

# 见微知著, APM在tutorabc的应用

杨大鹏

iTutorGroup集团云平台(TGOP)负责人

2017-08-11

*tutorabc*

真人在线 英语外教

tutorabc由美国硅谷技术团队研发创立, 全球首个在线教育“独角兽”公司。阿里巴巴集团、新加坡淡马锡基金、启明创投和日本软银SBI集团一亿美元B轮融资。

GIC、中俄基金、高盛、银翎资本等世界级投资方近两亿美元C轮融资。

*vipjr*

青少年在线教育

vipjr是由姚明代言的青少年在线教育品牌, 为5-18岁青少年提供包括英语、数学、托福雅思等多元化教学服务。

2017年4月, vipjr联合哈佛大学、宾夕法尼亚大学等最高学府的顶尖思想领袖, 成立美国学术顾问委员会。

*TutorABC*

*tutorJr*

*TutorMing*



LiveH2H

- 如何应对性能问题
- tutorabc对APM的一些实践
- TGOP私有云平台



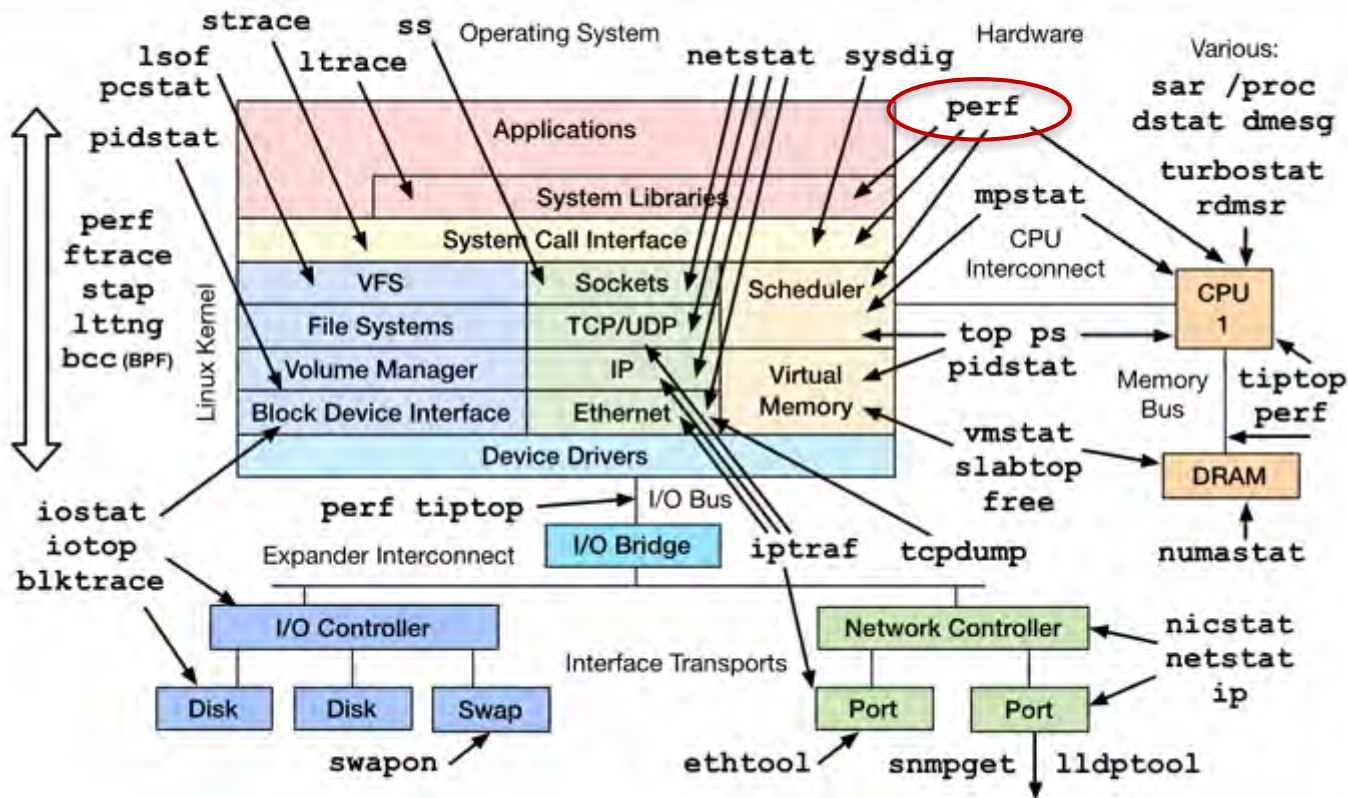
关注性能, 从现在开始!

线上某API服务高峰时段CPU用量超过95%。  
期间API响应变慢。  
重启服务后, 问题依旧存在。

之前通过添加新服务器缓解问题。  
考虑到近期业务量有大幅增长, 同时为节约计算资源, 需要对服务进行优化。  
PS: 本地环境可以重现该问题。

如何优化?

## Linux Performance Observability Tools





程序日志

JProfiler(Java), top, netstat等

gdb, Perf {PMU, TracePoint}

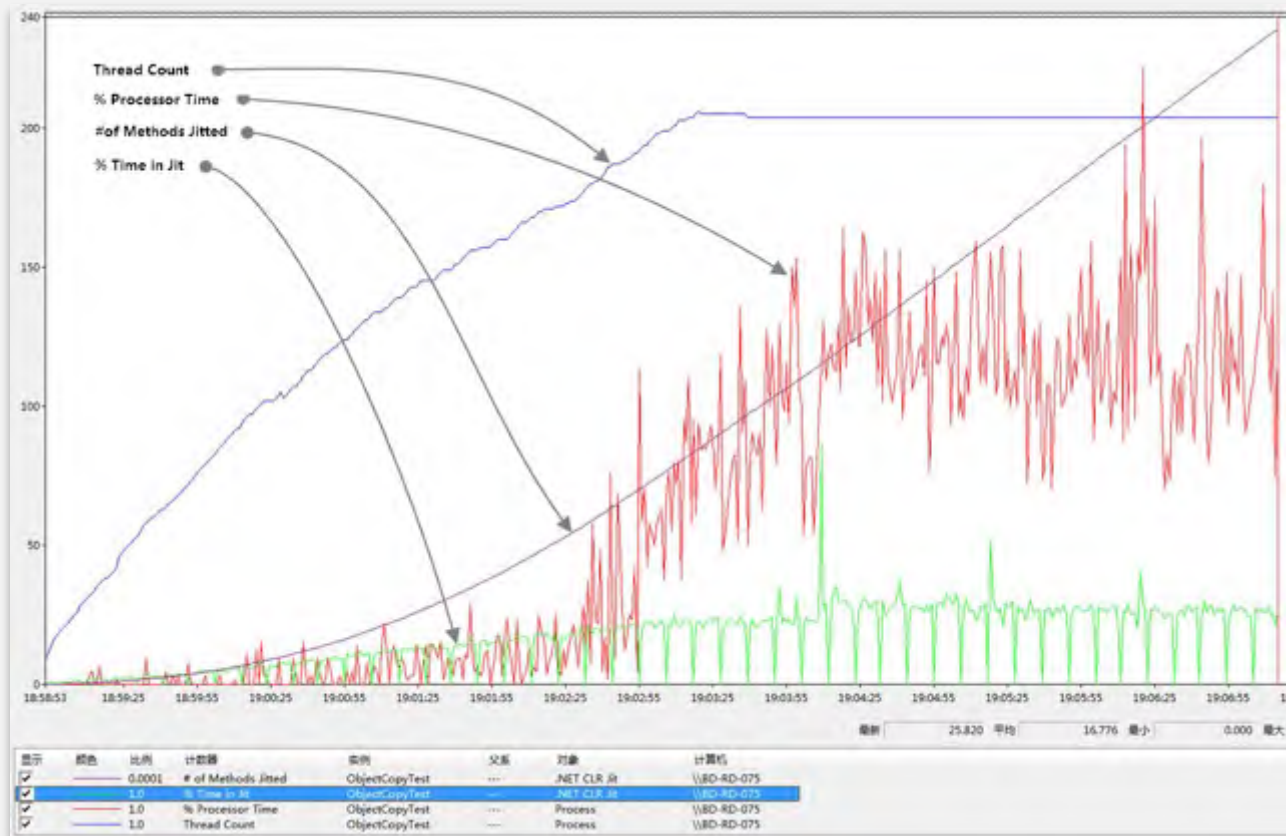


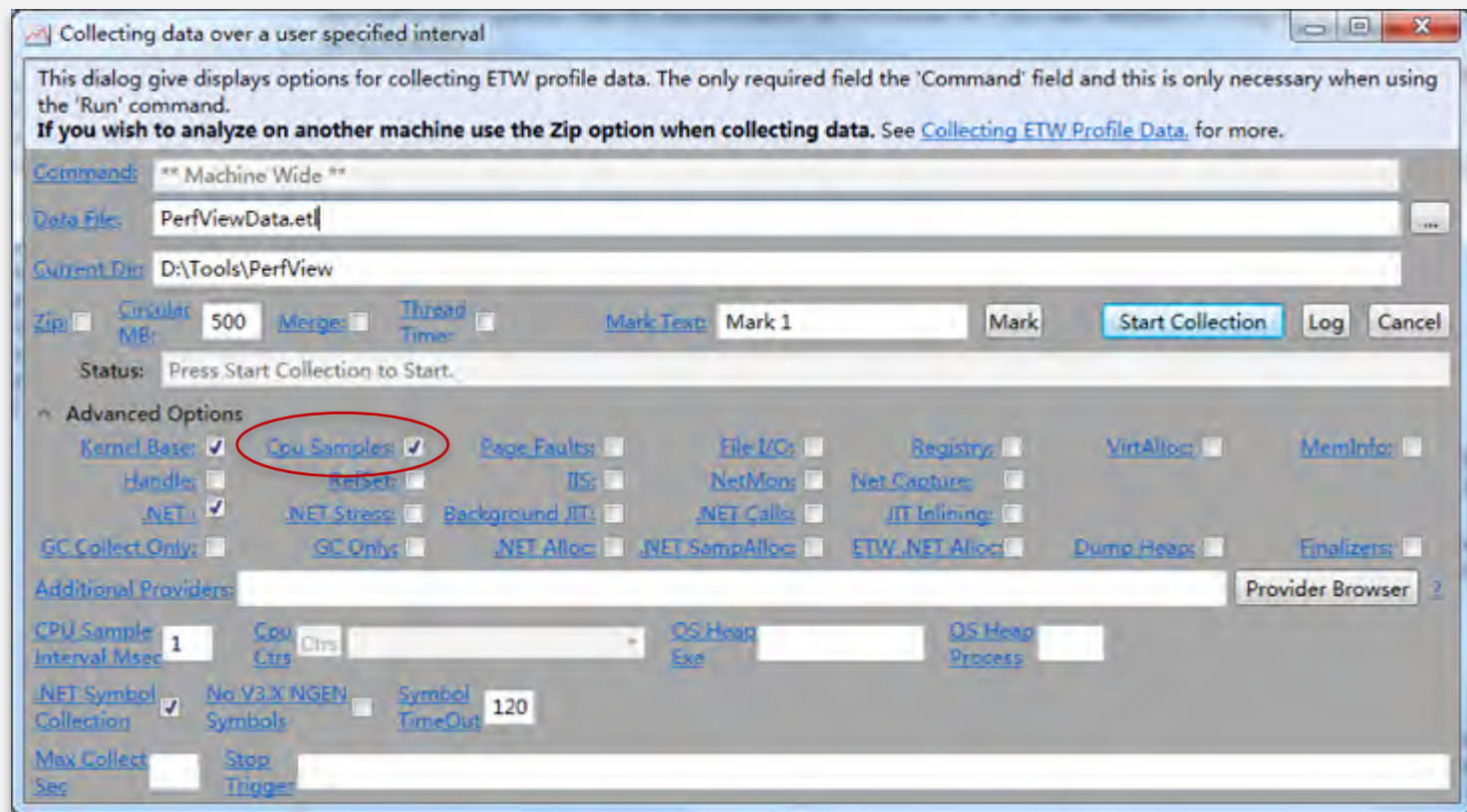
程序日志

VS Profiler, Perfmon, Sysinternals Suite等

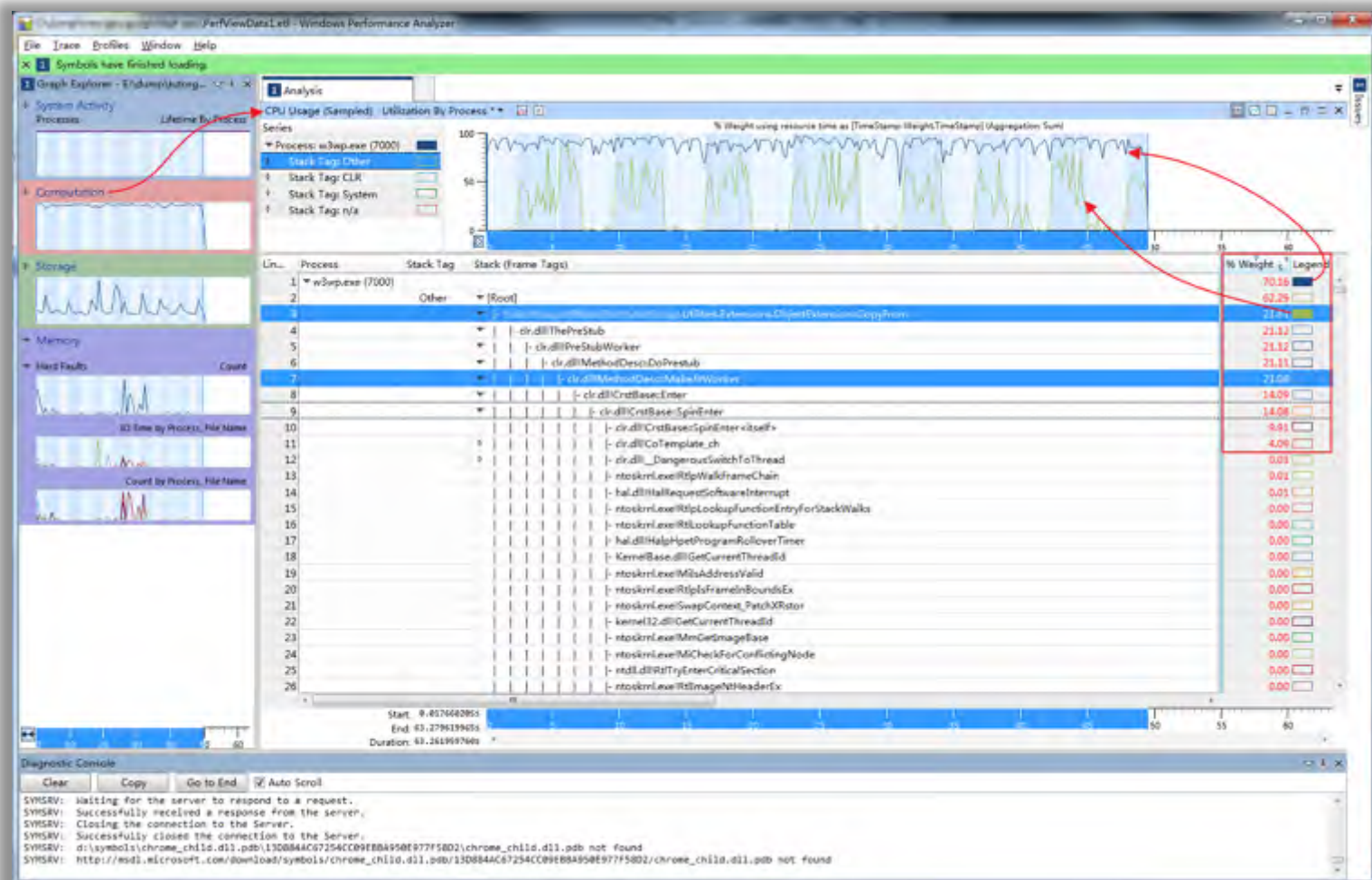
WinDbg, Perfview {ETW}

## 问题分析: 本地测试得到的性能参数





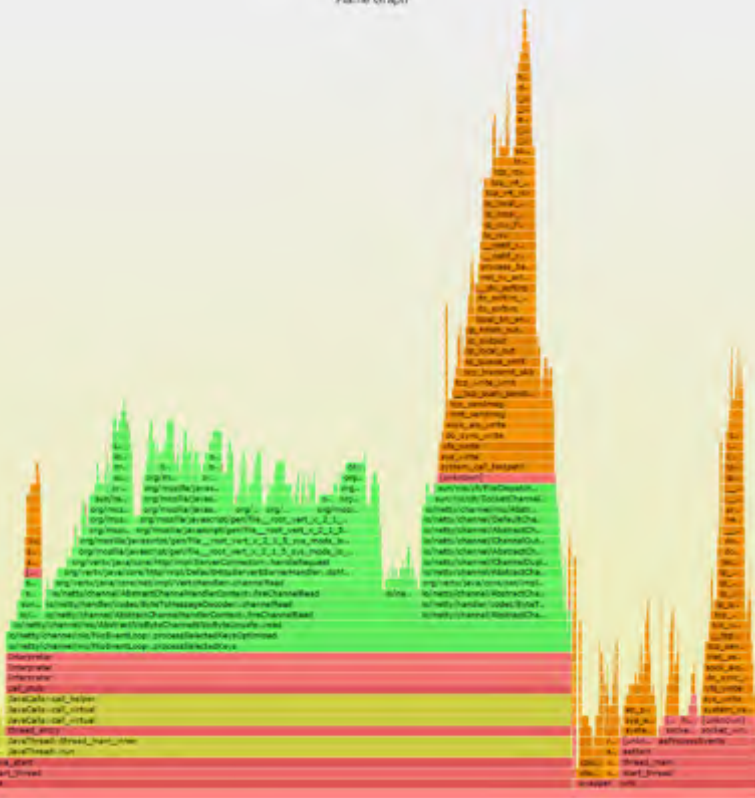
# 问题分析: XPerf查看CPU时间分布



## 如果是Perf...

```
Samples: 17K of event 'cpu-clock', Event count (approx.): 4477250000
Children    Self    Command    Shared Object    Symbol
+ 14.95%    0.15%    java       [kernel.kallsyms] [k] tracesys
+ 12.93%    0.04%    java       perf-3884.map     [.] 0x00007f1858392888
+ 12.81%    0.00%    java       perf-3884.map     [.] 0x00007f1855472dfc
+ 12.75%    0.01%    java       libjava.so        [.] Java_java_io_UnixFileS
+ 11.98%    0.00%    java       libjvm.so         [.] java_start
+ 11.98%    0.00%    java       libpthread-2.17.so [.] start_thread
- 11.78%    0.00%    java       libjvm.so         [.] GangWorker::loop
- GangWorker::loop
- 11.77% ParNewGenTask::work
- 9.94% GenCollectedHeap::gen_process_roots
- 9.64% ConcurrentMarkSweepGeneration::younger_refs_iterate
  CardTableRS::younger_refs_in_space_iterate
  CardTableModRefBS::non_clean_card_iterate_possibly_parallel
- CardTableModRefBS::non_clean_card_iterate_parallel_work
- 9.57% CardTableModRefBS::process_stride
  + 4.92% ClearNoncleanCardWrapper::do_MemRegion
  + 4.26% CardTableModRefBS::process_chunk_boundaries
+ 1.81% ParEvacuateFollowersClosure::do_void
+ 11.77%    0.00%    java       libjvm.so         [.] ParNewGenTask::work
+ 11.01%    0.02%    java       libjava.so        [.] canonicalize
+ 10.96%    1.70%    java       libc-2.17.so      [.] _lxstat64
+ 9.94%    0.00%    java       libjvm.so         [.] GenCollectedHeap::gen
+ 9.64%    0.01%    java       libjvm.so         [.] CardTableModRefBS::non
+ 9.64%    0.00%    java       libjvm.so         [.] CardTableModRefBS::non
+ 9.64%    0.00%    java       libjvm.so         [.] CardTableRS::younger_r
+ 9.64%    0.00%    java       libjvm.so         [.] ConcurrentMarkSweepGen
+ 9.57%    0.13%    java       libjvm.so         [.] CardTableModRefBS::pro
```

Flame Graph



```
36 protected CallInfo CallInfo;
37 protected DynamicMethod
38 protected EmitHelper
39
40 8 references
41 protected BaseEmitter
42
43 23 references
44
45 internal Delegate GetDelegate()
46 {
47     var action = cache.Get( CallInfo );
48     if( action == null )
49     {
50         Method = CreateDynamicMethod();
51         Generator = new EmitHelper( Method.GetILGenerator() );
52         action = CreateDelegate();
53         cache.Insert(CallInfo, action, CacheStrategy.Temporary);
54     }
55     return action;
56 }
```

public void Insert(TKey key, TValue value, CacheStrategy strategy)

{

entries[key] = strategy == CacheStrategy.Temporary

? new WeakReference(value)

: value as object;

}

Method = CreateDynamicMethod();

Generator = new EmitHelper( Method.GetILGenerator() );

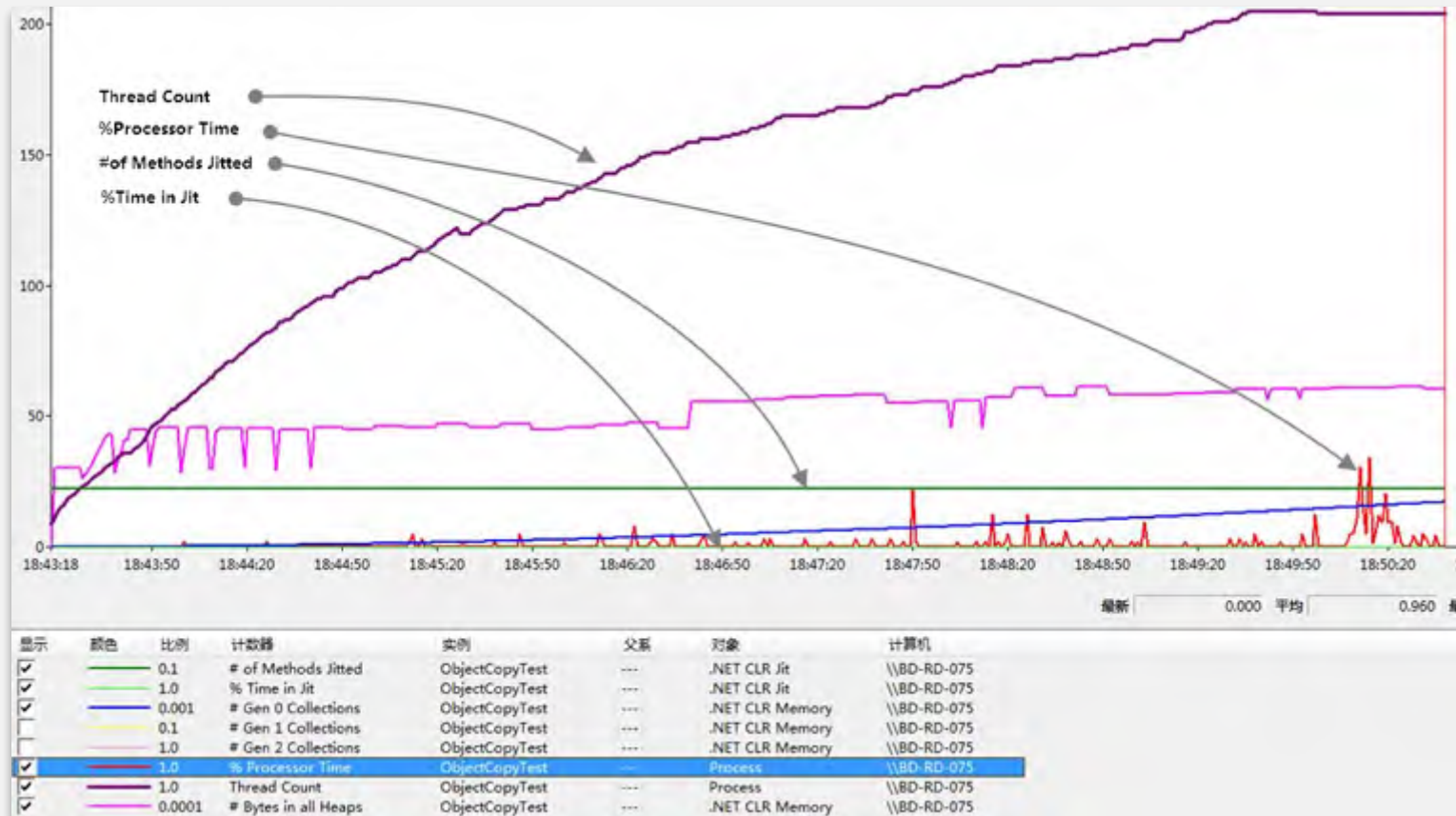
action = CreateDelegate();

cache.Insert(CallInfo, action, CacheStrategy.Temporary);

```

36     protected CallInfo CallInfo;
37     protected DynamicMethod Method;
38     protected EmitHelper Generator;
39
40     + protected BaseEmitter( CallInfo callInfo )...
44
45     - 23 references
46     internal Delegate GetDelegate()
47     {
48         var action = cache.Get( CallInfo );
49         if( action == null )
50         {
51             Method = CreateDynamicMethod();
52             Generator = new EmitHelper( Method.GetILGenerator() );
53             action = CreateDelegate();
54             cache.Insert(CallInfo, action, CacheStrategy.Permanent); //cache.Insert(CallInfo, action, CacheS
55         }
56         return action;
57     }

```



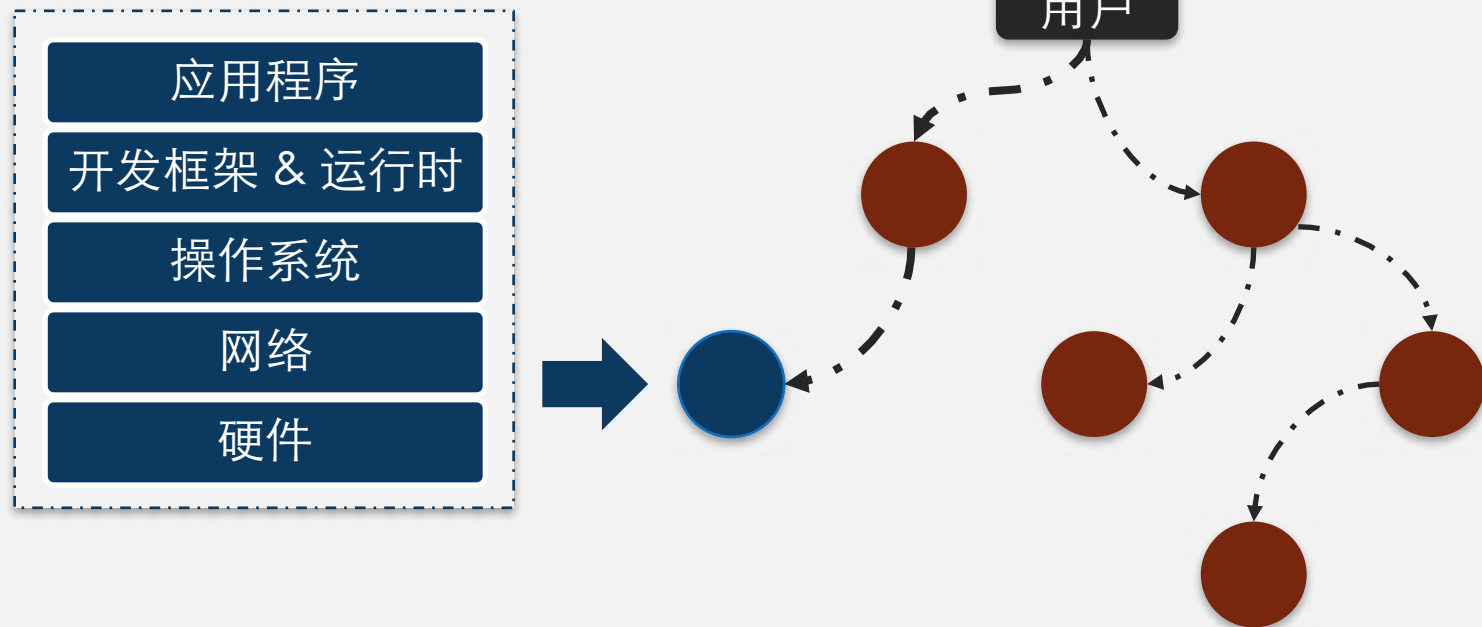
## 挑战

- 重现概率低, 依赖环境.
- 日志分析, 耗时耗力.
- 问题复杂, 牵涉知识点众多.
- 解决方案需要进行验证.



## 应对

- 监控覆盖足够的广度和深度.
- 日志具备多个维度的关联性.
- 合适的**工具**, 如Perf, Perfview等.
- **人**的经验与技术积累.
- 找出重现步骤, 对比测试, 有针对性的进行监控.



我们需要掌控全局...

APMCon

宏观

用户  
/业务

微服务/系统

应用程序实例

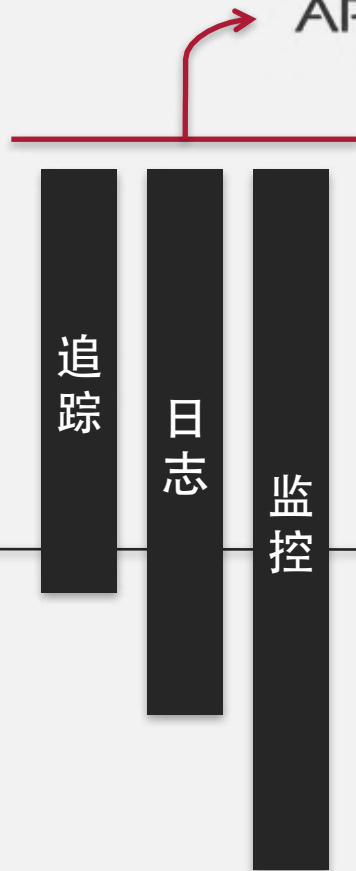
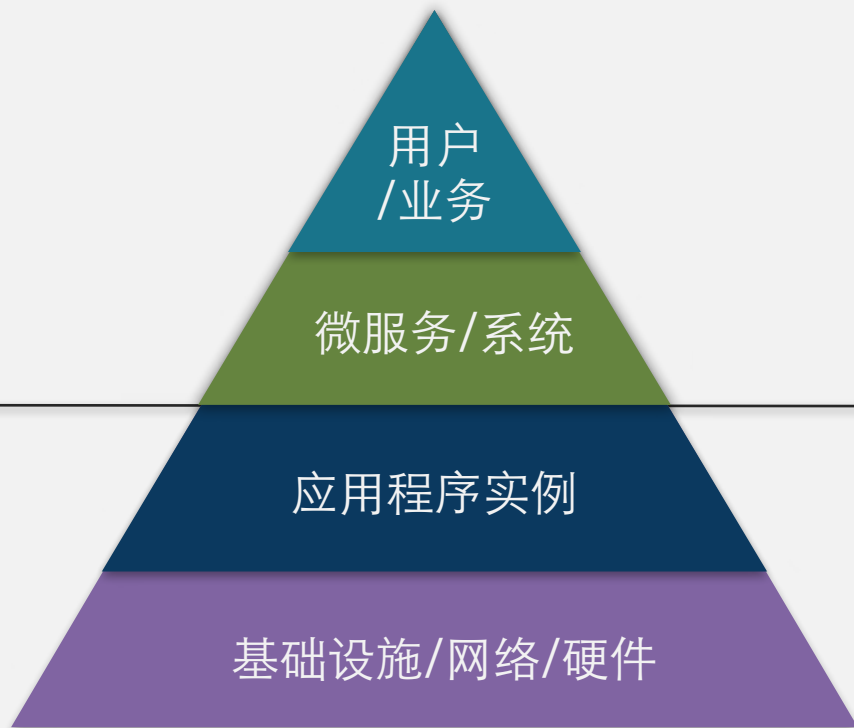
微观

基础设施/网络/硬件

追踪

日志

监控



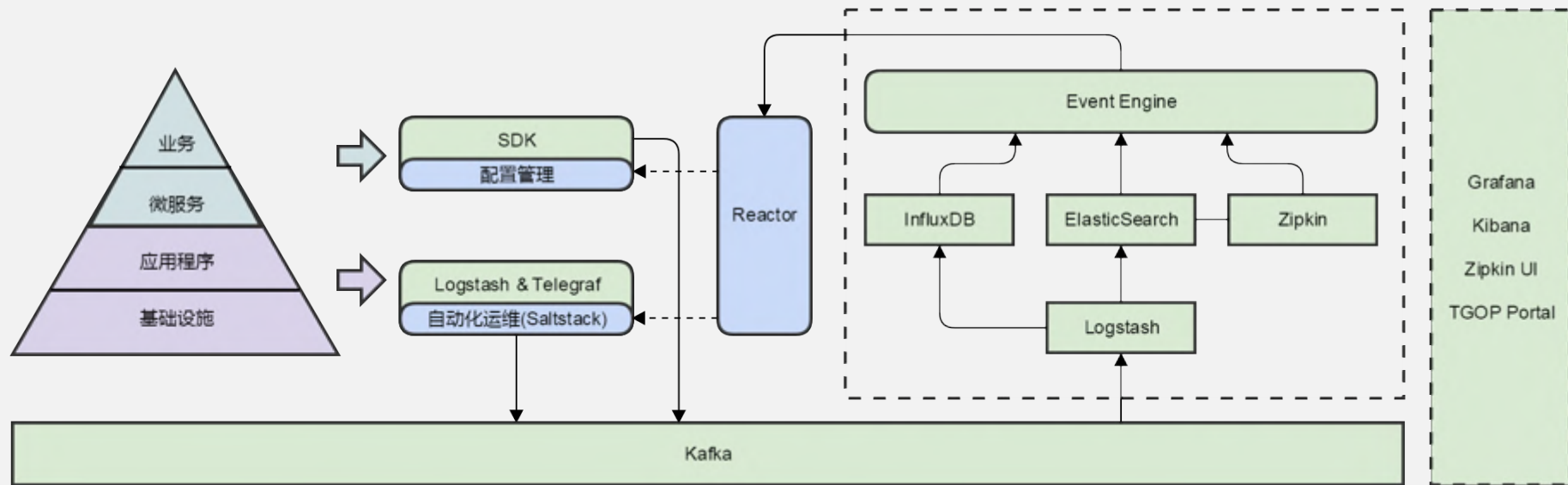
## 业务挑战

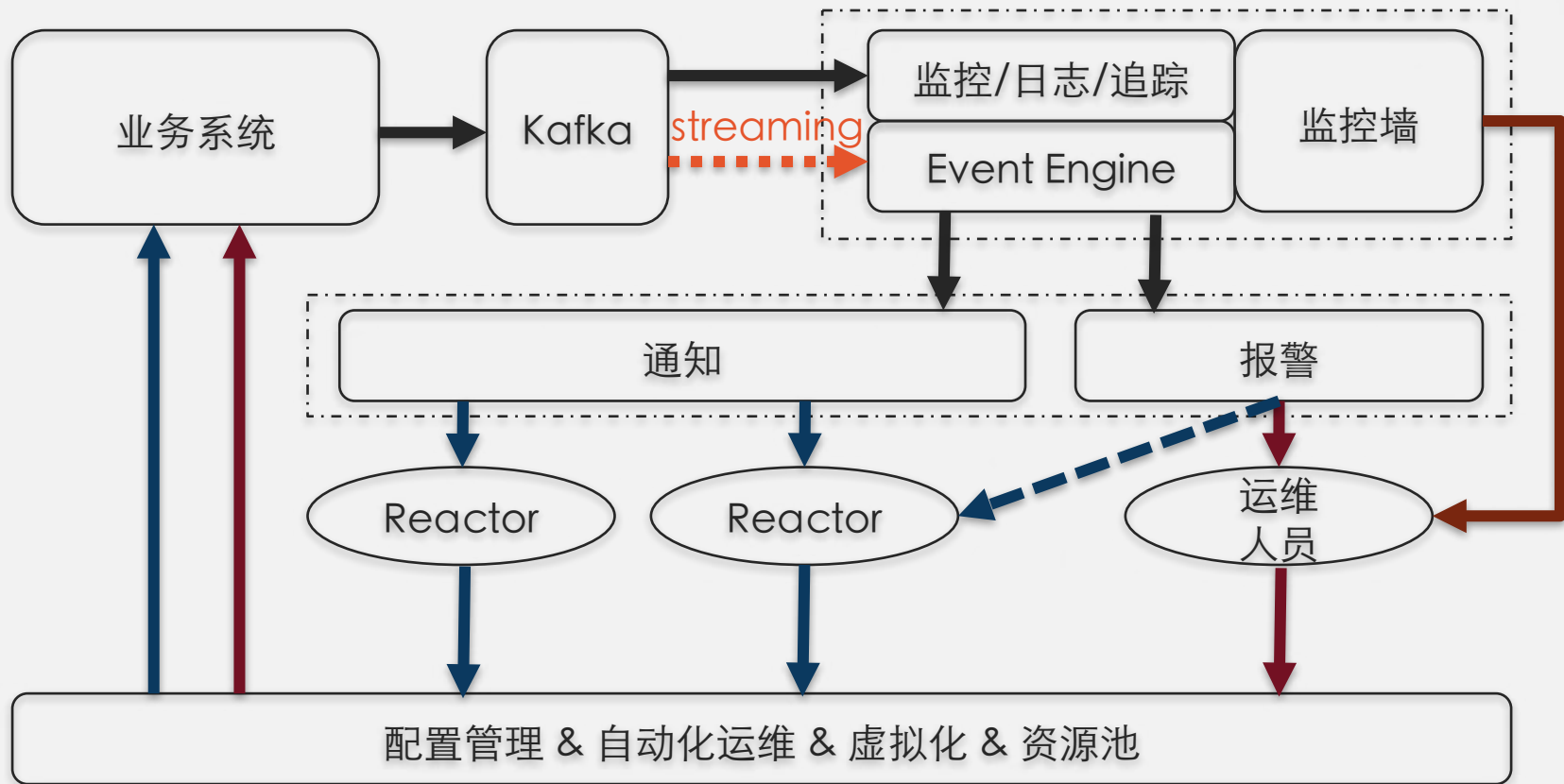
- 遍及全球135个国家和地区的客户群.
- 遍布全球80多个国家, 100多座城市的超过15,000名外籍顾问.
- 每年提供超过1000万堂在线课程.
- 为高质量客户群体提供高质量服务.

## 技术挑战

- 技术转型, 业务解耦.
- 多语言开发, 多平台部署.
- 遗留系统众多, 依赖关系复杂.

目标: 全系统APM驱动





## 数据整合

---

展示与追踪  
链关联的日志, 监  
控和报警数据, 以  
便于问题排查.

## 系统全景

---

实时展示各  
个服务节点的状态, 以及相互依赖  
关系.

## 动态采样

---

根据宿主机  
(以及调用链上游  
服务器)的负载调  
整采样率.





- 监控, 日志和追踪数据要具备关联性, 便于后续分析.
- 尽早建立CMDB, 方便对被监控的服务器和服务做分类, 打标签.
- 重SDK, 去中心化.
- InfluxDB适合存储以数字为主的记录, 文本较多的记录存入ES.
- InfluxDB需要设置Series上限, 防止服务器内存耗尽.

- Grafana报警功能使用单独的实例来运行.
- Grafana数据库Annotation表需要定期清理, 当数据量累积到30万以上时, 会严重影响页面性能.
- 追踪系统无法跨越消息队列, 可以在发送消息前将TraceID/SpanID信息植入消息内.
- 追踪系统SDK一般支持跨线程追踪, 但是对Timer线程需要清除追踪上下文.
- 追踪系统基于采样率模式, 所以无法指定追踪特定事件(比如登陆失败), 需要调研基于事件的追踪模式.

服务实例  
1000+

数据中心  
3

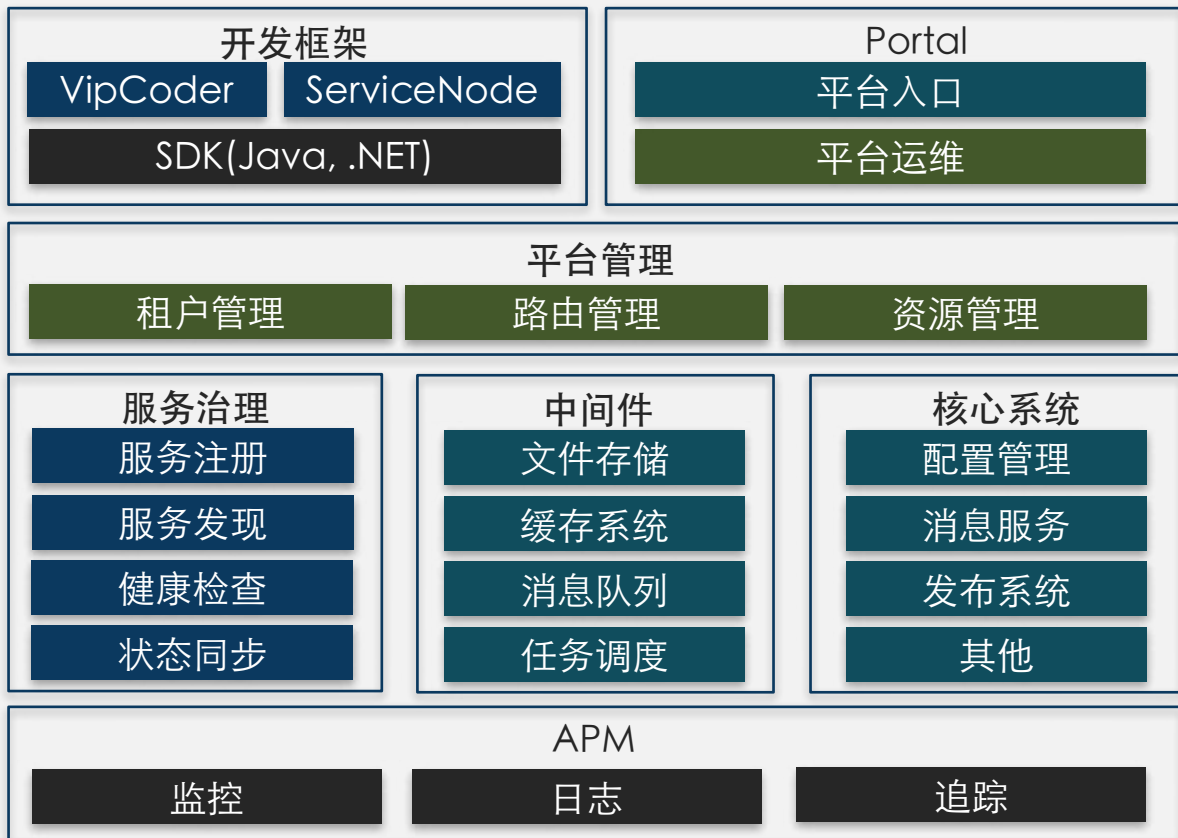
监控指标  
100+

日志  
500+ GB/天

每秒查询  
50+

每秒写入(单节点)  
10,000+

服务器  
600+





- 改进追踪系统UI, 增强与日志和监控(报警)系统的集成体验.
- 基于追踪系统, 实现分业务场景监控.
- 基于大数据进行未来24小时流量预测, 做到主动预警而非被动告警.
- 应用画像, 以利精准部署.
- 资源池化, 提高利用率.

未有监控, 不谈优化.

合适即可, 演化迭代.

工具重要, 人更重要.

少即是多, 见微知著.

杨大鹏

原微软工程师, 2015年加入tutorabc,  
带队负责TGOP平台设计, 实现, 运维与推广.  
熟悉CLR/JVM虚拟机,  
擅长Windows/Linux平台下应用性能调优.

