智慧能源管理系统UML类图说明文档

1. 文档概述

1.1 编写目的

本文档旨在详细说明智慧能源管理系统的UML类图设计，包括各业务模块的类结构、类之间的关系，以及面向对象设计的实现方案。通过类图的形式展现系统的静态结构，为系统的详细设计和编码实现提供指导。

1.2 文档范围

- 系统人员管理类图
- 厂区信息管理类图
- 配电网监控类图
- 综合能耗管理类图
- 分布式光伏类图
- 告警运维类图
- 大屏展示类图
- 全局UML类图整合

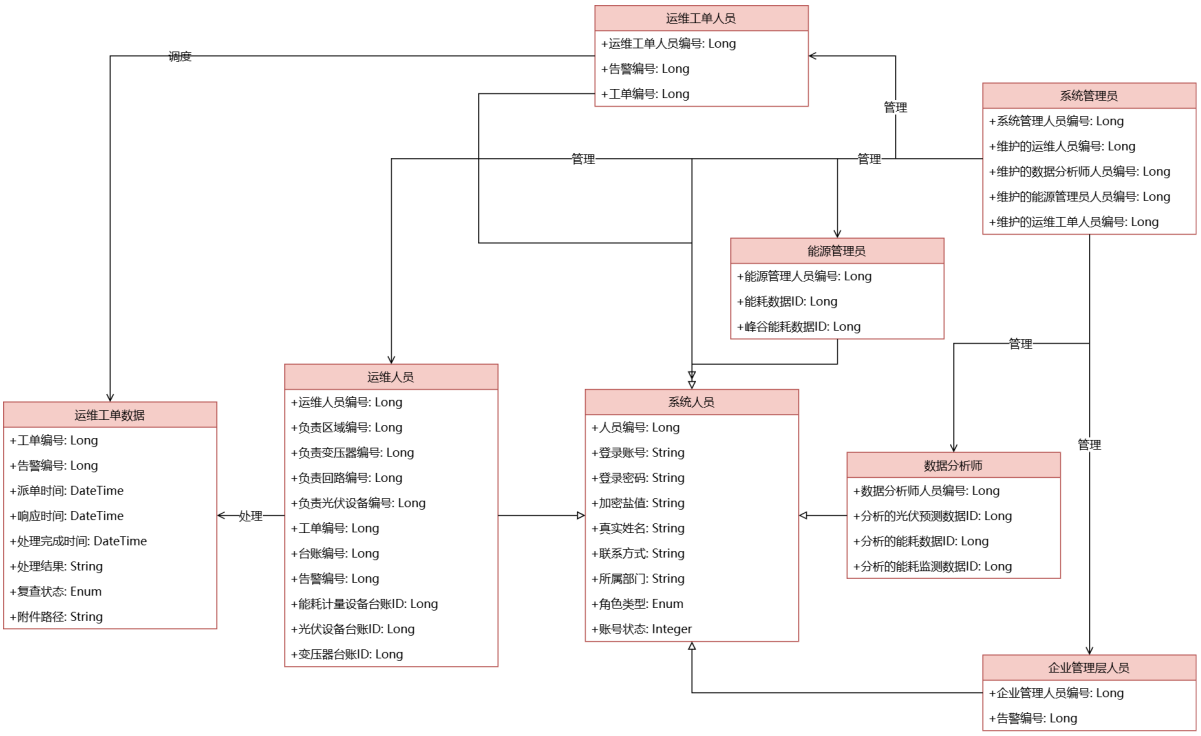
1.3 设计原则

- **封装性**：合理封装类的属性和方法
- **继承性**：通过继承实现代码复用
- **多态性**：支持多态机制的灵活调用
- **单一职责**：每个类承担单一的业务职责
- **开闭原则**：对扩展开放，对修改封闭

2. 局部UML类图设计

2.1 系统人员管理UML类图

2.1.1 类图展示



2.1.2 核心类设计

SysUser类

```
public class SysUser {
    // 属性
    private Long userId;           // 用户ID
    private String loginAccount;   // 登录账号
    private String loginPassword;  // 登录密码
    private String salt;           // 加密盐值
    private String realName;       // 真实姓名
    private String department;     // 所属部门
    private String contactPhone;   // 联系电话
    private Integer accountStatus; // 账号状态
    private LocalDateTime createdTime; // 创建时间

    // 关联关系
    private List<SysRoleAssignment> roleAssignments;

    // 方法
    public boolean validatePassword(String password);
    public void updateLastLoginTime();
    public boolean isActive();
}
```

SysRoleAssignment类

```
public class SysRoleAssignment {
    // 属性
    private Long assignmentId;    // 分配ID
    private Long userId;          // 用户ID
    private String roleType;      // 角色类型
    private Long assignedBy;      // 分配人
    private LocalDateTime assignedTime; // 分配时间
```

```
    // 关联关系
    private SysUser user;
    private RoleSysAdmin assignedByAdmin;

    // 方法
    public boolean isValidRole();
    public void assignRole();
    public void revokeRole();
}
```

角色类层次结构

```
// 抽象角色基类
public abstract class BaseRole {
    protected Long roleId;
    protected Long userId;

    public abstract List<String> getPermissions();
    public abstract String getRoleDescription();
}

// 具体角色类
public class RoleSysAdmin extends BaseRole {
    private List<String> managedUserTypes;

    @Override
    public List<String> getPermissions() {
        return Arrays.asList("USER_MANAGE", "SYSTEM_CONFIG", "DATA_EXPORT");
    }
}

public class RoleOandM extends BaseRole {
    private List<String> responsibleAreas;

    @Override
    public List<String> getPermissions() {
        return Arrays.asList("DEVICE_MONITOR", "ALARM_HANDLE", "WORK_ORDER");
    }
}
```

2.1.3 类关系说明

- **SysUser与SysRoleAssignment**：一对多组合关系，用户可以有多个角色分配
- **SysRoleAssignment与具体角色类**：多对一关联关系
- **角色类继承关系**：各具体角色类继承自BaseRole抽象类
- **依赖关系**：用户类依赖于密码工具类进行密码验证

2.2 厂区信息管理

2.2.1 核心类设计

BaseFactory类

```
public class BaseFactory {
    // 属性
    private Long factoryId;        // 厂区ID
    private String factoryName;    // 厂区名称
    private String areaDesc;       // 区域描述
    private Long managerUserId;    // 负责人用户ID

    // 关联关系
    private SysUser manager;
    private List<DeviceLedger> devices;
    private List<DistRoom> rooms;
    private List<EnergyMeter> energyMeters;

    // 方法
    public void addDevice(DeviceLedger device);
    public List<DeviceLedger> getDevicesByType(String deviceType);
    public void assignManager(SysUser manager);
    public FactoryStatistics getStatistics();
}
```

DeviceLedger类

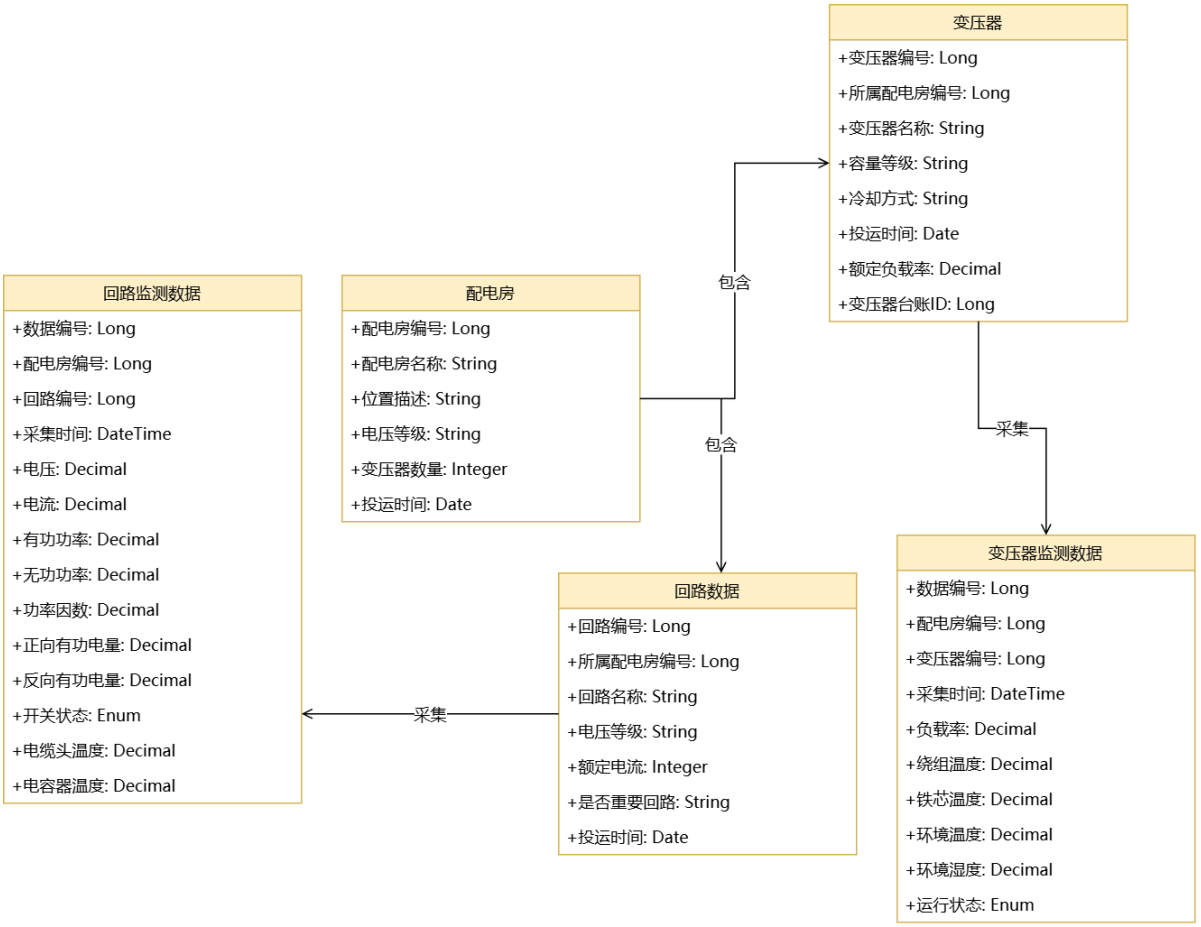
```
public class DeviceLedger {
    // 属性
    private Long ledgerId;         // 台账ID
    private String deviceName;     // 设备名称
    private String deviceType;     // 设备类型
    private String modelSpec;      // 型号规格
    private Date installTime;      // 安装时间
    private String scrapStatus;    // 报废状态

    // 关联关系
    private BaseFactory factory;
    private List<AlarmInfo> alarms;

    // 方法
    public void updateStatus(String status);
    public boolean isOperational();
    public void scheduleMaintenace();
    public MaintenanceRecord getMaintenanceHistory();
}
```

2.2.2 类关系说明

- **BaseFactory与DeviceLedger**：一对多聚合关系，厂区包含多个设备台账
- **BaseFactory与SysUser**：多对一关联关系，多个厂区可以有同一个负责人
- **DeviceLedger与AlarmInfo**：一对多关联关系，设备可以产生多个告警

2.3 配电网监控UML类图

2.3.1 类图展示



2.3.2 核心类设计

DistRoom类

```
public class DistRoom {
    // 属性
    private Long roomId;          // 配电室ID
    private String roomName;      // 配电室名称
    private String location;      // 位置
    private String voltageLevel;  // 电压等级
    private Long managerUserId;   // 负责人ID
    private Long factoryId;       // 厂区ID

    // 关联关系
    private BaseFactory factory;
    private SysUser manager;
    private List<DistTransformer> transformers;
    private List<DistCircuit> circuits;

    // 方法
    public void addTransformer(DistTransformer transformer);
    public void addCircuit(DistCircuit circuit);
    public RoomStatus getCurrentStatus();
    public List<AlarmInfo> getActiveAlarms();
```

```
}
```

DistTransformer类

```
public class DistTransformer {
    // 属性
    private Long transformerId;   // 变压器ID
    private String transformerName; // 变压器名称
    private Long roomId;          // 配电室ID
    private Long ledgerId;        // 台账ID

    // 关联关系
    private DistRoom room;
    private DeviceLedger ledger;
    private List<DataTransformer> monitoringData;

    // 方法
    public void collectData();
    public DataTransformer getLatestData();
    public boolean checkAlarmConditions();
    public void triggerAlarm(String alarmType);
}
```

DataTransformer类

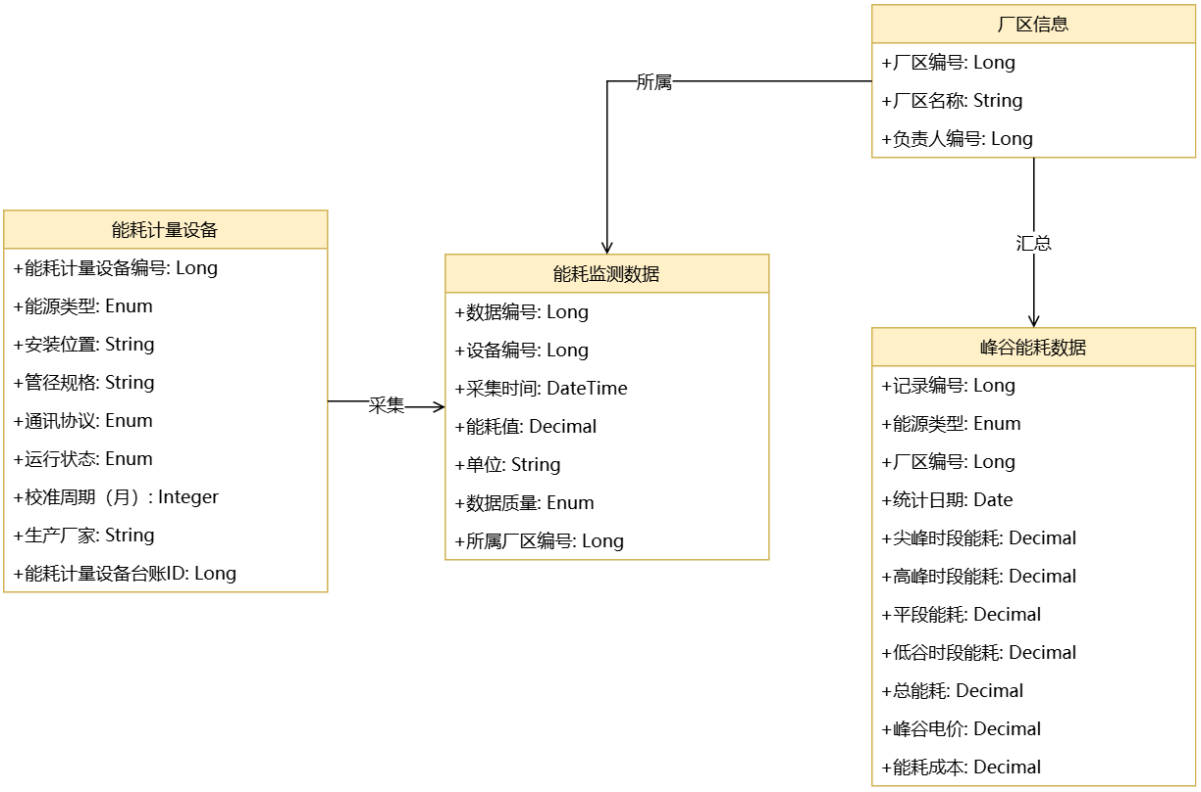
```
public class DataTransformer {
    // 属性
    private Long dataId;          // 数据ID
    private Long transformerId;   // 变压器ID
    private Timestamp collectTime; // 采集时间
    private BigDecimal windingTemp; // 绕组温度
    private BigDecimal coreTemp;   // 铁芯温度
    private BigDecimal loadRate;   // 负载率
    private Long factoryId;       // 厂区ID

    // 关联关系
    private DistTransformer transformer;

    // 方法
    public boolean isWithinNormalRange();
    public AlarmLevel evaluateAlarmLevel();
    public void archive();
}
```

2.3.3 类关系说明

- **DistRoom与DistTransformer/DistCircuit**：一对多组合关系
- **设备类与DeviceLedger**：一对一关联关系
- **设备类与监测数据类**：一对多聚合关系
- **监测数据类与告警类**：依赖关系，数据异常时创建告警

2.4 综合能耗管理UML类图

2.4.1 类图展示



2.4.2 核心类设计

EnergyMeter类

```
public class EnergyMeter {
    // 属性
    private Long meterId;          // 计量设备ID
    private String energyType;     // 能源类型
    private String commProtocol;   // 通讯协议
    private String runStatus;      // 运行状态
    private String installLocation; // 安装位置
    private Integer calibCycleMonths; // 校准周期
    private String manufacturer;   // 制造商
    private Long factoryId;        // 厂区ID
    private Long ledgerId;         // 台账ID

    // 关联关系
    private BaseFactory factory;
    private DeviceLedger ledger;
    private List<DataEnergy> energyData;

    // 方法
    public void collectEnergyData();
    public DataEnergy getLatestReading();
    public boolean needsCalibration();
    public void scheduleCalibration();
}
```

DataEnergy类

```
public class DataEnergy {
    // 属性
    private Long dataId;          // 数据ID
    private Long meterId;         // 计量设备ID
    private Timestamp collectTime; // 采集时间
    private BigDecimal value;     // 数值
    private String unit;          // 单位
    private String quality;       // 数据质量
    private Long factoryId;       // 厂区ID
    private Long pvRecordId;      // 峰谷记录ID

    // 关联关系
    private EnergyMeter meter;
    private DataPeakValley peakValleyRecord;

    // 方法
    public boolean isValidData();
    public BigDecimal convertToStandardUnit();
    public void assignToPeakValley();
}
```

ConfigPeakValley类

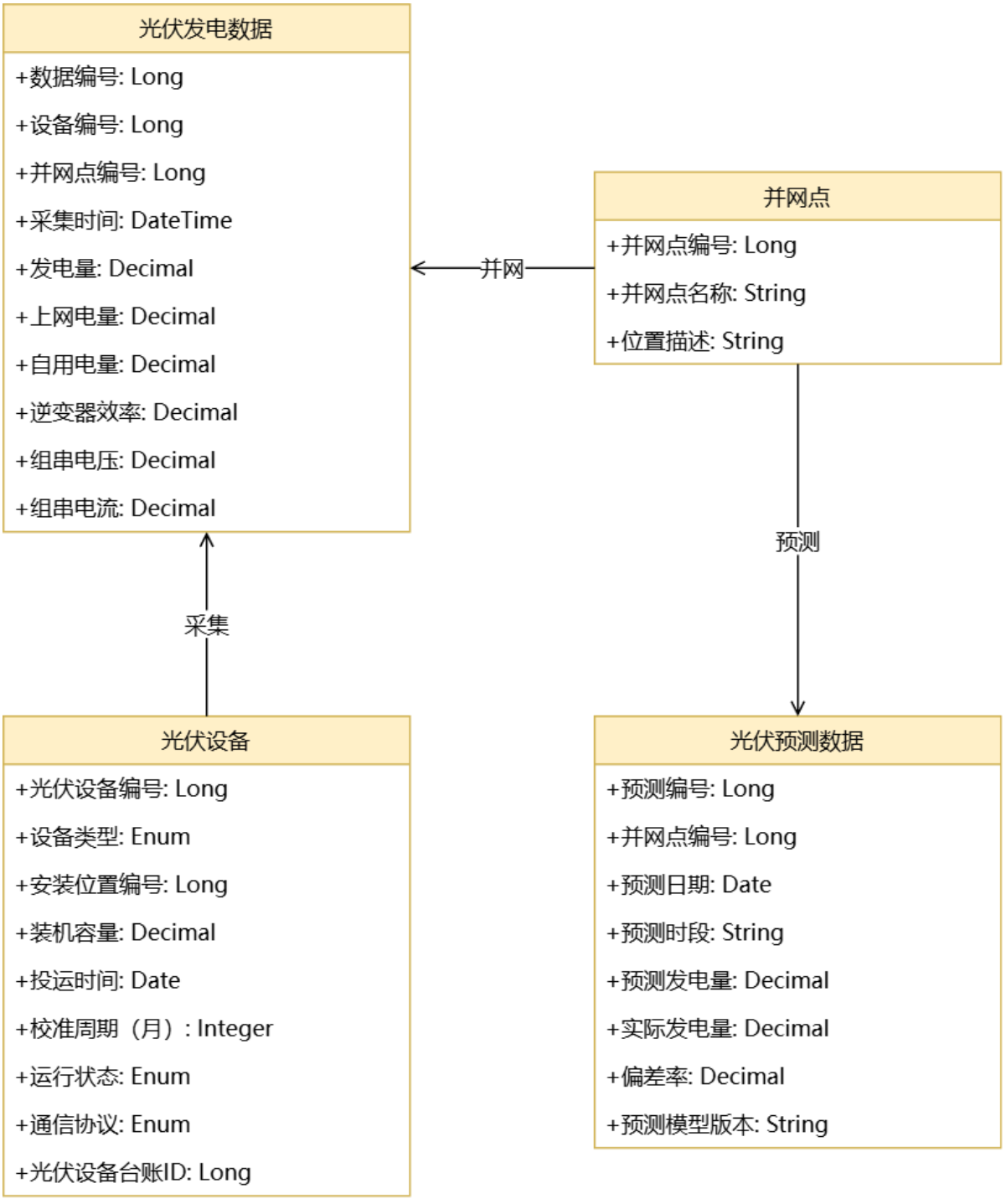
```
public class ConfigPeakValley {
    // 属性
    private Long configId;        // 配置ID
    private String timeType;      // 时段类型
    private Time startTime;       // 开始时间
    private Time endTime;         // 结束时间
    private BigDecimal priceRate; // 价格费率

    // 方法
    public boolean isInTimeRange(Timestamp time);
    public BigDecimal calculateCost(BigDecimal consumption);
    public static ConfigPeakValley getConfigByTime(Timestamp time);
}
```

2.4.3 类关系说明

- **EnergyMeter与DataEnergy**：一对多聚合关系
- **DataEnergy与DataPeakValley**：多对一关联关系
- **ConfigPeakValley与DataPeakValley**：一对多关联关系
- **策略模式应用**：不同能源类型使用不同的数据处理策略

2.5 分布式光伏UML类图

2.5.1 类图展示



2.5.2 核心类设计

PVGridPoint类

```
public class PVGridPoint {
    // 属性
    private Long pointId;          // 并网点ID
    private String pointName;      // 并网点名称
    private String location;       // 位置

    // 关联关系
    private List<PVDevice> devices;
```

```
    private List<DataPVForecast> forecasts;

    // 方法
    public void addDevice(PVDevice device);
    public BigDecimal getTotalCapacity();
    public BigDecimal getCurrentGeneration();
    public List<DataPVForecast> getForecastData(Date date);
}
```

PVDevice类

```
public class PVDevice {
    // 属性
    private Long deviceId;        // 设备ID
    private String deviceType;    // 设备类型
    private Double capacity;      // 装机容量
    private String runStatus;     // 运行状态
    private Date installDate;     // 安装日期
    private String protocol;      // 通讯协议
    private Long pointId;         // 并网点ID
    private Long ledgerId;        // 台账ID

    // 关联关系
    private PVGridPoint gridPoint;
    private DeviceLedger ledger;
    private List<DataPVGen> generationData;

    // 方法
    public void collectGenerationData();
    public DataPVGen getLatestGeneration();
    public Double getEfficiencyRate();
    public boolean isOperational();
}
```

PVForecastModel类

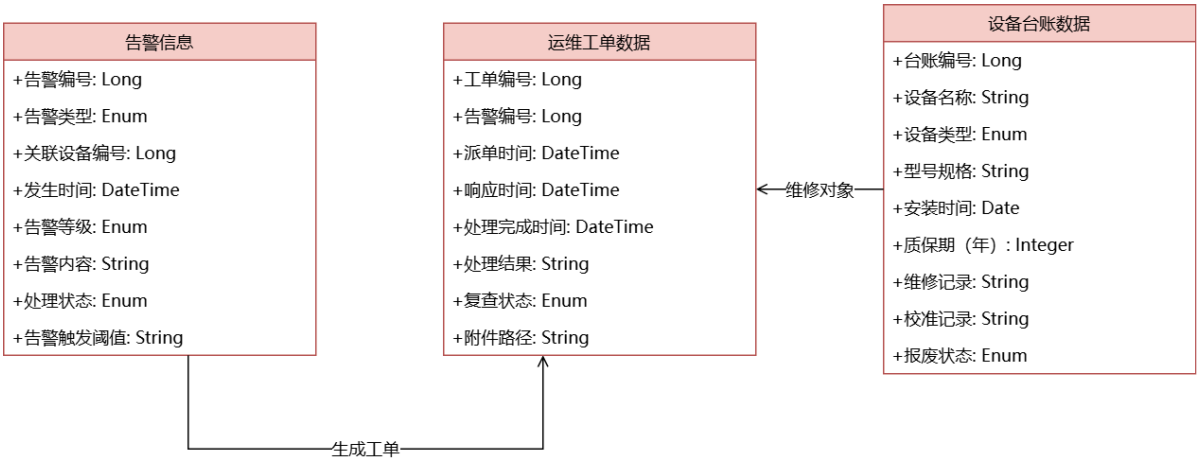
```
public class PVForecastModel {
    // 属性
    private String modelVersion;  // 模型版本
    private String modelName;     // 模型名称
    private String status;        // 状态
    private Timestamp updateTime; // 更新时间

    // 关联关系
    private List<DataPVForecast> forecasts;
    private List<PVModelAlert> alerts;

    // 方法
    public DataPVForecast generateForecast(Long pointId, Date date);
    public void updateModel();
    public Double evaluateAccuracy();
    public void triggerOptimizationAlert();
}
```

2.5.3 类关系说明

- **PVGridPoint与PVDevice**：一对多组合关系
- **PVDevice与DataPVGen**：一对多聚合关系
- **PVForecastModel与DataPVForecast**：一对多关联关系
- **观察者模式应用**：模型性能监控触发优化告警

2.6 告警运维UML类图

2.6.1 类图展示



2.6.2 核心类设计

AlarmInfo类

```
public class AlarmInfo {
    // 属性
    private Long alarmId;          // 告警ID
    private String alarmType;      // 告警类型
    private String alarmLevel;     // 告警等级
    private String content;        // 告警内容
    private LocalDateTime occurTime; // 发生时间
    private String processStatus; // 处理状态
    private Long ledgerId;         // 设备台账ID
    private Long factoryId;        // 厂区ID

    // 关联关系
    private DeviceLedger device;
    private BaseFactory factory;
    private WorkOrder workOrder;
    private List<AlarmHandlingLog> handlingLogs;

    // 方法
    public void createWorkOrder();
    public void updateStatus(String status);
    public boolean isHighPriority();
    public void escalate();
}
```

WorkOrder类

```
public class WorkOrder {
    // 属性
    private Long orderId;          // 工单ID
    private Long alarmId;          // 告警ID
    private Long oandMId;          // 运维人员ID
    private Long ledgerId;         // 设备台账ID
    private LocalDateTime dispatchTime; // 派单时间
    private LocalDateTime responseTime; // 响应时间
    private LocalDateTime finishTime;   // 完成时间
    private String resultDesc;     // 处理结果
    private String reviewStatus;   // 复查状态

    // 关联关系
    private AlarmInfo alarm;
    private RoleOandM maintainer;
    private DeviceLedger device;

    // 方法
    public void assignToMaintainer(RoleOandM maintainer);
    public void updateProgress(String progress);
    public void complete(String result);
    public boolean isOverdue();
}
```

AlarmHandlingLog类

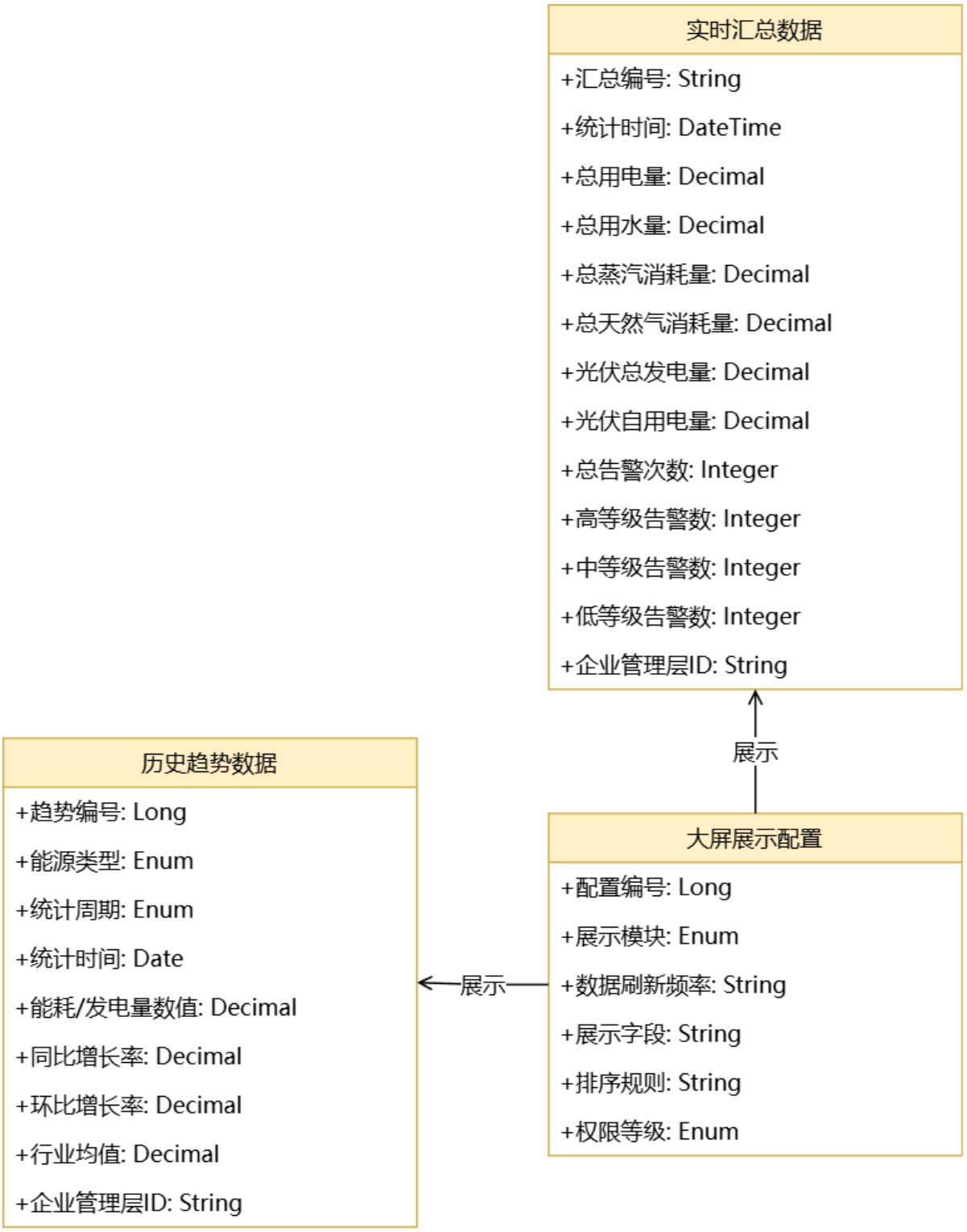
```
public class AlarmHandlingLog {
    // 属性
    private Long logId;            // 日志ID
    private Long alarmId;          // 告警ID
    private LocalDateTime handleTime; // 处理时间
    private String statusAfter;    // 处理后状态
    private Long oandMId;          // 运维人员ID
    private Long dispatcherId;     // 调度员ID

    // 关联关系
    private AlarmInfo alarm;
    private RoleOandM maintainer;
    private RoleDispatcher dispatcher;

    // 方法
    public void logAction(String action);
    public void recordStatusChange(String oldStatus, String newStatus);
}
```

2.6.3 类关系说明

- **AlarmInfo与WorkOrder**：一对一关联关系
- **WorkOrder与RoleOandM**：多对一关联关系
- **AlarmInfo与AlarmHandlingLog**：一对多组合关系
- **状态模式应用**：告警和工单的状态转换管理

2.7 大屏展示UML类图

2.7.1 类图展示



2.7.2 核心类设计

DashboardConfig类

```
public class DashboardConfig {
    // 属性
    private Long configId;          // 配置ID
    private String moduleName;      // 模块名称
```

```
    private String refreshRate;   // 刷新频率
    private String sortRule;      // 排序规则
    private String displayFields; // 显示字段
    private String authLevel;     // 权限等级

    // 关联关系
    private List<StatRealtime> realtimeData;
    private List<StatHistoryTrend> trendData;

    // 方法
    public boolean isAuthorized(String userRole);
    public void updateRefreshRate(String rate);
    public Map<String, Object> getDisplayData();
}
```

StatRealtime类

```
public class StatRealtime {
    // 属性
    private String summaryId;     // 汇总ID
    private Timestamp statTime;   // 统计时间
    private BigDecimal totalKWH;  // 总电量
    private Integer totalAlarm;   // 总告警数
    private BigDecimal pvGenKWH;  // 光伏发电量
    private Long configId;        // 配置ID
    private Long managerId;       // 管理层ID

    // 关联关系
    private DashboardConfig config;
    private RoleManager manager;

    // 方法
    public void updateStatistics();
    public Map<String, Object> toDisplayFormat();
    public boolean isDataFresh();
}
```

StatHistoryTrend类

```
public class StatHistoryTrend {
    // 属性
    private String trendId;       // 趋势ID
    private String energyType;    // 能源类型
    private String statCycle;     // 统计周期
    private Date statDate;        // 统计日期
    private BigDecimal value;     // 数值
    private BigDecimal yoyRate;   // 同比增长率
    private BigDecimal momRate;   // 环比增长率
    private Long configId;        // 配置ID
    private Long analystId;       // 分析师ID

    // 关联关系
    private DashboardConfig config;
    private RoleAnalyst analyst;
```

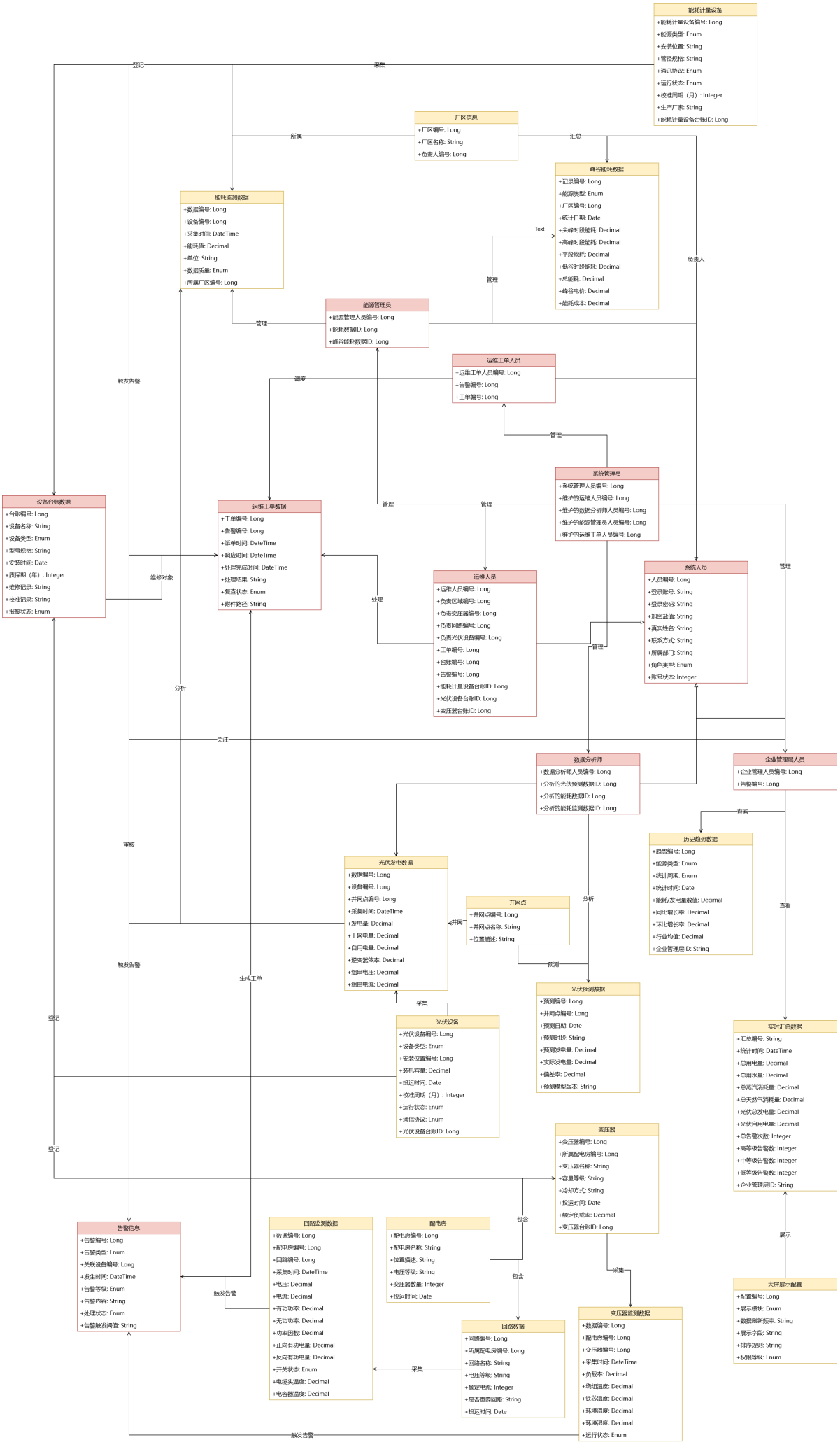
```
    // 方法
    public void calculateTrends();
    public ChartData generateChartData();
    public void exportToReport();
}
```

2.7.3 类关系说明

- **DashboardConfig与统计数据类**：一对多关联关系
- **统计数据类与角色类**：多对一关联关系
- **模板方法模式应用**：不同类型的统计数据使用统一的处理模板

3. 全局UML类图整合

3.1 全局类图展示


能耗计量设备
+能耗计量设备编号: Long
+能源类型: Enum
+安装位置: String
+管径规格: String
+通讯协议: Enum
+运行状态: Enum
+校准周期（月）: Integer
+生产厂家: String
+能耗计量设备台账ID: Long
登记
采集
厂区信息
+厂区编号: Long
+厂区名称: String
+负责人编号: Long
所属
汇总
峰谷能耗数据
+记录编号: Long
+能源类型: Enum
+厂区编号: Long
+统计日期: Date
+尖峰时段能耗: Decimal
+高峰时段能耗: Decimal
+平段能耗: Decimal
+低谷时段能耗: Decimal
+总能耗: Decimal
+峰谷电价: Decimal
+能耗成本: Decimal
能耗监测数据
+数据编号: Long
+设备编号: Long
+采集时间: DateTime
+能耗值: Decimal
+单位: String
+数据质量: Enum
+所属厂区编号: Long
Text
管理
负责人
能源管理员
+能源管理人员编号: Long
+能耗数据ID: Long
+峰谷能耗数据ID: Long
管理
管理
运维工单人员
+运维工单人员编号: Long
+告警编号: Long
+工单编号: Long
触发告警
调度
管理
系统管理员
+系统管理人员编号: Long
+维护的运维人员编号: Long
+维护的数据分析师人员编号: Long
+维护的能源管理员人员编号: Long
+维护的运维工单人员编号: Long
设备台账数据
+台账编号: Long
+设备名称: String
+设备类型: Enum
+型号规格: String
+安装时间: Date
+质保期（年）: Integer
+维修记录: String
+校准记录: String
+报废状态: Enum
运维工单数据
+工单编号: Long
+告警编号: Long
+派单时间: DateTime
+响应时间: DateTime
+处理完成时间: DateTime
+处理结果: String
+复查状态: Enum
+附件路径: String
管理
管理
管理
维修对象
处理
运维人员
+运维人员编号: Long
+负责区域编号: Long
+负责变压器编号: Long
+负责回路编号: Long
+负责光伏设备编号: Long
+工单编号: Long
+台账编号: Long
+告警编号: Long
+能耗计量设备台账ID: Long
+光伏设备台账ID: Long
+变压器台账ID: Long
系统人员
+人员编号: Long
+登录账号: String
+登录密码: String
+加密盐值: String
+真实姓名: String
+联系方式: String
+所属部门: String
+角色类型: Enum
+账号状态: Integer
管理
管理
分析
关注
数据分析师
+数据分析师人员编号: Long
+分析的光伏预测数据ID: Long
+分析的能耗数据ID: Long
+分析的能耗监测数据ID: Long
企业管理层人员
+企业管理人员编号: Long
+告警编号: Long
查看
历史趋势数据
+趋势编号: Long
+能源类型: Enum
+统计周期: Enum
+统计时间: Date
+能耗/发电量数值: Decimal
+同比增长率: Decimal
+环比增长率: Decimal
+行业均值: Decimal
+企业管理层ID: String
审核
光伏发电数据
+数据编号: Long
+设备编号: Long
+并网点编号: Long
+采集时间: DateTime
+发电量: Decimal
+上网电量: Decimal
+自用电量: Decimal
+逆变器效率: Decimal
+组串电压: Decimal
+组串电流: Decimal
并网
并网点
+并网点编号: Long
+并网点名称: String
+位置描述: String
分析
查看
触发告警
生成工单
预测
光伏预测数据
+预测编号: Long
+并网点编号: Long
+预测日期: Date
+预测时段: String
+预测发电量: Decimal
+实际发电量: Decimal
+偏差率: Decimal
+预测模型版本: String
采集
光伏设备
+光伏设备编号: Long
+设备类型: Enum
+安装位置编号: Long
+装机容量: Decimal
+投运时间: Date
+校准周期（月）: Integer
+运行状态: Enum
+通信协议: Enum
+光伏设备台账ID: Long
实时汇总数据
+汇总编号: String
+统计时间: DateTime
+总用电量: Decimal
+总用水量: Decimal
+总蒸汽消耗量: Decimal
+总天然气消耗量: Decimal
+光伏总发电量: Decimal
+光伏自用电量: Decimal
+总告警次数: Integer
+高等级告警数: Integer
+中等级告警数: Integer
+低等级告警数: Integer
+企业管理层ID: String
登记
登记
变压器
+变压器编号: Long
+所属配电房编号: Long
+变压器名称: String
+容量等级: String
+冷却方式: String
+投运时间: Date
+额定负载率: Decimal
+变压器台账ID: Long
告警信息
+告警编号: Long
+告警类型: Enum
+关联设备编号: Long
+发生时间: DateTime
+告警等级: Enum
+告警内容: String
+处理状态: Enum
+告警触发阈值: String
回路监测数据
+数据编号: Long
+配电房编号: Long
+回路编号: Long
+采集时间: DateTime
+电压: Decimal
+电流: Decimal
+有功功率: Decimal
+无功功率: Decimal
+功率因数: Decimal
+正向有功电量: Decimal
+反向有功电量: Decimal
+开关状态: Enum
+电缆头温度: Decimal
+电容器温度: Decimal
配电房
+配电房编号: Long
+配电房名称: String
+位置描述: String
+电压等级: String
+变压器数量: Integer
+投运时间: Date
包含
包含
展示
触发告警
采集
采集
回路数据
+回路编号: Long
+所属配电房编号: Long
+回路名称: String
+电压等级: String
+额定电流: Integer
+是否重要回路: String
+投运时间: Date
变压器监测数据
+数据编号: Long
+配电房编号: Long
+变压器编号: Long
+采集时间: DateTime
+负载率: Decimal
+绕组温度: Decimal
+铁芯温度: Decimal
+环境温度: Decimal
+环境湿度: Decimal
+运行状态: Enum
大屏展示配置
+配置编号: Long
+展示模块: Enum
+数据刷新频率: String
+展示字段: String
+排序规则: String
+权限等级: Enum
触发告警

3.2 核心类关系分析

3.2.1 继承关系

- **角色类继承体系**：所有角色类继承自BaseRole抽象类
- **设备类继承体系**：不同类型设备可以继承自BaseDevice抽象类
- **数据类继承体系**：各种数据类可以继承自BaseData抽象类

3.2.2 组合关系

- **用户-角色分配**：用户类组合角色分配类
- **厂区-设备**：厂区类组合设备台账类
- **配电室-设备**：配电室类组合变压器和回路类

3.2.3 聚合关系

- **设备-数据**：设备类聚合监测数据类
- **告警-日志**：告警类聚合处理日志类
- **配置-统计**：配置类聚合统计数据类

3.2.4 依赖关系

- **服务类依赖**：业务类依赖于服务类进行复杂操作
- **工具类依赖**：实体类依赖于工具类进行数据处理
- **异常类依赖**：所有类都可能依赖于异常类进行错误处理

3.3 类图设计特点

3.4.1 模块化设计

- 每个业务模块相对独立，降低耦合度
- 通过接口和抽象类定义模块间的交互规范
- 支持模块的独立开发和测试

3.4.2 可扩展性

- 使用继承和多态支持功能扩展
- 接口设计支持不同实现方式的替换
- 配置驱动的设计支持运行时行为调整

3.4.3 可维护性

- 清晰的类职责划分便于代码维护
- 统一的命名规范提高代码可读性
- 完善的异常处理机制保证系统稳定性