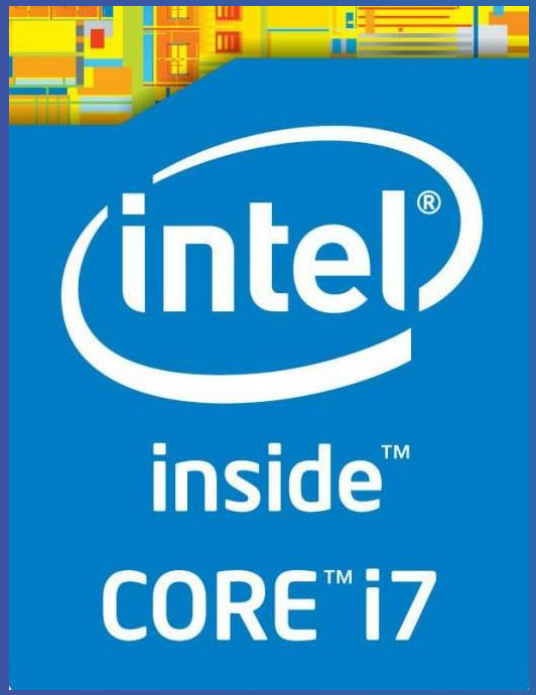
客户端性能测试的方案和优化

安全卫士QA

李进

目录

前言：配置升级，性能优化是否仍然有重要价值



千里之堤毁于蚁穴

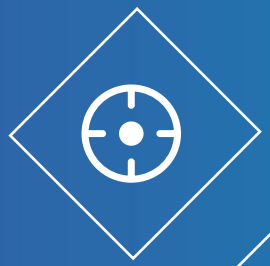
现实远比想象残酷

性能和流量的直接关系

性能优化遇到的困难和应对

性能瓶颈定位困难

使用性能分析工具，开拓定位思路



性能指标的量化

资源类性能指标使用通用工具

交互类性能指标使用自动化工具

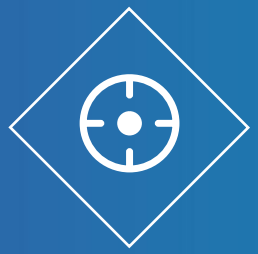
优化方案的寻找和取舍

从技术、产品层面双重优化，分析竞品

人力，时间不足

寻求自动化、日常化

01



性能瓶颈定位困难

使用性能分析工具，开拓定位思路

性能瓶颈定位之工具的使用——主流工具的使用

工具的强项

- 测量误差小，精确到毫秒以下
- 无开发成本，学习成本低
- 能精确到具体函数
- 统计功能强，输出报告详尽

注意事项

- 结果需要人工过滤分析
- 工具各有所长，需要配合使用
- 需要结合业务逻辑
- 一些工具观察者效应比较明显

性能瓶颈定位之思路——放大数据，对比寻找瓶颈

小数据场景 → 大数据场景 → 对比性能指标 → 重复验证

案例1

相同环境下，20M文件启动

exe名称	窗口弹出耗时(秒)							
	1	2	3	4	5	6	7	8
SoftMgr.exe	2.5340	1.7670	1.8240	1.8190	1.8330	1.7990	1.8310	1.8260

40K文件启动，注意第一次启动时间明显减少

exe名称	窗口弹出耗时(秒)							
	1	2	3	4	5	6	7	8
SoftMgr.exe	2.1240	1.7850	1.7810	1.7570	1.7760	1.7090	1.7570	1.7660

案例2

1、正常场景下，获取搜索页面打开时间

2、增加服务器返回数据，证页面耗时增加

3、减少服务器返回数据，证页面耗时减少

4、确定瓶颈位置，提出方案，提交开发修改

02

性能指标的量化

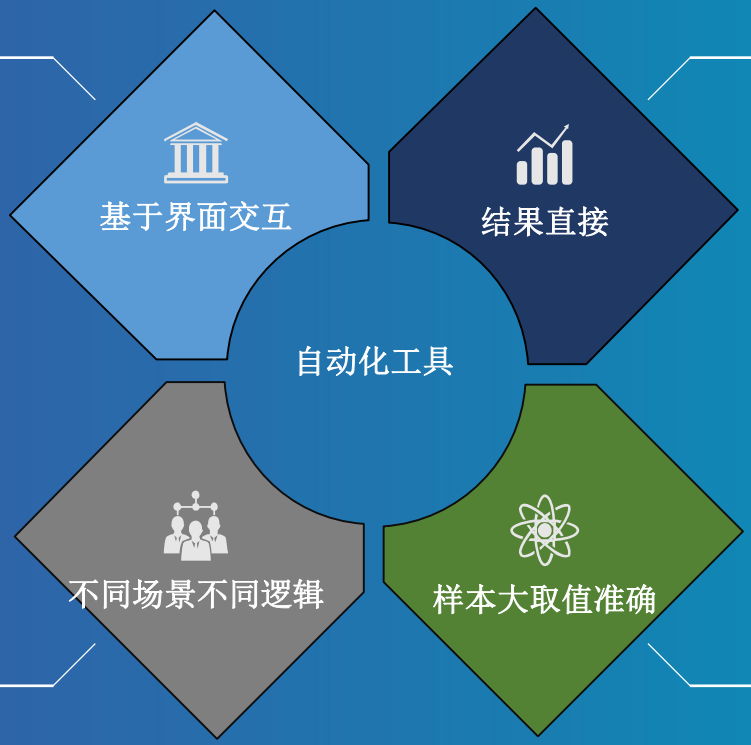
资源类性能指标使用通用工具

交互类性能指标使用自动化工具

交互类的性能指标的测量—定制自动化工具

通过脚本形式，监控界面元素，测量维度是用户真实体验

结果中除了具体数字外，还结算出具体性能得分，一目了然



冷启动、热启动，不同硬件环境下区分配置

脚本自动执行，可以多次取样求均值，减少了脚本测量带来的误差

自动化工具遇到的一些问题和解决方式

自动化脚本无法模拟冷启动情况	脚本中调用工具，清除文件系统缓存
测量场景有限，无法覆盖外网	布置多种环境，硬件，软件差异化
运行结果有误差	多次运行求平均值
程序运行过程中环境可能变动	每次运行前自动清理环境
不同环境上性能参考值不同	不同环境不同配置

03

优化方案的寻找和取舍

从技术、产品层面双重优化，分析竞品寻找思路

性能优化方案的三种方式

产品层面

产品层面进行调整，增加交互或者牺牲部分非必要功能

技术层面

工具定位，结合代码逻辑，函数级优化简单直接

竞品分析思路

通过分析竞品寻找思路，验证思路正确性

如何从产品层面和技术层面进行优化的案例

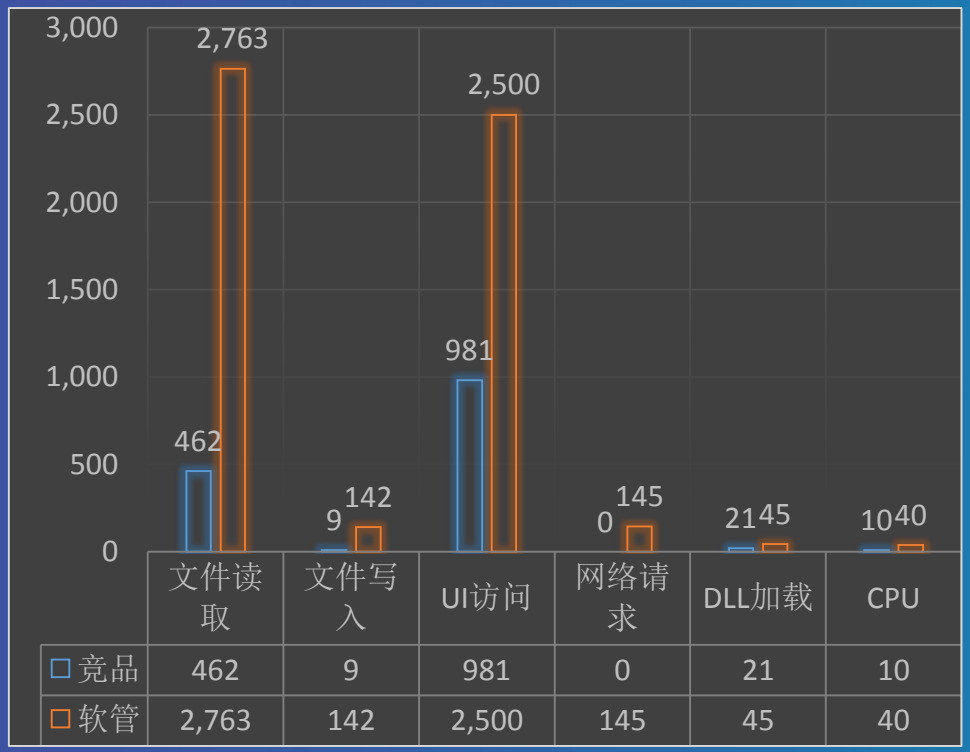
产品层面

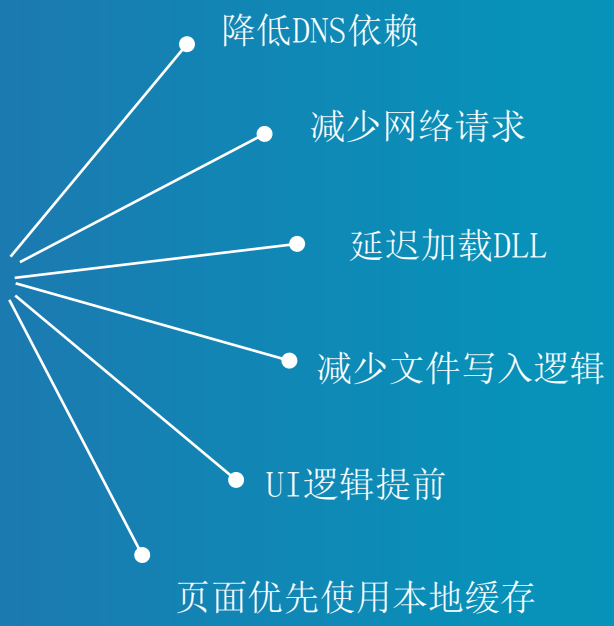
- 进程启动界面增加动画
- 非首页主要逻辑由用户点击触发
- 遍历操作减少不必要的有效性判断，减少IO
- 减少dll加载，相关功能通过其他方式实现

技术层面

- 减少请求量，精简服务端接口返回的冗余数据
- 减少启动过程中文件写操作，写操作延后
- 可信任文件减少签名校验
- 控制日志文件尺寸，延后读取逻辑

竞品分析的结果和带来的优化思路





04

人力，时间不足

寻求自动化、日常化、流程化

性能测试自动化 - 自动化用例增加性能监控

优势和问题

- 批量执行，全平台覆盖
- 特殊场景构建一劳永逸
- 覆盖用户所有核心操作
- 硬件、软件环境比较单一
- 测量的数值不够精准，只能做阈值限制

性能测试日常化 – 性能得分评估工具

窗口性能检测结果

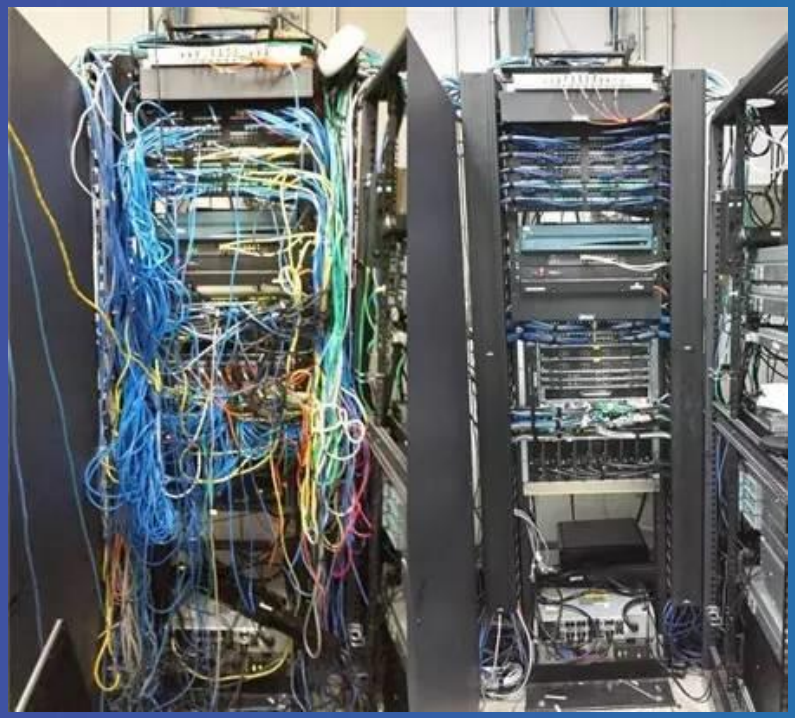
测试项目	测试结果										
	1	2	3	4	5	6	7	8	9	10	平均值
upgrade	3.6190s	3.5260s	3.5720s	3.6190s	3.5570s	3.6350s	3.6030s	3.8380s	3.5730s	3.5730s	3.6115s
main_init_time_sub	2.0313s	2.0340s	2.0395s	2.0357s	2.0541s	2.0402s	2.0145s	2.0381s	2.0446s	2.0703s	2.0402s
SoftMgr_CPU	100.0000%	100.0000%	100.0000%	100.0000%	70.5000%	72.6000%	100.0000%	100.0000%	96.8000%	无效	93.3222%
SoftMgr_RAM	2516.0000 K	39316.0000 K	46988.0000 K	47264.0000 K	49416.0000 K	49468.0000 K	49416.0000 K	49416.0000 K	39824.0000 K	40368.0000 K	41399.2000K
uninstall	3.9150s	3.6350s	3.8220s	3.9630s	3.6190s	3.6510s	3.6190s	3.8690s	3.7440s	3.7760s	3.7613s
SoftMgr.exe	2.8860s	2.8390s	2.7920s	2.8390s	2.8240s	2.8860s	2.9790s	2.8550s	2.8550s	3.0420s	2.8797s
baoku	2.9320s	2.9170s	3.1520s	2.9330s	2.9170s	2.9020s	3.1980s	2.8540s	2.9790s	2.9490s	2.9733s

测试项目	测试结果
总得分	52.414054594
单项得分	102.405998707

总得分=∑单项得分*权重

单项得分 = （合格数值-实际数值+理想数值）/合格数值*百分比

性能测试流程化 – 从上到下的性能管控



静态代码检查

自动化用例增加性能监控

性能评估工具

常规测试中的性能意识

客户端性能测试的愿景

性能全覆盖

性能测试覆盖全部资源类指数、交互类指数

监控可视化

性能监控可视化，形成优化->监控->优化良性循环

自动，日常，流程

完善的自动化体系，快捷有效的测试和流程覆盖

性能文化

项目内传播性能文化，人人都能重视性能问题

一些小心得

客户端性能优化的特点、坑、以及思路

性能结果依赖软硬件环境

网络问题不能忽视

不能放过细枝末节

优化逻辑远比优化代码有效

技术以外的tips

性能优化是个复杂工程

获得上层支持

与产品和开发的博弈

科学精神

谢谢