ICS 25.040
N 10

中华人民共和国国家标准

GB/T 19582.3—2008
代替 GB/Z 19582.3—2004

基于 Modbus 协议的工业自动化网络规范 第 3 部分：Modbus 协议在 TCP/IP 上的实现指南

Modbus industrial automation network specification—
Part 3：Modbus protocol implementation guide over TCP/IP

2008-02-27 发布　　2008-09-01 实施

中华人民共和国国家质量监督检验检疫总局
中国国家标准化管理委员会　发布

目　次

前　言

GB/T 19582—2008《基于 Modbus 协议的工业自动化网络规范》分为 3 部分。

——第 1 部分：Modbus 应用协议；

——第 2 部分：Modbus 协议在串行链路上的实现指南；

——第 3 部分：Modbus 协议在 TCP/IP 上的实现指南。

第 1 部分描述了 Modbus 事务处理；第 2 部分提供了有助于开发者在串行链路上实现 Modbus 应用层的参考信息；第 3 部分提供了有助于开发者在 TCP/IP 上实现 Modbus 应用层的参考信息。

GB/T 19582—2008 包括两个通信规程中使用的 Modbus 应用层协议和服务规范：

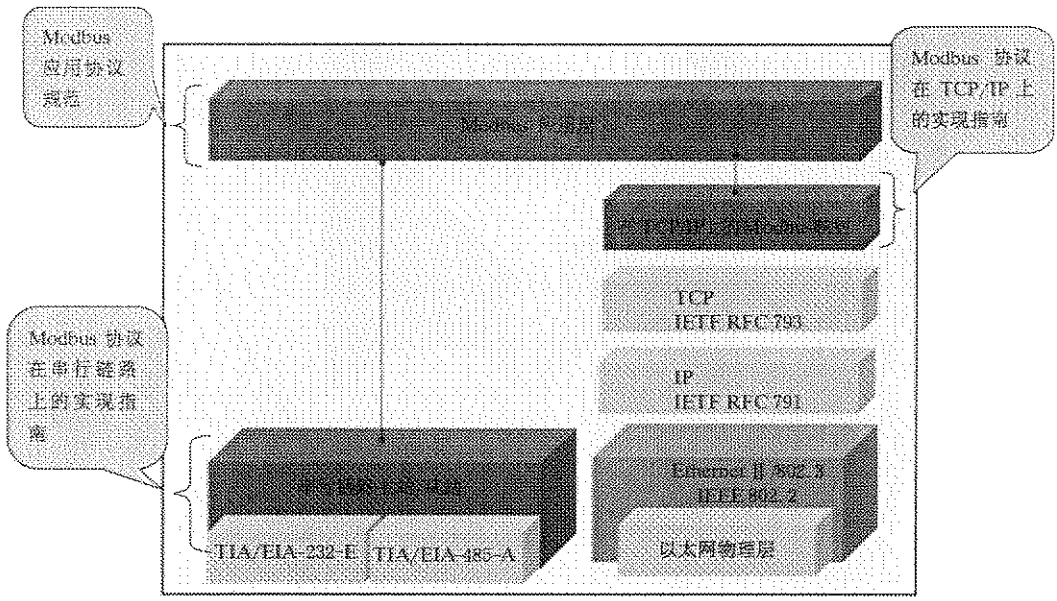
——串行链路上的 Modbus

Modbus 串行链路基于 TIA/EIA 标准：232-E 和 485-A。

——TCP/IP 上的 Modbus

Modbus TCP/IP 基于 IETF 标准：RFC793 和 RFC791。

串行链路和 TCP/IP 上的 Modbus 是根据相应 ISO 分层模型说明的两个通信规程。下图强调指出了 GB/T 19582—2008 的主要部分。深色方框表示规范，浅色方框表示已有的国际标准（TIA/EIA 和 IETF 标准）。



本部分从实施之日起代替 GB/Z 19582.3—2004；GB/Z 19582.3—2004 并于该日起予以废止。

本部分由中国机械工业联合会提出。

本部分由全国工业过程测量和控制标准化技术委员会第四分技术委员会归口。

本部分起草单位：机械工业仪器仪表综合技术经济研究所、西南大学、上海自动化仪表股份有限公司、北京交通大学现代通信研究所、北京机械工业自动化研究所、国家继电器质量监督检验中心、中国四联仪器仪表集团有限公司、中海石油研究中心、西北工业大学、施耐德电气（中国）投资有限公司。

本部分主要起草人：王玉敏、柳晓菁、刘枫、包伟华、孙昕、刘云勇、唐济扬、贺春、刘渝新、徐伟华、欧阳劲松、何军红、华铭、王勇。

GB/Z 19582.3 首次发布时间为 2004 年 9 月 21 日，本部分第一次修订。

引　言

GB/T 19582—2008是对GB/Z 19582—2004《基于Modbus协议的工业自动化网络规范》的修订，修订的依据是IEC 61158 CPF15(FDIS)：2006实时以太网Modbus-RTPS。本部分的结构与GB/Z 19582.3—2004基本一致，但在技术内容上对GB/Z 19582.3—2004进行了补充和完善。

基于 Modbus 协议的工业自动化网络规范
第 3 部分：Modbus 协议在 TCP/IP 上的实现指南

1 范围

本部分叙述了 TCP/IP 上的 Modbus 报文传输服务，提供参考信息以帮助软件开发者实现这种服务。本部分不包括 Modbus 功能码的编码内容，有关这些内容见 GB/T 19582.1—2008。

本部分全面准确地描述了 Modbus 报文传输服务的实现。其目的是促进使用 Modbus 报文传输服务的设备之间的互操作。

本部分主要由三部分组成：

——在 TCP/IP 上的 Modbus 协议概述；

——Modbus 客户机、服务器以及网关实现的功能描述；

——针对一个 Modbus 实现示例的对象模型建议的实现准则。

2 客户机/服务器模型

Modbus 报文传输服务提供了连接到 TCP/IP 以太网上的设备之间的客户机/服务器通信(见图 1)。

这个客户机/服务器模型基于 4 种报文类型：

——Modbus 请求；

——Modbus 证实；

——Modbus 指示；

——Modbus 响应。

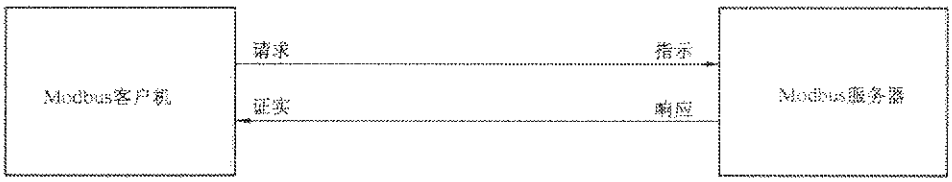


图 1 Modbus 客户机/服务器模型

Modbus 请求是客户机在网络上发送的用来启动事务处理的报文；

Modbus 指示是服务器侧接收的请求报文；

Modbus 响应是服务器发送的响应报文；

Modbus 证实是在客户机侧接收的响应报文。

Modbus 报文传输服务(客户机/服务器模型)用于实时信息交换：

——在两个设备应用程序之间；

——在设备应用和其他设备之间；

——在 HMI/SCADA 应用程序和设备之间；

——在一个 PC 和一个提供在线服务的设备程序之间。

3 规范性引用文件

下列文件中的条款通过 GB/T 19582 的本部分的引用而成为本部分的条款。凡是注日期的引用

文件，其随后所有的修改单（不包括勘误的内容）或修订版均不适用于本部分，然而，鼓励根据本部分达成协议的各方研究是否使用这些文件的最新版本。凡是不注日期的引用文件，其最新版本适用于本部分。

GB/T 19582.1—2008 基于 Modbus 协议的工业自动化网络规范 第 1 部分：Modbus 应用协议

RFC 1122 Requirements for Internet Hosts-Communication Layers

4 缩略语

ADU(Application Data Unit)	应用数据单元
IETF(Internet Engineering Task Force)	互联网工程工作组
IP(Internet Protocol)	因特网协议
MAC(Medium Access Control)	介质访问控制
MB(Modbus)	Modbus
MBAP(Modbus Application Protocol)	Modbus 应用协议
PDU(Protocol Data Unit)	协议数据单元
PLC(Programmable Logic Controller)	可编程序逻辑控制器
TCP(Transport Control Protocol)	传输控制协议
BSD(Berkeley Software Distribution)	伯克利软件发布
MSL(Maximum Segment Lifetime)	最大段寿命
SCADA(Supervisory Control and Data Acquisition)	数据采集和监控

5 背景概要

5.1 协议描述

5.1.1 总体通信结构

见图 2～图 3。

Modbus TCP/IP 的通信系统可以包括不同类型的设备：

——连接至 TCP/IP 网络的 Modbus TCP/IP 客户机和服务器设备；

——互连设备，例如，在 TCP/IP 网络和串行链路子网之间互连的网桥、路由器或网关，该子网允许将 Modbus 串行链路客户机和服务器终端设备连接起来。

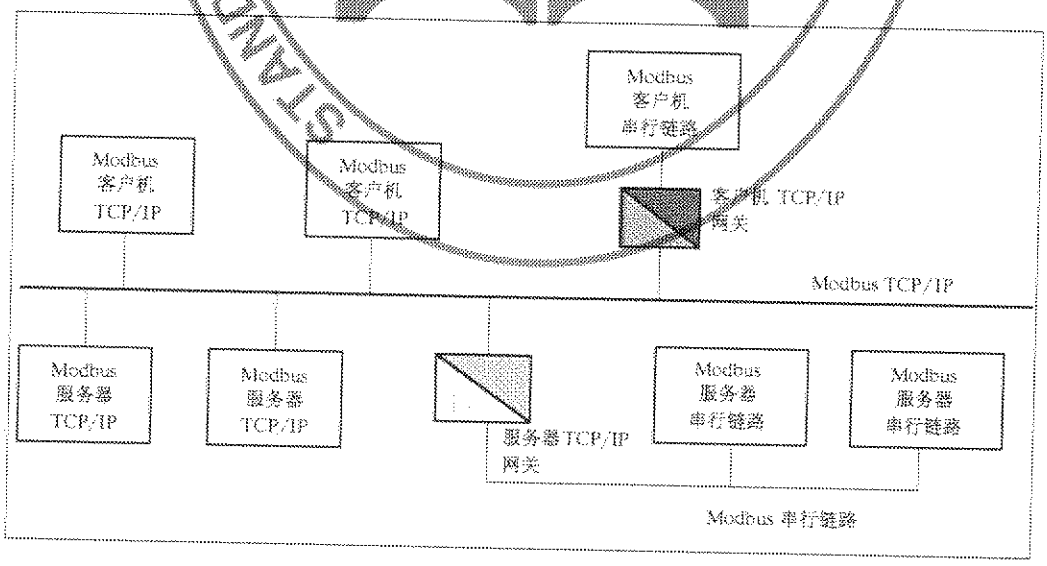


图 2 Modbus TCP/IP 通信结构

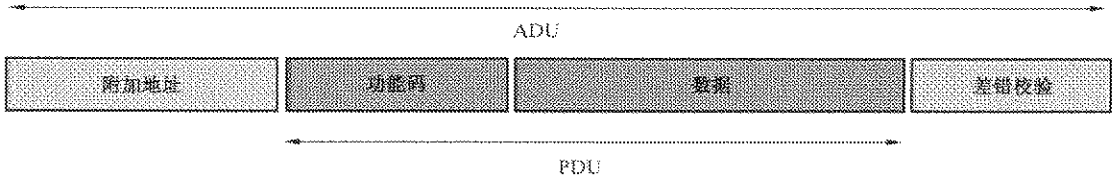


图 3 通用 Modbus 帧

Modbus 协议定义了一个与基础通信层无关的简单协议数据单元(PDU)。特定总线或网络上的 Modbus 协议映射能够在应用数据单元(ADU)上引入一些附加字段。

启动 Modbus 事务处理的客户机建立 Modbus 应用数据单元。这个功能码向服务器指示执行何种操作。

5.1.2 TCP/IP 上的 Modbus 应用数据单元

见图 4。

本条描述了 Modbus TCP/IP 网络上进行的 Modbus 请求或响应的封装。

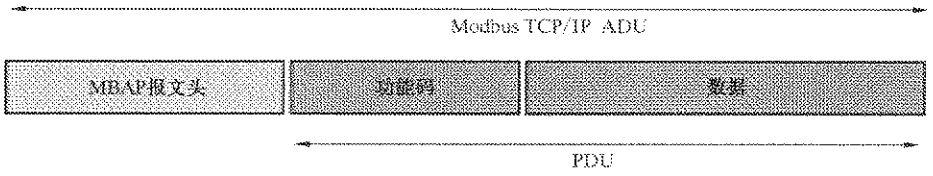


图 4 TCP/IP 上的 Modbus 请求/响应

在 TCP/IP 上使用一种专用报文头来识别 Modbus 应用数据单元。将这种报文头称为 MBAP 报文头(Modbus 应用协议报文头)。

与串行链路上使用的 Modbus RTU 应用数据单元相比,这种报文头有一些区别:

——用 MBAP 报文头中的单字节"单元标识符"取代 Modbus 串行链路上通常使用的 Modbus"从站地址"字段。这个"单元标识符"用于经由如网桥、路由器和网关等设备的通信,这些设备使用单个 IP 地址支持多个独立 Modbus 终端单元。

——用接收方可以验证报文结束的方式设计所有的 Modbus 请求和响应。对于 Modbus PDU 有固定长度的功能码来说,仅功能码就足够了。对于在请求或响应中传输一个可变数据量的功能码来说,数据字段包括字节数。

——当通过 TCP 传输 Modbus 协议时,即使已将报文分成多个信息包来传输,也需在 MBAP 报文头上传输附加长度信息,以便接收者能识别报文边界。显式和隐式长度规则的存在以及 CRC-32 差错校验码的使用(在以太网上),使未检出的请求或响应报文的差错降至极低。

5.1.3 MBAP 报文头描述

MBAP 报文头包括下列字段,见表 1。

表 1 MBAP 报文头的字段

字 段	长 度	描 述	客 户 机	服 务 器
事务处理标识符	2 字节	Modbus 请求/响应事务处理的识别	由客户机设置	服务器从接收的请求中重新复制
协议标识符	2 字节	0=Modbus 协议	由客户机设置	服务器从接收的请求中重新复制
长度	2 字节	随后字节的数量	由客户机设置(请求)	由服务器设置(响应)
单元标识符	1 字节	串行链路或其他总线上连接的远程从站的识别	由客户机设置	服务器从接收的请求中重新复制

报文头长度为 7 个字节：

事务处理标识符：用于事务处理配对。在响应中，Modbus 服务器复制请求的事务处理标识符。

协议标识符：用于系统内的多路复用。通过值 0 识别 Modbus 协议。

长度：长度字段是接续字段的字节数，包括单元标识符和数据字段。

单元标识符：此字段用于系统内路由选择。典型地用于通过 TCP-IP 以太网和 Modbus 串行链路之间的网关对 Modbus+或对 Modbus 串行链路从站的通信。Modbus 客户机在请求中设置这个字段，服务器必须在响应中用相同的值返回这个字段。

通过 TCP 将所有 Modbus/TCP ADU 发送至注册的 502 端口。

注：用最高有效字节在低地址存储的方式编码不同字段。

5.2 Modbus 功能码描述

在 GB/T 19582.1—2008 中详细说明了 Modbus 应用层协议上使用的标准功能码。

6 功能描述

本部分提供的 Modbus 组件结构是一个既包含 Modbus 客户机又包含 Modbus 服务器组件的通用模型，适用于任何设备。

有些设备可能仅提供服务器或客户机组件。

6.1 中给出有关 Modbus 报文传输服务组件结构的简要概述，并且对结构模型内每个组件进行描述。

6.1 Modbus 组件结构模型

见图 5～图 7 和表 2。

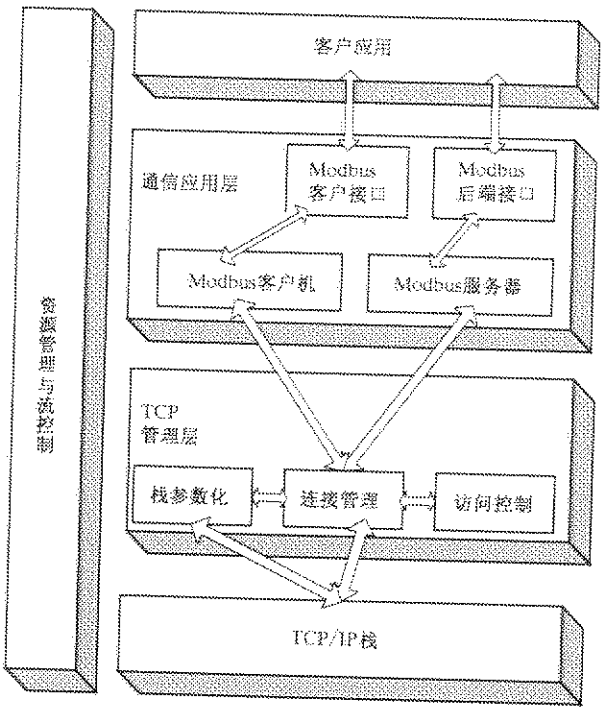


图 5 Modbus 报文传输服务概念结构

6.1.1 通信应用层

一个 Modbus 设备可以提供一个客户机和/或服务器 Modbus 接口。

可提供一个 Modbus 后端接口，间接地允许对用户应用对象的访问。

此接口由四个区域组成：离散量输入、离散量输出(线圈)、输入寄存器和保持寄存器。必须进行这个接口与用户应用数据之间的预映射(本地实现)。

表 2 Modbus 数据访问类型

基本表	对象类型	访问类型	注 释
离散量输入	单个位	只读	I/O 系统可提供这种类型的数据
线圈	单个位	读写	通过应用程序可改变这种类型的数据
输入寄存器	16 位字	只读	I/O 系统可提供这种类型的数据
保持寄存器	16 位字	读写	通过应用程序可改变这种类型的数据

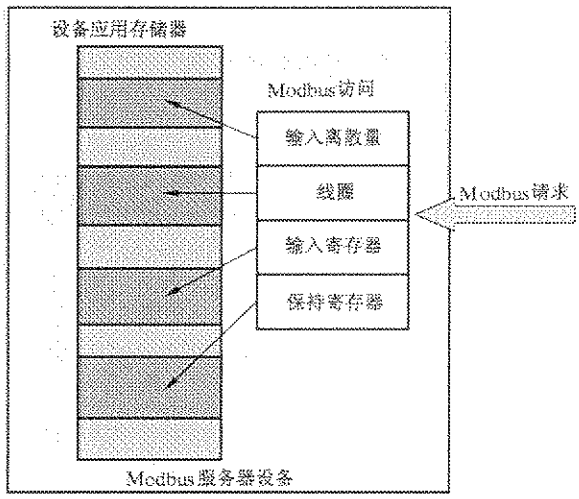


图 6 独立数据块的 Modbus 数据模型

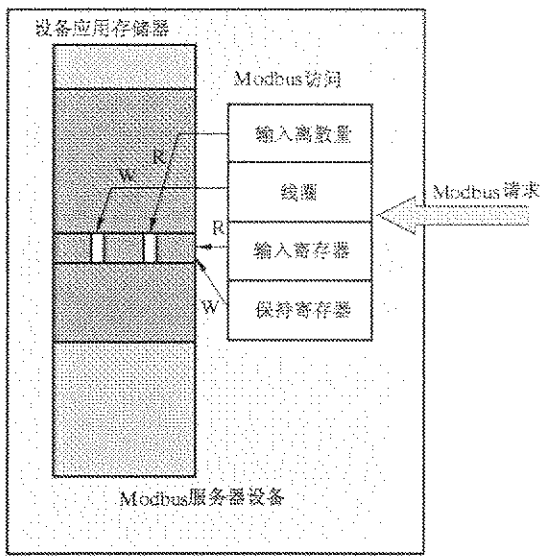


图 7 单个数据块的 Modbus 数据模型

——Modbus 客户机

Modbus 客户机允许用户应用显性地控制与远程设备的信息交换。Modbus 客户机根据用户应用向 Modbus 客户机接口发送的要求中所包含的参数来建立一个 Modbus 请求。

Modbus 客户机使用一个 Modbus 的事务处理，该事务处理管理包括对 Modbus 证实的等待和处理。

——Modbus 客户机接口

Modbus 客户机接口提供一个接口，使得用户应用能够生成各类 Modbus 服务的请求，该服务包括对 Modbus 应用对象的访问。在本规范中没有详细地描述 Modbus 客户机接口(API)，仅在实现模型中给出示例。

——Modbus 服务器

在收到一个 Modbus 请求以后，模块激活一个本地操作进行读、写或完成其他操作。这些操作的处理对应用程序开发人员来说都是完全透明的。Modbus 服务器的主要功能是等待来自 TCP 502 端口的 Modbus 请求，并处理这一请求，然后根据设备中的数据内容生成一个 Modbus 响应。

——Modbus 后端接口

Modbus 后端接口是一个从 Modbus 服务器到定义应用对象的用户应用之间的接口。

注：在本部分中未定义后端接口。

6.1.2 TCP 管理层

注：本部分讨论的 TCP/IP 部分基于 RFC1122，以帮助用户在 TCP/IP 上实现 GB/T 19582.1—2008。

报文传输服务的主要功能之一是管理通信的建立和结束，及管理在所建立的 TCP 连接上的数据流。

——连接管理

在客户机和服务器的 Modbus 模块之间的通信需要使用 TCP 连接管理模块。它负责全面管理报文传输 TCP 连接。

连接管理中存在两种可能：用户应用自身管理 TCP 连接，或全部由这个模块进行连接管理，因此对用户应用是完全透明的。后一种解决方案灵活性较差。

TCP 502 端口的监听是为 Modbus 通信保留的。在默认状态下，强制监听这个端口。然而，某些产品或应用可能需要其他端口作为 TCP 上 Modbus 的通信之用。为此，特别建议：客户机和服务器均应向用户提供对 TCP 端口号进行 Modbus 参数配置的可能性。重要的是：即使在某些特定的应用中为 Modbus 服务配置了其他 TCP 服务器端口，除一些特定应用端口外，TCP 服务器 502 端口仍然必须是可用的。

——访问控制模块

在某些至关重要的场合，必须禁止无关的主机对设备内部数据的访问。这既是需要的安全模式，也是在需要时实现安全处理的原因。

6.1.3 TCP/IP 栈层

可以对 TCP/IP 的栈进行参数配置，以适用对产品或系统的不同特定约束进行数据流控制、地址管理和连接管理。一般说来，BSD 套接字接口被用来管理 TCP 连接。

6.1.4 资源管理和数据流控制

为了平衡 Modbus 客户机与服务器之间进出报文传输的数据流，在 Modbus 报文传输栈的所有各层均提供了数据流控制机制。资源管理和数据流控制模块首先是基于 TCP 内部数据流控制，加上数据链路层的某些数据流控制，以及用户应用层的数据流控制。

6.2 TCP 连接管理

6.2.1 连接管理模块

6.2.1.1 总体描述

Modbus 通信需要建立客户机与服务器之间的 TCP 连接，TCP 连接管理操作见图 8。

连接的建立可以由用户应用模块显式激活，也可以由 TCP 连接管理模块自动激活。

在第一种情况下，用户应用模块必须提供应用程序接口，以便完全管理连接。这种方式为应用开发人员提供了灵活性，但需要 TCP/IP 机制方面的专长。

在第二种情况下，对仅发送和接收 Modbus 报文的用户应用来说，TCP 连接管理是完全隐含的。TCP 连接管理模块负责在需要时建立新的 TCP 连接。

TCP 客户机和服务器连接数量的定义不属于本部分的范围(在本部分中采用值 n)。根据设备能力,TCP 连接的数量可能不同。

实现规则:

a) 如果没有显式用户需求,建议采用自动的 TCP 连接管理。

b) 建议:保持与远程设备的连接,而不要在每次 Modbus/TCP 事务处理时打开和关闭连接。
注:Modbus 客户必须能够接收来自服务器的关闭请求,并关闭连接。当需要时,连接可以被重新打开。

c) 建议:一个 Modbus 客户机要打开与远程 Modbus 服务器的最低限度的 TCP 连接(同一 IP 地址)。最好的选择是一个应用建立一个连接。

d) 几个 Modbus 事务处理可以在同一个 TCP 连接上被同时激活。
注:如果以此方式,Modbus 事务处理标识符必须被用作唯一地识别请求与响应的匹配。

e) 在两个远程 Modbus 实体(一个客户机和一个服务器)之间双向通信的情况下,有必要为客户机数据流和服务器数据流分别建立连接。

f) 一个 TCP 帧只能传送一个 Modbus ADU。建议:不要在同一个 TCP PDU 中发送多个 Modbus 请求或响应。

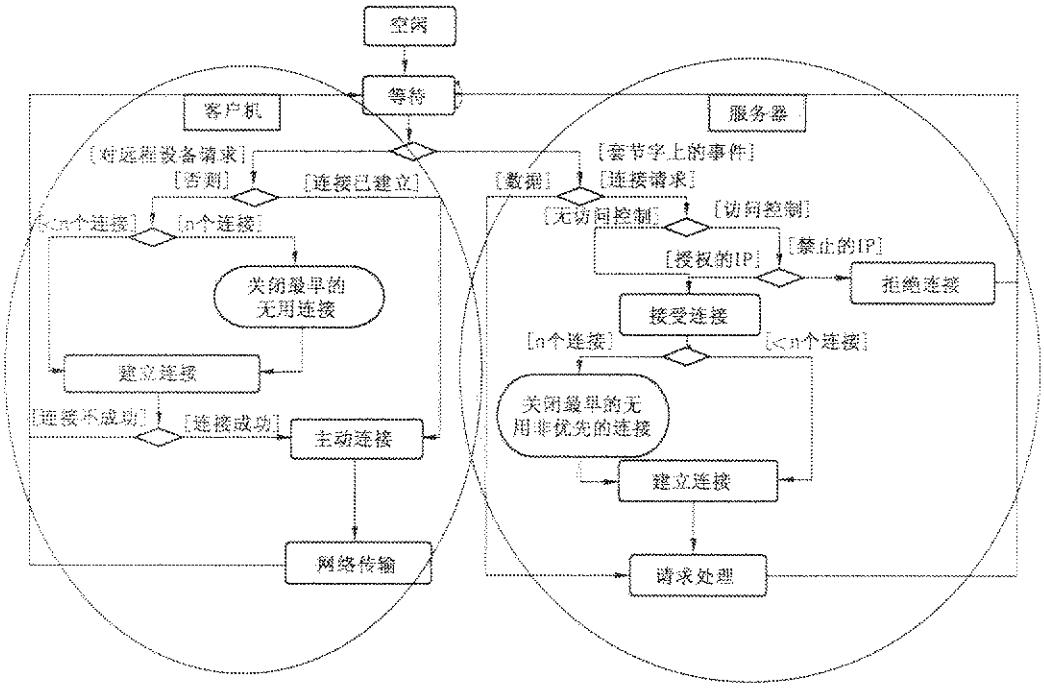


图 8 TCP 连接管理操作图

——显式 TCP 连接管理

用户应用模块负责管理所有的 TCP 连接:主动的和被动的建立及结束连接等。对客户机与服务器间所有的连接进行这种管理。在用户应用模块中使用 BSD 套接字接口来管理 TCP 连接。这种方案提供了完全的灵活性,但也意味着应用开发人员要具备充分的有关 TCP 知识。

考虑到设备的能力和需求,必须有限制地配置客户机与服务器间连接的数量。

——自动 TCP 连接管理

TCP 连接管理对用户应用模块是完全透明的。连接管理模块可以接受足够数量的客户机/服务器连接。

然而,在超过所授权数量的连接时,必须有一种实现机制。在这种情况下,建议:关闭最早建立的不

使用的连接。

当收到第一个来自远程客户机或本地用户应用的数据包时，就建立了与远程对象的连接。如果网络提出终止或本地设备决定终止，此连接将被关闭。在接收连接请求时，访问控制选项可用来禁止未授权的客户访问设备。

TCP 连接管理模块采用栈接口（通常用 BSD 套接字接口）来与 TCP/IP 栈进行通信。

为了保持系统需求与服务器资源之间的兼容，TCP 管理将保持两个连接池。

——第一个池（优先连接池）由那些从不被本地主动关闭的连接组成。必须提供一个配置来建立这个池。实现的原理是将这个池的每一个可能的连接与一个特定的 IP 地址联系起来。具有这个 IP 地址的设备被称为“标记的”。任何一个被“标记的”设备的新的连接请求必须被接收，并从优先连接池中取出。还有必要设置允许每个远程设备最多建立连接的数量，以避免同一设备使用优先连接池中所有的连接。

——第二个池（非优先连接池）包括了非标记设备的连接。这里采用的规则是：当有来自非标记设备的新的连接请求，以及池中没有连接可用时，关闭早些时候建立的连接。

可有选择地提供一个配置来分配每个池中可用连接的数量。然而（非强制性的），如果需要，设计人员可在设计期间设置连接的数量。

6.2.1.2 连接管理描述

——连接建立

见图 9。

Modbus 报文传输服务必须在 502 端口上提供一个监听套接字，允许接收新的连接和与其他设备交换数据。

当报文传输服务需要与远程服务器交换数据时，它必须与远程 502 端口建立一个新的客户机连接，以便远程交换数据。本地端口必须高于 1024，并且对每个客户机的连接各不相同。

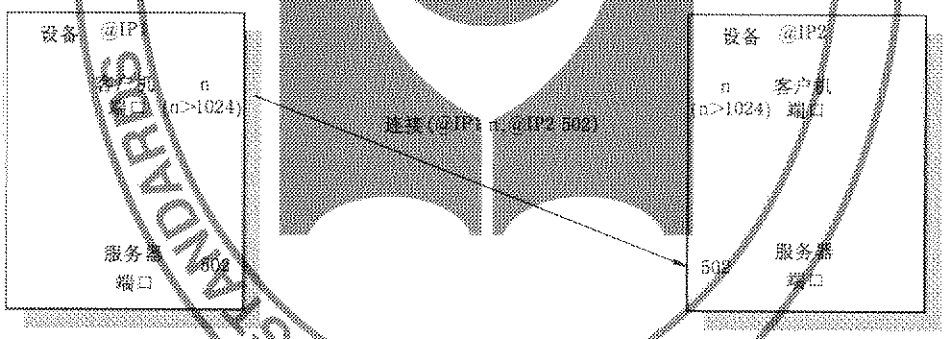


图 9 Modbus TCP/IP 连接建立

如果客户机与服务器的连接数量大于授权的连接数量，则最早建立的未用连接被关闭。访问控制机制可以被激活，用于检查远程客户机的 IP 地址是否被授权。如果未经授权，将拒绝该新的连接。

——Modbus 数据传送

一个 Modbus 请求必须在已经被打开的正确的 TCP 连接上发送。远程设备的 IP 地址用于寻找所建立的 TCP 连接。在与同一个远程设备建立多个连接时，必须选择其中一个连接来发送 Modbus 报文，可以采取不同的选择策略，例如：最早的连接和第一个连接。在 Modbus 通信的全过程中，连接必须始终保持打开。一个客户机可以向一个服务器启动多个 Modbus 事务处理，而不必等待先前的事物处理结束。

——连接关闭

当客户机与服务器间的 Modbus 通信结束时，客户机必须关闭用于通信的连接。

6.2.2 操作模式对 TCP 连接的影响

某些操作模式(两操作端点之间通信断开、一个端点的故障和重新启动)会对 TCP 连接产生影响。连接在一侧被视为关闭或异常终止而另一侧未确认,这种连接被称为半打开的连接。

本章描述各种主要操作模式的特性。这种描述基于在连接的两端采用了保持连接 TCP 机制的假设(见 6.3.2)。

6.2.2.1 两操作端之间通信断开

通信断开的原因可以是服务器侧以太网连接电缆断开。预期的特性是:

——如果在连接上没有正在发送数据包:

如果通信断开持续的时间小于"保持连接"计时器的值,将察觉不到通信断开。如果通信断开时间大于"保持连接"计时器的值,将一个错误返回到 TCP 管理层,由其复位连接。

——如果在连接断开的前后发送一些数据包:

TCP 重新传输算法(Jacobson 算法、Karn 算法以及指数补偿算法,参见第 6.3.2)被激活。这可以导致在"保持连接"计时器终止之前由栈的 TCP 层复位连接。

6.2.2.2 服务器端的故障和重新启动

在服务器故障和重新启动以后,客户机端处于"半打开"连接状态。预期的特性是:

——如果在半打开的连接上没有发送数据包:

只要"保持连接"计时器还在计时中,从客户机端看,这种 TCP 半打开的连接被视为打开的。然后,将返回一个错误到 TCP 管理层,由其复位连接。

——如果在半打开的连接上发送一些数据包:

服务器在不再存在的连接上接收数据。栈的 TCP 层发送一个复位指令来关闭客户机端的半打开的连接。

6.2.2.3 客户机端的故障和重新启动

在客户机故障和重新启动之后,服务器侧处于"半打开"连接状态。预期的特性是:

——如果在半打开的连接上没有发送数据包:

只要"保持连接"计时器还在计时中,从服务器端看,这种 TCP 半打开连接被视为打开的。然后,将返回一个错误到 TCP 管理层,由其复位连接。

——如果在"保持连接"计时器完成计时前,客户机打开一个新的连接:

必须分析两种情况:

1) 所打开的连接与服务器侧半打开的连接具有相同的特性(相同的源和目的端口、相同的源和目的 IP 地址),所以,在连接建立超时后(伯克利实现的多数情况下为 75 s),TCP 栈层将不能打开连接。为了避免在较长超时时间内不能进行通信,建议:在客户机端重新启动后,确保使用与原有连接不同的源端口号建立连接。
2) 所打开的连接与服务器侧半打开的连接具有不同的特性(不同的源端口和相同的目的端口、相同的源和目的 IP 地址),所以,在 TCP 栈层上打开连接,并向服务器侧的 TCP 管理层发送信号。

如果服务器侧 TCP 管理层仅支持一个远程客户机 IP 地址的连接,那么可以关闭原来的半打开的连接,使用新的连接。

如果服务器侧 TCP 管理层支持多个远程客户机 IP 地址的连接,那么新的连接保持打开状态,原来的连接也保持半打开状态,直到"保持连接"计时器计时结束,此时,将返回一个错误到 TCP 管理层。然后,TCP 管理层将能够复位原有的连接。

6.2.3 访问控制模块

这个模块的目的是检查每一个新的连接,使用一个合法授权的远程 IP 地址表,它可以授权或禁止一个远程客户机的 TCP 连接。

在至关重要的场合，应用开发人员需要选择访问控制模式来保证网络的访问。在这种情况下，需要对每个远程 IP 授权/禁止访问。用户需提供一个 IP 地址表，并特别注明每个 IP 地址是否合法授权。在默认情况下，在安全模式中，用户未配置的 IP 地址均被禁止。因此，借助于访问控制模式，关闭来自未知的 IP 地址的访问连接。

6.3 TCP/IP 栈的使用

见图 10。

TCP/IP 栈提供了一个接口，用来管理连接、发送和接收数据，还可以进行某些参数配置，以使得栈的特性适应于设备或系统的限制。

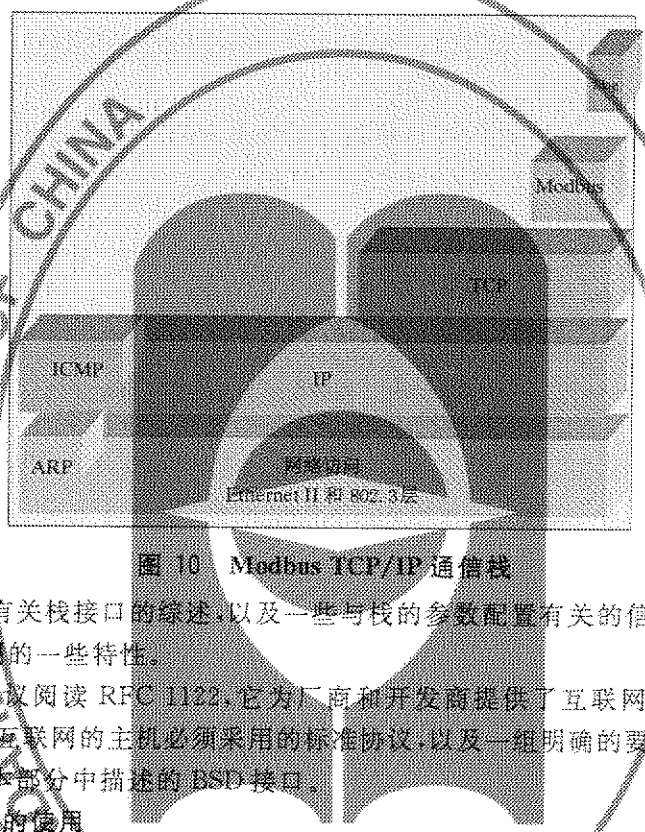


图 10 Modbus TCP/IP 通信栈

本章的目的是给出有关栈接口的综述，以及一些与栈的参数配置有关的信息。综述的主要内容是 Modbus 报文传输所使用的一些特性。

有关更多的信息，建议阅读 RFC 1122，它为厂商和开发商提供了互联网通信软件的指南。RFC 1122 详述了一个连接到互联网的主机必须采用的标准协议，以及一组明确的要求和选项。

栈接口一般是基于本部分中描述的 BSD 接口。

6.3.1 BSD 套接字接口的使用

注：有些 TCP/IP 栈从性能考虑提出其他类型的接口。Modbus 客户机或服务器可以使用这些特定的接口，但是在本部分中对这种使用不做描述。

一个套接字是一个通信端点，它是通信中的基本构成块。通过套接字发送和接收数据来执行一个 Modbus 通信。TCP/IP 库仅提供了使用 TCP 和提供基于连接的通信服务的流套接字。

socket()函数用来创建套接字。返回的一个套接字号被创建者用来访问该套接字。套接字创建时没有地址(IP 地址和端口号)。直到一个端口被绑定到该套接字时，才可用于接收数据。

bind()函数用来绑定一个端口号到套接字。bind()函数在套接字与所指定的端口号之间建立一种联系。

为了初始化一个连接，客户机必须发送 connect()函数来指定套接字号、远程 IP 地址和远程监听端口号(主动连接建立)。

为了完成连接，服务器必须发送 accept()函数来指定在先前 listen()调用中所指定的套接字号(被动连接建立)。一个新的套接字被创建，并具有与初始套接字相同的特性。这个新的套接字连接到客户机的套接字，而将其套接字号返回到服务器。于是，释放初始套接字，以便为其他欲与该服务器连接的客户机使用。

在 TCP 连接建立以后，数据即可被传送。将 Send()和 recv()函数专门地设计成与已连接的套接

字一起使用。

setsockopt()函数允许套接字的创建者将套接字与选项关联。这些选项修改了套接字的操作特征。在6.3.2给出这些选项的描述。

select()函数允许编程人员测试所有套接字上的事件。

shutdown()函数允许套接字的使用者利用套接字来终止send()和/或recv()。

一旦不再需要套接字,通过使用close()函数来放弃套接字的描述符。

图11给出了客户机与服务器间的完整的Modbus通信过程。客户机建立一个连接,向服务器发送3个Modbus请求,而不等待第一个请求的响应。在收到所有的响应后,客户机正常地关闭连接。

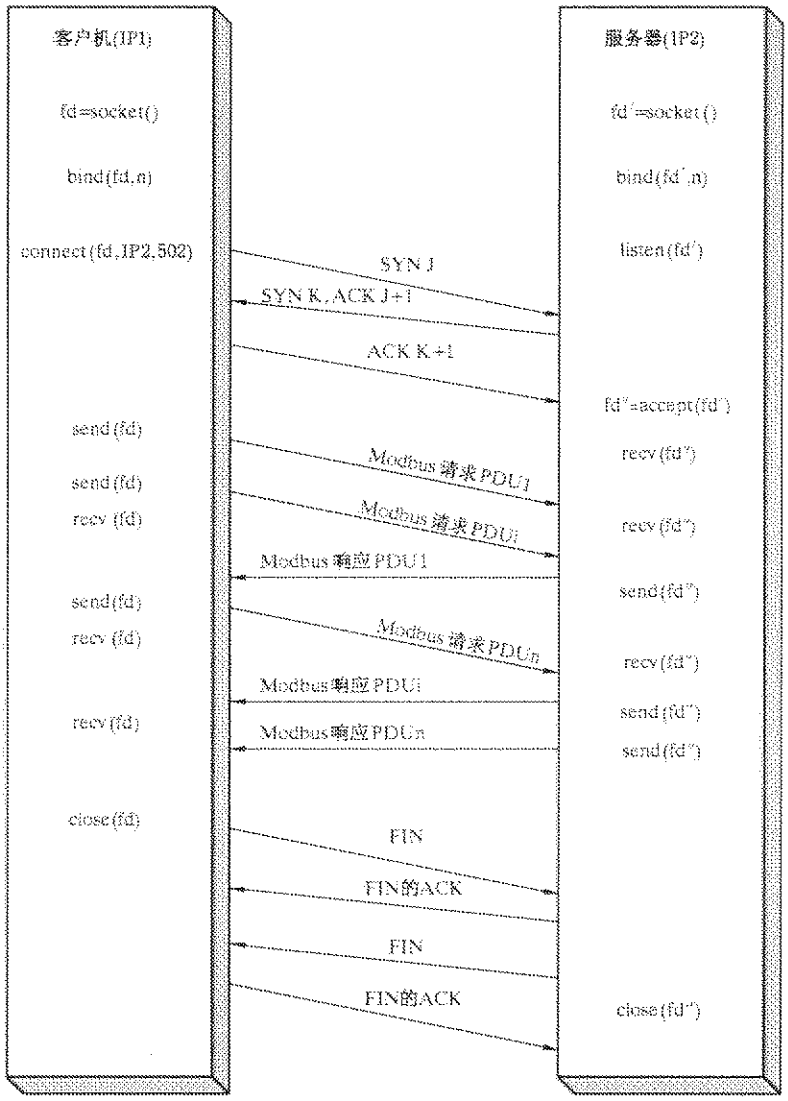


图11 Modbus信息交换

6.3.2 TCP层参数配置

可以调整TCP/IP栈的一些参数以使其特性适应产品或系统的限制。TCP层的下列参数可以进行调整:

——每个连接的参数

SO-RCVBUF, SO-SNDBUF:

这些参数允许设置发送和接收缓存区的大小。可以通过调整这些参数来实现流控制管理。接收缓

存区的大小是该连接的通告窗口的最大尺寸。为了提高性能，必须增加套接字缓存区的大小。然而，这些值必须小于内部驱动器的资源，以便在内部驱动器的资源耗尽之前关闭 TCP 窗口。

接收缓存区大小取决于 TCP 窗口大小、TCP 最大段的大小和接收输入帧所需的时间。由于最大段的大小为 300 个字节(一个 Modbus 请求需要最大 256 个字节＋MBAP 报文头)，如果需要 3 个帧进行缓存，可将套接字缓存区大小调整为 900 个字节。为了满足最大需求和最好的预定时间，可以增加 TCP 窗口的大小。

TCP-NODELAY：

通常，小报文包在局域网(LAN)上的传输不会产生问题，因为多数局域网是不拥塞的，但是，这些小报文包在广域网上将会造成拥塞。一种简单方案(称为“NAGLE 算法”)是：将若干小报文集中后，当前面报文的 TCP 确认到达时，用单个段发送它们。

为了获得更好的实时特性，应将小报文直接发送，而不要试图将其收集到一个段内再发送。这就是建议强制 TCP-NODELAY 选项的原因，这个选项禁止在客户机和服务器连接上采用“NAGLE 算法”。

SO-REUSEADDR：

当 Modbus 服务器关闭一个由远程客户启动的 TCP 连接时，在这个连接处于“时间等待”状态(2 个MSL：最大段寿命)的过程中，该连接所用的本地端口号不能被再次用来打开一个新的连接。

建议为每个客户机和服务器连接规定该 SO-REUSEADDR 选项，以旁路这个限制。此选项允许该进程为自身分配一个端口号，该端口号是在 2 个 MSL 期间内等待客户机并监听套接字的连接的一部分。

SO-KEEPALIVE：

在 TCP/IP 协议默认状态下，没有数据通过空闲的 TCP 连接发送。因此，如果在 TCP 连接端上没有进程发送数据，在 2 个 TCP 模块间就不交换任何数据。这就是假设客户机应用或服务器应用均采用定时器来检测连接的非激活性，以便关闭连接。

建议在客户机与服务器连接两端均采用 KEEPALIVE 选项，以便轮询另一端来了解是发生故障并死机，还是发生故障并重新启动。

然而，必须注意，采用 KEEPALIVE 可能引起一个非常良好的连接在瞬间故障时连接中断，如果保持连接的定时器定时的时间太短，将占用不必要的网络带宽。

——整个 TCP 层的参数

建立 TCP 连接超时：

多数基于伯克利的系统将建立新连接的时限设定为 75 s，应根据应用的实时限制，调整这个默认值。

保持连接参数：

连接的默认空闲时间是 2 h。超过此空闲时间将触发一个保持连接试探过程。在第一个保持连接试探后，在最大次数内每隔 75 s 发送一个试探，直到收到对试探的响应为止。

在一个空闲连接上发出保持连接试探的最大数是 8 次。如果发出最大试探次数之后而没有收到响应，TCP 向应用发出信号报告一个错误，由应用来决定关闭连接。

超时与重发参数：

如果检测到一个 TCP 报文包丢失，将重发此报文包。检测丢包的一种方法是管理重发超时(RTO)，如果在 RTO 终止之前没有收到来自远程端的确认，就认为 TCP 报文包丢失。

TCP 进行 RTO 的动态评估。为此，在发送每个非重发的报文包后测量往返时间(RTT)。往返时间(RTT)是指报文包到达远程设备并把从远程设备获得的一个确认返回给发送设备所用的时间。一个连接的往返时间是动态计算的，然而，如果 TCP 不能在 3 s 内获得 RTT 的估算，那么，就设定 RTT 的默认值为 3 s。

如果已经估算出 RTO，它将被用于下一个报文包的发送。如果在估算的 RTO 终止之前没有收到

该报文包的确认，启用指数补偿算法。在一个特定的时间段内，在对相同报文进行最大重发次数的重试后，如果还收不到确认，连接终止。

可以对某些栈的最大重发次数和连接终止之前重发的最长时间(tcp_ip_abort_interval)进行设置。

在 TCP 标准中定义了一些重发算法：

——Jacobson RTO 估算算法用来估算重发超时(RTO)；

——Karn 算法指出，在重发段，不应进行 RTO 估算；

——指数补偿算法定义：对于时间上限为 64 s 的每一次重发，加倍重发超时；

——快速重发算法允许在收到 3 个重复确认之后进行重发。考虑这个算法是因为：在 LAN 上可能会导致报文丢失的检测快于等待 RTO 终止的检测。

在 Modbus 实现中，推荐使用这些算法。

6.3.3 IP 层的参数配置

6.3.3.1 IP 参数

下列参数必须在 Modbus 实现的 IP 层进行配置：

——本地 IP 地址：IP 地址可以是 A、B 或 C 类中的一种。

——子网掩码：可基于各种原因，将 IP 网络划分成子网：使用不同的物理介质(例如：以太网、广域网等)、更有效地使用网络地址以及控制网络流量的能力。子网掩码必须与本地 IP 地址的 IP 地址类相一致。

——默认网关：默认网关的 IP 地址必须与本地 IP 地址在同一子网内。禁止使用值 0.0.0.0。如果没有定义网关，那么此值可设为 127.0.0.1 或本地 IP 地址。

注：Modbus 报文传输服务在 IP 层上不要求分段功能。

应该利用本地 IP 地址、子网掩码和默认网关(不同于 0.0.0.0)配置本地 IP 端点。

6.4 通信应用层

6.4.1 Modbus 客户机

Modbus 客户机单元见图 12。

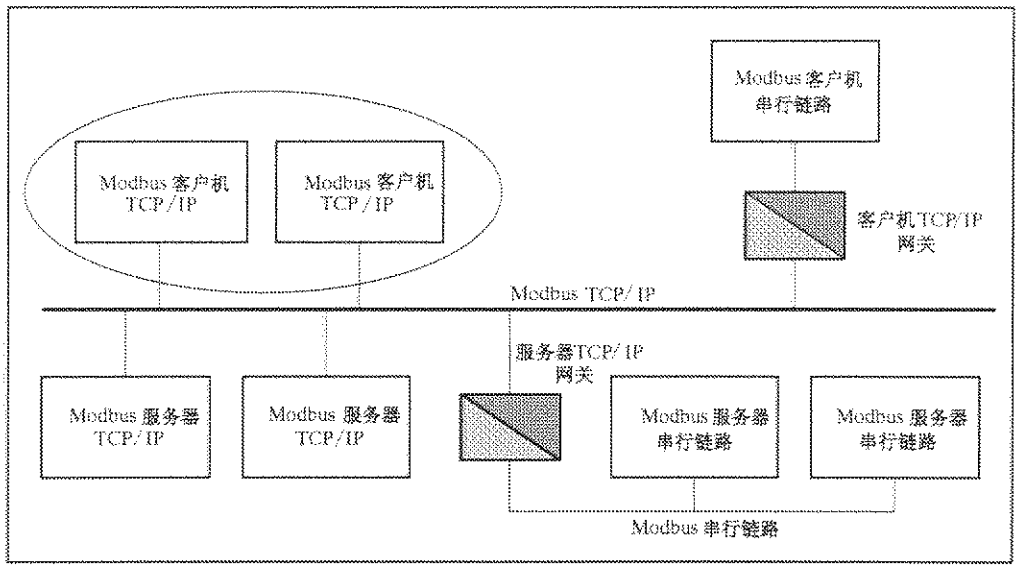


图 12 Modbus 客户机单元

6.4.1.1 Modbus 客户机设计

Modbus/TCP 协议的定义能够对一个客户机进行简单的设计。图 13 描述了客户机发送 Modbus 请求和处理 Modbus 响应的主要处理过程。

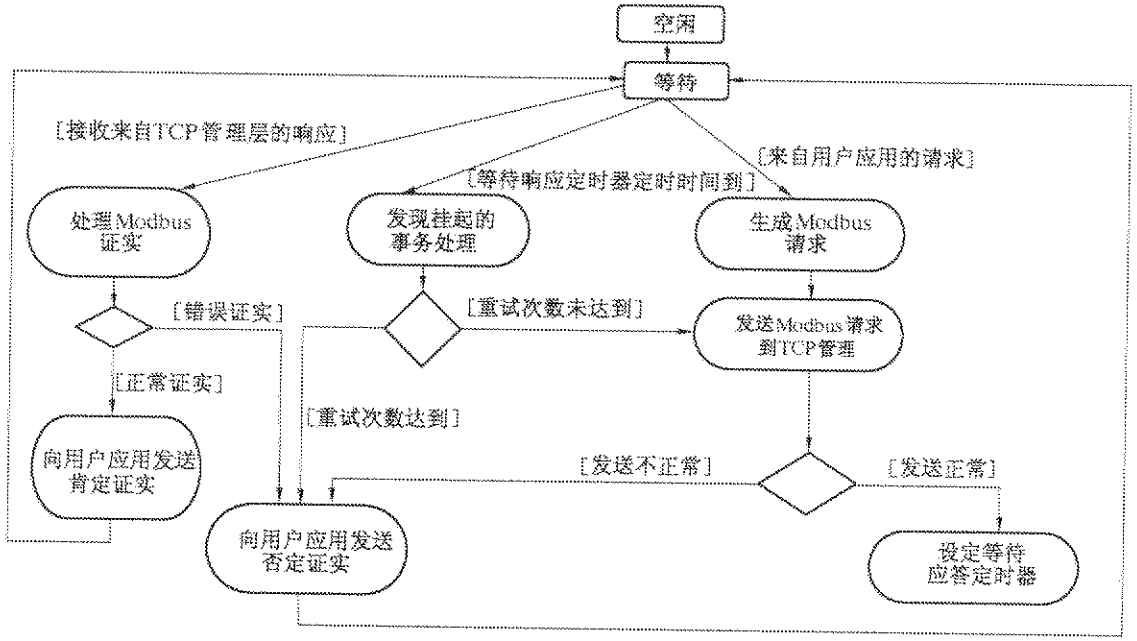


图 13 Modbus 客户机操作图

一个 Modbus 客户机可以接收 3 类事件：

——一个来自用户应用的发送请求的新要求，在这种情况下，必须对 Modbus 请求进行编码，并使用 TCP 管理组件服务通过网络进行发送 Modbus 请求。下层(TCP 管理模块)会返回一个由于 TCP 连接错误或其他错误信息而导致的错误信息。

——来自 TCP 管理的一个响应，在这种情况下，客户机必须分析响应的内容，并向用户应用发送一个证实。

——由于无响应而超时结束。可以通过网络发送一个重试信息，或向用户应用发送一个否定证实。

注：这些重试是由 Modbus 客户机启动的，可以在无 TCP 确认的情况下由 TCP 层来进行其他类型的重试。

6.4.1.2 Modbus 请求的生成

在收到来自用户应用的要求后，客户机必须生成一个 Modbus 请求，并发送到 TCP 管理。

可以将生成 Modbus 请求分解成为几个子任务：

——Modbus 事务处理的实例化，使客户机能够记忆所有需要的信息，以便将后续的响应与相应的请求匹配，并向用户应用发送证实。

——Modbus 请求(PDU+MPAB 报文头)的编码。启动要求的用户应用必须提供所有需要的信息，使得客户机能够进行请求的编码。根据 Modbus 应用协议进行 Modbus PDU 的编码(Modbus 功能码、相关参数和应用数据，见 GB/T 19582.1—2008)。填充 MBAP 报文头的所有字段。然后，将 MBAP 报文头作为 PDU 的前缀，生成 Modbus 请求 ADU。

——发送 Modbus 请求 ADU 到 TCP 管理模块，TCP 管理模块负责对远程服务器寻找正确的 TCP 套接字。除了 Modbus ADU 以外，还必须传递目的 IP 地址。

图 14 比图 13 更深入地描述了请求生成的操作过程。

表 3 描述了读远程服务器中 #5 寄存器的 Modbus 请求 ADU 编码示例。

——事务处理标识符

事务处理标识符用于将请求与未来响应之间建立联系。因此，对 TCP 连接来说，在同一时刻，这个标识符必须是唯一的。有几种使用此标识符的方式：

1) 例如：可以作为一个带有计数器的简单“TCP 顺序号”，对每一个请求将计数器增加 1；

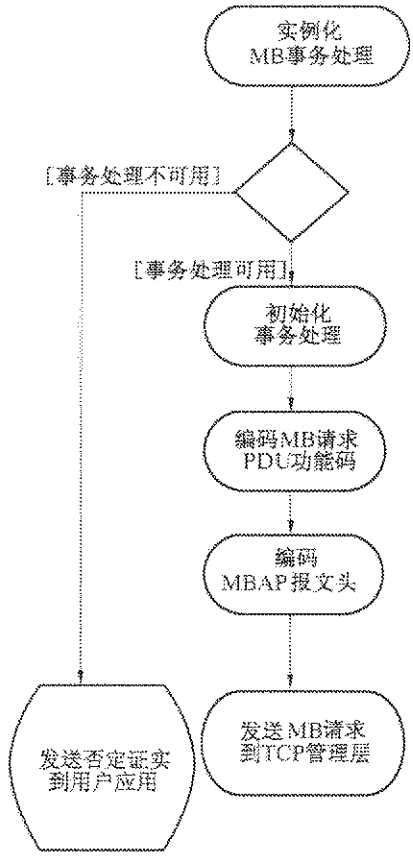


图 14　请求生成操作图

表 3　Modbus 请求 ADU 编码示例

	描　述	大　小	示　例
MBAP 报文头	事务处理标识符 Hi	1	0x15
	事务处理标识符 Lo	1	0x01
	协议标识符	2	0x0000
	长度	2	0x0006
	单元标识符	1	0xFF
Modbus 请求	功能码[a]	1	0x03
	起始地址	2	0x0004
	寄存器数量	2	0x0001

[a] 见 GB/T 19582.1—2008。

2)　也可以用作智能索引或指针，来识别事务处理的内容，以便记忆当前的远程服务器和挂起的请求。

通常，在 Modbus 串行链路上，客户机必须一次发送一个请求。这意味着这个客户机在发送第二个请求之前必须等待对第一个请求的回答。在 Modbus/TCP 上，可以向同一个服务器发送多个请求而无需等待服务器的证实。Modbus/TCP 到 Modbus 串行链路之间的网关负责保证这两种操作之间的兼容性。

服务器接受的请求数量取决于其容量，即：服务器资源量和 TCP 窗口大小。同样，客户机同时启动

事务处理的数量也取决于客户机的资源容量。这个实现参数称为"NumberMaxofClientTransaction"，必须作为 Modbus 客户机的一个特性进行描述。根据设备的类型，此参数取值为 1～16。

——单元标识符

在 Modbus+或 Modbus 串行链路子网中对设备进行寻址时，这个字段是用于路由的目的。在这种情况下，"单元标识符"携带一个远程设备的 Modbus 从站地址：

1) 如果 Modbus 服务器连接到 Modbus+或 Modbus 串行链路子网，并通过一个网桥或网关来寻址，Modbus 单元标识符对识别连接到网桥或网关后的子网的从站设备是必需的。目的 IP 地址识别网桥本身的地址，而网桥则使用 Modbus 单元标识符将请求转交给正确的从站设备。

2) 分配串行链路上 Modbus 从站设备地址为 1～247(十进制)。地址 0 作为广播地址。

对 TCP/IP 来说，利用 IP 地址寻址 Modbus 服务器；因此，Modbus 单元标识符是无用的。必需使用值 0xFF。

3) 当对直接连接到 TCP/IP 网络上的 Modbus 服务器寻址时，建议不要在"单元标识符"字段中使用有效的 Modbus 从站地址。在一个自动化系统中重新分配 IP 地址的情况下，并且如果以前分配给 Modbus 服务器的 IP 地址又被指配给网关，使用一个有效的从站地址可能会由于网关的路由不畅而引起麻烦。使用无效的从站地址，网关仅是简单地废弃 Modbus PDU，而不会有任何问题。建议采用 0xFF 作为"单元标识符"的无效值。

注：0x00 也可以用作"单元标识符"的无效值。

6.4.1.3 处理 Modbus 证实

在 TCP 连接中，当收到一个响应帧时，位于 MBAP 报文头中的事务处理标识符用来将该响应与先前发往 TCP 连接的原始请求联系起来：

——如果事务处理标识符没有指向任何 Modbus 挂起的事务处理，那么必须废弃该响应；

——如果事务处理标识符指向 Modbus 挂起的事务处理，那么必须分析该响应，以便向用户应用发送 Modbus 证实(肯定的或否定的证实)。

对该响应的分析包括验证 MBAP 报文头和 Modbus 响应 PDU：

——MBAP 报文头

在检验协议标识符必为 0x0000 以后，长度给出了 Modbus 响应的大小。

如果响应来自直接连接到 TCP/IP 网络的 Modbus 服务器设备，TCP 连接标识符足以清晰地识别出远程服务器。因此，MBAP 报文头中携带的单元标识符是没有意义的(值 0xFF)，必须被忽略。

如果将远程服务器连接在一个串行链路子网上，并且响应来自于一个网桥、路由器或网关，那么单元标识符(值≠0xFF)识别发送初始响应的远程 Modbus 服务器。

——Modbus 响应 PDU

必须检验功能码，并根据 Modbus 应用协议分析 Modbus 的响应格式：

1) 如果功能码与请求中所用的功能码相同，并且如果响应的格式是正确的，那么，向用户应用发出 Modbus 响应作为肯定的证实。

2) 如果功能码是一个 Modbus 异常功能码(功能码+80H)，向用户应用发出一个异常响应作为肯定的证实。

3) 如果功能码与请求中所用的功能码不同(非预期的功能码)，或如果响应的格式是错误的，那么，向用户应用发出一个出错信号作为否定的证实。

注：肯定证实是指服务器收到请求命令并做出响应的证实，并不意味着服务器能够成功地完成请求命令中要求的操作(Modbus 异常响应指明了执行操作失败)。

图 15 比图 13 更深入地描述了证实处理的操作过程。

6.4.1.4 超时管理

对 Modbus/TCP 上事务处理所需响应时间不刻意作出规定。

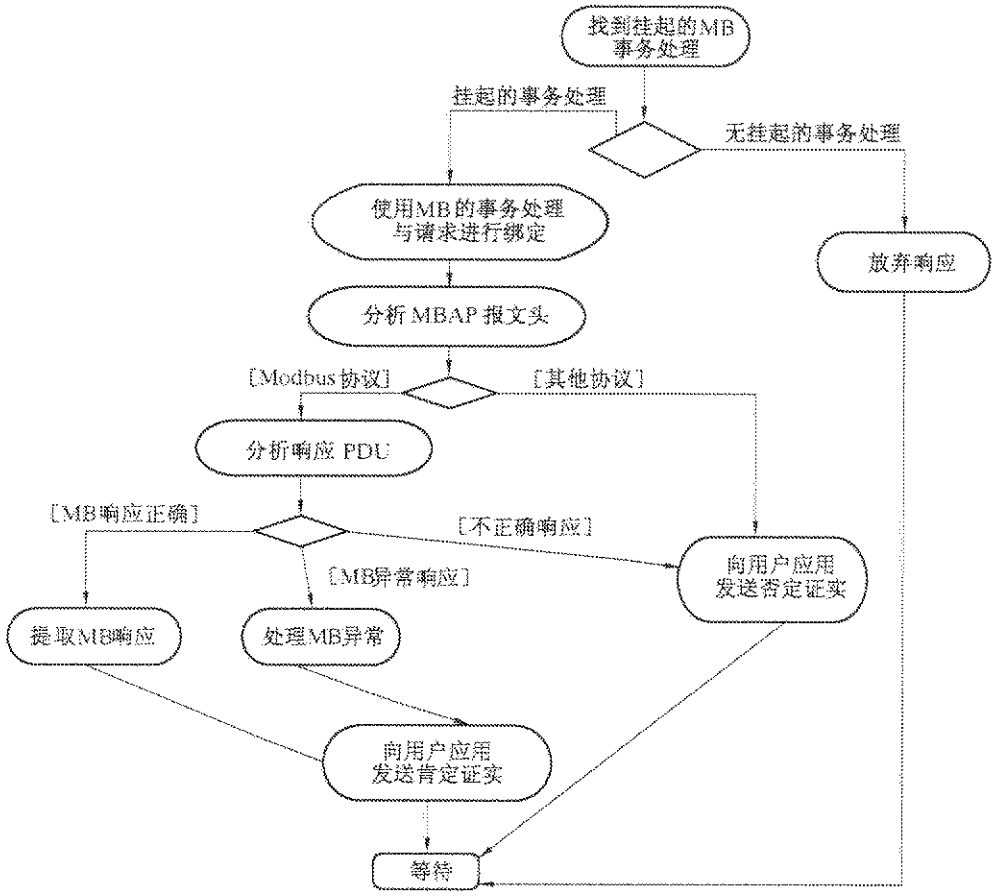


图 15 Modbus 证实处理操作图

这是因为，希望 Modbus/TCP 可用于尽可能宽泛的通信场合，从毫秒级定时的 I/O 扫描到延时几秒钟的远距离无线链接。

从客户机的角度，超时必须考虑网络上预期的传输延迟，以便确定一个合理的响应时间。这种传输延迟可能是交换式以太网中的几个毫秒，或广域网连接中的几百毫秒。

反过来讲，客户机启动应用重试所使用的任何超时时间应该大于预期的最大的"合理"响应时间。如果不遵循这一点，目标设备或网络就存在过度拥塞的潜在危险，而反过来会导致更多的错误。这是一个应该始终避免的特性。

因此，在实际中，在高性能应用中所使用的客户机超时似乎总是与网络拓扑和期望的客户机性能有关。

时间因素不很重要的系统经常采用标准 TCP 默认值作为超时值，在多数平台上，几秒钟之后将报告通信故障。

6.4.2 Modbus 服务器

见图 16。

Modbus 服务器的作用是为应用对象提供访问以及为远程客户机提供服务。

根据用户应用，可以提供不同类型的访问：

——简单访问：获得或设定应用对象的属性；

——高级访问：启动一个特定的应用服务。

Modbus 服务器必须：

——将一个应用对象映射成可读和可写的 Modbus 对象，以便获得或设定应用对象的属性；

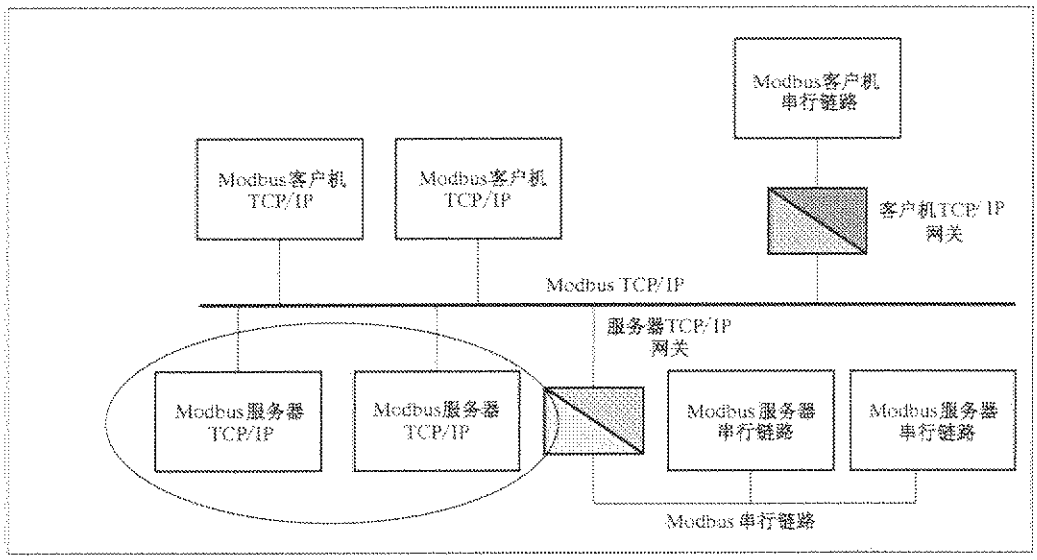


图 16 Modbus 服务器单元

——提供一种对应用对象启动服务的方法。

在运行过程中，Modbus 服务器必须分析接收到的 Modbus 请求，处理所需的操作，返回 Modbus 响应。

注：应用对象和后端接口的服务获得基于功能码的请求数据，由用户负责解决。

6.4.2.1 Modbus 服务器设计

Modbus 服务器设计取决于如下两个方面：

——对应用对象的访问类型(对属性的简单访问或对服务的高级访问)；

——Modbus 服务器与用户应用之间交互作用的类型(同步或异步)。

图 17 描述了服务器进行的主要处理过程，以便获得来自 TCP 管理的 Modbus 请求，然后，分析请求、处理所需的操作，返回 Modbus 响应。

如图 17 所示：

——Modbus 服务器本身可以立即处理一些服务，而无需直接与用户应用交互作用；

——有些服务还可能需要与被处理的用户应用进行显式交互操作；

——有些高级服务需要调用称为 Modbus 后端服务的特定接口。例如：可能根据用户应用层协议，使用若干个 Modbus 请求/响应事务处理来启动用户应用服务。后端服务负责所有单个 Modbus 事务处理的正确进行，以便执行全局用户应用服务。

在下列各章节中给出更完整的描述。

Modbus 服务器可以同时接受、处理多个 Modbus 请求。服务器可以同时接受 Modbus 请求的最大数量是 Modbus 服务器的主要特性之一。这个数量取决于服务器的设计以及它的处理和存储能力。这个实现参数被称为“NumberMaxOfServerTransaction”，对它必须作为 Modbus 服务器的一个特性来描述。根据设备的能力，它的取值范围可为：1～16。

“NumberMaxOfServerTransaction”参数对 Modbus 服务器的操作和性能有非常显著的影响。更为重要的是，所管理的并发 Modbus 事务处理的数量可能影响服务器对 Modbus 请求的响应时间。

6.4.2.2 Modbus PDU 校验

图 18 描述了 Modbus PDU 校验操作。

Modbus PDU 校验功能首先是分析 MBAP 报文头。必须校验协议标识符字段：

——如果与 Modbus 协议类型不同，那么就简单地废除这个指示。

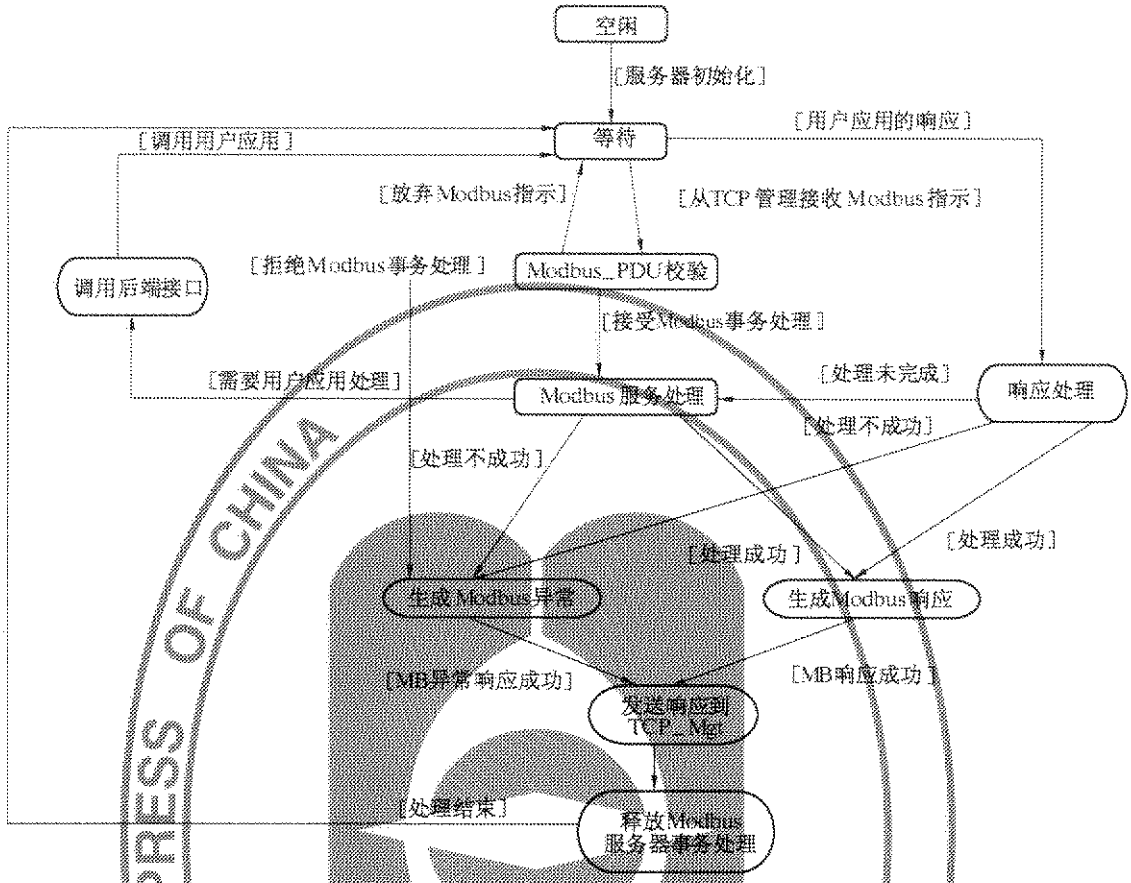


图 17 处理 Modbus 指示操作图

——如果是正确的(=Modbus 协议类型;值为 0x00),则建立了一个 Modbus 事务处理实例。

服务器可以实例化的 Modbus 事务处理的最大数量由参数"NumberMaxOfTransaction"(系统或配置参数)来定义。

在不可用的事务处理情况下,服务器生成一个 Modbus 异常响应(异常码 6:服务器忙)。

如果事务处理是可用的,它将被启动,以便存储下列信息:

——用于发送指示的 TCP 连接标识符(由 TCP 管理给出);

——Modbus 事务处理 ID(在 MBAP 报文头中给出);

——单元标识符(在 MBAP 报文头中给出)。

然后,分析 Modbus PDU。首先分析功能码:

——当无效时,生成 Modbus 异常响应(异常码 1:无效功能);

——如果接受功能码,服务器启动一个"Modbus 服务处理"操作。

6.4.2.3 Modbus 服务处理

见图 19。

根据后面示例中描述的设备软件和硬件结构,可以用不同的方式进行所要求的 Modbus 服务处理:

——在一个紧凑型设备或单线程体系结构内,Modbus 服务器可以直接访问用户应用数据,服务器自身可以"本地"处理所要求的服务,而无需调用后端服务。

根据 GB/T 19582.1—2008 进行这种处理。在出现错误的情况下,生成 Modbus 异常响应。

——在一个模块化的多处理器的设备或多线程体系结构中,"通信层"和"用户应用层"是两个独立的实体,通信实体可以完整地处理一些不重要的服务,而其他的服务需要应用后端服务与用户应用实体协调完成。

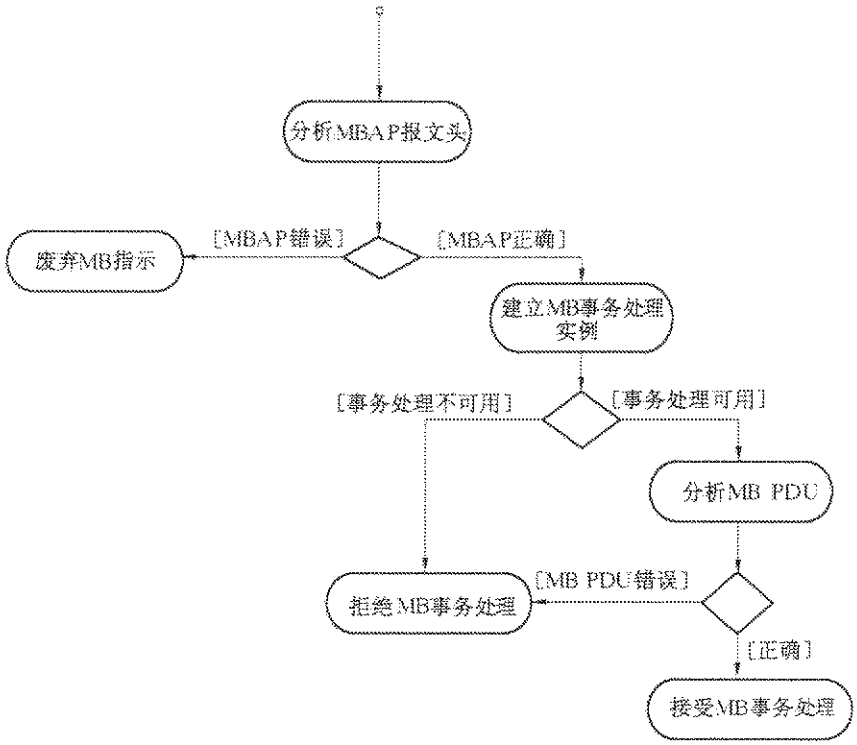


图 18 Modbus PDU 校验操作图

为了实现与用户应用的交互作用，Modbus 后端服务必须执行所有适当的机制，以便处理用户应用的事务处理，并且正确管理用户应用调用和相应的响应。

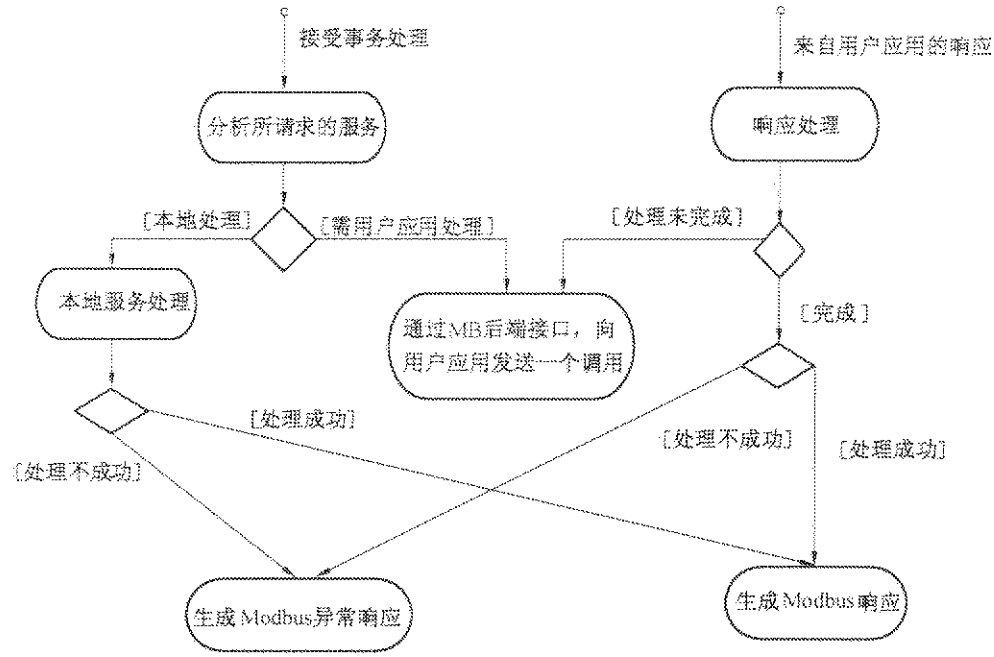


图 19 Modbus 服务处理操作图

6.4.2.4 用户应用接口(后端接口)

在 Modbus 后端服务中，可以使用几种策略来完成工作，尽管从用户网络吞吐量、接口带宽使用、响应时间、设计工作量的角度，这几种策略是不尽相同的。

Modbus 后端服务将对用户应用使用适当接口：

——或基于串行链路的物理接口，或双口 RAM 方案，或一条简单的 I/O 连线，或由操作系统提供的基于报文传输服务的逻辑接口。

——到用户应用的接口可以是同步的或异步的。

Modbus 后端服务还将使用适当的设计模式来获取/设定对象属性或触发服务。在某些情况下，一个简单的“网关模式”就足够了。在某些其他情况下，从简单的历史交换表到更复杂的重复机制中，设计者将必须实现带有高速缓存器的“代理服务器模式”。

Modbus 后端服务负责实现协议的转换，以便与用户应用进行交互作用。因此，它必须具有相应机制来实现报文的分拆/重组、数据一致性保证以及所有需要的同步等功能。

6.4.2.5 **Modbus 响应的生成**

一旦处理请求，Modbus 服务器就必须使用适当的 Modbus 服务器事务处理生成一个响应，并且必须将响应发送到 TCP 管理软件。

根据处理结果，可以生成两类响应：

——肯定的 Modbus 响应：响应功能码＝请求功能码；

——Modbus 异常响应：目的是为客户机提供与处理过程检测到的错误有关的信息；响应功能码＝请求功能码＋0x80；提供异常码来说明出错的原因，见表 4。

表 4 Modbus 异常响应

异常码	Modbus 名称	注 释
01	非法的功能码	服务器未知的功能码
02	非法的数据地址	与请求有关
03	非法的数据值	与请求有关
04	服务器故障	在执行过程中，服务器故障
05	确认	服务器接受服务调用，但是需要相对长的时间完成服务。因此，服务器仅返回一个服务调用接收的确认
06	服务器忙	服务器不能接受 Modbus 请求 PDU。客户机应用负责决定是否和何时重发请求
0A	网关故障	网关路径是无效的
0B	网关故障	目标设备没有响应。网关生成这个异常信息

Modbus 响应 PDU 必须以 MBAP 报文头作前缀，使用事务处理文本中的数据生成 MBAP 报文头。

——单元标识符

当在所收到的 Modbus 请求中给出单元标识符时，拷贝这个单元标识符，并将其存储在事务处理的文本中。

——长度

服务器计算 Modbus PDU 和单元标识符字节的大小。在“长度”字段中设置这个值。

——协议标识符

设置协议标识符字段为 0x0000(Modbus 协议)，在所收到的 Modbus 请求中给出协议标识符。

——事务处理标识符

设置这个字段为“事务处理标识符”值，它与初始请求有关，并由相关的事务处理存储。

利用事务处理存储的 TCP 连接必须对相应的 Modbus 客户机返回 Modbus 响应。在发送响应后，必须释放相应的事务处理。

7 实现指南

本章的目的是提出一个实现报文传输服务的示例。

下面所描述的模型可用作客户机或服务器实现 Modbus 报文传输服务过程的指南。

注：报文传输服务的实现由用户负责。

7.1 对象模型图

见图 20。

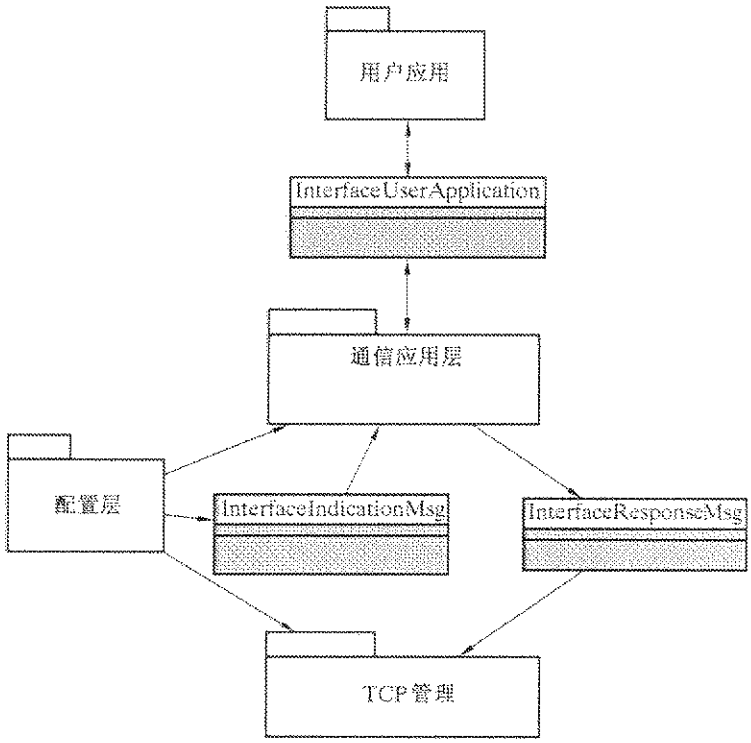


图 20 Modbus 报文传输服务对象模型图

四种主要程序包构成对象模型图：

——配置层，它配置和管理其他程序包组件的操作模式。

——TCP 管理，它提供 TCP/IP 栈和管理 TCP 连接的通信应用层连接的接口。这指的是套接字接口的管理。

——通信应用层，它由在一侧的 Modbus 客户机和在另一侧的 Modbus 服务器组成。该程序包和用户应用链接。

——用户应用，它和设备应用相对应，它完全与设备有关，因此在本部分中不予讨论。

本模型与实现的选择无关，例如：OS(操作系统)类型和存储管理等。为保证这种不相关性，在 TCP 管理层和通信层之间以及在通信层和用户应用层之间使用通用接口层。

用户用不同的实现方法实现该接口：两项任务之间的管道、共享存储器、串行链路接口以及程序调用等。

为定义下面的实现模型，进行下列假设：

——静态存储器管理；

——服务器的同步处理；

——处理有关所有套接字接收的任务。

7.1.1 TCP 管理程序包

见图 21。

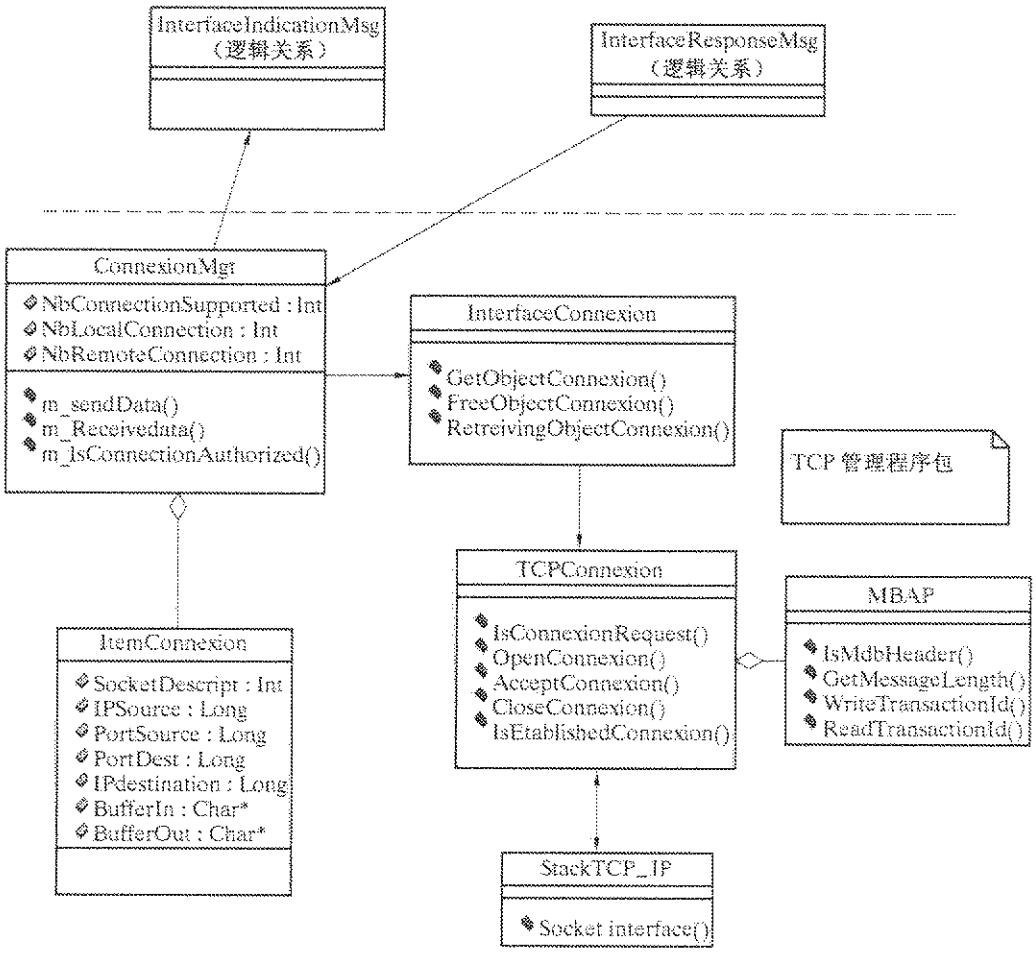


图 21　Modbus TCP 管理程序包

TCP 管理程序包包括下列类：

CInterfaceConnexion：该类的作用是管理用于连接的存储池。

CItemConnexion：该类含有描述连接所需要的所有信息。

CTCPConnexion：该类提供自动管理 TCP 连接的方法(CStackTCP_IP 提供接口套接字)。

CconnexionMngt：该类管理所有连接，并通过 CInterfaceindicationMsg 和 CInterfaceRosponseMsg 向 Modbus 服务器/Modbus 客户机发送请求/响应。该类还处理连接打开的访问控制。

CMBAP：该类提供读/写/分析 Modbus MBAP 的方法。

CStackTCP_IP：该类执行套接字服务并提供栈的参数配置。

7.1.2　配置层程序包

见图 22。

配置层程序包包括下列类：

TConfigureObject：该类将对每一个其他组件配置所需的全部数据进行分组。通过 COperatingMode 类中的 m_Configure 方法填充这个结构。每个需要配置的类从这个对象中获取自己的配置数据。配置数据与实现有关。因此，提供该类的属性表作为一个示例。

COperatingMode：该类的作用是填充 TConfigureObject(根据用户的配置)和管理下述所描述的类的操作模式。

——CModbusServer

——CModbusClient

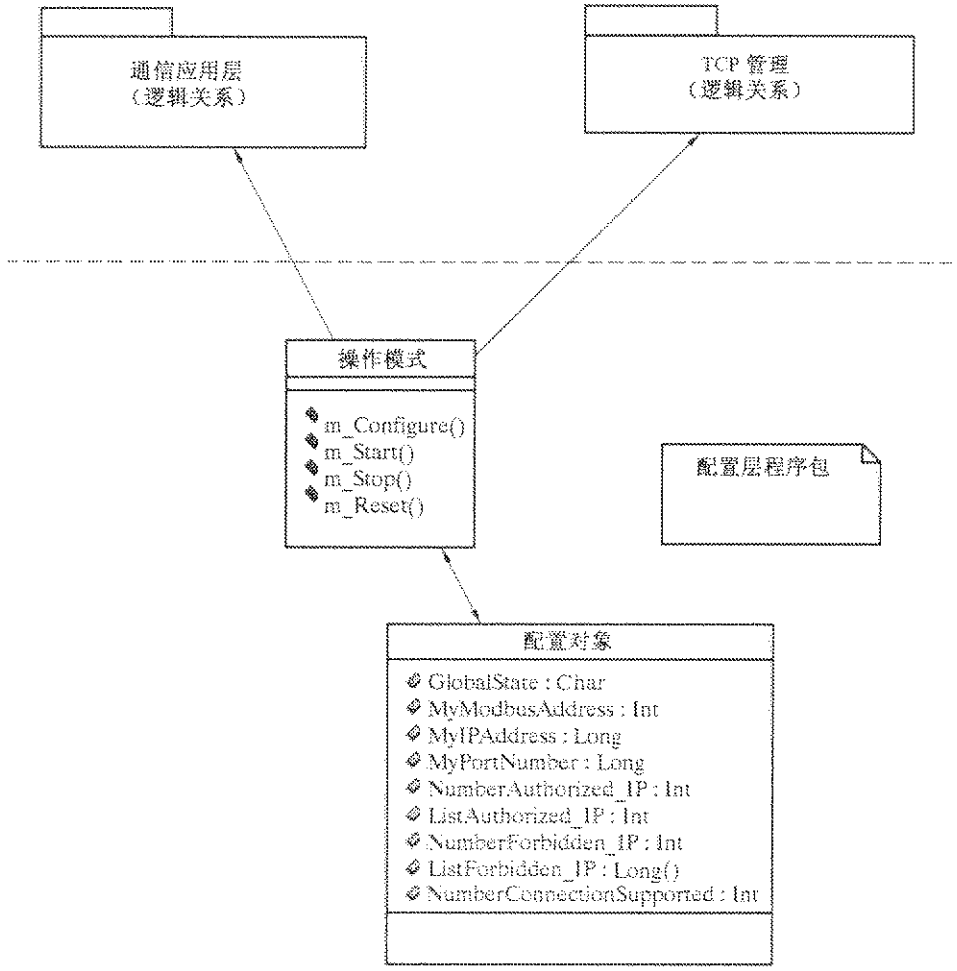


图 22　Modbus 配置层程序包

——CConnexionMngt

7.1.3　通信层程序包

见图 23。

通信应用层程序包包括下列类：

CModbusServer：从类 CInterfaceIndicationMsg 中接收 Modbus 询问（通过 m_ServerRecievingMessage 方法）。该类的作用是根据询问（从网络进入）生成 Modbus 响应或 Modbus 异常。该类实现 Modbus 服务器的状态图。仅当类 COperatingMode 发送了用户配置和正确的操作模式后，才能生成响应。

CModbusClient：从类 CInterfaceUserApplication 中读 Modbus 询问，客户机的任务是通过 m_ClientReceivingMessage 方法接收询问。该类实现 Modbus 客户机的状态图，并且管理链接响应和询问的事务处理（来自网络的）。仅当类 COperatingMode 发送了用户配置和正确的操作模式后，才能通过网络发送询问。

CTransaction：该类实现管理事务的方法和结构。

7.1.4　接口类

CInterfaceUserApplication：该类表示与用户应用的接口，它提供两种访问用户数据的方法。在实际的实现中，根据硬件和软件设备的能力，用不同的方式实现这种方法（相当于一个终端驱动器、访问 PCMCIA 的示例、共享存储器等）。

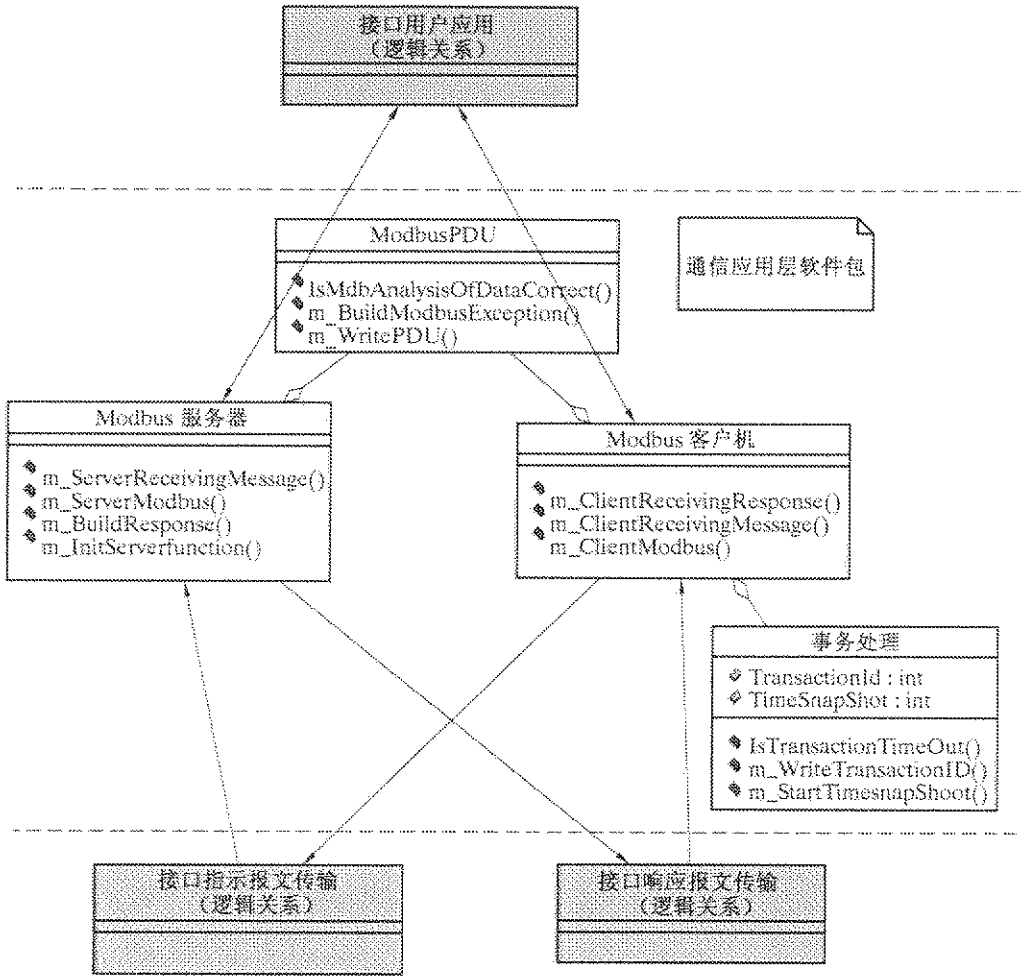


图 23　Modbus 通信应用层程序包

CInterfaceIndicationMsg：该接口类用来从网络向 Modbus 服务器发送询问，以及从网络向客户机发送响应。该类为 TCPManagement 和“通信应用层”程序包（来自网络的）提供接口。该类的实现与设备有关。

CInterfaceResponseMsg：该接口类用于从服务器接收响应，以及从客户机向网络发送询问。该类为“通信应用层”程序包和 TCPManagement（向网络）提供接口。该类的实现和设备有关。

7.2　实现类的图

图 24 是一个完整的实现方案。

7.3　序列图

图 25 和图 26 描述了两个序列图，以便说明客户机 Modbus 事务处理和服务器 Modbus 事务处理。

为更好地理解客户机序列图，通用说明如下：

第一步：来自用户应用的读询问（通过 m_Read 方法）。

第二步：“客户机”任务接收 Modbus 询问（通过 m_ClientReceivingMessage 方法）。这是客户机的进入点。为了使询问和相应的响应相互协调，当获得询问时，客户机使用事务处理资源（类名：CTransaction）。通过调用类接口 CInterfaceResponseMsg 向 TCP_Management 发送 Modbus 询问（通过 m_ModbusRequest 方法）。

第三步：如果已建立连接，则不需要对连接做进一步操作，那么报文可以通过网络发送。否则，在可以通过网络发送报文之前，必须打开连接。

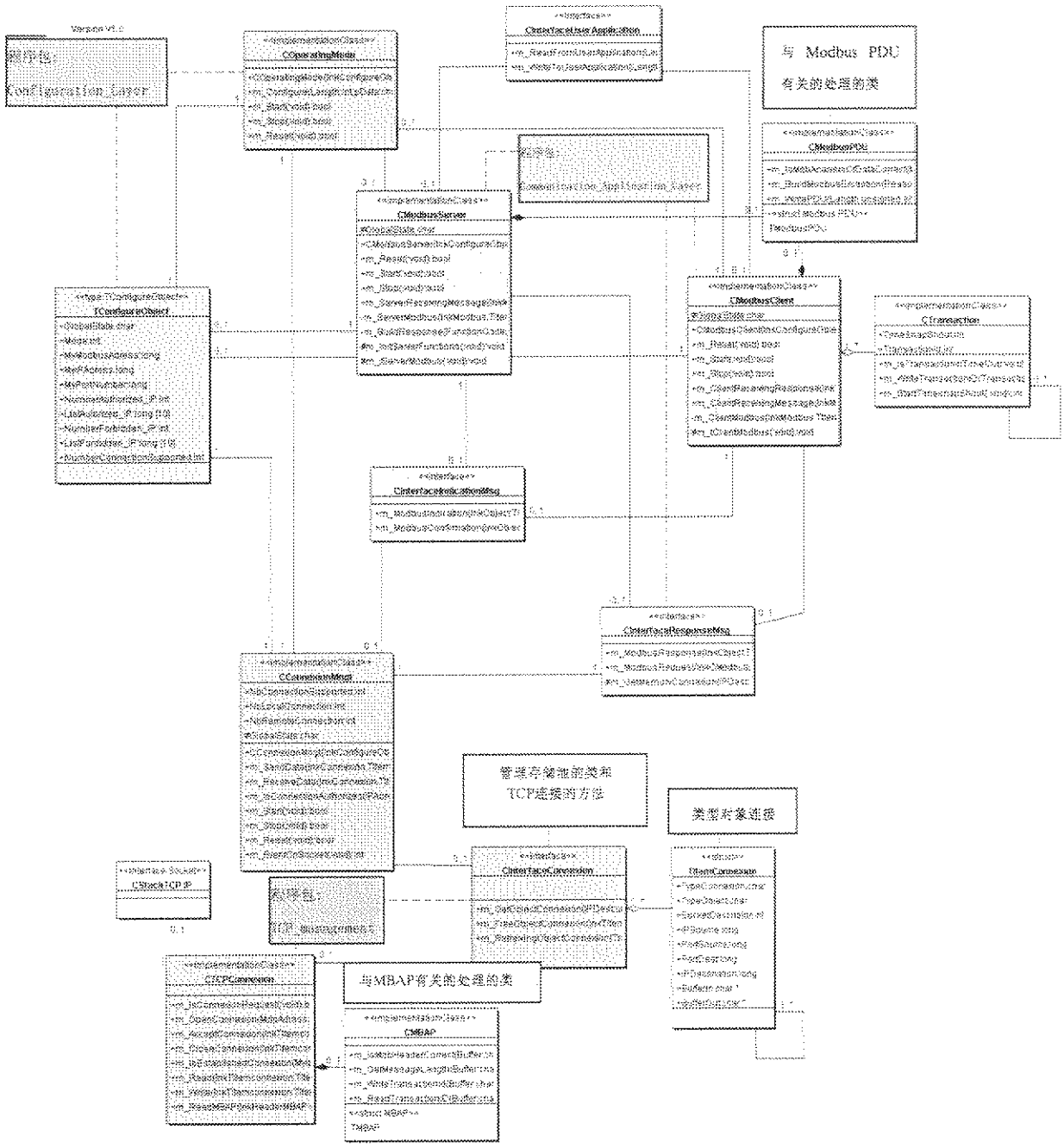


图 24 类的图

此时，客户机等待响应(来自远程服务器)。

第四步：一旦已经从网络得到响应，那么 TCP/IP 栈接收数据(通过隐性地调用 m_EventOnStack 方法)。

如果已建立连接，那么读 MBAP 以恢复连接对象(连接对象提供存储资源和其他信息)。

读来自网络的数据，通过类接口 CInterfaceIndicationMsg 向客户机任务发送证实(通过 m_Modbusconfirmation 方法)。客户机任务接收 Modbus 证实(通过 m_ClientReceivingResponse 方法)。

最后，向用户应用写入响应(通过 m_Writedata 方法)，并且事务处理资源被释放。

图 25 是 Modbus 服务器交换的示例。

为更好地理解服务器序列图，通用说明如下：

第一步：客户机已通过网络发送了询问(Modbus 询问)。TCP/IP 栈接收数据(通过隐性地调用 m_

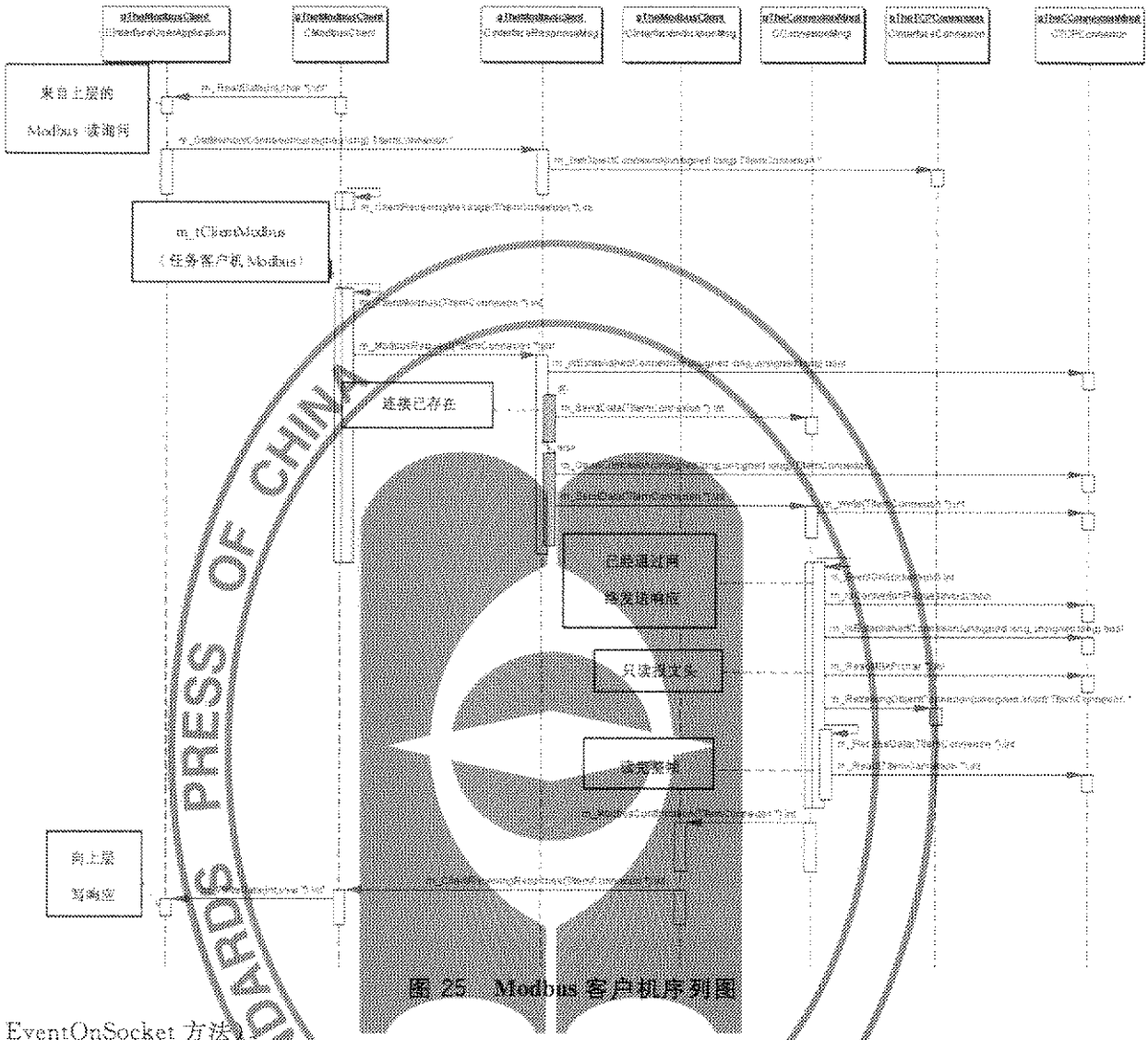


图 25 Modbus 客户机序列图

EventOnSocket 方法)。

第二步:询问可能是连接请求,也可能不是(通过 m_IsConnexionRequest 方法)。

如果询问是连接请求,那么分配连接对象及用于接收和发送 Modbus 帧的缓存区(通过 m_GetObjectConnexion 方法)。然后,必须检查和接受连接访问控制(通过 m_AcceptConnexion 方法)。

第三步:如果询问是 Modbus 请求,那么可以读出完整的 Modbus 询问(通过 m_ReceiveData 方法)。此时,必须分析 MBAP(通过 m_IsMdbHeadercorrect 方法)。通过 CInterfaceIndicationMessaging 类向服务器任务发送完整的帧(通过 m_ModbusIndication 方法)。服务器任务接收 Modbus 询问(通过 m_ServerReceivingMessage 方法)并加以分析。

在出错的情况下(不支持的功能码等),生成 Modbus 异常帧(通过 m_BuildModbusException 方法),否则生成响应。

第四步:通过 CInterfaceResponseMessaging 在网络上发送响应(通过 m_ModbusResponse 方法)。通过 m_SendData 方法处理连接对象(恢复连接描述符等)并在网络上发送数据。

7.4 类和方法的描述

7.4.1 Modbus 服务器的类

类名:CModbusServer,见表 5～表 7。

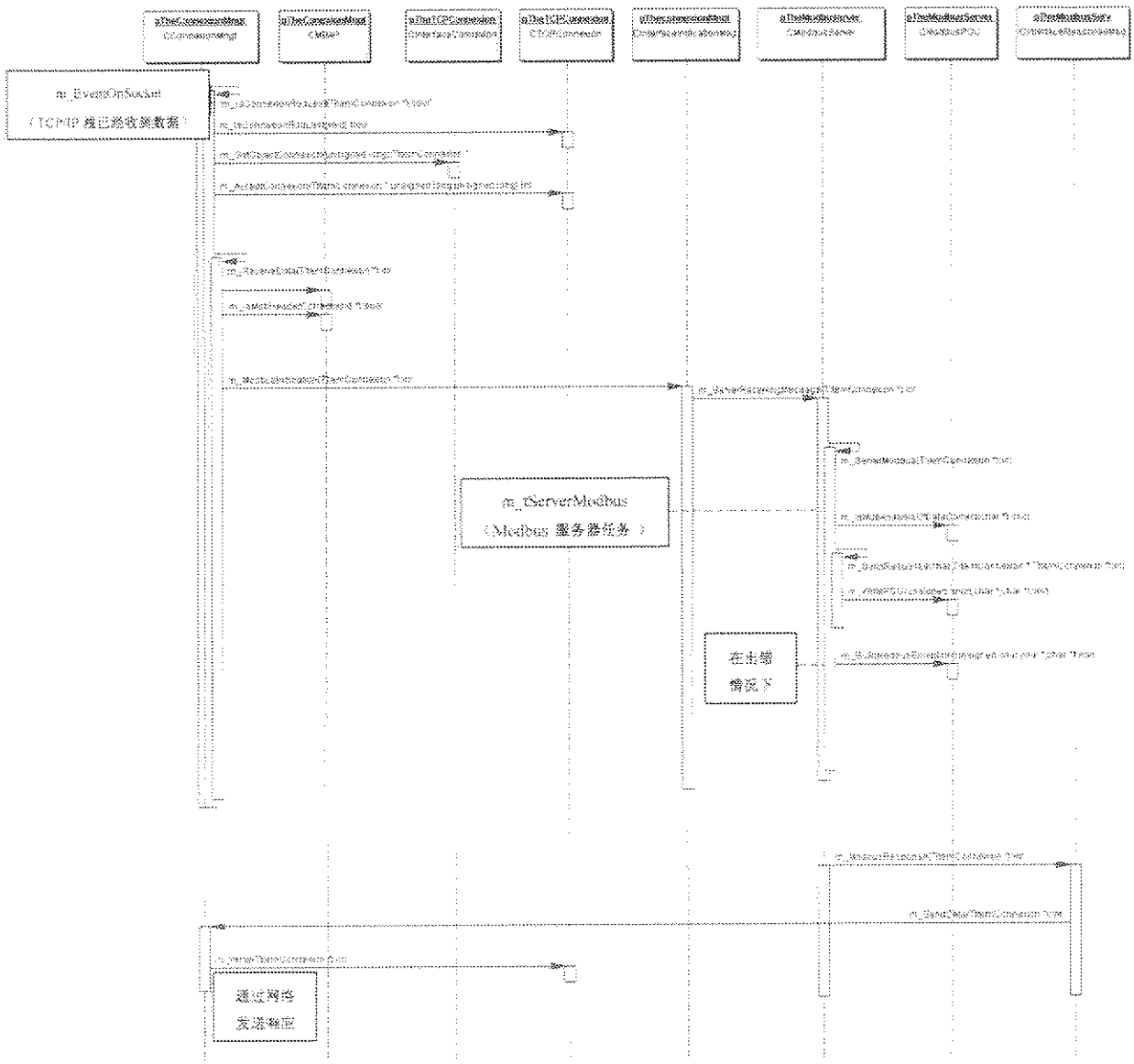


图 26　Modbus 服务器序列图

class CModbusServer

Stereotype 实现类

提供用服务器模式管理 Modbus 报文传输的方法

表 5　字段综述

字段综述	
protected char	GlobalState Modbus 服务器的状态

表 6 构造函数综述

构造函数综述	
CModbusServer(TConfigureObject * lnkConfigureObject) 构造函数:生成内部对象	

表 7 方法综述

方法综述	
protected void	m_InitServerFunctions(void) 为填充功能阵列"m_ServerFunction"由构造函数调用的功能
bool	m_Reset(void) 复位服务器的方法,如果被复位,那么返回"真"
int	m_ServerReceivingMessage(TItemConnexion * lnkModbus) 与 CindicationMsg:;m_ModbusIndication 的接口,用于从网络接收询问,如有问题,返回负值
bool	m_Start(void) 启动服务器的方法,如果被启动,返回"真"
bool	m_Stop(void) 停止服务器的方法,如果停止,返回"真"
protected void	m_tServerModbus(void) 服务器 Modbus 任务……

7.4.2 Modbus 客户机类

类名:CModbusClient,见表 8~表 10。

class CModbusClient
提供用客户机模式管理 Modbus 报文传输的方法
Stereotype 实现类

表 8 字段综述

字段综述	
protected char	GlobalState Modbus 客户机的状态

表 9 构造函数综述

构造函数综述	
CModbusClient(TConfigureObject * lnkConfigureObject) 构造函数:生成内部对象,将变量初始化为 0	

表 10 方法综述

方法综述	
int	m_ClientReceivingMessage(TItemConnexion * lnkModbus) 典型地为接收来自应用层的报文提供接口: 调用 CInterfaceUserApplication:;m_Read 用于读数据; 调用 CInterfaceConnexion:;m_GetObjectConnexion 用于获取事务处理的存储器, 如果有问题,那么返回负值

表 10(续)

方法综述	
int	m_ClientReceivingResponse(TItemConnexion * lnkTItemConnexion) 与 CindicationMsg::m_Confirmation 的接口,用于从网络接收响应, 如果有问题,那么返回负值
bool	m_Reset(void) 复位组件的方法,如果复位,返回"真"
bool	m_Start(void) 启动组件的方法,如果被启动,返回"真"
bool	m_Stop(void) 停止组件的方法,如果被停止,返回"真"
protected void	m_tClientModbus(void) 客户机 Modbus 任务……

7.4.3 接口类

7.4.3.1 接口指示类

类名:CInterfaceIndicationMsg,见表 11。

直接已知的子类

CConnexionMngt

---

class CInterfaceIndicationMsg

从 TCP_Management 向 Modbus 服务器或客户机发送报文的类

Stereotype 接口

表 11 CInterfaceIndicationMsg 方法综述

方法综述	
int	m_ModbusConfirmation(TItemConnexion * inkObject) 接收进入响应和调用客户机的方法:可通过参量值、报文队列、远程过程调用等方法
int	m_ModbusIndication(TItemConnexion * lnkObject) 读进入 Modbus 询问和调用服务器的方法:可通过参量值、报文队列、远程过程调用等方法

7.4.3.2 接口响应类

类名:CInterfaceResponseMsg,见表 12。

直接已知的子类:

CModbusClient、CModbusServer

---

class CInterfaceResponseMsg

从客户机或服务器向 TCP_Management 发送响应或询问的类

Stereotype 接口

表 12 CInterfaceResponseMsg 方法综述

方法综述	
TitemConnexion *	m_GetMemoryConnexion(unsigned long IPDest) 从存储池获得对象 ItemConnexion，如果存储不充足，返回值－1
int	m_ModbusRequest(TItemConnexion * lnkCModbus) 将进入 Modbus 询问客户机写入 ConnexionMngt 的方法，可通过参量值、报文队列、远程过程调用等方法
int	m_ModbusResponse(TItemConnexion * lnkObject) 从 Modbus 服务器将响应写入 ConnexionMngt 的方法，可通过参量值、报文队列、远程过程调用等方法

7.4.4 连接管理类

类名：CConnexionMngt，见表 13～表 15

---

class CConnexionMngt

管理所有 TCP 连接的类

Stereotype 实现类

表 13 字段综述

字段综述	
protected char	GlobalState 组件 ConnexionMngt 的全局状态
int	NbConnectionSupported 连接的全局数量
int	NbLocalConnection 本地客户机向远程服务器打开的连接数量
int	NbRemoteConnection 远程客户机向本地服务器打开的连接数量

表 14 构造函数综述

构造函数综述	
CconnexionMngt(TConfigureObject * lnkConfigureObject) 构造函数，生成内部对象，将变量初始化为 0	

表 15 方法综述

方法综述	
int	m_EventOnSocket(void) 唤醒
bool	m_IsConnectionAuthorized(unsigned long IPAdress) 如果授权新连接，返回“真”
int	m_ReceiveData(TitemConnexion * lnkConnexion) 与 CTCPConnexion，，write 方法的接口，用于从网络中读数据，如果有问题，返回负值
bool	m_Reset(void) 复位 ConnectionMngt 组件的方法，如果被复位，返回“真”

表 15(续)

方法综述	
int	m_SendData(TItemConnexion * lnkConnexion) 与 CTCPConnexion::read 方法的接口,用于向网络发送数据, 如果有问题,返回负值
bool	m_Start(void) 启动 ConnectionMngt 组件的方法,如果被启动,返回"真"
bool	m_Stop(void) 停止组件的方法,如果被停止,返回"真"

中华人民共和国
国家标准
基于Modbus协议的工业自动化网络规范
第3部分：Modbus协议在
TCP/IP上的实现指南
GB/T 19582.3—2008
*
中国标准出版社出版发行
北京复兴门外三里河北街16号
邮政编码：100045
网址 www.spc.net.cn
电话：68523946 68517548
中国标准出版社秦皇岛印刷厂印刷
各地新华书店经销
*
开本 880×1230 1/16 印张 2.5 字数 65千字
2008年5月第一版 2008年5月第一次印刷
*
书号：155066·1-31330 定价 28.00元

GB/T 19582.3-2008