E:\项目\国电投\cloudlogo.png

iuap 服务治理平台V3.5

发版说明

用友网络科技股份有限公司

2018年2月

# 目录

[目录 2](#_Toc31647)

[1 概述 3](#_Toc5424)

[2 本版修订 4](#_Toc32493)

[3 产品特性 5](#_Toc10776)

[4 产品范围 6](#_Toc3169)

[4.1 RPC调用框架 6](#_Toc9051)

[4.2 基础中间件 8](#_Toc10586)

[4.3 开发适配组件 9](#_Toc26951)

[4.4 稳定性组件 10](#_Toc23475)

[4.5 服务治理平台门户 11](#_Toc13827)

[5 产品主要功能 14](#_Toc31178)

[5.1 RPC调用框架 14](#_Toc16386)

[5.1.1 服务注册中心 14](#_Toc23461)

[5.1.2 服务健康检查 14](#_Toc5383)

[5.1.3 RPC框架 15](#_Toc32367)

[5.2 基础中间件 16](#_Toc4672)

[5.2.1 组件加载器 16](#_Toc29854)

[5.2.2 插件机制 16](#_Toc2629)

[5.2.3 配置中心 16](#_Toc4220)

[5.3 开发适配组件 17](#_Toc28669)

[5.3.1 Spring 适配 17](#_Toc25595)

[5.3.2 Dubbo 适配 17](#_Toc21939)

[5.3.3 iuap 适配 17](#_Toc18406)

[5.4 稳定性组件 17](#_Toc18922)

[5.4.1 服务动态限流控制 17](#_Toc28152)

[5.4.2 服务链路追踪 18](#_Toc10156)

[5.4.3 服务权限动态控制 18](#_Toc23655)

[5.5 服务治理平台门户 19](#_Toc24163)

[5.5.1 服务管理控制台 19](#_Toc26324)

[5.5.2 元数据管理 19](#_Toc5511)

[5.5.3 服务授权 19](#_Toc6089)

[5.5.4 服务搜索 20](#_Toc29345)

[5.5.5 服务调用统计 20](#_Toc22320)

# 概述

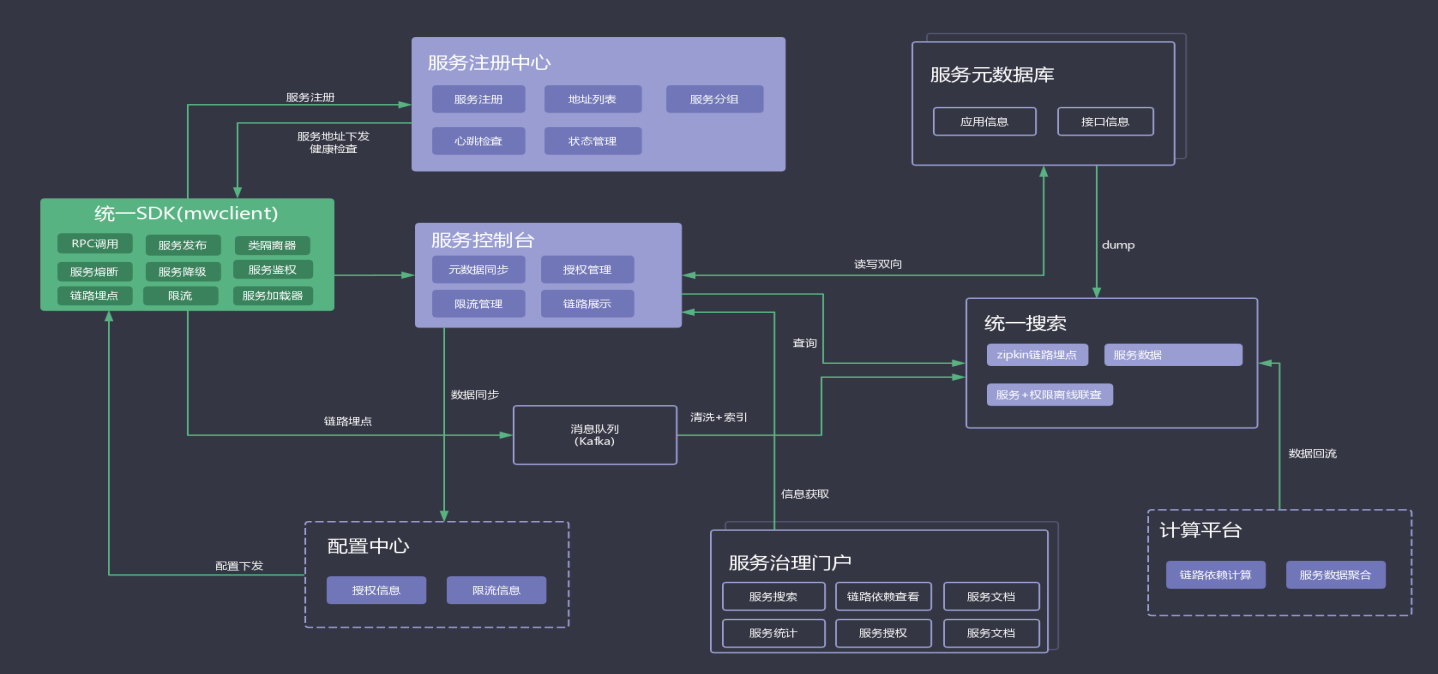
iuap 服务治理平台是云平台提供的针对微服务应用的统一管控和治理平台，以下简称服务治理平台。

服务治理平台旨在提供适合用友云产品和其他业务线的统一微服务治理工具集，包括RPC开发框架、基础的组件和中间件、稳定性组件以及服务搜索和管控门户，已此来帮助云平台其他云产品及用友的业务和领域产品更合理的进行微服务的拆分、开发、管控，统一规范，简化开发。

服务治理平台包括：

1. RPC框架：基础的微服务间RPC调用框架，支持多种序列化协议，提供token校验和心跳检查机制的服务注册中心，注册中心支持高可用架构集群部署；支持生成客户端代理Bean和动态代理；
2. 基础中间件：包括基础的组件加载器，上下文初始化机制；提供包隔离机制的完整组件，包含可扩展的插件机制，支持通过插件方式在调用方和提供方不同时机接入业务处理；包含配置中心的支持，动态拉取关键配置文件；
3. 稳定性组件：提供线程数和QPS方式的服务限流控制；提供服务链路追踪功能，可以查看服务调用层级和调用链路，调用的状态和成功率；提供应用级别和方法级别的权限控制；
4. 开发适配组件：适配组件包含对spring工程的适配，兼容大多数的spring版本；包含对dubbo工程的适配，解析dubbo原生的配置文件并创建代理；包含对iuap工程的上下文传递功能的适配；
5. 服务治理平台门户：治理平台门户包含统一的管控面板、对认证秘钥的管理、应用的管理、服务调用的权限控制、链路查看、服务搜索、元数据管理等；

用友云服务治理平台技术架构如下：



# 本版修订

|  |  |
| --- | --- |
| **服务治理平台** | |
| **关键改进点1** | 支持服务注册和服务发现、支持服务注册中心集群部署，保证高可用性 |
| **关键改进点2** | 支持服务注册后心跳检查，服务检查token验证 |
| **关键改进点3** | 注册中心支持调用者服务和被调用者服务的注册，注册信息包括租户ID,APPID。注册中心可以查看实例的所属应用和状态，当前应用的服务列表。 |
| **关键改进点4** | 提供基本RPC调用框架，支持组件加载机制和插件机制,支持指定IP调用 |
| **关键改进点5** | 支持服务注册的zone元数据设置，调用时可按照会话信息调用指定zone的服务 |
| **关键改进点6** | 支持网络多跃点智能检测，运行时按照最短路径进行调用 |
| **关键改进点7** | 支持服务限流，可以从线程数和QPS两个维度进行限流 |
| **关键改进点8** | 给服务提供者提供直观的服务依赖关系图和微服务看板，同时支持无痕埋点，埋点对应用无侵入，无性能影响 |
| **关键改进点9** | 通过分布式链路追踪功能，查看调用耗时，提供快速定位性能问题的能力 |
| **关键改进点10** | 支持对注册到微服务注册中心的服务进行检索、支持服务和方法级别查询 |
| **关键改进点11** | 支持调用统计，统计区间范围内的细化到方法级别的用户访问量 |
| **关键改进点12** | 监控客户端和服务端调用数据，统计调用成功率 |
| **关键改进点13** | 配置中心，支持应用配置文件的统一管理，当配置发生变化时实时更新，保证线上线下的配置文件的一致性 |
| **关键改进点14** | 支持服务注册和运行时的应用的多套环境的区分，环境隔离访问和统计监控 |
| **关键改进点15** | 支持不同环境之间的权限控制，token机制的调用验证 |
| **关键改进点16** | 支持服务和方法级别的授权控制，服务调用时权限验证 |
| **关键改进点17** | 支持类隔离机制的加载，提供隔离组件包，不影响业务工程的三方组件的版本依赖 |
| **关键改进点18** | 支持对开源开发组件的适配，包含spring不同版本、dubbo服务 |
| **关键改进点19** | 提供秘钥管理功能，不同角色分配不同的秘钥，保证调用过程环境的隔离验证 |
| **关键改进点20** | 提供元数据管控后台，根据元数据统计和控制接口和方法信息，提供文档支持 |

# 产品特性

|  |  |  |
| --- | --- | --- |
| 产品 | 模块 | 子模块 |
| 服务治理平台 | RPC调用框架 | 服务注册中心 |
| 服务健康检查 |
| RPC框架（iris相关组件） |
| 基础中间件 | 组件加载器（helix、middleware） |
| 类隔离机制 |
| 插件机制 |
| 配置中心（proteus） |
| 稳定性组件 | 服务动态限流控制（sentinel） |
| 服务链路追踪（yyeye） |
| 服务权限动态控制 |
| 开发适配组件 | dubbo适配器 |
| iuap适配器 |
| spring适配器 |
| 服务治理平台门户 | 控制台后端服务 |
| 元数据管理 |
| 服务调用授权 |
| 服务搜索 |
|  |  | 服务调用统计 |

# 产品范围

## RPC调用框架

|  |  |  |
| --- | --- | --- |
| 模块 | 功能 | 描述 |
| 注册中心 | 服务实例注册 | 服务调用方和服务提供方启动时，将服务的IP、端口、等信息注册到注册中心，注册中心统一维护 |
| 核心元数据注册 | 注册中心保持服务调用方和服务提供方的核心元数据信息，信息包括租户ID,APPID、可用状态、环境、zone等，调用过程中可以根据元数据信息进行路由 |
| 支持服务动态发现和缓存 | 发起远程调用时, 会注册中心拉取被调用服务的信息(如服务有多少实例,每个实例的IP/域名和端口是什么)并且缓存到本地, 方便使用负载均衡的方式去调用远程服务; 被缓存的远程服务信息会被定时从注册中心拉取刷新, 以便发现新的服务实例和删除有异常的服务实例 |
| 支持注册中心高可用 | 服务注册中心使用集群方式搭建, 防止单点故障, 在客户端和注册中心交互时会绕过集群中有异常的节点访问正常的节点. 集群异常节点恢复后会从集群中的其他节点拉取客户端注册的信息并且加入集群中对外正常提供服务 |
| 服务健康检查 | 服务注册检查 | 微服务客户端启动时扫描应用的元数据（接口名、方法信息）和地址信息(如:主机和端口等信息)上报且注册到注册中心. 注册过程会上报元数据和校验token，符合条件的环境和有权限的AK信息对应的服务会注册成功 |
| 心跳检查和连通性检查 | 服务注册中心通过接收此微服务应用的各个实例的心跳信息来检查各个实例的健康状态.一定时间内没有心跳的服务实例会及时剔除，新增的服务实例会更新到注册中心 |
| RPC框架 | 服务调用方 | 服务调用方进行代理Bean的声明、Stub创建，客户端可以构建bean依赖关系，待运行时远程调用 |
| 服务提供方 | 服务提供方提供接口的具体实现，提供统一的Servlet接入和权限拦截过滤器 |
| 序列化 | 服务调用过程中框架提供信息的序列化和反序列化，支持多种序列化方式、jackson、hessian |
| 网络多跃点检测 | 在复杂网络下的服务可能会有多个IP和端口, 在客户端注册服务到注册中心时, 会上报所有网络设备点的IP/域名和端口; |
| 路径优化 | 支持多层网络环境下，最短路径选择，在远程调用时, 会智能检测跃点数最少的可联通的地址. 以保证远程调用时跃点的数量、路径的速度, 提高服务的可靠性和吞吐量. |
| 开发过程注解支持 | 支持使用注解方式在微服务提供方声明服务和方法 |
| 容灾支持 | 服务注册中心中断服务时，客户端仍然可以使用本地实例列表副本，进行远程调用 |
| 客户端负载均衡 | 指服务调用方从服务注册中心获取一组可用的服务提供方实例，服务调用过程中，从维护的实例组中按照策略选择一实例进行使用。微服务的客户端负载均衡默认使用轮询策略进行服务实例的使用，支持扩展不同的规则。 |
| 错误实例标记 | 支持周期性错误数统计，及时摘除不可用的服务提供者实例 |

## 基础中间件

|  |  |  |
| --- | --- | --- |
| 模块 | 功能 | 描述 |
| 基础中间件 | 组件加载器 | 服务治理平台基础中间件中包含基础的组件加载器（helix），其包含基础公共依赖、app启动器、插件机制接口定义，上下文初始化，统一的日志处理器等，通过组件加载器，加载必要的组件和信息，为后续的RPC调用做准备 |
| 类隔离机制 | 治理平台提供基于类隔离机制的完整组件包，组件包内部组件自依赖，不依赖和影响外部业务工程的三方组件依赖，减少引入微服务治理平台组件后的组件版本冲突问题，内部通过SPI机制加载所需的组件 |
| 插件机制 | 服务治理平台基础组件支持插件机制，不同的内部组件可以实现公共依赖定义的插件的接口，在不同的调用时机插入各个功能实现，提供扩展机制可以增加其他插件 |
| 配置中心SDK | 与配置中心服务器端配套的功能jar包，应用引入该jar包并做相应配置就能通过配置中心管理引用的配置文件。实现近实时更新，统一分发，多种环境配置文件管理等功能。 |
| 简单的引入方式，maven方式组件依赖，提供注解式编程和无侵入配置多种引入方式 |
| 本地文件与线上文件的定时对比，保证配置文件一致性 |
| 强兼容性，通过配置启用和停用配置中心的功能，快速切换使用本地配置文件或者线上配置文件 |
| 监控微服务所需的权限和限流的配置文件的变化，配置变化后近实时更改服务调用方、服务提供方对服务实例的控制 |
| 配置中心后台 | 通过一个统一的配置中心实现管理多种应用、同一应用的多个实例、同一应用的多种环境（online/stage/test /dev）等多种环境的配置 |
| 实现对分布式应用配置文件的统一管理 |

## 开发适配组件

|  |  |  |
| --- | --- | --- |
| 模块 | 功能 | 描述 |
| 适配组件 | Spring工程适配 | 服务治理平台开发框架支持spring工程的微服务调用，隔离的机制使得组件适配不同的spring的版本，不限制业务工程对spring和iuap组件的版本，通过增加各个组件的spring-support，springboot-support组件可以进行支持，如iris-spring-support、iris-springboot-support、sentinel-spring-support等 |
| Dubbo工程的适配 | RPC调用框架可以原生支持dubbo工程，启动过程解析dubbo服务的服务端和调用端的配置文件，根据group信息进行应用的区分，同时支持dubbo调用时隐式参数传递的机制，可以在上下文中增加调用参数外的其他信息，组件通过额外引入iris-dubbo-support达到效果 |
| iuap适配 | 支持对原iuap工程的线程绑定变量的信息传递，提供iris-iuap-support组件 |

## 稳定性组件

|  |  |  |
| --- | --- | --- |
| 模块 | 功能 | 描述 |
| 稳定性组件 | 服务依赖分析 | 直观展现服务提供的各个接口的调用依赖 |
| 提供者提供直观的服务依赖关系,可以查看调用层级和调用消耗时间 |
| 无痕埋点，对应用代码无侵入 |
| 对应用性能影响几乎忽略不计 |
| 分布式链路追踪 | 支持直观展现服务调用流程 |
| 支持快捷定位异常发生的节点和方法 |
| 支持快捷展现各个节点时间消耗，找出性能瓶颈 |
| 链路搜索.支持用户根据时间段和服务调用消耗时间来查询调用链路信息 |
| 调用权限控制 | 服务接口权限分为公有接口、私有接口。公共状态下所有注册开发者中心的消费者都可以调用。私有状态下需要通过授权操作，对消费者应用进行授权，只有进过授权的消费者才可以调用。 |
| 配置中心统一管理权限配置，准实时推送到客户端，以实时调整调用权限 |
| 私有接口需授权访问，授权对象为调用方的应用，授权后的服务可以调用成功 |
| 服务限流 | 限流框架（potus）为微服务应用提供基于QPS和线程并发的限流方式 |
| 支持通过时间片切分的控制算法，能够对应用的访问流量进行削峰，保障应用服务能够快速拒绝自身容量之外的请求，防止系统雪崩 |
| 限流服务与RPC框架IRIS进行了深度的整合，依赖于中间件SDK后，能够自动识别RPC调用信息，并能够自动跟踪和统计RPC调用，进行限流拦截和控制，无需用户进行额外配置和编码 |
| QPS限流支持避免出现流量突刺 |
| 线程数的限流支持用于简单系统的负载保护，及自我保护优雅降级，防止依赖的下游服务RT变长的情况下，造成自身系统性能下降 |
| 支持微服务API限流 |
| 支持自定义限流，管理员可在门户控制台对指定的方法进行特殊限流配置 |

## 服务治理平台门户

|  |  |  |
| --- | --- | --- |
| 模块 | 功能 | 描述 |
| 服务  治理平台门户  控制台 | 应用管理 | 管理微服务类的应用信息，微服务的调用方和服务提供方都是对应治理平台的一个应用，通过可视化界面可以统一管理应用 |
| 应用的多套环境的支持，每个应用对应开发、测试、灰度、生产四个环境，运行时可以针对不同的环境进行调用，可以针对不同的环境的应用进行不同的授权 |
| 秘钥管理 | 门户集成AccessKey的管理功能，开发者和运维管理员可以申请对应身份的AccessKey和秘钥，服务启动时不同的秘钥可以对应不同的环境，不同环境间调用可以进行权限控制 |
| AccessKey和秘钥也是对请求进行签名的依据，注册中心的token验证，服务之间调用的验证都是通过秘钥进行加签依据 |
| 统一微服务控制台 | 服务描述信息是在微服务展示面板上，展示服务基本信息：是否私有、今日调用次数、调用成功率。并且提供权限、链路、限流、依赖、统计等功能的操作入口。 |
| 元数据管理 | 统一存储和管理微服务启动时通过扫描工具上报的元数据信息，包括应用级别和方法级别的元数据信息，为控制台提供访问和查询元数据信息的接口，同时将上报的元数据信息推送到ES集群进行存储，方便后续查询和聚合。 |
| 应用对应的服务接口列表的信息来源； |
| 服务接口的接收参数、返回参数等信息的来源。 |
| 服务搜索 | 对注册到用友云微服务注册中心的服务进行检索 |
| 对接口下的方法根据条件进行检索 |
| 选择某个服务的具体详情展示，包含服务的提供商，服务的请求参数，响应参数等 |
| 支持对服务进行关键字和名称的模糊搜索 |
| 调用统计 | 统计时间范围和耗时范围内的调用 |
| 统计显示调用成功率 |
| 统计应用下的接口数量和接口下的方法数量 |
| 服务调用授权 | 服务的方法公有私有权限的修改 |
| 服务的具体方法的授权和管理 |
| 服务文档 | 查看方法说明 |
| 查看方法的请求参数和返回参数说明 |

# 产品主要功能

## RPC调用框架

### 服务注册中心

服务治理平台的注册中心支持应用服务的自动扫描和发现，并且提供容灾措施，最大限度的保证服务的高可用性。

提供服务地址列表的更新对比和本地容灾功能，在服务注册中心失败的情况下，依然可以根据本地的容灾信息，进行服务实例地址的获取。

微服务SDK在应用启动时，会自动根据服务注解信息，扫描所有定义的服务接口信息，将服务定义的元数据(接口名、方法名称、方法参数和返回值、方法权限可见性)，以及服务实例信息(实例IP、应用和租户信息、中继网关信息)等上报到服务注册中心和元数据管理后台。

在启动过程中，如果服务注册失败，则会阻止应用启动，防止由于服务注册不成功，造成应用发布成功之后，服务没有注册造成的服务能力缩容，进而造成业务大面积失败，将风险提前阻断。

服务调用方在启动时，也会同样将自己的应用信息上报到注册中心，并且根据需要调用的服务信息，从注册中心获取需要调用的服务的实例地址信息。在连接到注册中心失败，调用方也会启动失败，避免由于获取不到服务调用信息，而导致启动后的业务失败，将风险提前阻断。

服务注册时会校验来源请求的合法性，服务注册请求会携带根据秘钥进行签名后的token，注册中心会进行验证。

服务注册中心可以提供不同zone的元数据标记，服务提供者在启动时可以配置指定的zone信息，利用zone信息，调用方在进行路由时可以进行选择。

### 服务健康检查

服务治理平台支持对服务提供方和调用方的心跳检查，以检测服务的可用性信息，及时进行失败的预警和检测。

服务实例启动注册之后，每隔固定的时间间隔，实例会向注册中心发送心跳信息。注册中心依据服务实例发送的心跳信息，确定服务实例健康度。在服务实例失败情况下，注册中心在超时未收到心跳信息之后，会将服务实例标记为不可用状态，然后更新服务地址列表信息，在调用方更新地址列表时，将会把不可用的服务信息踢出。

服务调用方也会定时向注册中心发送心跳信息，以便定时获取服务最新的地址列表信息。

服务治理平台各个组件间都加入了稳定性保障和容灾机制，保障在每个组件失败的情况下，都能有容灾方案，不会由于某个组件的失败，导致整体服务不可用。

平台各个服务端组件都以集群的方式搭建(注册中心、配置中心、元数据管理后端、链路跟踪平台，认证中心，服务搜索服务等)，保障服务本身的稳定性，不会由于服务本身某一实例的不可用，造成整体服务不可用。而在各个组件的依赖之间(包括之前写到的服务注册和RPC框架的客户的容灾机制)的本地容灾机制，保障了整体服务的稳定性。通过多级的稳定性容灾措施，提升整体平台服务的稳定性。

### RPC框架

服务治理平台提供微服务的RPC调用框架，支持服务之间的RPC调用，支持多种二进制协议，并且通过提供与spring的集成，提供便捷快速的RPC调用支持。

在使用基础的标准框架spring进行开发时，只需要在声明了微服务的RPC注解之后，就可以像本地Java bean一样，对RPC接口进行方法调用。

RPC框架的客户端包含服务发现模块，可以发现服务注册中心注册的提供者，在客户端调用服务端时，可以对服务提供者列表根据一定的规则进行负载均衡，也可以对不可用的服务提供者进行错误标记，及时排除到服务提供者列表之外。

RPC框架提供对多种序列化协议的支持，包括JSON、hessian等，为了传输消息和兼容性，默认使用hessian协议。框架提供了很好的机制，以支持新序列化的接入，用户可以进行自定义协议的扩展开发，并且能够快速的接入到RPC框架中。

RPC框架服务端提供统一的servlet，进行请求的接收和分发，服务调用请求达到servlet之前会进行权限的过滤和有效性验证，通过验证的请求允许后续动作。

框架提供注解方式进行编程，包含服务声明注解、权限注解、方法排除注解、文档注解等，方便开发者快速接入RPC框架。

## 基础中间件

### 组件加载器

服务治理平台基础中间件中包含基础的组件加载器（helix），其包含基础公共依赖、app启动器、插

件机制接口定义，上下文初始化，统一的日志处理器等，通过组件加载器，加载必要的组件和信息，为后续的RPC调用做准备。同时提供基于类隔离机制的完整组件包，组件包内部组件自依赖，不依赖和影响外部业务工程的三方组件依赖，减少引入微服务治理平台组件后的组件版本冲突问题，内部通过SPI机制加载所需的组件。

### 插件机制

服务治理平台基础组件支持插件机制，不同的内部组件可以实现公共依赖定义的插件的接口，在不同的调用时机插入各个功能实现，提供扩展机制可以增加其他插件。服务治理平台提供的默认的插入点包括

IBeforeInvoke、IAfterInvoke、IBeforeExecute、IAfterExecute等，分别对应客户端调用之前和之后，服务端被调用之前和之后。

框架内部的组件如权限、限流、链路等实现各个插件，处理不同的业务逻辑，各个插件通过SPI的机制统一进行加载和调用。插件管理和调用机制集成在框架内部，对业务开发者透明。

### 配置中心

通过统一的配置中心实现管理多种应用、同一应用的多个实例、同一应用的多种环境等多种配置模式，实现了对分布式应用配置文件的统一管理和对应用不同环境（online/test/stage/dev）配置文件的管理及配置变化后线上应用状态的实时更新。引入方式简单，单jar包依赖，提供注解式编程和无侵入引入方式与用友云配合提供对普通应用配置文件的管理能力。配置中心记录指定的配置文件与各个客户端实例的配置文件同步状态，可以查看保持同步的实例的地址信息并支持在线维护配置文件内容。

配置中心支持纯文本形式和K-V形式的动态配置下发，可以支持spring框架初始化配置的动态注入，并且支持对于查看配置连接的各个实例的健康状态。

服务治理平台中的权限和限流模块的动态生效，依赖配置中心SDK对配置文件的推送达到效果，门户的控制台中的权限和限流的操作反应到配置中心配置文件的变化，配置中心的SDK监控配置文件的变化，实时的触发权限和限流功能的生效。

配置中心的配置文件可以在门户控制台可视化的管理，针对应用分为不同的环境和不同的版本，可以针对不同的环境和版本进行不同的设置。

## 开发适配组件

### Spring 适配

平台开发框架支持spring工程的微服务调用，隔离的机制使得组件适配不同的spring的版本，不限制业务工程对spring和iuap组件的版本，通过增加各个组件的spring-support组件可以进行支持，如iris-spring-support、iris-springboot-support、sentinel-spring-support等。

### Dubbo 适配

RPC调用框架可以原生支持duboo工程，启动过程解析dubbo服务的服务端和调用端的配置文件，根据group信息进行应用的区分，同时支持dubbo调用时隐式参数传递的机制，可以在上下文中增加调用参数外的其他信息，组件通过额外引入iris-dubbo-support达到效果。

### iuap 适配

适配组件中包含对原iuap工程的适配组件，iuap框架中包含对线程绑定变量的信息传递，适配组件提供iris-iuap-support组件，完成iuap框架下信息到RPC调用上下文的转化，通过RemoteInvocation传递到服务提供者端。

## 稳定性组件

### 服务动态限流控制

服务限流主要包括如下功能：

（1）Potus为微服务应用提供基于QPS和线程并发的限流方式。

（2）支持通过时间片切分的控制算法，能够对应用的访问流量进行削峰，保障应用服务能够快速拒绝自身容量之外的请求，防止系统雪崩。

（3）限流服务与RPC框架IRIS进行了深度的整合，依赖于中间件SDK后，能够自动识别RPC调用信息，并能够自动跟踪和统计RPC调用，进行限流拦截和控制，无需用户进行额外配置和编码。

（4）QPS限流支持避免出现流量突刺。

（5）线程数的限流支持用于简单系统的负载保护，及自我的熔断保护优雅降级，防止依赖的下游服务RT变长的情况下，造成自身系统性能下降。

（6）支持自定义限流。限流框架通过微服务SDK提供自定义限流埋点功能，用户可以使用限流API，在代码流程的任意地方嵌入限流埋点，进行自定义的限流配置。可以方便的将限流功能集成到用户的系统代码中。

### 服务链路追踪

在微服务展示面板上，展示服务基本信息：是否私有、今日调用次数、调用成功率。并且提供权限、链路、限流、依赖、统计等功能的入口。

服务治理平台提供对于微服务RPC之间的调用链路追踪，为用户提供直观的链路追踪数据，便于服务的问题排查和数据统计。

服务追踪数据提供对于服务调用的服务名称、调用延迟、调用结果等信息的追踪，并且通过对于依赖数据的保存，可以层级的展示服务的调用关系信息。

可以通过应用名称、服务信息、时间范围等进行调用链路的搜索，方便用户快速定位到需要的调用信息。

微服务SDK还提供服务追踪数据埋点API，用户可以根据扩展自定义的链路追踪，支持对用户自定义协议和自定义链路追踪，和用户自定义数据的埋点。使用API进行埋点之后，可以将用户自定义的业务数据展示到链路追踪页面展示中。

### 服务权限动态控制

服务治理平台提供服务接口方法级别的调用权限控制。服务接口权限分为公有接口、私有接口。公共状态下所有注册开发者中心的消费者都可以调用。私有状态下需要通过授权操作，对消费者应用进行授权，只有进过授权的消费者应用才可以调用相关的私有服务。

通过设置服务的权限，可以通过配置中心实时推送到客户端，使其权限功能生效，保护核心服务的安全和可用性。

## 服务治理平台门户

服务治理平台门户，提供统一的可视化管控界面，包含登录控制，应用管理，配置管理、服务发现、权限管理、用户管理等功能。

### 服务管理控制台

管理控制台提供门户的后端请求，包含对应用的管理、应用的核查、元数据的查询、微服务的接口查询、链路查询，配置文件修改入口、邮件发送等。

应用的多套环境的支持，每个应用对应开发、测试、灰度、生产四个环境，运行时可以针对不同的环境进行调用，可以针对不同的环境的应用进行不同的授权。

通过服务管理控制台统一对接门户前端的请求，利用其串联元数据和ES信息，查询数据库中的应用信息等。

门户前端集成AccessKey的管理功能，开发者和运维管理员可以申请对应身份的AccessKey和秘钥，服务启动时不同的秘钥可以对应不同的环境，不同环境间调用可以进行权限控制。

### 元数据管理

元数据管理后端服务统一存储和维护所有微服务类应用的元数据信息，微服务应用启动时会根据注解来扫描服务的接口、方法、参数等信息，上报到后端服务，后续控制台根据此元数据信息进行信息的展示和统计。

元数据后端服务将应用级别和方法级别的元数据信息上报到ES集群，后续由ES集群提供快速的查询和聚合功能。

### 服务授权

微服务类应用可以根据不同环境为不同角色的人员分配管理和使用权限，方便开发者和运维人员分别控制开发态和生产态的应用。

服务治理平台支持对应用下指定接口的方法进行权限控制，方法分为公有和私有权限，公有权限的方法允许各个合法的客户端调用，设为私有的方法，允许授权的应用的客户端调用。

方法对应的授权对象为应用，即允许指定的微服务应用调用此私有权限的方法。

服务和方法的权限信息保存在后端的权限服务对应的存储中。

### 服务搜索

治理平台支持对注册到用友云微服务注册中心的服务进行检索。列表方式展示检索到的服务且可查看某个服务的具体详情，包含服务的提供商，服务的请求参数，响应参数等。服务搜索支持进行关键字或名称的模糊搜索。

门户首页还提供服务发现功能，可以根据关键字对发布的服务进行模糊搜索，查看API文档，查看方法的入参和出参、服务的提供者、所属应用等信息。

### 服务调用统计

服务治理平台提供对于服务接口信息、方法信息的展示和统计，提供对方法级别的调用次数和正确率的统计。同时可以按照时间范围和消耗时间段进行统计分析。

支持用户根据时间段和服务调用消耗时间来查询调用链路信息。