SOFAStack

微服务

产品简介

产品版本：AntStack Plus 1.13.1

文档版本：20230707

蚂蚁集团
ANT GROUP

法律声明

蚂蚁集团版权所有©2022，并保留一切权利。

未经蚂蚁集团事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。

商标声明

蚂蚁集团 ANT GROUP及其他蚂蚁集团相关的商标均为蚂蚁集团所有。本文档涉及的第三方的注册商标，依法由权利人所有。

免责声明

由于产品版本升级、调整或其他原因，本文档内容有可能变更。蚂蚁集团保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在蚂蚁集团授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过蚂蚁集团授权渠道下载、获取最新版的用户文档。如因文档使用不当造成的直接或间接损失，本公司不承担任何责任。

通用约定

格式	说明	样例
危险	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	危险 重置操作将丢失用户配置数据。
警告	该类警示信息可能会导致系统重大变更甚至故障，或者导致人身伤害等结果。	警告 重启操作将导致业务中断，恢复业务时间约十分钟。
注意	用于警示信息、补充说明等，是用户必须了解的内容。	注意 权重设置为0，该服务器不会再接受新请求。
说明	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	说明 您也可以通过按Ctrl+A选中全部文件。
>	多级菜单递进。	单击**设置**> **网络**> **设置网络类型**。
粗体	表示按键、菜单、页面名称等UI元素。	在**结果确认**页面，单击**确定**。
Courier字体	命令或代码。	执行 `cd /d C:/window` 命令，进入Windows系统文件夹。
斜体	表示参数、变量。	`bae log list --instanceid` *Instance_ID*
[] 或者 [a|b]	表示可选项，至多选择一个。	`ipconfig [-all|-t]`
{} 或者 {a|b}	表示必选项，至多选择一个。	`switch {active|stand}`

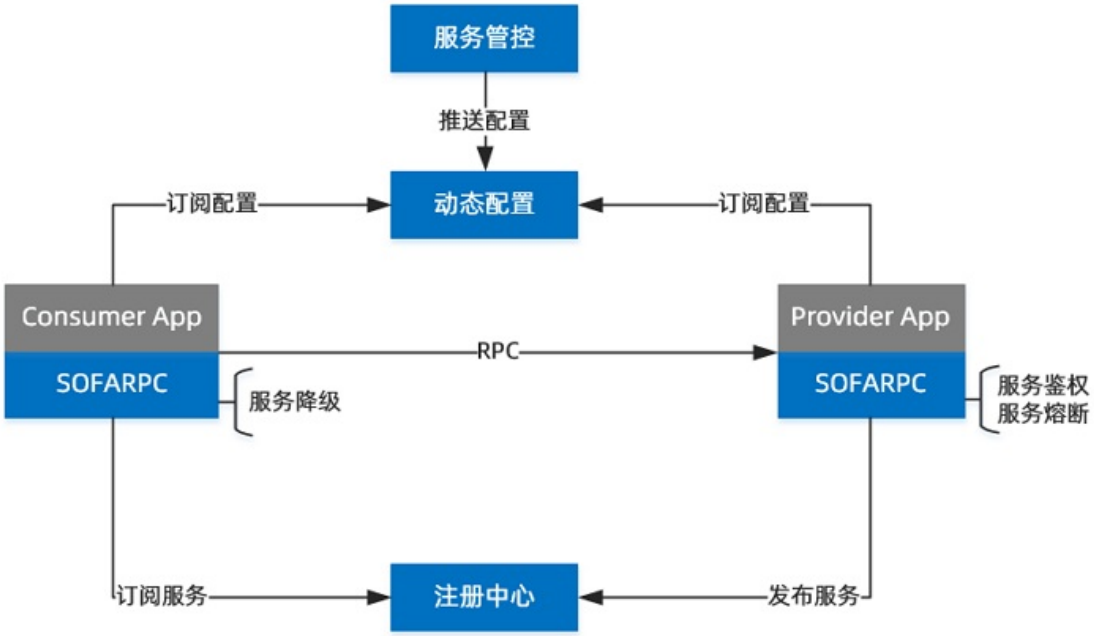
目录

1.概述

微服务（SOFAStack Microservices，简称 SOFAStack MS）主要提供分布式应用常用解决方案。使用微服务框架可以进行服务的自动注册与发现，以及服务管理相关操作。

使用 SOFABoot 开发应用，通过 SOFAStack 控制台部署到云端后，微服务会自动注册到服务注册中心。您可以通过微服务控制台进行服务管控和治理的相关操作。

微服务主要通过 SOFARPC 实现服务的发布和引用，其它模块都围绕 SOFARPC 展开。产品架构如下：



服务注册

服务注册通过注册中心（SOFARegistry）实现。注册中心是蚂蚁中间件的底层组件，用于存储所有服务提供方的地址信息，以及所有服务消费方的订阅信息。它和服务消费方、服务提供方都建立长连接，动态感知服务发布地址变更，并通知消费方。

RPC 服务

提供对 SOFARPC 的支持。SOFARPC 是一个分布式服务框架，为应用提供高性能、透明化、点对点的远程服务调用方案，具有高可伸缩性、高容错性。

动态配置

动态配置（Distributed Resource Management，简称 DRM）可以实现在应用运行时，动态修改配置的功能。提供动态配置的简便接入方式与集中化管理平台，可在管理平台维护动态配置元数据，并可对配置值进行推送，还可以实时查看接入动态配置的客户端应用节点的内存值。

服务治理

提供对业务系统的限流、熔断、降级服务，从而保证业务系统不会被大量突发请求击垮，提高系统稳定性。

应用依赖

应用通过 RPC 发布、订阅服务时，应用依赖可以提供实时分析结果，可展示不同应用之间的服务调用关系，以及应用发布和订阅的服务信息。

2.产品优势

微服务产品在蚂蚁集团内部已支撑数万个节点规模的分布式应用架构，具有高可用性、高可扩展性、高性能、高时效性、稳定可靠等核心优势，并提供丰富的功能来帮助用户简化分布式系统的管理，让业务开发人员可以专注于业务逻辑实现，提升研发效率。

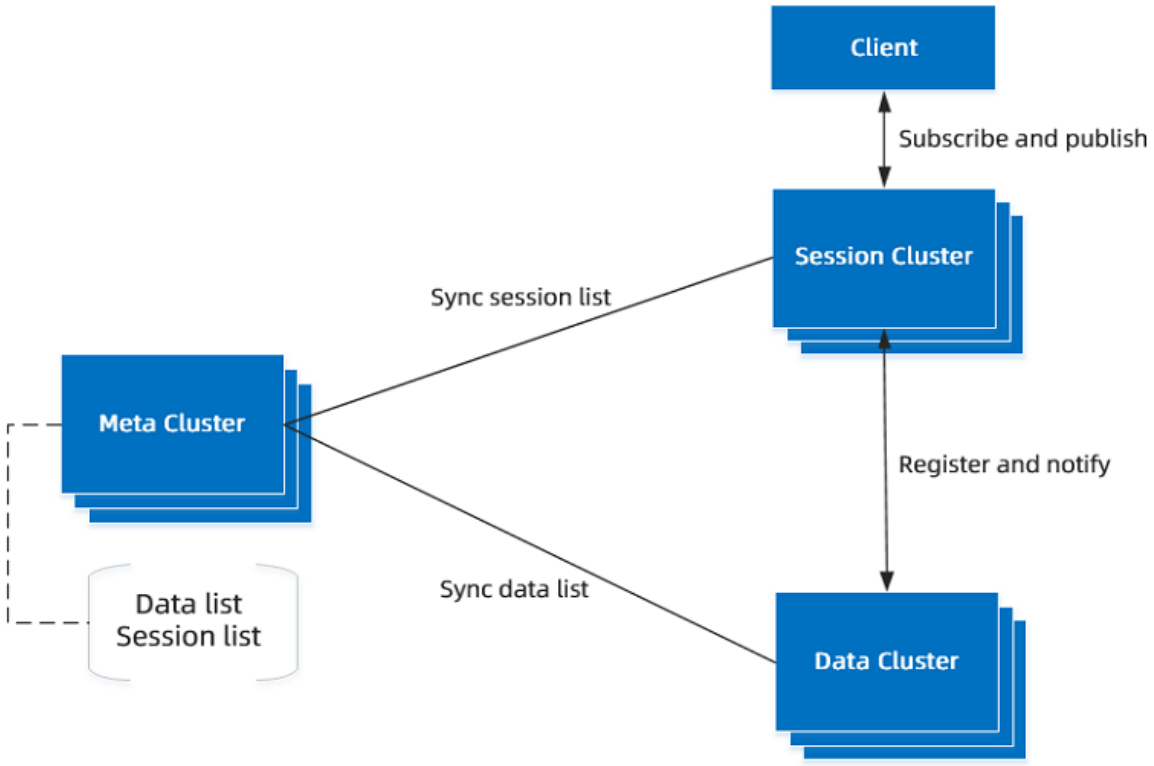
微服务产品提供金融级分布式架构的基础设施能力，包括 RPC 框架及服务治理、服务注册与发现、动态配置、定时任务、服务限流等，为传统单体应用架构深入拆分为分布式应用架构提供稳定可靠的基础设施能力，帮助企业级客户快速构建基于微服务架构的分布式应用，从而实现更灵活地响应业务变化，提高系统的可扩展性及性能。

3.产品架构

微服务包含服务注册、服务治理、动态配置及 RPC 服务，本文详细介绍提供各服务的组件架构信息。

SOFARegistry

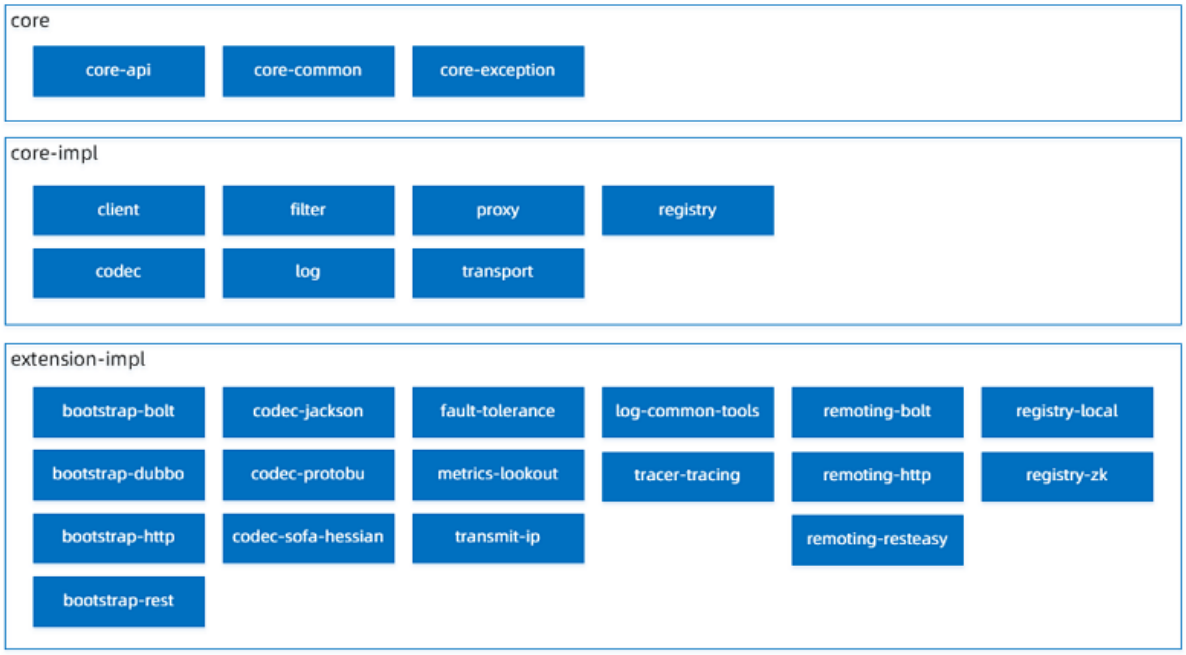
SOFARegistry 即服务注册中心，提供服务注册服务。产品架构如下：



- **Client（客户端）**：提供应用接入服务注册中心的基本 API 能力，应用系统依赖客户端 JAR 包，通过编程方式调用服务注册中心的服务订阅和服务发布能力。
- **SessionServer（会话服务器）**：提供客户端接入能力，接受客户端的服务发布及服务订阅请求，并作为一个中间层将发布数据转发至 DataServer 存储。SessionServer 可无限扩展以支持海量客户端连接。
- **DataServer（数据服务器）**：负责存储客户端发布数据，数据存储按照数据 ID 进行一致性 hash 分片存储数据存储按照数据 ID 使用自定义 slot 分配算法分片存储，支持多副本备份，保证数据高可用。DataServer 可无限扩展以支持海量数据。
- **MetaServer（元数据服务器）**：负责维护集群 SessionServer 和 DataServer 的一致列表，在节点变更时及时通知集群内其他节点。MetaServer 通过 SOFAJRaft 保证高可用和一致性。

SOFARPC

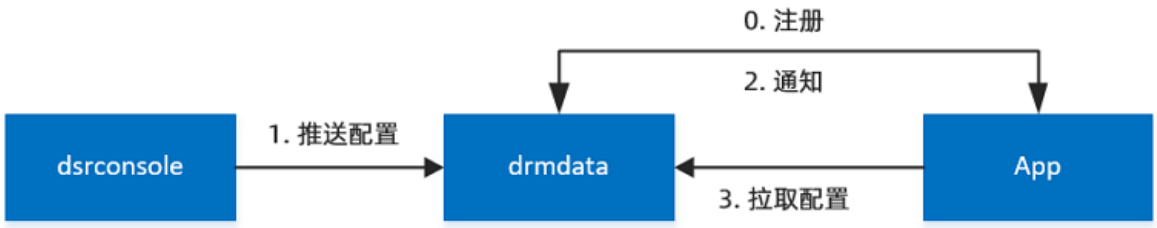
SOFARPC 是一个分布式服务框架，为应用提供高性能、透明化、点对点的远程服务调用方案。产品架构如下：



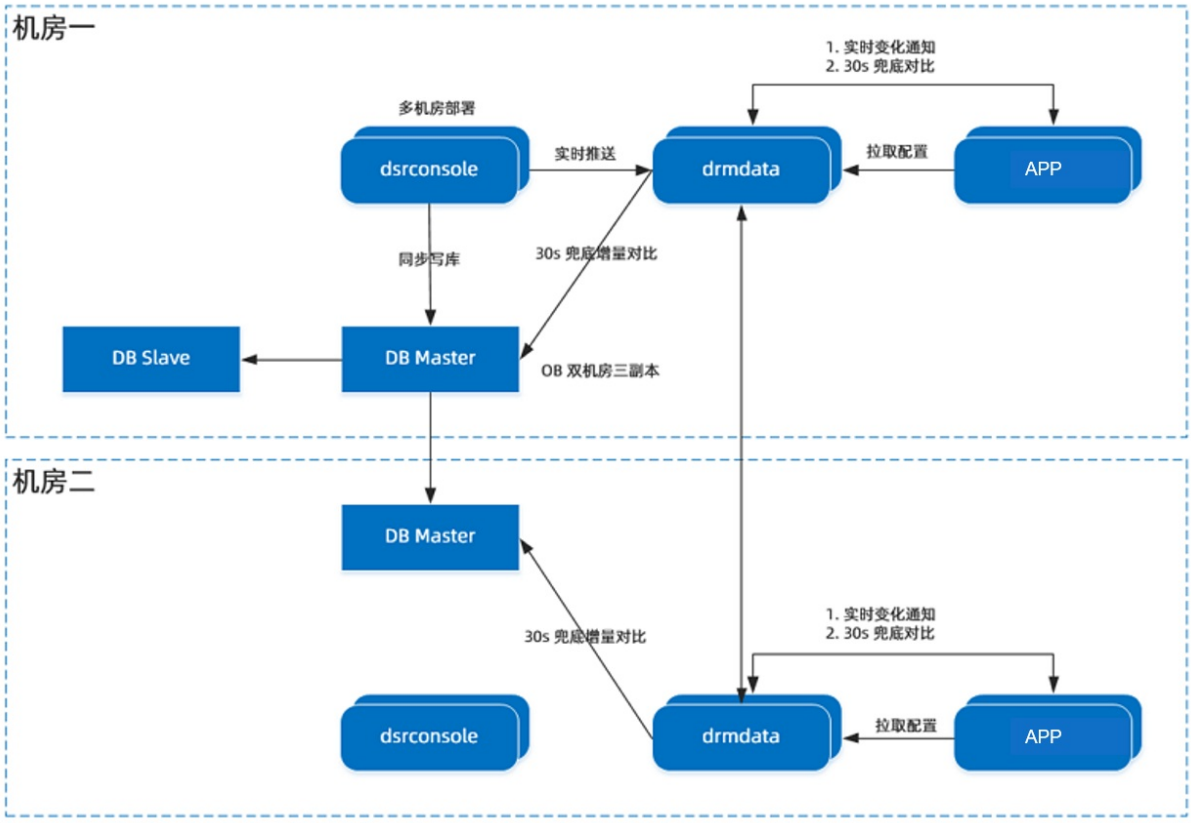
- **核心功能**：包括 core 和 core-impl 两部分，主要内容为 API 和一些扩展机制。
- **扩展及实现**：即 extension-impl 部分，主要包含了不同的实现和扩展，比如对 HTTP、REST、 metrics 以及其他注册中心的集成和扩展，例如 bootstrap 中对协议的支持，remoting 中对网络传输的支持，registry 中对注册中心的支持等。

DRM

DRM 的具体配置值并不通过服务注册中心推送，它由 dsrconsole 和 drmdata 两个组件组成。dsrconsole 负责实时推送时配置的持久化化，并同步给 drmdata。drmdata 则维护着与各个应用之间的注册链接，并在收到 dsrconsole 的推送通知时缓存配置并通知各个订阅应用来拉取配置。应用接收到该指令后，知道配置项发生了变更，向 drmdata 发起 HTTP 请求，读取真正的配置值，过程如下图所示：



DRM 的整套模型中，配置项的变更推送仅起到通知作用，推送的是版本号，而真实的配置值是依靠客户端接收到推送后主动发起的 HTTP 拉取。为了提高读取性能，drmdata 采用缓存提升读取效率，请求透传到 Java Server，经过 Java Server 的内存缓存过滤一次，未命中请求才可能请求到 DB。推送流程如下：

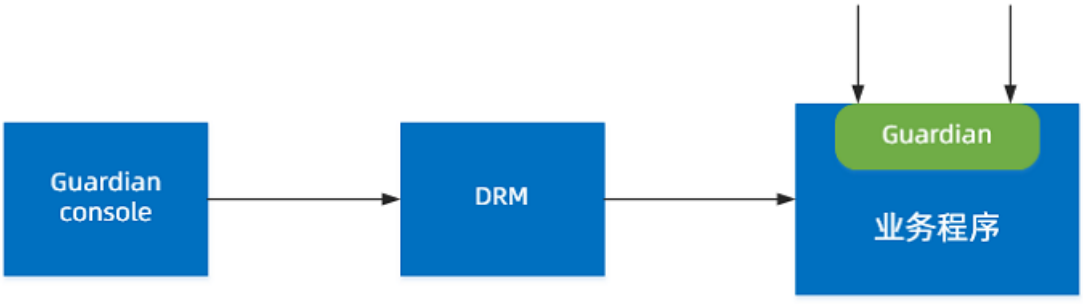


DRM 完整的推送流程是：

1. 在页面上，单击 **推送** 按钮。
2. dsrconsole 写最新的推送数据到 DB，DB 返回此 key 的最新版本号。
3. dsrconsole 通知推送消息到同机房的一台 drmdata 机器，消息内容包括：key、value、version。
4. drmdata 收到推送消息后，广播通知消息到所有的机房的 drmdata 机器。
5. 每台 drmdata 收到推送消息后，根据 key 查询所有 Client 建立的长连接信息，通过长连接发送给客户端 key 最新的版本号。
6. Client 收到推送消息后，拿着 drmdata 告知的最新版本号，随机调用同机房 drmdata 提供拉取配置的 HTTP 接口，拉取最新的配置。

Guardian

Guardian 主要用于服务限流，产品架构如下：

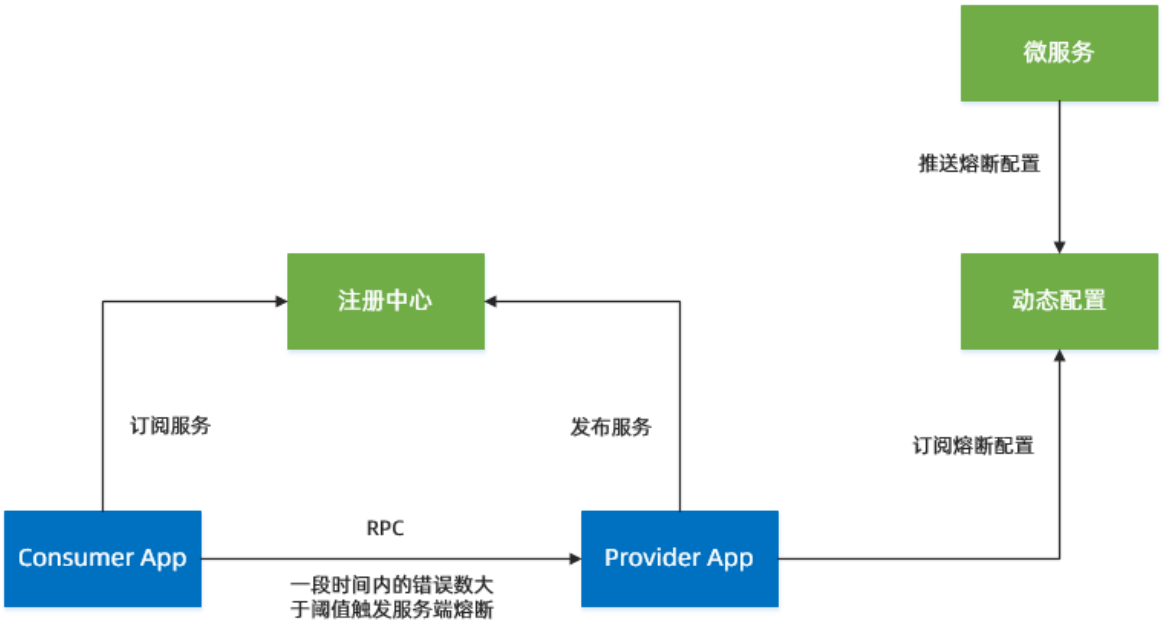


Guardian 运行主要分成为以下 4 个部分：

- **组件集成**：业务程序首先要集成 Guardian。
- **规则推送**：用户在 Guardian Console（控制台） 上编辑限流规则，并且把编辑好的规则推送到客端户，限流规则才会生效。
- **流量匹配**：当有流量进入到业务程序，首先会被 Guardian 组件拦截到，Guardian 会判断当前流量是否和限流规则匹配。
- **限流生效**：如果流量和限流规则匹配上，并且达到了预设的限流值，则限流。

服务熔断

服务熔断主要目的是当某个服务故障或者异常时，如果该服务触发熔断，可以防止其他调用方一直等待所导致的超时或者故障，从而防止雪崩。产品架构如下：



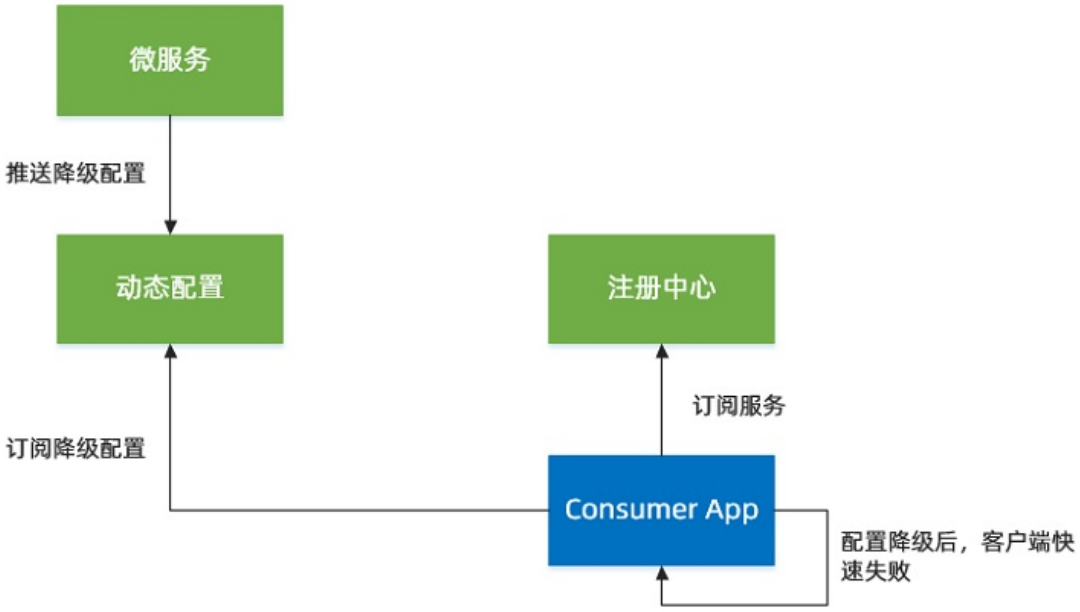
- **Provider App**：指服务提供端发布服务，并向注册中心注册。
- **Consumer App**：指服务使用端通过从注册中心拉取信息，来使用服务。
- **注册中心**：即 SOFARegistry，是 SOFA 中间件的底层组件，用于存储所有服务提供方的地址信息以及所有服务消费方的订阅信息；它和服务消费方、服务提供方都建立长连接，动态感知服务发布地址变更并通知消费方。

服务熔断主要分为以下 4 个部分：

- **订阅配置**：Provider App 首先要订阅熔断配置。
- **配置生效**：用户在微服务控制台编辑、推送熔断配置值，并通过动态配置（DRM）进行动态调整。配置推送后，Provider App 所订阅的配置即开始生效。
- **熔断触发**：Consumer App 通过 RPC 调用服务后，在规定时间内，如果错误数大于阈值，会触发服务端熔断配置。
- **熔断自动恢复**：当触发熔断后，RPC 会在一个时间窗口内快速失败。过了这个时间窗口后，系统会允许通过一个请求，去尝试探测服务是否恢复。如果恢复，则自动关闭熔断；如果没有恢复，则继续熔断，并等待下一个时间窗口。

服务降级

服务降级的主要目的是当服务器压力剧增的情况下，根据实际业务情况及流量，对某些不重要的服务，不处理或换种简单的方式处理，从而释放服务器资源以保证核心业务正常运作或高效运作。产品架构如下：



- **Consumer App**：指服务使用端通过从注册中心拉取信息，来使用服务。
- **注册中心**：即 SOFARegistry，是 SOFA 中间件的底层组件，用于存储所有服务提供方的地址信息以及所有服务消费方的订阅信息；它和服务消费方、服务提供方都建立长连接，动态感知服务发布地址变更并通知消费方。

服务降级主要分为以下3 个部分：

- **订阅配置**：Consumer App 首先要订阅降级配置。
- **配置生效**：用户在微服务控制台编辑、推送降级配置值，并通过动态配置进行动态调整。配置推送后，Consumer App 所订阅的配置即开始生效。
- **降级触发**：当 Consumer App 触发服务降级后，Consumer App 将无法使用服务，而且表现为快速失败。此时，需要业务方主动处理降级所带来的影响。

4.功能特性

微服务有高性能分布式服务框架、微服务治理中心、高可靠的轻量级配置中心、多活数据中心等特性，本文主要介绍这些特性。

高性能分布式服务框架

提供高性能和透明化的 RPC 远程服务调用，具有高可伸缩性、高容错性的特点。

- 支持多协议、多序列化、多语言，包括 Bolt（默认协议）、Dubbo、RESTful、WebService、Protobuf、Hessian、JSON 等。
- 服务自动注册与发现支持服务自动注册与发现，无需配置地址即可实现分布式环境下的负载均衡，并支持多种路由策略及健康检查。
- 依赖管理视图提供对 RPC 发布订阅的实时结果，可展示不同应用之间的服务调用关系，以及应用发布和订阅的服务信息。

微服务治理中心

提供一系列的服务治理策略，保障服务高质量运行，最终达到对外承诺的服务质量等级协议。

- 服务高可用支持客户端限流、集群容错（失败重试）、服务熔断（故障剔除）、故障注入、服务降级等保障服务高可用。
- 服务安全支持 CRC 校验，调用加解密，黑白名单等保障服务的安全。
- 服务的监控支持 Metrics 2.0 规范的日志埋点，支持成功率、调用次数、耗时、异常次数等多维度监控信息。

高可靠的轻量级配置中心

提供应用运行时动态修改配置的服务，并提供图形化的集中化管理界面。

- 配置动态推送实时生效支持按全量 IP 地址及指定 IP 地址进行配置推送，无需重启应用，并支持推送回滚。
- 客户端信息管理可查看客户端列表信息，包括客户端的当前内存值及服务端的推送值。
- 推送记录管理支持在控制台查看动态配置的推送记录，并支持以文件的方式对配置进行批量导入及导出。

多活数据中心

支持同城双活、异地多活架构，具备异地容灾能力，保障系统的可用性。

- 支持多种维度系统扩展支持应用级、数据库级、机房级、地域级的快速扩展。
- 按机房进行服务发现和路由支持跨 IDC 的服务发现，并支持按机房进行路由。
- 按数据中心进行配置修改支持按数据中心进行配置的动态推送，不同的机房的配置可根据业务需求设置为不同的值。

5.应用场景

SOFAStack 微服务具有高性能、高可靠、高可用的特点，适用于以下应用场景。

传统应用微服务改造

通过微服务产品将传统金融业务系统拆分为模块化、标准化、松耦合、可插拔、可扩展的微服务架构，可缩短产品面世周期，快速上架，抢占市场先机，不仅可确保客户服务的效率，也降低了运营成本。

- 开发简单：提供高性能微服务框架，轻松构建原生云应用，具备快速开发，持续交付和部署的能力。
- 管理简单：框架自带服务治理能力，使用门槛低，可轻松管理成千上万个服务实例，保障服务高质量运行。
- 接入门槛低：完全托管的 SaaS 服务，轻资产，且无需自己部署及运维，有效降低投入成本。

高并发业务快速扩展

通过微服务产品开发互联网金融业务可提高研发效率，更灵活地响应业务变化，快速迭代创新产品，并针对热点模块进行快速扩展来提高处理能力，轻松应对突发流量，同时提高用户体验，为更多小微客户提供个性化的金融产品和交易成本较低的便捷金融服务。

- 高性能：提供基于事件驱动的架构以及私有通信协议，轻松搭建低延迟、高吞吐的服务。
- 可扩展性强：支持无限水平扩展，无性能、容量瓶颈，在蚂蚁集团内部已支撑数万个节点规模的分布式应用架构。
- 可视化管理：在分布式系统中，面对爆发式增长的应用数量和服务器数量，提供图形化的集中式管理平台，简单易用，学习成本低。

多数据中心异地多活

通过微服务产品可快速构建高可扩展、高性能的金融级分布式核心系统，拥有弹性扩容和异地多活的能力，实现技术安全自主可控，突破业务发展瓶颈，并减少开发及运维成本。实现轻型银行，助力业务快速发展和持续创新。

- 异地多活：支持同城双活、异地多活架构，具备异地容灾能力。
- 弹性扩容：支持应用级、数据库级、机房级、地域级的快速扩展。
- 自主可控：基于支付宝的业务迭代衍生完全自主研发，产品拥有完全自主知识产权，自身开源开放，并兼容开源生态。

6.使用限制

SOFAStack微服务是高性能分布式微服务框架，在使用时具有以下环境要求和限制。

限制项	限制范围	限制说明
微服务开发框架	Dubbo/SpringCloud/SOFA	服务注册中心 SDK 支持这三种框架的兼容
SOFA SDK 语言限制	Java	SOFA SDK 支持 Java 语言开发
JDK 版本	JDK 1.8 及以上	JDK 版本支持 1.8 及以上

7.基础术语

本文根据模块对微服务涉及的基础术语进行说明。

SOFARegistry

中文	英文	释义
服务注册中心	SOFARegistry	蚂蚁集团开源的一款服务注册中心产品，基于“发布-订阅”模式实现服务发现功能。同时它并不假定总是用于服务发现，也可用于其他更一般的“发布-订阅”场景。
数据	Data	在服务发现场景下，特指服务提供者的网络地址及其它附加信息。其他场景下，也可以表示任意发布到 SOFARegistry 的信息。
单元	Zone	单元化架构关键概念，在服务发现场景下，单元是一组发布与订阅的集合，发布及订阅服务时需指定单元名，更多内容可参考异地多活单元化架构解决方案。
发布者	Publisher	发布数据到 SOFARegistry 的节点。在服务发现场景下，服务提供者就是“服务提供者的网络地址及其它附加信息”的发布者。
订阅者	Subscriber	从 SOFARegistry 订阅数据的节点。在服务发现场景下，服务消费者就是“服务提供者的网络地址及其它附加信息”的订阅者。
数据标识	DataId	用来标识数据的字符串。在服务发现场景下，通常由服务接口名、协议、版本号等信息组成，作为服务的标识。
分组标识	GroupId	用于为数据归类的字符串，可以作为数据标识的命名空间，即只有 DataId、GroupId、InstanceId 都相同的服务，才属于同一服务。
实例 ID	InstanceId	实例 ID，可以作为数据标识的命名空间，即只有 DataId、GroupId、InstanceId 都相同的服务，才属于同一服务。
会话服务器	SessionServer	SOFARegistry 内部负责跟客户端建立 TCP 长连接、进行数据交互的一种服务器角色。
数据服务器	DataServer	SOFARegistry 内部负责数据存储的一种服务器角色。

元信息服务器	MetaServer	SOFARegistry 内部基于 Raft 协议，负责集群内一致性协调的一种服务器角色。
数据中心	Data Center	物理位置、供电、网络具备一定独立性的物理区域，通常作为高可用设计的重要考量粒度。一般可认为：同一数据中心内，网络质量较高、网络传输延时较低、同时遇到灾难的概率较大；不同数据中心间，网络质量较低、网络延时较高、同时遇到灾难的概率较小。

SOFARPC

中文	英文	释义
RPC	RPC	远程方法调用（Remote Procedure Call）。
RPC 服务	RPC service	服务端提供接口的实现对象。
RPC 引用	RPC reference	客户端针对 RPC 服务创建的一个代理对象。
服务 ID	service ID	服务唯一标识，由接口全路径、版本、分组与通讯协议组成的唯一标识。
服务提供方	service provider	提供 RPC 服务的应用。
服务消费方	service consumer	使用 RPC 服务的应用。
服务注册中心	Service Registry	一个独立的应用集群，用来存储和维护所有在线的 RPC 应用地址列表。
服务参数	service parameters	服务提供者可被动态修改的参数，如权重、状态。
服务发现	Service Discovery	服务消费者获取服务提供者的网络地址的过程。

动态配置

中文	英文	释义

配置类	Configuration class	业务应用中的一个普通 Java 对象，按动态配置框架的编程 API 注册后，成为一个可被外界动态管理的资源，称为配置类。域、应用、类标识三者唯一标识一个配置类实例。
域	domian	配置类的一个命名空间，默认值为 Alipay，可通过编程注解修改。
所属应用	application	配置类所属的应用名。
类标识	class ID	代表配置类的一个字符串，跟应用代码中 @DObject 注解的 ID 字段一致，通常使用全类名。
属性	attribute	配置类对象的具有公有读写方法的私有属性。一个配置类下可以有多个属性。一个配置类属性对应业务的一个配置项。
属性名	attribute name	代表属性的字符串，跟业务代码中的私有属性命名一致。
DataId	DataId	用于全局唯一标识一个属性的字符串，由域、应用、类标识、属性名四者按一定规则拼接而成。
drm-client	drm-client	动态配置框架的客户端 JAR 包。

服务治理

中文	英文	释义
运行模式	running mode	指限流 guardian 客户端对限流的处理方式，分为监控模式和拦截模式。
拦截模式	intercept mode	限流匹配上后，会实际拦截请求。
监控模式	monitor mode	限流匹配上后，不会实际拦截请求，只会打印限流记录日志。
限流后操作：空处理	post-throttling operation: null process	不做任何处理，直接返回。 对于接口方法，返回 null；对于 Web 页面，返回为空，并结束本次页面访问。