



贝壳找房的Node服务 稳定性建设

极客邦科技 会议推荐2019



InfoQ官网 全新改版上线

促进软件开发领域知识与创新的传播



关注InfoQ网站
第一时间浏览原创IT新闻资讯



免费下载迷你书
阅读一线开发者的技术干货

自我介绍



徐辛承

- 7年前端工作经验，毕业于北航
- 人人网、百度、百度外卖/饿了么/阿里巴巴，技术专家
- 写过一本书：《Vue移动开发实战技巧》
- 贝壳找房，C端前端负责人

为什么要用Node

SEO

✓ 服务端渲染

效率

✓ 前后端同构

性能

✓ 首屏更快



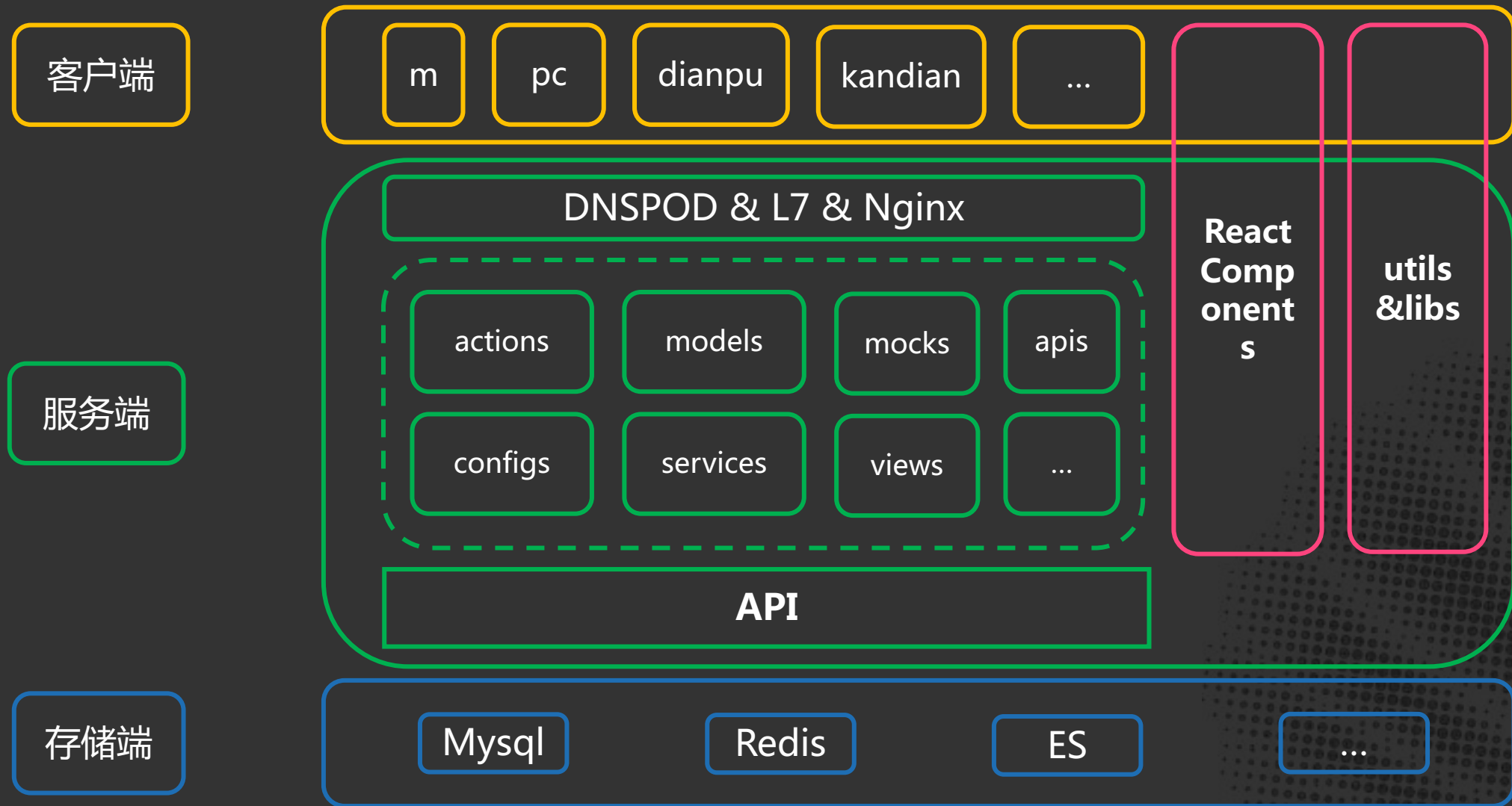
Bucky & React



+



技术架构



我们做到了



4000万

月活跃用户



300万

用户push



15万

经纪人整点抢购



稳定性

99.999%

目录

- 预防问题
- 发现问题
- 处理问题
- 总结问题

预防问题



压力估算和压测

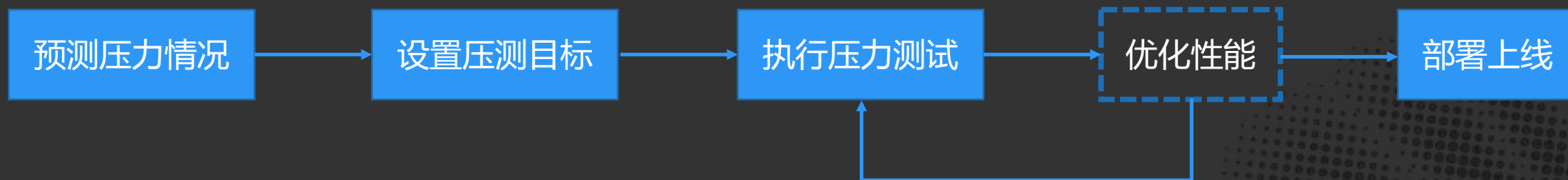


Code Review

预防问题：压力估算和压测

- 经纪人店铺新项目上线
- 经纪人抢购

能不能扛得住？



第一步：预测压力情况

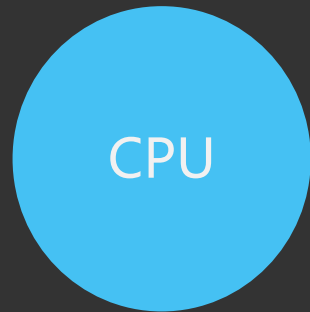
➤ 经纪人店铺

- 200万请求/天 ($PV \times \text{平均请求数/页面}$)
- 预计80%的流量集中在20%的时间
- 集群RPS : $2000000 \times 0.8 \div (24 \times 0.2 \times 3600) \approx 92.6 \text{ RPS}$

➤ 经纪人抢购

- 高峰抢购时大概 2000个用户/s , 并发访问量预计翻倍 : 4000/s
- 希望返回时间不超过1S
- 集群RPS : $4000 \div 1 = 4000 \text{ RPS}$

第二步：设置压测目标



✓ 多核：<30%



✓ 不成为瓶颈



✓ 70%



✓ 70%



第三步：执行压力测试

- 网络环境相同
- 独立的压力机
- 线上机器
 - ✓ 深夜压测
 - ✓ 混部，周知相关方



第四步：部署上线

- 集群机器数 = $\boxed{\text{期望RPS}} \div \boxed{\text{单机实际RPS}} \div \boxed{0.7}$
- 抢购：集群期望RPS 4000 \div 单机实际RPS 800 \div 0.7 \approx 7.14
- 部署7台机器



预防问题：Code Review

➤ 变量问题

- ✓ 根据UA不同，执行不同的逻辑

➤ 性能问题

- ✓ async await 串行 || 并行

➤ 硬编码问题

- ✓ 环境变量，api domain

```
// 开发环境配置
export const development = api => {
  // api.config.base = 'http://ayx.api.home.ke.com'
  api.config.base = config.devDomain
}

// 测试环境配置
export const testing = api => {
  api.config.base = config.testDomain
}

// 预发环境配置
export const preview = api => {
  api.config.base = 'http://test.api.home.ke.com'
}
```

发现问题



```
graph LR; A[值班机制] --> B[服务异常监控]; B --> C[服务器资源监控];
```

值班机制

服务异常监控

服务器资源监控

异常监控：监控类型

➤ Nginx日志

✓ 499

✓ 404

➤ 服务日志

✓ 5XX

HTTPCODE	499
HTTPCODE	404
HTTPCODE	503
HTTPREFER	*
PROVINCE	北京
PageSchema	-
REMOTEIP	123.114.40.206
REQID	*
REQUESTBODYSIZE	2030
REQUESTHOST	3meima.lianjia.com
REQUESTMETHOD	GET
REQUESTPATH	/dzls/index.php/index
REQUESTPROTOCOL	HTTP/1.1

异常监控：Node日志

➤ Node服务日志

- ✓ Access.log：客户端每一次请求
- ✓ Api.log：发起一次api请求
- ✓ Error.log：所有的错误

➤ 慎用try catch

- ✓ Metrics监控

```
"cost": 5011,
"method": "GET",
"actionUri": "http://dianpu.lianjia.com/renderNone",
"ip": "10.26.100.2",
"channel": "request",
"message": "output",
"uri": "http://dianpu.lianjia.com/shop/getbaseinfo/1000001000127830",
"sequence": 5,
"user_id": 0,
"context": "Internal Error<pre>Error: ETIMEDOUT\\n uri:
http://i.purist-api.lianjia.com/seo/pcfooter/innerlink?
cityId=110000&source=ljweb&url=http%3A%2F%2Fbj.lianjia.com&
request_source=ljweb\\n displayName: API.ljFooter.getPCFooter\\n file:
/data0/www/htdocs/dianpu.ke.com/dist/server/apis/ljFooter.js\\n\\n\\n
${app}/node_modules/request/request.js:849:19\\n\\n 845 | // time, not
the time between bytes sent by the server.\\n 846 | self.timeoutTimer =
setTimeout(function () {\\n 847 |
socket.removeListener(&#039;connect&#039;;, onReqSockConnect)\\n 848 |
self.abort()\\n&#039;&#039; 849 | var e = new Error(&#039;ETIMEDOUT&#039;)&#039;&#039;
^\\n 850 | e.code = &#039;ETIMEDOUT&#039;\\n 851 | e.connect = true\\n
852 | self.emit(&#039;error&#039;;, e)\\n 853 | }, timeout)\\n 854 | }
else {\\n\\n\\n at Timeout._onTimeout
(${app}/node_modules/request/request.js:849:19)\\n at ontimeout
(timers.js:469:11)\\n at tryOnTimeout (timers.js:304:5)\\n at
Timer.listOnTimeout (timers.js:264:5)</pre>",
"request_id": "c3e0bc52-8356-45a6-b616-adaf1158c0b7",
"timestamp": "2019-06-03T17:06:13.000+08:00"
```

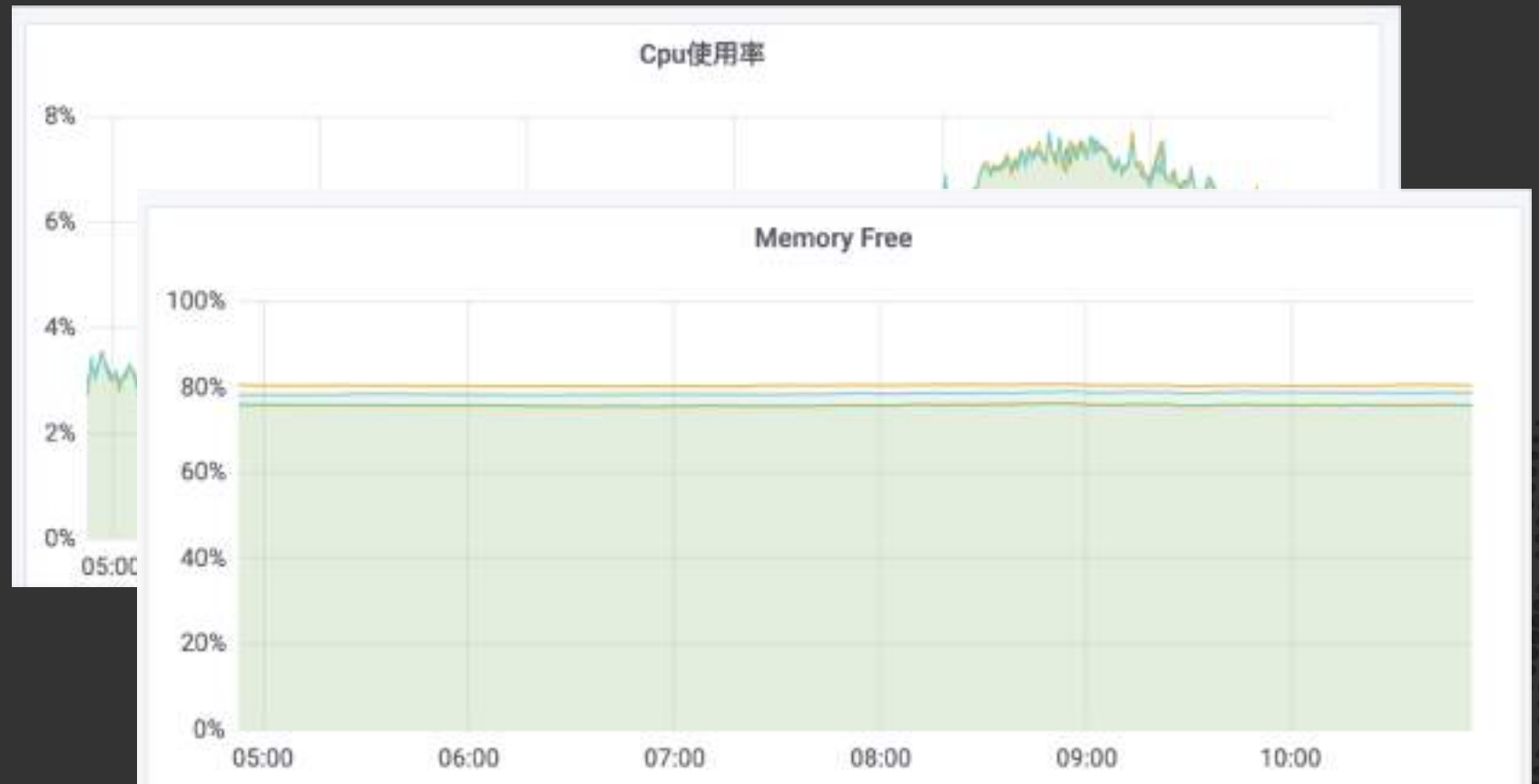
发现问题：资源监控

➤ CPU

✓ 40%

➤ Memory

✓ 70%



处理问题



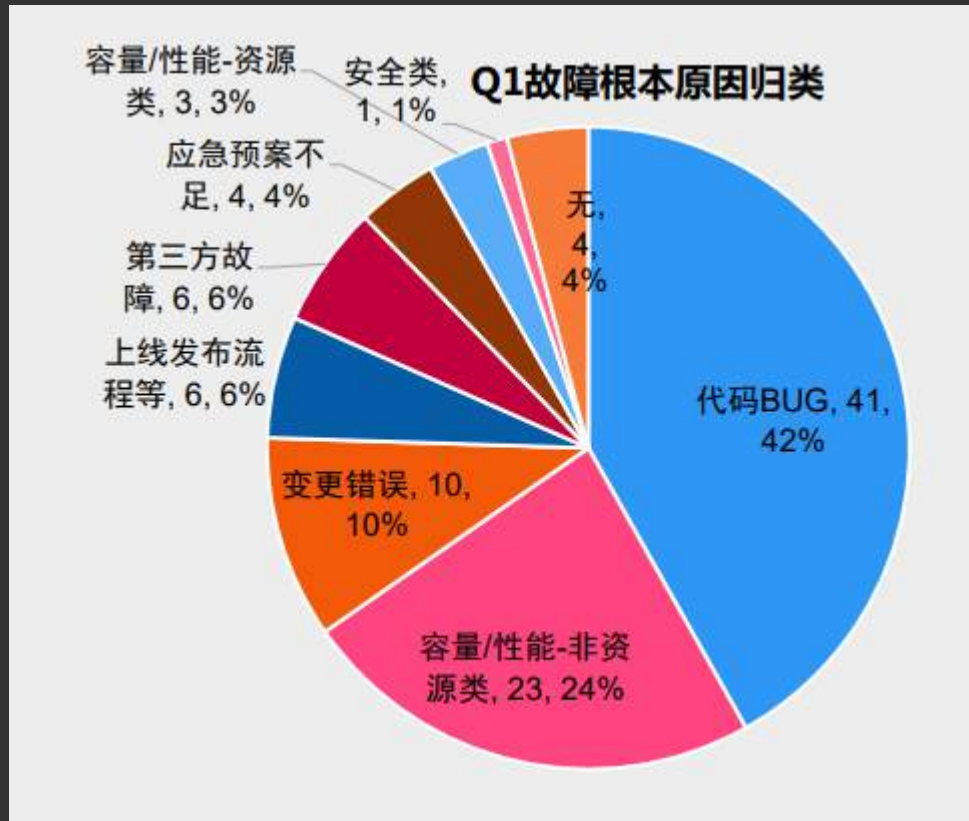
原则：恢复大于原因查找

➤ 快速回滚

➤ 快速定位

➤ 重启和扩容

处理问题：快速回滚



回滚步骤

发布平台

快速回滚

快速定位：日志检索

➤ 错误信息和请求信息

➤ 499

- 主动断开
- 大量超时

➤ 5XX

➤ 404

- 上线引发的路由问题
- 大量爬虫

Time	CLIENTIP
▶ June 18th 2019, 11:53:11.000	114.255.24.10
▶ June 18th 2019, 11:53:11.000	114.255.24.10
▶ June 18th 2019, 11:53:11.000	114.255.24.10
▶ June 18th 2019, 11:53:11.000	114.255.24.10
▶ June 18th 2019, 11:53:11.000	114.255.24.10
▶ June 18th 2019, 11:53:11.000	114.255.24.10
▶ June 18th 2019, 11:53:10.000	114.255.24.10
▶ June 18th 2019, 11:53:10.000	114.255.24.10
▶ June 18th 2019, 11:53:10.000	114.255.24.10
▶ June 18th 2019, 11:53:10.000	114.255.24.10
▶ June 18th 2019, 11:53:10.000	114.255.24.10
▶ June 18th 2019, 11:53:10.000	114.255.24.10
▶ June 18th 2019, 11:53:09.000	114.255.24.10
▶ June 18th 2019, 11:53:09.000	114.255.24.10
▶ June 18th 2019, 11:53:09.000	114.255.24.10



快速定位：无法直观看出出的问题

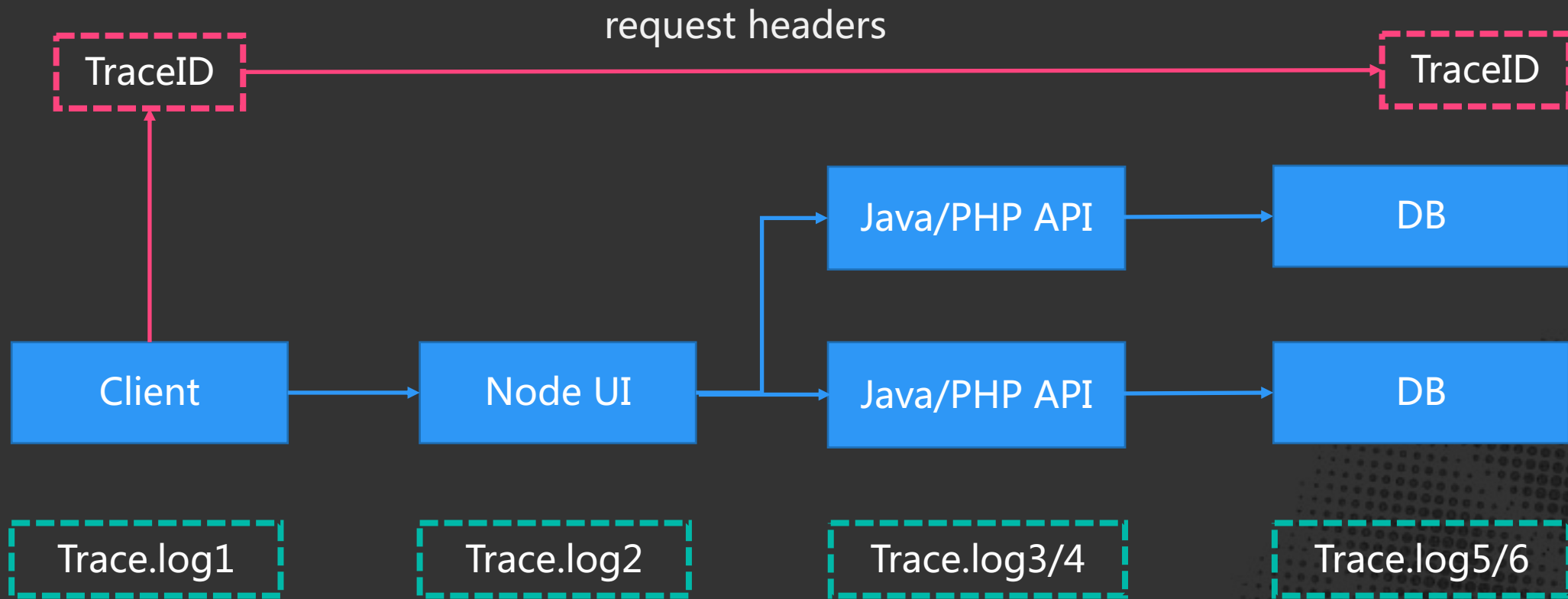
```
t _id      🔍 🔍 🔍 * AWtK-7vYNgjrvEijE_Kh
t _index   🔍 🔍 🔍 * index-719-967_2019-06-12
# _score    🔍 🔍 🔍 * -
t _type     🔍 🔍 🔍 * index-719-967
t channel   🔍 🔍 🔍 * error
t context   🔍 🔍 🔍 * Cannot read property '0' of undefined\n\n TypeError: Cannot read property '0' of undefined\n\n  ${app}/dist/serve
  /models/designerCase.js:98:37\n\n    94 |         // k.url = k.url.split('!')[0] + '!m.
      |         fill,m_' + bigImgWidth\n    95 |         });\n    96 |         } else i
      |         f (v.type === 'furniture') {\n    97 |             v.list && v.list.forEach(function (m) {\n
      |             if (m.url_list[0].url.indexOf('.gif') < 0) {\n
      |                 var resp2 = (0, _utils.dealPic)(m.url_list[0].url.split('!')[0], 1
      |                 m.url_list[0].url = resp2.url;\n    101 |                 m.url_list[0].cutItem = resp
      |                 2.cutItem;\n    103 |             }\n      |             }\n      |             \n      |             at ${app}/dist/server/models/DesignerCase.js:98:37
      |             \n      |             at Array.forEach (<anonymous>)\n      |             at ${app}/dist/server/models/DesignerCase.js:97:48\n      |             at Array.forEach (<anonymous>)\n      |             at ${app}/dist/server/models/DesignerCase.js:86:54\n      |             at Array.forEach (<anonymous>)\n      |             at ${app}/dist/server/models/DesignerCase.js:84:48\n      |             at Array.forEach (<anonymous>)\n      |             at Function._callee
      |             $ ($app)/dist/server/models/DesignerCase.js:68:24\n      |             at tryCatch ($app)/node_modules/regenerator-runtime/r
      |             untine.js:62:40)

t message   🔍 🔍 🔍 * TypeError
t request_id 🔍 🔍 🔍 * c4c1120b-3ed0-4a63-9a1d-a070a563ea17
t segmentId 🔍 🔍 🔍 * home.newn.ke.com-10.200.18.221-55033-1560331136910-Gp00v8aZiA
t spanId     🔍 🔍 🔍 * 0
t timestamp  🔍 🔍 🔍 * June 12th 2019, 17:18:56.000
t traceId    🔍 🔍 🔍 * home.newn.ke.com-10.200.18.221-55033-1560331136910-3RvNPw6Afp
t uri        🔍 🔍 🔍 * http://home.newn.ke.com/page/designer-case
```

✓ 看代码，可能哪个变量有问题

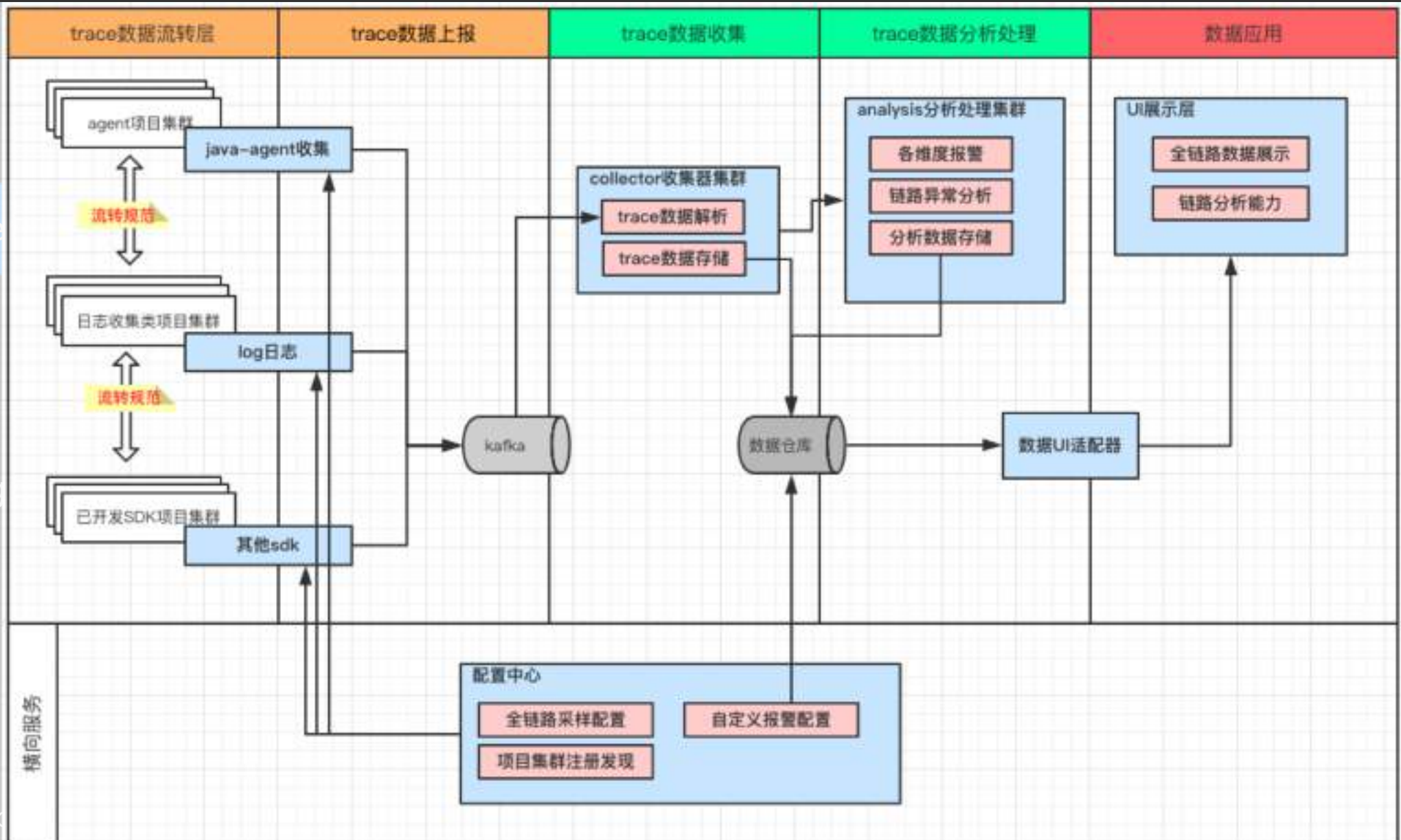
✓ 根据请求链路向下追溯

快速定位：链路追踪



- ✓ TraceID作为唯一标示串起整个链路
- ✓ 每个节点详细信息以log形式输出

快速定位：Ketrace



API

API

快速定位：链路追踪



处理问题：重启和扩容



内存占用过高



- 是否内存泄漏
- 重启服务



CPU Idle 过低



- 快速扩容
- 周期性部署
 - ✓ 预部署 + 周期性切流

总结问题



Case Study



- ✓ 处理过程：时间、人员、方式
- ✓ 直接原因和根本原因
- ✓ 可执行的整改计划

分享总结

➤ 预防问题

- ✓ 压力预估和压力测试
- ✓ Code Review

➤ 发现问题

- ✓ 异常监控
- ✓ 资源监控

➤ 处理问题

- ✓ 快速回滚
- ✓ 快速定位
- ✓ 重启和扩容

➤ 总结问题

- ✓ Case Study

展望

➤ 预防问题

- ✓ 压测平台
- ✓ 仿真环境&故障预演
- ✓ 自动化测试

➤ 发现问题

- ✓ 服务端埋点
- ✓ 线上巡检

➤ 处理问题

- ✓ 降级、限流、熔断、扩容、静态化预案
- ✓ 配置中心&预案平台



前端训练营

用3个月时间，彻底学透前端开发必备技能



了解详情

- ✓ 线下线上混合式学习
- ✓ 名师手把手教学
- ✓ 一线大厂项目实操
- ✓ 毕业即享内推服务



讲师：程劭非 (winter)
前手机淘宝前端负责人

重学前端

每天10分钟，重构你的前端知识体系

你将获得

告别零散技术点，搭建前端知识体系

打通JS、HTML、CSS、浏览器4大脉络

40+前端重难点完全解答

大厂前端工程实战演练



作者：winter（程劭非）

前手机淘宝前端负责人



扫码立即参与

到手价 **¥69** 原价¥99（仅限 **48** 小时）

参与拼团，结算时输入【GMTC用户专享优惠口令】：**2qianduan**

