



拍拍信

拍拍信数据服务（上海）有限公司

2017年10月

技术路上的沙漠穿越

-- 拍拍信在微服务化过程中的思考和实践

- 开始很美好 – 技术兴奋
- 过程很艰辛 – 痛并快乐
- 经常还要跪 – 挖坑爬坑
- 前路还很长 – 持续挑战



- **老板不理解**

- 有什么用？
 - 系统困境，不理不行，否则扛不住1年的发展
 - 业务实施复杂，系统耦合，容易触发事故，一旦发生，损失很大
 - 减少技术债的积累
- 要多久？
 - 短期计划和中长期计划结合，降低对业务的影响

- **其他部门的不理解**

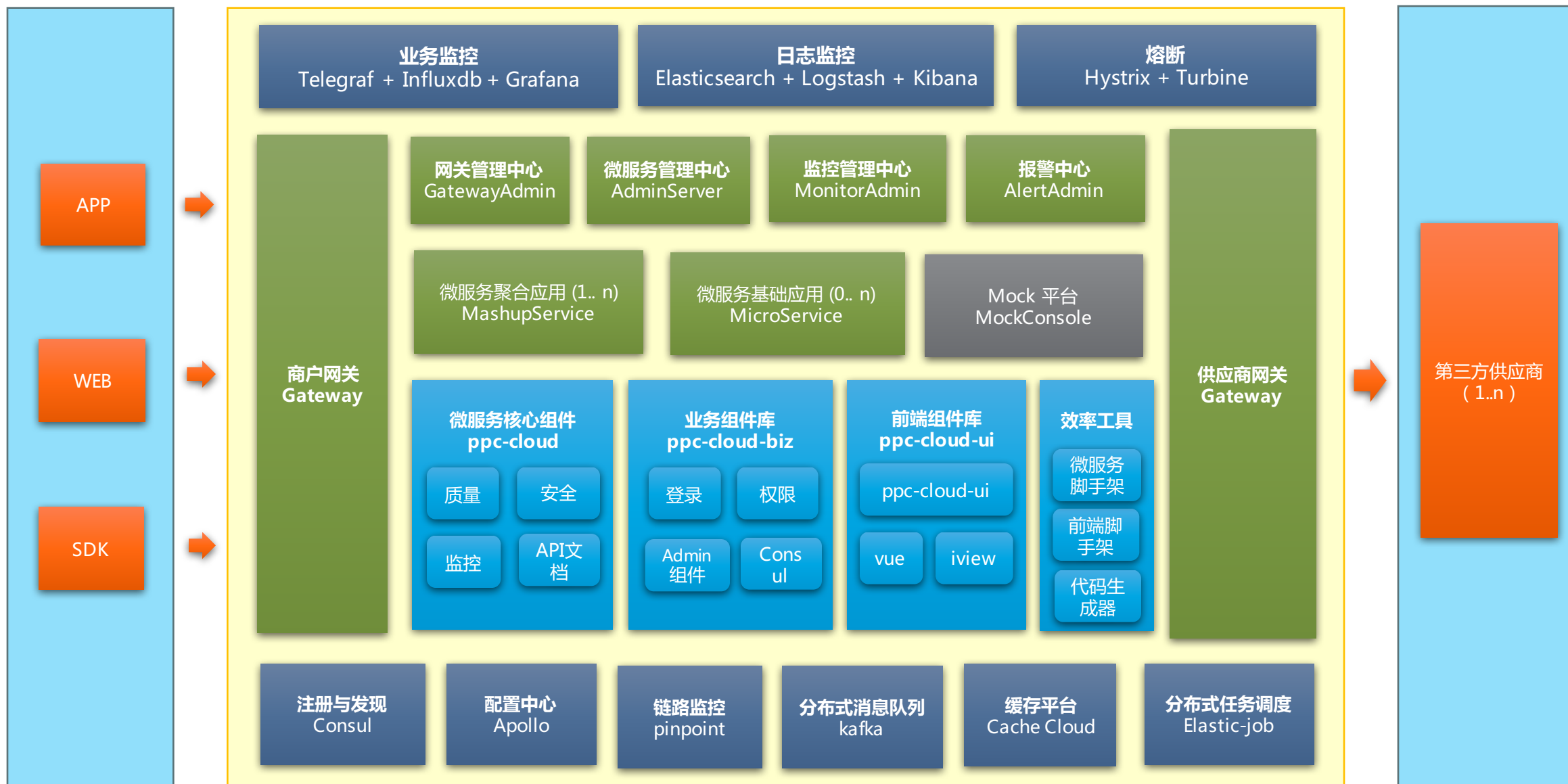
- 影响项目进度吗？
- 影响产品迭代吗？

- **开发和运维不理解**

- 我原来开发的蛮好的，效率挺高，改了这个问题我还需要学习上手
- 其他系统需要关联的开发，原来的东西要改

- **微服务改造需要实现的是整套完整体系**
 - 依赖的系统 – Spring Cloud生态体系
 - 服务中间件
 - 网关系统
 - 管理系统
 - 监控体系
 - 配置系统
 - 发布系统
- **体系必须逐步迭代完善，从小到大，从简到繁**
 - 每个阶段的产出必须要能够被呈现（难）
 - 架构的整体改变都可能影响很大
 - 合理切分上线功能
- **服务切换必须不能影响业务**
 - 合理规划设计
 - 测试：功能、性能、异常
 - 提前演练

拍拍信微服务框架体系



- 为什么选择Spring Cloud

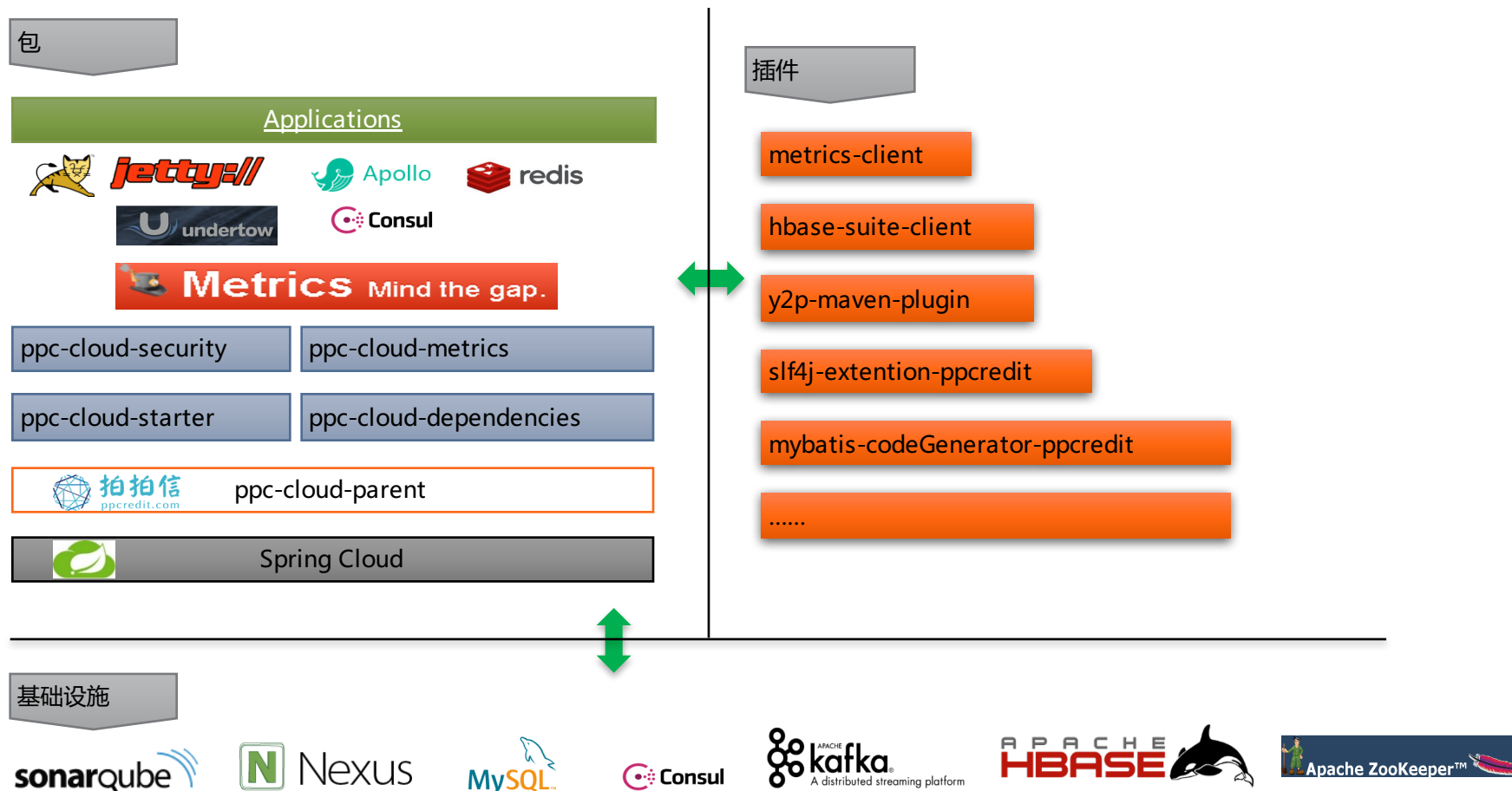
- 市面上常见的框架：dubbo/dubbox
- 轻量级，和现有Spring开发体系结合紧密，学习曲线平滑
- 社区生态体系完善，有足够的开源体系可以支持或者可供二次开发
- 和其他开源软件整合便捷
- 重点考虑：便捷、成本低、体系全！

- 人：
 - 思路的扭转
 - 学习（自驱和被动）
 - 提升（给未来的发展铺路）
- 事：
 - 中间件的支持（网关）
 - 发布系统支持
 - 微服务调用和httpclient接口同时支持
 - 缺陷：链路关系没法完全打通
 - 缺陷：记账体系不一致

- 拆分的原则
 - 不要为了拆而拆
 - 一个工程拆成100多个子项目见过么？
 - 不要为了拆而拆
 - 吃不准的先作为组件包
 - 独立出来服务的逻辑要独立

- **监控的不是系统，是生命线**
 - 项目开发完成不是终点，产品业务能否长久生存靠监控
 - 系统未成先想监控
 - 监控必须确保多个维度，想得越全越靠谱
 - 监控的维度尽可能保持一定交叉覆盖度
 - 悖论：监控系统的监控

- 脚手架用来作为微服务开发的基础组件



依赖于Zuul实现拍拍信网管系统

Home / Api 管理 / Api 列表 当前环境: test ▼

基本信息 — 定义API请求 — 定义API后端服务 — **4 定义返回结果**

```
{
  "resp_code": "api.resp.sys#success",
  "resp_msg": "调用接口成功",
  "resp_body": {
    "result": "$.success",
    "msg": {
      "sdkTokenStatusText": "$.sdkTokenSta
      "sdkTokenStatus": "$.sdkTokenStatus"
      "success": "$.success",
      "content": {
        "loginStatus": "$.content.loginStat
        "loginStatusText": "$.content.login
        "deviceId": "$.content.deviceId",
        "sdkToken": "$.content.sdkToken"
      },
      "statusCode": "$.statusCode"
    }
  },
  "resp_serial": "#.serialNumber"
}
```

重新输入JSON

字段名称	字段值	字段类型	操作
resp_code	api.resp.sys#sucr	string	
resp_msg	调用接口成功	string	
▼ resp_body		object	+ 新增属性
result	\$.success	string	
▼ msg		object	+ 新增属性
sdkTokenStatu	\$.sdkTokenStat	string	
sdkTokenStatu	\$.sdkTokenStat	string	
success	\$.success	string	
▼ content		object	+ 新增属性
loginStatus	\$.content.logir	string	
loginStatusTex	\$.content.logir	string	

• 转化配置方式

表达式前缀支持以下三种：

```
$.  
#.  
@.
```

#. 从运行上下文件中取java变量

```
例：  
{  
  ...  
  resp_serial: "#.serialNumber"  
  ...  
}
```

\$. 从源json中，取json path值

```
例：  
{  
  ...  
  deviceId: "$.content.deviceId",  
  sdkToken: "$.content.sdkToken"  
  ...  
}
```

@. 执行groovy脚本计算获取出参值

```
例：  
{  
  ...  
  method: "@.requestDto.getMethodName()",  
  callTime: "@.requestDto.getBeginTime()",  
  now: "@.com.ppcredit.cloud.gateway.util.DateUtil.getNowStringDate()"  
  ...  
}
```

获取入参值：

```
@.context.getRequest().getParameter("appkey");
```

- 完整实现整体微服务管理
 - VUE开发
 - 微服务大盘状态
 - 微服务注册和管理
 - 服务验证和测试
 - 服务日志管理

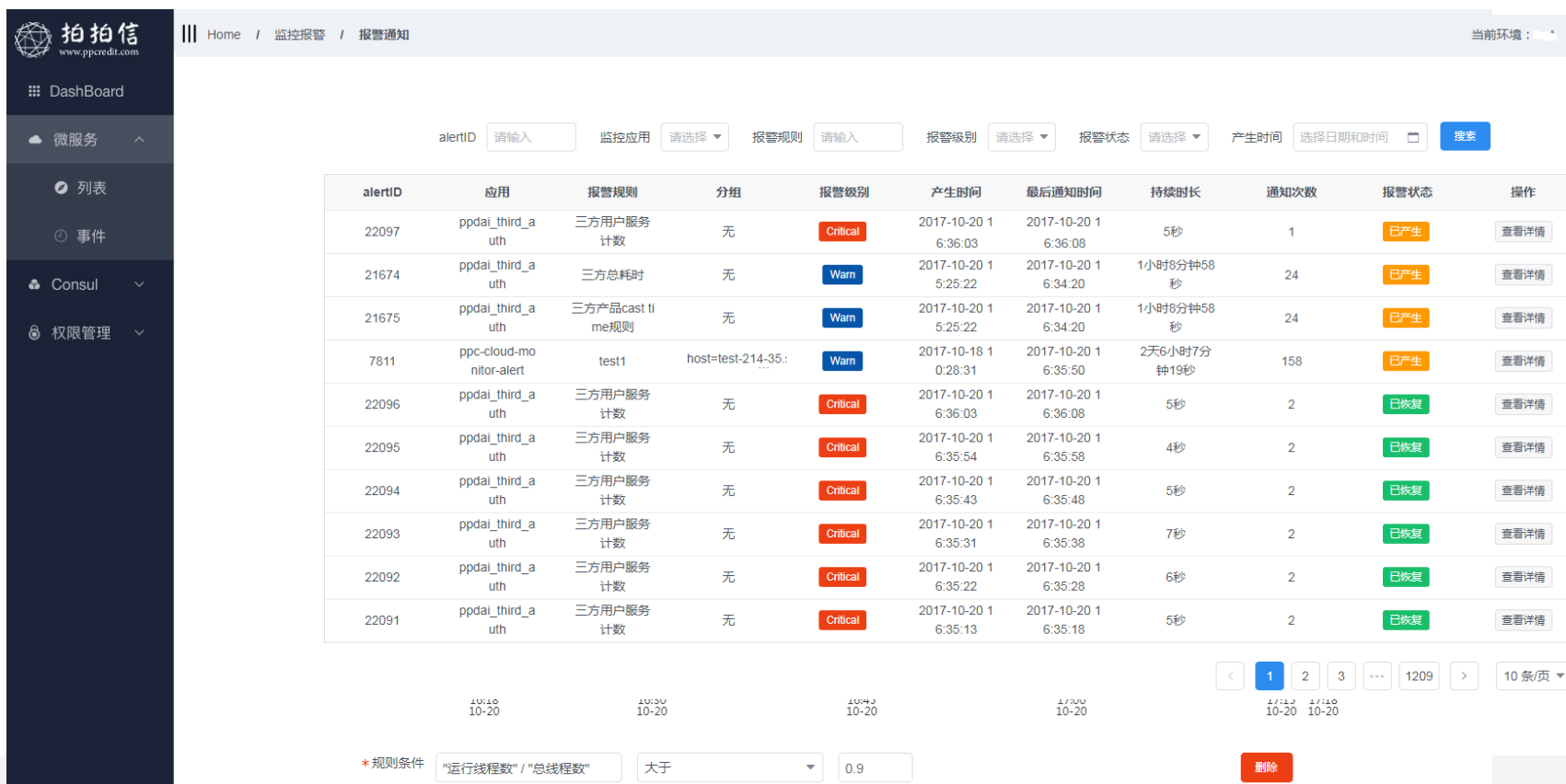


The screenshot displays a web interface for monitoring a microservice. The main content area shows the details for a server named 'UP ppc-cloud-admin-server (c38f3db9)'. The server address is listed as 'http://...'. The interface is organized into a tree view of JMX metrics:

- Domain: `ch.qos.logback.classic`
- Domain: `com.sun.management`
- Domain: `com.zaxxer.hikari`
- Domain: `java.lang`
- Domain: `java.nio`
- Domain: `java.util.logging`
- Domain: `JMImplementation`
- Domain: `jmx4peri`
- Domain: `jolokia`
- Domain: `metrics`
 - MBean: `jvm.memory.heap.init[]`
 - MBean: `jvm.memory.pools.Code-Cache.used[]`
 - MBean: `jvm.threads.timed_waiting.count[]`

The left sidebar contains navigation options: 详情, 环境, 监控指标, 线程, 日志级别, 日志, JMX (selected), 时间轴, 文档, and JVM.

- ELK – 支持原有服务
- 拍拍信监控系统
 - Metris打点

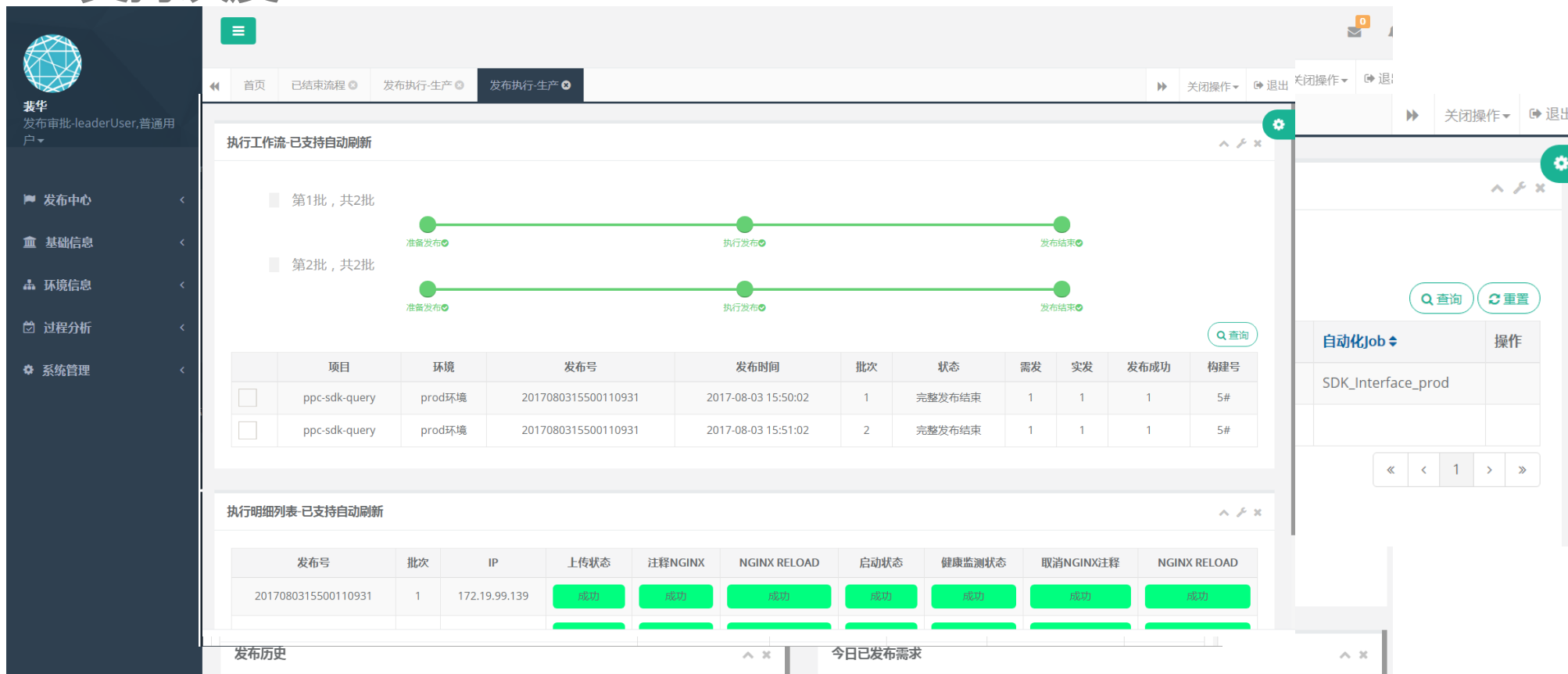


The screenshot displays the PPAI monitoring system's alert management interface. The top navigation bar includes the PPAI logo, a breadcrumb trail (Home / 监控报警 / 报警通知), and the current environment. A search bar at the top allows filtering by alertID, application, rule, level, status, and time. The main content is a table of alerts with columns for alertID, application, rule, group, level, time, duration, frequency, status, and actions. A sidebar on the left provides navigation for Dashboard, Microservices, Lists, Events, Consul, and Permissions. At the bottom, there is a filter rule configuration for '运行线程数 / 总线线程数' and a pagination control.

alertID	应用	报警规则	分组	报警级别	产生时间	最后通知时间	持续时长	通知次数	报警状态	操作
22097	ppdai_third_auth	三方用户服务计数	无	Critical	2017-10-20 16:36:03	2017-10-20 16:36:08	5秒	1	已产生	查看详情
21674	ppdai_third_auth	三方总耗时	无	Warn	2017-10-20 15:25:22	2017-10-20 16:34:20	1小时8分钟58秒	24	已产生	查看详情
21675	ppdai_third_auth	三方产品cast time规则	无	Warn	2017-10-20 15:25:22	2017-10-20 16:34:20	1小时8分钟58秒	24	已产生	查看详情
7811	ppc-cloud-monitor-alert	test1	host=test-214-35...	Warn	2017-10-18 10:28:31	2017-10-20 16:35:50	2天6小时7分钟19秒	158	已产生	查看详情
22096	ppdai_third_auth	三方用户服务计数	无	Critical	2017-10-20 16:36:03	2017-10-20 16:36:08	5秒	2	已恢复	查看详情
22095	ppdai_third_auth	三方用户服务计数	无	Critical	2017-10-20 16:35:54	2017-10-20 16:35:58	4秒	2	已恢复	查看详情
22094	ppdai_third_auth	三方用户服务计数	无	Critical	2017-10-20 16:35:43	2017-10-20 16:35:48	5秒	2	已恢复	查看详情
22093	ppdai_third_auth	三方用户服务计数	无	Critical	2017-10-20 16:35:31	2017-10-20 16:35:38	7秒	2	已恢复	查看详情
22092	ppdai_third_auth	三方用户服务计数	无	Critical	2017-10-20 16:35:22	2017-10-20 16:35:28	6秒	2	已恢复	查看详情
22091	ppdai_third_auth	三方用户服务计数	无	Critical	2017-10-20 16:35:13	2017-10-20 16:35:18	5秒	2	已恢复	查看详情

*规则条件 "运行线程数" / "总线线程数" 大于 0.9 删除

- 发布系统改造
 - 支持微服务和微服务
 - 支持灰度



发布系统改造后的界面展示，包含发布中心、基础信息、环境信息、过程分析和系统管理等模块。界面显示了发布执行的进度和详细列表。

执行工作流-已支持自动刷新

第1批, 共2批

第2批, 共2批

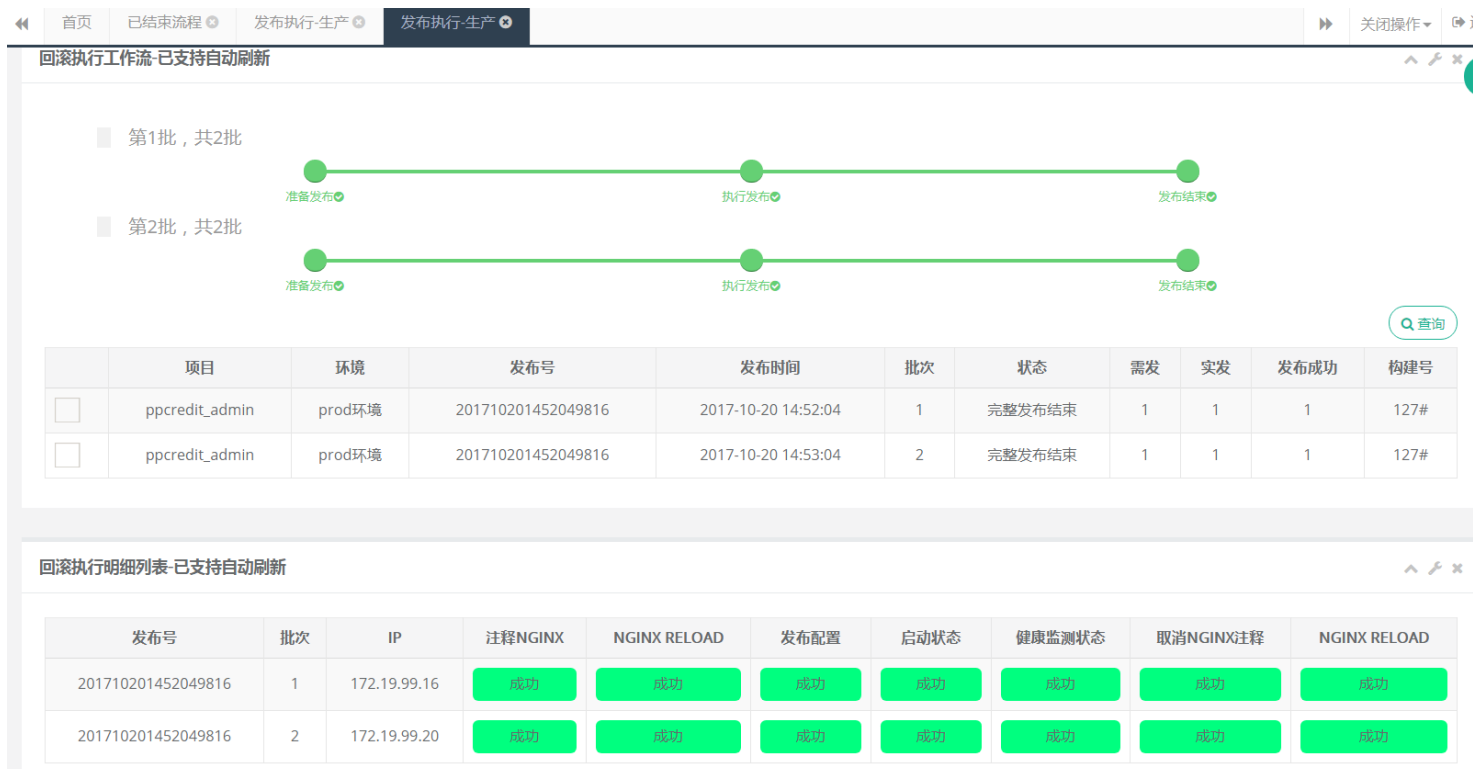
项目	环境	发布号	发布时间	批次	状态	需发	实发	发布成功	构建号
ppc-sdk-query	prod环境	2017080315500110931	2017-08-03 15:50:02	1	完整发布结束	1	1	1	5#
ppc-sdk-query	prod环境	2017080315500110931	2017-08-03 15:51:02	2	完整发布结束	1	1	1	5#

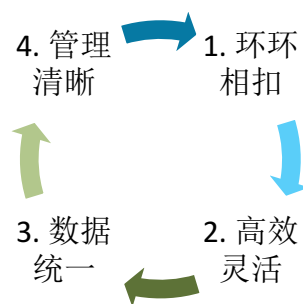
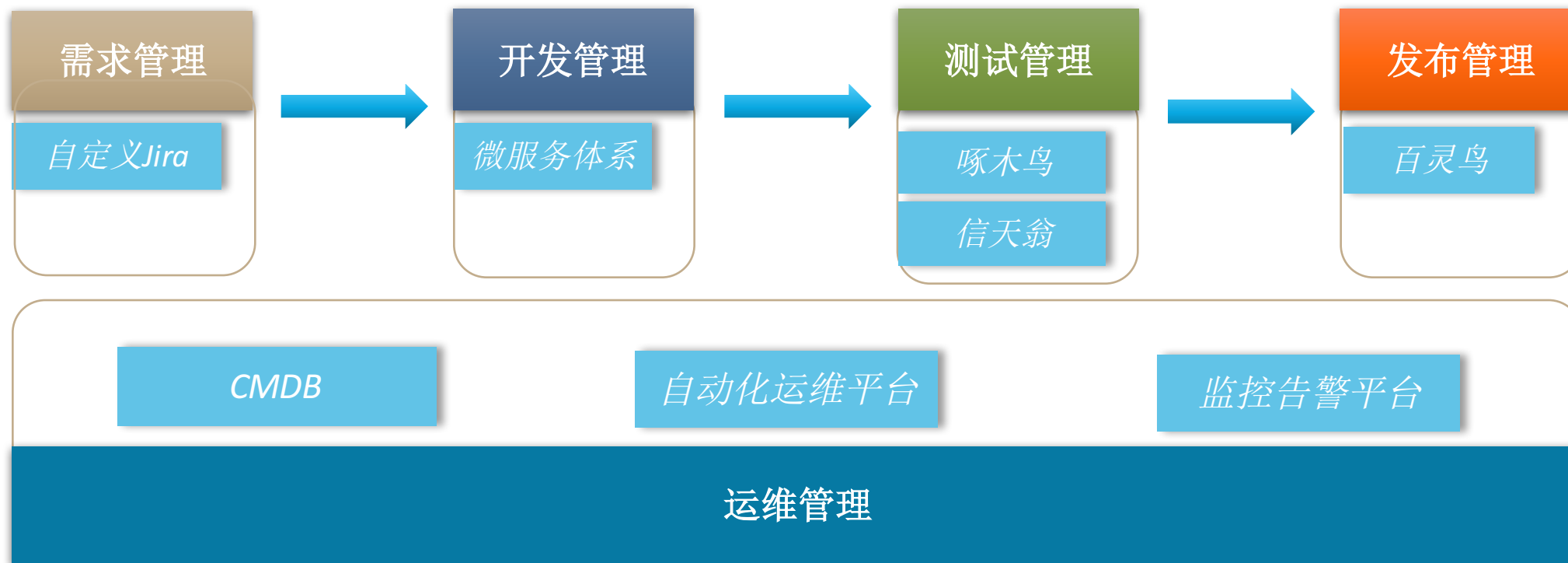
执行明细列表-已支持自动刷新

发布号	批次	IP	上传状态	注释NGINX	NGINX RELOAD	启动状态	健康监测状态	取消NGINX注释	NGINX RELOAD
2017080315500110931	1	172.19.99.139	成功	成功	成功	成功	成功	成功	成功

• 集成Apollo配置系统

- 支持yml本地开发配置，服务端统一读取Apollo配置中心
- 和发布系统整合，支持配置灰度发布





- 养成站在公司层面去思考问题的习惯
 - 技术服务于业务，不要纯粹的为了做技术
 - 用最合理的技术方式去解决业务
 - 帮助公司实现战略目标
- 技术管理者要有自己构建技术体系的方法论
 - 没有完美的架构，只有适合的架构，注重架构体系的演进
 - 时刻记得要还债
 - 对事故心怀敬畏
- 沟通和换位思考
 - 上级、平级、下级
 - 倾听
 - 规划发展
- 落地并应用于业务
 - 提前布局
 - 小步快跑

让我们在技术路上持续前行！