统一的边缘侧软硬件架构

设计标准及参考设计

智能网联研究中心

2025年1月10日星期五

**文档信息**

|  |  |  |  |
| --- | --- | --- | --- |
| **文档名称** | 统一的边缘侧软硬件架构设计标准及参考设计 | | |
| **文档编号** |  | **版本号** | V0.4 |
| **编写人** | 李玉展、赵桥驿、张伟、肖光明、朱毅、陶勇刚 | **编写日期** | 2025年1月 |
| **审核人** | 李强、沈峰 | **审核日期** |  |
| **版本历史** | v0.1~0.3 | **修改内容** | 初始化 |
|  | v0.4 | **修改内容** | 增加消息主题规则设计 |

目录

[一、架构设计目标 5](#_Toc187741684)

[1.1、信息汇聚和分发 5](#_Toc187741685)

[1.2、信息处理与增强 5](#_Toc187741686)

[1.3、边缘侧算法和应用的管理 5](#_Toc187741687)

[1.4、规划边缘应用范畴 5](#_Toc187741688)

[1.5、划分对外接口和对接方式 5](#_Toc187741689)

[二、整体架构设计 6](#_Toc187741690)

[2.1、技术架构图 6](#_Toc187741691)

[2.2、 实例架构图 6](#_Toc187741692)

[2.3、架构说明 7](#_Toc187741693)

[边缘应用分类： 7](#_Toc187741694)

[2.3.1、边缘应用管理平台 7](#_Toc187741695)

[2.3.2、公共边缘应用服务 7](#_Toc187741696)

[2.3.3、通用边缘应用 7](#_Toc187741697)

[2.4、基础硬件环境 7](#_Toc187741698)

[2.5、感知设备层 7](#_Toc187741699)

[2.6、通信服务（数据总线等） 8](#_Toc187741700)

[2.7、适配服务 8](#_Toc187741701)

[2.7、存储服务 8](#_Toc187741702)

[2.8、成熟算法和应用集成 9](#_Toc187741703)

[2.9、通用的公共服务（由集成管理平台集成管理） 14](#_Toc187741704)

[2.10、其他公共服务 15](#_Toc187741705)

[2.11、非功能性需求： 16](#_Toc187741706)

[2.11.1. 性能要求 16](#_Toc187741707)

[2.11.2. 安全要求 17](#_Toc187741708)

[2.11.3. 可靠性要求 18](#_Toc187741709)

[2.11.4. 性能参数调优 18](#_Toc187741710)

[2.11.5．建议 18](#_Toc187741711)

[三、标准化的数据规范 19](#_Toc187741712)

[3.1 数据逻辑流图 19](#_Toc187741713)

[3.2 数据指标项 20](#_Toc187741714)

[3.3 数据主题规则 25](#_Toc187741715)

[3.3.1 集成平台主题规则设计参考 25](#_Toc187741716)

[3.3.2 业务主体规则设计参考 26](#_Toc187741717)

[四、数据总线和集成中台设计 26](#_Toc187741718)

[4.1. 数据总线架构概述 26](#_Toc187741719)

[4.1.1 核心设计理念 28](#_Toc187741720)

[4.1.3. 核心组件 29](#_Toc187741721)

[4.1.4. 关键特性 31](#_Toc187741722)

[4.1.4. 技术优势 31](#_Toc187741723)

[4.1.5. 应用场景 34](#_Toc187741724)

[4.2 集成平台概要设计 35](#_Toc187741725)

[4.2.1. 节点管理模块 35](#_Toc187741726)

[4.2.2. 应用管理模块 36](#_Toc187741727)

[4.2.3. 数据总线管理模块 37](#_Toc187741728)

[4.2.4. 数据源管理模块 38](#_Toc187741729)

[4.2.4. 安全管理模块 39](#_Toc187741730)

[4.2.5. 运维管理模块 40](#_Toc187741731)

[4.2.5.2 运维自动化 40](#_Toc187741732)

[4.2.7. 工具集模块 40](#_Toc187741733)

[4.2.8集成中台边缘侧功能概要设计 41](#_Toc187741734)

[五、边缘应用平台规范和约束 47](#_Toc187741735)

[5.1、数据总线规范 48](#_Toc187741736)

[5.1.1 应用区域划分 48](#_Toc187741737)

[5.1.2 外部通信统一由总线侧处理： 48](#_Toc187741738)

[5.1.3 应用必须遵循以下职责： 48](#_Toc187741739)

[5.2、应用部署规范 48](#_Toc187741740)

[5.2.1支持的应用类型： 48](#_Toc187741741)

[5.2.2 硬件支持： 48](#_Toc187741742)

[5.2.3 应用部署要求： 49](#_Toc187741743)

[5.3、监控运维规范 49](#_Toc187741744)

[5.3.1系统维度监控： 49](#_Toc187741745)

[5.3.2 应用进程维度监控： 49](#_Toc187741746)

[5.3.3 业务指标监控： 49](#_Toc187741747)

[5.4、自检报告机制 49](#_Toc187741748)

[5.4.1应用级自检(按需) 49](#_Toc187741749)

[5.4.2 节点级自检(按需) 50](#_Toc187741750)

[5.4.3 平台级自检(按需) 51](#_Toc187741751)

[5.5、数据同步规范 53](#_Toc187741752)

[5.6、报警机制 53](#_Toc187741753)

[5.7、协议与扩展 53](#_Toc187741754)

# 一、架构设计目标

通过收集和整理边缘应用，逐步统一和规范边缘应用生态，形成轻量化边缘应用平台，并能依靠平台积累和迭代优化边缘应用，方便的联通各方数据，相对的安全可控，进而推动技术统一迭代和升级，最终达到丰富业务功能的目的。

## 1.1、信息汇聚和分发

部署在网络边缘侧的软硬件，具备高带宽低时延的能力，能够提高业务质量，降低业务时延，提升业务安全等级。如雷达、相机数据的采集，V2V、V2I的信息转发，车辆与车辆直接的感知数据共享，动态高精地图及视频流录制和下发等。

## 1.2、信息处理与增强

利用边缘侧硬件的部分存储及计算能力进行信息处理、增强以及合成。如点云数据的融合处理、视频流的实时分析与处理、违章预警、危险驾驶处理、交叉路口通信及大范围协同调度等。

## 1.3、边缘侧算法和应用的管理

通过统一的边缘侧算法/应用软件环境，集成成熟的算法和边缘应用，完成算法/应用注册、算法/应用管控、算法/应用更新等；同时为其他边缘应用提供标准化的接入途径，利用贯穿上下的总线系统进行数据的交互和处理，为算法/应用提供全生命周期的管理和调度。

## 1.4、规划边缘应用范畴

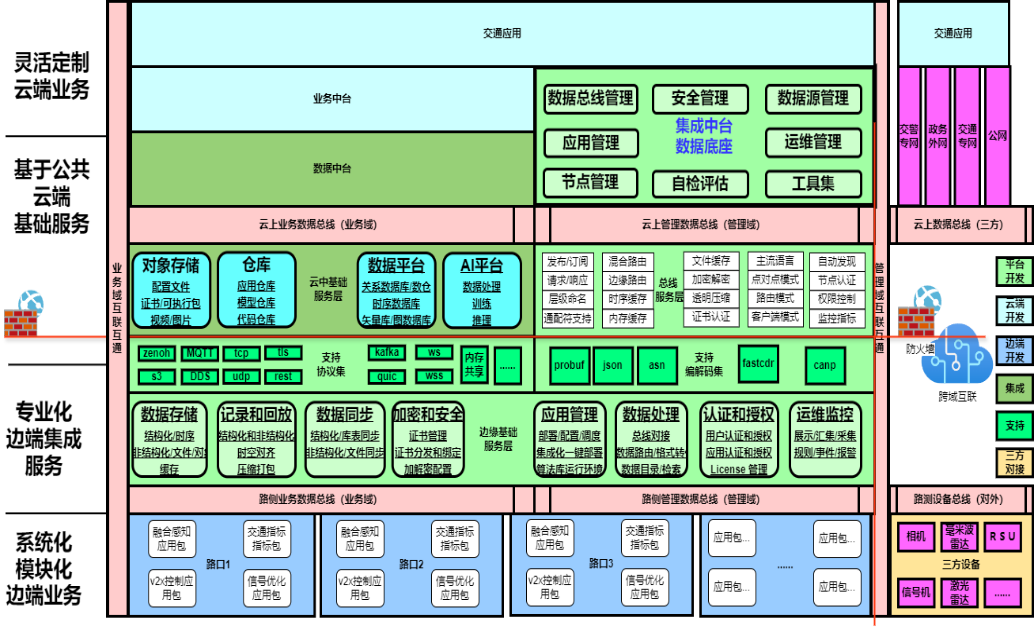
通过调研边缘应用，确认未来边缘应用的范畴，对内自主研发的主要产品图谱，内部需要支撑的公共功能，涉及的主要技术范围。

## 1.5、划分对外接口和对接方式

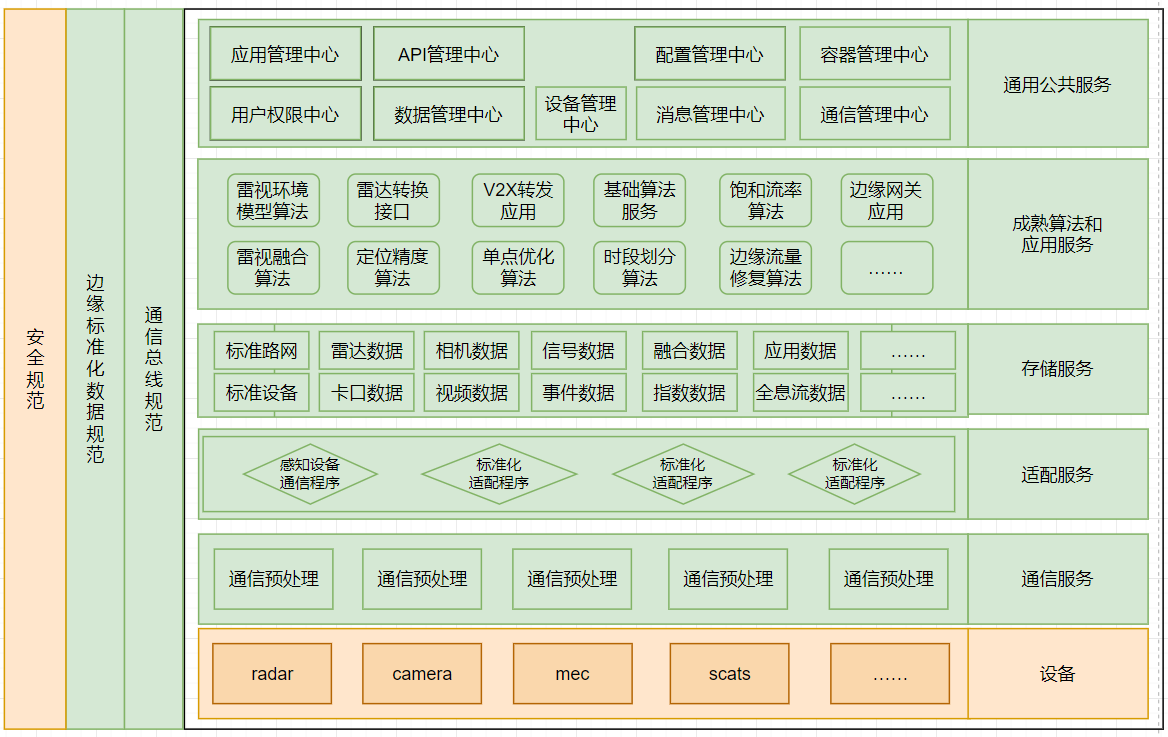
划分那些工作是边缘云应用承担的，那些是需要外部支撑或者对接的，从而划分内外工作界面，便于工作划分和有序化。

# 二、整体架构设计

## 2.1、技术架构图



## 实例架构图



## 2.3、架构说明

### 边缘应用分类：

### 2.3.1、边缘应用管理平台

主要包括：集成管理平台；

### 2.3.2、公共边缘应用服务

主要包括：数据总线、数据存储（数据和文件）、数据同步转发（结构化和非结构化）、安全加密、监控运维等；

### 2.3.3、通用边缘应用

主要包括：各种业务功能软件，包括应用类和算法类两类功能软件；后续由整理清单

## 2.4、基础硬件环境

1. 基于Arm 或 x86硬件环境，部分带有GPU卡
2. 基于linux的操作系统，ubuntu或者debian
3. 基本存储SSD硬盘2T+，根据需要集成GPU卡
4. 有线网，无线网4G/5G支持等

## 2.5、感知设备层

涉及多类设备，雷达，相机，雷视一体机，MEC，信号机等

1. 雷达：电科智能（型号：xxxx）首选，已完成边缘侧统一架构的集成；其他雷达设备暂不支持，需要按统一总线规范开发对接后接入系统；
2. 相机：电科智能（型号：xxxx）首选，已完成边缘侧统一架构的集成；其他设备需要按照总线统一规范进行标定开发对接后，接入系统。
3. 雷视一体机：电科智能（型号：xxxx）首选，已完成边缘侧统一架构的集成；其他雷视设备暂不支持，需要按统一总线规范开发对接后接入系统；
4. MEC：电科智能（型号：xxxx）必选，已按照本标准完成软硬件环境测试；
5. 信号机：骏马（型号：xxx），Scats（型号：xxx）,已完成边缘侧统一架构的集成；

## 2.6、通信服务（数据总线等）

负责对接设备层设备，负责原始感知数据的获取和转发，一般采用TCP/UDP/IP、ModBus、MQTT、GPRC协议对接各设备厂家，通信服务侧设备对接需要按照实际设备型号进行一一对接和处理；

设备：雷达：型号xxx，雷视一体机：型号xxx，相机：型号xxxx，MEC：型号xxx，信号机：xxxx型号，已完成集成和对接测试。

## 2.7、适配服务

适配服务完成原始感知数据到标准数据的转换和处理，完成感知数据的上传和下达，数据规范参照《边缘计算标准化数据规范》，其中涉及：

路侧感知数据：路侧安全消息 RSM数据，路侧地图数据 MAP数据，信号灯相位与配时消息 SPAT数据，路侧单元信息 RSI数据，路侧安全消息 RSM数据；

车辆动态数据：车辆基本安全消息 BSM数据；

设备自身数据：路侧配置信息数据，路侧设备状态数据，路侧设备心跳数据；

控制类数据：路侧单元RSU信息下发数据，路侧感知事件确认数据，路侧感知事件取消数据；

交通类数据：交通参与者数据，交通流统计数据；

适配服务需要按照设备一一完成对接和集成测试，目前已完成对设备（雷达：xxx型号，雷视一体机：xxxx型号，相机：xxxx型号，信号机：xxxx型号的对接和测试）的集成。其他外部设备需要按照统一总线规范进行对接和集成，参照 统一总线规范章节，获取集成指南；

## 2.7、存储服务

存储服务涉及存储软件的选择和存储数据的定义。定义已经确定的数据的表、视图、方法等，非关系库的键和簇等，对象库的各类图片、视频的存储路径定义等；关系库中表已T\_开头，视图已VW\_开头，方法已FUN\_开头；非关系库键KEY\_，簇以FM\_开头；图片和视频存储路径，以设备所处位置为基准，辅以雪花ID组成，图片：images/device\_id/snow\_id\_xxx.jpg（xxx考虑有多个图片，以001序号增加），视频：video/device\_id/snow\_id\_xxxx.mp4（xxxx如上）

1. 支持的存储软件环境：

* 关系数据库

MySQL、TiDB、PostgreSQL

* 时序数据库

InfluxDB、TimescaleDB

* 对象存储库

MinIO

1. 目前已确定的存储数据：

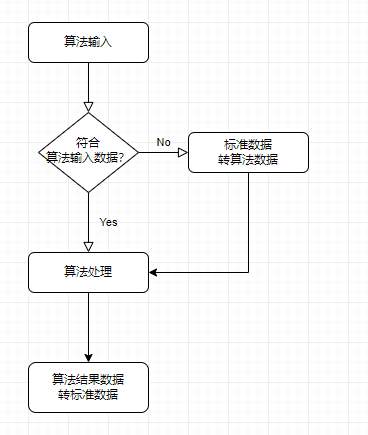
|  |  |
| --- | --- |
| 标准路网 | 区域，道路，路段，路口，方向定义等 |
| 标准设备 | 雷达，相机，雷视设备，MEC，RSU，信号机等 |
| 雷达数据 | 点云数据 |
| 相机数据 | 点云和卡口数据 |
| 区域\路口定义数据 | 路口信息，路段信息，道路信息，MAP数据 |
| 感知数据 | RSM、SPAT、RSI、RSM数据 |
| 融合感知数据 | 交通流统计数据，交通参与者数据 |
| 图片和视频数据 | 录制、回放类，事件类视频片段 |

需要补充，具体详细的结构定义《标准化数据规范》

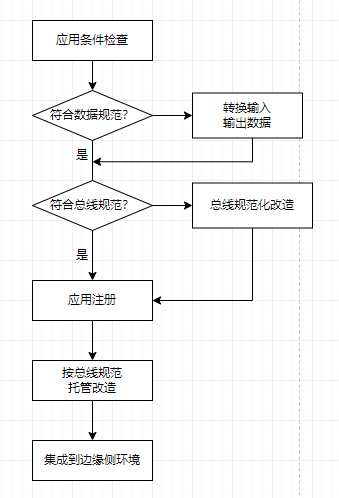
## 2.8、成熟算法和应用集成

目前已经集成如下成熟算法和应用，后续根据公司和项目产品需求，继续集成成熟的算法和应用；根据数据规范和应用集成规范，应按照算法/应用接入流程，进行算法/应用的集成和测试。

1、算法类接入流程：根据算法所需输入数据情况，在标准数据集中查找，如果数据集满足算法输入标准，直接运行算法，计算结果，如果不符合算法输入标准，则需要算法进行标准数据到算法输入数据的转换，转换后进行算法处理，最后算法结果在转换成标准数据，存储到边缘侧存储环境中。



2、应用类接入流程：应用先对自身进行检查，参照数据规范检查数据输入和输入是否符合要求，参照总线规范检查，应用是否可让总线进行托管，某项不符合要求，即需要进行改造，改造后的应用，可以在注册中心注册应用，纳入应用管理中心管理。



1. 根据前期整理，以下算法和应用集成在边缘侧：

|  |  |  |
| --- | --- | --- |
|  | **算法/应用名** | **简易描述** |
| 后台算法类 | v2x数据通讯 databus | 将内部数据转为标准数据，同时多种数据包打包为一个数据包，转发到RSU和云端，从融合感知程序收数据，处理好后通过mqtt发送到云端和RSU |
| 融合程序fusion | 相机和雷达轨迹数据融合，单车指标计算，输出数据到redis，保存到本地db，通过redis或zmq从comm获取原始感知轨迹数据，计算后通过redis、zmq或kafka发布数据 |
| 基础指标计算程序 | 接收路口单车轨迹数据和信号数据，计算车道、进口道、转向和路口各级多维度指标，输出到redis；从redis或zmq订阅单车轨迹和信号数据，通过zmq或redis发布指标结果 |
| 全息流实时编码 | 全息流数据实时编码，将轨迹流、信号流、指标流、事件流、标签流统一编码；从zmq订阅各种数据流，编码后zmq发布 |
| 视频流媒体程序 | 第三方开源改造程序，实现边缘侧视频流接入并转发，提供流媒体服务；rtsp拉流和推流 |
| 全息流摘要程序 | 全息流编码存储摘要，告诉平台哪些时间段有数据，方便使用数据；zmq订阅全息流编码数据，摘要通过edgedata写入pg db |
| 感知设备通信程序 | 感知设备通信采集程序，采集路口相机、雷达感知数据，包含轨迹和卡口数据，未来计划升级为虚拟设备，统一进行设备接入，转为标准内部协议；通过定制协议和设备通信，tcp/udp方式采集感知设备数据，通过redis、zmq、kafka推送数据，支持本地存储数据副本 |
| spat信号采集程序 | 交通信号采集替代方案，通过旁路采集的方式物理采集信号灯灯色状态，并通过算法转换成spat信号和方案周期信息，可以替代sidbox；定制设备通信协议采集数据，通过redis或zmq推送数据 |
| 全息流媒体程序 | 全息流媒体服务，提供全息流实时播放和录像回放的数据流控制和推送服务；通过zmq推送全息流数据 |
| 全息流历史编码 | 全息流历史数据存储编码和管理服务；通过zqm订阅数据，编码后进行数据存储 |
| 信号机通信程序 | 信号机协议通信，采集信号机配置信息、实时相位信息及状态信息，支持redis消息推送和指令发送，同时本地数据db留存,三类，骏马(ntcip)，海康(20999)，jkc；通过定制协议和信号机进行通信，本地sqlitedb缓存信号配置和实时数据，通过redis或kafka发布数据 |
| 边缘流量修复算法 | 基于感知采集的卡口数据，进行实时流量修复，并结合历史流量数据，实现路口的周流量模型，定时计算，读取db数据，结果落库 |
| 时段划分算法 | 基于流量特征数据计算时段划分结果，定时计算，读取db数据，结果落库 |
| 饱和流率程序 | 基于车道和路口等级，以及卡口数据，计算车道和路口饱和流率，定时计算，读取db数据，结果落库 |
| 英伟达事件接收程序 | ai事件对接服务程序，可接入基于ai识别输出的轨迹、事件和卡口数据，定制协议和第三方ai算法服务通信，通过redis、zmq和kafka发布数据，支持sqlite保留本地数据副本 |
| SIDEBOX通信程序 | sibox数据对接程序，可接入sibox设备的交通信号spat数据，面向淞虹，智能交通，定制设备通信协议采集数据，通过redis或zmq推送数据 |
| 雷视融合算法Fusion | 视觉目标检测跟踪，雷达目标跟踪，传感器融合状态维护；单方向；集成了车牌识别，后端视频编码（rtsp），RTK融合和激光融合；已通过内部总线流转 |
| 雷视融合算法Fusion结构化版 | 基于第三方设备检测跟踪的结构化数据进行融合；单方向；已通过内部总线流转 |
| 多路融合算法 | 多方向Fusion的结果融合到全局视角，已通过内部总线流转 |
| 雷视环境模型算法 | 雷达目标或者雷视目标的长期状态维护，已通过内部总线流转 |
| 交通应用算法 | 基于环境模型结果，根据轨迹计算交通事件或者交通指标，已通过内部总线流转 |
| 雷达转换接口Can口 | 雷达模组的输入转换到内部总线，与模组使用CAN口通信，已通过内部总线流转 |
| 雷达转换接口Eth口 | 雷达模组的输入转换到内部总线，与模组使用网口通信，已通过内部总线流转 |
| 雷视和交通算法输出接口 | 将内部总线上各层面的输出数据根据需求发送到外部总线，已通过内部总线到外部总线的流转 |
| 内部通信数据转化并发送 | 内部总线接收融合轨迹数据，转换数据结构后，发往外部总线 |
| 应用类 | 边缘后台服务程序ets | 统一后台管理程序，实现数据层数据的接入（包括融合轨迹、信号机实时状态和实时相位、AI事件），业务处理，并提供应用层的接口服务及数据转发，业务总后台；通过redis或zmq订阅轨迹、信号、指标、事件等数据，通过api相应请求，通过ws和kafka，mqtt推送业务数据 |
| 单点优化控制程序 | 对饱和流量程序提供的车道饱和流率数据和实时流率数据进行处理，并将处理后的数据发送给单点优化算法程序，生成路口单点优化方案。通过edgedata读取数据库数据，调用optim-algo-server计算单点优化算法，结果落库以及kafka推送中心，会有前后依赖，流式计算 |
| 边缘信令程序 | 全息流信令服务，提供全息流播放控制请求服务通道；提供api调用 |
| 边缘摄像机拉流程序 | 基于边缘摄像机配置控制边缘流媒体服务拉流转发，界面程序，告诉怎么拉流；提供api调用 |
| 单点优化算法程序 | 根据单点优化程序提供的数据计算当前路口的单点优化方案。通过api被optim服务调用 |
| 算法基础服务 | python版本基础指标计算服务，接收路口单车轨迹数据和信号数据，计算车道、进口道、转向和路口各级多维度指标，api被ets调用，返回秒级指标结果 |
| 边缘网关程序 | 提供边缘侧统一应用请求接口，主要面向外部，提供api调用 |
| 边缘PG数据应用管理服务 | 实现边缘侧统一数据库访问服务接口，提供api调用，操作访问pg数据库 |
| 边缘配置和OTA代理服务 | 提供边缘配置中心，以及OTA边缘代理服务，支持通过OTA进行边缘侧应用和配置上报和下发；定制api供中心OTA调用，实现OTA程序镜像和配置拉取，本地文件管理存储镜像和配置文件，同时支持配置服务接口，为各个应用提供配置信息 |

## 2.9、通用的公共服务（由集成管理平台集成管理）

通用公共服务，提供一个统一的对外交互接口或者交互界面，使得接入的算法或者应用能够方便的管理和使用，在算法或应用的整个生命周期，全链路管控；提供一些公共服务：

1. 应用管理中心

提供应用的生命周期管理，包括创建、部署、配置、监控和退役。它支持多租户环境，并提供应用性能监测（APM）工具，以确保应用稳定运行。

1. 用户权限中心

管理用户身份验证和授权，支持多种认证机制（如OAuth, SAML等），并实现细粒度访问控制，确保安全性和合规性。

1. API管理中心

为开发者提供API的定义、发布、版本控制和文档生成。同时支持API流量管理和分析，帮助优化服务调用效率。

1. 数据管理中心

统一管理数据资源，包括数据存储、备份恢复、迁移以及数据分析。它也支持数据治理，确保数据质量、隐私和安全性。

1. 设备管理中心

负责联网设备或其他连接设备的注册、状态监控、远程配置更新和故障诊断。

1. 配置管理中心

集中化管理和分发应用配置参数，支持环境特定设置，允许动态调整而无需重启服务，提高灵活性和响应速度。

1. 消息管理中心

提供可靠的消息队列和事件驱动架构的支持，确保异步通信和系统解耦，支持高并发和分布式系统的高效协作。提供消息相关的统计，提供消息管控的图形化交互方式；

1. 容器管理中心

所有集成的算法或应用，都应基于容器技术（如Docker、Kubernetes）进行应用的打包、部署和服务编排，以简化微服务架构下的开发运维流程。提供图像化的交互界面，简化容器应用的管理。

1. 通信管理中心

处理不同组件或服务间的网络通信需求，包括但不限于协议转换、负载均衡、安全传输层（TLS/SSL）、防火墙规则设置等，确保通信的安全与效率。提供通信的来源目的监控界面，直观的监测通信数据全链路状态。

1. 授时管理中心

保证整个平台时间的同步；

## 2.10、其他公共服务

1. 全息流媒体程序服务

全息流媒体服务，提供全息流实时播放和录像回放的数据流控制和推送服务，通过zmq推送全息流数据；可以为其他集成应用提供全息流实时播放和回放服务；

1. 视频流媒体程序服务

第三方开源改造的应用程序，实现了边缘侧视频流接入并转发，可以为上层应用提供流媒体服务；

## 2.11、非功能性需求：

### 2.11.1. 性能要求

#### 2.11.1.1 通信性能

- 吞吐量：

- 单节点发布速率 > 1GB/s

- 多节点聚合吞吐量 > 10GB/s

- 支持零拷贝传输优化

- 延迟：

- 本地端到端延迟 < 100μs

- 局域网端到端延迟 < 1ms

- 广域网端到端延迟 < 100ms (取决于网络状况)

- QoS支持：

- 可靠性：支持 best-effort 和 reliable 传输

- 持久性：支持 volatile 和 persistent 存储

- 优先级：支持 8 级消息优先级

#### 2.11.1.2 资源占用

- CPU使用率：

- 空闲状态 < 1%

- 正常负载 < 30%

- 峰值负载 < 70%

- 内存使用：

- 基础内存占用 < 100MB

- 运行时内存增长率 < 10MB/s

- 最大内存使用限制可配置

- 存储要求：

- 本地缓存大小可配置

- 持久化存储大小可配置

- 日志存储控制策略可配置

#### 2.11.1.3 扩展性

- 连接规模：

- 单节点支持并发连接 > 1000

- 集群总连接数 > 10000

- 动态扩缩容支持

- 主题数量：

- 单节点主题数 > 10000

- 集群总主题数 > 100000

- 动态主题创建/删除支持

### 2.11.2. 安全要求

需要考虑兼顾三级等保和国密体系的要求

#### 2.11.2.1 认证与授权

- 身份认证：

- 访问控制：

#### 2.11.2.2 数据安全

- 传输加密：

- 数据隔离：

#### 2.11.2.3 安全审计

#### 2.11.2.4 网络安全

### 2.11.3. 可靠性要求

3.1 故障恢复

3.2 监控告警

3.3 高可用配置

### 2.11.4. 性能参数调优

4.1 缓冲区设置

4.2 线程池配置

4.3 批处理设置

### 2.11.5．建议

这些配置参数应根据具体部署环境和业务需求进行调整。

#### 2.11.5.1. 性能测试

- 进行压力测试验证配置

- 监控系统资源使用情况

- 记录性能数据用于优化

#### 2.11.5.2. 安全审计

- 定期进行安全扫描

- 检查访问日志

- 更新安全策略

#### 2.11.5.3. 运维监控

- 设置合理的告警阈值

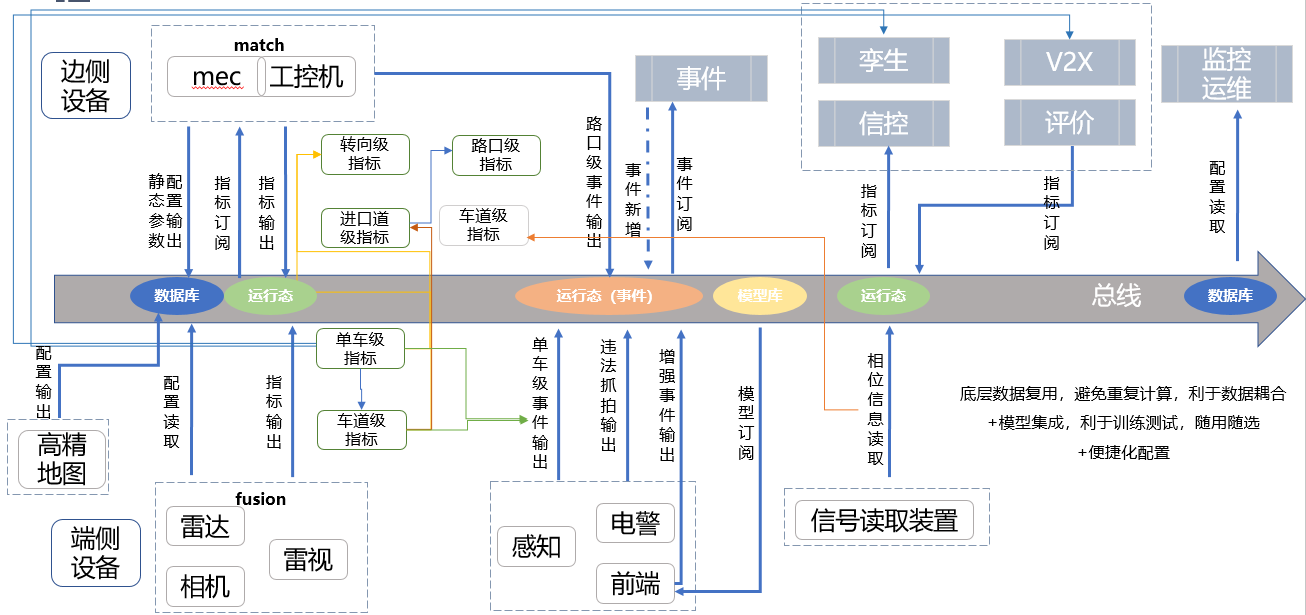
- 建立故障响应机制

- 定期检查系统状态

# 三、标准化的数据规范

总体数据参照团标《车路协同 智能决策道路 第3部分：系统数据接口规范》。路侧感知数据、车辆动态数据参照T/CSAE 159- 2024，路侧下发、路侧配置、路侧上传、路侧状态感知确认、感知上传、感知状态数据参照 CSAE 294.3，视频请求数据参照GB28181（或 W3C WebRTC1.0），设备数据信号灯相关标准可参考GA/T 1049.2，时空融合数据、控制指令参照CSAE 294.2- 2023等。

## 3.1 数据逻辑流图

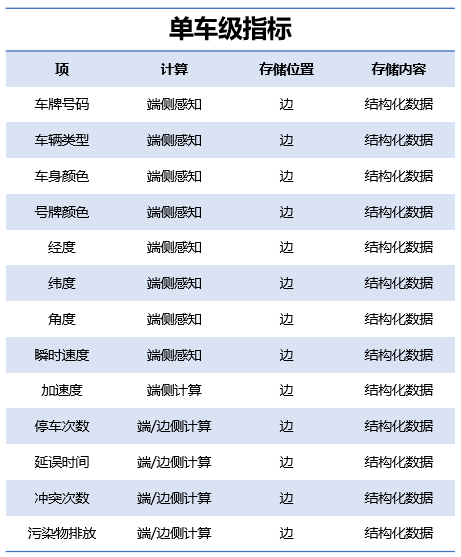


## 3.2 数据指标项







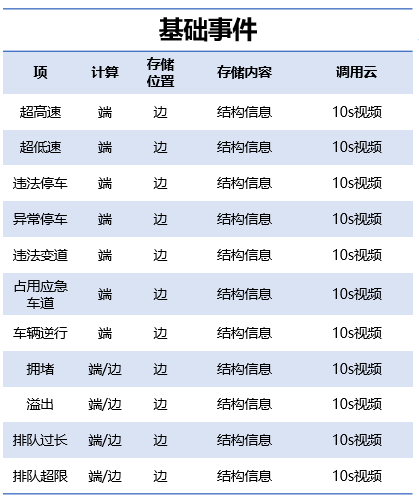


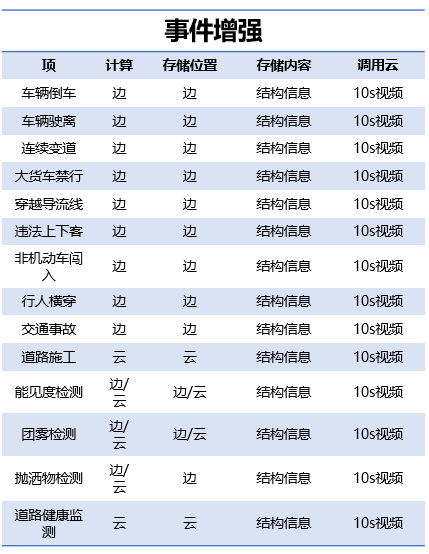


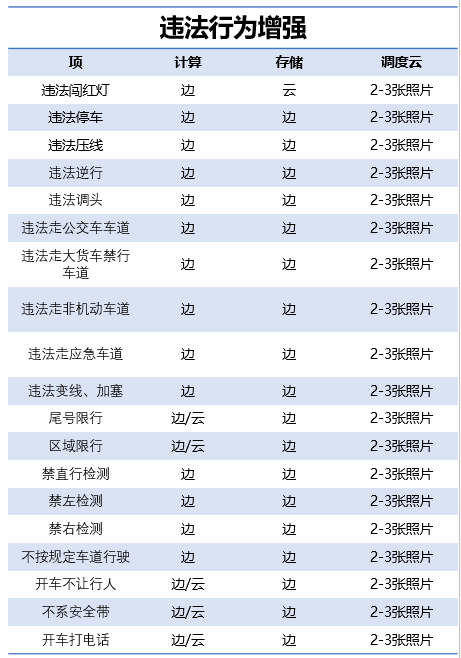






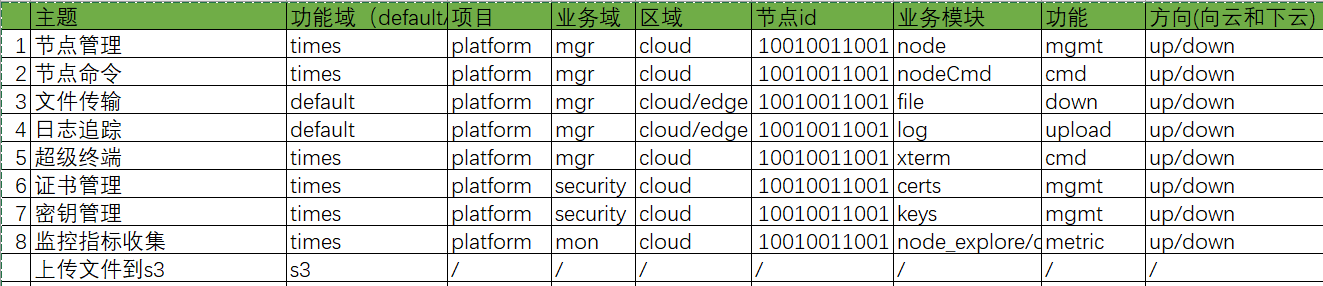




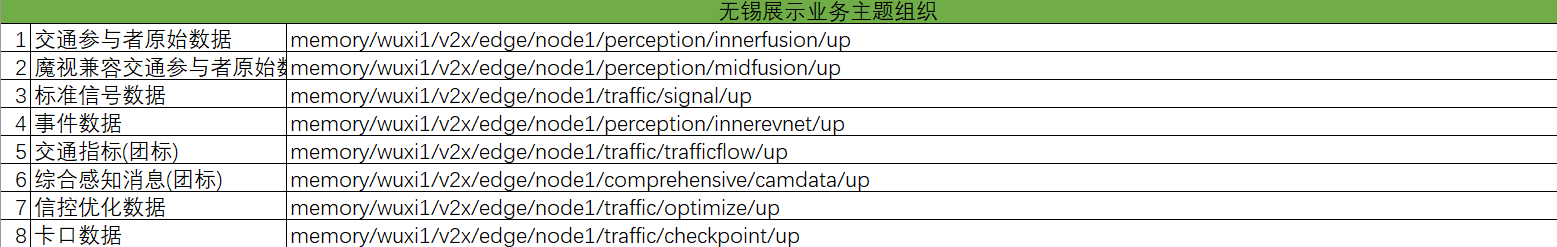


## 3.3 数据主题规则

### 3.3.1 集成平台主题规则设计参考



### 3.3.2 业务主体规则设计参考



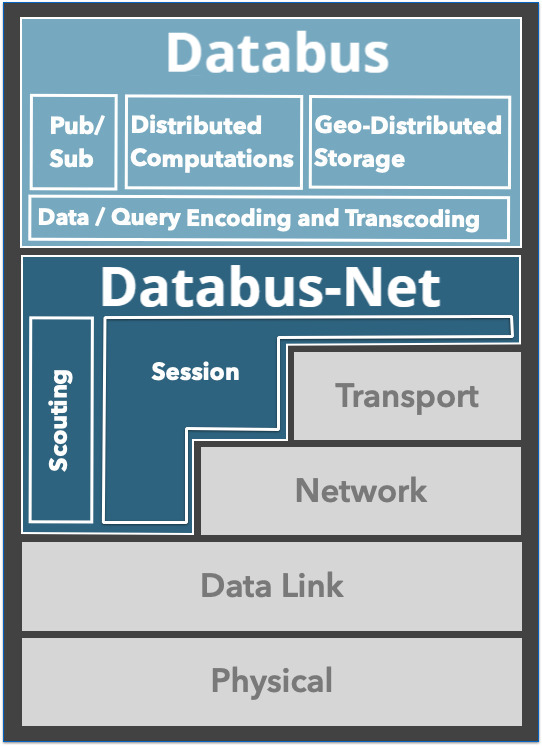
# 四、数据总线和集成中台设计

## 4.1. 数据总线架构概述

Databus是一个创新性的分布式数据传输系统，采用先进的微服务架构和事件驱动设计模式。它的核心是提供一个统一的数据空间抽象层，实现高效的跨平台、跨语言数据交换。系统采用零拷贝技术和发布/订阅模式，显著提升了数据传输效率。在性能方面，通过异步I/O和内存映射等技术，实现了微秒级的传输延迟和GB级的吞吐量。特别适用于对实时性要求高、数据量大的场景，如工业物联网、智能制造、金融交易等领域。同时，系统提供完整的安全机制，包括身份认证、访问控制、数据加密等多层次安全保护，确保数据传输的安全性和可靠性。

Databus 是一个创新的数据分发中间件，专为IoT和边缘计算场景设计。它采用了完全去中心化的架构，通过统一的抽象层来处理异构系统间的数据交换。在性能方面，其零拷贝设计实现了微秒级的端到端延迟，在基准测试中，在千兆网络环境下可实现35μs的往返延迟，单节点吞吐量可达到10Gbps。系统采用Rust语言开发核心组件，确保了内存安全性和高性能。支持包括TCP、UDP和共享内存在内的多种传输协议，并可无缝切换。其独特的位置透明寻址机制允许应用程序无需关心数据的物理位置，显著简化了分布式系统的开发复杂度。为了支持不同的应用场景，系统提供了C、C++、Python、Java、Rust等多语言绑定。

总线提供两级API：



**Databus-Net**

实现能够在数据链路，网络或传输层之上运行的网络层。

Databus-Net支持点对点和路由通信，提供了允许Pub/Sub(Push)通信以及Query/Reply (Pull) 通信的关键原语。它支持分段和有序可靠交付，并为 Discovery(ROS2 Discovery Serivice)提供了可插入的侦查抽象。

Databus-Net定义并建立在会话协议的基础上，该会话协议为有序尽力而为(best effort)和具有不受限制的MTU的可靠信道提供了抽象，而MTU与底层无关。

Databus-Net层仅关心数据传输，不关心数据内容或存储数据。

提供的原语：

* **Write**：将实时数据推送给相对应的订阅者。
* **Subscribe**：订阅实时数据。
* **Query**：从匹配的查询对象中查询数据。
* **Queryable**：能够响应查询的实体。

**Databus**

Databus层为Pub/Sub和分布式查询提供了高级API，以更简单和更面向数据的方式提供与Databus-Net API相同的抽象，并提供所有构建块来创建分布式存储。Databus层知道数据内容，并且可以应用基于内容的过滤和代码转换。

它处理数据表示编码和代码转换，并提供地理分布式存储和分布式计算值的实现。 Databus支持一系列数据编码，例如JSON，Protobuf，Idl，Asn等，以及跨支持格式的转码。 Databus层还提供了存储后端插件API，以简化第三方存储技术的集成。 当前支持的存储后端是Memory，MySQL，MariaDB，PostgreSQL，SQLite和InfluxDB。默认情况下，地理分布式存储在最终一致性下工作。

提供的原语：

* **Put**：将实时数据推送到匹配的订阅者和存储。 （相当于Databus-Net write）
* **Subscribe**：实时数据的订阅者。 （相当于Databus-Net订阅）
* **Get**：从匹配的存储和演算中获取数据。 （相当于Databus-Net查询）
* **Storage**：一个Databus-Net 订阅者侦听要存储的实时数据，以及一个Databus-Net可查询以响应匹配的get请求的组合。
* **eval**：能够响应请求的实体。 通常用于按需提供数据或构建RPC系统。 （相当于Databus-Net可查询）

### 4.1.1 核心设计理念

#### 4.1.1.1 统一数据空间

统一数据空间是Databus的核心抽象概念，它提供了一个全局一致的数据视图和访问机制。系统使用分层的键值对结构组织数据，支持复杂的数据模型和查询操作。键的设计采用层次化的命名空间管理，类似于文件系统的路径结构，便于数据的组织和访问控制。值可以是任意类型的数据，支持基础数据类型、复杂对象、二进制流等。此外，统一数据空间还提供了强大的元数据管理功能，支持数据版本控制、生命周期管理、数据血缘追踪等高级特性，为上层应用提供丰富的数据管理能力。

在Databus中，统一数据空间采用了类似URI的层次化键值结构，例如："/company/department/device1/temperature"。键空间支持通配符匹配（如"/company/\*/temperature"）和表达式过滤。数据值支持任意类型，包括原始数据类型、结构化数据（JSON、Protocol Buffers等）和二进制流。系统实现了一个分布式键值存储引擎，使用内存映射文件技术来优化大数据集的处理性能。权限控制基于细粒度的访问控制列表（ACL），支持按路径和操作类型进行访问控制。在实际部署中，该设计在处理每秒数百万个数据点的场景下表现出色，存储引擎的写入延迟通常保持在微秒级别。

#### 4.1.1.2 去中心化架构

去中心化架构是Databus的另一个核心特性，采用分布式设计确保系统的高可用性和可扩展性。节点间采用P2P（点对点）通信模型，每个节点既可以作为数据提供者，也可以作为数据消费者。系统使用基于DHT（分布式哈希表）的路由算法，确保数据能够快速准确地到达目标节点。动态发现机制支持节点的自动发现和健康检查，提供故障自愈能力。负载均衡机制确保系统资源的合理利用，防止单点过载。数据分片和复制策略提供了数据的高可用性和容灾能力。

Databus采用基于Chord算法的改进版DHT（分布式哈希表）实现路由寻址，每个节点维护一个最多包含O(log N)个路由表项的路由表。节点发现使用组播和基于gossip协议的混合机制，支持跨网段自动发现。在广域网部署中，系统使用动态隧道技术处理NAT穿透问题。路由决策考虑网络延迟、带宽和节点负载等多个因素，通过动态权重算法选择最优路径。实测表明，在包含1000个节点的网络中，节点发现和路由表收敛时间通常在3秒内完成。系统还实现了分布式一致性协议，确保在网络分区情况下的数据一致性。

#### 4.1.1.3 性能优化

性能优化贯穿Databus的整个设计过程，从底层通信到上层应用都进行了全方位的优化。采用零拷贝技术减少数据传输过程中的内存拷贝次数，显著提升传输效率。使用内存池和对象池技术减少内存分配和回收的开销。实现了智能的批处理机制，可以自适应地合并小数据包，优化网络传输效率。采用多级缓存策略，包括进程内缓存、分布式缓存等，提高热点数据的访问速度。异步处理机制确保高并发场景下的系统吞吐量。

零拷贝实现基于操作系统的sendfile/splice机制和用户态协议栈（如DPDK），在数据传输过程中完全避免了用户空间和内核空间的数据拷贝。异步通信框架采用基于tokio的事件驱动模型，支持高并发连接处理。数据批处理机制会智能合并小数据包，当累积数据大小达到网络MTU大小或等待时间超过配置阈值（默认100微秒）时触发发送。QoS策略支持可靠传输和尽力而为两种模式，可靠传输模式使用NACK机制进行丢包恢复，同时实现了基于接收方窗口的流量控制。在实际测试中，在千兆网络环境下，单连接可实现接近线速（约900Mbps）的持续传输性能，同时CPU使用率保持在较低水平（通常小于10%）。

### 4.1.3. 核心组件

#### 4.1.3.1 Router（路由器）

Router是Databus的核心组件，负责数据的智能路由和转发。它维护完整的网络拓扑信息，实时更新节点状态和连接关系。路由算法支持多种策略，包括最短路径、最低延迟、带宽优先等，可根据实际需求动态选择。支持跨网络段的数据传输，自动处理网络地址转换和协议转换。实现了高效的订阅关系管理，使用布隆过滤器等技术优化订阅匹配性能。提供流量控制和拥塞管理机制，确保网络的稳定运行。

#### 4.1.3.2 Peer（对等节点）

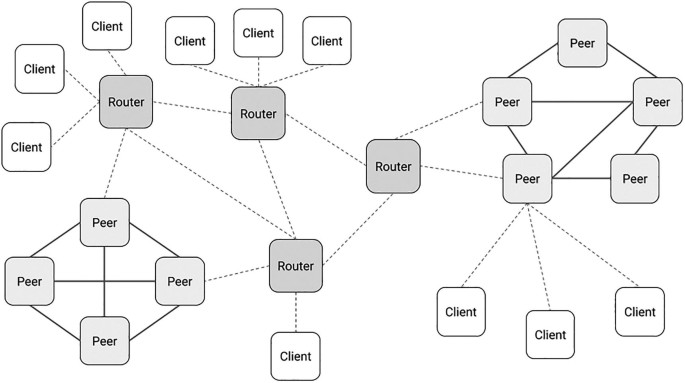
Peer作为系统的基础节点，提供核心的数据交换和存储功能。每个Peer都维护本地缓存，支持数据的快速访问和临时存储。实现了分布式查询引擎，支持复杂的数据检索操作。提供数据压缩和解压缩功能，优化存储空间和传输效率。支持数据的版本控制和冲突解决，确保数据一致性。实现了本地的事件处理引擎，支持数据的实时处理和转换。提供完整的监控接口，便于系统运维和问题诊断。

#### 4.1.3.3 Client（客户端）

Client提供简洁易用的API接口，支持多种编程语言和开发环境。实现了智能的连接池管理，优化网络连接资源的使用。支持数据的异步操作和批量处理，提供灵活的错误处理和重试机制。实现了本地的数据缓存和预取功能，优化应用性能。提供丰富的配置选项，支持客户端的个性化定制。包含完整的诊断工具，便于问题排查和性能优化。支持插件机制，方便功能扩展。

#### 4.1.3.4三种部署模式

* 对等方（peers）。用于在 对等网络（peer-to-peer）、网状（mesh）拓扑网络和其他节点通信，也可以通过路由器与广泛的系统通信。
* 客户（clients）。用于连接单个路由器（或点）来和系统其他部分通信。
* 路由器（routers）。用于在 client和peers之间、集团拓扑和网状拓扑网络之间路由数据。



### 4.1.4. 关键特性

#### 4.1.4.1 通信模式

Databus支持多种灵活的通信模式，适应不同的应用场景需求。发布/订阅模式支持异步的消息通信，实现了消息的定时发送、延迟发送、条件触发等高级特性。查询/响应模式提供同步的数据访问能力，支持复杂的查询条件和结果过滤。推/拉模式结合了主动推送和按需获取的优点，优化数据传输效率。点对点通信模式支持节点间的直接数据交换，减少网络延迟。所有通信模式都支持QoS策略，确保服务质量。

#### 4.1.4.2 数据管理

数据管理功能覆盖数据的全生命周期，提供完整的数据治理能力。实时数据流处理支持流式计算和实时分析。历史数据存储实现了数据的持久化和归档。数据索引机制支持高效的检索和查询。数据压缩和解压缩优化存储空间和传输效率。数据加密确保安全性，支持细粒度的访问控制。数据备份和恢复机制保证数据的可靠性。提供完整的数据监控和审计功能。

#### 4.1.4.3 网络优化

网络优化是系统性能的关键因素，Databus在这方面进行了深入的优化。智能路由选择基于网络状况动态调整传输路径。负载均衡确保网络资源的合理利用。链路聚合提高带宽利用率。网络分片减少广播风暴。QoS策略保证关键业务的服务质量。网络监控实时掌握系统状态。自适应流控防止网络拥塞。错误重传确保数据可靠传输。

### 4.1.4. 技术优势

#### 4.1.4.1 高性能

高性能是Databus的核心竞争力之一。通过零拷贝技术显著减少CPU和内存开销。异步处理机制提高系统并发能力。内存池技术优化资源利用。多级缓存加速数据访问。智能批处理提升传输效率。硬件加速支持提高处理速度。负载均衡确保资源充分利用。性能监控和调优工具助力系统优化。

#### 4.1.4.2 可靠性

可靠性设计贯穿系统各个层面，确保数据传输和存储的可靠性。故障检测和自动恢复机制处理各类异常情况。数据一致性协议保证分布式环境下的数据准确性。多副本策略提供数据冗余保护。事务机制确保操作的原子性。日志机制便于问题追踪和恢复。监控告警及时发现并处理问题。灾难恢复方案保障业务连续性。

#### 4.1.4.3 扩展性

系统的扩展性体现在多个维度。水平扩展支持动态增加节点，提升系统容量。垂直扩展通过升级硬件提高单节点性能。功能扩展通过插件机制增加新特性。协议扩展支持新的通信协议。存储扩展支持多种存储引擎。安全扩展支持新的安全机制。监控扩展支持自定义监控指标。接口扩展支持新的编程语言和框架。

#### 4.1.4.4 支持协议集

协议类型

a. 传输协议

- tcp：基础传输控制协议

- udp：用户数据报协议，适用于快速传输场景

- tls：传输层安全协议

- 共享内存：本地进程间通信协议

b. 服务协议

- DDS：数据分发服务协议，支持实时数据通信

- REST：表述性状态传输协议，基于HTTP的API设计规范

- zenoh：分布式消息队列协议，支持发布订阅模式

- mqtt/mqtts：轻量级物联网消息(及加密)传输协议

- quic：快速UDP互联网连接协议

c. 扩展协议

- WS：WebSocket协议，支持全双工通信

- WSS：安全WebSocket协议，基于TLS加密

#### 4.1.4.4. 支持编解码集

a. 数据格式

- Protobuf：Google的开源序列化框架

- JSON：轻量级数据交换格式

- ASN：抽象语法标记

- FastCDR：快速序列化协议

- CANP：CAN协议编解码格式

b. 特性说明

1. Protobuf特性：

- 二进制格式，高效压缩

- 跨语言支持

- 向前向后兼容

- 自动代码生成

2. JSON特性：

- 文本格式，可读性强

- 语言无关性

- 灵活的数据结构

- 广泛的工具支持

3. ASN特性：

- 标准化的编码规则

- 支持复杂数据类型

- 严格的类型定义

- 适用于通信协议

4. FastCDR特性：

- 高性能序列化

- 低延迟处理

- 内存效率高

- 适合实时系统

4. CANP特性：

- 专用于CAN总线

- 紧凑的数据格式

- 实时性能好

- 支持标准帧和扩展帧

c 使用场景

1. 实时数据传输：FastCDR、CANP

2. Web应用交互：JSON、WebSocket

3. 跨平台服务：Protobuf、REST

4. 工业现场总线：CANP、DDS

4. 物联网通信：MQTT、UDP

5. 安全通信：TLS、WSS

d 性能考虑

1. 编码效率：

- 二进制协议：Protobuf、FastCDR

- 文本协议：JSON、REST

2. 传输效率：

- 高效传输：UDP、QUIC

- 可靠传输：TCP、TLS

3. 资源占用：

- 轻量级：MQTT、UDP

- 重量级：DDS、WebSocket

4. 实时性能：

- 高实时：CANP、FastCDR

- 普通实时：其他协议

这些协议和编解码格式的选择需要根据具体应用场景、性能需求和系统架构来确定，通常会组合使用多种协议来满足不同层面的需求。

### 4.1.5. 应用场景

#### 4.1.5.1 边缘计算

在边缘计算场景中，Databus发挥着关键作用。实现设备数据的高效采集和预处理，支持本地的实时分析和决策。边缘节点间的协同计算提高处理效率。数据过滤和压缩优化传输效率。本地缓存减少云端依赖。安全机制保护边缘数据。智能路由优化网络资源。故障自愈确保服务可用性。监控管理实现边缘设备的集中管理。

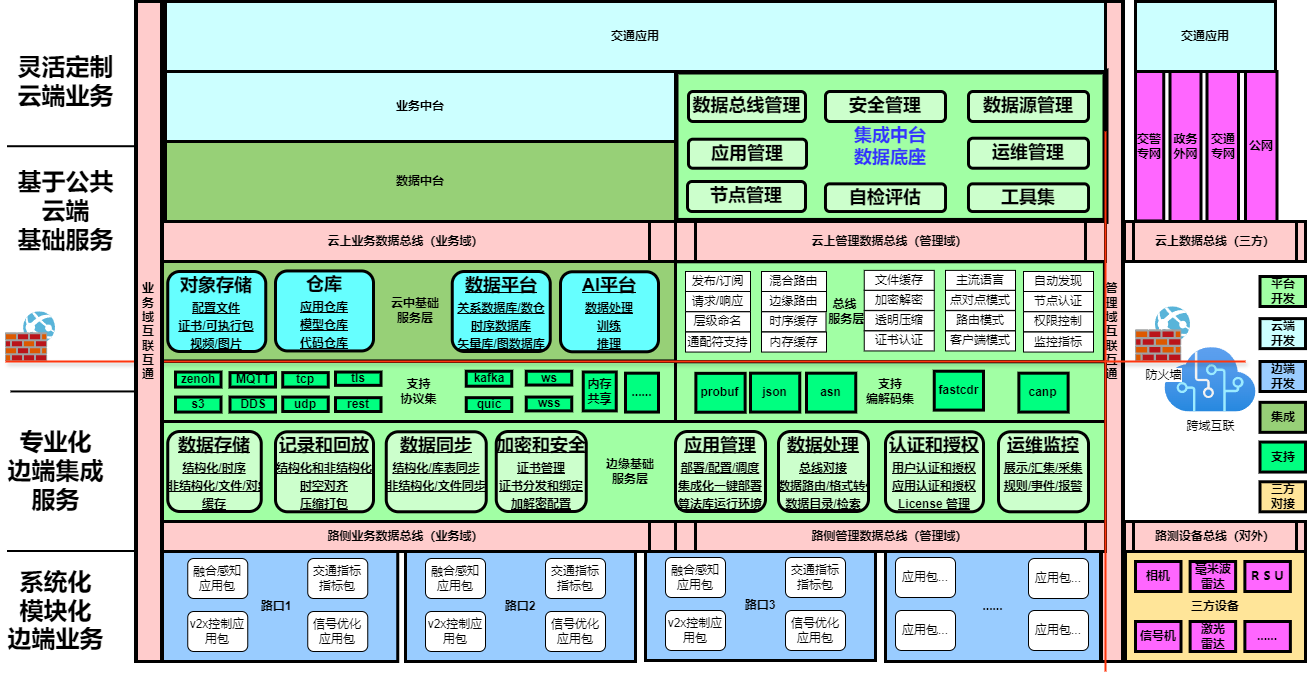
#### 4.1.5.2 物联网系统

在物联网领域，Databus提供了完整的数据采集和处理解决方案。支持海量设备的接入和管理，实现数据的实时采集和处理。设备状态监控确保系统稳定运行。控制指令可靠下发保证设备可控。数据分析提供决策支持。协议适配支持多种物联网协议。边缘计算减少云端压力。安全机制保护物联网安全。运维工具简化系统管理。

#### 4.1.5.3 分布式系统

在分布式系统中，Databus作为关键的数据基础设施，支持复杂的业务场景。微服务通信提供可靠的服务间数据交换。数据同步确保分布式环境下的数据一致性。事件驱动架构支持松耦合的系统集成。分布式缓存提高数据访问效率。事务处理确保操作的原子性。服务发现简化系统配置。监控告警保障系统可用性。运维工具辅助系统管理。

## 4.2 集成平台概要设计



集成中台架构示意图

### 4.2.1. 节点管理模块

节点管理模块作为集成中台的基础设施层，承担着分布式系统中所有节点的生命周期管理责任。该模块通过建立统一的管理接口和标准化的通信协议，实现了节点的自动发现、注册、配置管理、状态监控和资源调度等核心功能。在架构设计上采用分布式设计理念，支持节点的动态扩展和弹性伸缩。通过引入心跳机制和健康检查，实时监控节点状态，并具备故障自动转移能力。模块还实现了跨地域节点管理，通过建立安全通道确保节点间通信的可靠性和数据传输的安全性。

#### 4.2.1.1 节点基础管理

节点基础管理子模块提供节点生命周期的完整管理能力，包括节点注册、配置、启停等基础操作。通过标准化的节点注册流程，实现新节点的自动发现和接入。配置管理采用中心化设计，支持配置的统一管理和动态下发。节点状态管理实现了实时监控和故障检测，通过心跳机制和健康检查确保节点的稳定运行。资源管理功能对节点的计算、存储、网络等资源进行统一管理，支持资源的动态分配和回收。子模块还提供了节点级别的性能优化和故障诊断能力。

#### 4.2.1.2 节点文件管理

节点文件管理子模块负责处理节点间的文件传输、存储和同步需求。该子模块实现了高效的文件传输机制，支持大文件传输和断点续传。文件同步功能确保配置文件、日志文件等关键数据在节点间保持一致。存储管理采用分布式设计，支持多种存储介质和存储策略。文件版本控制功能实现了文件的版本管理和历史追溯。子模块还提供了文件压缩、加密等增值功能，确保文件传输和存储的效率与安全性。

#### 4.2.1.3 节点监控调试

节点监控调试子模块提供全方位的监控和调试能力。通过采集系统层面的CPU、内存、磁盘IO等基础指标，实现对节点性能的实时监控。进程监控功能关注节点上运行的关键进程状态，支持进程级别的操作和调试。网络监控实时采集网络连接状态和性能指标。调试功能支持远程调试和日志实时查看，方便运维人员进行问题诊断和处理。子模块还集成了性能分析工具，支持系统性能瓶颈分析和优化建议。

### 4.2.2. 应用管理模块

应用管理模块是实现应用全生命周期管理的核心组件，提供了从应用开发、测试到部署、运维的完整解决方案。该模块通过标准化的应用封装和管理流程，支持多种类型应用的统一管理，包括传统应用、微服务应用和容器化应用。采用中心化的配置管理和分布式的部署架构，确保应用配置的一致性和部署的灵活性。模块集成了完整的监控体系，实时采集应用运行状态和性能指标，支持异常检测和告警。

#### 4.2.2.1 应用配置管理

应用配置管理子模块负责应用的参数配置和环境管理。采用配置中心化设计，实现配置的统一管理、版本控制和动态下发。支持多环境配置管理，方便开发、测试、生产环境的配置隔离。配置变更管理实现了变更审核、灰度发布和回滚机制。配置加密功能确保敏感配置的安全性。子模块还提供配置依赖分析和影响评估能力，降低配置变更风险。

#### 4.2.2.2 应用部署监控

应用部署监控子模块提供应用部署的全流程管理和监控能力。支持应用包的版本管理和部署策略配置，实现自动化部署和回滚。部署过程监控实时展示部署进度和状态，支持部署过程的暂停、继续和终止操作。部署验证功能自动检查部署结果，确保应用正常运行。子模块还包含部署历史记录和对比分析功能，方便追踪部署变更。

#### 4.2.2.3 应用运行监控

应用运行监控子模块构建了全方位的应用监控体系。通过轻量级的监控探针，实时采集应用的运行状态、性能指标和业务指标。性能监控覆盖CPU使用率、内存占用、响应时间等关键指标。异常监控支持自定义异常规则，实现异常的实时告警。业务监控通过埋点机制采集业务指标，支持业务趋势分析。子模块还提供了丰富的可视化展示和报表功能，方便运维人员掌握应用运行状况。

### 4.2.3. 数据总线管理模块

数据总线管理模块是整个平台的核心数据流转中枢，负责实现数据的统一接入、传输、转换和分发。采用分布式消息队列架构，支持高并发数据处理和实时数据流转。模块实现了数据格式标准化和协议适配，确保异构系统间的数据互通。通过主题管理和数据域划分，实现了数据的逻辑隔离和精细化管理。模块还集成了数据质量控制和监控功能，保证数据传输的准确性和可靠性。

#### 4.2.3.1 数据域管理

数据域管理子模块实现了数据的逻辑分区和管理。通过定义数据域，将数据按业务属性、安全级别等维度进行分类管理。数据域间的访问控制确保数据隔离和安全共享。支持数据域的动态创建、配置和扩展，满足业务发展需求。数据域监控功能实时展示数据流转状况和资源使用情况。子模块还提供了数据域间的数据映射和转换能力，支持跨域数据交换。

#### 4.2.3.2 数据集成管理

数据集成管理子模块提供了统一的数据接入和处理框架。支持多种数据源的接入适配，包括数据库、消息队列、文件系统等。实现了数据清洗、转换、加工的标准化处理流程。数据质量控制贯穿整个集成过程，通过规则引擎实现数据校验和处理。提供了数据集成任务的调度管理和监控功能。子模块还支持实时数据同步和批量数据处理，满足不同场景的数据集成需求。

#### 4.2.3.3 协议及序列化管理

协议及序列化管理子模块负责数据传输协议的适配和转换。支持主流的数据交换协议，如HTTP、WebSocket、MQTT等。实现了多种序列化格式的转换，包括JSON、XML、Protobuf等。协议适配层提供统一的接口规范，简化协议转换开发。性能优化机制确保协议转换的高效性。子模块还包含协议安全性验证和性能监控功能，保证数据传输的安全和效率。

### 4.2.4. 数据源管理模块

数据源管理模块是实现数据接入和处理的基础组件，提供了统一的数据源接入框架和标准化的数据处理流程。模块支持多种类型数据源的管理，包括关系型数据库、非关系型数据库、文件系统等。通过统一的接口规范和数据模型，实现了异构数据源的统一管理。模块还提供了数据质量控制、数据安全防护和数据生命周期管理等核心功能，确保数据的可靠性和安全性。

#### 4.2.4.1 结构化数据管理

结构化数据管理子模块专注于关系型数据的处理和管理。实现了主流关系型数据库的接入适配，支持Oracle、MySQL、PostgreSQL等数据源的统一管理。提供数据模型管理功能，实现表结构的自动发现和同步。数据同步功能支持实时同步和批量同步两种模式，并提供断点续传能力。查询优化功能通过智能分析提供最优查询方案。子模块还包含数据版本管理和变更追踪功能，确保数据的一致性和可追溯性。

#### 4.2.4.2 非结构化数据管理

非结构化数据管理子模块处理文档、图片、视频等多媒体数据。实现了文件存储系统的统一管理，支持本地存储和云存储的混合部署。文件处理功能支持格式转换、压缩、提取等操作。元数据管理功能为非结构化数据提供结构化描述和检索能力。存储优化机制实现了数据的分级存储和冷热分离。子模块还提供了文件分发和同步功能，确保大规模文件的高效传输。

### 4.2.4. 安全管理模块

安全管理模块构建了全面的安全防护体系，从身份认证、访问控制、数据加密等多个维度确保平台的安全性。采用PKI体系实现身份认证和传输加密，RBAC模型实现细粒度的权限控制。模块支持多种安全策略的配置和管理，包括密码策略、访问策略、审计策略等。通过安全审计和监控，实现了安全事件的实时检测和响应。

#### 4.2.4.1 身份认证与授权

身份认证与授权子模块实现统一的认证和授权管理。支持多种认证方式，包括用户名密码、证书、令牌等。集成了单点登录(SSO)功能，提供统一的用户认证入口。授权管理基于RBAC模型，支持角色的继承和组合。权限管理支持功能权限和数据权限的精细化控制。子模块还提供了认证授权过程的审计日志，记录关键操作痕迹。

#### 4.2.4.2 数据安全防护

数据安全防护子模块负责数据全生命周期的安全防护。实现了传输加密、存储加密和字段级加密等多层次加密机制。数据脱敏功能支持静态脱敏和动态脱敏，保护敏感信息。数据防泄漏系统(DLP)实现了敏感数据的识别和防护。访问控制功能确保数据访问的合规性。子模块还包含数据备份和恢复机制，保障数据安全。

### 4.2.5. 运维管理模块

运维管理模块提供全面的运维能力，确保平台的稳定运行。模块实现了多维度的监控体系，覆盖系统资源、应用性能、业务指标等方面。告警管理系统支持多级别告警策略配置和智能告警分析。通过自动化运维工具，简化日常运维操作。资源管理功能实现了IT资源的可视化管理和优化配置。

运维管理模块同时支持采集各种应用的业务指标，这些要求应用开发和暴露其业业主指标，通过运维采集软件统一采集和监控。

#### 4.2.5.1 系统监控

系统监控子模块建立了全面的监控体系。实时采集系统层面的性能指标，包括CPU、内存、磁盘、网络等。应用监控功能关注应用的运行状态和性能表现。业务监控通过自定义指标实现业务层面的监控。告警系统支持多渠道告警通知和告警升级。子模块还提供了丰富的可视化展示，直观展现系统运行状况。

### 4.2.5.2 运维自动化

运维自动化子模块提供了一系列自动化工具和脚本。作业调度系统支持复杂运维任务的自动化执行。变更管理功能实现了变更流程的规范化和自动化。配置管理实现了配置的自动化下发和验证。自动化测试工具支持系统功能和性能的自动化测试。子模块还包含问题自动诊断和修复能力，提高运维效率。

### 4.2.7. 工具集模块

工具集模块提供了全面的开发和运维工具支持。包括开发工具、测试工具、部署工具和运维工具等多个类别。工具采用插件式架构，支持功能的灵活扩展。提供统一的工具管理界面，简化工具的使用和维护。通过标准化的接口规范，确保工具间的互操作性。模块还包含完整的使用文档和示例，方便用户快速掌握工具的使用方法。

以上设计方案通过模块化的架构设计，实现了系统各个功能模块的解耦和独立演进，同时通过统一的数据总线确保了模块间的有效协同。每个模块都具备完整的功能体系和清晰的职责边界，为整个平台的可扩展性和可维护性提供了保障。

### 4.2.8集成中台边缘侧功能概要设计



#### 4.2.8.1. 设计总览

基于图中八大核心功能模块，边缘侧设计需要重点考虑数据采集、本地存储、安全防护和轻量级处理能力。设计将遵循"边缘智能、本地闭环、按需上云"的原则。

这些基础能力基本以数据总线的形式向边缘和云端提供服务。

#### 4.2.8.2. 功能模块设计

##### 4.2.8.2.1. 数据存储

结构化数据

功能职责：

- 支持总线上数据缓存

- 支持存到内存，文件和时序数据库

关键特性：

- 透明特性

- 支持缓存和查询

- 自带加密解密

非结构化文件存储

功能职责：

- 处理文档、图片等大文件存储

- 支持对象存储和本地文件存储

关键特性：

- 可以通过通过绑定topic直接上传文件到对象存储

- 可以通过动态监控目录的方式，实现文件同步到对象存储

缓存

功能职责：

- 提供高速数据缓存服务（轻量级redis）

##### 4.2.8.2.2. 记录和回放

结构化和非结构化数据存储

功能职责：

- 统一记录多类型数据

- 数据通过总线后进行统一的时空对齐，并整体保存

- 生成文件索引和关键点信息

- 支持文件播放，点播，快进

关键特性：

- 支持多种数据格式的统一存储

- 实现数据实时记录和批量导入

- 提供数据标签化管理能力

- 支持数据索引自动建立

- 实现数据关联分析功能

历史回放

功能职责：

- 提供历史数据查询

- 支持时间序列回放

- 管理回放速度控制

- 实现回放场景还原

关键特性：

- 支持多维度时间轴回放

- 实现数据回放速率动态调节

- 提供回放断点续播功能

- 支持多线程并发回放

- 内置回放性能优化机制

- 支持孪生平台和总线上统一回放

压缩打包

功能职责：

- 为总线上数据提供透明的压缩和解压缩

关键特性：

- 支持配置方式打开和关闭

##### 4.2.8.2.3. 数据同步

结构化库表同步

功能职责：

- 管理同步数据源

- 管理数据库表级同步

- 控制同步任务调度

- 监控同步状态管理

关键特性：

- 支持全量和增量同步策略

- 实现表结构映射

- 提供事务一致性保证机制

- 内置同步冲突检测和解决

- 支持自定义同步规则配置

非结构化文件同步

功能职责：

- 主要为本地文件同步到对象存储

- 处理大文件同步传输

- 实现目录结构同步

关键特性：

- 支持断点续传和并行传输

- 提供文件完整性校验机制

- 支持带宽限流和优先级控制

- 内置文件同步状态监控

##### 4.2.8.2.4. 加密和安全

证书管理

功能职责：

- 管理节点数字证书生命周期

- 处理证书申请和更新

- 控制证书存储安全

- 实现证书验证服务

- 实现证书和总线节点的绑定解绑，从而保证通讯安全

- 实现基于证书的ACL控制

关键特性：

- 实现证书自动续期机制

- 提供证书吊销列表管理

- 支持证书密钥保护

- 内置证书有效性监控

证书分发和授权

功能职责：

- 处理证书分发流程

- 管理授权策略控制

- 实现证书部署自动化

- 监控证书使用状态

关键特性：

- 支持批量证书分发机制

- 实现基于角色的授权控制

- 提供证书使用审计功能

- 支持证书分发加密传输

- 内置分发状态追踪能力

加密配置

功能职责：

- 支持基于证书解密解密（x509）

- 控制密钥生命周期

##### 4.2.8.2.4. 应用管理

边缘基础服务层

功能职责：

- 提供基础运行环境

- 管理应用版本，支持应用的配置，部署，运行监控

- 控制应用生命周期

- 实现应用监控告警

关键特性：

- 支持容器化部署管理

- 提供应用健康检查能力

- 支持应用资源管理，扩缩容（容器和容器云应用）

配置管理调度

功能职责：

- 管理应用配置信息

- 处理配置动态更新

- 控制配置版本管理

- 实现配置同步分发

关键特性：

- 支持配置热更新机制（需应用支持）

- 实现配置版本回滚能力

- 提供配置加密存储选项

- 关键配置自检和自测

版本化管理部署

功能职责：

- 管理应用版本发布

- 控制部署流程自动化

- 处理版本回滚操作

- 实现多环境部署管理

关键特性：

- 支持蓝绿部署和灰度发布

- 实现自动化部署流水线

- 提供部署状态实时监控

##### 4.2.8.2.5. 数据处理

功能职责：

- 提供数据的协议转换

- 提供数据的编解码转化

- 对数据质量进行评估

- 处理任务依赖关系

关键特性：

- 提供动态配置，数据在总线上动态转换

- 提供资源限流控制

- 支持多任务并行处理

- 内置任务监控和告警

##### 4.2.8.2.7. 认证和授权

用户认证和授权

功能职责：

- 管理用户身份和数字认证之间绑定关系

- 控制访问权限分配

应用认证和授权

功能职责：

- 管理节点身份和数字认证之间绑定关系

- 管理应用接入认证

- 控制API访问授权

- 处理应用间通信安全

- 实现总线上数据访问鉴别授权

关键特性：

- 支持x509证书认证

- 实现基于数据总线的网关集成

- 提供服务间认证机制

- 支持权限动态调整

- 内置接口调用监控

License管理

功能职责：

- 管理应用软件许可证（应用数管理）

- 管理总线接入许可证（接入数管理）

- 处理授权期限管理

- 实现授权验证服务

关键特性：

- 支持离线/在线激活

- 实现授权自动续期

- 提供授权使用统计

- 支持批量授权管理

- 内置授权合规检查

##### 4.2.8.2.8. 运维监控

展示汇集统计

功能职责：

- 管理监控数据展示

- 处理统计分析报表

- 控制数据可视化

- 实现多维度分析

关键特性：

- 支持实时数据展示

- 实现多样化图表展示

- 提供自定义报表能力

- 支持数据钻取分析

- 内置报表导出功能

监控事件展示

功能职责：

- 管理系统事件采集

- 处理告警事件展示

- 控制事件分级分类

- 实现事件追踪分析

关键特性：

- 支持事件实时推送

- 实现告警级别管理

- 提供事件关联分析

- 支持事件处理流程

- 内置事件统计报表

运维工具管理

功能职责：

- 管理运维工具集成，支持环境检测和节点，应用，总线自检

- 控制工具权限分配

- 处理工具调用日志

- 实现工具版本管理

关键特性：

- 支持工具在线升级

- 实现工具参数配置

- 提供工具使用审计

- 支持自定义工具接入

- 内置工具效能分析

# 五、边缘应用平台规范和约束

平台应该具备可拆卸能力，从而应对各种项目环境。

遵循整体规划，逐步实施，尽量轻量化的原则。

平台和应用支持整体化部署（预组装好）和批量部署，支持在线部署模式和离线部署模式。

## 5.1、数据总线规范

所有数据传输、通信和共享必须通过数据总线进行

数据总线应划分不同数据域，用于：

### 5.1.1 应用区域划分

数据隔离

数据安全保障

通信资源调度

### 5.1.2 外部通信统一由总线侧处理：

（南向）设备接入

（北向）云端通信

### 5.1.3 应用必须遵循以下职责：

负责订阅总线上所需数据并处理

负责向总线发布数据

## 5.2、应用部署规范

### 5.2.1支持的应用类型：

原生应用

容器应用

预留容器云应用扩展能力

### 5.2.2 硬件支持：

x86架构

ARM架构

NVIDIA GPU(CUDA框架)

### 5.2.3 应用部署要求：

支持一键部署

支持批量部署

确保应用可正常启动和运行

### 5.3、监控运维规范

### 5.3.1系统维度监控：

CPU使用率

内存使用情况

磁盘使用情况

网络状态

进程状态

### 5.3.2 应用进程维度监控：

进程CPU使用率

进程内存使用情况

进程网络使用情况

### 5.3.3 业务指标监控：

遵循Prometheus规范

提供Grafana配置模板

## 5.4、自检报告机制

### 5.4.1应用级自检(按需)

普通级：非侵入式监控(进程、内存、网络通信)

专业级：核心功能自检(需实现专用接口)

### 5.4.2 节点级自检(按需)

#### 5.4.2.2硬件资源自检：

CPU运行状态与负载检测

内存使用率与可用空间检测

磁盘使用率、I/O性能与健康状态检测

GPU状态检测（如适用）

网络接口状态与性能检测

#### 5.4.2.3系统服务自检：

操作系统核心服务状态检测

系统日志检查

系统时间同步状态

关键系统进程状态

系统安全状态（防火墙、端口、访问控制等）

#### 5.4.2.3基础设施自检（按需）：

Docker服务状态（如适用）

数据总线服务状态

数据库服务状态

存储服务状态

网络服务状态（DNS、网关等）

节点连接性自检：

与其他节点的连接状态

与云端的连接状态

网络延迟和丢包率检测

关键服务的端口可访问性

#### 5.4.2.4节点数据自检（按需）：

配置文件完整性检查

数据存储空间检查

数据备份状态检查

临时文件清理状态

### 5.4.3 平台级自检(按需)

#### 5.4.3.1 集群健康状态：

节点在线状态监测

节点负载均衡状态

集群资源使用统计

集群网络拓扑状态

集群服务可用性统计

#### 5.4.3.2应用管理状态：

应用部署成功率统计

应用运行状态汇总

应用资源使用统计

应用更新/回滚状态

容器运行状态汇总

#### 5.4.3.3数据总线状态：

消息吞吐量监测

消息延迟统计

数据域隔离状态

协议适配器状态

数据流转效率分析

#### 5.4.3.4存储系统状态：

分布式存储健康状态

数据同步状态

存储容量使用趋势

数据备份完整性

存储性能指标

#### 5.4.3.5安全状态：

平台访问控制有效性

数据加密状态

安全策略执行状态

异常访问检测

安全日志分析

#### 5.4.3.5运维管理状态：

监控系统可用性

告警系统有效性

日志收集完整性

运维操作审计

自动化运维执行状态

业务指标状态：

关键业务指标汇总

业务处理效率统计

业务异常统计

业务流程完整性检查

SLA达成状况

#### 5.4.3.6系统优化建议：

资源使用优化建议

性能提升建议

安全加固建议

运维效率提升建议

成本优化建议

#### 5.4.3.7报告输出：

定期自检报告生成

实时状态数据展示

历史趋势分析

问题跟踪记录

改进效果评估

#### 5.4.4自检机制要求：

具备自动执行能力

提供详细的检测日志

支持自定义检测项

具备问题自动修复能力

提供清晰的告警机制

支持历史数据追溯

具备可视化展示能力

支持检测结果导出

通过这些全面的自检机制，可以：

及时发现系统问题

预防潜在故障

确保系统稳定性

提供优化依据

支持运维决策

## 5.5、数据同步规范

结构化数据：

通过总线同步

通过数据库同步

非结构化数据：

通过总线同步

通过对象存储同步

支持目录监测和自动同步

## 5.6、报警机制

统一的告警规则处理

支持多渠道告警：

总线通知

邮件通知

短信通知(可选)

## 5.7、协议与扩展

不支持的协议处理：

纳入总线管理

通过通用协议适配器扩展

统一的数据协议格式：

确保数据交互标准化

支持灵活组装

降低开发成本