(19)国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 115883353 A
(43)申请公布日 2023.03.31

(21)申请号 202211572564.3

(22)申请日 2022.12.08

(71)申请人 普元信息技术股份有限公司
地址 201203 上海市浦东新区中国(上海)自由贸易试验区碧波路456号4楼

(72)发明人 倪坚 刘相 高阳

(74)专利代理机构 上海智信专利代理有限公司 31002
专利代理师 王洁 郑暄

(51)Int.Cl.
H04L 41/0803(2022.01)
H04L 41/22(2022.01)
H04L 41/0894(2022.01)
H04L 12/66(2006.01)
H04L 67/133(2022.01)

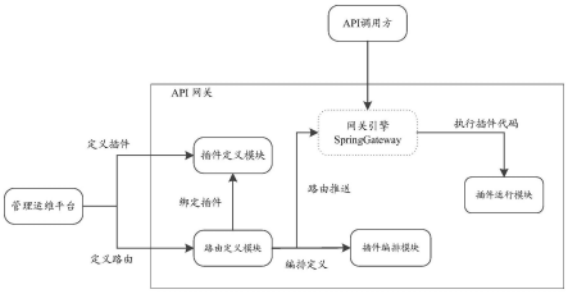
权利要求书2页 说明书10页 附图4页

(54)发明名称

信创环境下基于Spring Cloud Gateway网关路由插件编排和执行系统

(57)摘要

本发明涉及一种信创环境下基于Spring Cloud Gateway网关路由上插件编排和执行的系统,其中,所述的系统包括:插件定义模块,用于根据网关插件开发使用规范在管理运维平台上进行标准插件的定义和开发;路由定义模块,用于根据路由定义为管理运维平台定义不同类型的API请求建立路由规则,并将API请求按照设定的规则绑定到特定路由上;插件编排模块,与所述的路由定义模块相连接,用于在路由上进行插件的选择,并从插件库中选择需要执行的插件以及多个插件之间执行的顺序;以及插件运行模块,用于根据路由上插件的编排定义,按照顺序执行每个插件的逻辑。采用了本发明的该信创环境下基于Spring Cloud Gateway网关路由上插件编排和执行的系统,解决了用户基于Spring cloud gateway网关的图形化路由配置方式。



CN 115883353 A

1.一种信创环境下基于Spring Cloud Gateway网关路由上插件编排和执行的系统,其特征在于,所述的系统包括:

插件定义模块,用于根据网关插件开发使用规范在管理运维平台上进行标准插件的定义和开发;

路由定义模块,用于根据路由定义为管理运维平台定义不同类型的API请求建立路由规则,并将API请求按照设定的规则绑定到特定路由上;

插件编排模块,与所述的路由定义模块相连接,用于在路由上进行插件的选择,并从插件库中选择需要执行的插件以及多个插件之间执行的顺序;以及

插件运行模块,用于根据路由上插件的编排定义,按照顺序执行每个插件的逻辑。

2.根据权利要求1所述的信创环境下基于Spring Cloud Gateway网关路由上插件编排和执行的系统,其特征在于,所述的插件定义模块具体为:

设置有数据持久化组件,所述的数据持久化组件与所述的管理运维平台的插件管理模块相连接,并通过所述的插件管理模块保存生成的数据,进行插件定义以及资源包的绑定;

元数据管理单元,与所述的管理运维平台的前端表单渲染模块相连接,并通过所述的前端表单渲染模块获取插件元数据。

3.根据权利要求1所述的信创环境下基于Spring Cloud Gateway网关路由上插件编排和执行的系统,其特征在于,所述的路由定义模块具体为:

设置有路由管理单元,所述的路由管理单元用于配置路由相关信息,包括断言、代理地址、插件,并保存成Json描述文件,再将相应的描述文件通过持久化组件保存到关系数据库表中,并同步保存到Redis中;网关引擎通过订阅Redis消息动态获取路由信息,并将此路由对象加载到内存中,从而实现动态路由的能力。

4.根据权利要求3所述的信创环境下基于Spring Cloud Gateway网关路由上插件编排和执行的系统,其特征在于,所述的插件编排模块具体为:

所述的插件编排模块与所述的路由管理单元相连接,用于在特定路由上进行插件的编排处理,并通过图形化的方式从所有启用的插件中选择需要在路由上执行的插件,调整插件的执行顺序,读取插件的运行元数据信息,通过表单渲染界面将元数据渲染成表单,配置运行元数据的值,保存插件编排的描述并将其绑定到路由上。

5.根据权利要求4所述的信创环境下基于Spring Cloud Gateway网关路由上插件编排和执行的系统,其特征在于,所述的插件编排模块具体进行以下处理:

(1)进行插件选择;

(2)判断是否进行自定义页面处理,如果是,则直接读取Vue页面;否则,直接读取插件元数据;

(3)通过表单渲染界面进行页面渲染处理;

(4)将需要启用的插件进行运行参数的配置处理。

6.根据权利要求5所述的信创环境下基于Spring Cloud Gateway网关路由上插件编排和执行的系统,其特征在于,所述的页面渲染处理,具体包括:

针对自定义的页面,渲染时直接根据插件名称到前端plugins目录下找到对应文件名的vue文件,在界面上以div的方式显示;

针对插件元数据,渲染时根据元数据定义的数据类型、字段名称、描述、输入提示、校验

规则、是否必填的相关信息，来动态显示页面。

7.根据权利要求5所述的信创环境下基于Spring Cloud Gateway网关路由上插件编排和执行的系统，其特征在于，所述的插件运行模块具体进行以下处理：

当请求到达网关后，进行特定路由的匹配处理，并由Spring cloud gateway内核按照路由上插件编排的顺序调用对应的插件代码。

信创环境下基于Spring Cloud Gateway网关路由插件编排和执行系统

技术领域

[0001] 本发明涉及计算机软件技术领域，尤其涉及API网关技术领域，具体是指一种信创环境下基于Spring Cloud Gateway网关路由上插件编排和执行的系统。

背景技术

[0002] 随着企业数字化转型，很多企业的应用系统技术架构由传统的单体应用转变到微服务架构，在微服务下，应用之间的调用都是通过API进行，系统间的集成也由传统的ESB的方式转到API网关的方式。

[0003] API网关是企业在构建基于云原生应用、微服务架构系统下必备的基础软件，目前市场上主流的微服务网关有Kong、Zuul、Spring Cloud Gateway等等，作为企业API接入接出的关键枢纽，API网关承担了很多像统一身份认证、访问日志、流量控制、负载均衡、请求防重、协议转换、灰度、数据缓存、数据加密、数据脱敏等通用功能的实现，这些功能通常都以网关插件的方式提供，但是目前API网关提供的插件无法提供细粒度的控制，比如无法配置某个特定API返回的数据进行脱敏或者加密，要实现这个功能需要专业的开发人员进行扩展开发支持，同样对于不同API，有的需要进行流量控制，有的需要记录访问日志信息，有的需要进行协议转换，有的需要指定插件的执行顺序，等等，目前网关无法提供插件的选择以及执行顺序的控制，很多特定使用场景用户无法灵活配置，需要开发人员进行扩展开发，增加了复杂度。

[0004] 目前Spring Cloud Gateway网关没有插件的概念，主要是通过过滤器来实现流量控制、身份认证等功能，组件提供全局过滤器和网关过滤器基类，由开发人员基于过滤器基类来扩展开发、程序打包、介质部署来完成，对于所有进入网关的请求都需要进行的操作，比如访问记录、请求防重，通过定义全局过滤器来实现，对于指定在特定路由上进行的操作，比如流量控制、灰度等，通过定义网关过滤器来实现，再通过配置文件配置的方式在路由上配置相关的路由过滤器。对于API细粒度的控制通过在路由上配置断言，选择符合条件的API访问请求。

[0005] 请参阅图1所示，其为Spring Cloud Gateway的请求处理过程，主要是过滤器的执行机制以及断言对于路由的选择过程：

[0006] spring.cloud.gateway.routes[0].uri＝lb:http://servicedemo-auto

[0007] spring.cloud.gateway.routes[0].predicates[0]＝Path＝/**

[0008] spring.cloud.gateway.routes[0].filters[0].name＝RedirectTo

[0009] spring.cloud.gateway.routes[0].filters[0].args.status＝306

[0010] spring.cloud.gateway.routes[0].filters[0].args.url＝https://www.baidu.com/

[0011] 以上是Spring Cloud Gateway的配置方式，提供了断言的配置、过滤器的配置以及最终的代理地址。

[0012] 通过以上Spring cloud gateway的原理可以知道,目前Spring Cloud Gateway网关在企业应用中存在很多方面的不足:比如通过配置文件的方式配置,需要配置人员对语法格式、每个配置项的含义要理解,稍有偏差会导致网关无法启动,另外通过配置方式每次增加路由配置后,需要重新启动网关才能生效;通过过滤器的方式进行扩展开发,无法进行功能的热插拔,也无法让运维人员知道网关中提供了哪些扩展能力,在进行路由配置时需要依赖开发人员或者详细的操作说明;无法在网关运行过程中根据业务场景动态调整配置限流、脱敏等措施,比如某个API需要在高峰时间段10点至20点对特定对象的调用进行限流,或者需要根据调用方对API返回信息中的敏感字段进行脱敏;无法对某个路由上配置的过滤器执行顺序进行动态调整,运维管理人员没法直观知道请求进到网关后的执行逻辑,在路由上配置了身份认证、流量控制等过滤器,到底是先执行了身份认证还是执行了流量控制。

发明内容

[0013] 本发明的目的是克服了上述现有技术的缺点,提供了一种支持路由动态配置的信创环境下基于Spring Cloud Gateway网关路由上插件编排和执行的系统。

[0014] 为了实现上述目的,本发明的信创环境下基于Spring Cloud Gateway网关路由上插件编排和执行的系统如下:

[0015] 该信创环境下基于Spring Cloud Gateway网关路由上插件编排和执行的系统,其主要特点是,所述的系统包括:

[0016] 插件定义模块,用于根据网关插件开发使用规范在管理运维平台上进行标准插件的定义和开发;

[0017] 路由定义模块,用于根据路由定义为管理运维平台定义不同类型的API请求建立路由规则,并将API请求按照设定的规则绑定到特定路由上;

[0018] 插件编排模块,与所述的路由定义模块相连接,用于在路由上进行插件的选择,并从插件库中选择需要执行的插件以及多个插件之间执行的顺序;以及

[0019] 插件运行模块,用于根据路由上插件的编排定义,按照顺序执行每个插件的逻辑。

[0020] 较佳地,所述的插件定义模块具体为:

[0021] 设置有数据持久化组件,所述的数据持久化组件与所述的管理运维平台的插件管理模块相连接,并通过所述的插件管理模块保存生成的数据,进行插件定义以及资源包的绑定;

[0022] 元数据管理单元,与所述的管理运维平台的前端表单渲染模块相连接,并通过所述的前端表单渲染模块获取插件元数据。

[0023] 较佳地,所述的路由定义模块具体为:

[0024] 设置有路由管理单元,所述的路由管理单元用于配置路由相关信息,包括断言、代理地址、插件,并保存成Json描述文件,再将相应的描述文件通过持久化组件保存到关系数据库表中,并同步保存到Redis中;网关引擎通过订阅Redis消息动态获取路由信息,并将此路由对象加载到内存中,从而实现动态路由的能力。

[0025] 较佳地,所述的插件编排模块具体为:

[0026] 所述的插件编排模块与所述的路由管理单元相连接,用于在特定路由上进行插件

的编排处理，并通过图形化的方式从所有启用的插件中选择需要在路由上执行的插件，调整插件的执行顺序，读取插件的运行元数据信息，通过表单渲染界面将元数据渲染成表单，配置运行元数据的值，保存插件编排的描述并将其绑定到路由上。

[0027] 较佳地，所述的插件编排模块具体进行以下处理：

[0028] (1)进行插件选择；

[0029] (2)判断是否进行自定义页面处理，如果是，则直接读取Vue页面；否则，直接读取插件元数据；

[0030] (3)通过表单渲染界面进行页面渲染处理；

[0031] (4)将需要启用的插件进行运行参数的配置处理。

[0032] 较佳地，所述的页面渲染处理，具体包括：

[0033] 针对自定义的页面，渲染时直接根据插件名称到前端plugins目录下找到对应文件名的vue文件，在界面上以div的方式显示；

[0034] 针对插件元数据，渲染时根据元数据定义的数据类型、字段名称、描述、输入提示、校验规则、是否必填的相关信息，来动态显示页面。

[0035] 较佳地，所述的插件运行模块具体进行以下处理：

[0036] 当请求到达网关后，进行特定路由的匹配处理，并由Spring cloud gateway内核按照路由上插件编排的顺序调用对应的插件代码。

[0037] 采用了本发明的该信创环境下基于Spring Cloud Gateway网关路由上插件编排和执行的系统，通过将过滤器的概念抽象成插件，对插件进行定义管理、元数据配置，使运维人员可以将基于过滤器扩展开发的功能作为插件进行管理，对插件进行热插拔、启用停用等操作，同时通过图形化的方式对路由进行配置，可以配置断言以及相关插件选择，可以通过拖拽的方式对所选插件进行编排，在执行时按照编排的顺序执行，支持了路由的动态配置生效，满足特定业务场景下的需求，增强了网关的适应性。解决了用户基于Spring cloud gateway网关的图形化路由配置方式，使路由配置及时生效，实现了让用户动态管理过滤器的能力，进行过滤器热插拔，同时由于采用了编排方式，任意组合各种插件，满足了用户很多实际业务场景下的复杂需求，提升了网关的适应性、易用性。

附图说明

[0038] 图1为Spring Cloud Gateway的请求处理过程的流程图。

[0039] 图2为本发明的信创环境下基于Spring Cloud Gateway网关路由上插件编排和执行的系统的结构示意图。

[0040] 图3为本发明的插件定义模块的结构示意图。

[0041] 图4为本发明的路由定义模块的结构示意图。

[0042] 图5为本发明的插件编排模块的结构示意图。

[0043] 图6为本发明进行路由配置时的定义路由描述文件的格式示意图。

[0044] 图7为本发明在路由定义上进行插件的编排的示意图。

具体实施方式

[0045] 为了能够更清楚地描述本发明的技术内容，下面结合具体实施例来进行进一步的

描述。

[0046] 在详细说明根据本发明的实施例前，应该注意到的是，在下文中，术语“包括”、“包含”或任何其他变体旨在涵盖非排他性的包含，由此使得包括一系列要素的过程、方法、物品或者设备不仅包含这些要素，而且还包含没有明确列出的其他要素，或者为这种过程、方法、物品或者设备所固有的要素。

[0047] 请参阅图2所示，该信创环境下基于Spring Cloud Gateway网关路由上插件编排和执行的系统，其中，所述的系统包括：

[0048] 插件定义模块，用于根据网关插件开发使用规范在管理运维平台上进行标准插件的定义和开发；

[0049] 路由定义模块，用于根据路由定义为管理运维平台定义不同类型的API请求建立路由规则，并将API请求按照设定的规则绑定到特定路由上；

[0050] 插件编排模块，与所述的路由定义模块相连接，用于在路由上进行插件的选择，并从插件库中选择需要执行的插件以及多个插件之间执行的顺序；以及

[0051] 插件运行模块，用于根据路由上插件的编排定义，按照顺序执行每个插件的逻辑。

[0052] 请参阅图3所示，作为本发明的优选实施方式，所述的插件定义模块具体为：

[0053] 设置有数据持久化组件，所述的数据持久化组件与所述的管理运维平台的插件管理模块相连接，并通过所述的插件管理模块保存生成的数据，进行插件定义以及资源包的绑定；

[0054] 元数据管理单元，与所述的管理运维平台的前端表单渲染模块相连接，并通过所述的前端表单渲染模块获取插件元数据。

[0055] 在实际应用当中，插件定义模块让使用人员在管理平台上定义插件，比如数据脱敏插件、数据加解密插件，填写相关的插件属性，上传该插件对应的代码部署包(因为是基于SpringCloudGateway扩展开发的插件，因此这边的部署包是基于AbstractGatewayFilterFactory基类扩展再编译打包成jar文件)，并进行插件定义和jar包的绑定，再通过插件配置界面定义插件的元数据，比如插件启动时需要依赖哪些外面环境，以及插件在运行时需要的配置参数，配置完成后启用插件，插件启用后就可以在路由定义时选择插件。

[0056] 请参阅图4所示，作为本发明的优选实施方式，所述的路由定义模块具体为：

[0057] 设置有路由管理单元，所述的路由管理单元用于配置路由相关信息，包括断言、代理地址、插件，并保存成Json描述文件，再将相应的描述文件通过持久化组件保存到关系数据库表中，并同步保存到Redis中；网关引擎通过订阅Redis消息动态获取路由信息，并将此路由对象加载到内存中，从而实现动态路由的能力。

[0058] 在实际应用当中，路由定义模块主要让用户能够通过图形化界面定义Spring Cloud Gateway的Router对象，并配置路由相关信息，包括断言、代理地址、插件，保存成Json描述文件，再将描述文件保存到关系数据库表中并同步保存到Redis，网关通过订阅Redis消息来动态获取路由信息，并将此路由对象加载到内存中，实现动态路由的能力。

[0059] 作为本发明的优选实施方式，所述的插件编排模块具体为：

[0060] 所述的插件编排模块与所述的路由管理单元相连接，用于在特定路由上进行插件的编排处理，并通过图形化的方式从所有启用的插件中选择需要在路由上执行的插件，调

整插件的执行顺序,读取插件的运行元数据信息,通过表单渲染界面将元数据渲染成表单,配置运行元数据的值,保存插件编排的描述并将其绑定到路由上。

[0061] 请参阅图5所示,作为本发明的优选实施方式,所述的插件编排模块具体进行以下处理:

[0062] (1)进行插件选择;

[0063] (2)判断是否进行自定义页面处理,如果是,则直接读取Vue页面;否则,直接读取插件元数据;

[0064] (3)通过表单渲染界面进行页面渲染处理;

[0065] (4)将需要启用的插件进行运行参数的配置处理。

[0066] 在实际应用当中,插件编排模块主要是在某个路由上进行插件的编排,通过图形化的方式从所有启用的插件中选择需要在路由上执行的插件,调整插件的执行顺序,读取插件的运行元数据信息(元数据主要分配置元数据、运行元数据),通过表单渲染界面将元数据渲染成表单,配置运行元数据的值,保存插件编排的描述并绑定到路由上。

[0067] 页面渲染分两类,一类是自定义的页面,渲染时直接根据插件名称到前端plugins目录下找到对应文件名的vue文件,在界面上以div的方式显示,另一类是通过插件元数据渲染,根据元数据定义的数据类型,字段名称,描述、输入提示、校验规则、是否必填等信息,来动态显示页面。

[0068] 作为本发明的优选实施方式,所述的页面渲染处理,具体包括:

[0069] 针对自定义的页面,渲染时直接根据插件名称到前端plugins目录下找到对应文件名的vue文件,在界面上以div的方式显示;

[0070] 针对插件元数据,渲染时根据元数据定义的数据类型、字段名称、描述、输入提示、校验规则、是否必填的相关信息,来动态显示页面。

[0071] 作为本发明的优选实施方式,所述的插件运行模块具体进行以下处理:

[0072] 当请求到达网关后,进行特定路由的匹配处理,并由Spring cloud gateway内核按照路由上插件编排的顺序调用对应的插件代码。

[0073] 在本发明的具体实施方式中,本技术方案是基于Java语言的Spring Cloud Gateway框架实现的,因此文中出现的英文单词都是Java术语以及代码,基于Spring Cloud Gateway网关上的插件编排和执行方法的具体实施步骤如下:

[0074] 1.用户进行插件开发和定义

[0075] 此模块的具体实施步骤如下:

[0076] 1)创建插件存储需要的数据库表,一个插件表用来存储插件基础信息,一个是插件元数据表存储插件元数据信息

[0077] CREATE TABLE`api_plugin`(

[0078] `ID`varchar(128)CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NOT NULL COMMENT'primary key id',

[0079] `NAME`varchar(62)CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NOT NULL COMMENT'插件名',

[0080] `DESCRIBES`varchar(255)CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NULL DEFAULT NULL COMMENT'插件描述(中文名)',

[0081] `TYPE_CODE`varchar(4)CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NULL DEFAULT NULL COMMENT'插件类型(1＝＝路由,)',

[0082] `ICON`varchar(128)CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NULL DEFAULT NULL COMMENT'插件图片',

[0083] `CONFIG`text CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NULL COMMENT'插件处理配置',

[0084] `ROLE`varchar(64)CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NULL DEFAULT NULL COMMENT'角色',

[0085] `SORT`int(11)NULL DEFAULT NULL COMMENT'排序',

[0086] `CUSTOM_STATUS`varchar(4)CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NULL DEFAULT NULL COMMENT'自定义页面字段(0:关闭,1:开启)',

[0087] `DESCRIPTIONS`varchar(255)CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NULL DEFAULT NULL COMMENT'详细描述',

[0088] `ENABLED`varchar(4)CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NULL DEFAULT'0'COMMENT'状态(0,not open,1open)',

[0089] `CREATE_BY`varchar(64)CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NULL DEFAULT NULL COMMENT'创建人',

[0090] `CREATE_DATE`datetime NULL DEFAULT CURRENT_TIMESTAMP COMMENT'创建时间',

[0091] `UPDATE_BY`varchar(64)CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NULL DEFAULT NULL COMMENT'更新时间',

[0092] `UPDATE_DATE`datetime NULL DEFAULT NULL ON UPDATE CURRENT_TIMESTAMP COMMENT'更新时间',

[0093] PRIMARY KEY(`ID`)USING BTREE

[0094])ENGINE＝InnoDB CHARACTER SET＝utf8mb4 COLLATE＝utf8mb4_unicode_ci COMMENT＝'插件表'ROW_FORMAT＝DYNAMIC;

[0095] CREATE TABLE`api_plugin_handle`(

[0096] `ID`varchar(128)CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NOT NULL COMMENT'primary key id',

[0097] `PLUGIN_ID`varchar(128)CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NOT NULL COMMENT'插件id',

[0098] `FIELD`varchar(100)CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NOT NULL COMMENT'字段',

[0099] `LABEL`varchar(100)CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NULL DEFAULT NULL COMMENT'描述',

[0100] `DATA_TYPE`varchar(6)CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NOT NULL DEFAULT'1'COMMENT'数据类型1number 2string',

[0101] `TYPE_CODE`varchar(10)CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NULL DEFAULT NULL COMMENT'字段所属类型,1means selector,2means rule,3means

plugin',

[0102] `SORT`bigint(11)NULL DEFAULT NULL COMMENT'排序',

[0103] `REQUIRED`varchar(4)CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NULL DEFAULT NULL COMMENT'是否必填(0非1必填)',

[0104] `DEFAULT_VALUE`varchar(255)CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NULL DEFAULT NULL COMMENT'默认值',

[0105] `PLACE_HOLDER`varchar(255)CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NULL DEFAULT NULL COMMENT'输入提示',

[0106] `RULE`varchar(255)CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NULL DEFAULT NULL COMMENT'校验规则',

[0107] `EXT_OBJ`varchar(4096)CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NULL DEFAULT NULL COMMENT'extra configuration(j son format data)',

[0108] `CREATE_BY`varchar(64)CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NULL DEFAULT NULL COMMENT'创建人',

[0109] `CREATE_DATE`datetime NULL DEFAULT CURRENT_TIMESTAMP COMMENT'创建时间',

[0110] `UPDATE_DATE`datetime NULL DEFAULT CURRENT_TIMESTAMP COMMENT'更新人',

[0111] `UPDATE_BY`varchar(64)CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci NULL DEFAULT NULL COMMENT'更新时间',

[0112] PRIMARY KEY(`ID`)USING BTREE,

[0113] UNIQUE INDEX`plugin_id_field_type`(`PLUGIN_ID`,`FIELD`,`TYPE_CODE`)USING BTREE

[0114])ENGINE＝InnoDB CHARACTER SET＝utf8mb4 COLLATE＝utf8mb4_unicode_ci COMMENT＝'插件元数据表'ROW_FORMAT＝DYNAMIC;

[0115] 2)插件开发通过idea等java开发工具,创建项目,增加spring cloud gateway的依赖

[0116]
```
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-gateway-dependencies</artifactId>
    <version>${spring-cloud-gateway.version}</version>
    <type>pom</type>
    <scope>import</scope>
</dependency>
```

[0117] 编写插件功能实现代码类,继承AbstractGatewayFilterFactory,在apply方法中实现插件代码逻辑

```
public class AuthGatewayFilterFactory extends
AbstractGatewayFilterFactory<AuthGatewayFilterFactory.Config> {

    public AuthGatewayFilterFactory() {
        super(Config.class);
    }
    @Override
    public GatewayFilter apply(Config config) {
        return (exchange, chain) -> {
            //编写具体插件的执行代码
            return chain.filter(exchange);
        };
    }
[0118]  public static class Config {

        // 验证地址
        private String webHookUrl;
        public Config() {
        }
        public String getWebHookUrl() {
            return webHookUrl;
        }
        public void setWebHookUrl(String webHookUrl) {
            this.webHookUrl = webHookUrl;
        }
    }
}
```

[0119] 修改config类的代码逻辑,主要是config对象解析成插件运行的元数据参数。

[0120] 3)通过开发工具idea或eclipse将代码打包成java虚拟机可翻译的字节码jar包,jar文件的名称需要跟插件名称一致,插件在进行热插拔的时候需要根据插件的名称加载卸载对应的jar。

[0121] 4)对于插件自定义表单的实现,需要用户通过前端开发工具进行表单的实现,将表单的数据封装成插件config对象的json数据,将表单页面保存成插件名.vue的文件格式,并存放到plugins目录下,插件在表单渲染时,会先判断此插件是否是自定义表单,如果是则会从plugins目录下根据插件名称找到vue文件直接加载显示,如果否则根据插件定义的元数据信息,动态渲染表单界面。

[0122] 5)实现插件的持久化功能,通过页面创建插件定义,将插件的资源包jar文件上传到服务器的plugins目录下。

[0123] 6)进行插件启用,将对应的插件资源包文件推送到网关上,存储到网关按照目录的plugins下,由插件运行模块进行调用。

[0124] 2.进行路由定义配置,实现路由动态更新

[0125] 此模块的具体实施步骤如下:

[0126] 1)如图6所示,进行定义路由描述文件的格式

[0127] Json格式

[0128]

属性	描述
Id	路由定义的唯一标识
Name	路由名称
Order	顺序优先级,在有多个路由匹配的情况下,优先选择order值小的
Gatewaycode	路由发布到网关的code,对应的网关加载此路由
Rulejson	插件编排规则描述

Singleroute	预留
State	路由是否启用,只有启用的路由网关才会加载

[0129] 关系表

[0130] 2)在前端进行路由的定义,并将输入的参数拼接成步骤一中定义的格式,存储到关系表中以及推送到redis中。在关系表中保存需要和redis保存保持一致,redis启动时会从关系表中读取并初始化,redis存储时的key定义需要按“gtw[API-GATEWAY-DEVELOPER].route.id[1402].route”这样的格式,其中API-GATEWAY-DEVELOPER是标识该路由是推送到哪个网关上的,1402标识路由的唯一标识id程序中实现订阅监听redis数据变化事件,继承MessageListener接口,重写onMessage方法,通过message.getBody()先判断redis数据是del还是set,如果是set则标识是新增路由或者路由变更,如果是del则标识删除路由,通过获取spring.application.name值与路由中的gatewaycode字段比较是否是一致,如果不一致表示此路由不在当前网关上,如果是则表示当前网关需要处理。

[0131] 3)根据redis的数据构建路由RouteDefinition对象,实现RouteDefinitionRepository接口,重写save、delete、getRouteDefinitions等方法实现路由的动态更新。

[0132] 3.在路由定义上进行插件的编排

[0133] 1)如图7所示,在本实施例中,用户可以在路由定义界面上进行插件编排,选择待编排插件并设置各个待编排插件的执行顺序、运行参数。具体的,用户根据业务场景需要,从插件编排界面的插件列表中选择多个插件,选中的插件列表按照默认的顺序在界面上显示插件图标和名称,用户可以通过拖拽的方式调整执行各个插件间的顺序。用户再根据所需实现的功能的业务逻辑,按拖拽的顺序逐个配置每个插件的运行参数,运行参数是根据插件定义的元数据动态显示的,会根据元数据描述,决定是字符型、数值型还是日期型显示控件类型,提交时会根据元数据判断该参数是否必填,必填项的提示语显示,按照校验规则进行输入校验。最后客户端通过识别用户的拖拽操作、点击操作、输入操作,从而获得到用户的插件编排数据,并将此数据绑定到路由上,由动态路由定义将数据提交到服务器端。

[0134] 4.插件运行环境创建以及插件编排的执行

[0135] 1)在本实施例中,在网关上创建一个事件监听器,监听用户启用停用插件的指令,具体指令可以通过界面单击按钮发出,监听器监听到指令后,如果是启用指令,运行环境则从plugins插件目录下找到对应的jar包文件,并通过classload方式将jar文件中的class注入到Spring IOC容器中,如果是停用指令,则将IOC容器中插件相关的bean对象销毁。

[0136] 2)网关运行过程中,请求到达网关,由Spring Cloud Gateway内核匹配到具体的路由,再根据动态定义的路由信息,获取到路由上插件的编排数据,确定需要执行哪些插件,执行的顺序,再按照插件的执行顺序,逐个调用Spring IOC容器中插件对应的Bean对象,将路由定义中的插件运行参数传递到Bean对象的config中,由Spring Cloud Gateway内核执行插件中的Apply方法,完成插件的业务逻辑功能。

[0137] 流程图中或在此以其他方式描述的任何过程或方法描述可以被理解为,表示包括一个或更多个用于实现特定逻辑功能或过程的步骤的可执行指令的代码的模块、片段或部分,并且本发明的优选实施方式的范围包括另外的实现,其中可以不按所示出或讨论的顺序,包括根据所涉及的功能按基本同时的方式或按相反的顺序,来执行功能,这应被本发明

的实施例所属技术领域的技术人员所理解。

[0138] 应当理解，本发明的各部分可以用硬件、软件、固件或它们的组合来实现。在上述实施方式中，多个步骤或方法可以用存储在存储器中且由合适的指令执行装置执行的软件或固件来实现。

[0139] 本技术领域的普通技术人员可以理解实现上述实施例方法携带的全部或部分步骤是可以通过程序来指令相关的硬件完成的，程序可以存储于一种计算机可读存储介质中，该程序在执行时，包括方法实施例的步骤之一或其组合。

[0140] 上述提到的存储介质可以是只读存储器，磁盘或光盘等。

[0141] 在本说明书的描述中，参考术语“一实施例”、“一些实施例”、“示例”、“具体示例”、或“实施例”等的描述意指结合该实施例或示例描述的具体特征、结构、材料或者特点包含于本发明的至少一个实施例或示例中。在本说明书中，对上述术语的示意性表述不一定指的是相同的实施例或示例。而且，描述的具体特征、结构、材料或者特点可以在任何的一个或多个实施例或示例中以合适的方式结合。

[0142] 尽管上面已经示出和描述了本发明的实施例，可以理解的是，上述实施例是示例性的，不能理解为对本发明的限制，本领域的普通技术人员在本发明的范围内可以对上述实施例进行变化、修改、替换和变型。

[0143] 采用了本发明的该信创环境下基于Spring Cloud Gateway网关路由上插件编排和执行的系统，通过将过滤器的概念抽象成插件，对插件进行定义管理、元数据配置，使运维人员可以将基于过滤器扩展开发的功能作为插件进行管理，对插件进行热插拔、启用停用等操作，同时通过图形化的方式对路由进行配置，可以配置断言以及相关插件选择，可以通过拖拽的方式对所选插件进行编排，在执行时按照编排的顺序执行，支持了路由的动态配置生效，满足特定业务场景下的需求，增强了网关的适应性。解决了用户基于Spring cloud gateway网关的图形化路由配置方式，使路由配置及时生效，实现了让用户动态管理过滤器的能力，进行过滤器热插拔，同时由于采用了编排方式，任意组合各种插件，满足了用户很多实际业务场景下的复杂需求，提升了网关的适应性、易用性。

[0144] 在此说明书中，本发明已参照其特定的实施例作了描述。但是，很显然仍可以作出各种修改和变换而不背离本发明的精神和范围。因此，说明书和附图应被认为是说明性的而非限制性的。

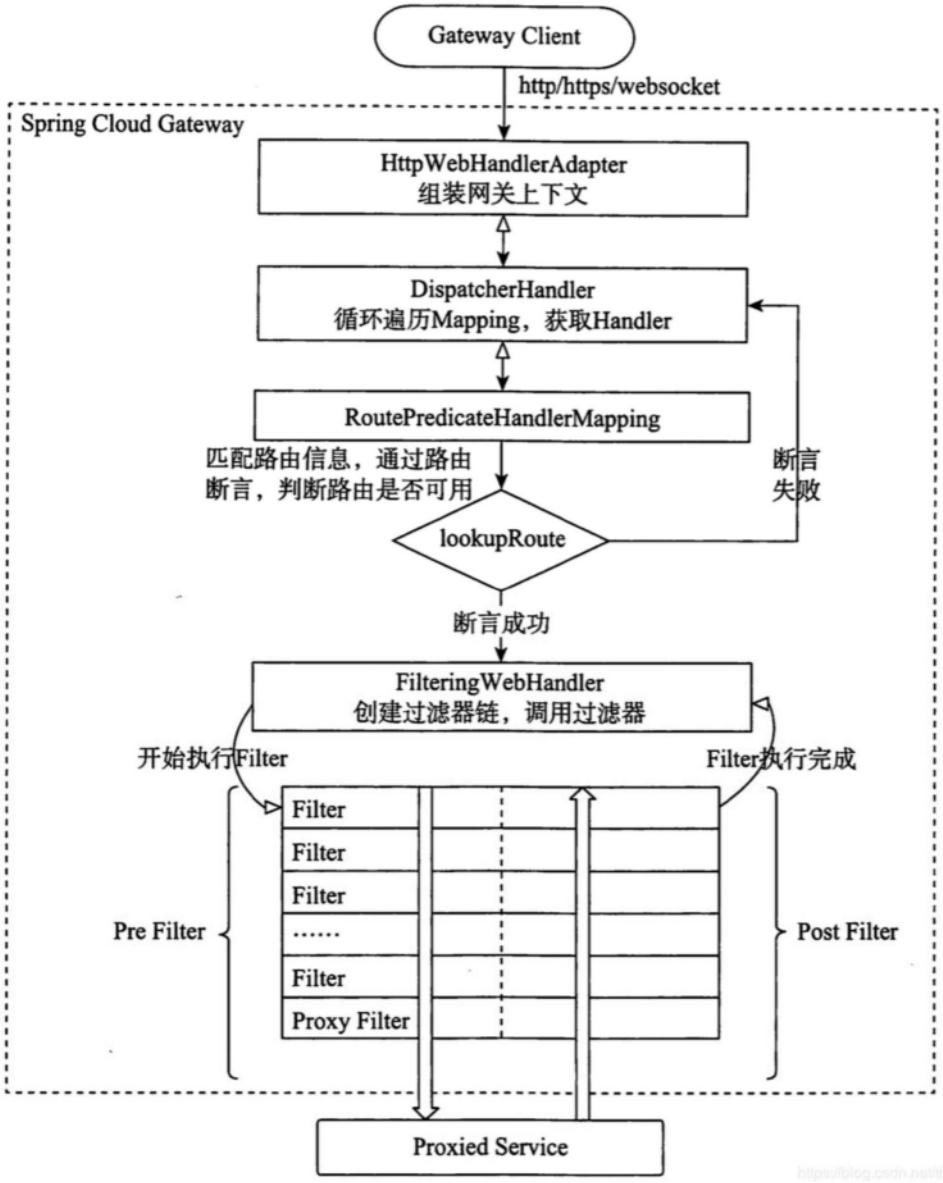

Gateway Client
http/https/websocket
Spring Cloud Gateway
HttpWebHandlerAdapter
组装网关上下文
DispatcherHandler
循环遍历Mapping，获取Handler
RoutePredicateHandlerMapping
匹配路由信息，通过路由
断言，判断路由是否可用
断言
失败
lookupRoute
断言成功
FilteringWebHandler
创建过滤器链，调用过滤器
开始执行Filter
Filter执行完成
Filter
Filter
Filter
.....
Filter
Proxy Filter
Pre Filter
Post Filter
Proxied Service

图1

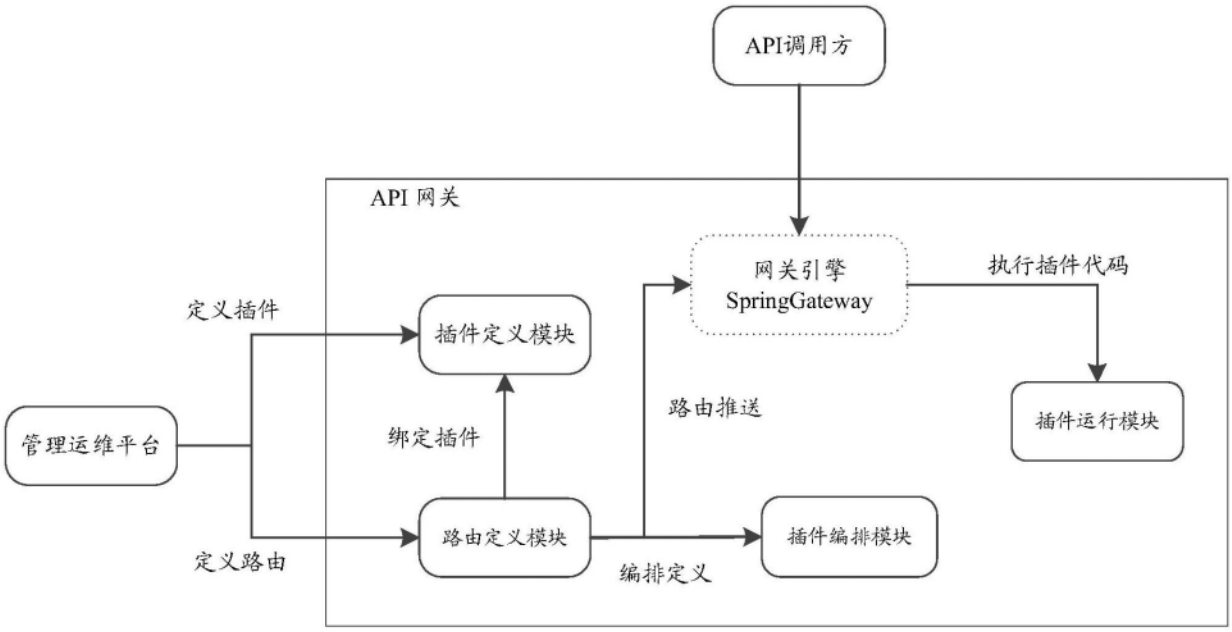

API调用方
API 网关
网关引擎
SpringGateway
执行插件代码
定义插件
插件定义模块
管理运维平台
绑定插件
路由推送
插件运行模块
路由定义模块
插件编排模块
定义路由
编排定义

图2

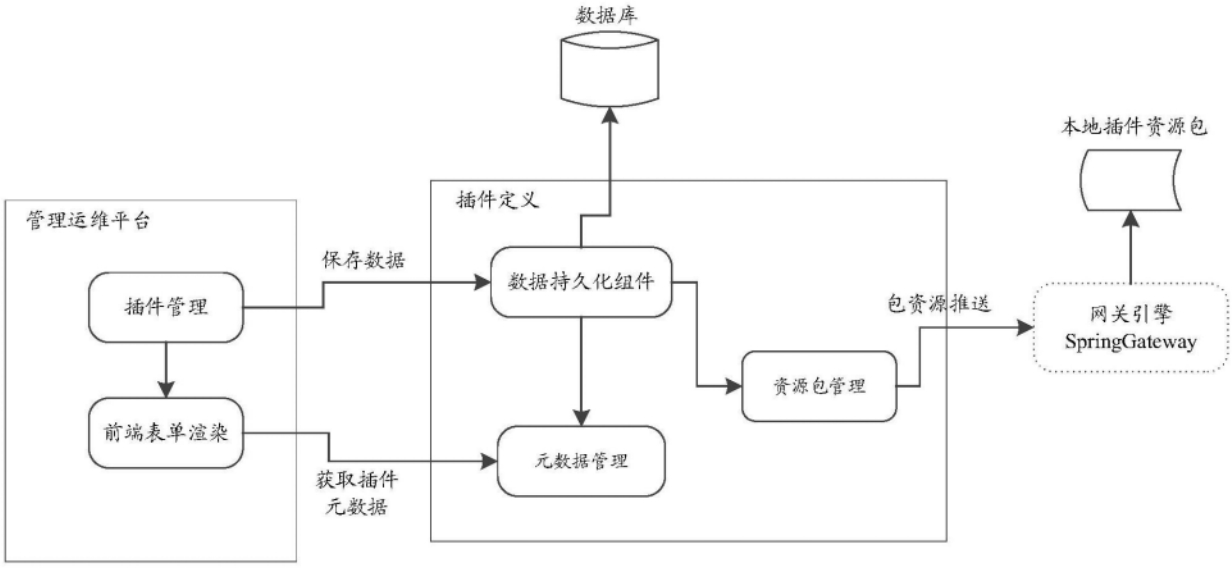

数据库
本地插件资源包
插件定义
管理运维平台
保存数据
插件管理
数据持久化组件
包资源推送
网关引擎
SpringGateway
资源包管理
前端表单渲染
元数据管理
获取插件
元数据

图3

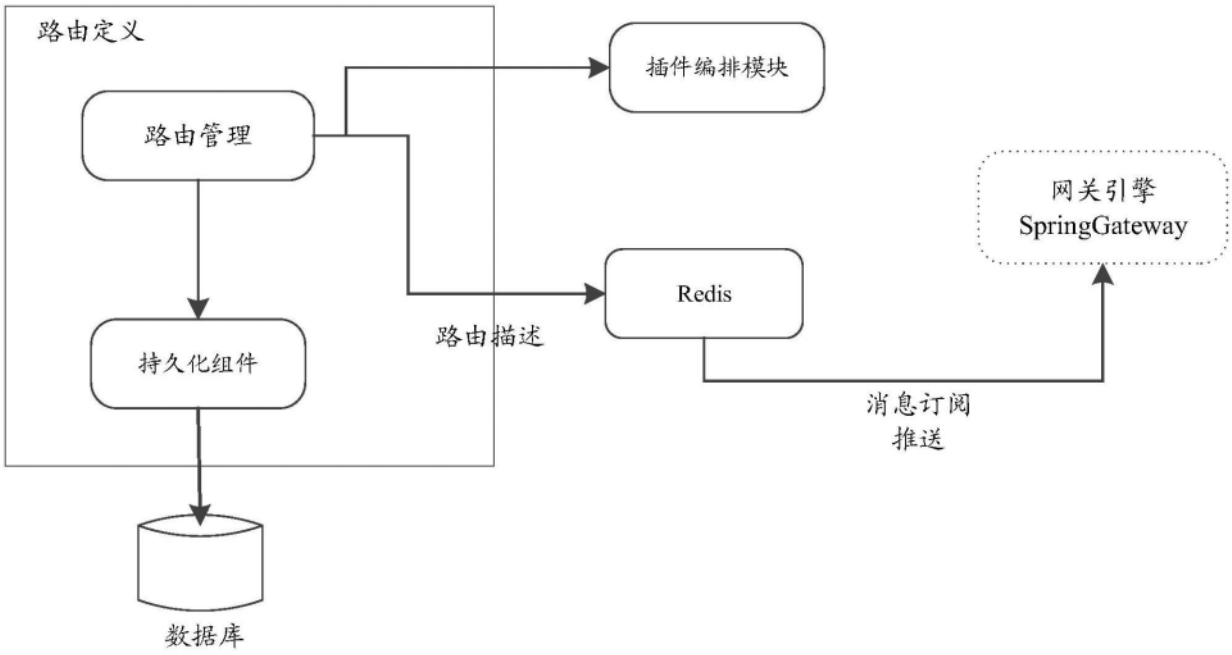

路由定义
插件编排模块
路由管理
网关引擎
SpringGateway
Redis
持久化组件
路由描述
消息订阅
推送
数据库

图4

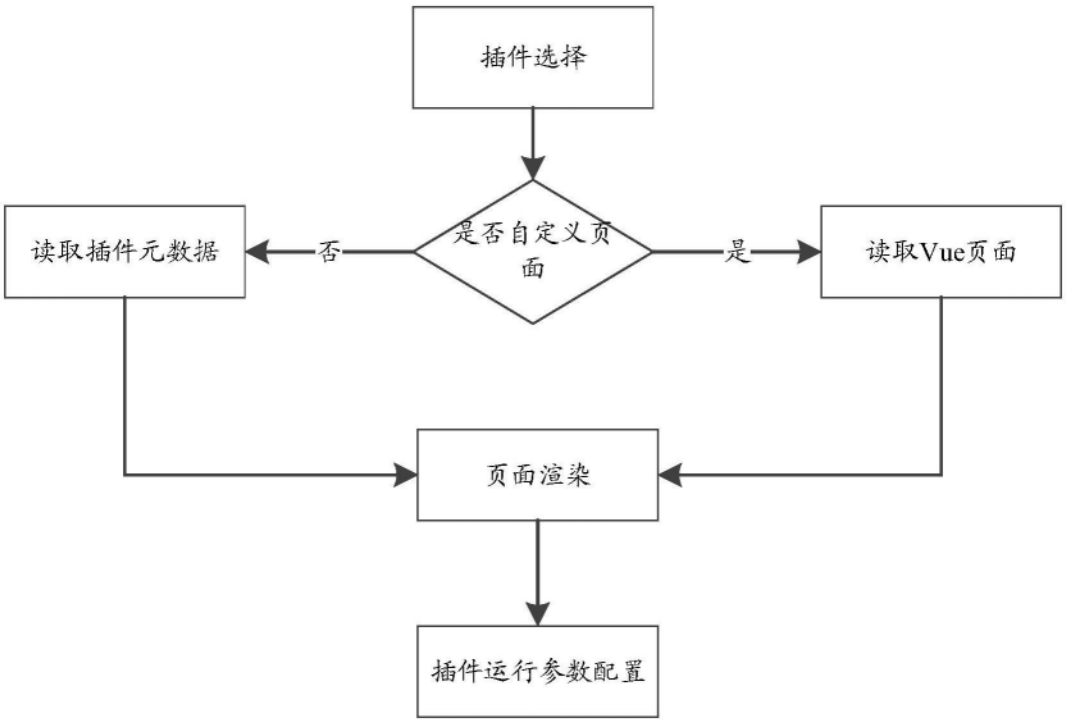

插件选择
是否自定义页面
否
是
读取插件元数据
读取Vue页面
页面渲染
插件运行参数配置

图5

```
{
  "id": "1409",
  "name": "老版demo带鉴权",
  "order": 1,
  "gatewayCode": "API-GATEWAY-DEVELOPER",
  "ruleJson": "{\"predicates\":[{\"name\":\"Path\",\"args\":{\"_genkey_0\":\"/API-GATEWAY-TEST/gatewayTest/post\"}}],\"
filters\":[{\"name\":\"StripPrefix\",\"args\":{\"_genkey_0\":\"1\"}},{\"name\":\"Auth\",\"args\":{\"webHookUrl\":\"http
://192.168.1.216:28092/api/gateway/business/apiGatewaySubscriber/verifyToken\"}}],\"uri\":\"http://192.168.1.216:28092\
"}",
  "singleRoute": false,
  "state": "ENABLED"
}
```


图6

图7