SOPHGO 算能

目 录

Agenda

SOPHON.NET平台总体介绍

平台总体框架

方案架构图

应用服务

监控中心 | 报警中心 | 信息可视化中心 | 应急处理中心

数据处理及存储 | 视频监控分析 | 可视化地图定位 | 报警推送 | 实时预警 | 实时分析 | ...

算法监测

通用算法

人员逗留 | 睡岗识别 | 人脸识别 | 异常聚集识别

在岗识别 | 烟雾、明火识别 | 跌倒识别 | 人流统计 | ...

可选算法

消防通道占用识别 | 可疑人员识别 | 垃圾堆放识别

危险行为 | 灭火器识别 | 危险物品识别 | ...

算力硬件

高算力需求

单台支持200路+视频分析

SG6 智算服务器

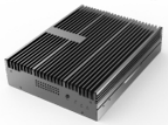
中算力需求

单台支持96~192路视频分析

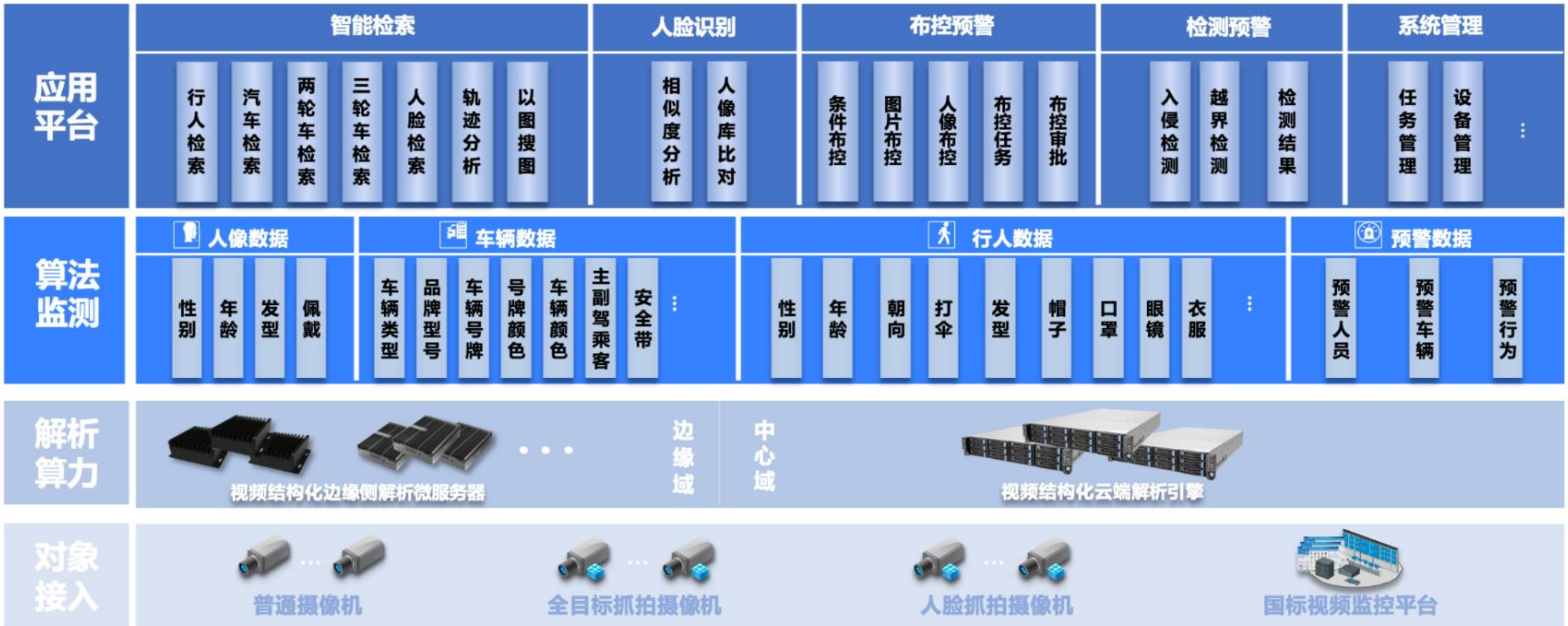
SE6微服务器

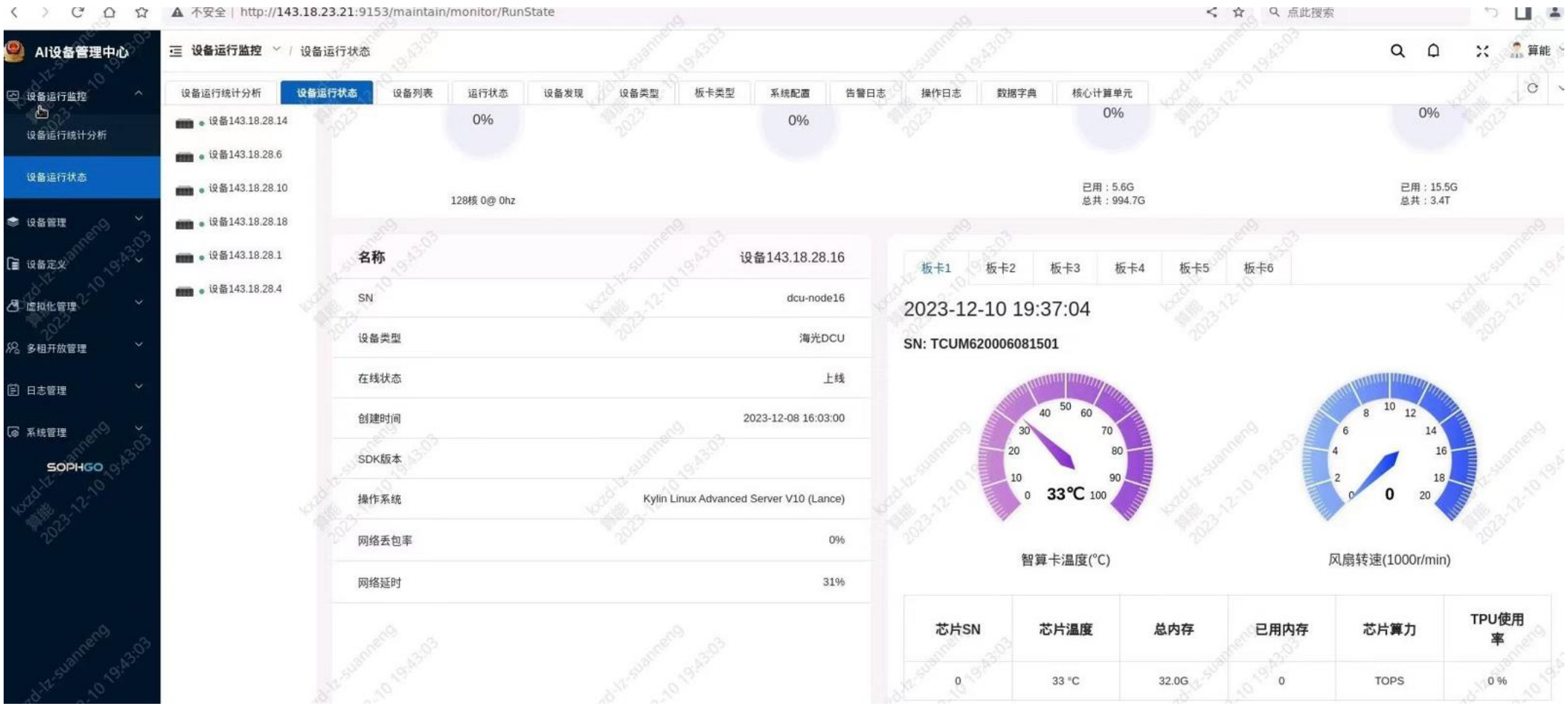
小算力需求

单台支持8~32路视频分析

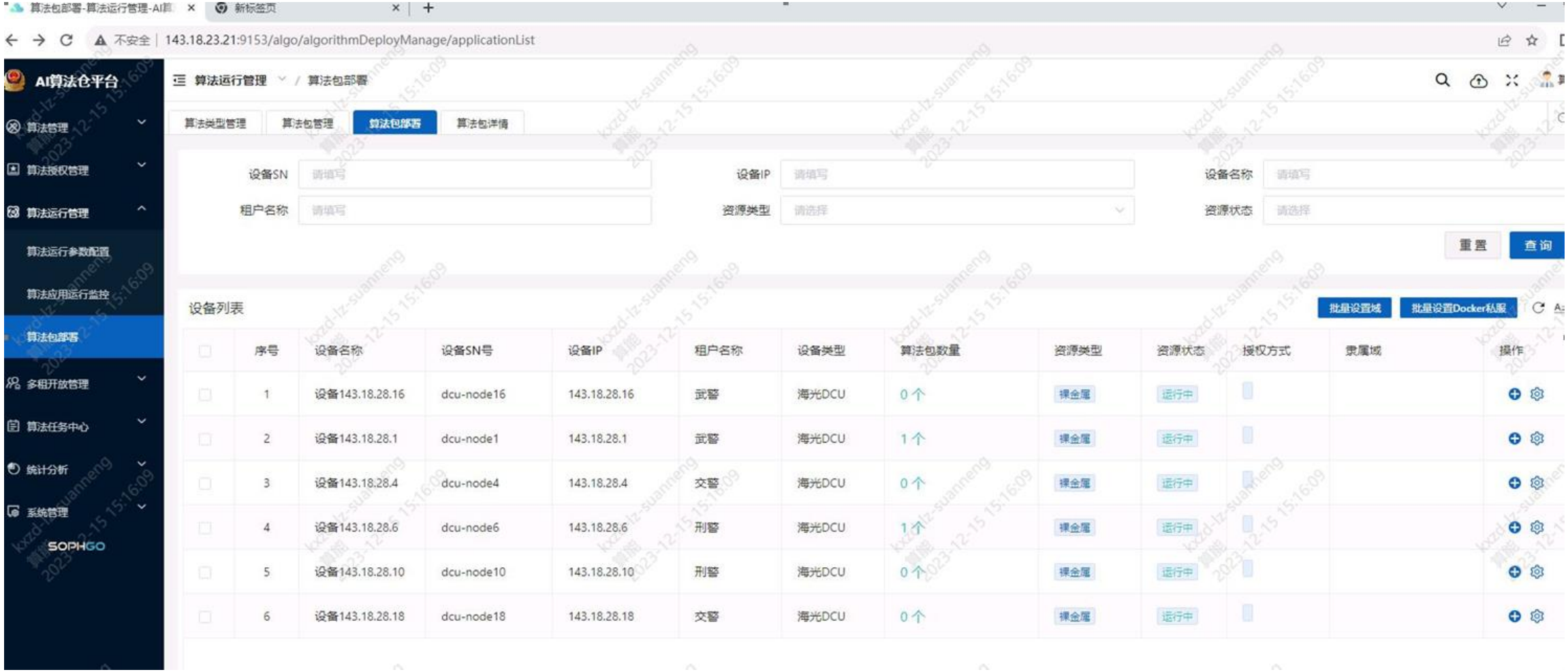


SE5微服务器

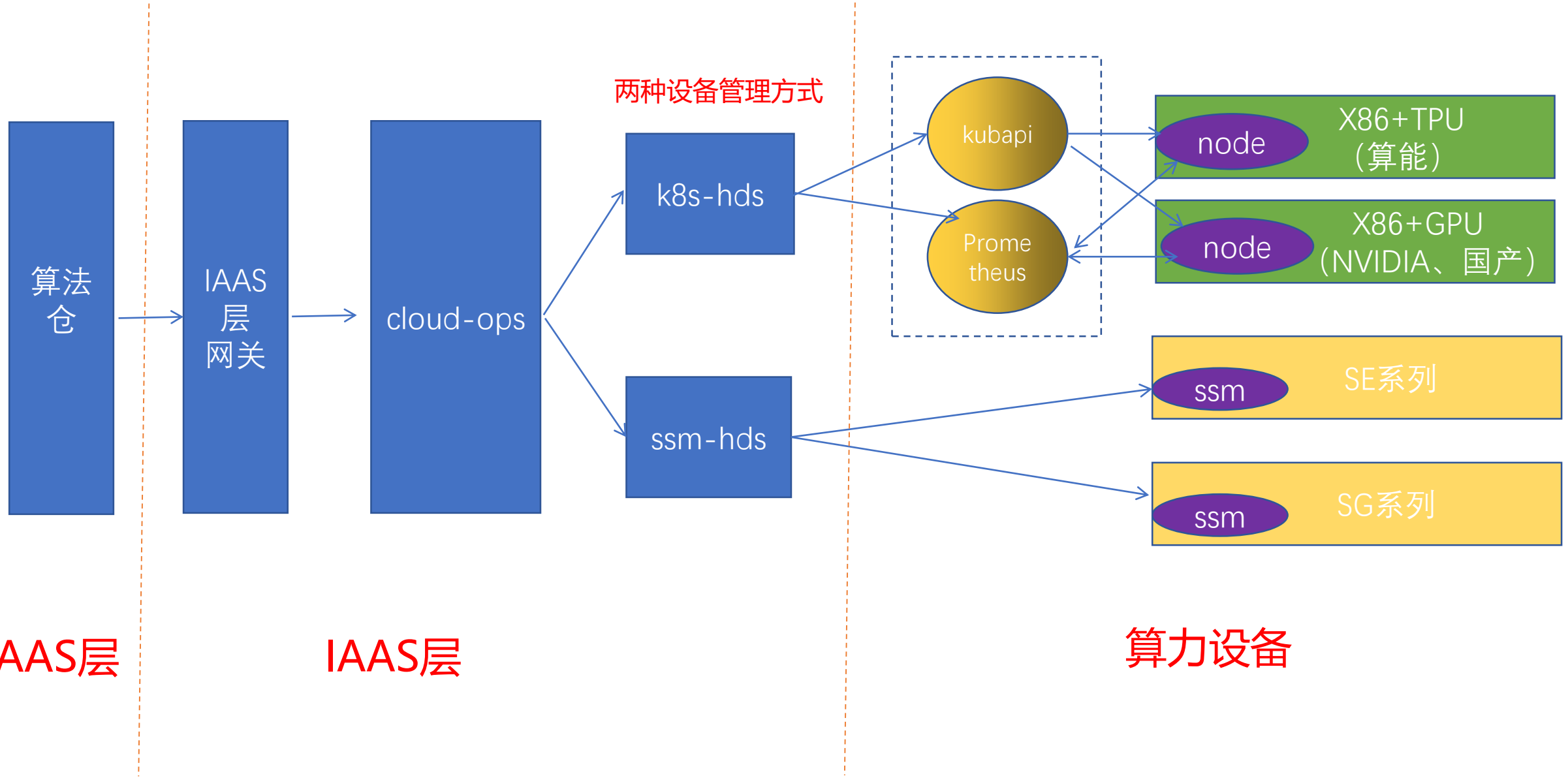

应用平台
智能检索
行人检索
汽车检索
两轮车检索
三轮车检索
人脸检索
轨迹分析
以图搜图
人脸识别
相似度分析
人像库比对
布控预警
条件布控
图片布控
人像布控
布控任务
布控审批
检测预警
入侵检测
越界检测
检测结果
系统管理
任务管理
设备管理
算法监测
人像数据
性别
年龄
发型
佩戴
车辆数据
车辆类型
品牌型号
车辆号牌
号牌颜色
车辆颜色
主副驾乘客
安全带
行人数据
性别
年龄
朝向
打伞
发型
帽子
口罩
眼镜
衣服
预警数据
预警人员
预警车辆
预警行为
解析算力
视频结构化边缘侧解析微服务器
边缘域
中心域
视频结构化云端解析引擎
对象接入
普通摄像机
全目标抓拍摄像机
人脸抓拍摄像机
国标视频监控平台


不安全 | http://143.18.23.21:9153/maintain/monitor/RunState
AI设备管理中心
设备运行监控
设备运行统计分析
设备运行状态
设备管理
设备定义
虚拟化管理
多租开放管理
日志管理
系统管理
SOPHGO
设备运行监控 / 设备运行状态
设备运行统计分析
设备运行状态
设备列表
运行状态
设备发现
设备类型
板卡类型
系统配置
告警日志
操作日志
数据字典
核心计算单元
设备143.18.28.14
设备143.18.28.6
设备143.18.28.10
设备143.18.28.18
设备143.18.28.1
设备143.18.28.4
0%
0%
0%
0%
128核 0@ 0hz
已用：5.6G
总共：994.7G
已用：15.5G
总共：3.4T
名称
设备143.18.28.16
SN
dcu-node16
设备类型
海光DCU
在线状态
上线
创建时间
2023-12-08 16:03:00
SDK版本
操作系统
Kylin Linux Advanced Server V10 (Lance)
网络丢包率
0%
网络延时
31%
板卡1
板卡2
板卡3
板卡4
板卡5
板卡6
2023-12-10 19:37:04
SN: TCUM620006081501
33℃
智算卡温度(℃)
0
风扇转速(1000r/min)
芯片SN
芯片温度
总内存
已用内存
芯片算力
TPU使用率
0
33 °C
32.0G
0
TOPS
0 %


7%
4%
81%
SN: HQDZW7BBCJAAA0017
43℃
0


算法包部署-算法运行管理-AI算
新标签页
不安全 | 143.18.23.21:9153/algo/algorithmDeployManage/applicationList
AI算法仓平台
算法运行管理 / 算法包部署
算法管理
算法授权管理
算法运行管理
算法运行参数配置
算法应用运行监控
算法包部署
多租开放管理
算法任务中心
统计分析
系统管理
SOPHGO
算法类型管理
算法包管理
算法包部署
算法包详情
设备SN
请填写
设备IP
设备名称
租户名称
资源类型
请选择
资源状态
重置
查询
设备列表
批量设置域
批量设置Docker私服
序号
设备名称
设备SN号
设备IP
租户名称
设备类型
算法包数量
资源类型
资源状态
授权方式
隶属域
操作
1 设备143.18.28.16 dcu-node16 143.18.28.16 武警 海光DCU 0 个 裸金属 运行中
2 设备143.18.28.1 dcu-node1 143.18.28.1 武警 海光DCU 1 个 裸金属 运行中
3 设备143.18.28.4 dcu-node4 143.18.28.4 交警 海光DCU 0 个 裸金属 运行中
4 设备143.18.28.6 dcu-node6 143.18.28.6 刑警 海光DCU 1 个 裸金属 运行中
5 设备143.18.28.10 dcu-node10 143.18.28.10 刑警 海光DCU 0 个 裸金属 运行中
6 设备143.18.28.18 dcu-node18 143.18.28.18 交警 海光DCU 0 个 裸金属 运行中

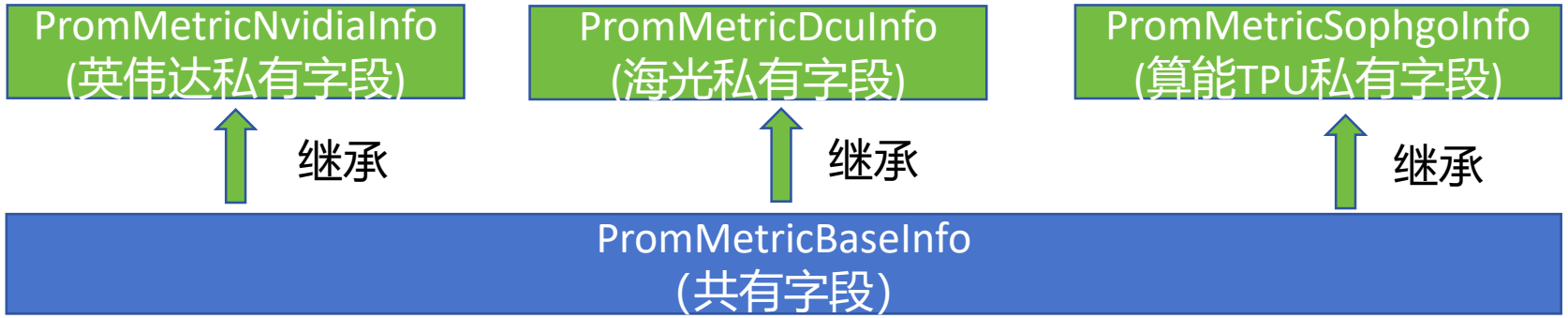
IAAS层架构


两种设备管理方式
算法仓
IAAS层网关
cloud-ops
k8s-hds
ssm-hds
kubapi
Prometheus
node
X86+TPU（算能）
node
X86+GPU（NVIDIA、国产）
ssm
SE系列
ssm
SG系列
PAAS层
IAAS层
算力设备

Prometheus指标根据厂商分类（不同厂商不同插件）

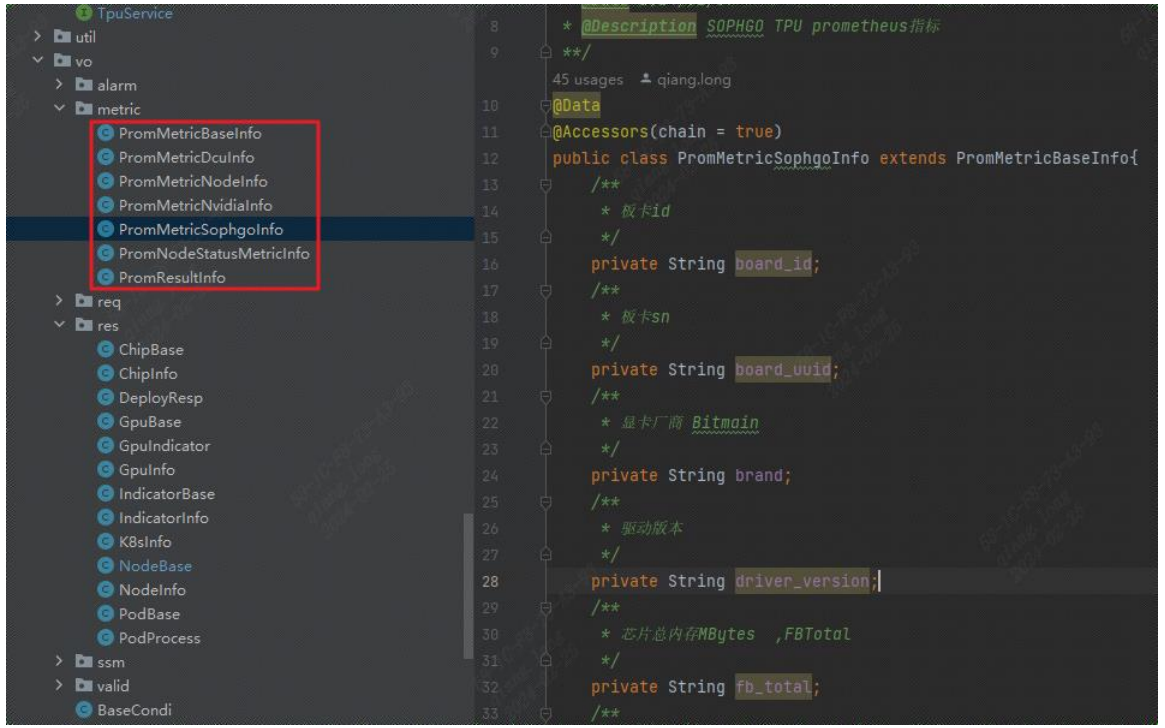
```java
 * @Description prometheus指标接收类
 **/
qiang.long
@Data
@Accessors(chain = true)
public class PromResultInfo<T> {
    /**
     * prometheus指标属性
     */
    private T metric;
    /**
     * 别名
     */
    private String alias;
    /**
     * prometheus指标值
     */
    private String[] value;

}
```




```java
 * @Date 2024/02/07
 * @Description prometheus指标metric共有字段封装  其他指标类的基类（定义时严
 **/
8 usages  5 inheritors  qiang.long
@Data
@Accessors(chain = true)
public class PromMetricBaseInfo {
    /**
     * prometheus指标名称
     */
    private String __name__;
    /**
     * 组件名
     **/
    private String service;
    /**
     * 系统pod名称
     */
    private String pod;
    /**
     * 系统pod命名空间
     */
    private String namespace;
    /**
     * prometheus实例名称
     */
    private String instance;

    /**
     * prometheus任务名称
```




```java
 * @Description SOPHGO TPU prometheus指标
 **/
45 usages  qiang.long
@Data
@Accessors(chain = true)
public class PromMetricSophgoInfo extends PromMetricBaseInfo{
    /**
     * 板卡id
     */
    private String board_id;
    /**
     * 板卡sn
     */
    private String board_uuid;
    /**
     * 显卡厂商 Bitmain
     */
    private String brand;
    /**
     * 驱动版本
     */
    private String driver_version;
    /**
     * 芯片总内存MBytes  ,FBTotal
     */
    private String fb_total;
    /**
```

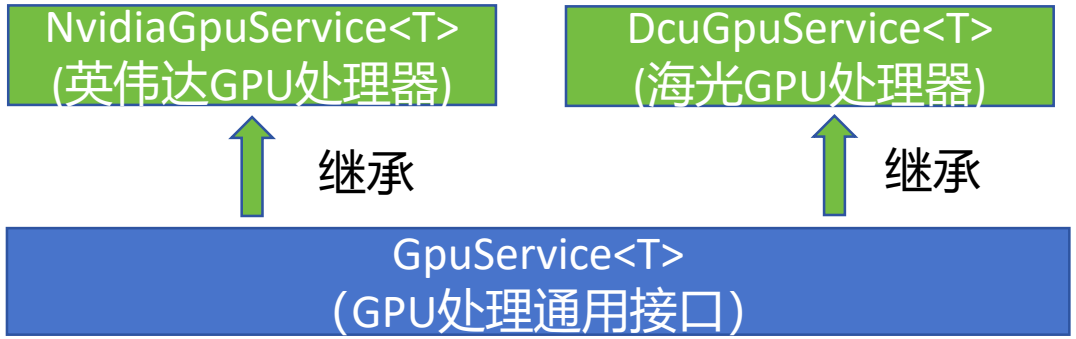
```java
 * @Description nvidia prometheus指标
 **/
18 usages  qiang.long
@Data
@Accessors(chain = true)
public class PromMetricNvidiaInfo extends PromMetricBaseInfo {
    /**
     * dcgm 驱动版本
     */
    @JsonProperty("DCGM_FI_DRIVER_VERSION")
    private String dcgmFiDriverVersion;
    /**
     * 板子编号
     */
    @JsonProperty("UUID")
    private String uuid;
    /**
     * 所在节点hostname
     */
    @JsonProperty("Hostname")
    private String hostName;

    /**
     * gpu名称，如nvidia0
     */
    private String device;
    /**
     * gpu编号
```

GpuService：通用GPU处理接口

```
 * @Description Gpu通用处理接口
 **/
11 usages  6 implementations  qiang.long
public interface GpuService<T> {
    /**
     * 查询gpu的信息
     * @param gpuReq
     * @return
     */
    3 implementations  qiang.long
    List<GpuInfo> gpuInfos(GpuReq gpuReq);
    /**
     * @description: 查询单个GPU详情
     * @param:
     * @param gpuReq
     * @return: Result<GpuInfo>
     **/
    1 usage  3 implementations  qiang.long
    Result<GpuInfo> gpuInfo(GpuReq gpuReq);
    /**
     * @description: Prometheus Gpu信息转化
     * @param:
     * @param promResultInfo
     * @return: GpuInfo
     **/
    4 usages  3 implementations  qiang.long
    GpuInfo gpuInfo(PromResultInfo<T> promResultInfo);


    /**
     * @description: 获取简略GPU信息
     * @param:
     * @param uuid GPU唯一标识
     * @return: GpuInfo
     **/
    3 usages  3 implementations  qiang.long
    GpuInfo simpleGpuInfo(String uuid);
```




```
    GpuBaseInfo getGpuBaseInfo();

    10 usages  qiang.long
    @Data
    @Accessors(chain = true)
    class GpuBaseInfo {
        /**
         * gpu类型
         */
        String  gpuType;
        /**
         * 名称
         */
        String gpuServiceName;
    }
    1 usage  qiang.long
    @Component
    class Tool {
        1 usage
        @Resource
        private NvidiaGpuService nvidiaGpuService;
        1 usage
        @Resource
        private DcuGpuService dcuGpuService;
        /**
         * @param
         * @description: 根据gpu类型判断哪种GPU
         * @param:
         * @return: GpuService
         **/
        2 usages  qiang.long
        public GpuService gpuServiceFetcher(GpuBrandEnum gpuBrandEnum) {
            switch (gpuBrandEnum)
            {
                case NVIDIA:
                    return nvidiaGpuService;
                case DCU:
                    return dcuGpuService;
                default:
                    return null;
            }
        }
    }
}
```

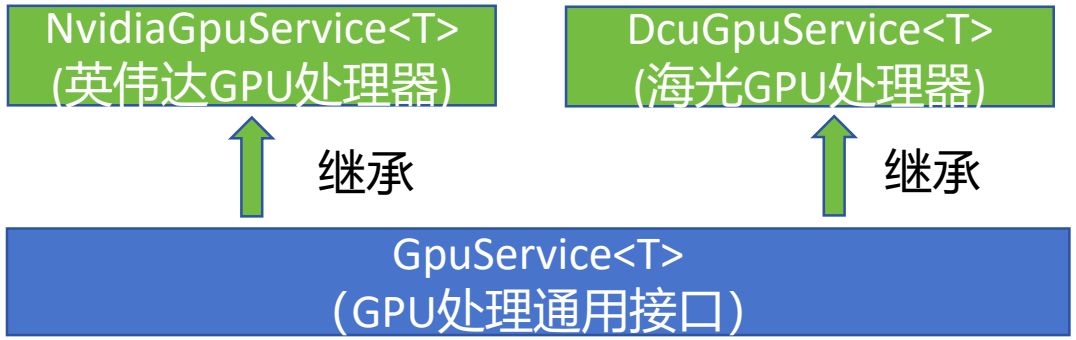
1. GpuService：主要封装指标适配和告警功能，具体实现根据GPU厂商类型转到各自Impl
2. 节点上或某一GPU的GPU厂商类型从缓存中查

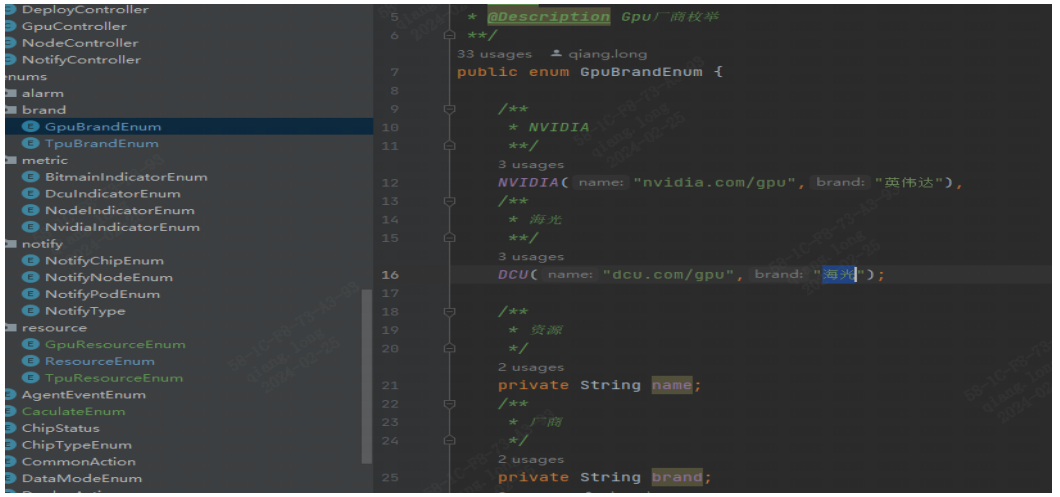
GpuService：通用GPU处理接口

GpuService中定义的方法

方法	说明
List<GpuInfo> gpuInfos(GpuReq gpuReq);	查询GPU列表（writeCache为true更新缓存，否则走正常流程）
Result<GpuInfo> gpuInfo(GpuReq gpuReq)	查询单个GPU详情
GpuInfo gpuInfo(PromResultInfo<T> promResultInfo)	将prometheus指标转换为GpuInfo，其中T继承PromMetricBaseInfo，根据各厂商插件不同
GpuInfo simpleGpuInfo(String uuid)	根据GPU唯一标识查询单个GPU简略信息
List<GpuBase> gpuBase(GpuReq gpuReq)	根据namespace和pod查绑定的gpu列表（基本信息）
GpuBase gpuBase(PromResultInfo<T> promResultInfo)	将prometheus接收结果转化为GpuBase
Map<String, String> gpuIndicators(String uuid)	获取GPU级指标
List<GpuIndicator> gpuIndicator(String alias, GpuReq gpuReq)	单个GPU指标查询所有
String labelPromeSql(GpuReq gpuReq, String indicator)	拼接promSQL
K8sInfo supplyK8sInfo(PromResultInfo<T> promNewResultInfo)	补充GPU上k8s关联信息
List<PrometheusRule.Spec.Rule> alarmConfigReqToRule(AlarmConfigReq.AlarmThreshold alarmThreshold)	告警参数配置请求转为Rule
GpuBaseInfo getGpuBaseInfo()	获取GpuService基本信息

通用GpuService接口处理不同的GPU





GPU分类处理器

GPU厂商枚举

```
1 usage
@Resource
private GpuService.Tool gpuTool;

/**
* @description: GPU分类处理
**/
no usages
private final BiFunction<GpuEnum, Function<GpuService, ?>, ?> gpuDealer = (gpuEnum, func) ->
        Optional.ofNullable(gpuEnum) Optional<GpuEnum>
                .map(gpuTool::gpuServiceFetcher) Optional<GpuService>
                .map(func) Optional<capture of ?>
                .orElse( other: null);
```

TpuService：通用TPU处理接口

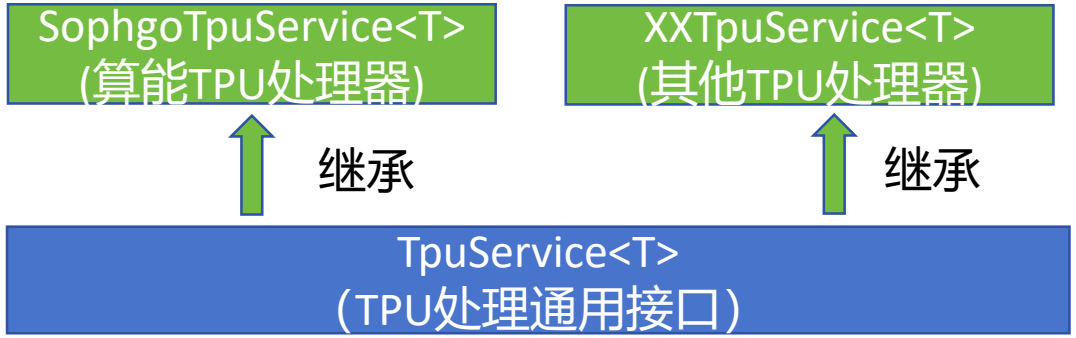
```java
/**
 * @Description Tpu通用处理接口
 **/
10 usages  2 implementations  qiang.long
public interface TpuService<T> {
    /**
     * @description: 查询Tpu芯片列表 （writeCache为true更新缓存，否则走正常流程）
     * @param:
     * @param chipReq
     * @return: List<ChipInfo>
     **/
    1 implementation  qiang.long
    List<ChipInfo> chipInfos(ChipReq chipReq);

    /**
     * @description: Prometheus 信息转化为ChipInfo
     * @param:
     * @param promResultInfo
     * @return: ChipInfo
     **/
    3 usages  1 implementation  qiang.long
    ChipInfo chipInfo(PromResultInfo<T> promResultInfo);

    /**
     * @description: 查询单个TPU信息
     * @param:
     * @param chipReq
     * @return: Result<ChipInfo>
     **/
    1 usage  1 implementation  qiang.long
    Result<ChipInfo> chipInfo(ChipReq chipReq);

    /**
     * @description: 根据TPU芯片唯一标识和节点ip查简略芯片信息
     * @param:
     * @param slot
     * @param nodeIp
     * @return: ChipInfo
     **/
    2 usages  1 implementation  qiang.long
    ChipInfo simpleChipInfo(String slot,String nodeIp);

    /**
     * @description: 查询TPU芯片列表：如根据namespace和pod查询绑定的TPU芯片，以及根据状态（是否健康）
     * @param:
```




```java
@Data
@Accessors(chain = true)
class TpuBaseInfo {
    /**
     * tpu厂商类型
     */
    String  tpuType;
    /**
     * 名称
     */
    String tpuServiceName;
}
1 usage  qiang.long
@Component
class Tool {
    1 usage
    @Resource
    private SophgoTpuService sophgoTpuService;
    /**
     * @param
     * @description: 根据tpu类型判断哪种TpuService
     * @param:
     * @return: TpuService
     **/
    2 usages  qiang.long
    public TpuService tpuServiceFetcher(TpuBrandEnum tpuBrandEnum) {
        switch (tpuBrandEnum)
        {
            case SOPHGO:
                return sophgoTpuService;
            default:
                return null;
        }
    }
}
```

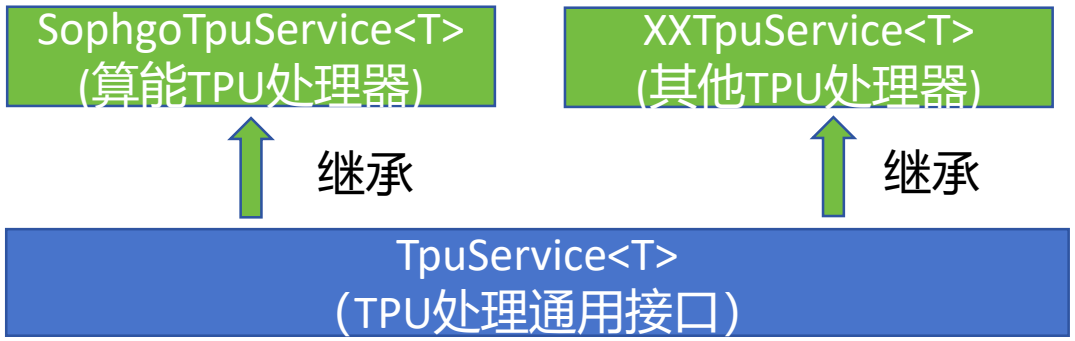
1. TpuService：主要封装指标适配和告警功能，具体实现根据TPU厂商类型转到各自Impl
2. 节点上或某一TPU的 TPU厂商类型从缓存中查

TpuService：通用TPU处理接口

TpuService中定义的方法

方法	说明
List<ChipInfo> chipInfos(ChipReq chipReq)	查询Tpu芯片列表 （writeCache为true更新缓存，否则走正常流程）
ChipInfo chipInfo(PromResultInfo<T> promResultInfo)	Prometheus 信息转化为ChipInfo
Result<ChipInfo> chipInfo(ChipReq chipReq)	查询单个TPU信息
ChipInfo simpleChipInfo(String slot,String nodeIp)	根据TPU芯片唯一标识和节点ip查简略芯片信息
List<ChipBase> chipBase(ChipReq chipReq)	查询TPU芯片列表：如根据namespace和pod查询绑定的TPU芯片，以及根据状态（是否健康）
void setchipBase(PromResultInfo<PromMetricSophgoInfo> promResultInfo,ChipBase chipBase)	Prometheus数据转ChipBase
Map<String,String> tpuIndicators(String slot,String node)	查询单个tpu芯片的 全部指标
List<IndicatorInfo> chipIndicator(String alias, ChipReq chipReq)	单个TPU指标查询全部
String simpleIndicator(String slot,String instance,String indicator)	查询单个tpu的简易指标
String labelPromeSql(ChipReq chipReq, String indicator)	拼接promSQL
List<PrometheusRule.Spec.Rule> alarmConfigReqToRule(AlarmConfigReq.AlarmThreshold alarmThreshold)	告警参数配置请求转为Rule
TpuBaseInfo getTpuBaseInfo()	获取TpuService基本信息

通用TpuService接口处理不同的TPU



TPU分类处理器

```
 * @Description TPU厂商枚举
 **/
11 usages   qiang.long
@Getter
public enum TpuBrandEnum {
    SOPHGO( num: 0, brand: "算能科技"),
    OTHER( num: 1, brand: "其他");

    Integer num;
    String brand;
    2 usages   qiang.long
    TpuBrandEnum(Integer num, String brand) {
        this.num = num;
        this.brand = brand;
    }
```

TPU厂商枚举

```
@Resource
private TpuService.Tool tpuTool;

/**
 * @description: TPU分类处理
 **/
1 usage
private final BiFunction<TpuBrandEnum, Function<TpuService, ?>, ?> tpuDealer = (tpuEnum, func) ->
        Optional.ofNullable(tpuEnum) Optional<TpuBrandEnum>
                .map(tpuTool::tpuServiceFetcher) Optional<TpuService>
                .map(func) Optional<capture of ?>
                .orElse( other: null);
```

Redis缓存： 存储每种设备厂商下的节点ip集合

使用redis

定时任务实时同步:

1、每个厂商所含节点IP

2、每个厂商所含tpu或gpu的唯一标识uuid

TPU按厂家分

SOPHGO_TPU_SET {uuid1, uuid2,….,uuidn} 所有算能TPU的uuid集合

SOPHGO_TPU_NODE_SET {ip1, ip2,….,ip_n} 所有算能TPU节点集合

GPU按厂家分：

NVIDIA_GPU_SET {uuid1, uuid2,….,uuidn} 所有NVIDIA GPU的uuid集合

NVIDIA_GPU_NODE_SET {ip1, ip2,….,ip_n} 所有NVIDIA GPU节点集合

DCU_GPU_SET {uuid1, uuid2,….,uuidn} 所有海光 GPU的uuid集合

DCU_GPU_NODE_SET {ip1, ip2,….,ip_n} 所有海光 GPU节点集合