



百度贴吧LAMP架构

夏绪宏 @reeze

AGENDA

1. 贴吧简介
2. 贴吧架构发展历史
3. 架构现状及未来



A man with dark hair, wearing a dark jacket and a grey scarf, is looking directly at the camera with a slight smile. The background is dark and textured, possibly a curtain or wall. The lighting is soft, highlighting his face.

您是屌丝么

贾君鹏你妈妈喊你回家吃饭

关于贴吧架构，元芳，你怎么看？

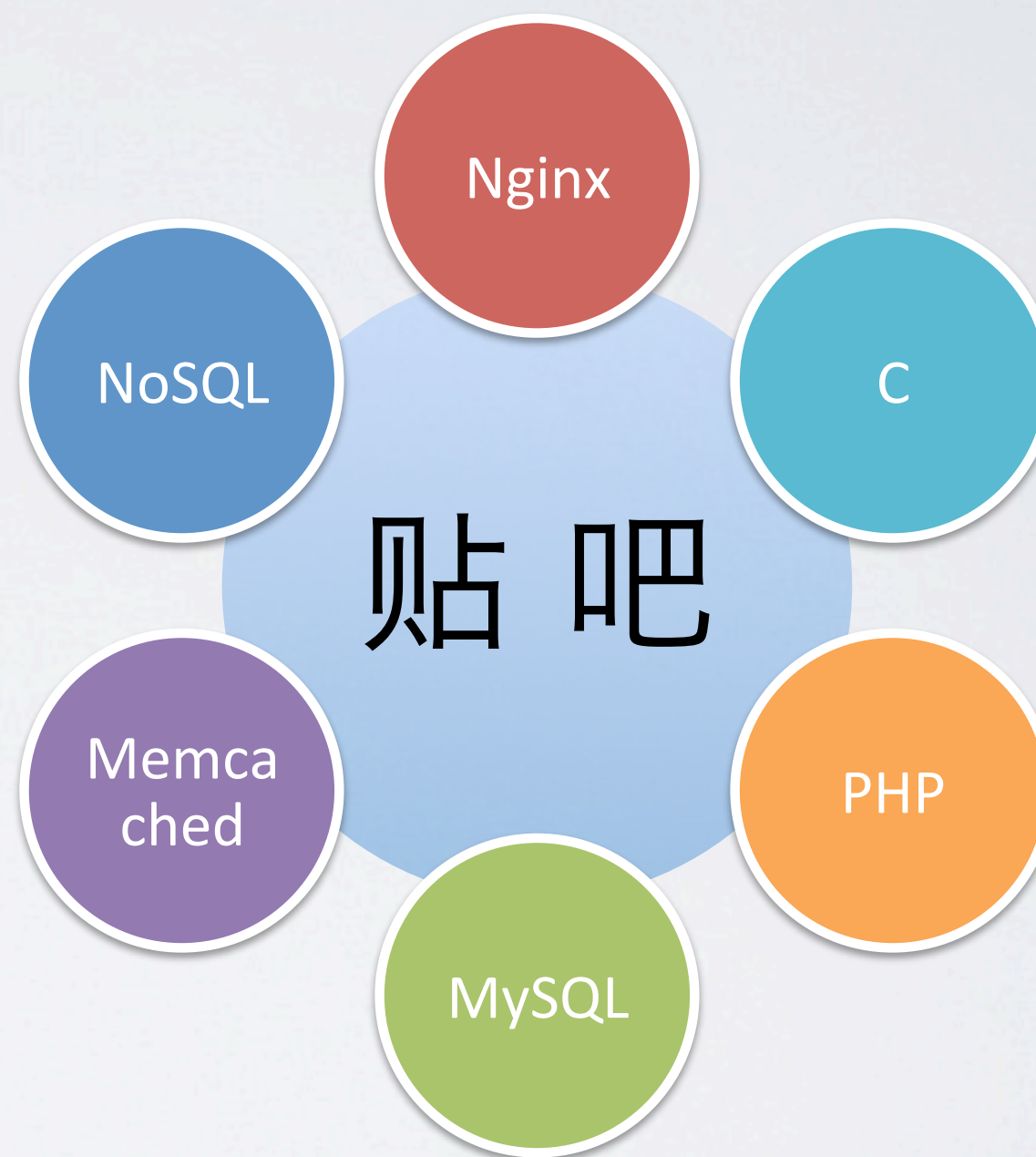
白富美，高富帅

你懂的。



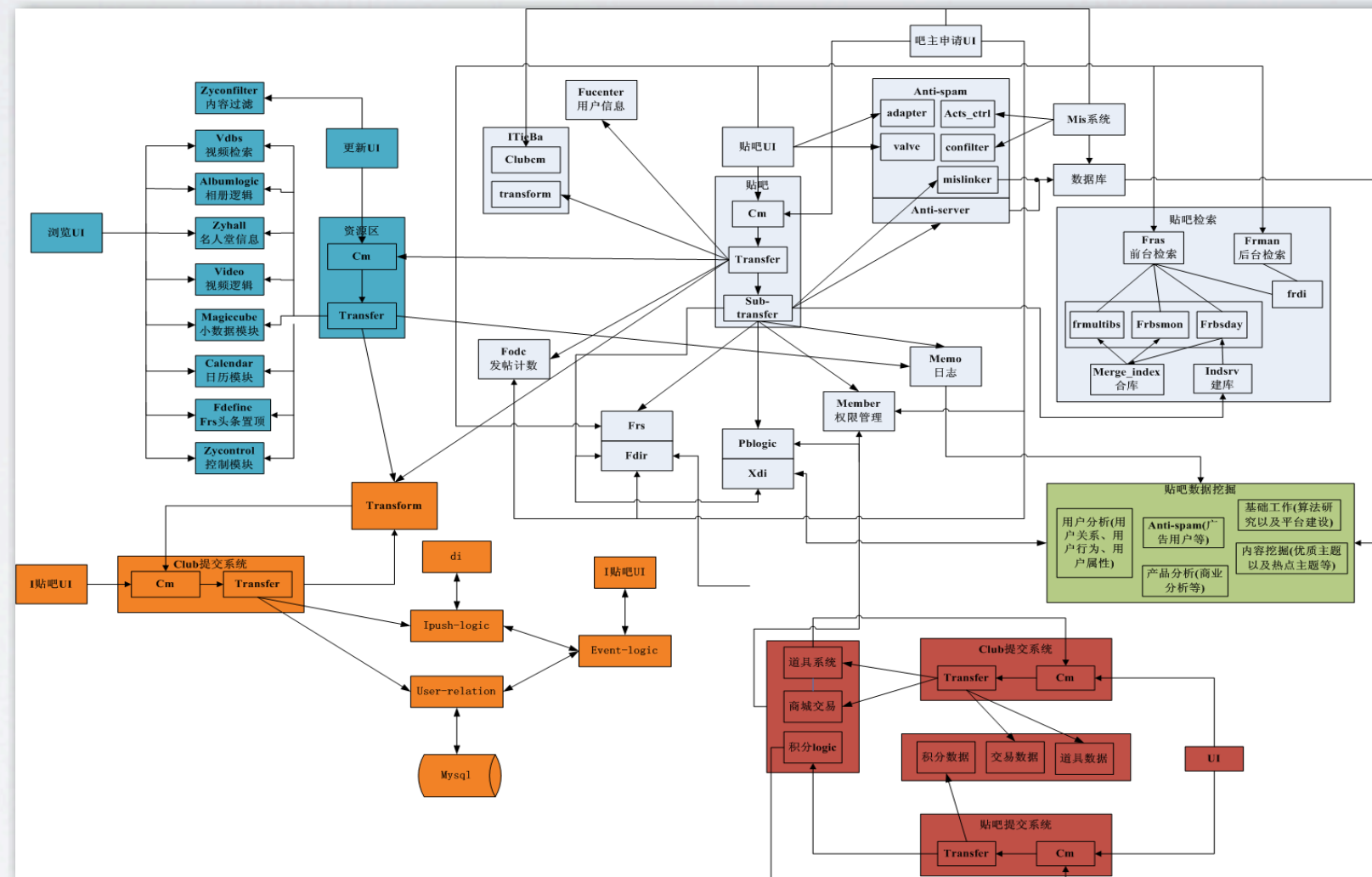
TECHNOLOGY STACK

- Nginx
- C/PHP
- Mysql、分布式数据库
- Memcached
- NoSQL

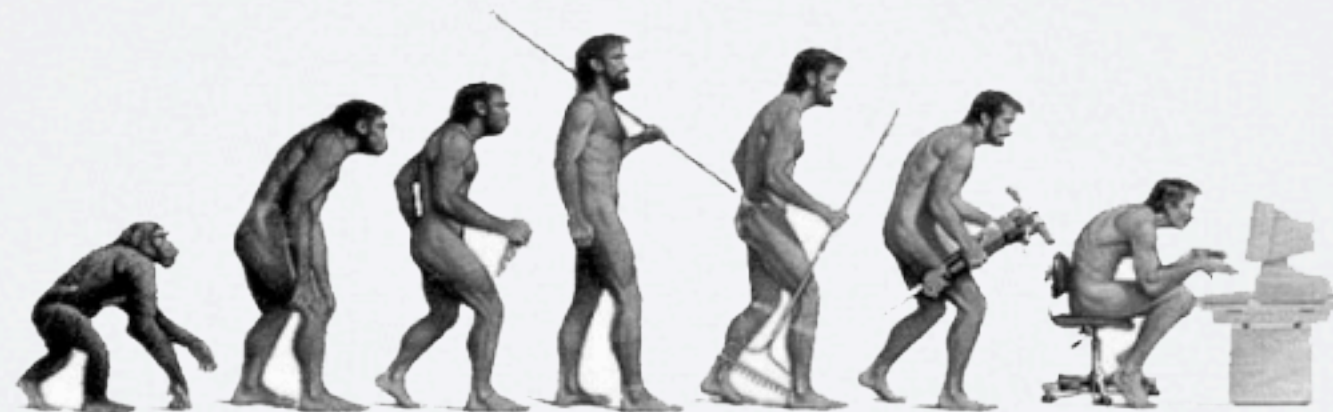


概况

- 30+子系统
- 200+模块
- 十亿级PV
- 亿级提交量
- T级数据



贴吧架构发展历史



架构演进

- 三个阶段

1. 性能稳定性优先

2. 迭代速速优先

3. 平台化，运维效率，敏捷，自动化优先



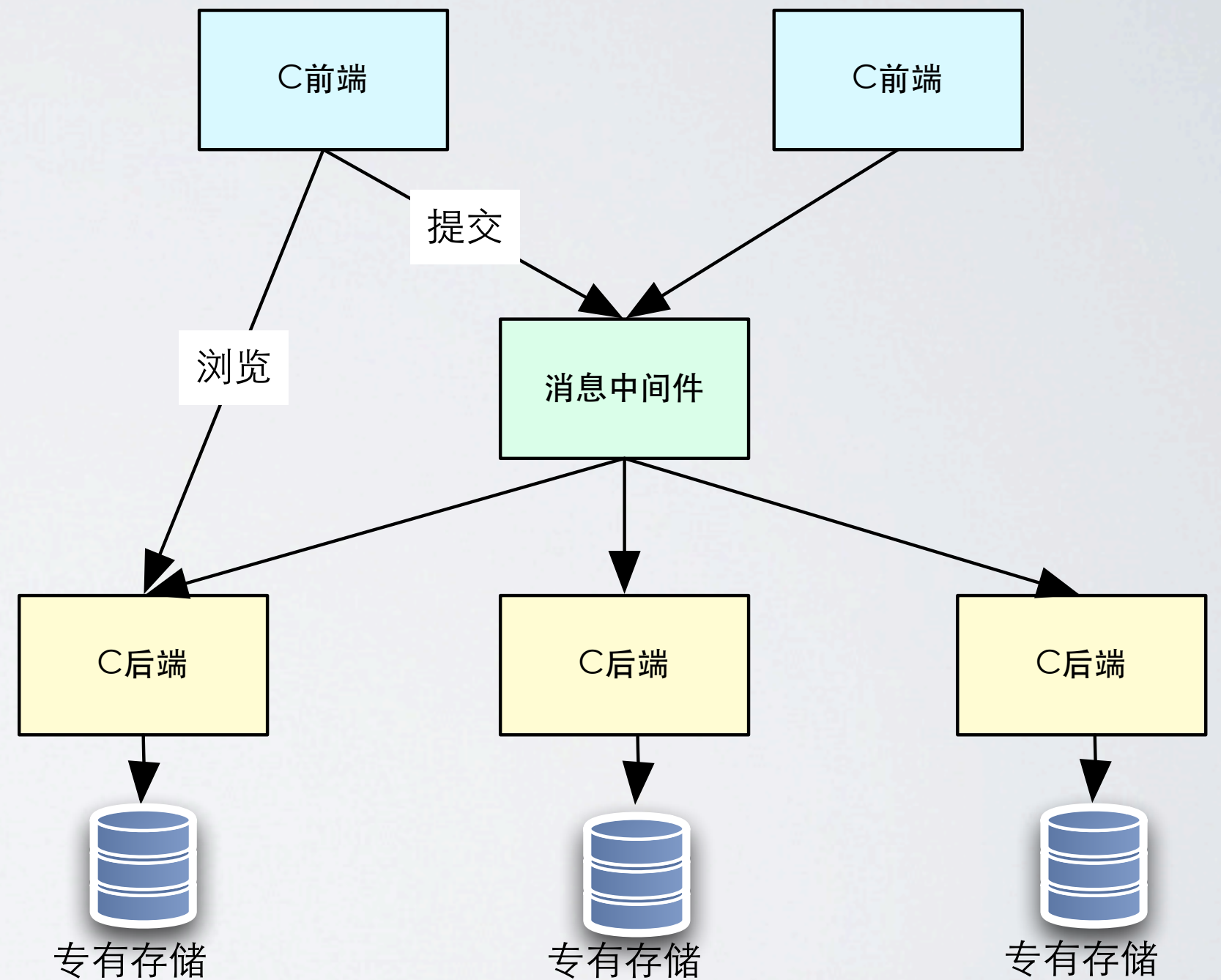
性能优先

- 产品开始功能较少：首页，主题列表及帖子
- 追求性能和稳定性



历程

- 08年：C前后端，专有存储
 - < 1亿PV < 400w发帖
 - 业务规模小，迭代慢
 - 业界缺乏成熟开源方案



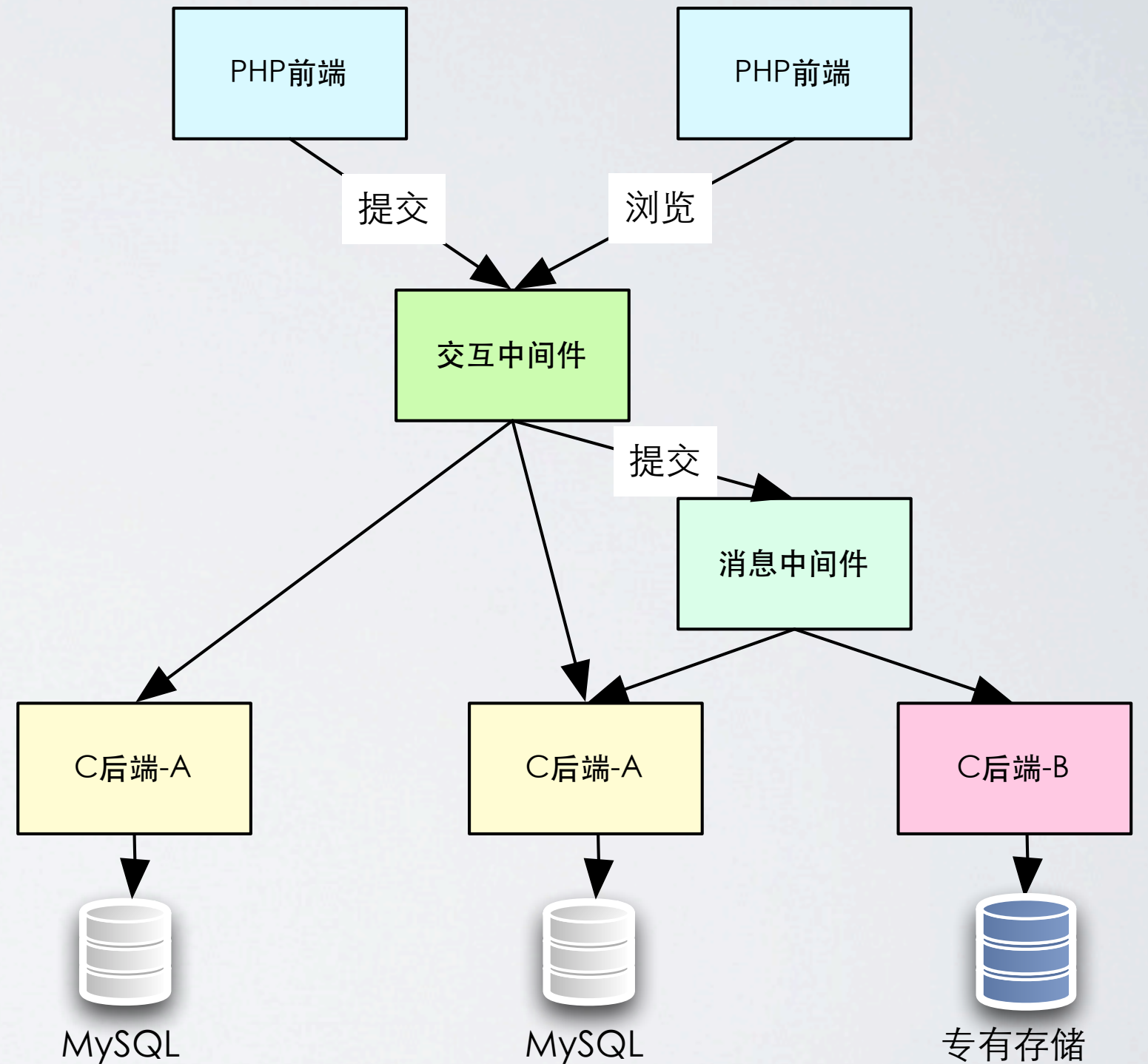
历程

- C编写业务实现**成本偏高**
- 项目动辄一两个月，**周期太长**
- 新项目越来越多，对迭代开发速度开始要求



迭代开发速度优先

- 09年-10年
- 存储更多采用MySQL
- 前端PHP化
- 异构后端，增加交互中间层



交互中间件

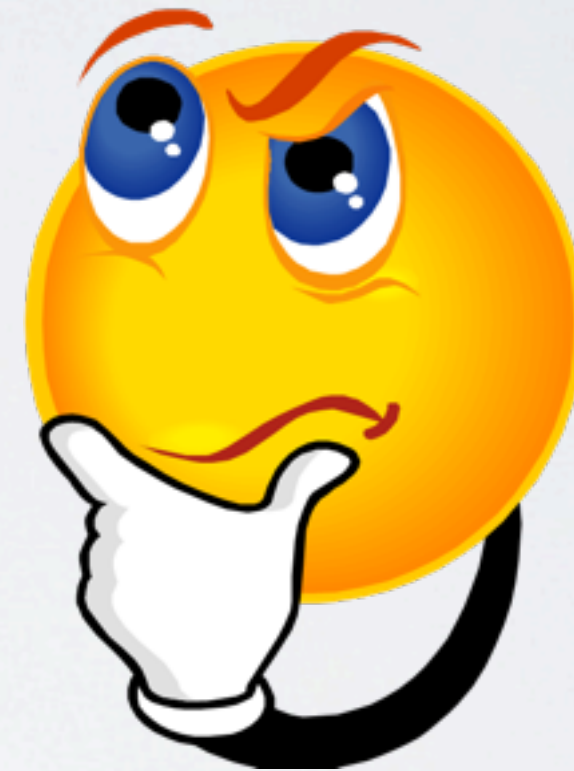
- 统一资源定位，负载均衡
- 屏蔽交互细节，后端可以是任何的协议和数据打包格式。
- 后端资源配置收敛，简化运维
 - 资源包括：后端C模块，LAMP模块，Mysql，cache等

```
<?php $result = ral('any_backend_service', 'method', $params); ?>
```

- 以PHP扩展形式实现：除非真的很必要，不要用扩展。

问题

1. 业务规模开始变大，模块增多
2. 并行开发问题
3. C模块运维代价上升
4. 后端开发效率不高

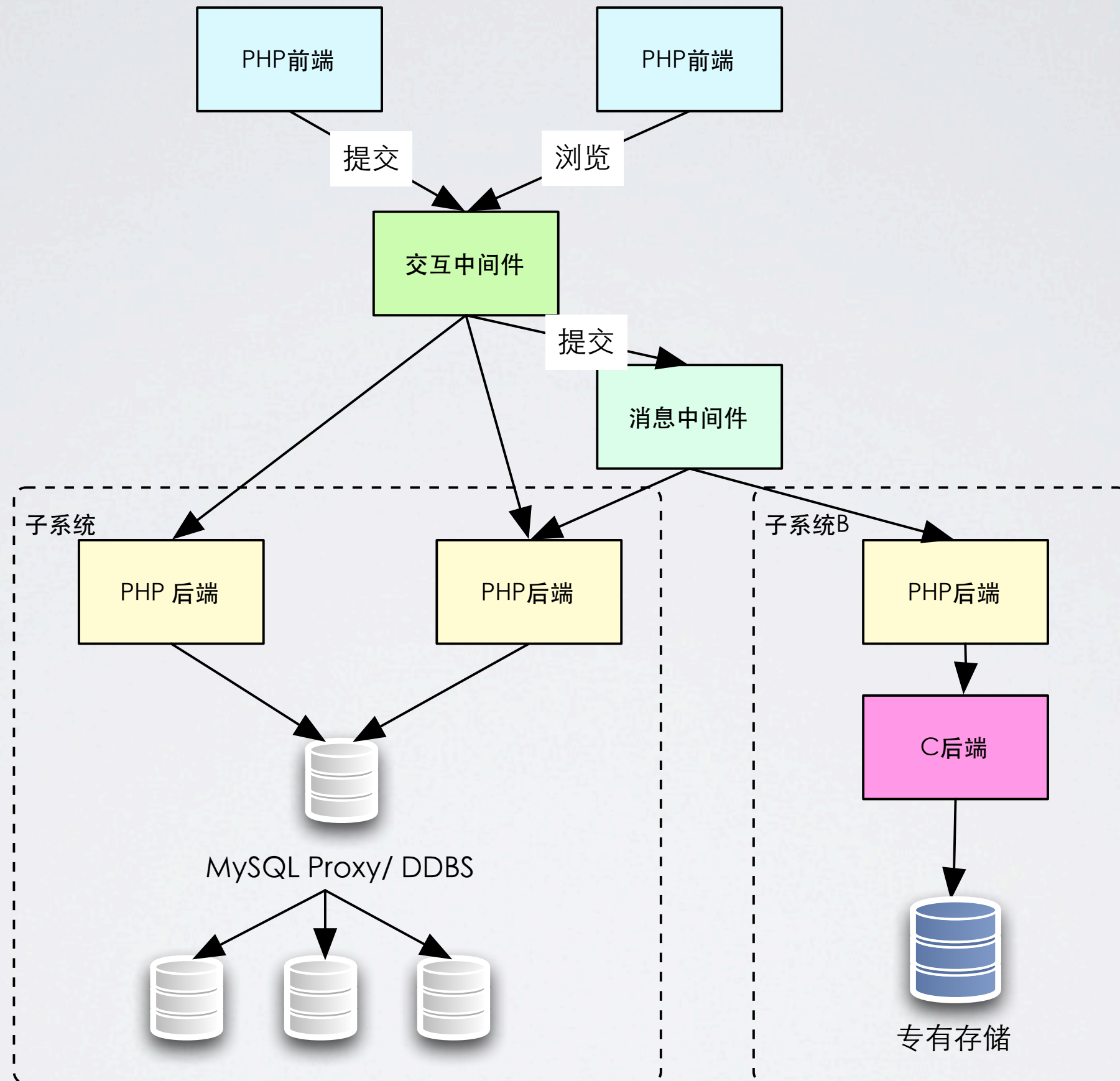


平台化，可运维

- 11年~：规模，业务爆发
 - 十亿级PV，亿级提交
 - 40+前端模块，70+C后端模块
 - 每周~100+并行项目
 - 移动无线业务爆发式增长

历程

- 11年- ~：规模，业务爆发
 - 通用化，集群化
 - 前后端LAMP化
 - 服务化：系统划分，API

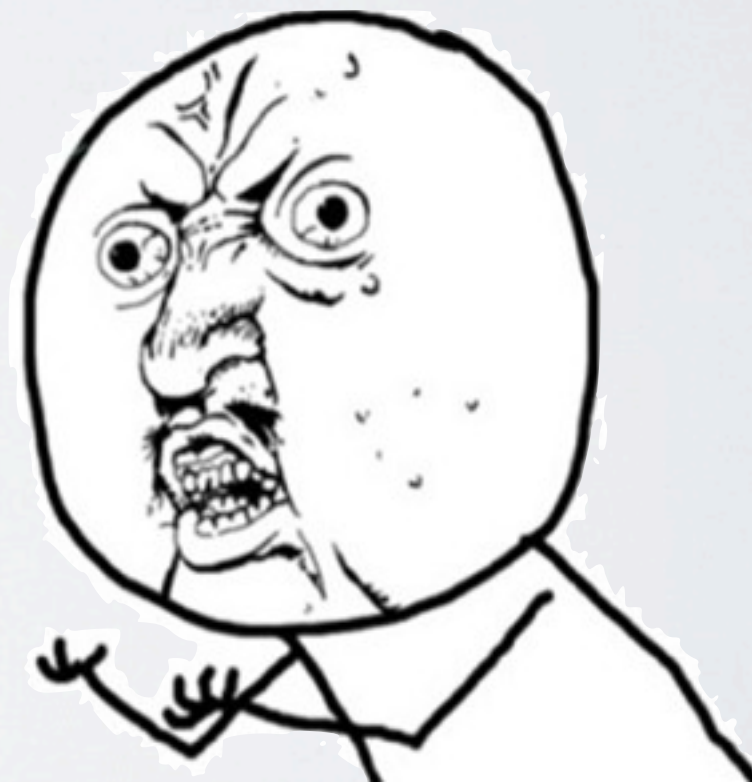


LAMP化

- C -> PHP
- 有状态 -> 无状态 Share Nothing
- 专有存储 -> 通用存储
- RPC: 自有协议 -> HTTP协议

性能下降

- 预期之内
- 响应时间变长：PHP本身的性能消耗
- 吞吐下降

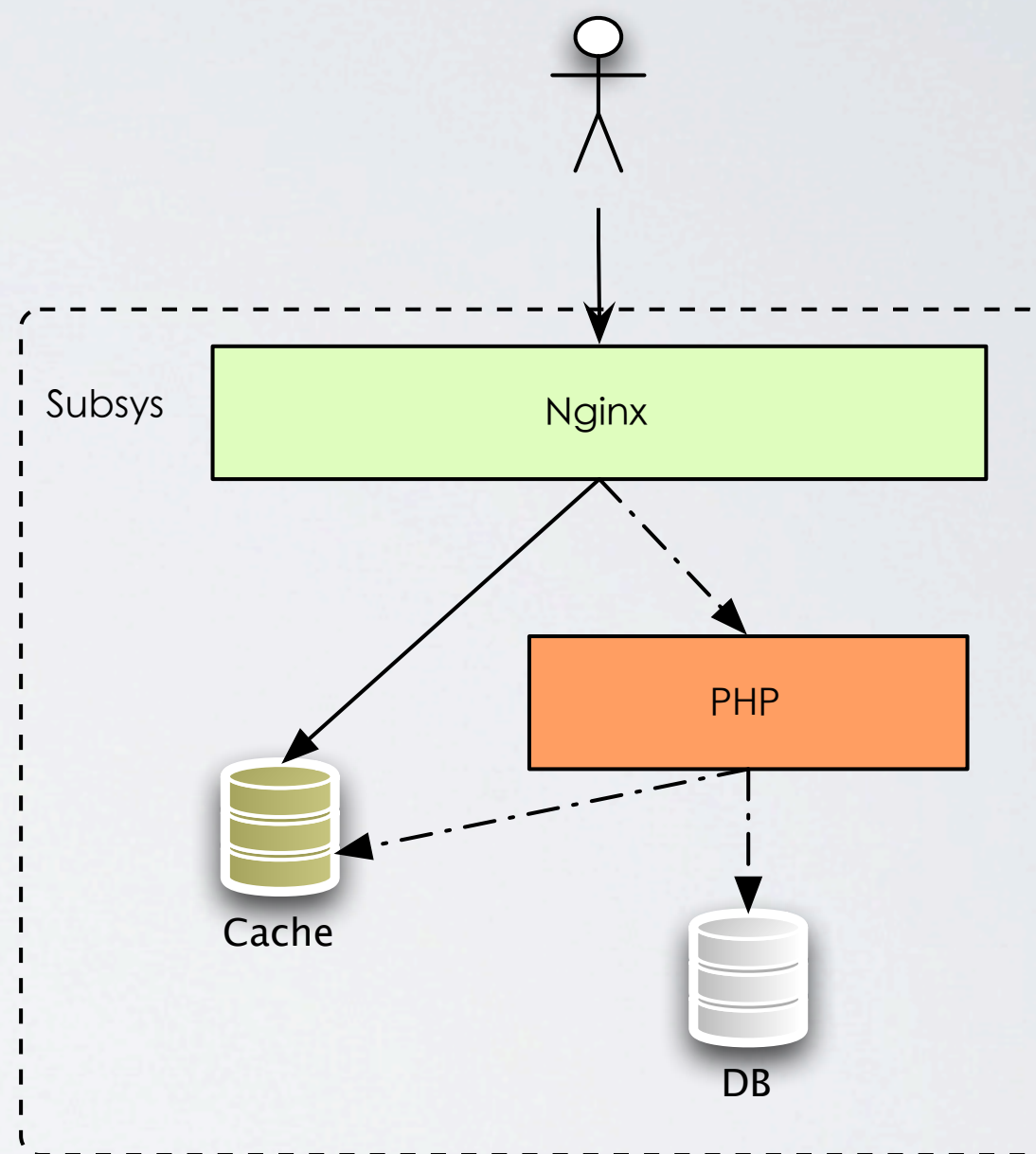


LAMP化

1. 高层cache nginx cache:

- 减少php消耗。节省~50%+机器
- 2ms -> 10ms -> 2ms

2. 并行化: 后端请求并行化





- 后端调用串行变并行
- 185ms -> 63ms
- 只是理想情况
- 现实业务比这复杂

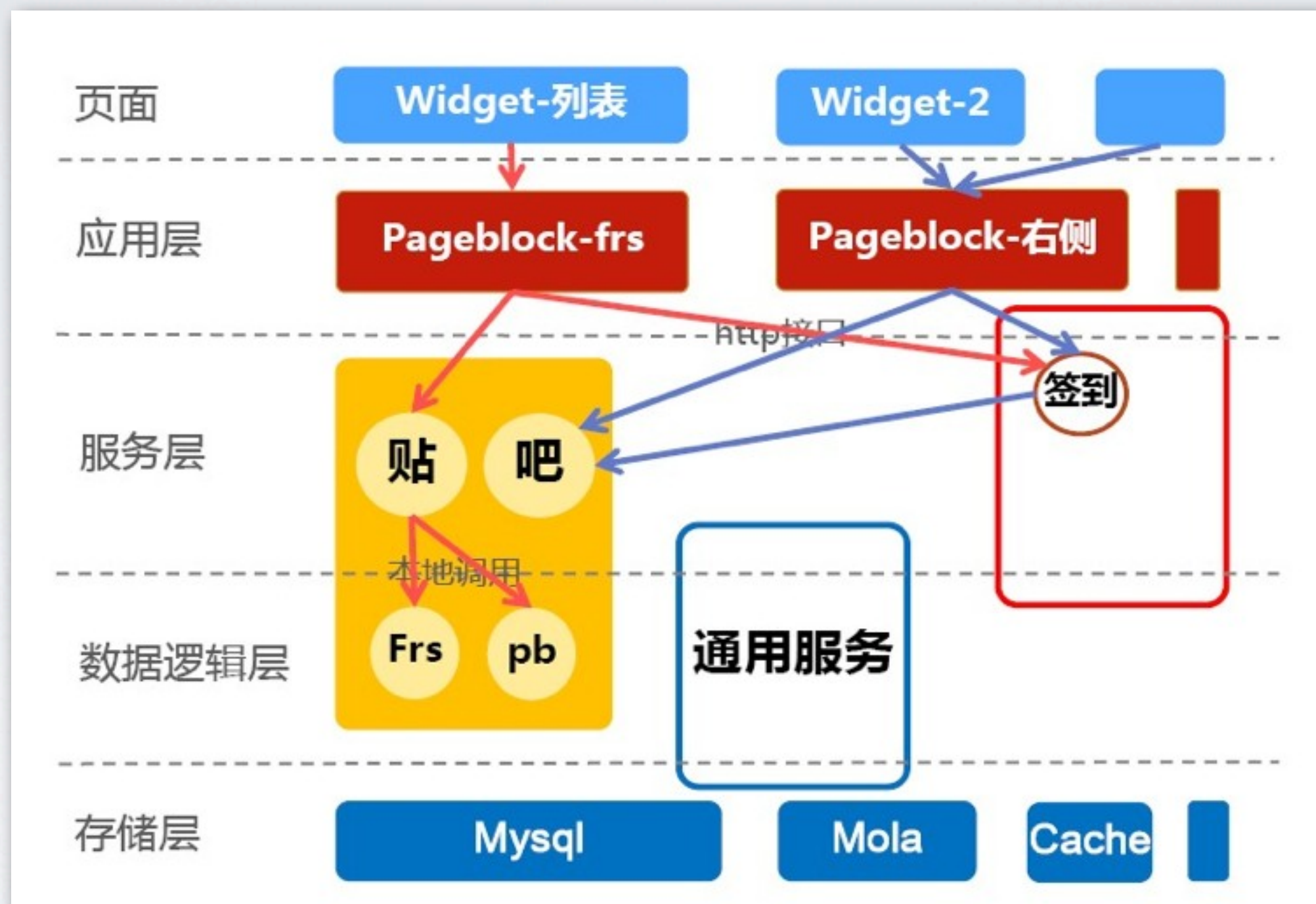
理想架构

- 开发效率：快速迭代，并行开发，简单统一的规范
- 运维上线效率：自动化运维，可扩展
- 可扩展：可插拔，模块化，系统划分

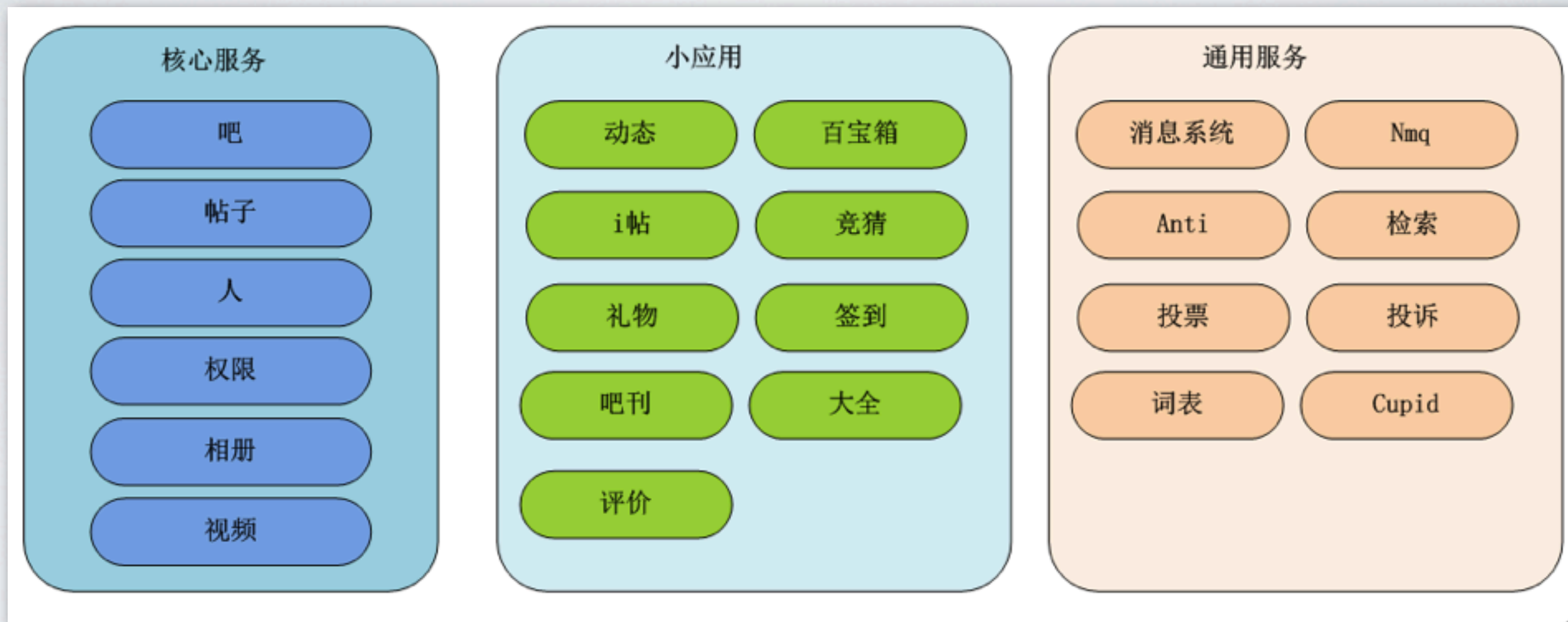
• 开发速度 + 运维效率 > 性能



水平分层



子系统划分



全流程平台化

- **开发**: 基准环境 快速构建
- **测试**: 持续集成, 在线测试OTP
- **上线**: 快速上线, 自动化上线
- **运行环境**: ORP (Online Runtime Platform)
- **在线监控**: OMP(Online Management Platform)

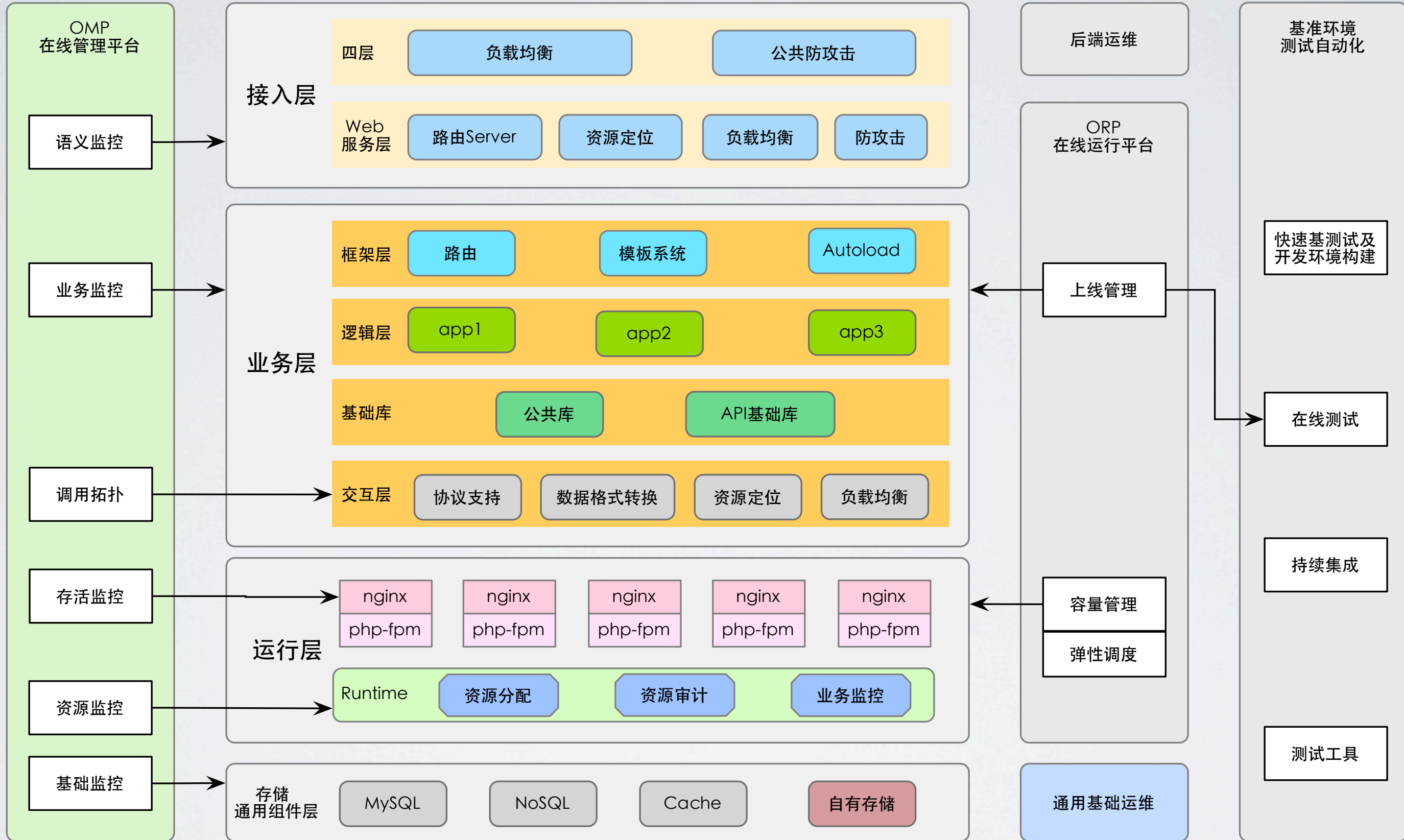


监控

开发/运行

运维

测试

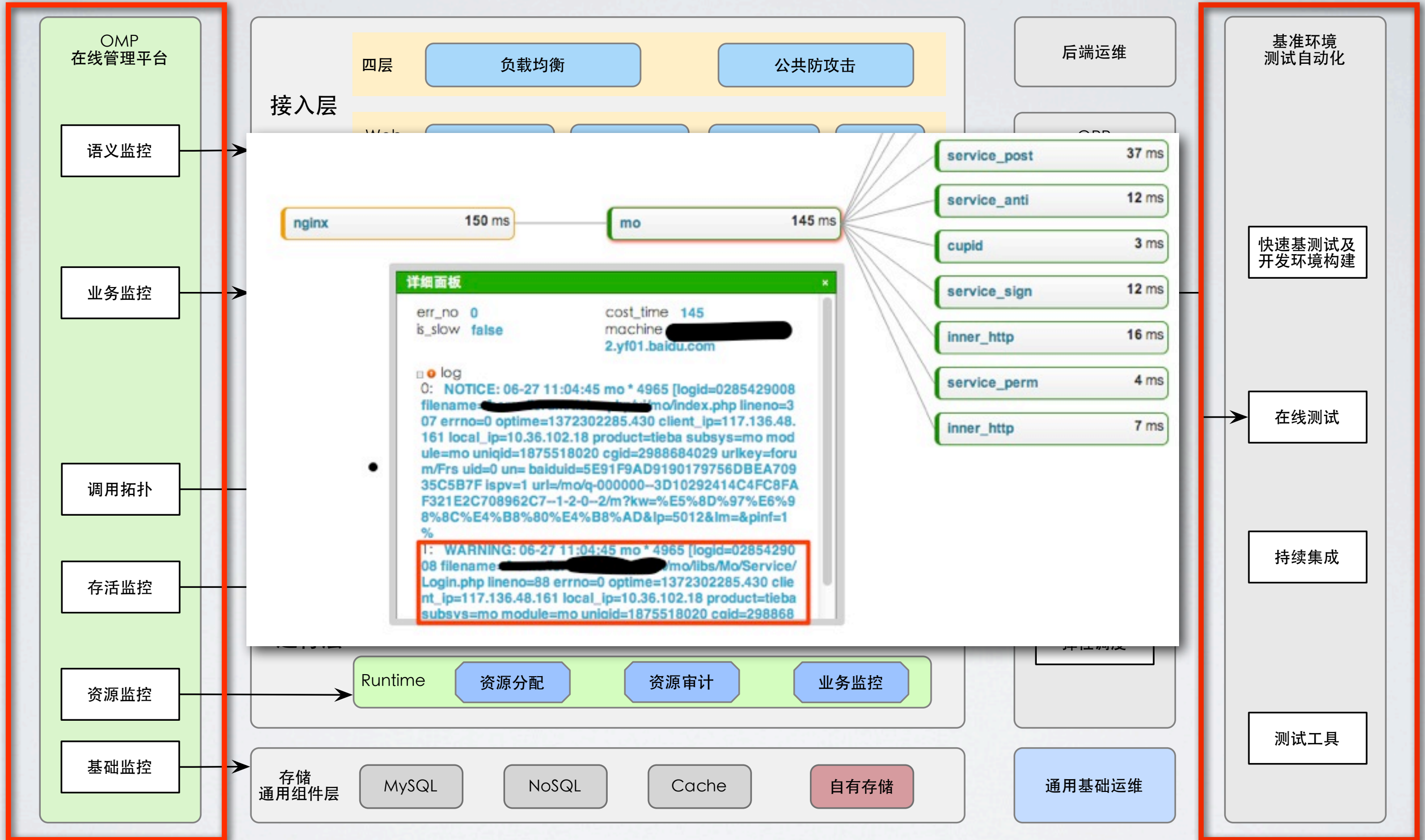


监控

开发/运行

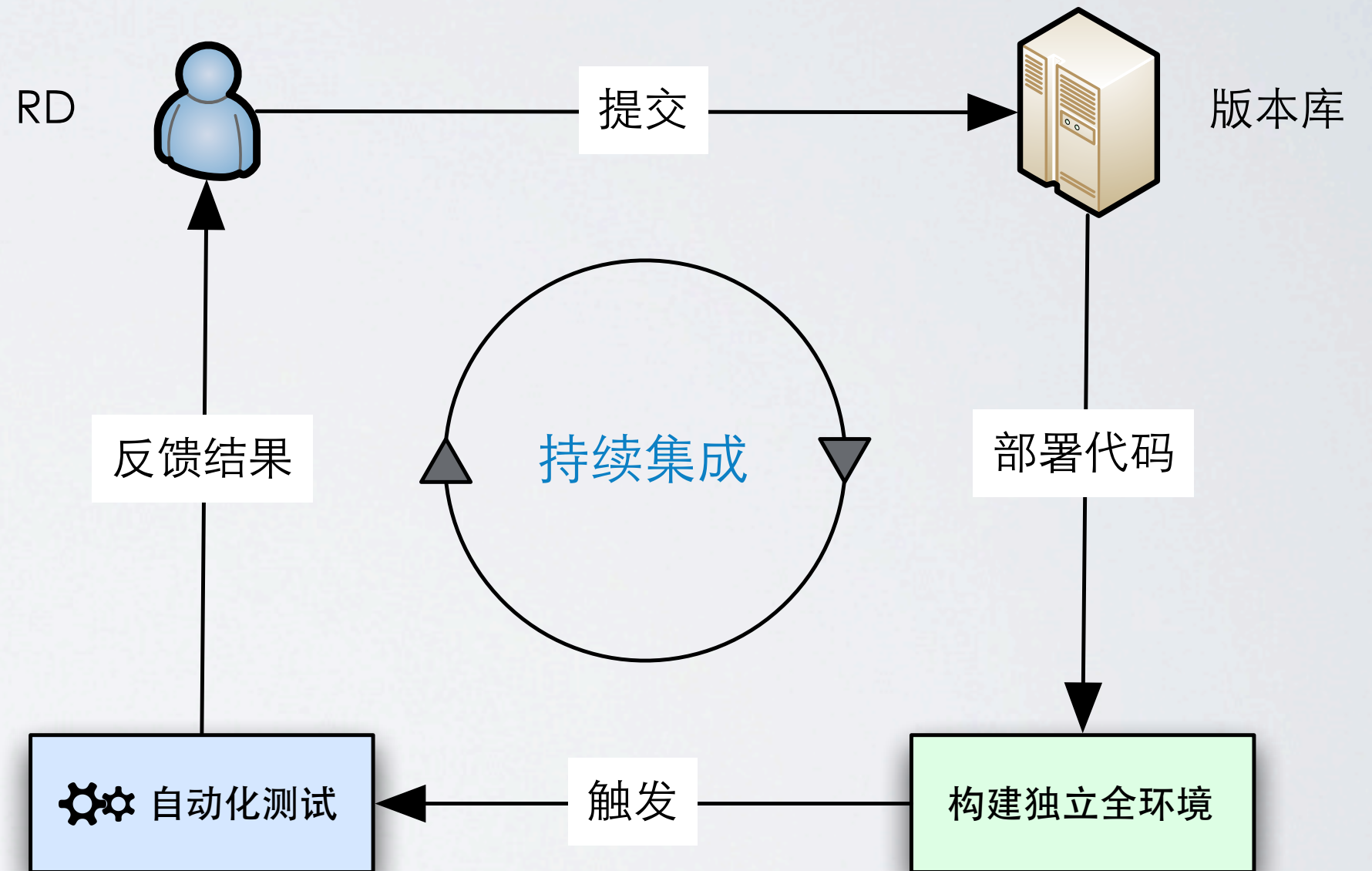
运维

测试



测试：持续集成

- 自动(架构, 编码)规范检查
- 减少QA人力成本
- 提高代码质量



运维

1. 上线，扩容流程复杂，自动化程度不高。
2. 模块混部，关联复杂，手动维护代价高
3. 资源利用率不高

运维

- 构建自有PaaS云平台：ORP (Online Runtime Platform)
 - ★ 统一运维
 - ★ 资源隔离：资源保证
 - ★ 充分利用资源
 - ★ 弹性调度

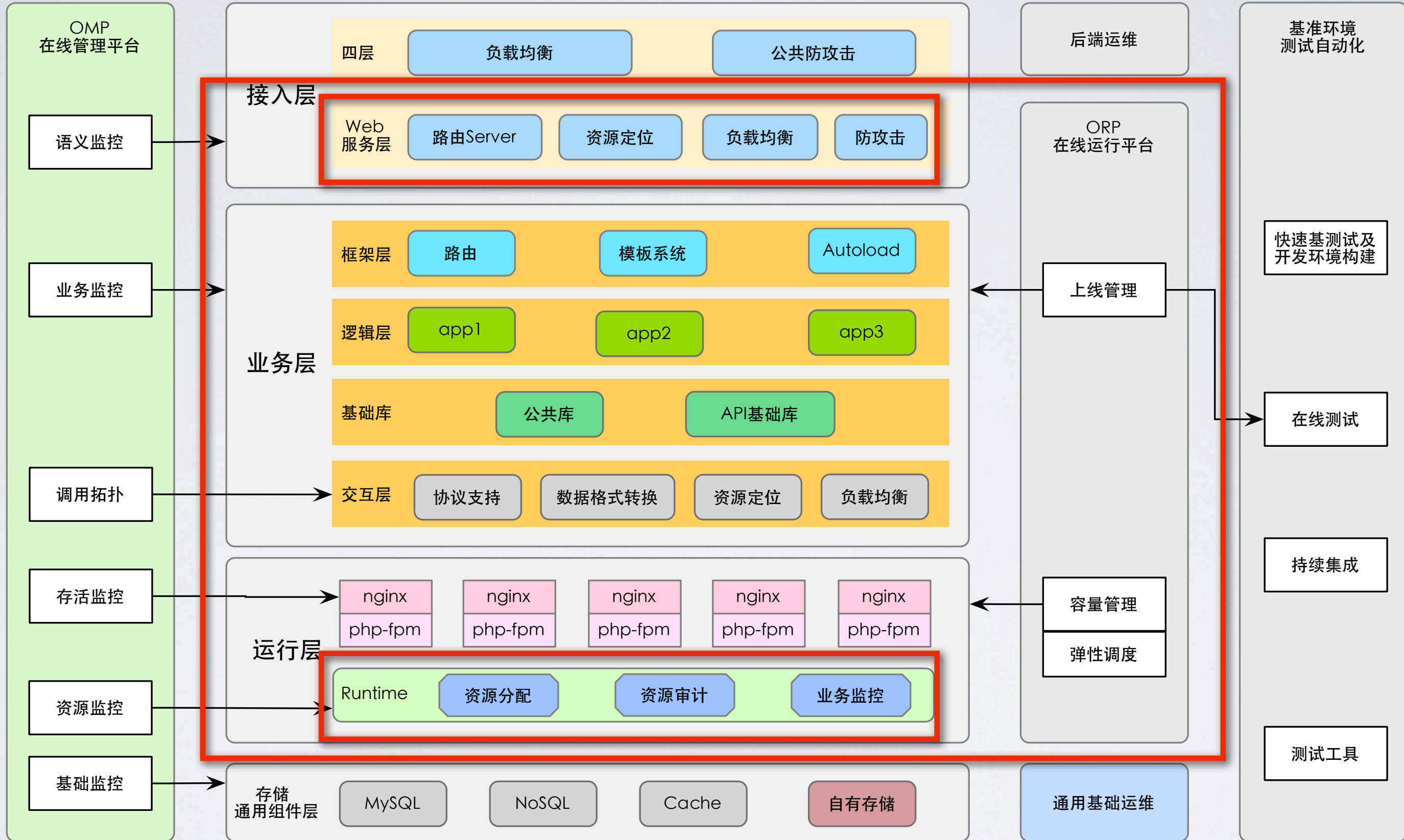


监控

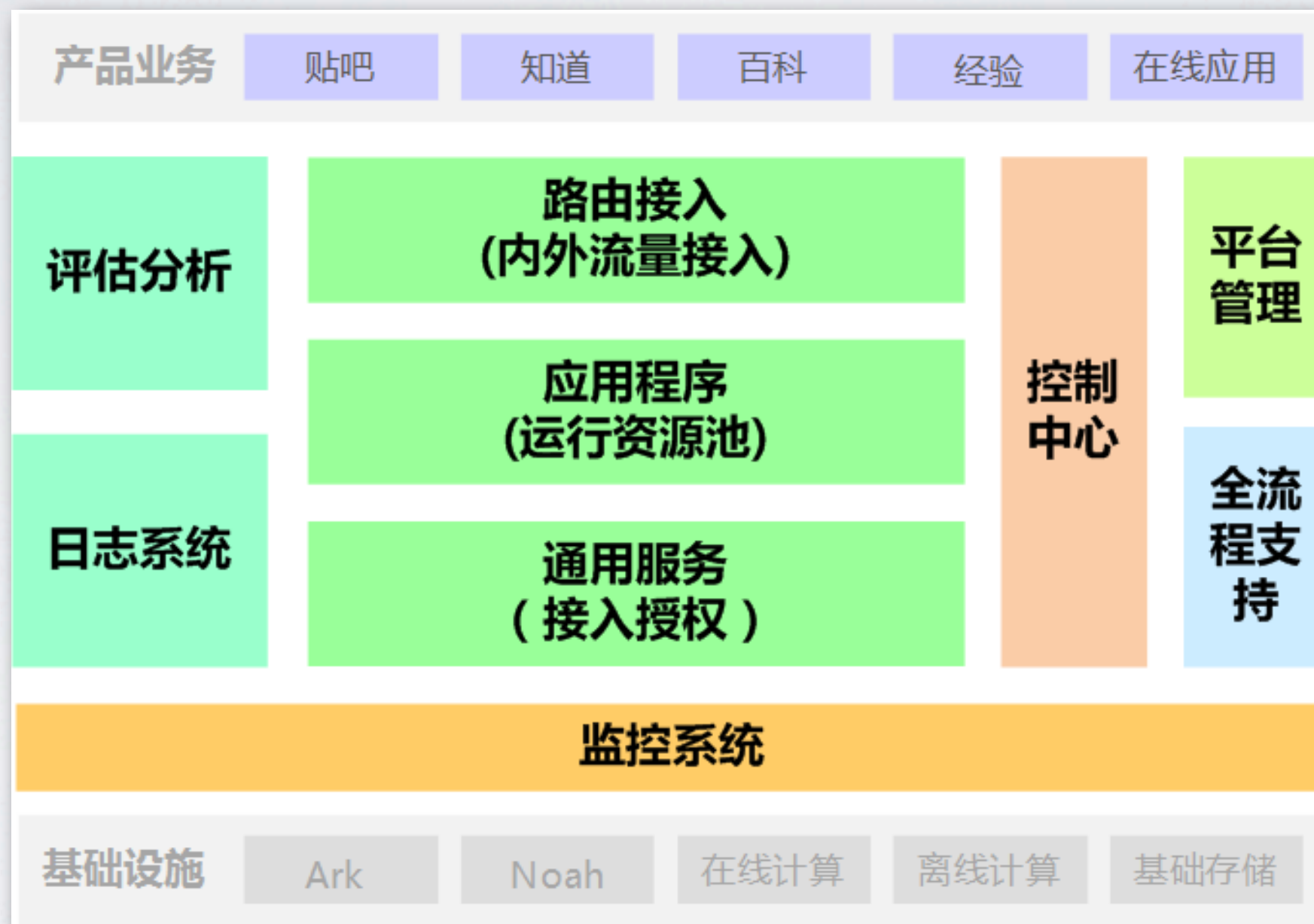
开发/运行

运维

测试



架构模型



收益

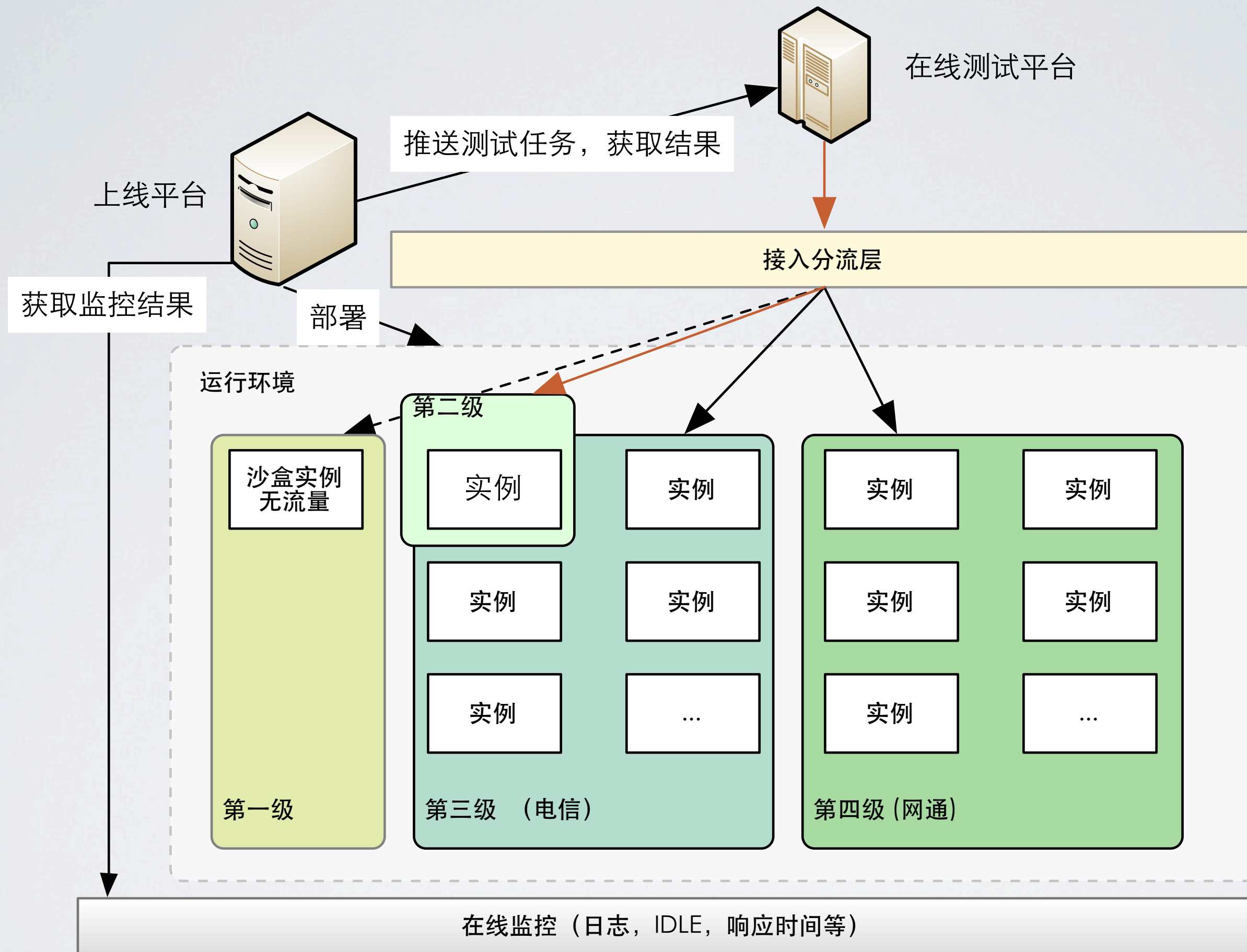
- 上线效率： 30-45分钟 -> 5-10分钟
- 人力成本降低： 中小产品线无需专门OP
- 机器利用率提升： 总体节省约40%+



上线质量保证

- 分级灰度发布
 - 自动化在线功能测试
 - 实时监控整合
- 及时发现上线问题，控制影响






CONCLUSION

- LAMP化
- 运维效率，平台化，自动化。
- 展望：
 - 基础性能优化：Nginx, PHP
 - 云平台的进一步优化：打通全流程，自动化上线等。

贴吧欢迎你

- Talk to me
- Resumes are welcome
- Email: reeze@php.net





说好的PHP
执行及Zend
引擎呢?

- Follow me: 微博 @reeze
- <http://github.com/reeze>
- <http://www.php-internals.com>

参考

1. 《贴吧LAMP解决方案》 <http://stblog.baidu-tech.com/?p=1324>
2. 《老王的技术博客》 <http://blog.sina.com.cn/zgawangbo001>