**综合监控（ISCS）概要设计说明书**

**实时数据库（RTDB）**

|  |  |  |  |
| --- | --- | --- | --- |
| 文档版本号： | V1.0 | 文档编号： | NG\_TS\_RTDB\_SDS |
| 文档密级： | 内部公开 | 归属部门/项目： | 专业系统部 |
| 编写人： | 阚文第 | 生效日期： | 2017-12-22 |

**版权信息**

本文件涉及之信息，属南京轨道交通系统工程有限公司所有。

未经南京轨道交通系统工程有限公司允许，文件中的任何部分都不能以任何形式向第三方散发。

网址：http://www.nanjingrail.com/

**文档修订记录**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **版本号** | **修订日期** | **修订人** | **修订说明** | **修订状态** | **审核日期** | **审核人** | **批准人** |
| V1.0 | 2017-12-22 | 阚文第 | 草稿版 | A |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

修订状态：A--增加，M--修改，D--删除

日期格式：YYYY-MM-DD

**目　录**

[1. 引言 1](#_Toc287377107)

[1.1. 编写目的 1](#_Toc287377108)

[1.2. 背景 1](#_Toc287377109)

[1.3. 术语 1](#_Toc287377110)

[1.4. 预期读者与阅读建议 1](#_Toc287377111)

[1.5. 参考资料 1](#_Toc287377112)

[2. 总体设计 2](#_Toc287377113)

[2.1. 设计概述 2](#_Toc287377114)

[2.1.1 设计约束 2](#_Toc287377115)

[2.1.2 设计策略 2](#_Toc287377116)

[2.1.3 设计实现 2](#_Toc287377117)

[2.2. 设计目标 2](#_Toc287377118)

[2.3. 运行环境 2](#_Toc287377119)

[2.4. 平台架构 2](#_Toc287377120)

[2.5. 总体设计思路和处理流程 3](#_Toc287377121)

[2.6. 模块结构设计 3](#_Toc287377122)

[2.7. 功能需求与程序模块的关系（可选） 3](#_Toc287377123)

[2.8. 尚未解决的问题 3](#_Toc287377124)

[3. 接口设计 3](#_Toc287377125)

[3.1. 用户接口 3](#_Toc287377126)

[3.2. 外部接口 4](#_Toc287377127)

[3.3. 内部接口 4](#_Toc287377128)

[4. 界面总体设计 4](#_Toc287377129)

[5. 数据结构设计 4](#_Toc287377130)

[5.1. 设计原则 4](#_Toc287377131)

[5.2. 数据库环境说明 4](#_Toc287377132)

[5.3. 数据库命名规则 4](#_Toc287377133)

[5.4. 逻辑结构 4](#_Toc287377134)

[5.5. 物理存储 4](#_Toc287377135)

[5.6. 数据备份和恢复 4](#_Toc287377136)

[6. 系统出错处理设计[可选] 4](#_Toc287377137)

[6.1. 出错信息 4](#_Toc287377138)

[6.2. 补救措施 5](#_Toc287377139)

[6.3. 系统维护设计 5](#_Toc287377140)

[7. 系统安全设计 5](#_Toc287377141)

[7.1. 数据传输安全性设计 5](#_Toc287377142)

[7.2. 应用系统安全性设计 5](#_Toc287377143)

[7.3. 数据存储安全性设计 5](#_Toc287377144)

[8. 系统部署（可选） 6](#_Toc287377145)

# 引言

* 1. 编写目的

本文为地铁ISCS系统实时数据库子系统设计说明书，本部分与ISCS系统总体设计和其他各子系统设计共同组成ISCS概要设计文件。

系统的接口设计遵守《地铁ISCS系统技术规范》的规定，此技术标准与本概要设计作为软件详细设计和程序编制的设计依据。

* 1. 背景

ISCS系统负责地铁一条线路上的全部设备和系统监视和控制，ISCS系统需要维护大概30~100万个监视和控制数据点。这些点的状态发生变化，ISCS系统能够及时把状态的变化成现在监视画面上；监视终端给设备或系统发送控制命令，ISCS系统能够及时把控制下发给对应的设备或系统。

ISCS系统使用实时数据库RTDB来管理维护这些监控和控制数据点，充分利用实时数据库的特性，来提高整个ISCS系统的数据处理能力和数据查询速度。

RTDB作为可以独立部署运行的程序存在。

* 1. 术语

术语和缩写参照《综合监控（实时数据库）软件需求规格说明书》。

* 1. 预期读者与阅读建议

|  |  |
| --- | --- |
| 预期读者 | 阅读重点 |
| 需求发起者和需求调研人员 | 系统架构和功能模块是否满足需求 |
| 项目负责人 | 系统架构设计是否合理，功能模块划分是否明确 |
| 系统设计人员 | 系统架构设计、功能模块划分是否合理 |
| 代码编写人员 | 根据系统架构、功能模块设计编码实现 |
| 系统测试人员 | 根据模块功能进行测试 |
|  |  |

* 1. 参考资料

N/A。

# 总体设计

* 1. 设计概述

### 2.1.1 设计约束

本部分的设计必须以《综合监控（实时数据库）软件需求规格说明书》定义的需求为依据，按照需求文档的要求设计RTDB的系统架构和定义系统的功能模块。如果设计的功能是需求文档未做要求的情况，该功能不能影响或改变需求定义的功能。

### 2.1.2 设计策略

系统架构设计按照功能和业务逻辑关系进行模块化划分，不同的模块实现不同的功能，不允许夸模块的功能。模块与模块要低耦合高内聚，模块与模块通过接口交换数据。

### 2.1.3 设计实现

RTDB系统分为两个部分：RTDB接口和RTDB服务。RTDB接口与RTDB服务是相互独立的两个部分，接口与服务之间通过TCP协议进行通信（图1）。



图1. RTDB系统数据流程图

RTDB接口设计为两个独立部分：RTDB数据接口和RTDB客户端接口。接口以库的形式提供（动态库、静态库、jar包）。

RTDB数据接口，接口实现RTDB数据的更新、命令转发接口。FEP/FEPS调用RTDB数据更新接口，更新RTDB中的设备数据；FEP/FEPS调用命令转发接口，接收RTDB下发的控制命令。

RTDB客户端接口，客户端接口RTDB数据操作（数据点的增、删、改、查等）、命令发送接口、RTDB系统状态查询、RTDB集群管理等接口。HMI、ISCS业务系统、管理工具等外部系统调用客户端接口对RTDB管理的设备点、RTDB内部状态和集群等进行查询和修改。还可以向RTDB发送对设备的控制命令，RTDB把控制命令转发给下联的FEP/FEPS。

RTDB数据接口和客户端接口需要分别在接口内部实现接口与RTDB服务的网络管理功能。网络功能包括网络连接的创建、网络连接异常检测、数据发送和数据接收，RTDB服务异常状态下接口与RTDB主备服务系统的网络自动切换。接口初始化时，接口内部负责创建接口与RTDB服务之间的TCP网络连接；网络连接建立之后，接口需要与RTDB服务之间发送心跳报文来检测连接是否异常，接口需要及时关闭异常的连接，并创建新的有效连接。接口对数据发送和接收采用采用事件触发异步处理，设置接收和发送缓存队列，发送/接收的数据先进入发送/接收缓存， 然后再对数据进行异步处理。

RTDB服务，实现数据更新、数据查询和控制命令下发三个主要的数据处理功能。

RTDB数据更新，RTDB实现与数据接口对应的网络处理，RTDB负责接收处理数据接口上传的设备数据。过程是RTDB接收数据接口的网络连接请求，对请求进行权限验证和身份识别等访问控制检查，只有通过验证的请求才被允许数据通信，验证失败的请求被直接关闭。网络连接建立之后，RTDB负责从连接接收上传的设备状态（SOE、SOC事件等）数据，然后对数据执行表达式计算、脚本计算、告警计算等一系列的分析计算处理，然后接收到的设备原始数据和计算的结果数据更新到RTDB服务内核数据区的设备数据中，如果数据被订阅的，还需要把数据发布给订阅者。RTDB采用缓存队列，实现高效的数据接收和异步处理。

RTDB数据查询，RTDB实现与客户端接口对应的网络处理，RTDB负责接收客户端发送的数据查询、更新、修改和删除请求。过程是RTDB接收客户端接口的网络连接请求，对请求进行权限验证和身份识别等访问控制检查，只有通过验证的请求才被允许数据通信，验证失败的请求被直接关闭。网络连接连理之后，RTDB从客户端接收请求指令，然后按照指令的要求，对RTDB内核数据区的设备数据进行对应的查找、删除、修改和查询操作，并把结果发送给客户端接口。

RTDB控制命令下发，RTDB实现与客户端接口对应的网络处理，RTDB负责接收客户端发送的数据查询、更新、修改和删除请求。过程是RTDB接收客户端接口的网络连接请求，对请求进行权限验证和身份识别等访问控制检查，只有通过验证的请求才被允许数据通信，验证失败的请求被直接关闭。RTDB从客户端接收到控制命令后，对命令进行权限校验，只有拥有对应权限的连接发送的命令才能够被执行。RTDB把命令转发给对应的数据接口进行处理。最后，最后返回命令的结果（命令是否被执行，是否发送成功。目标设备对命令的执行结果通过设备的状态判断）。

* 1. 设计目标

RTDB支持自动构建监控数据模型，为ISCS的业务系统、服务、HMI等子系统提供实时的监控数据和命令控制。

RTDB支持监控设备动态数据建模，即RTDB管理维护的监控设备的数据对象是通过建立数据模型动态加载生成的。在不同的工程实施时，可以根据需要为RTDB建立不同的设备监控数据模型，RTDB通过加载这些数据模型，动态生成ISCS系统需要的设备监控对象。RTDB支持运行过程中动态添加、更新、删除设备数据模型。

RTDB为ISCS的业务系统、服务、HMI等子系统通过高效的实时数据查询。RTDB支持支持大量客户端的并发查询请求，并及时对请求作出处理和响应。RTDB提供丰富的查询接口，查询单个设备的单个属性；查询单个设备的多个属性、查询单个设备的全部属性；查询多个设备的单个属性；查询多个设备的多个属性；查询指定数据类型的设备的单个属性；查询指定数据类型的设备的多个属性；查询指定数据类型的设备的全部属性等查询接口。

RTDB支持多FEP/FEPS的接入，并发处理多FEP/FEPS业务数据。RTDB对FEP/FEPS上传的从设备采集的监控数据，实时地进行统一的分析处理（包括执行表达式计算、执行脚本计算、执行告警计算等一系列的统一化的数据分析过程），生成监控需要的告警事件、统计信息等基础监控数据，并及时的把发生变化的设备状态、告警事件、统计信息等数据通知给ISCS的相关的子系统和服务。

RTDB支持表达式计算。RTDB支持对监控设备的数据点配置表达式，RTDB在对设备监控数据处理时，自动检测和执行设备的表达式处理。

RTDB支持脚本。RTDB支持LUA脚本处理，RTDB支持为单个监控设备、多个监控设备配置LUA脚本。RTDB在数据处理时，自动检测和执行脚本处理。

RTDB支持命令处理。RTDB支持对命令的权限控制、命令的执行环境检测、命令的发送、命令的执行结果检查等整个命令生命周期的管理。

RTDB通过动态监控技术，为ISCS系统提供统一、实时、高并发的数据服务。

* 1. 运行环境

OS版本。实时数据库支持的OS版本有RHEL7\_x64、CentOS7\_x64、Ubuntu\_Server14\_x64、Linux主流发行版x64（内核2.8及以上）。

硬件要求。处理器8核（及以上），单核频率3.0 GHZ（及以上），内存32 GB（及以上），磁盘1 TB（及以上），1000M以太网卡，1000M网络带宽。

* 1. 平台架构



图2.RTDB平台架构

在ISCS系统中RTDB以分布式集群模式进行构架和部署。一个车站构成一个独立的集群，中心（主和备）组成一个独立的集群，所有的车站集群和中心集群组成系统集群。

### 2.4.1 车站集群

#### 2.4.1.1集群构成

车站RTDB集群构成：数据节点和管理节点。

数据节点，即RTDB节点。RTDB节点有2个，分别作为主、备节点。只有RTDB主节点对外提供数据服务，RTDB备节点用于同步RTDB主节点的数据，不对外提供数据服务。

管理节点，即SysManager节点。管理节点有2个，分别作为SysManager主、备节点。SysManager主节点和SysManager备节点构成管理节点集群，只有SysManager主节点提供集群管理，SysManager备节点仅作为SysManager主节点的备用，不提供集群管理。

#### 2.4.1.2 集群数据

车站RTDB集群只拥有和管理本车站内的设备的监控数据。RTDB主节点和RTDB备节点拥有相同的设备数据。

RTDB主节点的数据备更新时，RTDB主节点主动同步发生变化的数据到RTDB备节点，以保证RTDB主、备节点数据的一致性。

#### 2.4.2.3 节点切换

集群节点的启动要严格按照固定的启动顺序，先启动SysManager节点，后启动RTDB节点。SysManager节点之间不做启动顺序要求，RTDB节点之间不做启动顺序要求。

##### ***2.4.2.3.1 管理节点的切换***

启动时SysManager主、备节点的裁决。先启动的SysManager作为主节点，后启动的SysManager作为备节点。

故障时SysManager主、备节点切换。SysManager备节点故障时，SysManager不做主备切换。SysManager主节点故障时，SysManager备节点自动升级为主节点。SysManager主、备节点都故障，集群管理节点失效，车站集群失效。

故障恢复时SysManager主、备节点切换。节点故障恢复后，以备节点运行；主、备节点同时发生故障恢复后，采用启动时主、备裁决机制。

##### ***2.4.2.3.2 数据节点的切换***

启动时RTDB主、备节点的裁决。RTDB节点启动后，同时向SysManager主、备节点注册。SysManager主节按照RTDB注册的先后顺序决定RTDB节点的主、备角色，先注册的RTDB节点作为主节点运行，后注册的RTDB节点作为备节点运行。

故障时RTDB主、备节点的切换。RTDB备节点故障时，不做RTDB主、备节点切换；RTDB主节点故障时，SysManager主管理节点通知RTDB备节点升级为主节点；RTDB主、备节点都故障时，车站集群失效。

故障恢复时RTDB主、备节点切换。节点故障恢复后，以备节点角色运行；RTDB主、备节点同时故障恢复后，采用启动时RTDB主、备裁决机制。

### 2.4.2 中心集群

#### 2.4.2.1 集群构成

中心RTDB集群构成：数据节点和管理节点。

数据节点，即RTDB节点。RTDB节点有4个，2个位于主中心，具有相同的优先级，且优先级高于备中心；2个位于备中心，具有相同的优先级，且优先级低于主中心。优先级高的一个作为主节点，其它三个作为备节点。只有主节点对外提供数据服务，备节点用于同步主节点的数据，不对外提供数据服务。

管理节点，即SysManager节点。SysManager节点有4个，2个位于主中心，具有相同的优先级，且优先级高于备节点；2个位于备节点，具有相同的优先级，且优先级低于主中心。优先级高的一个作为主节点，其它作为备节点。只有主节点提供集群管理，备节点仅作为主节点的备用，不提供集群管理。

#### 2.4.2.2 集群数据

中心RTDB集群只拥有和管理中心内的设备的监控数据。RTDB主、备节点拥有相同的设备数据。

主中心的RTDB与备中心的RTDB之间无数据同步。

主中心的RTDB之间数据同步。当RTDB主节点在主中心时，主中心的RTDB节点之间做数据同步。即RTDB主节点向中心RTDB备节点同步数据。主节点不在主中心时，主节点的RTDB节点之间无数据同步。

备中心的RTDB之间数据同步。当RTDB主节点在备中心时，备中心的RTDB节点之间做数据同步。即RTDB主节点向备中心的RTDB备节点同步数据。主节点不在备中心时，备中心的RTDB节点之间无数据同步。

#### 2.4.2.3 集群切换

集群节点的启动要严格按照固定的启动顺序，先启动SysManager节点，后启动RTDB节点。SysManager节点之间不做启动顺序要求，RTDB节点之间不做启动顺序要求。

##### ***2.4.2.3.1 管理节点的切换***

启动时SysManager主、备节点的裁决。优先级高且先启动的SysManager作为主节点，优先级低的SysManager作为备节点。优先级相同且启动时间相同时，随机选择主备节点。

故障时SysManager主、备节点切换。SysManager备节点故障时，SysManager不做主备切换。SysManager主节点故障时，优先级高的SysManager备节点自动升级为主节点，优先级相同时随机挑选一个备节点升级为主节点。SysManager主、备节点都故障，集群管理节点失效，车站集群失效。

故障恢复时SysManager主、备节点切换。节点故障恢复后，如果当前没有主节点，直接以主节点角色运行；否则，如果节点的优先级不高于当前主节点时，以备用节点角色运行；否则，当前主节点降为备用节点，以主节点角色运行。

##### ***2.4.2.3.2 数据节点的切换***

启动时RTDB主、备节点的裁决。RTDB节点启动后，同时向SysManager主、备节点注册。SysManager主节按照RTDB的优先级和注册次序决定RTDB节点的主、备角色，优先级高且先注册的RTDB节点作为主节点运行，优先级低且后注册的RTDB节点作为备节点运行。优先级相同且启动时间相同的，随机选择一个作为主节点，其它作为备节点。

故障时RTDB主、备节点的切换。RTDB备节点故障时，不做RTDB主、备节点切换。RTDB主节点故障时，优先级不高于当前节点时，不做主备切换；否则，以主节点角色运行，当前主节点降级为备用节点。RTDB主、备节点都故障时，车站集群失效。

故障恢复时RTDB主、备切换。节点故障恢复后，如果当前没有主节点，直接以主节点角色运行；否则，如果节点的优先级不高于当前主节点时，以备用节点角色运行；否则，当前主节点降为备用节点，以主节点角色运行。

* 1. 总体设计思路和处理流程

根据整体数据的业务处理流程进行RTDB的系统设（见图3），RTDB业务流程设计为两个部分主要部分：数据处理（上行数据流处理）和控制处理（下行数据流处理）。

### 2.5.1 数据处理

RTDB的数据处理分为数据更新和数据查询。

数据更新流程：

1. 接收数据

RTDB接收FEP/FEPS上报的设备数据（状态数据、SOC、SOE），对数据进行有效性检查（时间是否有效等），并更新到内核数据区的设备对象。

1. 表达式处理

依序执行数据对应的RTDB中数据模型对象的表达式序列。并依次向直接父亲节点递归，依序执行直接父亲数据对象配属的表达式。表达式执行完成后，要把执行结果直接更新到内核设备对象中。

1. 脚本处理

依次执行数据对应的RTDB中数据模型对应的脚本。并依次向直接父亲节点递归，依次执行直接父亲配属的脚本。脚本执行过程中，要直接把执行结果更新到RTDB内核数据区的对应数据对象。

1. 告警分析

对数据进行告警计算。依照数据模型配属的计算规则，对数据进行规则配置生成不同级别的事件。

1. 数据同步

RTDB内核数据区的数据对象的数据发送变化的，需要把最新的数据同步到备RTDB节点。

1. 历史数据处理

把接收的原始数据，表达式计算、脚本计算、告警计算的结果数据存入历史数据库。

1. 数据发布

数据处理过程中，如果数据发生了变化且该数据被订阅，则把变化的数据发布给订阅系统。

### 2.5.2 控制处理

RTDB的控制处理内容是ISCS系统的控制命令。

控制处理流程：

1. 权限校验

RTDB接收到控制命令之后，对命令进行权限校验。只有用户拥有该控制命令执行权限，如果权限校验失败，则更新命令结果状态为权限错误，然后返回。

1. 前置环境监测

命令通过权限校验后，在发送到FEP/FEPS执行之前，需要对命令的执行条件进行检查。如果检查失败，则更新命令结果状态（不具备条件或检查超时），然后返回。否则更新命令前置检查完成。

1. 命令发送

发送命令命令给FEP/FEPS，发送超时则更新命令执行失败结果为发送超时，命令返回。否则更新命令结果为发送成功。

1. 后置环境监测

命令发送成功后，执行后置环境检查。如果在超时时间内检查成功，则更新命令执行结果为命令成功执行并返回；否则，更新命令执行结果为后置检查失败并返回。



图3. RTDB数据处理流图

* 1. 制作购买或复用的分析

N/A。

* 1. 模块结构设计

按照实现的功能对RTDB系统进行模块划分和设计。RTDB系统划分为客户端接口模块、数据接口模块、角色与访问控制模块、订阅模块、控制模块、数据模型模块、代理模块、历史数据模块、同步模块、集群模块等功能模块（图4）。



图4. RTDB模块层次结构图

### 2.7.1客户端接口

接口提供数据模型对象的添加、修改、删除和查询，提供数据的订阅、控制命令等操作。

#### 2.7.1.1 模型添加接口

RTDB模型添加用于运行过程中动态的添加设备模型对象。模型只能添加到当前RTDB节点，模型添加不支持数据代理功能。

##### ***2.7.1.1.1 接口***

RTDB模型添加接口有添加单个模型对象和添加批量模型对象。

##### ***2.7.1.1.2 处理流程***

RTDB数据模型添加的处理流程如下：

步骤1：RTDB从客户端接收添加模型对象请求。

步骤2：在内核模型对象数据区检索要添加的模型对象。如果模型存在则转步骤4。

步骤3：把接收到的模型参数转换成RTDB的模型对象数据，入内核模型对象数据区。

步骤4：转步骤2，直至所有要添加的模型执行完成。

步骤5：添加结束，结果返回。

#### 2.7.1.2 模型修改

RTDB模型修改用于修改模型对象属性的默认值或当前值。当被修改的模型不在本地RTDB节点时，需要把模型修改信息重定向到模型所在的RTDB，修改操作交由模型拥有者RTDB执行。

##### ***2.7.1.2.1 接口***

RTDB模型修改接口有修改指定的模型对象的单个属性、修改模型对象的多个属性。

##### ***2.7.1.2.2 处理流程***

RTDB模型修改处理流程如下：

步骤1：RTDB从客户端接收修改模型请求。

步骤2：在内核模型对象数据区检索模型对象，模型不存在转步骤4；模型不在本地，

转步骤5。

步骤3：在模型对象中检索要修改的属性，检索成功则把数据值修改成指定的值。

步骤4：重复步骤2，直至请求的模型都修改完成，转步骤6。

步骤5：重定向到实际管理模型的RTDB，执行模型修改，转步骤2。

步骤6：修改接收，结果返回。

#### 2.7.1.3 模型查询

RTDB模型查询用于查询模型的对象的属性。当被查询的模型不在本地RTDB时，需要把查询代理到模型所在的RTDB。拥有模型的RTDB执行查询操作，然后返回查询结果给转发代理的RTDB，最终完成查询操作。

##### ***2.7.1.3.1 接口***

RTDB模型查询有单个模型单个属性查询、单个模型多个属性查询、单个模型查询；多个模型的单个属性查询、多个模型的多个属性查询、多个模型查询；指定模型类型的模型的单个属性查询、指定模型类型的模型的多个属性查询、指定模型类型的模型查询；

##### ***2.7.2.3.2 处理流程***

RTDB模型查询处理流程如下：

步骤1：RTDB从客户端接收模型查询请求。

步骤2：在内核模型对象数据区检索模型对象，模型不存在转步骤4；模型不在本地，

转步骤5。

步骤3：在模型中检索指定的属性，并生检索结果，转步骤2。

步骤4：重复步骤2，直至请求的模型都检索完成。转步骤6。

步骤5：重定向到模型所在的RTDB，执行模型查询操作，并返回查询结果。转步骤2。

步骤6：查询结束，结果返回。

#### 2.7.1.4 数据订阅

RTDB提供模型对象的订阅，当被订阅的数据发生变化时，RTDB把变化的数据发布给数据订阅者。

被订阅的模型不在RTDB本地时，RTDB把订阅代理到模型所在的RTDB。模型所在的RTDB把发生变化的订阅数据发布给执行代理的RTDB，再由RTDB把发布的数据最终发布给订阅客户端。

##### ***2.7.1.4.1 接口***

RTDB订阅接口有订阅单个模型的单个属性、订阅单个模型的多个属性、订阅单个模型；订阅多个模型的单个属性、订阅多个模型的多个属性、订阅多个模型；订阅指定模型类型的模型的单个属性、订阅指定模型类型的多个属性、订阅指定模型类型的模型；订阅分组中全部模型的单个属性、订阅分组中全部模型的多个属性、订阅分组中的全部模型。

##### ***2.7.1.4.2 处理流程***

RTDB订阅处理流程：

步骤1：RTDB接收客户端的模型订阅请求。

步骤2：在内核模型对象数据区检索模型对象，模型不存在转步4；模型不在本地转5。

步骤3：在模型中检索订阅的属性，属性存在则执行订阅。

步骤4：转步骤2，直至所有的模型都订阅完成，转步骤6。

步骤5：把订阅代理到模型所在的RTDB，由模型所在的RTDB执行模型订阅。转步骤2。

步骤6：订阅结束，结果返回。

RTDB发布处理流程：

步骤1：RTDB检测模型数据发生变化，生成模型发布数据。

步骤2：RTDB把发布数据发布给模型订阅者。订阅者连接失效，订阅数据丢弃。

步骤3：发布结束。

#### 2.7.1.5 控制命令

RTDB提供控制命令接口，用于接收客户端发送的设备控制命令。客户端的控制命令必须在RTDB中有对应的命令模型。

##### ***2.7.1.5.1 接口***

RTDB命令有给一个设备发送控制命令。

##### ***2.7.1.5.2 处理流程***

RTDB命令处理流程：

步骤1：RTDB从客户端接收控制命令。

步骤2：在内核模型对象数据区检索命令对象，模型不存在转步骤5；不在本地转4。

步骤3：执行名控制过程，转步骤5。

步骤4：把命令代理到实际用户模型对象的RTDB，由用于模型的RTDB执行命令。

步骤5：命令结果返回。

### 2.7.2数据接口

RTDB的数据接口有设备数据接口和控制命令接口。

#### 2.7.2.1 设备数据接口

设备数据接口用于接收写入RTDB的设备数据，如设备状态、设备SOE、设备SOC等。

##### ***2.7.2.1.1 接口***

设备数据接口有设备数据写RTDB接口。

##### ***2.7.2.1.2 处理流程***

设备数据写RTDB流程：

步骤1：RTDB从数据接口接收设备数据。

步骤2：设备数据解析成RTDB设备记录，并存入数据接口缓存队列等待处理。

步骤3：结束。

#### 2.7.2.2 控制命令接口

控制命令接口用于发送设备控制命令，这些命令是HMI、业务系统等发送给RTDB的。

##### ***2.7.2.2.1 接口***

控制命令接口有发送控制命令。

##### ***2.7.2.2.2 处理流程***

控制命令接口处理流程：

步骤1：RTDB命令处理模块生成命令消息。

步骤2：RTDB调用控制命令发送接口发送控制命令。

步骤3：命令发送状态结果返回，控制命令发送接收。

### 2.7.3角色和访问控制

RTDB为了系统和数据的安全，需要对客户端的连接进行授权检查、数据访问进行权限控制。只有授权的用户可以连接到RTDB，只有拥有数据操作权限的用户才可以进行授权的数据操作。

RTDB的权限控制信息存储在ISCS系统的配置数据库（关系数据），RTDB需要从ISCS系统的系统配置管理系统加载（图5）。



图5 RTDB配置与配置管理

#### 2.7.3.1 授权数据

授权数据由授权验证信息和权限控制信息组成。

##### ***2.7.3.1.1 权限***

授权验证信息用于对客户端的TCP连接请求进行控制，只有授权的客户端才被允许连接到RTDB系统，没有授权的客户端不允许连接到RTDB。授权验证信息内容由ID和IP组构成。

权限控制信息用于对客户端的数据访问操作、RTDB系统控制操作、设备命令操作进行操作权限验证，RTDB只允许客户端进行授权的操作，RTDB禁止客户端执行没有授权的操作。权限控制信息内容由权限控制码组构成。

##### ***2.7.3.1.2 数据***

客户端ID、客户端IP和权限控制码的数据格式分别见表1、表2。

表1 授权验证信息数据

|  |  |  |
| --- | --- | --- |
| 内容 | 数据类型 | 描述 |
| ID | String | ID是最大长度为20个字节的字符串。  例如“MAQUN\_HMI\_001”用来表示马群车站001号HMI。 |
| IP | String | IP是最大长度为40个字节的字符串。  IPV4格式X.X.X.X，其中每个X表示地址中的8比特，用十进制数字表示。例如192.168.2.100  IPV6格式X:X:X:X:X:X:X:X，其中每个X表示地址中的16比特，用4个十六进制字符表示。例如0001:0002:0003:0004:0005:0006:0007:0008 |

表2 权限码

|  |  |  |
| --- | --- | --- |
| 权限编码 | 数据类型 | 功能 |
| 0x10000001 | Integer | 查询权限，允许查询RTDB中设备的模型数据。 |
| 0x10000002 | Integer | 查询权限，允许查询RTDB的系统状态。 |
| 0x10000003 | Integer | 查询权限，允许查询RTDB的集群状态。 |
| 0x20000001 | Integer | 订阅权限，允许订阅RTDB中设备的模型数据。 |
| 0x30000001 | Integer | 修改权限，允许向RTDB添加、修改、删除设备模型。 |
| 0x30000002 | Integer | 修改权限，允许修改RTDB系统运行参数，启停功能模块。 |
| 0x30000003 | Integer | 修改权限，允许修改RTDB的集群运行数据。 |
| 0x40000001 | Integer | 命令权限，云向RTDB发送系统控制命令、设备控制命令。 |

#### 2.7.3.2 TCP通信

RTDB客户端与RTDB之间的TCP通信分为三个过程，创建TCP连接链路、认证TCP连接和TCP数据通信。

创建TCP连接链路。RTDB启动后，监听TCP服务端口等待客户端发起连接请求。有客户端向RTDB发起连接请求后，RTDB为请求客户端建立一个TCP连接链路。

认证TCP连接。RTDB为客户端创建TCP连接链路后，等待客户端发送授权认证信息，来认证客户端是否由权限连接到RTDB。验证失败的客户端将被拒绝连接RTDB。

TCP数据通信。客户端的TCP连接通过授权认证之后，才能用于RTDB与客户端之间的数据通信。在客户端的连接授权认证没有完成之前，RTDB不会处理客户端发送的任何非验证数据；RTDB也不会向客户端发送任何根验证结果无关的数据。

#### 2.7.3.3 连接认证

RTDB对所有客户端的TCP连接请求进行授权认证，根据认证结果来决定是否接受客户端的连接请求。

RTDB的客户端授权认证过程见图6。首先，RTDB监听客户端的TCP连接请求。当有客户端向RTDB发起连接请求时，RTDB为客户端创建一个TCP连接链路。然后，RTDB等待客户端发送授权认证（在认证期间，RTDB仅处理客户端的授权认证信息）。授权认证失败或认证超时，RTDB关闭当前TCP连接，客户端的本次请求失败。客户端在超时时间内通过授权认证，客户端连接正式确立。



图6 RTDB连接认证流程

RTDB对客户端发送的ID和IP地址进行校验。如果客户端的ID和IP在RTDB本地的授权数据允许列表中，则认为客户端有权限，否则没有权限。

#### 2.7.3.4 权限认证

RTDB对客户端的操作请求进行授权检查，只有客户端请求的操作拥有授权，RTDB才会执行客户端的请求操作。

RTDB的授权过程。首先，RTDB从接收客户端的操作请求。然后，RTDB根据请求类型，从客户端对应的授权对象中查找相关授权码，如果查找到对应的操作码则RTDB执行客户端的操作请求；否则，RTDB拒绝客户端请求。

### 2.7.4数据订阅

数据订阅用于定义RTDB管理的设备模型对象数据，当被订阅的设备模型对象的值发生变化时，RTDB则把变化的模型数据通知给模型订阅者。RTDB仅提供对设备模型对象的订阅，设备模型以为的数据只能通过查询获取。

数据订阅模块由数据订阅子模块、数据发布子模块、连接管理子模块和代理子模块构成（见图7）。



图7 数据订阅结构图

#### 2.7.4.1 连接管理

连接管理模块用于管理维护订阅者的网络连接。订阅管理的连接有客户端订阅连接、RTDB订阅连接、RTDB代理订阅连接（见图8）。



图8 订阅连接管理处理流程

##### ***2.7.4.1.1 连接检查***

RTDB对订阅连接进行可用性检查，并及时处理不可用的订阅连接。可用性检查的方法是向订阅连接发送数据包，根据数据包是否发送成功来裁定连接是否可用。如果数据包发送成功，连接可用；如果发送失败，则连接不可用。检查用的数据包有两种发布的数据和心跳。

发布数据的可用性检查。当订阅连接通道有订阅的数据要发布时，RTDB向订阅连接通路发送发布数据。如果发布的数据成功发送则认为连接可用，否则连接不可用。

心跳的可用性检查。当订阅连接通道没有需要发布的数据时，则通过定时发送心跳数据包来检测订阅连接是否可用。如果发布的心跳包成功发送则认为链接可用，否则连接不可用。

##### ***2.7.4.1.2 失效处理***

当检测到订阅的连接不可用时（网络中断、网络错等故障发生时），订阅模块需要对不可用的订阅连接及时进行处理。订阅模块根据订阅来源，决定是直接删除订阅，还是重新发起订阅。

RTDB客户端发起的订阅。当检测到订阅连接不可用时，订阅模块需要关闭不可用的订阅连接并删除本订阅。如果本订阅有关联的代理订阅，订阅模块需要关闭代理订阅的网络连接并删除代理订阅。

RTDB发起的订阅。当检测到订阅连接不可用时，订阅模块需要关闭订阅连接并重新向目标RTDB发起订阅，直至订阅成功。

RTDB发起的代理订阅（代理订阅指客户端向RTDB发起订阅，但是要订阅的数据不在RTDB本地时，RTDB通过代理向数据所在的RTDB发起的订阅）。当检测到订阅连接不可用时，如果客户端的订阅有效时，订阅模块需要重新发起代理订阅，直至订阅成功。如果客户端的订阅也无效时，订阅模块需要关闭客户端的订阅连接并删除订阅，同时关闭客户端订阅对应的代理订阅的网络连接并删除订阅。

#### 2.7.4.2 订阅管理

订阅管理用于接收和处理对RTDB管理的设备模型对象的订阅（订阅处理流程见图9）。

RTDB启动后，RTDB监听订阅端口，等待订阅者的请求。当有订阅请求到来时，RTDB为订阅者执行授权认证（[参见2.7.3](#_2.7.3角色和访问控制)）。如果授权认证失败，RTDB拒绝订阅者订阅任何数据，订阅失败。如果通过授权认证，RTDB从订阅者接收订阅数据。RTDB在内核设备模型对象数据区检索要订阅的模型对象，如果对象数据在RTDB本地时，直接执行模型的订阅；如果对象不再RTDB本地（即本地只有模型的代理信息，没有数据信息）时，RTDB执行代理订阅。



图9 订阅处理流程

#### 2.7.4.3 发布管理

订阅发布管理用于管理发布数据（见图10）。

订阅发布处理流程为如下：如果有代理订阅，则监听代理订阅的网络连接，从代理订阅接收代理发布的数据，并把接收到的订阅数据加入本地发布缓存队列。发布管理模块定期扫描发布缓冲队列，如果缓冲队列不为空，则把缓冲队列的数据通过订阅连接通道发布给订阅者。其中，从代理接收代理发布的数据与扫描本地发布缓冲队列发布数据为异步处理。



图10 数据发布流程图

#### 2.7.4.4 代理

数据订阅支持代理功能，即代理订阅和代理发布。

代理订阅。RTDB接收模型数据订阅，如果被订阅的模型在本RTDB中是代理模型，由RTDB向代理指向的RTDB发起代理模型的订阅。

代理发布。如果RTDB有向别的RTDB发起的代理订阅，RTDB需要监听代理订阅通道，从订阅通道接收代理订阅发布的数据，然后把数据发布给模型订阅者。

代理模型指模型的数据存放在远端RTDB，本地RTDB只存存放模型的代理信息，即模型数据所在的RTDB的地址信息。

### 2.7.5控制

本模块用于处理ISCS系统的设备命令，对命令的接收、条件检查、发送等整个命令周期进行管理和控制。并提供命令的执行阶段和状态的查询。

RTDB提供命令模板功能，通过配置命令模板实现ISCS系统复杂的控制命令。命令模板的基本执行单元是DO、AO、SO这三种RTDB输出数据点，模板就是通过顺次条件执行这些基本输出点对应的设备指令实现复杂的组合命令控制（如程控、顺控）。

#### 2.7.5.1 控制模板

#### 2.7.5.2 基本控制

基本控制指ISCS中的单指令控制控制对象是DO、AO、SO这三类输出类型的数据点。

##### ***2.7.5.2.1 模拟输出控制***

模拟输出控制用于向设备发送遥调信号，RTDB中用于控制遥调的模型为AIO（见图11 AIO控制流程）。

模拟输出控制流程：RTDB接收控制命令，查找模拟输出指向的模拟输出数据点AIO。如果AIO不存在，则RTDB不支持本类型的模拟输出控制，控制失败返回。检查AIO对应的设备模型的控制禁止状态，如果设备禁止控制，则更新AIO的执行状态为禁止控制并返回。检查命令初始执行条件，检查失败或检查超时，则更新命令状态为初始条件不满足或超时并返回。检查模拟输出控制的命令值，如果值超出有效范围，则更新AIO的执行状态为值超限并返回。发送控制给FEP，发送失败或超时，则更新AIO的执行状态发送失败或发送超时并返回。检查命令返回状态，检查失败或超时，更新AIO的执行状态结果失败或超时并返回；否则更新AIO的执行状态为成功并返回。



图11 AIO控制

##### ***2.7.5.2.2 离散输出控制***

离散输出控制用于向设备发送遥控信号，RTDB中用于控制遥控的模型DAIO（见图11 DIO控制流程）。

离散输出控制流程：RTDB接收控制命令，查找模拟输出指向的离散输出数据点DIO。如果DIO不存在，则RTDB不支持本离散类型控制，控制失败返回。检查DIO对应的设备模型的控制禁止状态，如果设备禁止控制，则更新DIO的执行状态为禁止控制并返回。检查命令初始执行条件，检查失败或检查超时，则更新命令状态为初始条件不满足或超时并返回。发送控制给FEP，发送失败或超时，则更新DIO的执行状态发送失败或发送超时并返回。检查命令返回状态，检查失败或超时，更新DIO的执行状态结果失败或超时并返回；否则更新DIO的执行状态为成功并返回。



图12 DIO控制

##### ***2.7.5.2.3 模拟输出代理控制***

模拟输出代理控制实现模拟输出控制相同的功能，唯一的不同是它作用于RTDB的AIO的代理点。

RTDB接收模拟输出控制命令，RTDB检索模拟输出控制命令指定的AIO点。AIO点不存在，则命令失败返回。如果AIO的数据在本地RTDB，则执行模拟输出控制流程；否则AIO是代理数据点（即AIO的数据存放在远端RTDB，本地AIO只保留AIO数据所在RTDB的代理信息），则RTDB通过代理把模拟输出控制指令发送给实际拥有AIO数据的目标RTDB去执行。本地RTDB监视目标RTDB的AIO执行过程，直至目标RTDB完成AIO的执行，本地RTDB从目标RTDB或AIO的执行结果并返回。

##### ***2.7.5.2.4 离线输出代理控制***

离线输出代理控制，参照模拟输出代理控制。

#### 2.7.5.3 组控制

顺序执行一组控制命令，如果某个命令执行条件不满足、或命令执行失败，则组控失败返回。

### 2.7.6数据模型

RTDB数据模型包括：数字量模型、模拟量模型、累积量模型、告警模型、数字量控制模型、模拟量控制模型等。

RTDB从配置库加载数据模型，然后生成RTDB内部的模型对象。

#### 2.7.6.1 数字量点

数字量点是由数字量输入点和数字量输出点组合成的点。如果一个数字量点是控制点，则该数字量点既是输入点又是输出点；否则，该数字量点仅是输入点。

数字量输入点支持值有多路来源，值的来源有采集、强制、预估。根据系统需要设置数字量的值为HMI的强制值、FES的上报值或系统的预估值。

数字量输入点支持值多分量功能。数字量输入点是由多个位的信号构成时，需要把多个分量的值进行合成处理。

数字量输出点，支持控制分组和联锁关系。具体参见《RTDB数据建模表结构设计》。

#### 2.7.6.2 模拟量点

模拟量点是由模拟量输入点和模拟量输出点组合成的点。如果一个模拟量点是控制点，则该模拟量点既是输入点又是输出点；否则，该模拟量点仅是输入点。

模拟量输入点支持值有多路来源，值的来源有采集、强制、预估。根据系统需要设置数字量的值为HMI的强制值、FES的上报值或系统的预估值。

模拟量输出点，支持控制分组和联锁关系。具体参见《RTDB数据建模表结构设计》。

### 2.7.7代理

RTDB的数据代理功能有：查询代理、订阅代理、表达式代理、脚本代理、控制代理。

查询代理，客户端查询的数据不在本地RTDB时，由RTDB把查询代理到目标RTDB执行，并返回结果。

订阅代理，被代理的数据不在本地RTDB时，由RTDB把订阅代理到目标RTDB，从目标RTDB接收发布消息。

表达式代理，表达式中的数据不在本地RTDB时，由RTDB把查询代理到目标RTDB执行，并返回结果。

脚本代理，脚本使用的数据不在本地RTDB时，由RTDB把查询代理到目标RTDB执行，并返回结果。

控制代理，备控制的点不在本地RTDB时，由RTDB把控制代理到目标RTDB执行，并返回结果。

代理的详细内容参照各模块设计。

### 2.7.8历史数据

RTDB把FES上传的数字量、模拟量、SOE等数据保存历史数据库。

### 2.7.9同步

RTDB根据需要把FES上传数据、内部数据（命令执行状态、HMI设置的属性值等）等同步到集群的对等RTDB节点。

#### 2.7.9.1 同步机制

RTDB对FES上传数据、内部数据分别进行同步控制。通过配置同步策略精确的控制需要同步的数据。

FES上传数据同步。RTDB首先检查是否需要同步该类型数据，如果需要同步，则根据策略配置的目标RTDB节点信息，把数据加入同步缓存队列等待数据同步。

内部数据同步。RTDB检测到内部数据发生变化后，首先检查是否允许同步该类型数据，如果不允许同步则同步结束；否则把数据加入同步缓存队列等待同步。

### 2.7.10集群

RTDB维护配置的集群节点信息（对等集群节点）。详情见《集群和系统管理》。

### 2.7.11表达式

RTDB的表达式处理分为三个过程：表达式的解析、表达式的计算、表达式结果返回。

表达式解析。RTDB把要执行的表达式分解成运算符、常量参数、变量参数、括号、流程控制关键字等表达式项。需要把表达式中的数据模型参数转换成具体的数据。

表达式计算。RTDB把解析出来的表达式项进行算数运算和逻辑运算。并生成运算的算数结果或逻辑结果。

表达式结果返回。RTDB完成表达式计算后，把计算的结果返回给表达式调用者，完成表达式处理。

### 2.7.12脚本

RTDB使用LUA脚本。

RTDB的脚本执行过程是加载脚本、解析脚本、执行脚本、返回脚本结果。

其中解析脚本，需要把脚本中指定的RTDB数据点模型参数转换成具体数据参数。

### 2.7.13告警

RTDB告警处理数字量告警、模拟量告警、FES上传告警。

### 2.7.14协议

RTDB与FES的数据协议。参照《RTDB与FES接口实现》。

用一览表及框图的形式说明本系统的系统元素（各层模块、子程序、公用程序等）的划分，扼要说明每个系统元素的标识符和功能，分层次地给出各元素之间的控制与被控制关系。

具体描述每一功能模块的结构，<对模块1等模块中各子模块之间用功能逻辑图加以说明，并对子模块之间的关系加以说明>

功能逻辑图例样：



模块的名称定义和需求规格说明书中的功能需求描述中的功能模块名一一对应。可分子目录列分别<对模块1、模块2等模块的功能进行结构，技术实现，逻辑处理说明>。

用一览表及框图的形式说明模块性质分类，列出哪些模块需要自主开发，哪些模块可以复用，哪些模块需要采购等。一览表样例如下：

|  |  |
| --- | --- |
| **模块名称** | **模块实现类型** |
| 模块一 | 自主开发 |
| 模块二 | 复用 |
| 模块三 | 采购 |
| …… | …… |

* 1. 功能需求与程序模块的关系（可选）

本条用一张如下的矩阵图说明各项功能需求的实现同各块程序模块的分配关系，本表可用需求跟踪距阵替代：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 程序1 | 程序2 | …… | 程序m |
| 功能需求1 | √ |  |  |  |
| 功能需求2 |  | √ |  |  |
| …… |  |  |  |  |
| 功能需求n |  | √ |  | √ |

* 1. 尚未解决的问题

说明在概要设计过程中尚未解决而设计者认为在系统完成之前必须解决的各个问题。

# 接口设计

* 1. 用户接口

说明与用户接口的输入输出关系，以及将向用户接口提供的命令和它们的语法结构，以及软件的回答信息。

* 1. 外部接口

说明本系统同外界的所有接口的安排包括软件与硬件之间的接口、本系统与各支持软件之间的接口关系、协议要求等。

* 1. 内部接口

说明本系统之内的各个系统元素（各层模块、子程序、公用程序等）之间的接口的安排，包括设计用关系、输入输出要求、语法结构等。

# 界面总体设计

说明界面总体布局和风格设计。

# 数据结构设计

可以在本文说明也可以单独使用数据库设计说明书描述

* 1. 设计原则

给出系统数据库的设计原则。

* 1. 数据库环境说明

简单介绍一些数据库直接有关的支持软件，如数据库管理系统、存储定位程序和用于装入、生成、修改、更新数据库的程序等。说明这些软件的名称、版本号和主要功能特性。

* 1. 数据库命名规则

联系用途，详细说明用于唯一地标识该数据库的代码、名称或标识符，附加的描述性信息亦要给出。

* 1. 逻辑结构

数据库设计人员根据需求文档，利用数据建模技术来描述逻辑数据库结构。要求使用PowerDesigner或Visio创建数据库PDM模型。此处只需列出PDM模型名称。

* 1. 物理存储

描述整个逻辑数据模型是如何被转换为数据文件（物理模式）。文件结构类型在这里应清楚的体现。

* 1. 数据备份和恢复

描述数据库的备份和恢复策略。

# 系统出错处理设计[可选]

* 1. 出错信息

用一览表的方式说明出错的类型，以及每种可能的出错或故障情况出现时，系统输出信息的形式、含意及处理方法。例如：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **错误分类** | **子项及其编码** | **错误名称** | **错误代码** | **备注** |
| 数据库错误 | 连接 | 连接超时 | 100001001 |  |
| 连接断开 | 100001002 |  |
| 数据库本身错误代码 | 数据库本身错误代码 | 100002+数据库错误代码 |  |
| TCP连接错误 | 连接 | 连接超时 | 101001001 |  |
| 连接断开 | 101001002 |  |
| 其它TCP连接错误(socket自身错误代码) |  | 101002+ socket错误代码 |  |
| 配置信息错误 | 未配置输入参数 |  | 102001 |  |
| 未配置输出参数 |  | 102002 |  |

* 1. 补救措施

说明故障出现后可能采取的变通措施，包括：

a．后备技术 说明准备采用的后图示技术，当原始系统数据万一丢失时启用的副本的建立和启动的技术，例如周期性地把磁盘信息记录到磁带上去就是对于磁盘媒体的一种后备技术；

b．降效技术 说明准备采用的后备技术，使用另一个效率稍低的系统或方法来求得所需结果的某些部分，例如一个自动系统的降效技术可以是手工操作和数据的人工记录；

c.恢复及再启动技术 说明将使用的恢复再启动技术，使软件从故障点恢复执行或使软件从头开始重新运行的方法。

* 1. 系统维护设计

说明为了系统维护的方便而在程序内部设计中作出安排，包括在程序中专门安排用于系统的检查与维护的检测点和专用模块。

# 系统安全设计

* 1. 数据传输安全性设计

说明在数据通信和传输过程中安全性设计。

* 1. 应用系统安全性设计

说明在访问应用系统过程中用户以及访问权限、操作等安全性设计。

* 1. 数据存储安全性设计

说明在数据和文件在存储过程中的安全性设计。

# 系统部署（可选）

给出系统部署方案，尽量使用图表的形式，并辅以必要的文字说明。