Apusic 金蝶天燕

生成时间：2026-10-23



产品白皮书

金蝶Apusic分布式配置中心v1.0

版权所有 © 深圳市金蝶天燕云计算股份有限公司2026。保留所有权利。

版权声明

本文档所涉及的软件著作权、版权等知识产权已依法进行了注册，由金蝶天燕云计算股份有限公司合法拥有。受《中华人民共和国著作权法》《计算机软件保护条例》《知识产权保护条例》和相关国际版权条约、法律、法规以及其它知识产权法律和条约的保护。未经授权许可，不得非法使用。

免责声明

本文档包含的版权信息由金蝶天燕云计算股份有限公司合法拥有，受法律的保护，金蝶天燕云计算股份有限公司对本文档可能涉及到的非金蝶天燕云计算股份有限公司的信息不承担任何责任。在法律允许的范围内，您可以查阅并仅能够在《中华人民共和国著作权法》规定的合法范围内复制和打印本文档。任何单位和个人未经金蝶天燕云计算股份有限公司书面授权许可，不得使用、修改、再发布本文档的任何部分和内容，否则将被视为侵权，金蝶天燕云计算股份有限公司有依法追究其责任的权利。

本文档如有更新，不另行通知。对本文档中的问题您可向金蝶天燕云计算股份有限公司告知或查询。未经本公司明确授予的任何权利均予保留。

商标声明

Apusic是深圳市金蝶天燕云计算股份有限公司向中华人民共和国国家商标局申请注册的注册商标，注册商标专用权由金蝶天燕合法拥有，受法律保护。未经金蝶天燕的书面许可，任何单位及个人不得以任何方式或理由对该商标的任何部分进行使用、复制、修改、传播、抄录或与其它产品捆绑使用销售。凡侵犯金蝶天燕商标权的，金蝶天燕将依法追究其法律责任。本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

目录

- 1 前言
  - 1.1 概要
  - 1.2 范围和读者
- 2 产品介绍
  - 2.1 产品概述
  - 2.2 产品介绍
  - 2.3 面向的客户
- 3 ADCC for Zookeeper介绍
  - 3.1 产品介绍
  - 3.2 设计目标
  - 3.3 产品架构
  - 3.4 产品优势
    - 3.4.1 保证性
    - 3.4.2 简单API
    - 3.4.3 高可用性
    - 3.4.4 高性能
    - 3.4.5 可靠性
    - 3.4.6 多环境支持
  - 3.5 应用场景
    - 3.5.1 **数据发布和订阅**
    - 3.5.2 **统一命名服务**
    - 3.5.3 **集群管理**
    - 3.5.4 **分布式锁**
    - 3.5.5 **分布式队列**
    - 3.5.6 **国产环境适配**
- 4 ADCC for nacos介绍
  - 4.1 产品介绍
  - 4.2 设计目标
  - 4.3 基本架构及概念
    - 4.3.1 服务 (Service)
    - 4.3.2 服务注册中心 (Service Registry)
    - 4.3.3 服务元数据 (Service Metadata)

- 4.3.4 服务提供方 (Service Provider)
- 4.3.5 服务消费方 (Service Consumer)
- 4.3.6 配置 (Configuration)
- 4.3.7 配置管理 (Configuration Management)
- 4.3.8 命名服务 (Naming Service)
- 4.3.9 配置服务 (Configuration Service)
- 4.4 逻辑架构及其组件介绍
- 4.5 产品优势
  - 4.5.1 **服务注册与发现**
  - 4.5.2 **高效配置中心**
  - 4.5.3 **易用灵活性**
  - 4.5.4 **多环境支持**
- 4.6 **应用场景**
  - 4.6.1 **分布式服务发现和配置管理**
  - 4.6.2 **动态路由**
  - 4.6.3 **服务监控和健康检查**
  - 4.6.4 **集群管理**
  - 4.6.5 **国产环境适配**

1 前言

1.1 概要

分布式配置中心在现代企业的分布式系统中具有重要的地位和作用。它能够提高配置管理效率、实现配置的动态更新、保证配置的一致性、支持多环境配置管理，简化配置管理、提高系统的可维护性、增强系统的灵活性和保障系统的安全性。随着分布式架构、微服务架构和云原生应用的发展，分布式配置中心的市场需求将持续增长。

1.2 范围和读者

本文面向技术决策者和企业架构师、应用开发工程师、应用运维工程师。

2 产品介绍

2.1 产品概述

分布式配置中心是一种用于集中管理分布式系统配置信息的关键工具。它能将分散在各个服务中的配置统一存储和管理，实现动态更新配置而无需重新部署应用。其核心设计理念包括中心化管理以确保配置统一存储与控制；支持动态更新，实时推送变化且应用无需重启；通过冗余设计和故障切换保障高可用性；采用严格权限管理、加密存储和审计日志确保安全可靠；同时提供简单易用的界面、自动化配置以及良好的文档和支持，方便用户高效管理配置。

2.2 产品介绍

金蝶 Apusic 分布式配置中心（Apusic Distributed Config Center，简称 ADCC）是金蝶天燕研发的一款提供配置管理和服务的高性能协调服务产品，旨在为企业级应用提供高效、可靠的集中配置解决方案。它将配置信息从应用程序中分离出来，存储在一个中央位置，使得配置信息可以被多个应用程序实时访问和获取。当需要修改配置信息时，只需要修改中央副本就能实现统一更新，提高了配置管理的效率和安全性。

它提供统一配置管理、实时更新功能，支持配置变更订阅，实现动态配置修改，有效解决配置的动态变更、持久化、运维成本等问题。

ADCC推出ADCC for Zookeeper和ADCC for Nacos版本，用户可根据其应用场景和需求选取。

2.3 面向的客户

微服务架构的应用企业、分布式系统企业、需要高可用性和可拓展性的应用企业。

3 ADCC for Zookeeper介绍

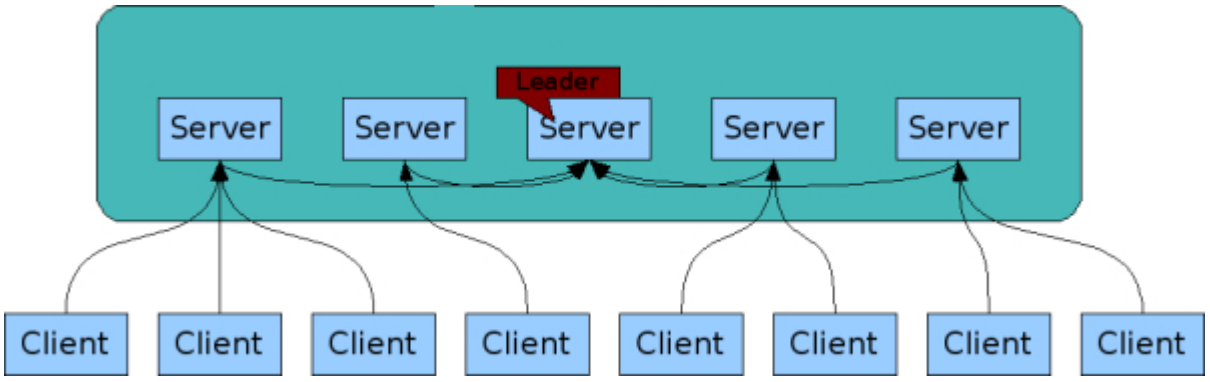
3.1 产品介绍

ADCC for Zookeeper是一款为分布式应用系统提供高性能协调服务的产品，针对分布式技术而产生的新型的配置手段，通过简单的接口能够对分布式应用系统的配置进行统一的管理、实时更新，并支持配置更改的订阅，实现不需要重启服务器而动态的修改配置文件的内容，有效提高工作效率。

同时，ADCC for Zookeeper为分布式应用提供一致性、可靠性和高可用性，支持配置信息的集中管理和动态更新，允许节点在系统中注册和发现服务，提供分布式锁机制解决资源竞争问题，并可以实现分布式队列用于消息传递和任务调度。

3.2 设计目标

- **灵活简便**，它允许分布式进程通过共享的分层命名空间相互协调，该命名空间的组织方式类似于标准文件系统。名称空间由数据寄存器组成，用ADCC的说法称为znodes，这些寄存器类似于文件和目录。与设计用于存储的典型文件系统不同，ADCC数据保存在内存中，这意味着可以实现高吞吐量和低延迟数。
- **高性能、高可用性和严格有序的访问**，意味着它可以用于大型分布式系统，可靠性方面使其不会成为单一故障点。严格的排序意味着可以在客户端实现复杂的同步原语。
- **可复制性**，与它协调的分布式进程一样，ADCC本身旨在通过一组称为集成的主机进行复制。



组成ADCC服务的服务器必须相互了解。它们在持久存储中维护状态的内存映像以及事务日志和快照。只要大多数服务器可用，ADCC服务将可用。

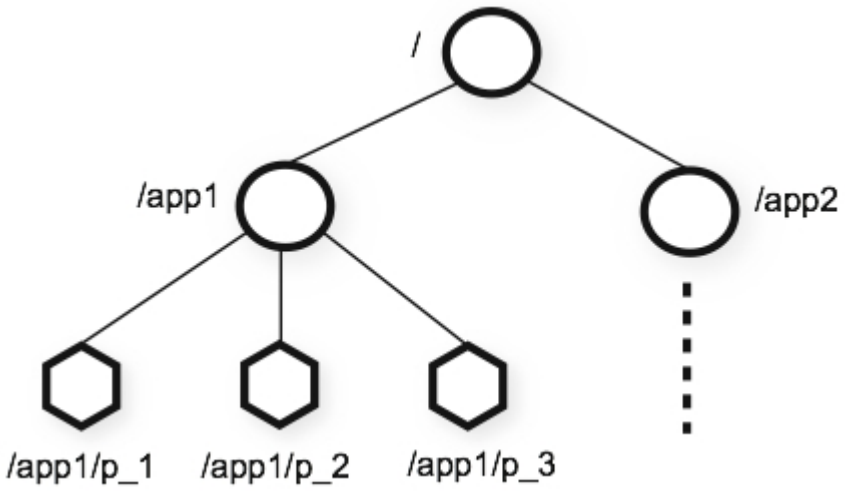
客户端连接到单个ADCC服务器。客户端维护TCP连接，通过该连接发送请求、获取响应、获取监视事件和发送心跳。如果与服务器的TCP连接中断，客户端将连接到其他服务器。

- **有序性**，ADCC用一个数字标记每个更新，该数字反映了所有ADCC事务的顺序。后续操作可以使用该命令来实现更高级别的抽象，例如同步原语。

- **快速性**，它在"读主导"工作负载中特别快。ADCC应用程序在数千台机器上运行，在读比写更常见的情况下，它的性能最好，大约为10:1。

3.3 产品架构

ADCC提供的名称空间与标准文件系统的名称空间非常相似。名称是由斜杠（/）分隔的路径元素序列。ADCC命名空间中的每个节点都由路径标识。ADCC的分层命名空间架构如下：



- 节点和短暂节点

与标准文件系统不同，ADCC命名空间中的每个节点都可以有与其关联的数据以及子节点。这就像拥有一个文件系统，允许一个文件同时也是一个目录。（ADCC被设计用于存储协调数据：状态信息、配置、位置信息等，因此每个节点存储的数据通常很小，在字节到千字节的范围内。）我们使用术语znode来明确我们所指的是ADCC数据节点。

zNode维护一个统计结构，其中包括数据更改、ACL更改和时间戳的版本号，以允许缓存验证和协调更新。每次znode的数据更改，版本号都会增加。例如，每当客户端检索数据时，它也会接收数据的版本。

存储在命名空间中每个znode处的数据以原子方式读取和写入。读操作获取与znode关联的所有数据字节，写操作替换所有数据。每个节点都有一个访问控制列表（ACL），限制谁可以做什么。

ADCC也有短暂节点的概念。只要创建znode的会话处于活动状态，这些znode就存在。会话结束时，znode将被删除。

- 条件更新和监视

ADCC支持的监视概念。客户可以在znode上设置监视。当znode发生变化时，将触发并移除监视。当监视被触发时，客户端接收到一个数据包，表示znode已更改。如果客户端与ADCC服务器之一之间的连接断开，客户端将收到本地通知。

3.4 产品优势

3.4.1 保证性

ADCC非常快速和简单。但是，由于其目标是作为构建更复杂的服务（如同步）的基础，因此它提供了一组保证。这些是：

- 顺序一致性-客户端的更新将按应用发送的顺序。
- 原子性-更新要么成功要么失败，没有部分成功。
- 单个系统映像-无论客户端连接到哪个服务器，客户端都将看到相同的服务视图。即使客户端故障切换到具有相同会话的不同服务器，客户端也永远不会看到系统的旧视图。
- 可靠性-一旦应用了更新，它将从那时起一直存在，直到客户端覆盖更新。
- 及时性-保证客户对系统的看法在一定时间范围内是最新的。

3.4.2 简单API

ADCC的设计目标之一是提供一个非常简单的编程API。因此，它只支持以下操作：

创建：在树中的某个位置创建节点

删除：删除节点

存在：测试某个位置是否存在节点

获取数据：从节点读取数据

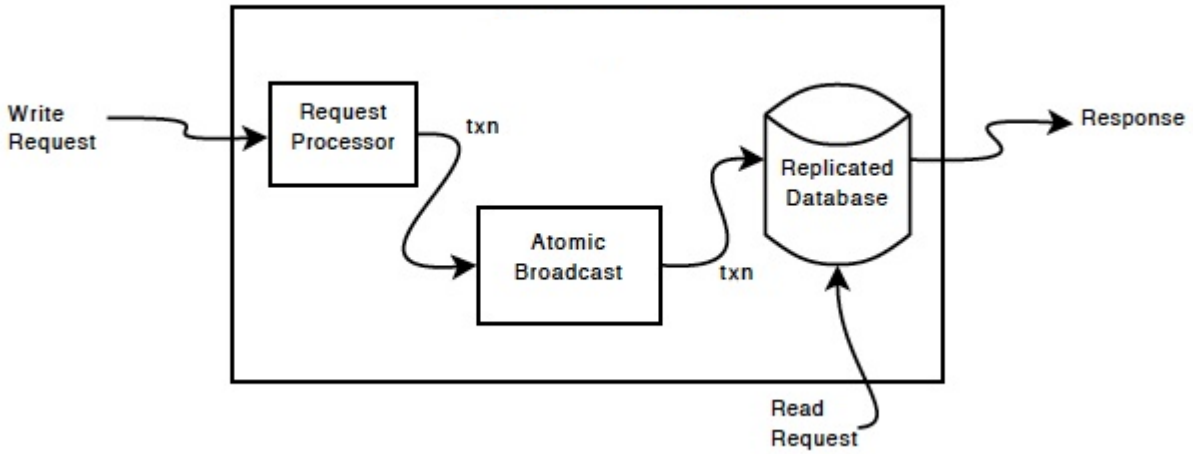
设置数据：将数据写入节点

获取子节点：检索节点的子节点列表

同步：等待数据传播

3.4.3 高可用性

下图显示ADCC服务的高级组件。除了请求处理器之外，组成ADCC服务的每个服务器都复制每个组件的自己副本。



复制数据库是包含整个数据树的内存数据库。更新将记录到磁盘以实现可恢复性，写入将在应用到内存数据库之前序列化到磁盘。

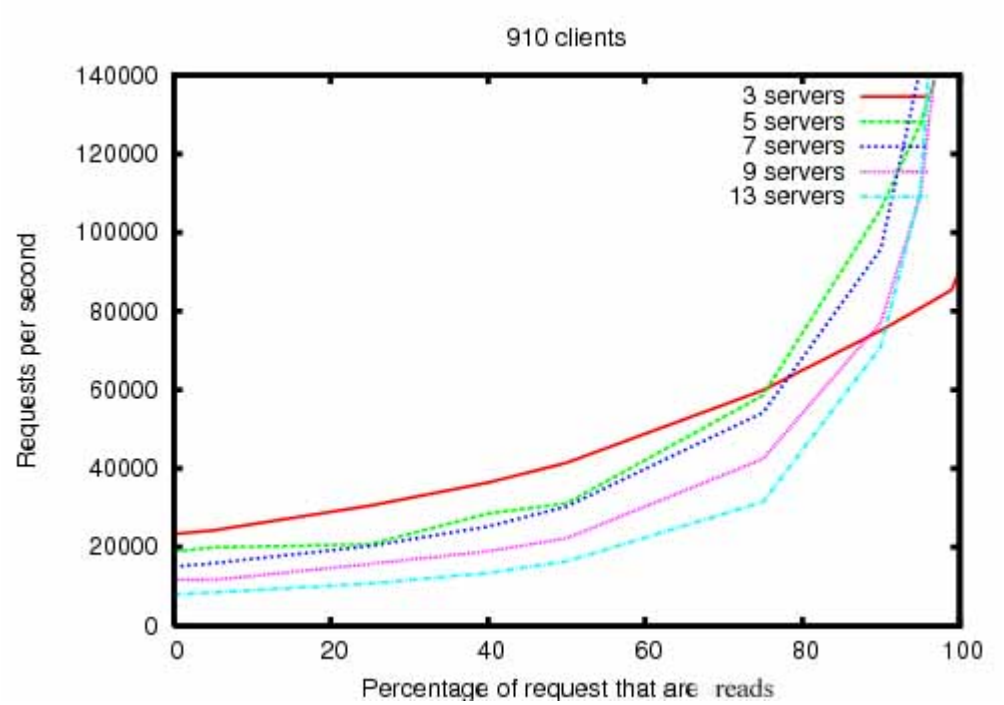
每个ADCC服务器都为客户端提供服务。客户端只连接到一台服务器以提交请求。读取请求由每个服务器数据库的本地副本提供服务。更改服务状态的请求（写请求）由协议协议处理。

作为协议协议的一部分，来自客户端的所有写请求都被转发到一个称为leader的服务器。其余的ADCC服务器称为followers，从leader那里接收消息建议并同意消息传递。消息传递层负责在失败时替换leader，并将followers与leader同步。

ADCC使用自定义的原子消息传递协议。由于消息传递层是原子的，ADCC可以保证本地副本不会发散。当领导者接收到写请求时，它计算应用写时系统的状态，并将其转换为捕获此新状态的事务。

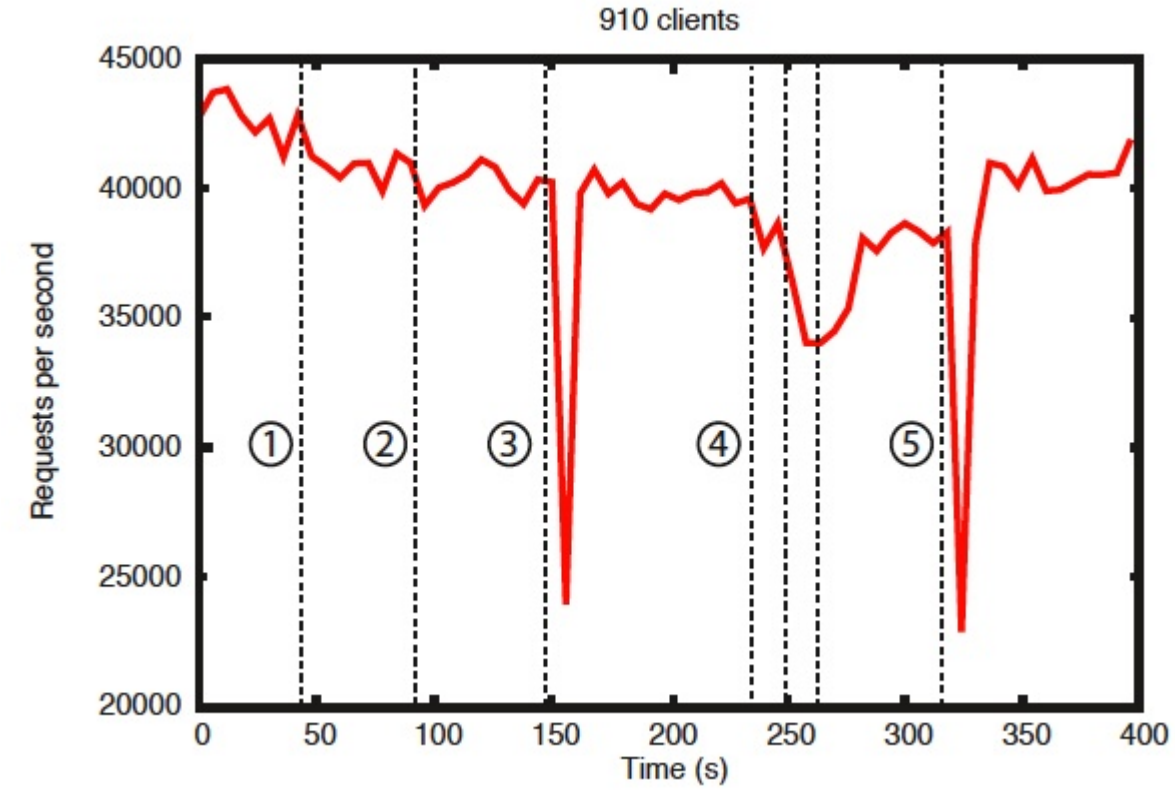
3.4.4 高性能

ADCC旨在实现高性能。在读写次数较多的应用程序中，它的性能尤其高，因为写操作涉及同步所有服务器的状态。



3.4.5 可靠性

为了显示注入故障时系统随时间的行为，我们运行了由7台机器组成的ADCC服务。我们运行了与之前相同的饱和基准，但这次我们将写百分比保持在恒定的30%，这是我们预期工作负载的保守比率。



从这张图中有几个重要的观察结果。首先，如果跟随器出现故障并快速恢复，那么ADCC即使出现故障也能够保持高吞吐量。但也许更重要的是，Leader选举算法允许系统足够快地恢复，以防止吞吐量大幅下降。根据我们的观察，ADCC只需不到200秒就可以选出新的Leader。第三，随着followers的恢复，ADCC能够在开始处理请求后再次提高吞吐量。

3.4.6 多环境支持

产品广泛兼容适配国内外主流软硬件平台，如鲲鹏、飞腾、海光、龙芯、海思、统信UOS、麒麟、中科方德、Linux等。

3.5 应用场景

3.5.1 数据发布和订阅

发布者可以将数据发布到ADCC的节点上，供订阅者动态获取数据，实现配置信息的集中式管理和动态更新。

3.5.2 统一命名服务

将集群中的机器名称、IP地址、服务的唯一标识等与实际地址进行映射。客户端可以通过这些名称或标识来访问服务，而不需要知道服务的实际地址。

3.5.3 集群管理

提供了实时监控系统对集群的管理功能，实时感知集群中机器存活性，及时发现并处理异常。

3.5.4 分布式锁

通过创建特殊的节点来实现分布式锁，保证分布式系统中的一致性。各个节点可以竞争同一个锁，确保在并发环境下数据的安全和准确性。

3.5.5 分布式队列

通过创建一个特殊的节点，各个节点可以加入或离开队列，同时队列中的节点可以按照一定的顺序进行排序和执行。

3.5.6 国产环境适配

适应国产环境的分布式配置中心软件，可与Zookeeper无缝切换。

4 ADCC for nacos介绍

4.1 产品介绍

ADCC for Nacos是一个动态服务发现、配置管理和服务管理平台，构建以服务为中心的现代应用架构（微服务、云原生）的基础设施，致力于微服务的发现、管理和信息配置，帮助开发者快速实现动态服务发现、服务配置、服务元数据及流量管理，从而更敏捷、更简单的构建、交付和管理微服务平台。

同时，ADCC for Nacos提供丰富的API、可视化统一管理控制台等功能，方便用户进行服务和配置管理。支持多种主流框架和协议，具有高可用性和可扩展性，能够满足大规模分布式系统的需求。

4.2 设计目标

ADCC For Nacos 支持几乎所有主流类型的"服务"的发现、配置和管理，包括：

Kubernetes Service

gRPC & Dubbo RPC Service

Spring Cloud RESTful Service

ADCC For Nacos 的关键特性包括:

- **服务发现和服务健康监测**

  当服务提供者注册了服务后，ADCC For Nacos就会对服务进行实时健康检测，以防止向不健康的的主机或服务实例发送请求，ADCC For Nacos还提供一些健康检测的仪表盘，帮助管理服务的可用性和流量。

  服务消费者通过基于DNS或基于RPC（Dubbo/gRPC）的机制发现并消费服务。

- **动态配置服务**

  动态配置消除了配置变更时重现部署应用和服务的需要，配置中心化管理让实现无状态服务变得更简单，让服务按需弹性扩展更容易。

  通过ADCC For Nacos提供的管理控制台，用户可以管理所有应用和服务的配置，并提供包括配置版本跟踪、金丝雀发布、一键回滚配置以及客户端配置状态跟踪等一系列开箱即用的配置管理特性。

- **动态 DNS 服务**

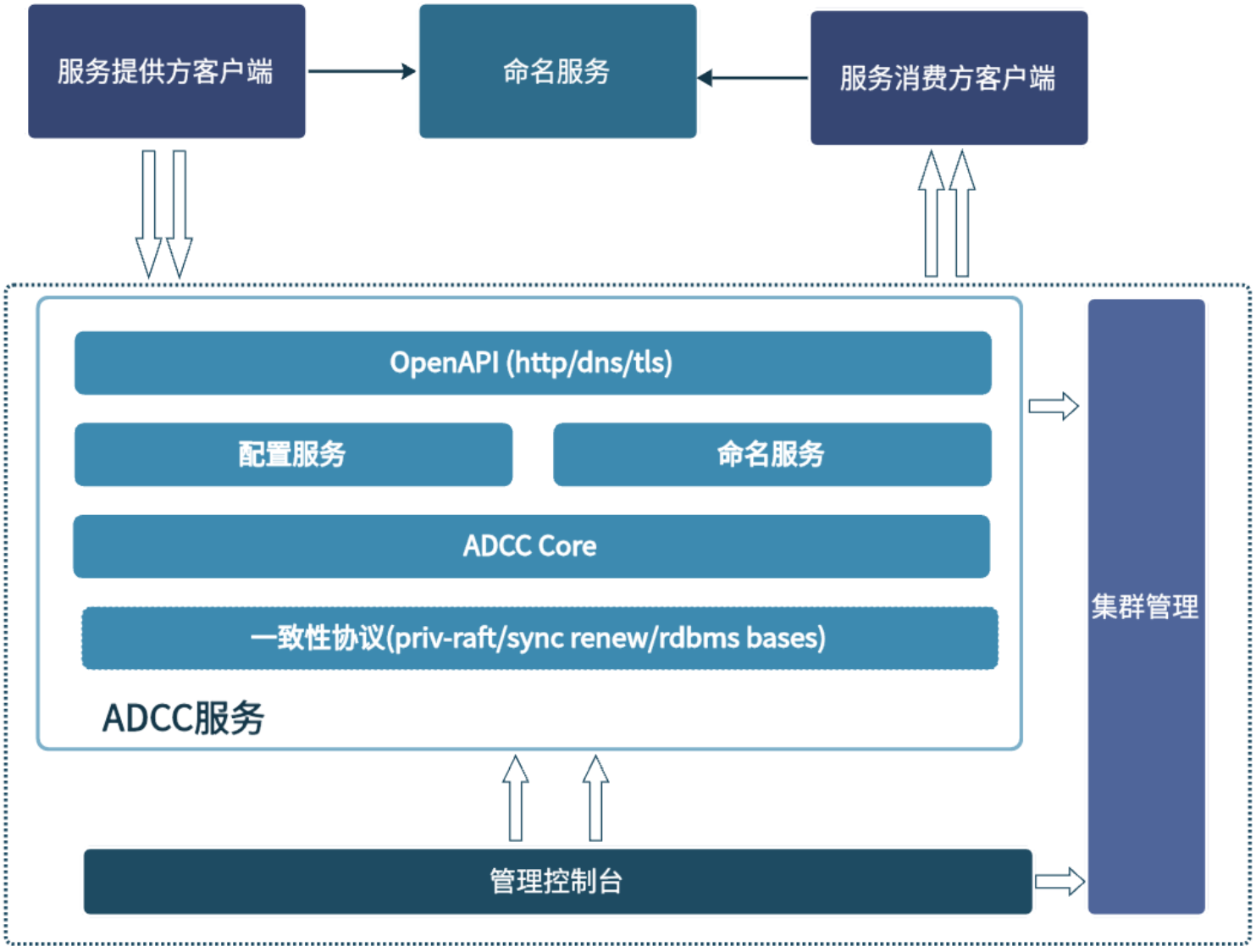
  ADCC For Nacos的动态DNS服务支持权重路由，更容易实现灵活的路由策略。

- **服务及其元数据管理**

ADCC For Nacos能让管理者以全局视角来评估整个微服务平台，提供包括服务描述、服务生命周期、服务健康状态、流量管理、路由及安全策略、以及metrics统计数据。

4.3 基本架构及概念

ADCC for nacos架构图如下：



4.3.1 服务 (Service)

服务是指一个或一组软件功能（例如特定信息的检索或一组操作的执行），其目的是不同的客户端可以为不同的目的重用（例如通过跨进程的网络调用）。ADCC 支持主流的服务生态，如 Kubernetes Service、gRPC|Dubbo RPC Service 或者 Spring Cloud RESTful Service。

4.3.2 服务注册中心 (Service Registry)

服务注册中心，它是服务，其实例及元数据的数据库。服务实例在启动时注册到服务注册表，并在关闭时注销。服务和路由器的客户端查询服务注册表以查找服务的可用实例。服务注册中心可能会调用服务实例的健康检查 API 来验证它是否能够处理请求。

4.3.3 服务元数据 (Service Metadata)

服务元数据是指包括服务端点(endpoints)、服务标签、服务版本号、服务实例权重、路由规则、安全策略等描述服务的数据。

4.3.4 服务提供方 (Service Provider)

是指提供可复用和可调用服务的应用方。

4.3.5 服务消费方 (Service Consumer)

是指会发起对某个服务调用的应用方。

4.3.6 配置 (Configuration)

在系统开发过程中通常会将一些需要变更的参数、变量等从代码中分离出来独立管理，以独立的配置文件的形式存在。目的是让静态的系统工件或者交付物（如 WAR，JAR 包等）更好地和实际的物理运行环境进行适配。配置管理一般包含在系统部署的过程中，由系统管理员或者运维人员完成这个步骤。配置变更是调整系统运行时的行为的有效手段之一。

4.3.7 配置管理 (Configuration Management)

在数据中心中，系统中所有配置的编辑、存储、分发、变更管理、历史版本管理、变更审计等所有与配置相关的活动统称为配置管理。

4.3.8 命名服务 (Naming Service)

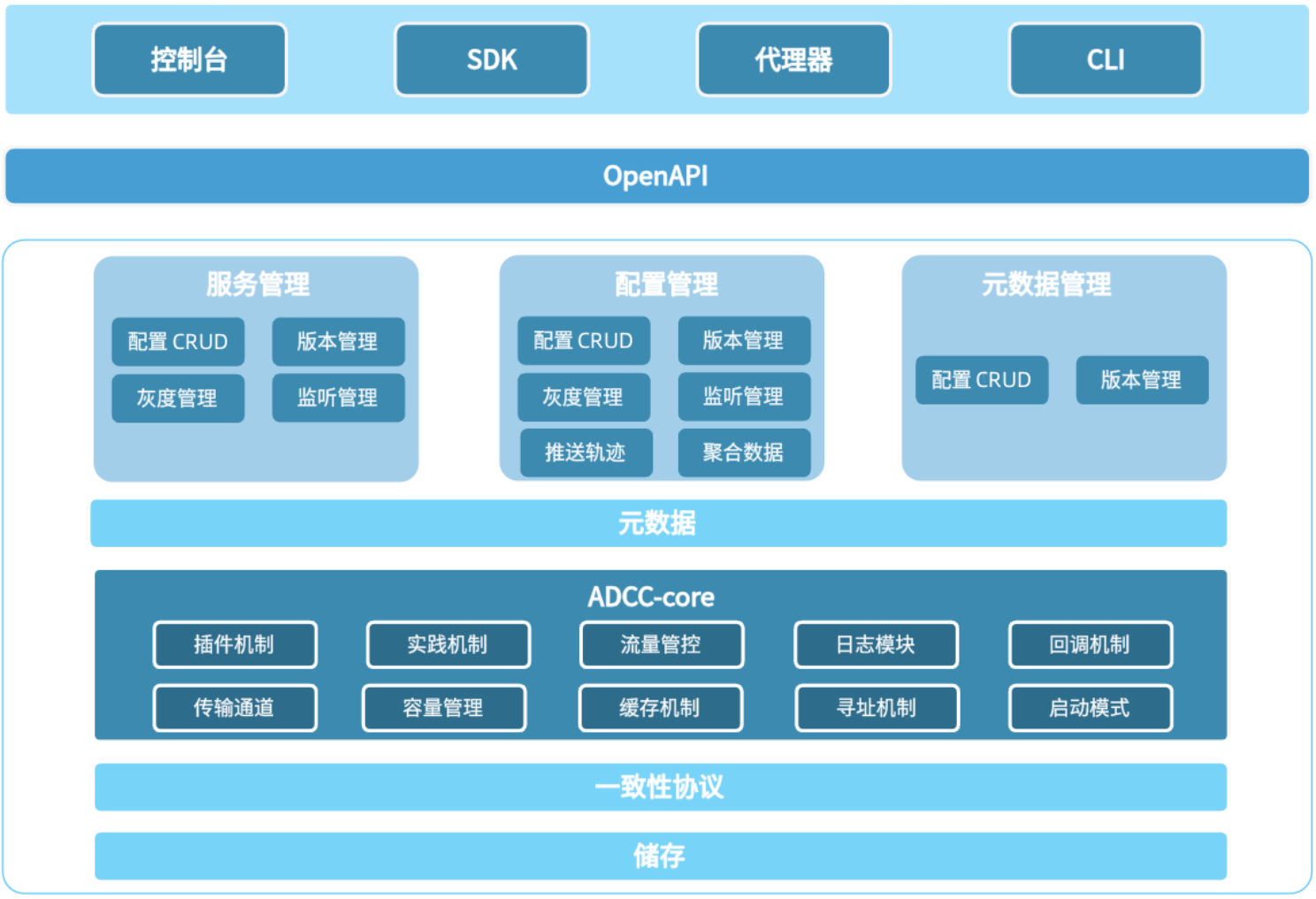
提供分布式系统中所有对象(Object)、实体(Entity)的“命名”到关联的元数据之间的映射管理服务，例如 ServiceName -> Endpoints Info, Distributed Lock Name -> Lock Owner/Status Info, DNS Domain Name -> IP List, 服务发现和 DNS 就是命名服务的2大场景。

4.3.9 配置服务 (Configuration Service)

在服务或者应用运行过程中，提供动态配置或者元数据以及配置管理的服务提供者。

4.4 逻辑架构及其组件介绍

ADCC for nacos的逻辑架构及其组件如下图：



- 服务管理：实现服务CRUD（创建、读取、更新、删除），域名CRUD，服务健康状态检查，服务权重管理等功能
- 配置管理：实现配置管CRUD，版本管理，灰度管理，监听管理，推送轨迹，聚合数据等功能
- 元数据管理：提供元数据CRUD 和打标能力
- 插件机制：实现三个模块可分可合能力，实现扩展点SPI机制
- 事件机制：实现异步化事件通知，sdk数据变化异步通知等逻辑
- 日志模块：管理日志分类，日志级别，日志可移植性（尤其避免冲突），日志格式，异常码+帮助文档
- 回调机制：sdk通知数据，通过统一的模式回调用户处理。接口和数据结构需要具备可扩展性
- 寻址模式：解决ip，域名，nameserver、广播等多种寻址模式，需要可扩展
- 推送通道：解决server与存储、server间、server与sdk间推送性能问题
- 容量管理：管理每个租户，分组下的容量，防止存储被写爆，影响服务可用性
- 流量管理：按照租户，分组等多个维度对请求频率，长链接个数，报文大小，请求流控进行控制
- 缓存机制：容灾目录，本地缓存，server缓存机制。容灾目录使用需要工具
- 启动模式：按照单机模式，配置模式，服务模式，dns模式，或者all模式，启动不同的程序+UI
- 一致性协议：解决不同数据，不同一致性要求情况下，不同一致性机制
- 存储模块：解决数据持久化、非持久化存储，解决数据分片问题
- OpenAPI：暴露标准Rest风格HTTP接口，简单易用，方便多语言集成

- 控制台：易用控制台，做服务管理、配置管理等操作
- SDK：多语言sdk
- 代理器：dns-f类似模式，或者与mesh等方案集成
- CLI：命令行对产品进行轻量化管理，像git一样好用

4.5 产品优势

4.5.1 服务注册与发现

支持基于DNS和HTTP的服务注册和发现，轻松实现服务的自动化注册和发现，减少了服务间通信的复杂程度。

4.5.2 高效配置中心

支持各种数据格式的配置管理，包括properties、XML、JSON等，同时支持动态配置更新。

4.5.3 易用灵活性

支持Java、Go等多种编程语言，可以让开发人员轻松地在不同编程语言中使用；提供了开发的API接口，可以方便地与其他系统进行集成。提供可视化统一管理控制台。支持集群管理。

4.5.4 多环境支持

产品广泛兼容适配国产主流软硬件平台，如龙芯、麒麟、达梦、人大金仓、Linux等。

4.6 应用场景

4.6.1 分布式服务发现和配置管理

用于构建和管理现代化的分布式、微服务化和容器化应用程序；为应用程序的发现、配置和管理提供了一种集中且可拓展的解决方案。

4.6.2 动态路由

可与服务网关集成实现动态路由，根据服务实例的可用性和负载来路由请求。

4.6.3 服务监控和健康检查

可以用于监控服务的健康状态，并通过心跳检查来检测不健康的服务实例。

4.6.4 集群管理

可以统一管理各个节点和实例；可以动态扩展或缩减服务实例，以应对负载的变化。

4.6.5 国产环境适配

适应国产环境的分布式配置中心软件，可与nacos无缝切换。

全国统一服务热线

4008-555-800



金蝶天燕云计算股份有限公司(简称“金蝶天燕云”)成立于2000年，前身为“金蝶中间件公司”，是金蝶集团旗下新一代软件基础云平台服务商，云计算国家标准制定企业，国家信创产业核心软件企业。金蝶天燕是国家863重点研发计划与核高基重大专项承接企业，也是“两网一站四库十二金”国家重点工程的基础平台提供商，产品广泛应用于政府、军工、金融、能源等关键行业，累计服务客户总数超过10万家。

Apusic 金蝶天燕

云计算国家标准制定企业

金蝶集团旗下基础软件企业

信息技术应用创新核心企业

官网：www.apusic.com