

从微服务到 Serverless 架构： 享受纯粹的编程乐趣

王晓波

同程旅游 首席架构师



CNUTCon 2017

全球运维技术大会

上海·光大会展中心大酒店 | 2017.9.10-11

智能时代的新运维

大数据运维
安全
SRE
DevOps
Kubernetes
Serverless
游戏运维
AIOps
智能化运维
基础架构
监控
互联网金融



主办方

Geekbang > InfoQ

极客邦科技



实践驱动的IT教育



<http://www.stuq.org>

斯达克学院(StuQ)，极客邦旗下实践驱动的IT教育平台。通过线下和线上多种形式的综合学习解决方案，帮助IT从业者和研发团队提升技能水平。



10大职业技术领域课程

TABLE OF

CONTENTS 大纲

- 同程实践 Serverless 的背景
- Serverless 与传统架构比较
- 同程的 Serverless 实现了什么
- Serverless 架构在同程的一些实际使用那些业务场景分析
- 我们的下一步在 Serverless 上要做什么

一条 SQL 到一个服务的距离到底有多远

环境，框架，依赖，太难搞



- 各种环境的统一性
- 各种开发框架的烦心事
- 各种调用的乱依赖
- 代码以外花了多少时间

部署，运维，扩容，太麻烦



- 部署在哪，多少实例
- 各种配置，上线，下线
- 扩个容各种关注点
- 真的可以人人是全栈吗

调试，性能，安全，太复杂



- 开发调试最烦对外依赖
- 性能往往做好后才细想
- 安全出了问题才懂了它
- 专业领域还有多少

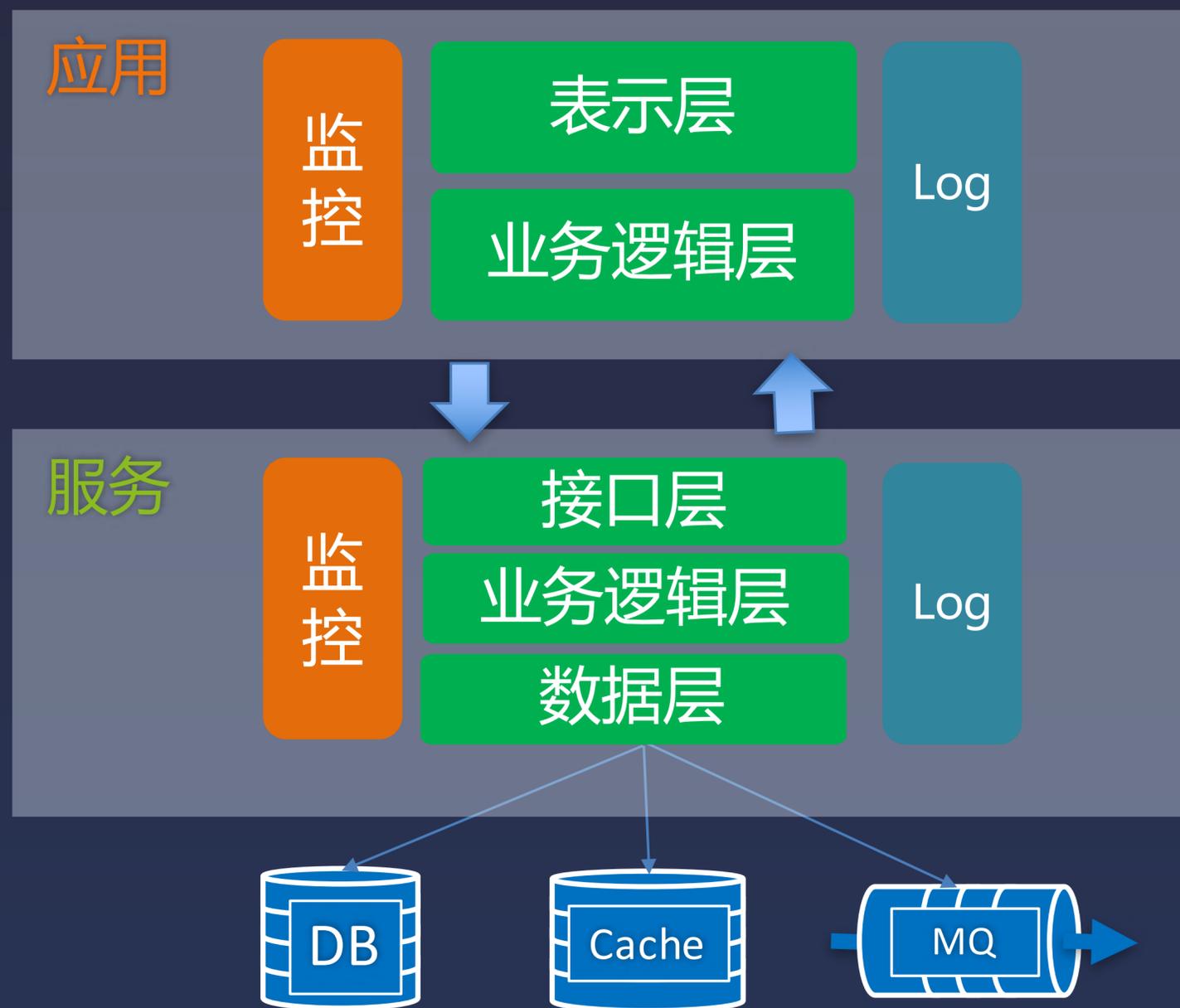
从一条 SQL 到一个可用服务距离真的很远很远

如何安安静静的写代码 ???

Sheer Coding Pleasure

让程序员享受纯粹的编码乐趣

传统的系统架构



特点:

应用多年大家多理解
也有粗粒度的服务展现
拥有一定的系统伸缩性能力
重了框架国轻了架构

传统架构的缺点

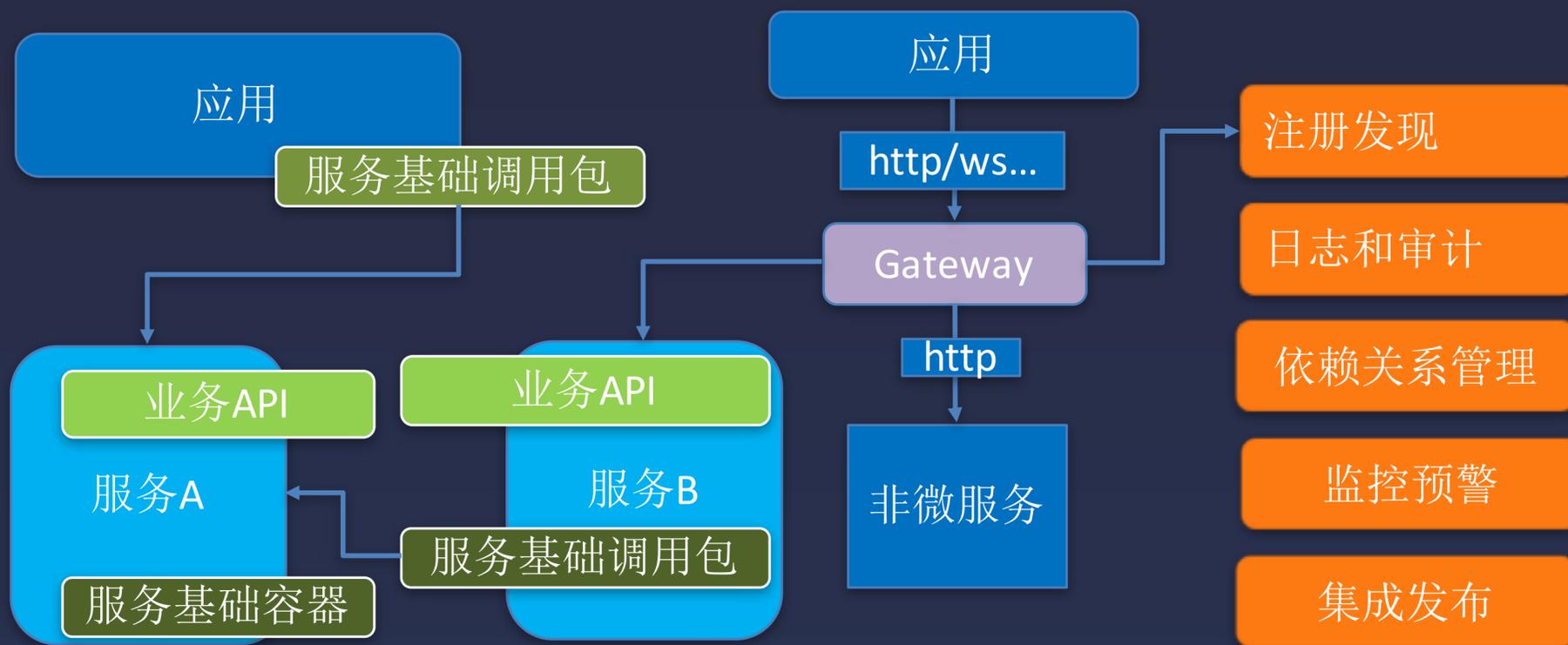
★ 一个简单的应用变的不简单了

★ 问题:

- 不只是个网站
- 各种应用增加到N千多个，想一想玩不动了
- 流量动不动就是海量（每天数亿级请求量
- 原来很简单的一句SQL，现在没法用了
- 最痛苦的是连缓存也跑不动了
- 服务器加到不知道放到哪里好
- 运维最忙的事怎么是各种背不完的锅
- 总体看来就是又大又乱，一头雾水



同程的微服务架构



- 注册发现
- 日志和审计
- 依赖关系管理
- 监控预警
- 集成发布

- 接口的统一
- 容错处理 (限流、回退、隔离、熔断)
- 全链路的服务监控
- 服务注册发现, 控制服务的API数量
- 统一代码框架, 支持多种编程语言
- 服务依赖关系管理(服务的分级)
- 服务的CI/CD

微服务架构的缺点

- ★ 微服务整体推进是要时间，不可能一次改完，中间过程需要过渡
- ★ 还有很多为快速打样的做的单体应用，或一些简单应用
- ★ 一个微服务经过长时间迭代更新后也需要新重构

问题:

整个体系中并不是只有微服务

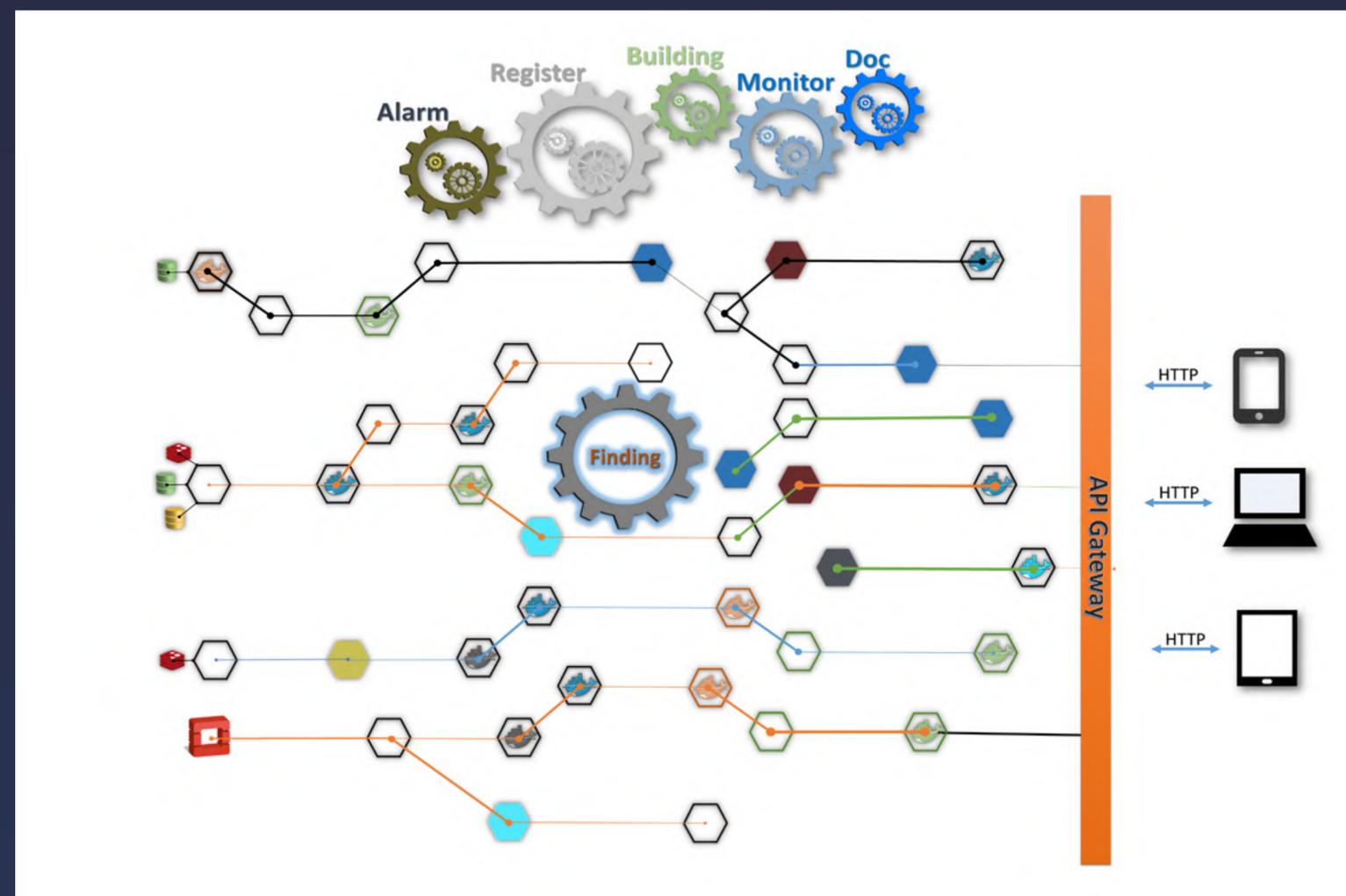
管理，调用多会因为这个不得不变复杂起来了

写一个微服务的成本也并不小

开发过程的调试比较麻烦

服务的数量越来越多虽然跑在Docker之上但也是资源

运维虽然DEVOPS了但依然变的有点麻烦



一个代码脚本能不能就是一个微服务

同程在微服务化之后的需求

★ 总结起来这一切就是自己挖个坑和自己再填上坑

- ① 用一两个小时写个服务并上线
- ① 不用关心服务器在哪部多少
- ② 只需要配管人员的配置接下来就完全是代码的事
- ③ 开发人员可以像操作IDE那样完成他不熟悉的技能
- ④ 代码没跑最好不要计算资源成本
- ⑤ 不需要人人都是全栈工程师，他就想好好写倾
- ⑥ 运维就是智能的，让系统与算法去解决问题



Serverless架构是什么

- ★ 跟很多其它软件类似，对Serverless还没有清晰定义

- ★ 如果你的PaaS可以将以前半秒启动的应用在20ms内启动，就叫它Serverless。——
Adrian Cockcroft

```
if (top === window) {
  function calcWidth() {
    var wW = 0;
    if (typeof window.innerWidth === 'number') {
      wW = window.innerWidth;
    } else if (document.documentElement && document
      wW = document.documentElement.clientWidth;
    } else if (document.body && document.body.c
      wW = document.body.clientWidth;
    }
    if (sH = document.documentElement.scrollHeight
      var wH = window.innerHeight || document
      wH = !document.all && (sH > wH)
    }
  }
}
```



- ★ Serverless意味无维护，Serverless不代表完全去除服务器，而是代表去除有关对服务器运行状态的关心和担心，它们是否在工作，应用是否跑起来正常运行

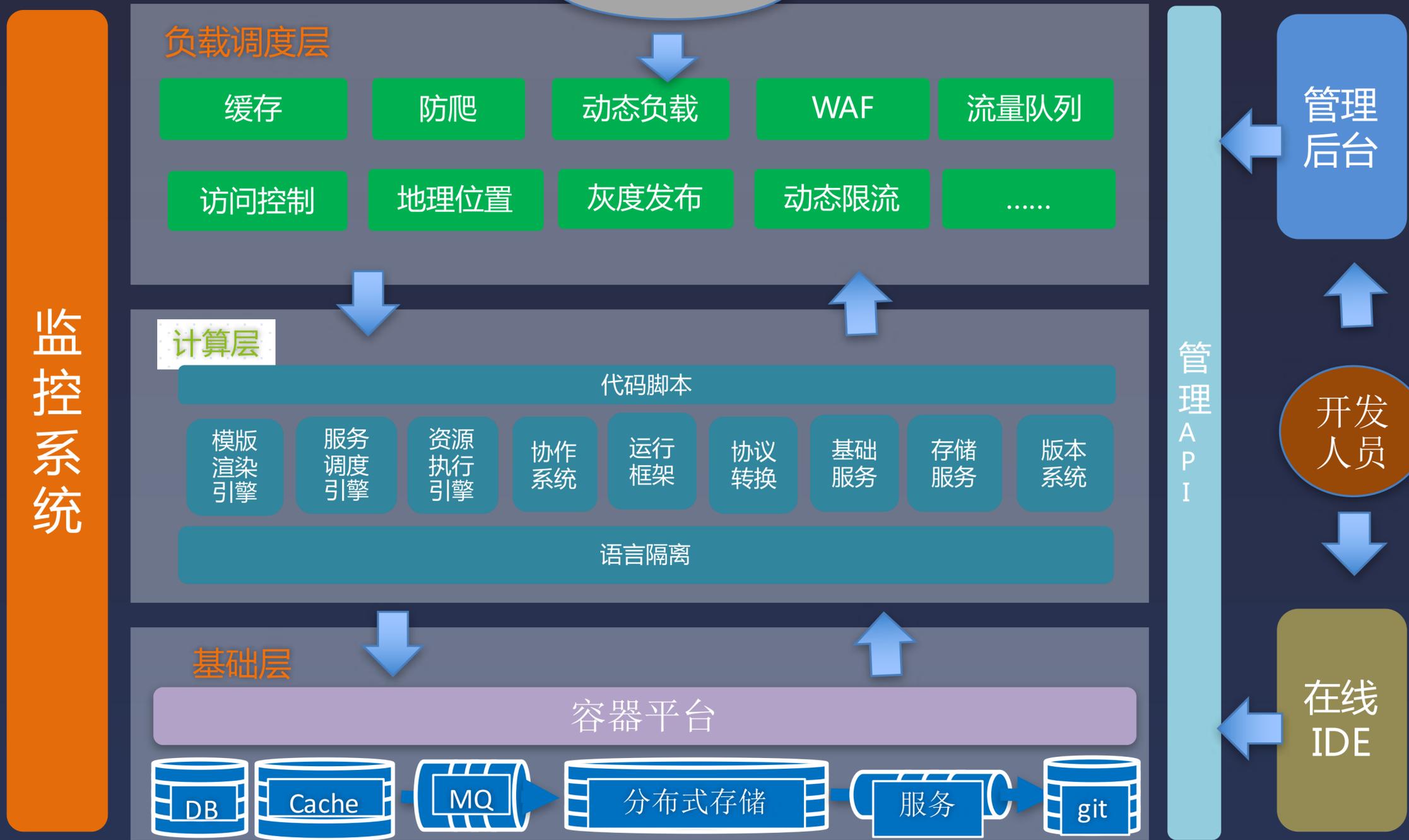
Serverless架构解决同程的问题

- 解决我们最新的痛点：业务快速变化中如何更快地开发。其实在这快速地开发中有很多是快不起来的（开发环境的问题，上线部署的问题，应用弹性设计问题，可运维性的问题等等）。

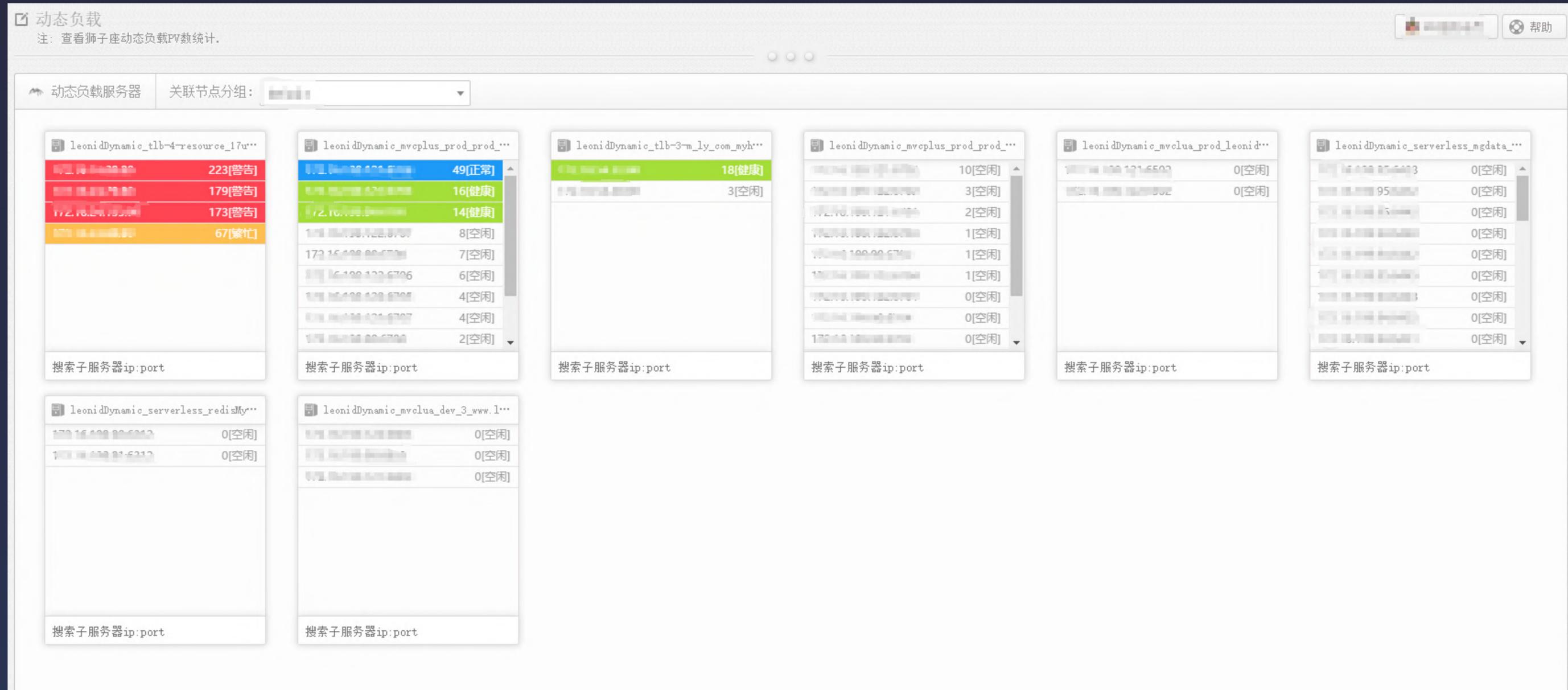


同程Serverless架构的结构

- ① 居于原有基础服务
- ② 利用原有容器平台
- ③ 集成能集成的一切



Serverless架构的资源利用率



★ 最小单元部署4台物理服务器最多可支持10万个应用

编程框架的云化建设

- ★ 编程SDK升级无感
- ★ 所有使用均于代码方式使用

```
tpls.js • orderprint.js orderdetail.js ordercommentsuccess.js ordercomment.js demo.js
1 let data = await utils.pGetTplAsync(['...', '...', '...', '...'], req);
2 res.
3 await appendOrderAsync('...', data);
   append
   clearCookie
   cookie
   get
   header
   json
   jsonp
   redirect
   renderAsync
   send
   status
   type
```

```
reqOption = {
  url: `${util...}${orderid}`,
  headers: {
    "Referer": "http://www.taobao.com",
    "Host": "${util...}",
    "Cookie": "${ut...}"
  },
  timeout:5000
};

orderInfo = {};
s1 = Date.now();
[err, resp, rspInfo] = await utils.pGetTplAsync(['...', '...', '...', '...'], req);
s2 = Date.now() - s1;
var rspJson = JSON.parse(rspInfo);
if (rspJson && rspJson.orderInfo) {
  orderInfo = rspJson.orderInfo;
} else {
  res.redirect("/");
  return;
}

Object.keys(orderInfo).length === 0 ? {
  console.error(resp);
  res.redirect("/");
  return;
}
```

创建一个新数据源

MySQL 数据源

只读

0

备注

确定 取消

例：从订单数据库中拉出数据

Server.DB.Order.Get("SELECT * FROM ORDER")

Serverless平台化建设:动态负载均衡器

★ Serverless服务拥有毫秒级别的弹性扩容和缩容的能力



多种负载模式

分组模式：分组负载 轮询模式：轮询集群中所有机器
弹性模式：根据访问情况自动增减负载

根据访问量，自动扩容和缩容集群

将应用的分级为6级，根据请求量和后端应用的响应时间，
动态调整负载机器数量



热加载

免去每次对负载均衡的Reload操作，减少系统吞吐量的影响

适用任何应用

动态负载均衡器可以适用任何应用，包括：JAVA, IIS, Go 等等



Serverless平台化建设:数据服务系统

★ Serverless服务拥有访问常规数据源的能力

主流数据库操作能力

兼容多种数据库 Redis, MySql, MongoDB

慢日志记录, 慢查询报警

对于调用数据服务的应用, 当出现慢日志或慢查询时, 会对项目相关人员进行报警

微服务

可以直接在系统中, 方便的对微服务进行操作, 未来支持直接挂载服务到微服务中心

分组负载均衡

分组负载均衡, 连接池复用, 统一协议转换, 高效和稳定

Serverless在同程的情况：开发、发布和运维 的效率

90%

初始化项目脚手架

申请git
安装框架
安装模块
调试脚手架

系统框架自带

80%

数据源、路由和日志

定义编写路由
数据源连接
拦截器、控制器
日志记录

MVC+系统进行配置

40%

编码开发

工具函数，常用类库
逻辑编写
程序调试
代码版本控制

云端编辑器编写代码

90%

发布和运维

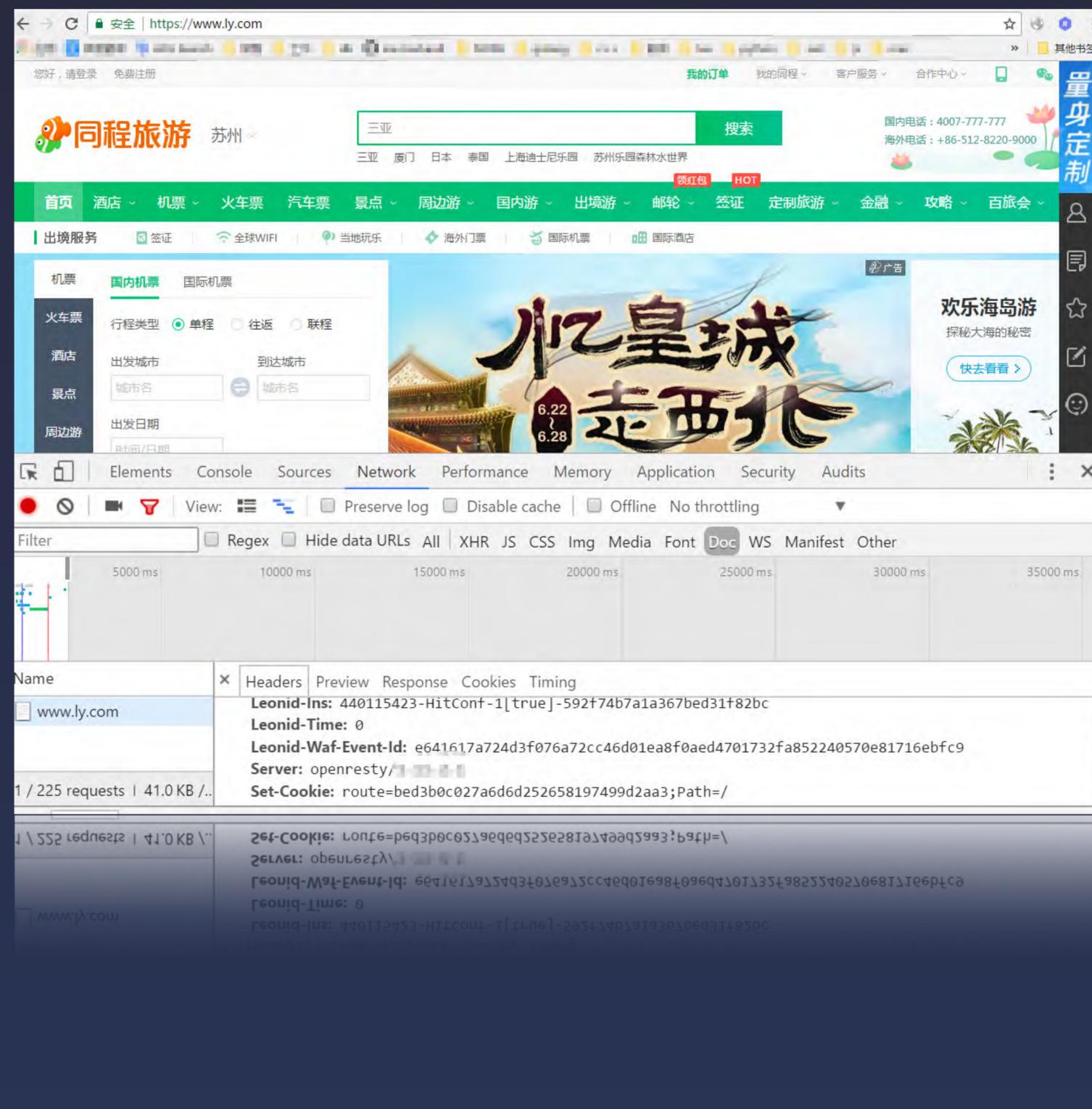
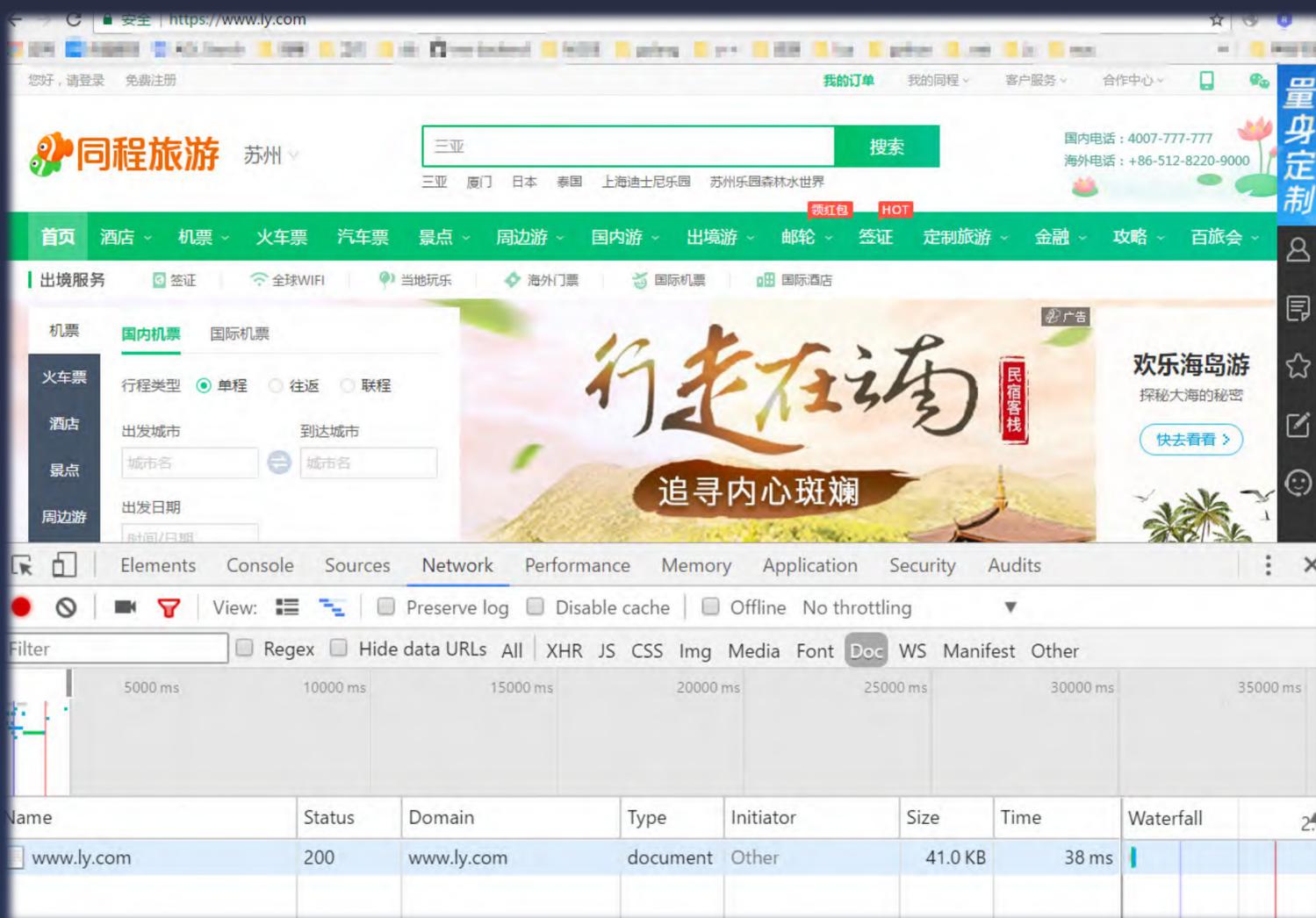
申请应用上线
配置运行环境
申请访问路径
应用运行情况和日志

应用系统一键发布

我们把什么放在Serverless平台跑了

Serverless在同程的情况：Web应用

- ★ 所有的网页
- ★ 所有的活动推广
- ★ 部分变化量大的后台



Serverless在同程的情况：轻型服务

- ★ 一些逻辑简单的业务服务
- ★ 一些逻辑变化快的业务服务
- ★ 大量的临时的小服务

```
ipDefender.lua  f1.lua  main.lua x  index.lua  utils.lua
29         else
30             console.error('设置黑名单失败 ip:' .. v.ip .. 'err:' .. err)
31         end
32     end
33 end
34 end
35 else
36     console.error("未获取到黑名单列表")
37 end
38 console.error("设置黑名单列表完成")
39
40 console.error("设置配置列表完成")
41 ]], 10)
42 if err then
43     console.error('计时器启动失败' .. err)
44 end
45
46 local requestText = req.raw_body
47 if requestText and requestText ~= '' then
48     local json, err = global.json_parse(requestText)
49     if err then
50         console.error('代理转发请求失败 json反序列化失败:' .. err)
51         utils.response(res, utils.RspType.Exception, utils.RspCode.RspCode_7000)
52     else
53         local resp, body, err = req.proxy()
54         if resp and resp.status >= 400 then
55             console.error('代理转发请求失败 response status:' .. resp.status)
56             utils.response(res, utils.RspType.Exception, utils.RspCode.RspCode_7000)
57         elseif err then
58             console.error('代理转发请求失败 error:' .. err)
59             utils.response(res, utils.RspType.Exception, utils.RspCode.RspCode_7000)
60         else
61             res.send(body)
62         end
63     end
64 else
65     console.error('代理转发请求失败 未传入请求体')
66     utils.response(res, utils.RspType.Exception, utils.RspCode.RspCode_7000)
67 end
68
69
70
71
72
73
```

Serverless在同程的情况：配套功能集成

- 价格实时计算服务

The screenshot displays a hotel booking interface for '北京金坛乐宾馆'. The main content area shows a table of room options with columns for name, bed type, breakfast, window, cancellation rules, and daily price. A '特价单人(无窗)' room is highlighted with a price of ¥222. To the right, a map shows the hotel's location in Beijing. Below the map, a list of nearby hotels is provided, including '北京君鑫酒店' and '飘HOME连锁酒店'.

名称	床型	早餐	窗户	取消规则	日均价	预订
[同程自营]	大床	双早	无窗	不可取消	¥222	预订 在线付
(双人入住) 代理	大床	双早	无窗	不可取消	¥229	预订 在线付
[标准价-双早]	单人床	双早	无窗	不可取消	¥240	预订 在线付
特价单人(无窗)	大床	无早	无窗	不可取消	¥273	预订 在线付
特价单人(无窗)	大床	无早	无窗	免费取消	¥288	预订 到店付

附近酒店

酒店名称	距离该酒店	价格
北京君鑫酒店	57米	¥199起
飘HOME连锁酒店(北京五棵松店)	124米	¥160起
北京毛林顺丰大酒店	454米	¥376起
海友酒店(北京定慧寺店)	549米	¥139起
北京长荣酒店	616米	¥209起
如家快捷酒店(北京定慧桥五路居地铁站店)	701米	¥226起

- 我们下一步在 Serverless 上要做什么

- ★ 加入更多的语言
- ★ Web化IDE能力提升
- ★ 更多的技能被配置化
- ★ 将自动测试系统并入



SPEAKER INTRODUCE

王晓波 首席架构师



晓波 

江苏 苏州

扫一扫上面的二维码图案，加我微信



THANKS

让创新技术推动社会进步

HELP TO BUILD A BETTER SOCIETY WITH
INNOVATIVE TECHNOLOGIES

Geekbang >

极客邦科技

InfoQ ueue

专注中高端技术人员的技术媒体



EGO EXTRA GEEKS' ORGANIZATION
NETWORKS

高端技术人员学习型社交平台



StuQ ueue
斯达克学院

实践驱动的 IT 教育平台

