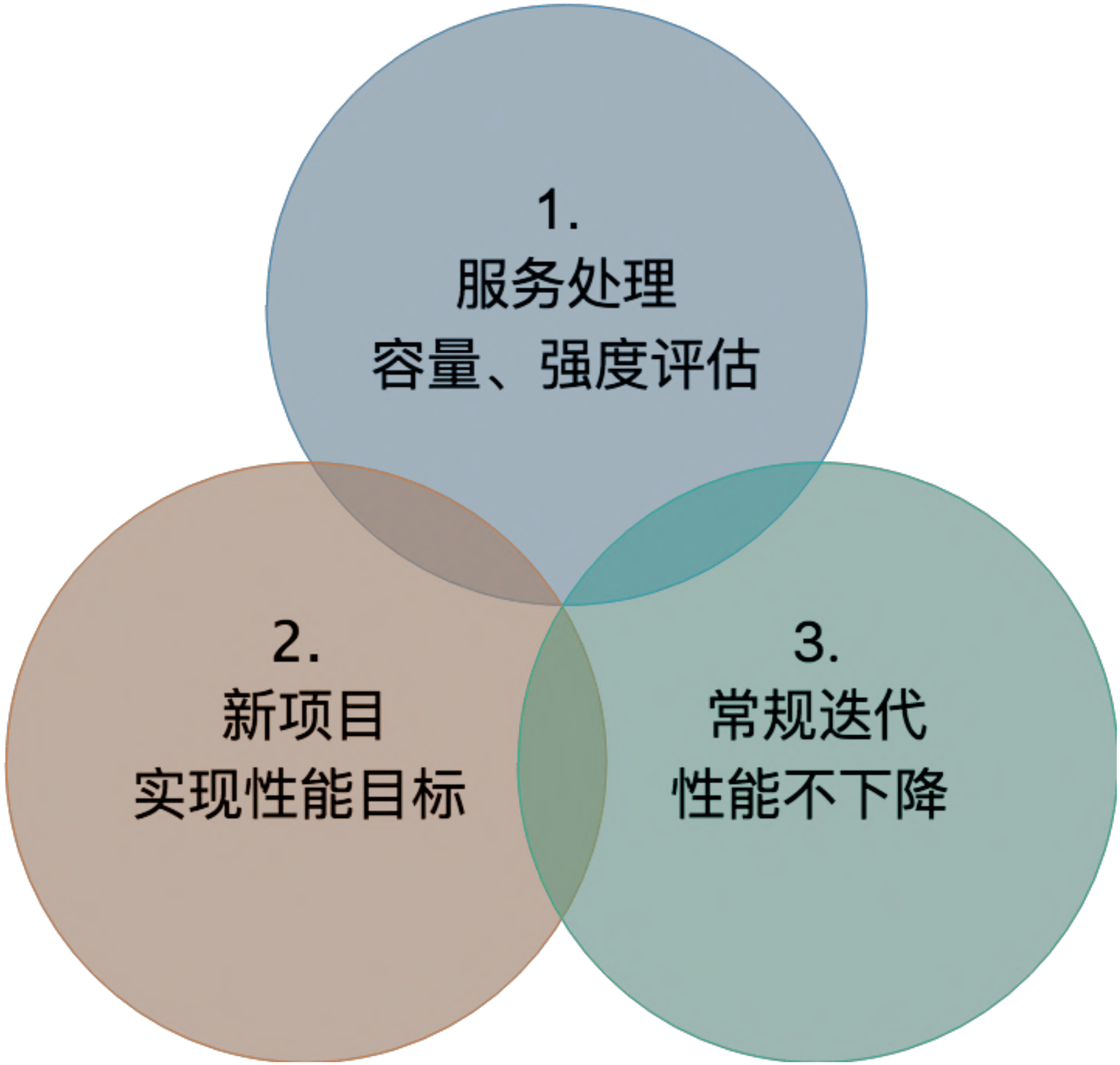
一、需求

- 有状态的TCP应用
- 长连接
- 实时、强交互聊天室系统
- 私有协议
- 需要支持海量用户

1.1 项目背景

- 配给：一名中级工程师 + 一名初级工程师
- 产品要求系统性能报告

1.2 挑战



二、实践过程

1. 设立性能考核标准

- 单机为16G内存24核服务器
- 支持50W用户
- QPS 5000
- 平均耗时50毫秒

2. 趁手工具

- 加压速度快，CPU&内存消耗低
- 支持分布式环境，可产生海量用户
- 支持TCP、HTTP、SSL等通道
- 支持复杂情景、强交互会话设定
- 可以生成压测结果报表
- 开源，免费，需要时可定制

2.1 现状

- 没有多余空闲机器用作压测机
- 线上服务器白天时段负载不高
- 性能测试将是一个长期持续行为

2.2 选择哪一款呢？

TcpCopy
JMeter
nGrinder
Tsung
.....

TcpCopy

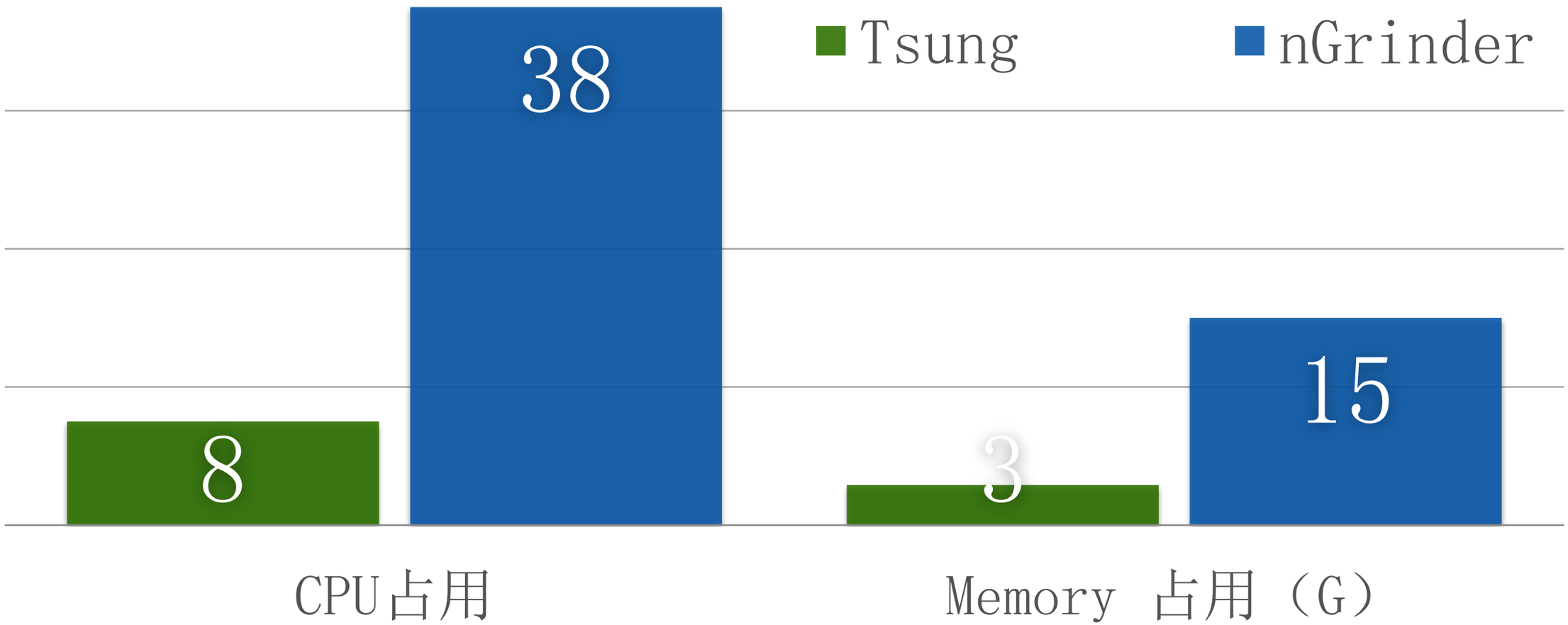
- 线上引流
- 针对新功能/项目不适合
- 想达到指定量用户不适合
- 交互弱

JMeter

- Java应用
- 1万线程，CPU上下文切换频繁
- 大量并发时会有内存溢出问题
- 输出压力不稳定

nGrinder

单机5万长连接资源消耗对比



为什么是Tsung

- 基于Erlang，并发处理性能好，可模拟海量用户，和压测机器数量成正比
- 受益于Erlang，天然支持分布式，可集群作业
- 支持协议：WebDAV/WebScoket/MQTT/MySQL/PGSQL/Shell/AQMP/Jabber/XMPP/LDAP等
- 传输通道支持 TCP/UDP/SSL，更底层支持IPv4/IPv6
- 支持单机多个IP：虚拟IP/网卡绑定IP
- 支持监控被压测的服务器：Erlang/SNMP/Munin
- 压测会话XML可配，基于情景的压力测试行为清单，模拟复杂、具体业务场景
  - 多个业务场景可混合到一起
  - 可编程的业务需求

2.3 Tsung不足-SSH

- Tsung主节点需要通过SSH连接并启动从机
- SSH免密钥登陆设置麻烦
- 网络机房不支持服务器之间SSH联通
  - Tsung集群无效

SSH替换

- 设计一套简单有效的SSH替换方案
- 需要两个shell脚本：客户端 + 服务器端
- 资源协调特性，避免占用资源过多
- 已开源：
  https://github.com/weibomobile/tsung_rsh
- 提升200%运维效率

2.4 Tsung不足-主机名

- 只支持主机名，不支持IP直连
- 每增加新的服务器都需要配置/etc/hosts
- 网络访问会增加一次DNS解析请求过程

对Tsung的修改

- 修改源码提供IP直连支持
- 源码：
  https://github.com/weibomobile/tsung
- 已提交修改：
  https://github.com/processone/tsung/pull/189
- 运维效率提升100%

2.5 其它修改

- SO_REUSEPORT特性支持
- 强交互变量获取优化，提升200%
- ……

2.6 压测机计算方式

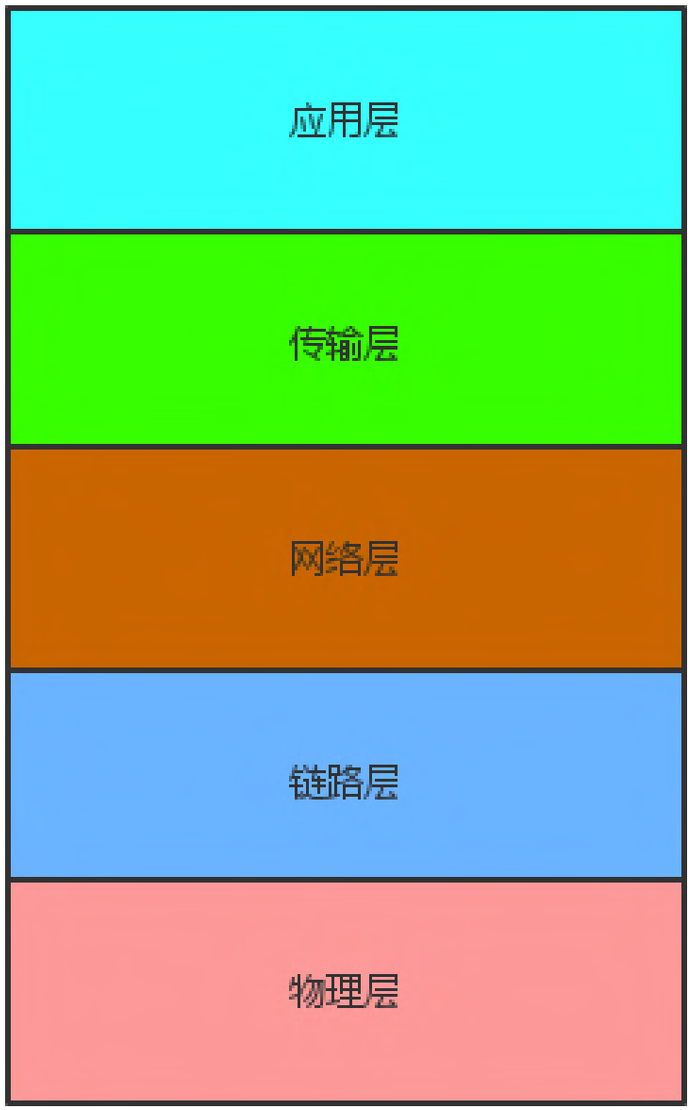
- 可用端口数量
- 内存占用
- 可用IP地址

端口相关

- {srcIP，srcPort，target_IP，target_Port}
- Unix/Linux端口为16位正整数，1~65535范围
- 1024以下端口需要超级管理员ROOT用户权限
- 一般设置对外范围 1024~65535 ≈ 6W

```
sysctl -w net.ipv4.ip_local_port_range="1024 65535"
sysctl -p
```

- 理论上对外可建立连接数 $Total = N_{ip} * Port_{6w}$



内存占用

- Tsung默认的网络发送、接收缓冲区为16K
- 一个网络连接对应一个用户，业务内存约10K
- 一个用户内存占用50K，1万用户占用500M内存
- 还有程序自身占用内存
- 实际，一个压测机模拟6W用户，约4G内存

需要多少台压测机

- 100万用户压测所需资源：
  - 100万/6万 ≈ 17个IP地址
    - 500M * 100万 / 1000 ≈ 50G内存
- 服务器单一IP，承载100万用户：1台主机 + 17台从机
- 若：单机64G内存 + 17个可用IP地址 -》一台压测机器搞定100万用户

2.7 除此之外

- 网络内核参数，资源快速回收作用

```
sysctl -w net.ipv4.tcp_syncookies=1
sysctl -w net.ipv4.tcp_tw_reuse=1
sysctl -w net.ipv4.tcp_tw_recycle=1
sysctl -w net.ipv4.tcp_fin_timeout=30
```

- 打开文件句柄限制

```
echo "* soft nofile 300000"   << /etc/security/limits.conf
echo "* hard nofile 300000" << /etc/security/limits.conf
```

- 更多细节，请直接参考《Tsung笔记之压测端资源限制篇》
  http://www.blogjava.net/yongboy/archive/2016/07/26/431322.html

3. 设计测试会话内容

三个原则

模拟 **全面** **强度**

- 模拟用户真实的交互行为
- 覆盖面要广：加入房间、点赞、打赏、关注、发言、退出等
- 完整反馈到压测会话内容中
- 实际场景的复杂度和强度

3.1 设计压测会话内容

- 包含协议定义所有行为示范
- 一个用户完整交互集合
  - 用户加入 x1
  - 用户发言内容随机 X51
  - 心跳 X8
  - 用户点击 X20
  - 用户空闲时间 > 24分钟
  - 用户退出 x1
  - ...

4、关注点

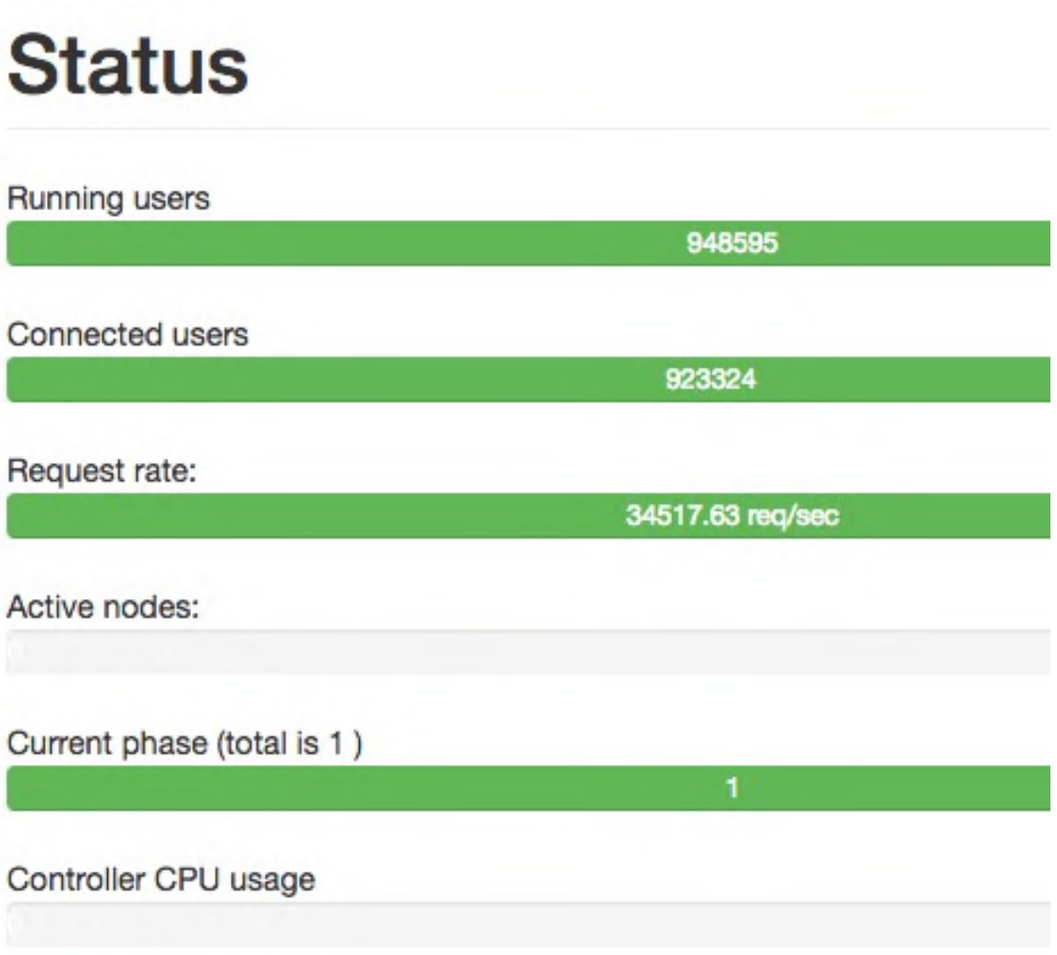
- 测试前
- 测试中
- 测试后

4.1 准备

- 计数器
- 核心、非核心模块
- 诊断日志
- ……

4.2 压测中，我们该关注什么

- 业务各项报表
- 服务器CPU/内存/IO占用
- 日志文件中出错、异常信息
- 作为终端用户亲身体验、参与到互动中！
- Tsung运行时报表：http://IP:8091



4.3 测试后

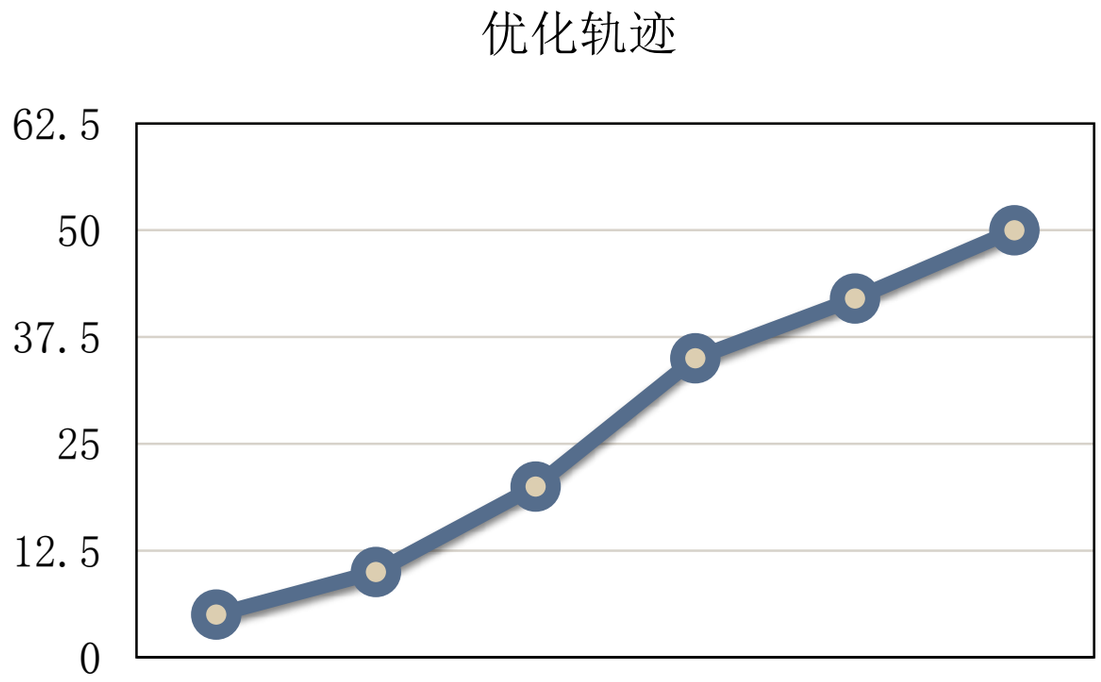
- 复查各种报表
- 从用户体验上发现问题
- 针对日志寻找异常
- 认真思考，修改问题
- 继续下一轮压测

5. 全链路压测

- 单机全链路
- 线上全链路

5.1 线下单机

- 50万用户压测结果
- QPS突破7K
- 平均耗时16毫秒



Main Statistics

Name	highest 10sec mean	lowest 10sec mean	Highest Rate	Mean Rate	Mean	Count
connect	2.13 msec	1.36 msec	312.6 / sec	120.50 / sec	1.48 msec	500000
page	55.74 msec	4.02 msec	14919.7 / sec	7366.88 / sec	16.09 msec	30500000
request	55.74 msec	4.02 msec	14919.7 / sec	7366.88 / sec	16.09 msec	30500000
session	40mn 36sec	40mn 35sec	314.5 / sec	292.40 / sec	40mn 36sec	500000

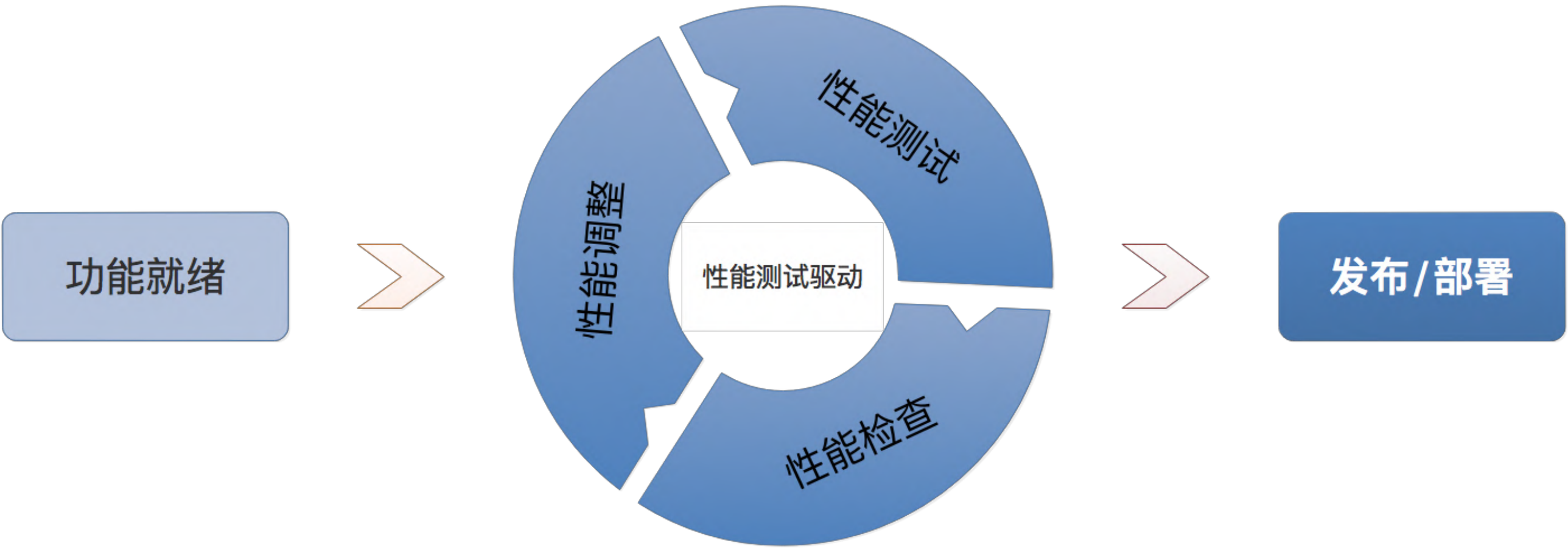
5.2 线上全链路压测

Main Statistics

Name	highest 10sec mean	lowest 10sec mean	Highest Rate	Mean Rate	Mean	Count
connect	2.50 msec	1.12 msec	1023.5 / sec	663.69 / sec	1.39 msec	2953345
page	42.03 msec	3.80 msec	28300 / sec	13495.10 / sec	5.00 msec	60053123
request	42.03 msec	3.80 msec	28300 / sec	13495.10 / sec	5.00 msec	60053123
session	40mn 36sec	5.00 sec	528.9 / sec	222.72 / sec	39mn 57sec	999911

- 避免影响线上业务
- 线上处理容量的1/20
- 全程监控，防止意外

三、实践总结



常规迭代


功能测试
确保系统业务逻辑运行正确
系统稳定性如何，会不会崩溃
并发请求是系统响应时间如何
高负载下业务功能正常工作
性能测试
一定用户量请求系统表现稳定
并发请求响应时间不低于标准
高负载下业务逻辑运行正确
目标达成
性能优化
解决系统崩溃问题
提高业务处理速度
确保核心逻辑正常运行

Thank You :))