四方通讯模块软件概要设计

概述

目的

> 本概要设计文档旨在在需求分析基础上，对系统的整体架构、模块划分、功能分配、数据流及关键技术方案进行规划与说明。通过本设计，明确系统结构与各子模块的功能边界，为详细设计、编码实现和后续测试提供依据，确保软件开发过程的规范性、可控性与高效性。

设计原则

1. **模块化分离**

   将系统拆分为功能独立的模块，增强可维护性与扩展性。

2. **低耦合，高内聚**

   降低模块间依赖关系，确保每个模块职责单一、内部逻辑紧凑。

3. **开放封闭原则**

   对扩展开放，对修改封闭，鼓励通过接口扩展功能而非更改已有代码。

4. **单一职责原则**

   每个类或模块只负责一个功能，避免功能杂糅。

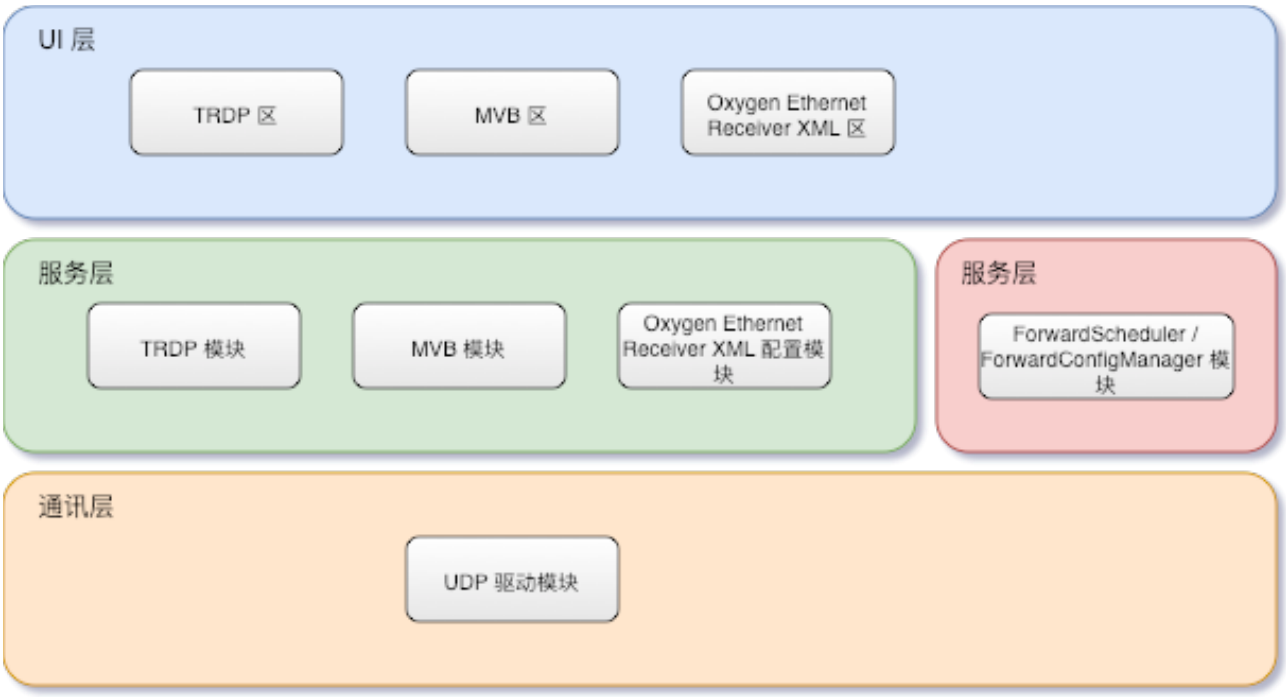
5. **容错与鲁棒性设计**

   系统需对错误输入和异常情况有良好的处理与恢复机制。

术语说明

术语	全称 / 英文	说明
TRDP	Train Real-time Data Protocol	一种用于铁路车辆通信的实时数据协议，支持发布/订阅与请求/响应机制，常用于车载系统（如 TCMS）中的设备间数据交换。由 TCN 协会制定，适用于以太网通信环境。
MVB	Multifunction Vehicle Bus	多功能车辆总线，是铁路行业的标准通信总线，用于连接列车控制子系统。其通信稳定性高，适用于低速控制信息传输，是 TCN（Train Communication Network）的一部分。
Oxygen	DEWETRON Oxygen Software	DEWETRON 公司开发的数据采集与分析软件，支持多通道同步采集、分析和报告生成，广泛应用于测试测量行业。支持 SCPI 命令远程控制。
Ethernet	-	局域网通信标准，常用于工业设备间高速数据传输。在 TRDP、Oxygen 等系统中作为底层通信载体，支持 TCP/IP 或 UDP 通信方式。
Receiver	-	在数据通信中表示“接收方”，通常指接收数据包、指令或采样信号的终端设备或模块。
XML	eXtensible Markup Language	可扩展标记语言，用于结构化数据的存储与交换。常用于配置文件、数据报文、接口协议定义等，在 TRDP 报文描述或 Oxygen 配置中可能作为格式使用。
UDP	User Datagram Protocol（用户数据报协议）	一种无连接的网络传输层协议，提供简单高效的数据传输方式，不保证数据包顺序和可靠送达，适用于对实时性要求高但容错能力强的应用场景，如视频流、传感器数据、TRDP 协议、工业以太网通信等。

系统设计



成员名称	说明
UI 层	对服务层的相应模块的图形化配置

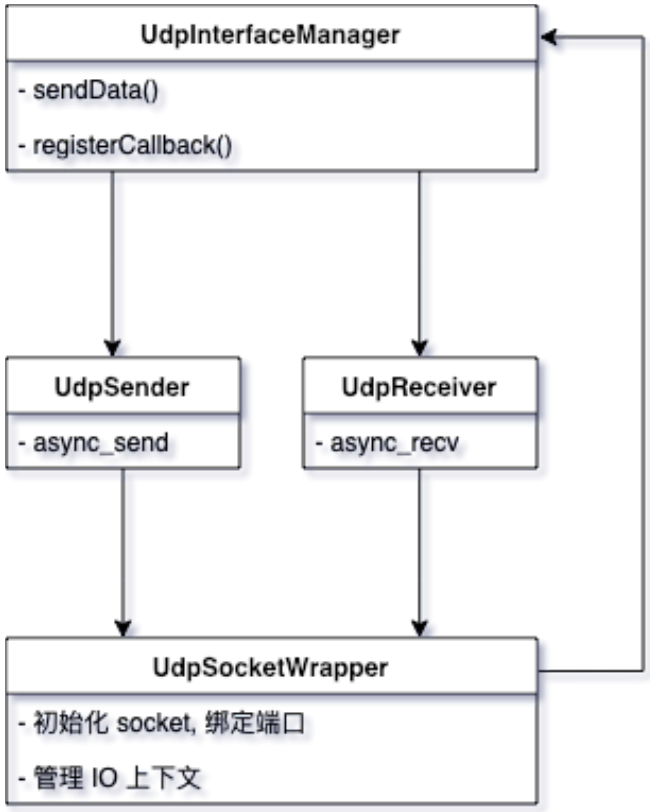
模块结构

UDP 驱动模块设计

1. 模块概述

UDP 驱动模块作为通信底层核心，负责统一封装基于 UDP 协议的数据收发能力，为上层 TRDP 模块和 MVB 模块提供异步、线程安全的收发接口。模块底层基于 **Asio** 实现，支持多端口绑定、异步 IO 与线程调度。

2. 模块结构图



3. 功能职责

子模块	职责说明
UdpInterfaceManager	统一暴露收发接口，供上层模块注册回调、发送数据
UdpSender	实现异步数据发送（async_send_to）
UdpReceiver	实现异步数据接收（async_receive_from）
UdpSocketWrapper	封装 Asio 的 socket 创建、绑定、关闭等操作，管理 io_context 与线程

4. 与上层接口说明

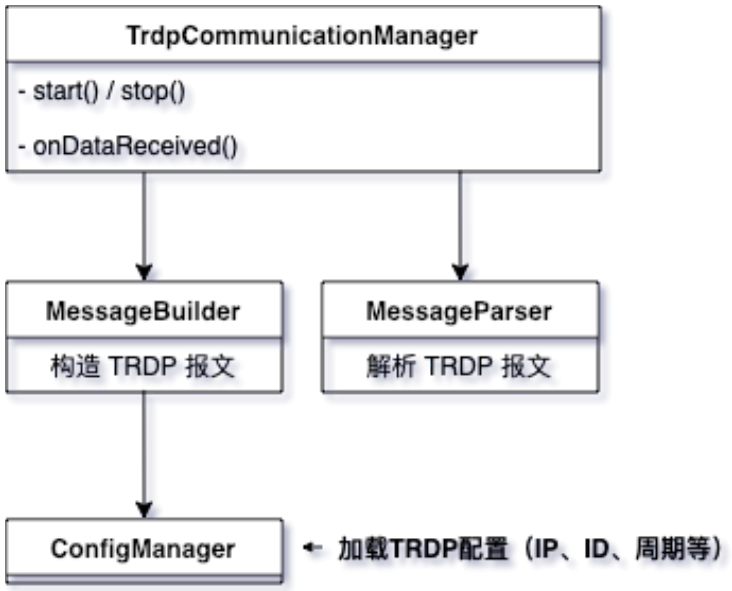
接口名	描述	参数	实现模块
initialize(port)	初始化并绑定端口	int port	UdpSocketWrapper
sendData(ip, port, data)	异步发送数据	string, uint16, QByteArray	UdpSender
registerCallback(func)	注册接收数据回调	std::function	UdpReceiver

TRDP 模块设计

1. 模块概述

抽象 TRDP 协议，支持 ComId 配置、数据抽象和解析，支持按周期转发到目标 IP + 端口

2. 模块结构图



3. 功能职责

子模块	职责说明
TrdpConfigManager	读取 TRDP 报文配置（目标 IP、周期、报文类型）
TrdpMessageBuilder	将业务数据打包成 TRDP 报文（含报文头、ID、时间戳等）
TrdpMessageParser	将收到的数据还原为 TRDP 报文结构体
TrdpCommunicationManager	调用 UDP 接口，负责发送/接收调度与业务分发

4. 报文结构说明（示意）

```
struct TrdpHeader {
    uint8_t TID;
    uint8_t netifID;
    uint16_t dataLength;
    uint32_t comId;
    uint32_t sequenceCounter;
    uint32_t srcIpAddress;
    std::vector<uint8_t> userData;
};
```

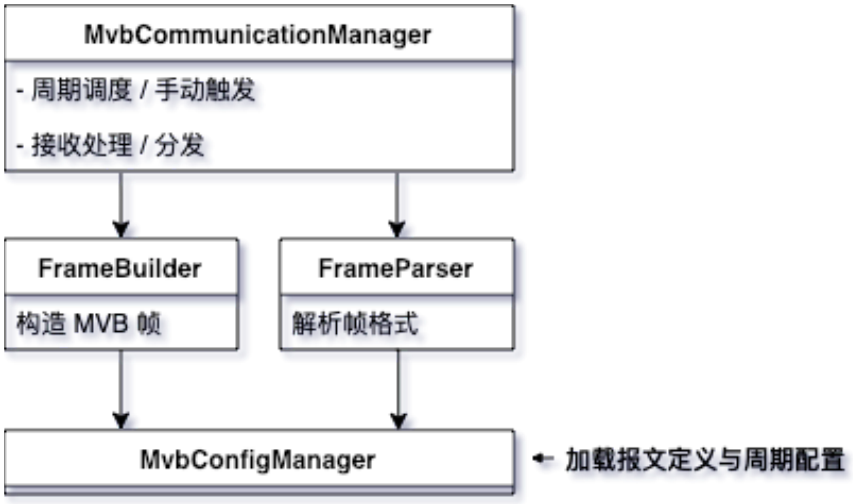
字节偏移量	字段名称	描述	取值
0	TID	消息 ID	必须为 `0x31`
1	netifID	接收以太网接口编号	• 1: TRDP 网口 1 • 2: TRDP 网口 2（如果有）
2	dataLength	应用数据长度高字节	1 ~ 1432，大端模式
3	dataLength	应用数据长度低字节	
4	comId	ComId 高字节	
5		ComId 次高字节	
6		ComId 次低字节	
7		ComId 低字节	
8	sequenceCounter	序号计数器高字节	有效数据每次 +1
9		序号计数器次高字节	
10		序号计数器次低字节	
11		序号计数器低字节	
12	srcIpAddress	TRDP 源 IP 高字节	
13		TRDP 源 IP 次高字节	
14		TRDP 源 IP 次低字节	
15		TRDP 源 IP 低字节	
17 ... N	userData	应用数据字节 0 ~ N	

MVB 模块设计

1. 模块概述

抽象 MVB 通信协议，给实际设备或虚拟设备处理 PD 通道，支持按周期转发

2. 模块结构图



3. 功能职责

子模块	职责说明
MvbConfigManager	加载 MVB 报文定义（地址、数据长度、周期等）
MvbFrameBuilder	构造符合 MVB 协议规范的数据帧
MvbFrameParser	将 UDP 接收数据还原为 MVB 协议结构体
MvbCommunicationManager	统一管理与 UDP 的交互，周期性触发发送

4. 报文结构说明（示意）

```
struct MvbFrame {
    uint8_t TID;
    uint8_t fcode;
    uint16_t pdPort;
    std::vector<uint8_t> pdData;
};
```

字节偏移量	字段名称	描述	取值
0	TID	消息 ID	必须为 `0x21`
1	fcode	PD 端口 F_code	0、1、2、3、4 分别表示 2、4、8、16、32 字节
2	pdPort	PD 端口号高字节	0 ~ 4095，大端模式
3		PD 端口号低字节	
4 … N	pdData	PD 数据字节 0 ~ N	可变长数据，N 可为 2、4、8、16、32，依据 PD 端口大小

Oxygen Ethernet Receiver XML 模块

Ethernet Receiver使用XML配置文件定义接收和解析网络数据包的方法，提供自定义通道定义、同步及拓扑结构功能。

查看 《Oxygen Ethernet Receiver XML 配置手册 (开发人员指南) 》

ForwardScheduler / ForwardConfigManager 模块设计

1. 模块概述

ForwardScheduler 模块与 ForwardConfigManager 模块共同构成系统中 TRDP 与 MVB 模块的“周期转发”功能核心。其作用是根据配置文件（或 Oxygen 工具导入的参数）自动创建并调度数据转发任务，将指定 comId 或 pdPort 的数据周期性地转发至目标 IP 和端口。该模块独立于 TRDP/MVB 协议本身，仅依赖其上层报文解析和数据缓存结果。

2. 模块结构图

3. ForwardScheduler 功能说明

函数 / 组件名	说明
`addOrUpdateTask(id, period, targetIP, targetPort)`	创建或更新转发任务，若已存在则刷新周期
`removeTask(id)`	删除对应 comId 或 pdPort 的转发任务
`tick()`	被定时器定期调用，从数据缓存中取数据并触发发送
`bindSender(std::function)`	与 UDP 模块的 sendData 函数建立连接

> 技术实现：基于 `asio::steady_timer` 构建任务队列；使用 ID + hash 做索引。

4. ForwardConfigManager 功能说明

函数名	说明
`loadFromXml(const QString &path)`	加载并解析配置项
`getTarget(comId or pdPort)`	返回目标 IP、端口、周期等转发参数
`reload()`	支持运行时重新加载配置文件（热更新）

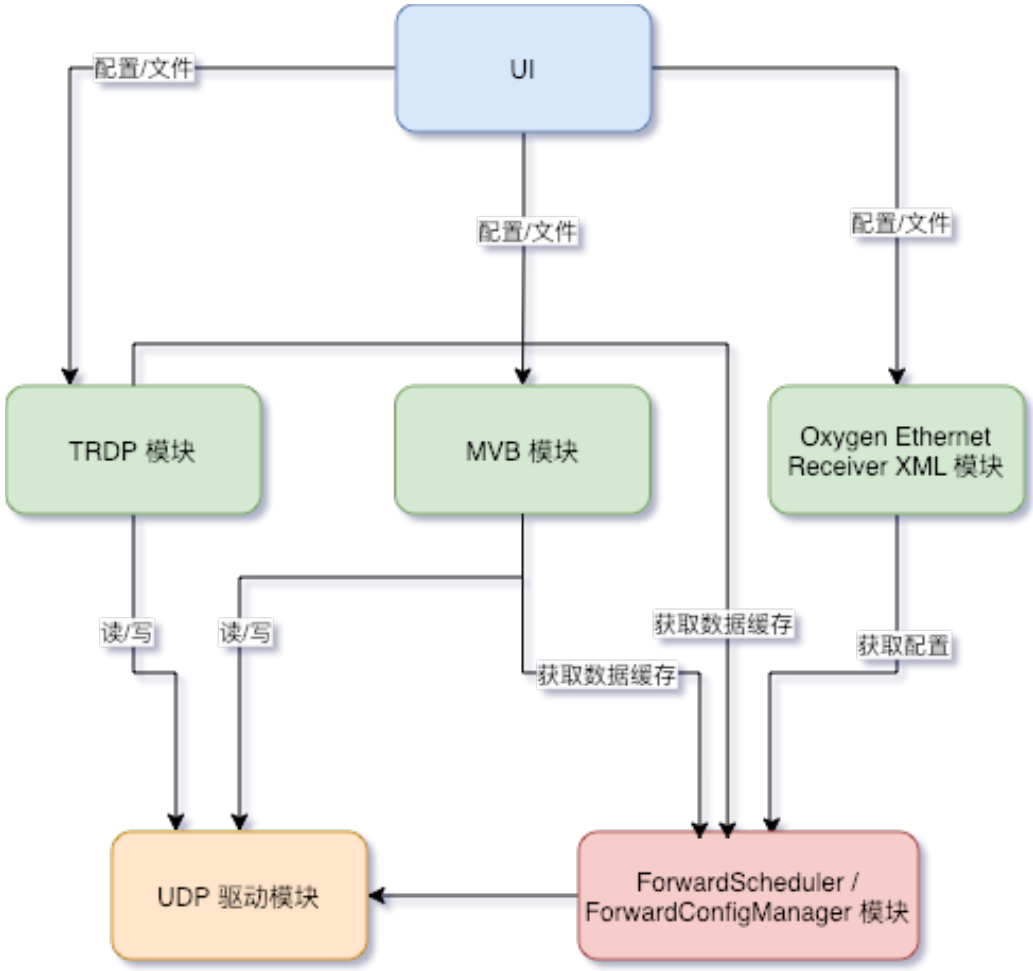
5. 模块使用关系

- **TRDP / MVB 模块**：作为上层调用者注册数据源（如 ComId → buffer），然后调用 `addOrUpdateTask()` 建立周期任务。
- **UDP 驱动模块**：作为数据实际发送者，由 `ForwardScheduler` 绑定其 `sendData(ip, port, data)` 接口。

6. 模块特性总结

- 支持动态增删任务
- 支持转发周期粒度最小可配置为 1ms（受限于系统定时器）
- 多协议复用同一套转发表与调度器逻辑
- 可结合 Oxygen 工具动态生成转发配置

关键流程设计



✅ 模块开发时间安排（共 5 周）

周次	时间范围（示例）	模块名称	工作内容概要
第 1 周	7月1日 - 7月4日	UDP 驱动模块（基础通信）	搭建基于 Asio 的 UDP 收发架构；实现 socket 封装与异步通信接口
第 2 周	7月7日 - 7月11日	TRDP 模块	实现 TRDP 报文结构、ComId 配置、发送解析逻辑
第 3 周	7月14日 - 7月18日	MVB 模块	实现 PD 端口数据结构、数据帧封装与解析逻辑
第 4 周	7月21日 - 7月25日	ForwardScheduler / ConfigManager	周期调度器实现、配置加载、任务表维护逻辑
第 5 周	7月28日 - 8月31日	XML / 界面配置模块	实现 XML 解析、界面字段映射、配置加载入口及界面联动
第6周	8月4日 - 8月8日	UI 界面	实现所有模块联动, 美化等