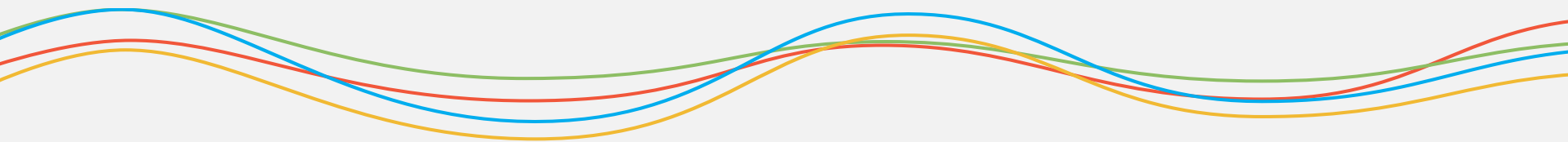


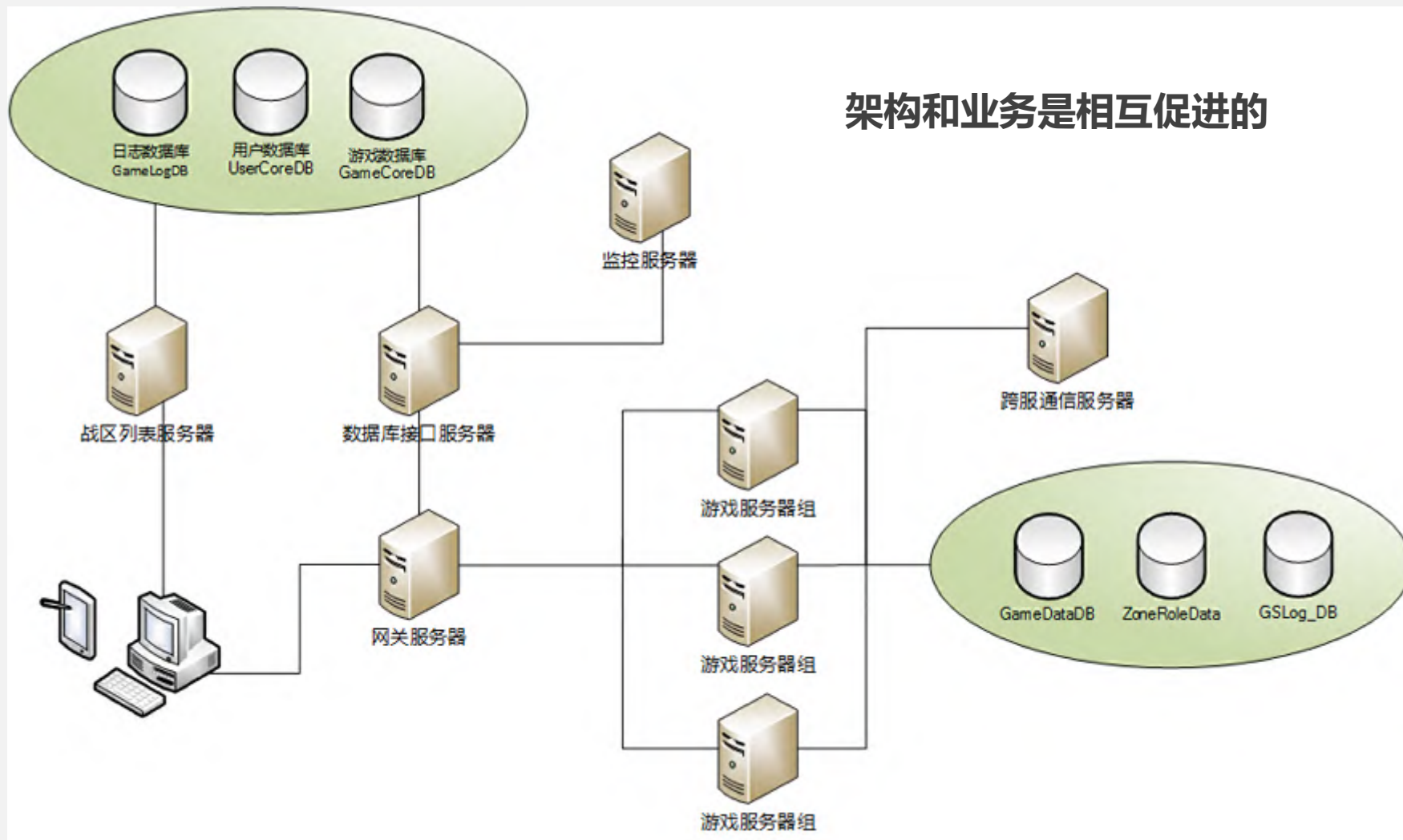
服务器监控及性能优化

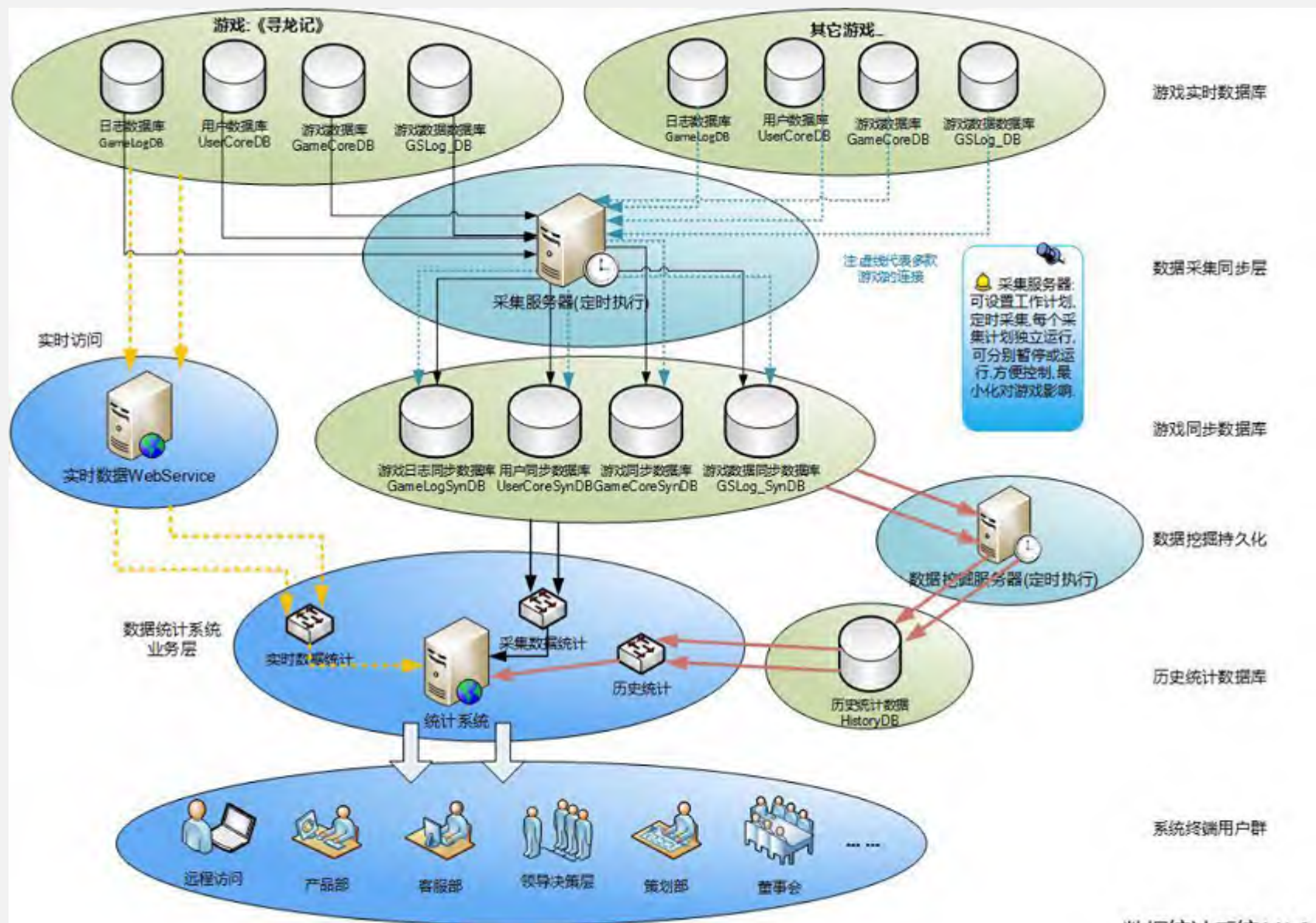
田博辉

2016.8



- **MMO游戏的常用架构**
- **服务器系统及应用健康监控体系**
- **游戏内常用的效率分析及对应的优化手段**
- **与其他互联网产品的互通性思考**
- **Q&A 环节**



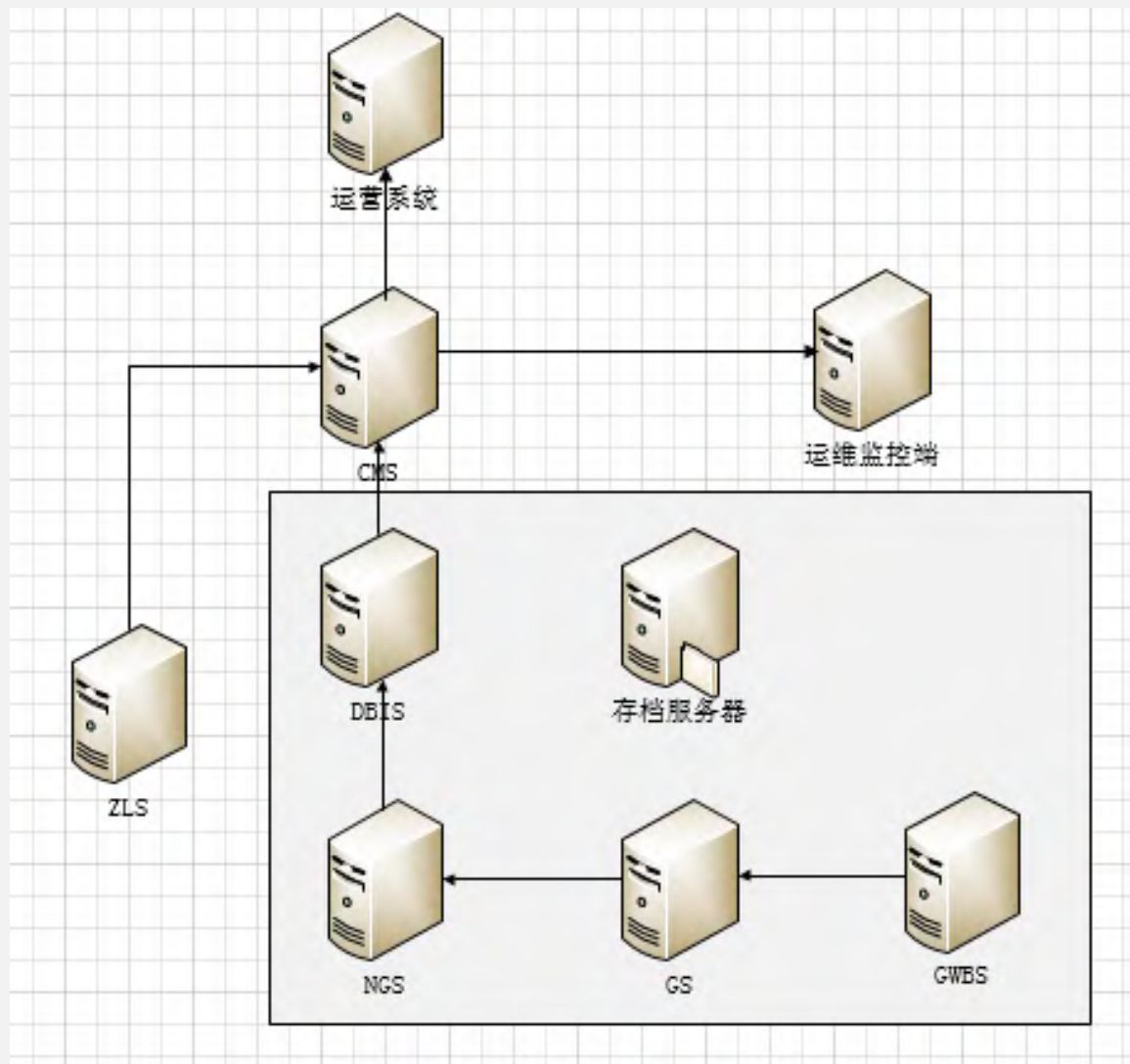


监控信息汇总到CMS

每个服务器定时汇报自身

各个指标信息

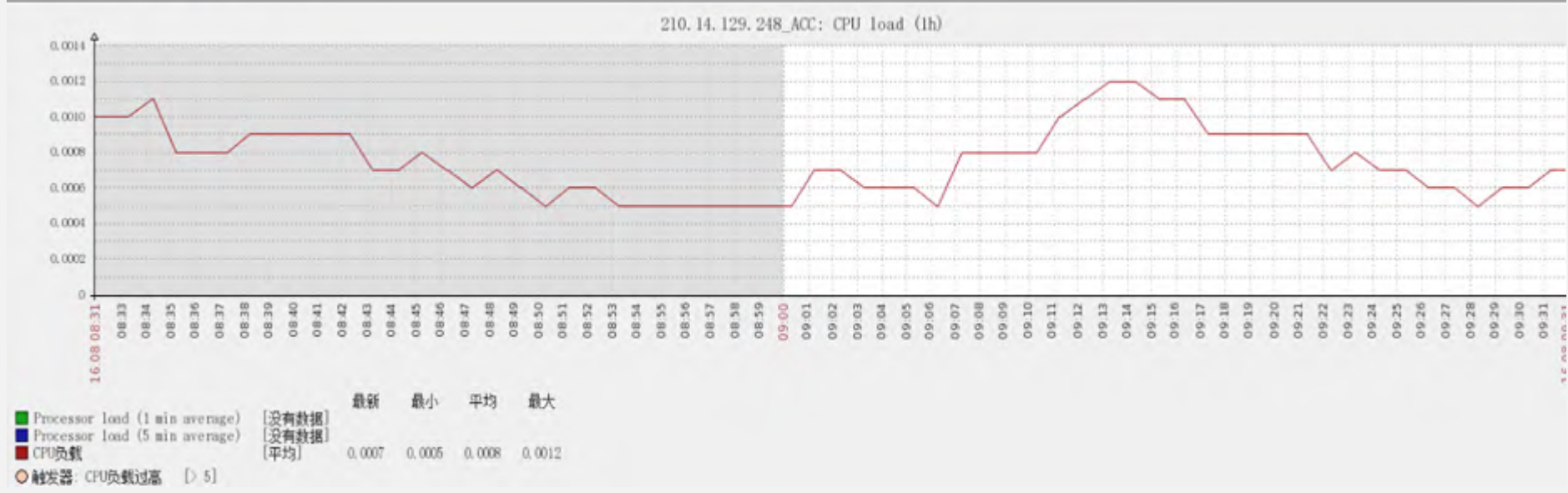
运营系统记录汇总绘图



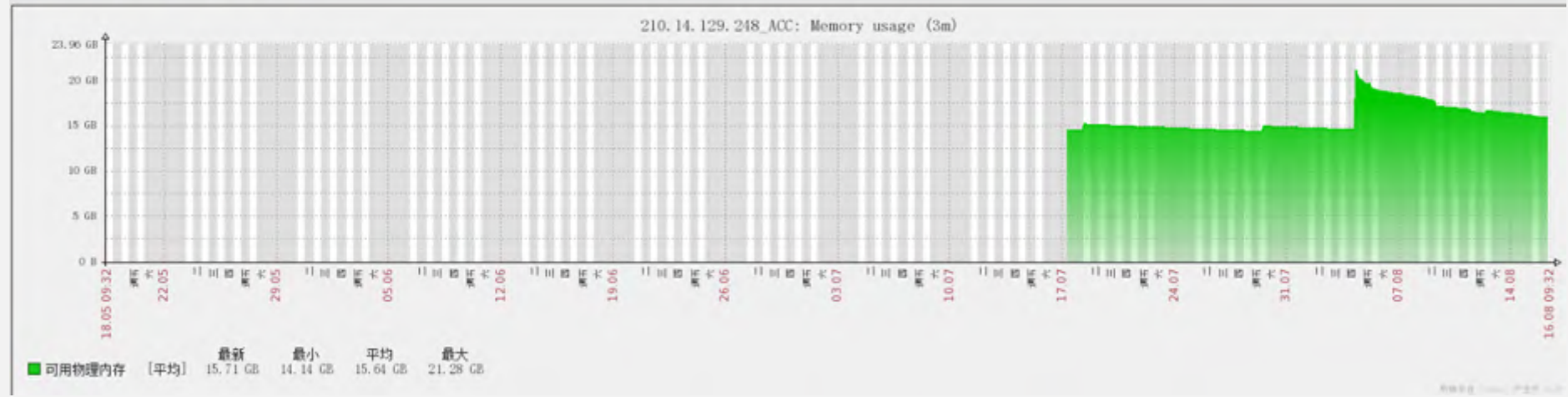
硬件监控项



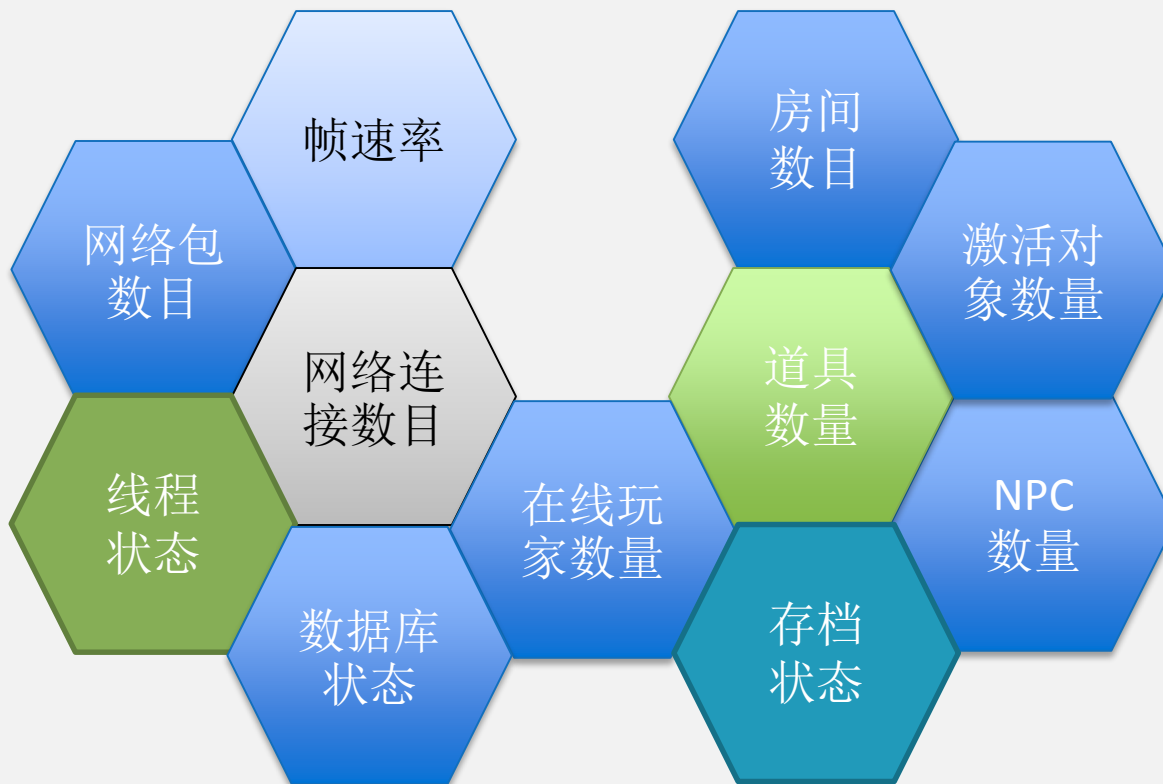
1m 7d 1d 12h 1h | 1h 12h 1d 7d 1m >> 1h (图)



1m 7d 1d 12h 1h | 1h 12h 1d 7d 1m >> 1m (图)

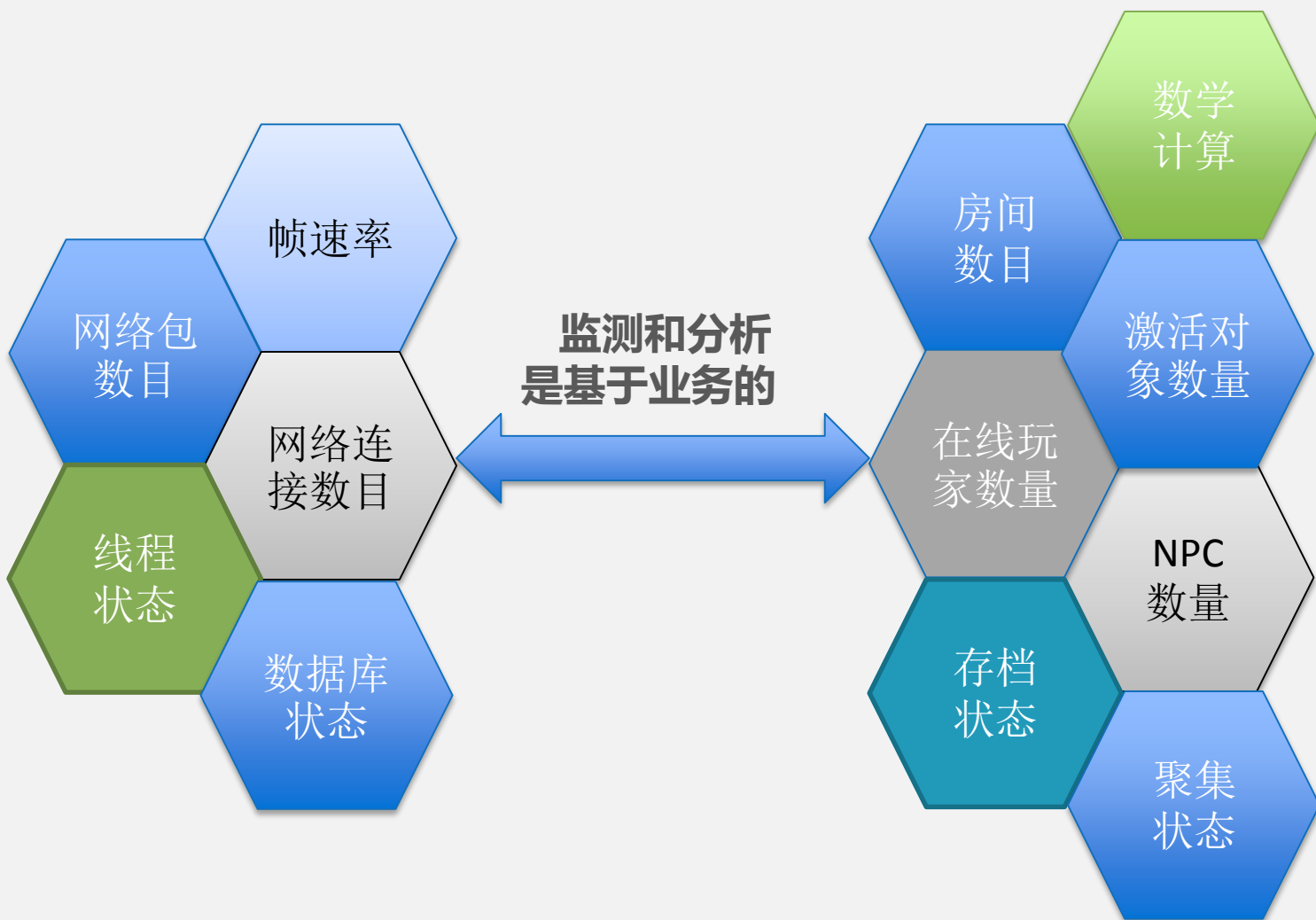


软件监控项





健康监控体系报警，然后呢？ ---分析



静态

服务器异常信息

服务器名称	帧数率	在线...	登录...	状态
[-] 【战区】洪荒之巅				正常
[-] 【GWBS】	333.00			正常
[-] 【DBIS】 192.168.116.205	97.00			正常
[-] 【NGS】 金线(210.14.135.205)	65.00	4	4	正常
[-] 【GS】 金线(192.168.116.205)	4.93	4		正常
[-] 【NGS】 活动线(210.14.135.205)	65.00	0	0	正常
[-] 【GS】 活动线(192.168.116.205)	4.93	0		正常
【DBIS】 192.168.116.205	97.00			正常
[-] 【战区】金蛇狂舞				正常
[-] 【ZLS】 210.14.135.200	32.00			正常
[-] 【ZLS】 210.14.135.200	32.00			正常
[-] 【GWBS】	333.00			正常
[-] 【DBIS】 192.168.116.200	97.00			正常
[-] 【NGS】 金线(210.14.135.200)	65.00	9	9	正常
[-] 【GS】 金线(192.168.116.200)	4.93	9		正常
[-] 【DBIS】 192.168.116.200	97.00			正常
[-] 【NGS】 活动线(210.14.135.200)	65.00	0	0	正常
[-] 【GS】 活动线(192.168.116.200)	4.93	0		正常

游戏功能异常报警

写在之前

**对于在运行系统，优化可能牵一发而动全身，
尽快利用各种手段解决问题，保证项目运行。**

● 逻辑帧速率优化 (尽量控制150ms)

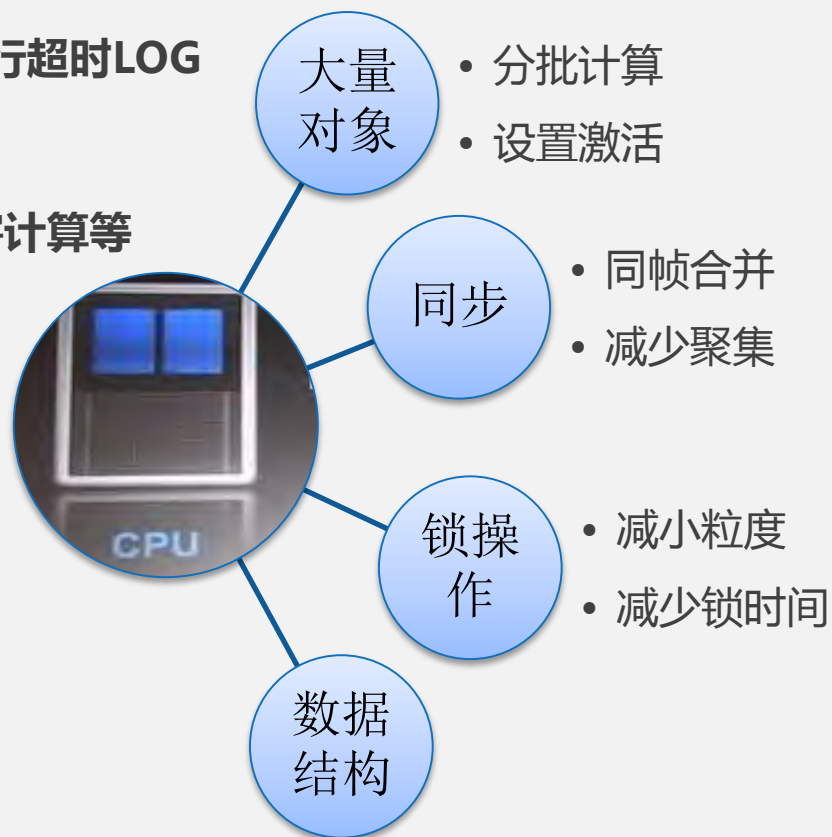
----找到最耗时的函数，内嵌检测，运行超时LOG

对象数量过多，大量道具，NPC等

数学计算过多，位置计算，子弹碰撞，伤害计算等

异常聚集，不可控的玩家行为

跨线程访问，不合理的线程粒度



- 大量的网络包优化

 - 找到发送最多的包，流量统计，LOG记录

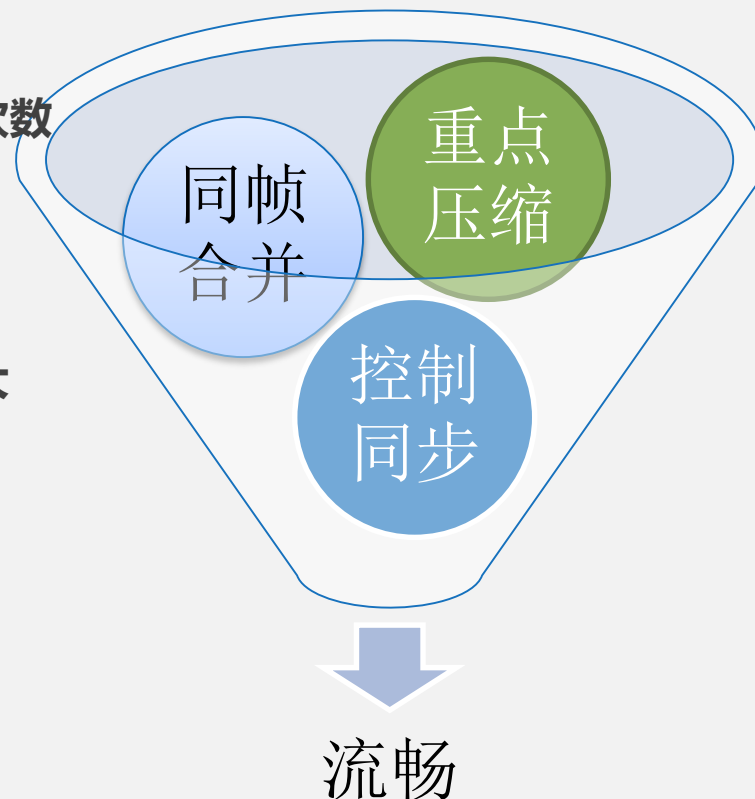
同步的消息在同逻辑帧合并发送，减少投递次数

大量的网络IO重点优化包

MMO的大量包产生在同步，控制范围

使用内存池，大量小内存的申请释放消耗很大

异常来回发送等逻辑BUG



● 网络链接优化

创建链接开销大，使用网络连接池解决

开服、积分墙刷广告，从设计上支持动态增加网关服务器解决

撞库等异常的网络攻击，及时彻底释放，封IP解决



- 线程操作优化

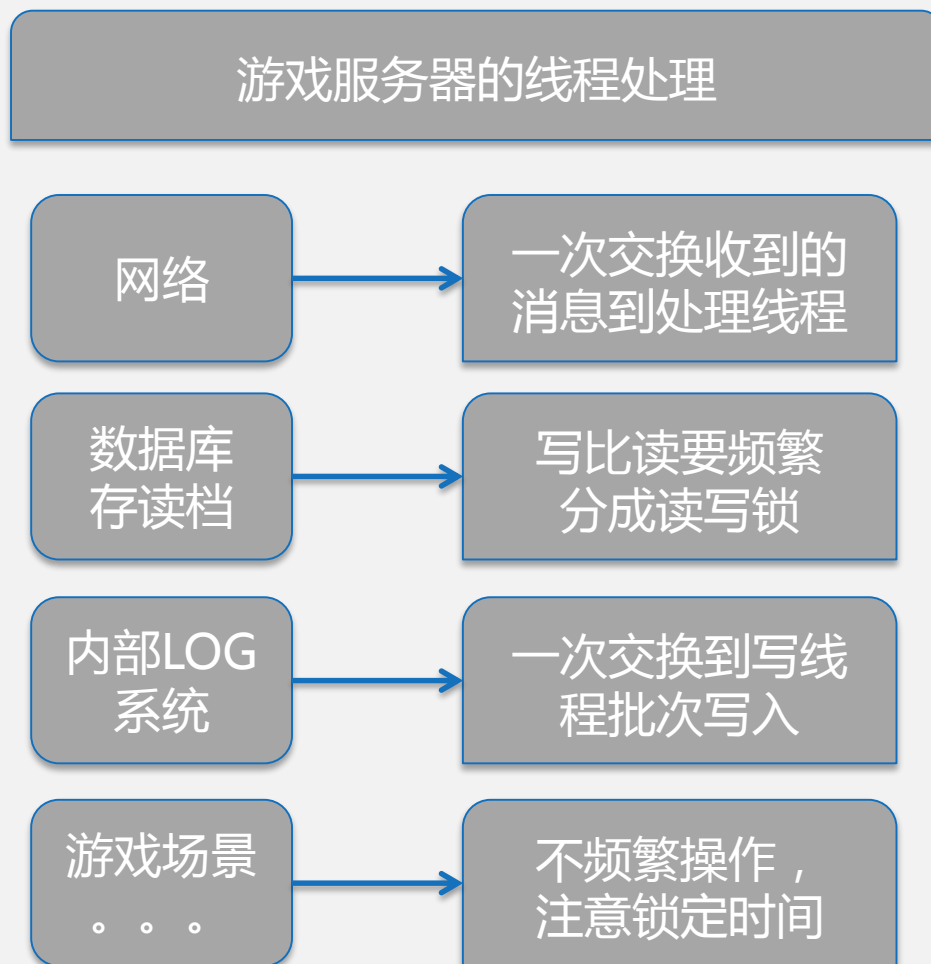
-----尽量减少锁的时间

尽可能的少调用锁

减小锁粒度

线程数控制，线程间切换开销

利用析构自解锁，防止死锁



- 存档数据库操作优化
 - 尽量保证不回档

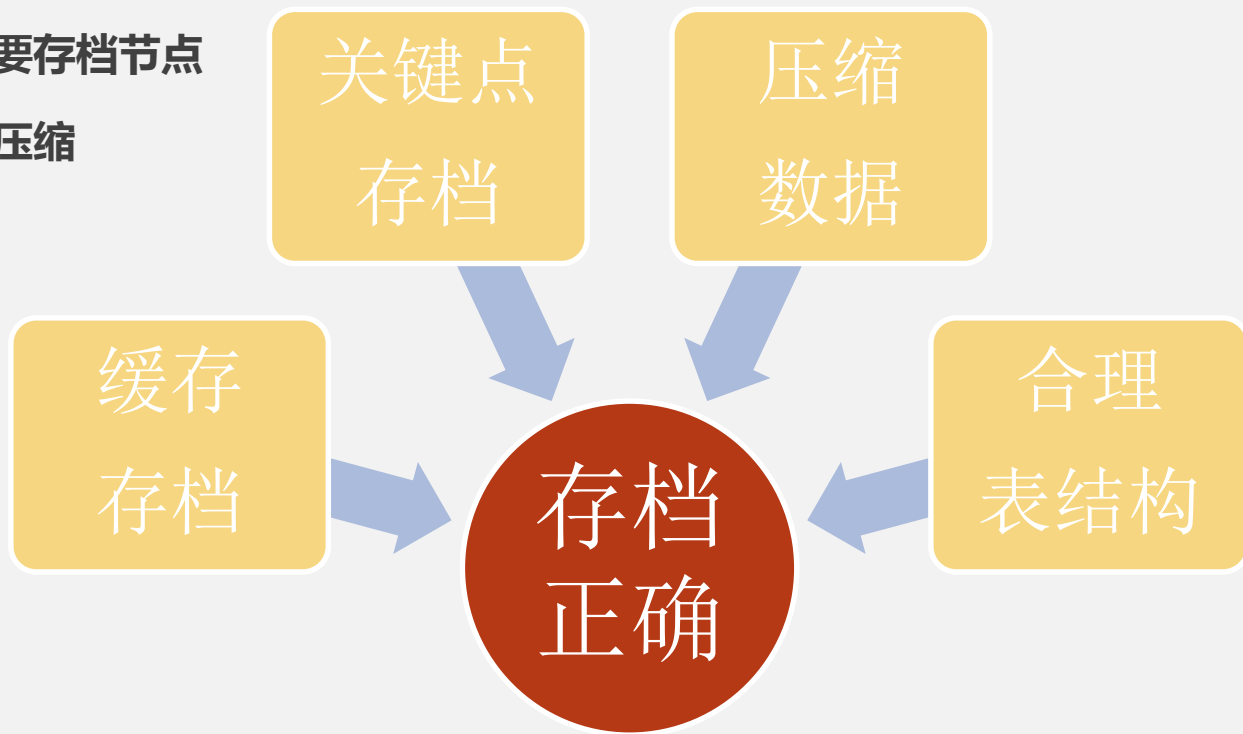
设计存读档缓冲，减少直接对数据操作

增加存档频率，设定重要存档节点

控制存档数据大小，可压缩

数据表设计合理

按战区分存档库



● 内存优化

单个对象的内存占用尽量少，比如使用标记位

频繁申请释放的对象使用对象池，消息，道具，子弹，NPC等

碎内存控制，长时间运行后会积累

重写new delete，用于统计和分析效率点和泄露

根据功能分多进程



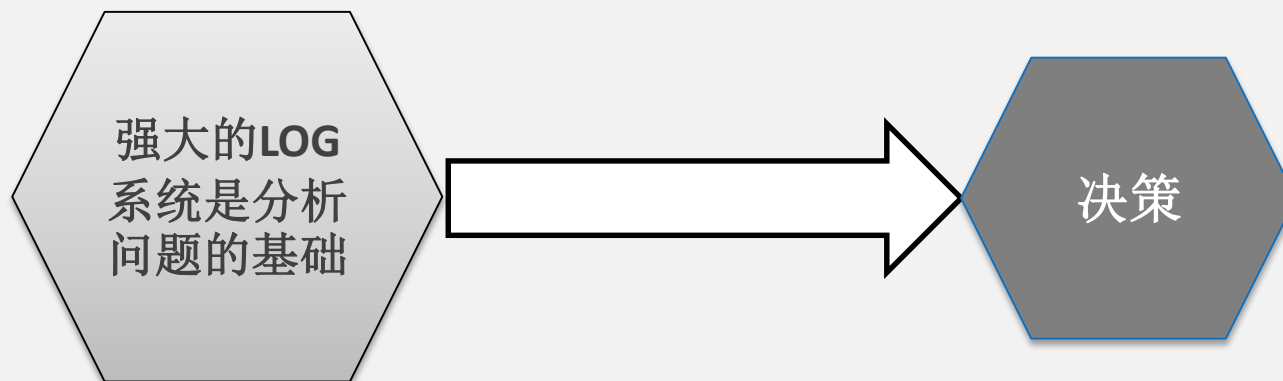
● LOG系统优化

计算极限，各个相关设计写入读取频率和数量

大量LOG写入，分批次持续写入

LOG系统分级，控制写入阈值

统计系统同步，时间点选择



- 调用第三方API的优化

- 不是说第三方API不靠谱，只是考虑全面尽量少受影响

- 除非必要，否则不必须全信任并等待

- 开辟专门的线程或者服务等用于第三方API调用，并设置长度

- 第三方调用、超时、失败要统计

保证自身系统受第三方影响降到最低

- 和业务上配合的优化

在线、存留的要求前期的新手村多出生点

虚假繁荣点的设定

活动修改,错峰进行



高性能的设计是一种意识，而不单单是功能上的实现

架构上设计的高效和可扩展性是基础

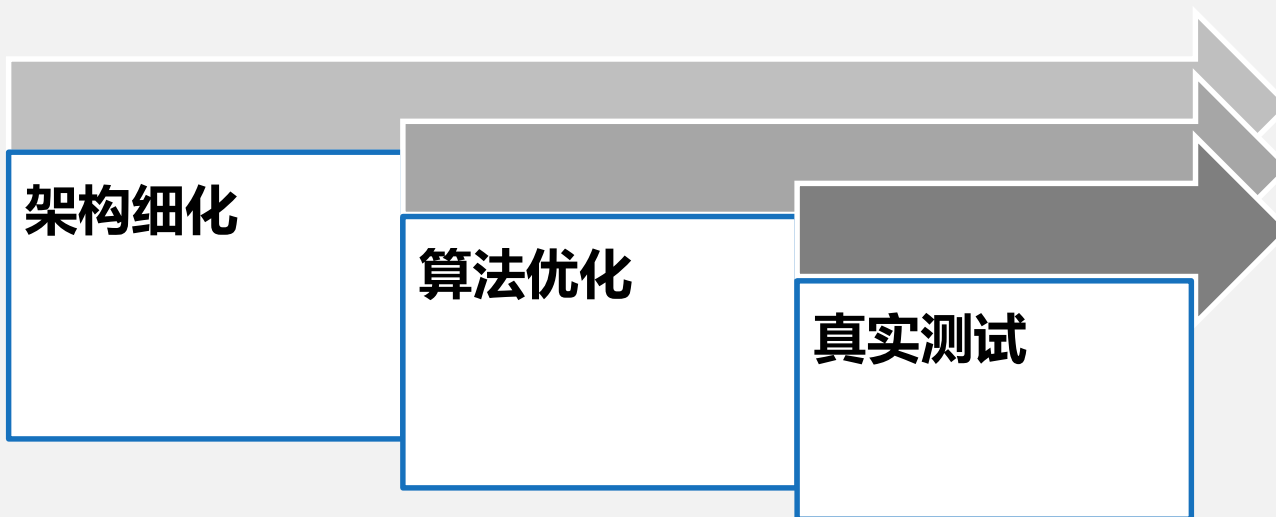
- 利用流程规避风险

细化设计文档，设定承载，内存，网络，IO，数据库操作频率等，帮助思考

过程监督，主要是数据结构上的选用是否合理，算法是否高效

定期利用工具诊断效率问题（Gprof Kprof gprof2dot.py等）

真实的外网测试环境，内外网络差异造成



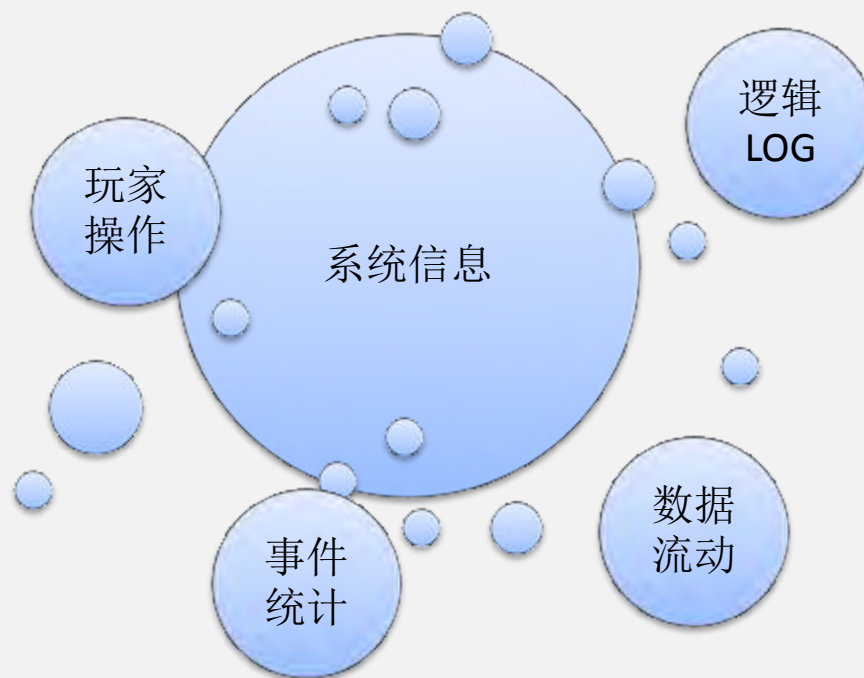
- 服务器要有强大的LOG系统

可以实时设定等级LOG记录等级

分模块分功能的统计分析功能

峰值、均值、执行次数等统计

特殊性指定记录，某特定行为等



论系统LOG的重要性

- 灵活的可控制的开关系统和配置文件

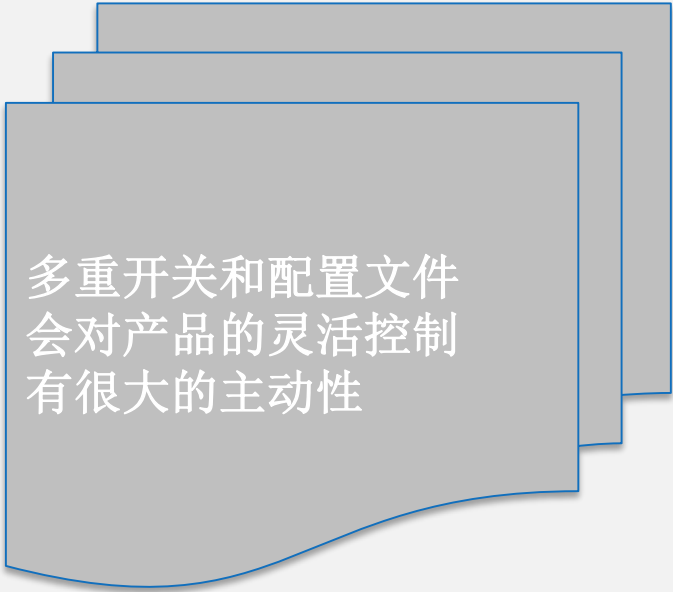
随时指定某模块开放或关闭

随时控制物品的产生

随时控制任务ID的完成和接取

随时控制活动的开放和关闭

控制重新加载配置



多重开关和配置文件
会对产品的灵活控制
有很大的主动性



Q & A

THANK YOU

E-mail :tianbohui@163.com

