



(10) 授权公告号 CN 114490318 B

(45) 授权公告日 2024. 07. 23

(21) 申请号 202111562841.8

G06F 8/41 (2018.01)

(22) 申请日 2021.12.20

H04L 67/61 (2022.01)

H04L 67/133 (2022.01)

(65) 同一申请的已公布的文献号

申请公布号 CN 114490318 A

(43) 申请公布日 2022.05.13

(73) 专利权人 江苏康众汽配有限公司

地址 210019 江苏省南京市建邺区嘉陵江
东街8号科技创新综合体B3幢2单元18
楼

(56) 对比文件

杨子涵. 基于硬件虚拟化的客户虚拟机内部
执行环境隔离机制.《cnki优秀硕士学位论文
全文库 信息科技辑》.2022,(第01期),第I137-5
页.

审查员 彭凤鸣

(72) 发明人 徐海锋 朱二平 巩夫建

(74) 专利代理机构 南京智转慧移知识产权代理
有限公司 32649

专利代理师 邱兴天

(51) Int. Cl.

G06F 11/36 (2006.01)

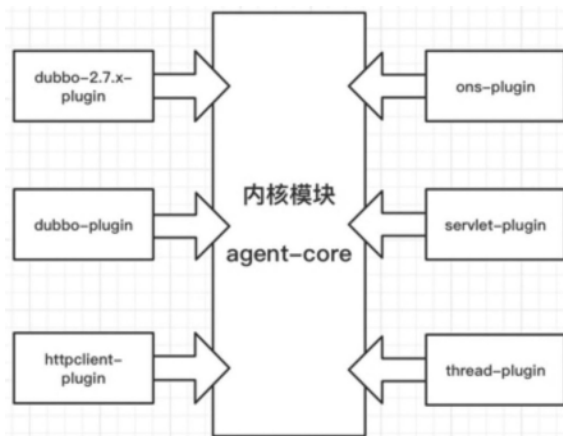
权利要求书1页 说明书4页 附图3页

(54) 发明名称

一种基于字节码增强实现弱隔离的环境治
理系统和方法

(57) 摘要

本发明公开了一种基于字节码增强实现弱隔离的环境治理系统和方法,属于软件测试技术领域。该系统包括字节码增加的代码实现模块、基于软件实现的弱隔离模块、分布式调用链路 agent-trace-id 透传实现模块、请求路由优先级控制模块。本发明将应用的环境分为项目环境和基础环境,项目环境可以依赖基础环境,项目环境只部署项目相关变更的应用,基础环境部署全量的应用;通过字节码增强的方式,实现应用的 Dubbo、MQ、Http、多线程各个场景调用路由隔离,并且自定义实现数据透传协议,实现分布式链路场景的数据透传,确保涉及多个应用的调用链路能够准确、安全的隔离调用;可用于多个应用调用场景的环境弱隔离。



1. 一种基于字节码增强实现弱隔离的环境治理系统,其特征在于,包括字节码增强的代码实现、基于软件实现的弱隔离模块、分布式调用链路agent-trace-id透传实现模块、请求路由优先级控制模块;字节码增强的代码实现为对每种调用工具的隔离实现可插拔,可快速部署,可复用;基于软件实现的弱隔离模块用于将开发环境、测试环境都细分为项目环境和基础环境;分布式调用链路agent-trace-id透传实现模块在基于ThreadLocal技术条件下,完成整个分布式调用链路的agent-trace-id的传递,实现一个请求的每次调用的路由优先级都是可控的;请求路由优先级控制模块用于在请求调用时,找到对应的请求路由优先级控制代码,完成请求的路由优先级控制;

字节码增强的代码实现是按照微内核的模式构建,并且包含一个core模块和多个plugin模块;

每个项目都有对应的项目环境,项目环境只部署当前项目中有代码或配置变更的应用;基础环境部署所有应用的环境,所部署的应用代码和配置都与线上环境一致,保证服务的稳定性;项目环境的应用可以依赖基础环境的应用完成请求调用,即项目环境的应用可以调用基础环境的应用,通过基础环境的应用兜底,完成整个调用链路的正常调用;

在每个应用内部调用阶段,都将agent-trace-id存储在调用上下文中,便于随时获取,在两个调用阶段之间,通过具体的调用技术框架的自定义参数功能,所述具体的技术框架包括Dubbo、MQ、Http、多线程,完成agent-trace-id的传递,以此完成整个分布式调用链路的agent-trace-id的传递;

请求路由优先级控制模块是在Java应用内或应用间进行请求调用时,包括Dubbo调用、MQ调用、Http调用、多线程调用,通过每种技术的调用源码研究,找到对应的请求路由优先级控制代码,按照特定的路由控制策略实现,完成请求的路由优先级控制。

2. 一种基于字节码增强实现弱隔离的环境治理方法,应用于如权利要求1所述系统实现,其特征在于,包括以下步骤:

(1) 立项阶段,先确定立项的项目代号,然后确定立项的项目包含的应用,在立项平台中完成立项操作,该过程会完成应用配置信息新建,以线上环境的配置为基线来创建,并且会自动创建该应用所有的项目环境GID,同时会生成该应用的项目环境访问域名;

(2) 部署阶段,按照项目环境标识来做应用部署,确保环境标识、项目标识、应用标识都正确通过ID的形式传参到Java应用中,并且以javaagent的方式启动,具体的agent路径配置实现生成agent.jar;

(3) 调用阶段,对应前端的入口,需要使用对应应用的项目环境访问域名来请求,只要确保入口请求的访问域名正确,后续的请求链路会正常完成调用,agent.jar会自动完成链路过程中agent-trace-id的透出和请求路由处理。

一种基于字节码增强实现弱隔离的环境治理系统及方法

技术领域

[0001] 本发明属于软件测试技术领域,具体涉及一种基于字节码增强实现弱隔离的环境治理系统及方法。

背景技术

[0002] Byte Buddy是一个代码生成和操作库,用于在Java应用程序运行时创建和修改Java类,而无需编译器的帮助。除了Java类库附带的代码生成实用程序外,Byte Buddy还允许创建任意类,并且不限于实现用于创建运行时代理的接口。此外,Byte Buddy提供了一种方便的API,可以使用Java代理或在构建过程中手动更改。

[0003] Byte Buddy相比其他字节码操作库有如下优势:无需理解字节码格式,即可操作,简单易行的API能很容易操作字节码。支持Java任何版本,库轻量,仅取决于Java字节代码解析器库ASM的访问者API,它本身不需要任何其他依赖项。比起JDK动态代理、cglib、Javassist,Byte Buddy在性能上具有优势。

[0004] 一个项目周期内,应用的环境资源会被占用,尤其在开发环境、测试环境所占用的时间比例超过80%,那么一个应用隶属于多个项目的场景,会出现环境资源冲突的问题。

发明内容

[0005] 针对现有技术中存在的问题,本发明要解决的技术问题在于提供一种基于字节码增强实现弱隔离的环境治理系统及方法。通过字节码增强的方式,实现应用的弱隔离,配合一定的规范和约定,解决多项目并行场景下的环境资源冲突问题。

[0006] 为了解决上述问题,本发明所采用的技术方案如下:

[0007] 一种基于字节码增强实现弱隔离的环境治理系统,包括字节码增加的代码实现模块、基于软件实现的弱隔离模块、分布式调用链路agent-trace-id透传实现模块、请求路由优先级控制模块;

[0008] 字节码增加的代码实现模块为对每种调用工具的隔离实现可插拔,可快速部署,可复用;

[0009] 基于软件实现的弱隔离模块用于将开发环境、测试环境都细分为项目环境和基础环境;

[0010] 分布式调用链路agent-trace-id透传实现模块在基于ThreadLocal技术条件下,完成整个分布式调用链路的agent-trace-id的传递,实现一个请求的每次调用的路由优先级都是可控的;

[0011] 请求路由优先级控制模块用于在请求调用时,找到对应的请求路由优先级控制代码,完成请求的路由优先级控制。

[0012] 所述基于字节码增强实现弱隔离的环境治理系统,字节码增加的代码实现模块是按照微内核的模式构建,并且包含一个core模块和多个plugin模块。

[0013] 所述基于字节码增强实现弱隔离的环境治理系统,每个项目都有对应的项目环

境,项目环境只部署当前项目中有代码或配置变更的应用;基础环境部署所有应用的环境,所部署的应用代码和配置都与线上环境一致,保证服务的稳定性;项目环境的应用可以依赖基础环境的应用完成请求调用,即项目环境的应用可以调用基础环境的应用,通过基础环境的应用兜底,完成整个调用链路的正常调用。

[0014] 所述基于字节码增强实现弱隔离的环境治理系统,在每个应用内部调用阶段,都将agent-trace-id存储在调用上下文中,便于随时获取,在两个调用阶段之间,通过具体的调用技术框架的自定义参数功能,所述技术框架包括Dubbo、MQ、Http、多线程,完成agent-trace-id的传递,以此完成整个分布式调用链路的agent-trace-id的传递。

[0015] 所述基于字节码增强实现弱隔离的环境治理系统,请求路由优先级控制模块是在Java应用内或应用间进行请求调用时,包括Dubbo调用、MQ调用、Http调用、多线程调用,通过每种技术的调用源码研究,找到对应的请求路由优先级控制代码,按照特定的路由控制策略实现的,完成请求的路由优先级控制。

[0016] 一种基于字节码增强实现弱隔离的环境治理方法,流程包括以下步骤:

[0017] (1) 立项阶段,先确定立项的项目代号,然后确定立项的项目包含的应用,在立项平台中完成立项操作,该过程会完成应用配置信息新建,以线上环境的配置为基线来创建,并且会自动创建该应用所有的项目环境GID,同时会生成该应用的项目环境访问域名;

[0018] (2) 部署阶段,按照项目环境标识来做应用部署,确保环境标识、项目标识、应用标识都正确通过-D的形式传参到Java应用中,并且以javaagent的方式启动,具体的agent路径配置实现生成agent.jar;

[0019] (3) 调用阶段,对应前端的入口,需要使用对应应用的项目环境访问域名来请求,只要确保入口请求的访问域名正确,后续的请求链路会正常完成调用,agent.jar会自动完成链路过程中agent-trace-id的透出和请求路由处理。

[0020] 有益效果:与现有的技术相比,本发明的优点包括:

[0021] (1) 本发明通过字节码增强的方式,实现应用的弱隔离,配合一定的规范和约定,解决多项目并行场景下的环境资源冲突问题;有很高的通用性,很容易在Java应用中推行并使用。

[0022] (2) 本发明将应用的环境分为项目环境和基础环境,项目环境可以依赖基础环境,项目环境只部署项目相关变更的应用,基础环境部署全量的应用;通过字节码增强的方式,实现应用的Dubbo、MQ、Http、多线程各个场景调用路由隔离,并且自定义实现数据透传协议,实现分布式链路场景的数据透传,确保涉及多个应用的调用链路能够准确、安全的隔离调用。本发明用于多个应用调用场景的环境弱隔离。

附图说明

[0023] 图1为微内核模式实现原理图;

[0024] 图2为Http调用隔离实现原理图;

[0025] 图3为Dubbo调用隔离实现原理图;

[0026] 图4为MQ调用隔离实现原理图;

[0027] 图5为多线程调用隔离实现原理图;

[0028] 图6为环境治理方法流程图。

具体实施方式

[0029] 为使本发明的上述目的、特征和优点能够更加明显易懂,下面结合具体实施例对本发明的具体实施方式做详细的说明。

[0030] 实施例1

[0031] 一种基于字节码增强实现弱隔离的环境治理系统,包括字节码增加的代码实现模块、基于软件实现的弱隔离模块、分布式调用链路agent-trace-id透传实现模块、请求路由优先级控制模块,具体为:

[0032] (1) 字节码增加的代码实现模块,针对字节码增强技术做了一层抽象,按照微内核模式(微内核模式实现原理如图1所示),构建了core模块,和多个plugin模块,实现对每种调用工具的隔离实现可插拔,可快速部署,可复用。

[0033] (2) 基于软件实现的弱隔离模块,是将开发环境、测试环境都细分为项目环境和基础环境;每个项目都有对应的项目环境,项目环境只部署当前项目中有代码或配置变更的应用;基础环境部署所有应用的环境,所部署的应用代码和配置都与线上环境一致,保证服务的稳定性;项目环境的应用可以依赖基础环境的应用完成请求调用,即项目环境的应用可以调用基础环境的应用,通过基础环境的应用兜底,完成整个调用链路的正常调用。

[0034] (3) 分布式调用链路agent-trace-id透传实现模块,是基于ThreadLocal技术,针对分布式调用链路,实现调用上下文的构建,在每个应用内部调用阶段,都将agent-trace-id存储在调用上下文中,便于随时获取,在两个调用阶段之间,通过具体的调用技术框架的自定义参数功能,所述技术框架包括Dubbo调用、MQ调用、Http调用、多线程调用,完成agent-trace-id的传递,以此完成整个分布式调用链路的agent-trace-id的传递;

[0035] 对于Http调用隔离实现(原理如图2所示),给每个应用都配置一个项目环境对应的访问域名,并且域名按照http://{环境标识}-{项目标识}-{应用标识}.domain.com的统一格式,在Http请求到底的应用服务中,通过读取请求头信息中的host数据,识别出环境标识、项目标识、应用标识,来唯一标识一个请求源,应用服务内部使用请求源信息加上时间戳信息,在加上一个4位的随机数,生成一个全局唯一的agent-trace-id,作为请求源信息的载体,完成后续的请求中源信息的透传。并将agent-trace-id存在Http请求的处理线程的ThreadLocal中,确保当前调用的在当前JVM中都能够获取到agent-trace-id。

[0036] 对于Dubbo调用隔离实现(原理如图3所示),在服务消费端Comsumer,通过ThreadLocal获取到agent-trace-id,将agent-trace-id以自定义参数设置在请求的attachments参数中;在服务提供端Provider,通过从请求的attachments参数中获取到agent-trace-id,解析出源环境信息,在服务路由环节,过滤出目标group对应的服务,如果目标group的服务不存在,则以基础环境的服务为兜底,来完成服务调用。

[0037] 对于MQ调用隔离实现(原理如图4所示),在消息生产端Producer,通过ThreadLocal获取到agent-trace-id,将agent-trace-id以自定义参数设置消息的userProperties中;在消息消费端Comsumer,通过从消息的userProperties参数中获取agent-trace-id,解析出源环境信息。

[0038] 非基础环境:若当前应用所处的环境和源环境信息不匹配,则直接将消息置为消费成功;

[0039] 基础环境:若当前应用所处的环境和源环境信息不匹配,并且源环境对应的应用

不存在,则直接将消息置为消费成功;

[0040] 若应用所处的环境和源环境信息匹配,则进行正常的业务处理,正常返回。

[0041] 对于多线程调用隔离实现(原理如图5所示),通过对ThreadPoolExecutor类execute、submit方法进行字节码增强,实现主线程和子线程间的agent-trace-id透传,并且约定所有的应用都需要使用ThreadPoolExecutor来做多线程场景调用的代码实现。

[0042] (4) 请求路由优先级控制模块,是在Java应用内或应用间进行请求上述调用时,通过每种技术的调用源码研究,找到对应的请求路由优先级控制代码,按照特定的路由控制策略实现,完成请求的路由优先级控制模块。

[0043] 一种基于字节码增强实现弱隔离的环境治理方法,流程包括以下步骤:

[0044] (1) 立项阶段,先确定立项的项目代号,然后确定立项的项目包含的应用,在立项平台中完成立项操作,该过程会完成应用配置信息新建,以线上环境的配置为基线来创建,并且会自动创建该应用所有的项目环境GID,同时会生成该应用的项目环境访问域名;

[0045] (2) 部署阶段,按照项目环境标识来做应用部署,确保环境标识、项目标识、应用标识都正确通过-D的形式传参到Java应用中,并且以javaagent的方式启动,具体的agent路径配置本发明实现生成的agent.jar。

[0046] (3) 调用阶段,对应前端的入口,需要使用对应应用的项目环境访问域名来请求,只要确保入口请求的访问域名正确,后续的请求链路会正常完成调用,本发明实现生成的agent.jar会自动完成链路过程中的agent-trace-id的透出和请求路由处理。

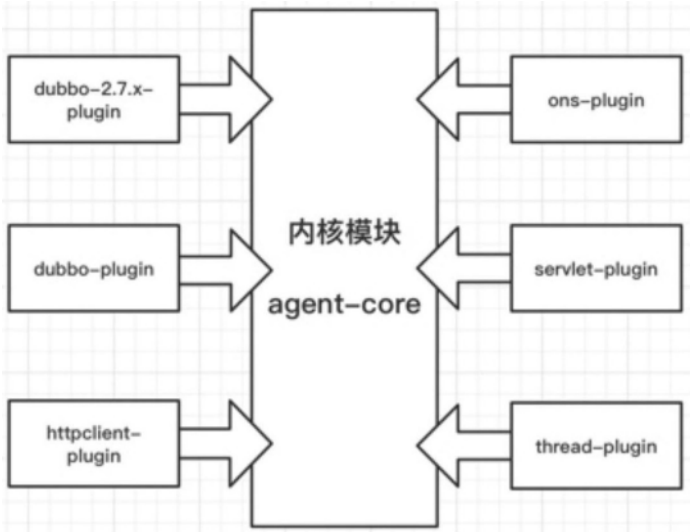


图1

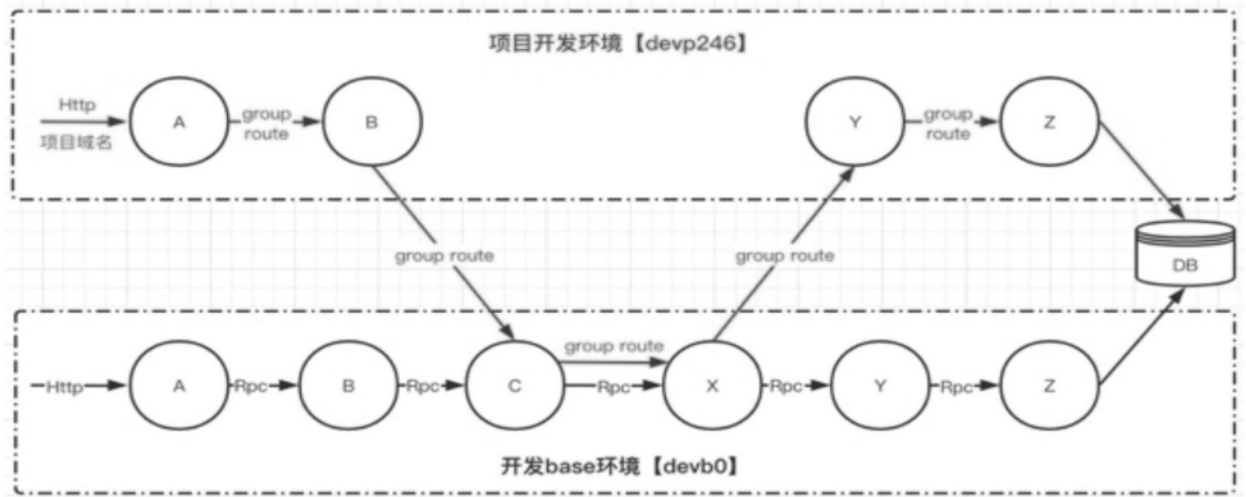


图2

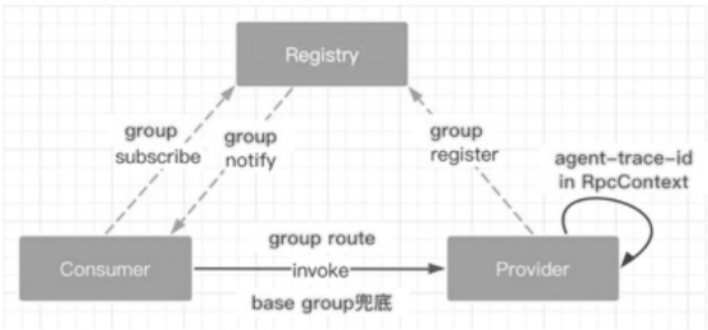


图3

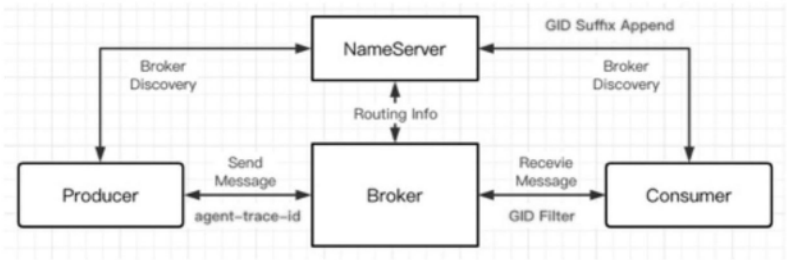


图4

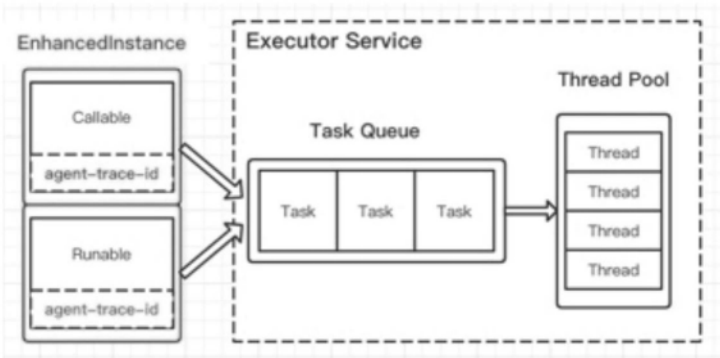


图5

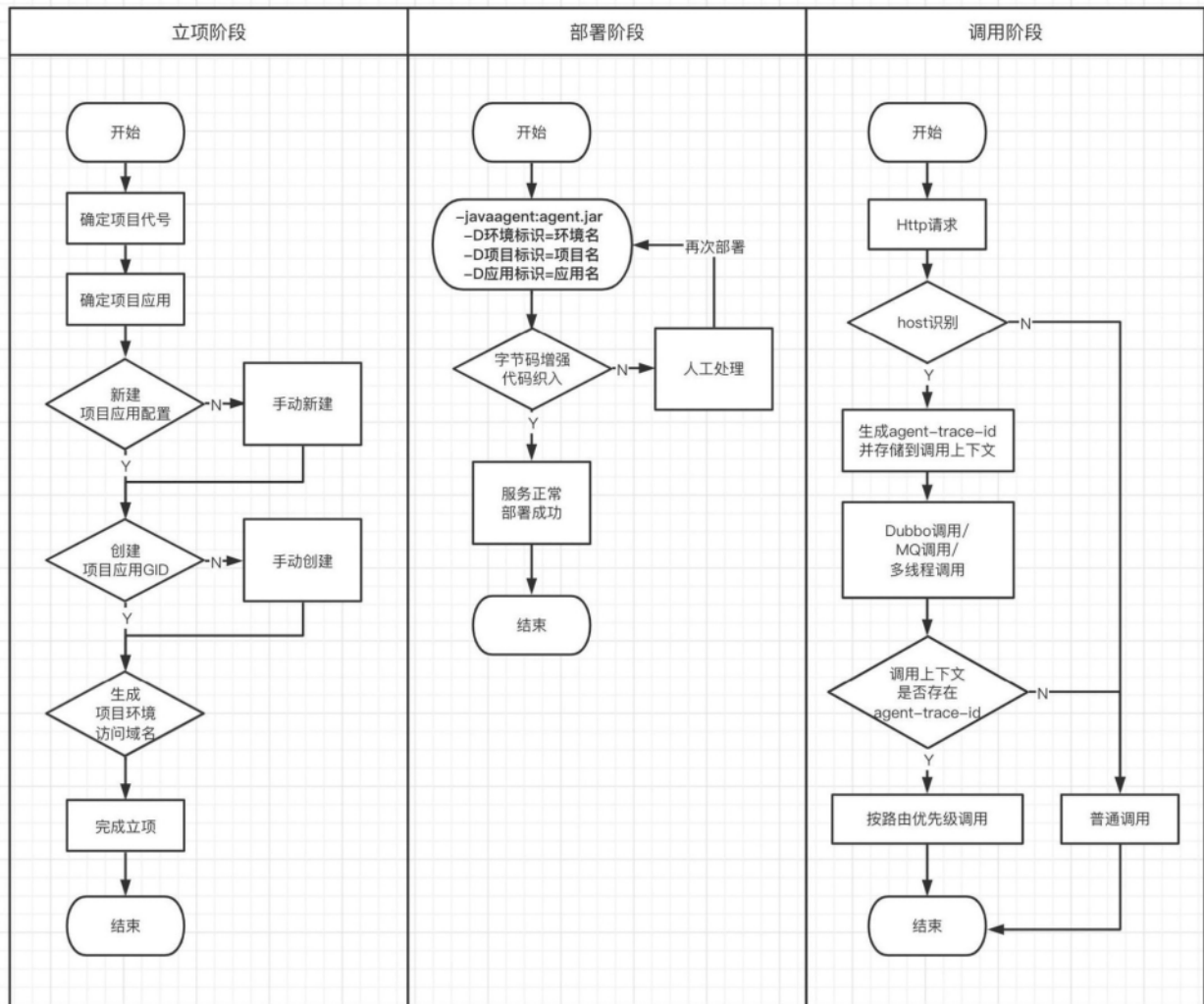


图6