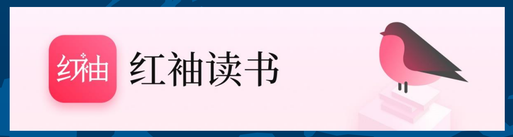
TARS 持续集成

如何在企业中做智能化测试

阅文集团-俞慧涛



个人介绍

俞慧涛（Timmy）阅文集团后台开发专家，阅文集团用户中心架构负责人，负责整个集团的核心用户相关的服务。曾就职于盛大游戏，负责盛大游戏“传奇手游”数据sdk的开发和架构设计工作。腾讯TARS项目贡献者，对java服务、消息系统有丰富的经验。

什么是TARS?

1、TARS是基于名字服务使用Tars协议的高性能RPC开发框架， 可以快速的帮助企业搭建分布式应用。

2、TARS是将腾讯内部使用的微服务架构TAF（Total Application Framework）多年的实践成果总结而成的开源项目。 可以让开发者聚焦业务逻辑，让运营更高效，一切尽在掌握。3、在阅文内部，有100多个业务、1.6多万台服务器上运行使用。

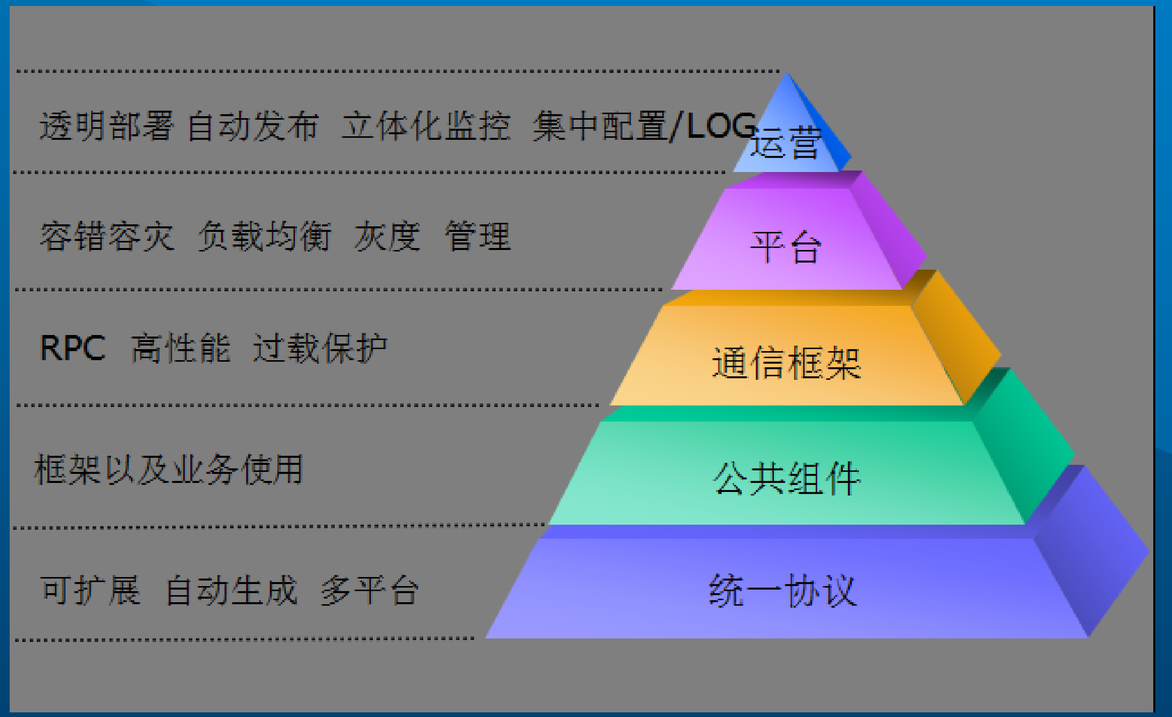
4、最高的服务QPS可以达一秒数百万

什么是TARS?

1、**通信协议统一**、拥有自己的协议语言，语言兼容不再是问题

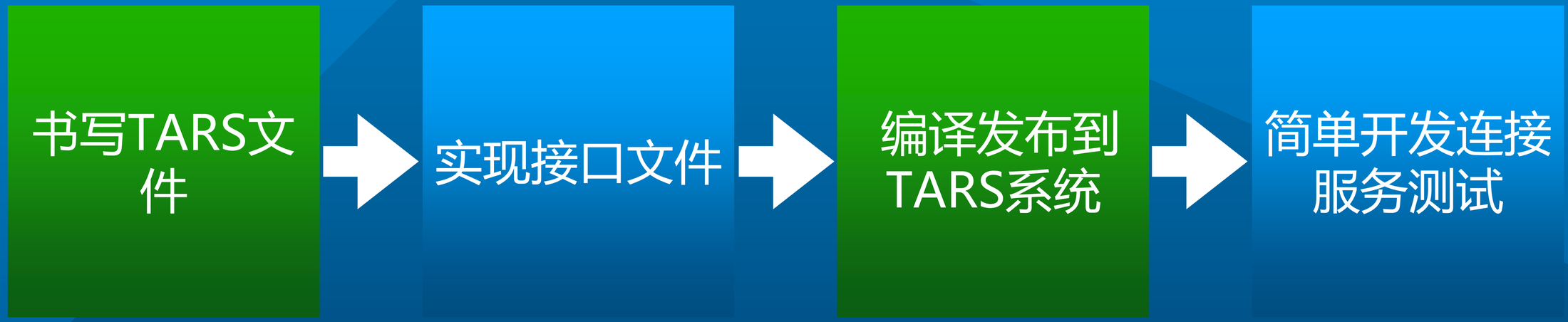
3、**开发省力**。在开发层面，服务的容灾、容错、等不需要再考虑，在tars层都处理掉， 没有各种架构上的烦恼

2、**可运维程度高**，运维只需要关注日常的服务部署、发布、配置、监控、调度管理等操作。



为什么用TARS？ 和HTTP有啥不同？

现在的TARS开发流程



现在开发流程中存在的问题?

现在开发流程中存在的问题?

1、微服务上万个， 业务多

2、新兴业务中业务逻辑页面众多，逻辑错纵复杂，容易出问题

3、新的业务一直上，如何保证服务的**质量、性能**?

如何对应?

引入更科学的开发流程

架构师拆分需求

设计IDL 接口文件

提交到代码文件

部署至jenkins 周期跑

根据IDL文件，做测试用例

书写对应的测试用例

生成测试报告和接口测试用例

引入流程后就没有问题了吗？

引入流程后就没有问题了吗?

1、 IDL描述的语言的接口，如何测试?
2、 如何统一的管理IDL文件?
3、各个服务的依赖如何处理?

1、 如何解决?

1、研发了基于IDL的服务路由服务(TARSProxy)
2、 结合IDL的管理平台（ TARSExtension)

1、研发了基于IDL的服务路由服务(TARSProxy)

1、 拥有简单的TARS 的路由功能
2、所有的请求格式都是Json格式
3、附合系统要求

1、研发了基于IDL的服务路由服务(TARSProxy)

1、 所有的TARS请求格式都统一化
2、自动化的寻址功能
3、QPS单个测试节点可以支持12万QPS

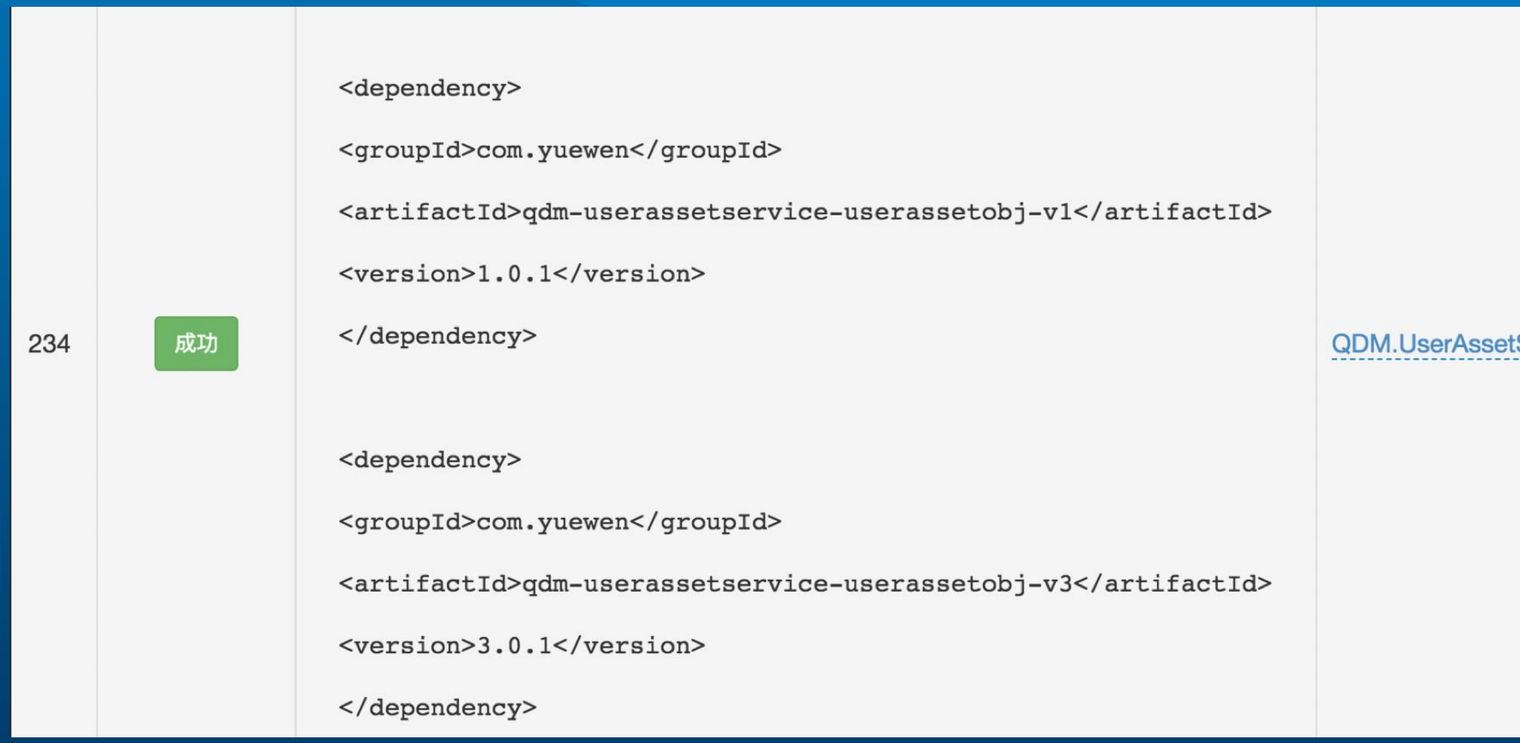

```
{
   "method":"getGuidByNickname",
   "return_result":0,
   "proxy_env":"国内Tars213环境",
   "invoker_server_ip":"127.0.0.1:10036",
   "return_ret":0,
   "time":7,
   "outParam":{
      "value":{
         "common":{
            "returnCode":-41000,
            "returnMessage":"DCACHE empty!",
            "reviewId":"",
            "batchOutStatusList":[

            ]
         },
         "data":[

         ]
      }
   },
   "return_taf_message":"服务器端处理成功",
   "servantName":"UserBase.YuewenNicknameAvatarServer.YuewenNicknameAvatarServant"
}
```

2、 结合IDL的管理平台（ TARSExtension）

1、 **依赖调用方便**，提交你的代码后，即可生成POM坐标，无缝和maven集成

2、 **自动生成**HTTPRouter接口

3、 支持其他的测试应用（ Yapi、 Postman）



2、 简单的测试用例

```
@org.junit.Test
public void testPromission() throws IOException, InterruptedException {
    HttpInvokerClient httpInvokerClient = new HttpInvokerClient(Environment.TARS213);
    UserPromissionServiceServant servantPrx = httpInvokerClient
            .createProxy( servantName: "UserBase.YuewenPromissionServer.YuewenPromissionServant",
                    UserPromissionServiceServant.class);
    Holder<java.util.List<UserPromissionEntity>> list = new Holder<>();
    Holder<YuewenPromissionOutParam> outParams = new Holder<>();
    servantPrx.queryPromissionByBook( cbid: 350L, outParams, list);
    org.junit.Assert.assertTrue( condition: outParams.getValue().getCode() > 0);
    System.out.println(JsonUtils.toJson(outParams));
}
```

2、 输出是否正确

```
Holder<java.util.List<UserPromissionEntity>> list = new Holder<>();
Holder<YuewenPromissionOutParam> outParams = new Holder<>();
servantPrx.queryPromissionByBook( cbid: 350L, outParams, list);
org.junit.Assert.assertTrue( condition: outParams.getValue().getCode() > 0);
System.out.println(JsonUtils.toJson(outParams));
}
```


收益

1、 解决QA测试时，各个TARS的http协议不一致的问题
2、 节省开发调用的流程，发布IDL语言时，可以自动化部署到Maven(依赖jenkins)
3、 提升服务质量和性能
4、提升服务测试效率

以后的发展

1、引入Service Mesh架构， 节省发布环节
2、 AI测试各个接口，更智的数据分析

Question?


TARS
Github.com/TarsCloud

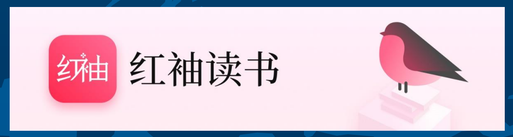

Tencent
open
腾讯开源


阅文集团
CHINA READING


QQ阅读
海量原著 想读就读


起点读书
阅文集团旗下移动阅读软件


WEBNOVEL
FICTIONAL STORIES HUB


红袖读书

扫码加入3.23上海源创会 TARS交流群

扫码添加TARS小助手为好友，拉你入群

（PS：申请时请备注“源创会”）

TARS Github.com/TarsCloud

Tencent open 腾讯开源

阅文集团 CHINA READING

QQ阅读 海量原著 想读就读

起点读书 阅文集团旗下移动阅读软件

WEBNOVEL FICTIONAL STORIES HUB

红袖读书