2016年年终总结

wangyifan

2016年主要做了四个项目:

* 分布式服务项目「Hydra」
* ffuse项目
* 基准测试项目「JMH」
* 参与分布式缓存项目

# 分布式服务项目「Hydra」

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 人员(小时) | 设计 | 客户端 | 管理中心 | 服务节点 | 维护 | 小计 |
| 王一帆 | 288.25 | 44 | 37.5 | 878.25 | 205.5 | 1453.5 |
| 丁国洋 |  | 22.25 | 539 |  |  | 561.25 |
| 总计 |  |  |  |  |  | 2014.75 |

## 目标

* 降低使用Hydra框架开发服务的难度,提高服务开发效率
* 优化服务发布流程
* 提供完善的服务治理，方便运维
* 替换掉OSGi

## 执行情况

* [x] Hydra3整体架构设计及技术预研
* [x] Hydra3客户端设计与开发及Hydra3管理中心页面设计
* [x] Hydra3管理中心详细设计与前台编码以及Hydra3服务详细设计
* [x] Hydra3服务开发及流程优化
* [x] Hydra3联调测试及文档整理
* [ ] FPay部署Hydra3 [进行中]
* [ ] 百卓部署Hydra3 [进行中]
* [ ] Hydra2维护 [进行中]

## 成绩

* 老版Hydra暴露了过多的技术细节，目前项目组使用Hydra开发服务过于复杂，影响开发效率。复杂的主要原因是项目组开发人员需要关注OSGi的相关配置，比如模块的导入导出，且相关配置需要在运行时才会提示是否有错误！在Hydra3中，摒弃了复杂的OSGi技术，并对技术细节进行了封装，实现了本地测试、debug和远程debug功能，极大的提高了开发效率。且在部署上提高了易用性，初步得到了项目组的认可。
* 老版Hydra的发布流程与运维发布流程无法很好的结合，老版Hydra默认基于命令行的发布方式，而运维通常使用shell拷贝方式进行方法，OSGi需要通过设置优先级来判断bundle的启动顺序，运维的拷贝方式无法进行优先级设置，可能导致bundle启动顺序错乱的问题。Hydra3的部署结合UMC进行发布，实现界面操作即可完成发布流程，同时UMC提供了监控功能，能方便的对Hydra3进行服务调用、运行情况的监控与报警。
* 老版Hydra缺少完善的服务治理，老版Hydra只有服务查看功能，而没有治理功能。Hydra3提供了完善的服务治理功能，包括服务升降级、客户端管理、服务端管理、权重管理、执行队列管理等功能。

## 问题

1. 设计阶段缺少对实际使用情况的考虑，在后续实际使用过程中，又针对项目组反馈的问题做了调整
2. Hydra3使用了Spring，对版本没有做隔离，项目组Spring版本只能跟Hydra3走，限制了一部分的灵活性
3. 项目开发完成后，推广使用有较长的空档期

# ffuse项目

## 目标

实现日志信息能自动推送到消息队列中，将「拉日志」改成「推日志」！

## 执行情况

* [x] 完成ffuse的架构设计
* [x] 完成ffuse的代码开发
* [ ] ffuse整体功能测试
* [ ] ffuse整体性能测试
* [ ] ffuse打包部署

## 成绩

目前日志分析平台通过tail的方式将日志写到消息队列中！而tail可能会失败，当失败后没有很好的恢复机制，导致最终消费端日志与实际日志不同步！使用ffuse通过用户空间的文件系统来替代tail的方式，当应用向文件中写日志时，自动触发推送消息队列功能，将日志信息自动推送到消息队列中！当消息队列出现问题时，可暂时将日志信息缓存，待消息队列恢复后，自动再将日志信息推送过去。保证本地日志信息与消费端日志信息同步！

## 问题

* 在开发层面，ffuse使用C语言开发，学习、开发、调试都花了较多的时间

# JMH项目

## 目标

提供部门使用的公共的、便利的基准测试平台！

## 执行情况

* [x] 完成基准测试平台的架构设计
* [ ] 完成基准测试平台的原型及流程设计
* [ ] 搭建开发框架
* [ ] 完成基准测试平台的开发
* [ ] 完成基准测试平台的测试
* [ ] 完成基准测试平台的部署

## 成绩

目前已完成基准测试平台的架构设计，正在进行原型及流程设计！后台基于Hydra3开发，直接复用Hydra3的节点功能，节省服务端的开发.

# 缓存项目

## 目标

提供公用的缓存平台！

## 执行情况

* [x] 梳理现有分布式缓存(Codis)的实现

# 个人成长与不足

2016年主要锻炼自己架构能力！通过分布式服务框架Hydra、ffuse以及基准测试框架JMH架构设计的锻炼，架构能力有了一定程度的提升，也对架构设计有了新的认识！

架构设计「是在限定的上下文下，做出最适合的决策的过程，决策的结果就是一个个的子系统、组件及接口」！与「程序员思维」不同的是，对于某个问题，「程序员思维」选择的是最短路径！架构设计选择的是最优路径！

今年也在刻意锻炼自己的思考能力！以前在实现功能时都是先实现、先跑起来，然后进行重构、调优！90%的行动+10%的思考！今年特意克制自己，在实现一个功能前，先梳理整个流程，确认这个流程是否有问题？有问题的话，该怎么改？这个流程怎么样来实现？现有技术是否可以实现？如果不可以，是否有替代技术？整个流程确定后，才动手！这个时候的行动就是个相对机械的过程，行动效率要比以前高很多！编码完成，基本一次就能Run起来！

演讲方面，今年进行了四次内部分享，一次针对初中级开发人员的培训！对内部分享、培训等场合来说，基本没什么大问题！今年还参加了公司的技术大会！整个体验和内部分享及培训完全不同，这方面的经验还很不足，需要进一步的学习与练习！

# 一些想法和建议

当初从ttnet到平台架构部，一是，希望能在技术方面有成长和学习架构方面的知识。这个目的已经达到了。二是，在ttnet有很多繁琐的项目管理工作要做，没法安心思考技术实现方面的问题！这个问题也解决了。三是，ttnet已经从当初刚收购时的重点项目，慢慢变成了边缘项目，因为基本没有盈利能力！今年南京技术团队差点就解散了！感觉个人也没有能力逆转这个趋势，算是逃避吧！

来到平台架构部以后，发现了类似的问题！今年感觉更强烈了！做完的项目没人用！ttnet虽然不受重视了，但是还有用户，做的功能是产品和需求定下来的，做了肯定会上线！在平台东西做完了，也就做完了！能不能上线，有没有人用是另一方面的事情！从这个角度看的话，其实成就感比ttnet还小！

个人觉得目前横向部门的处境比较尴尬！除了已经公司范围使用的ffs,搜索,bi等平台，其它未在公司内部广泛使用的、或正在开发的项目在推广层面都有比较大的阻力！这点我也比较理解，当初在ttnet的时候，平台也来推广过hydra，当时业务正紧，只是做了了解，而没有实际使用！纵向部门有自己的业务压力，如果最终平台的项目做完了，只在内部消化，项目意义大打折扣！

我觉得出现这种情况的原因有两个！一是，横纵向利益点不同！纵向的重点在业务，技术能服务于自身业务即可，不管用什么技术，不影响业务就可以，绩效考核也是看业务。而横向考虑的是公司各个项目的通用性，考核点也主要在技术上，比如通用性、扩展性等。二是，都是技术人员，能自己搞定的就自己搞定了。部门之间扯皮的时间，还不如自己闷头搞！

个人认为的一个解决方案是利益绑定！如果一个项目是横纵向合作项目，则考核级别远大于单独的纵向或横向项目。比如公司层面的创新评判偏向合作项目，合作项目的相关人员绩效高于非合作项目等！从利益层面驱动纵向和横向的合作以及跨部门开发人员合作的积极性！

# 2017年项目规划和个人规划

**项目规划：**

* 协助支付、百卓升级Hydra的工作
* 根据项目组反馈优化Hydra3
* 优化Hydra3异常结构，方便问题追踪
* 完成JMH的开发测试与部署
* 完成ffuse的开发与测试
* 协助完成缓存平台的技术预研与开发工作

**个人规划：**

* 继续深入技术与架构设计
* 能做更深层次的分享，不单单在技术层面，还能分享个人的想法
* 扩大知识范围,与技术融合