基于 CAN 的较高层协议和子协议

作者：Pro.Dr.-Ing.K.Etschberger

概述：本文主要介绍了几个基于 CAN 的较高层协议：CAL/CANopen、DeviceNet、SDS。并且对这几个较高层协议的重要功能作了一些比较，使读者更深入地了解这几个较高层协议。

1. 简介

作为通用、有效、可靠及经济的平台，CAN 协议已经广泛地受到了欢迎。它可以使用于汽车系统、机械、技术设备和工业自动化里几乎任何类型的数据通信。由于成熟的标准化较高层协议和子协议，基于 CAN 的开放式自动化技术成功地在分布式自动化系统市场里发挥了它的竞争优势。基于 CAN 的系统之所以取得巨大成功，其中一个主要的原因显然要归功于 CAN 协议的许多特殊功能，特别是它面向产生者－使用者的数据传送原理以及多主机能力。从技术的观点看，这些属性使 CAN 协议在分布式系统里颇具魅力。

当提到“CAN 标准”或“CAN 协议”时，我们需要理解 ISO 11898 标准中[1]和[2]各自的功能性。这个标准包括由 ISO 参考模型的物理层（第一层）和数据链路层（第二层）。其中，第一层负责譬如物理信号传输，译码，位时序和位同步等的功能，而第二层负责像总线仲裁、信息分段以及数据安全、数据确认、错误检测、信号传输和错误控制的功能。CAN 标准没有规定媒体的连接单元以及其驻留媒体，也没有规定应用层。

CAN 协议的第二层给用户提供了两种无连接类型的发送服务：

- 一个 CAN 信息的不响应发送
- 一个 CAN 信息的不响应远程请求

无连接发送指的是在执行一个信息发送或请求之前不必建立数据链路连接。CAN 芯片以不同类型的对象过滤和对象缓冲方式支持信息的接收。根据 CAN V2.0 规范的一第二层 CAN 数据信息，其定义取决于信息标识符、标准 / 扩展格式指示、数据长度和所要发送的数据。

由于 CAN 协议没有规定信息标识符的分配，因此可以根据不同应用使用不同的方法。所以，在设计一个基于 CAN 的通讯系统时，确定 CAN 标识符的分配非常重要，标识符的分配和定位也是较高层解决手段的其中一个主要的项目。

较高层的要求

实际上，即使在执行一些非常简单的基于 CAN 的分布式系统时，除了基本的第二层服务之外，还要求或希望有更多的功能，如发送长于 8 字节的数据块，响应或确定数据传送，标识符分配，网络启动或监控节点。由于这些附加的功能直接支持应用过程，所以它可以被认作“应用层”。如果正确执行，则应用层以及相应的应用层接口的简介（子协议）为通讯和应用过程提供一个清晰定义的分界以便把它们区分开来。

因为 CAN 协议提供有许多非常独特的功能，因此，大部分已知的较高层协议通过对数据链路层的服务提供直接访问而将这些不同的功能保留给应用层的用户（基本功能不需要额外的上层协议）。

特别在工业自动化应用中，越来越需要一个开放、标准化的较高层协议，这个协议支持不同生产厂家设备的互用性和可交换性。因此，要求有标准设备模型的规范,即，基本功能性的“标准设备”和“标准应用”的规范，以作为对标准化应用层的补充。

接下来，首先对主要的较高层协议的解决方案做一简短的概述，然后介绍不同解决方案的主要项目。在本论文里，只可能评价某些主要的方面。这里要着重介绍的是工业自动化中的较高层协议。

2. 基于 CAN 的较高层协议的调查

除了很多特殊的解决方法之外，还有许多 CAN 网络的普遍应用，它们有不同的目的和不同要求。鉴于此，

基于 CAN 较高层协议的几个主要标准在当今已经成型。根据不同的要求，这些解决方案在关于作用范围和性能方面有着显著的不同之处。

通过应用直接采纳的应用层，其标准广泛地得到了接受。代表它们的是 CAL[3]和 OSEK[4]。CAL 可以被认为是不依赖于应用的应用层，它适用于各种基于 CAN 且直接使用应用层服务的应用里，而 OSEK-Com/Net 标准则具有应用层和网络管理的功能性，主要用于汽车网络中。

CAL（CAN Application Layer）发布于 1993 年，是 CiA 的首批的效力条款之一。CAL 为基于 CAN 的分布式系统[6]的实现提供了一个不依赖于应用、面向对象的环境。它为通讯、标识符分布、网络和层管理提供了对象和服务。CAL 的主要应用在基于 CAN 的分布式系统，这个系统不要求可配置性以及标准化的设备建模。CAL 的其中一个子集是作为 CANopen 的应用层。因此，CANopen 的设备可以用在指定应用的 CAL 系统。

OSEK/VDX 是汽车行业里的一个联合（开发）项目，其目的是为汽车的分布式系统提供一工业标准以便具有开放式的结构。这个标准包括一个实时操作系统的定义，软件接口的定义以及一个通讯和网络管理系统的定义。OSEK 操作系统提供有服务以便于任务管理和同步、中断管理、警告和错误处理。这个操作系统的主要目的是规定一个通用平台以集成不同厂家的软件模块。由于想把操作系统使用在任何类型的控制单元中，因此它必须支持大多数硬件的实时应用（time-critical applications）。OSEK 通讯规定[6]定义了一个硬件以及总线系统独立的应用接口。本地和远程任务的通讯是由操作系统通过“信息对象”执行的。这里要区分两种信息：“状态信息”和“事件信息”。状态信息通常表示大多数系统变量（没有缓冲）的实际状态，并通过事件信息报告事件。因此，使用者必须处理每个信息。这两种类型的信息可以在点对点和多播传送方式中使用，传送模式有周期性、事件驱动和周期性 / 事件驱动。传输层服务额外地向不响应不分段的数据层服务提供响应的分段的数据传送。

由于汽车内系统的通讯要求非常高，为了确保通讯网络的安全性和可靠性提出了一个完善的网络管理系统[7]。系统使用“节点监控”，即每个节点都被网络中的所有其他节点监控（直接监控）。被监控的节点根据一个专门和统一的算法发送一个 NM 信息。直接节点监控要求网络范围内的 NM 信息要同步。因此，这里使用了一个逻辑环（logical ring）。任何节点都必须能够将 NM 信息发送到所有其他节点并从其他节点接收信息。

如果觉得直接监控对于一个设备来说太复杂，可以使用“间接监控”原则。这个原则基于应用信息的观察，并受限制于定期发送信息的节点。这种类型的节点可能被一个或更多的其他节点监控。

还有一个十分不同的开放式系统，其解决方案由 SAE J1939[11]标准提供。这个标准是由**汽车工程师重型汽车社团和总线部门**为了向电子系统提供一个开放的互联系统而定义的。这个系统主要的应用范围是面向路面或非路面设计的轻、中、重型机车，以及为获取部件而专有的静止应用场合。机车包括行驶在公路上的卡车和拖车、建筑装备、农业装备和船用仪器。J1939 标准是基于 29 位信息标识符的用法。这个标准化的信息标识符使得 8 个优先级别、预定义信息类型、指定目标的通讯和广播各有差异。J1939/7x 定义了标准的汽车内信息和诊断信息。因此，数据类型、数据的范围、数据重复率等，以及相应的参数组号码，它们确定各自的信息标识符。此外，J1939 还定义了信息映射如何到参数组的 CAN 数据区。

工业应用中，主要代表开放式分布系统的标准是 CANopen[8]、DeviceNet[9]和 SDS[10]。

开放式分布系统标准的工业应用包括工业自动化中由工业器件（传感器、执行器、控制器、人机接口）组成的低层网络。这种应用主要要求的有：可配置性、灵活性和可扩展性。为了保持生产厂商的独立性，必须以“设备子协议（Device Profile）”的形式定义器件的功能性。因此，那种类型的通讯系统解决方案提供了一个

完整的通讯框架和系统服务、设备建模、以及设施，其目的是便于系统配置和设备参数化。

CANopen 标准是由 CiA（CAN-in Automation）旨在解答 EU-研究程序结果的一组成员编制的。CANopen 在通讯和系统服务以及网络管理的方面使用了 CAL（CAN Application Layer）子集。设备建模是借助于对象目录而基于设备功能性的描述。这种方法广泛地符合于其他现场总线（Interbus-S，Profibus）使用的设备描述形式。标准设备以“设备子协议（Device Profile）”的形式规定。CANopen 标准由 CiA 同行机构集团支持，设备子协议（Device Profile）由 CiA 中专门的同行机构集团规定。

DeviceNet™ 是由 Allen-Bradley 开发的非常成熟的开放式网络。它根据抽象对象模型来定义。这个模型是指可用的通讯服务和一个 DeviceNet 节点的外部可见行为。DeviceNet 标准由一个独立的供应者组织（ODVA，Open DeviceNet Vendor Association）管理，这个组织也同时广泛地支持 DeviceNet 的市场。相应的设备子协议（Device Profile）规定同类设备的行为。

SDS™（Smart Distribution Systems）是由 Honeywell Micro Switch 开发的一个开放式网络标准。由于它基于特定的应用层协议，因此定义了一个面向 对象的等级设备模型以便在 SDS 设备之间建立互用性。SDS 是特别为分布式二进制传感器和执行器设计的。

3. 基于 CAN 的较高层协议的主要项目

以下将工业自动化的主要解决方案----CAL/CANopen, DeviceNet 和 SDS 综合起来做一比较（评估），比较的标准是以下基于 CAN 较高层协议的主要项目：

- 信息标识符分配系统
- 过程数据交换的方法
- 点对点通讯信道
- 建立过程数据信息连接的方法
- 网络管理
- 设备建模和设备子协议（Device Profile）的原则

3.1 信息标识符分配系统

由于 CAN 信息的标识符决定了信息相关的优先权和信息的等待时间，信息标识符分配的方法被认为是基于 CAN 的系统的主要结构元素。它同时也影响了信息滤波适用性、合理的通讯结构适用性和标识符使用的效率。

由于标识符的分配方法很不相同，因此已经把不同原则的选取考虑在不同系统的解决方案中。CAL 和 CANopen，它们除了保留一些用于管理目的的标识符外，不会因为整个系统结构而采用预定义的标识符。但是，DeviceNet 和 SDS 要使用预定义的标识符。

CAL/CANopen 提供一通用的标识符库（源），可根据设备的需求应用于手动或自动地定位（分配）标识符的所有设备和一中间实例。这样，就可以完全由系统设计者和综合者决定标识符的使用以及数据通讯系统的实时行为。由于几乎整组信息标识符都可以分布，所以可以最大限度地使用这些可用的标识符。

图 3.1-1a 和图 3.1-2a 是关于通用 CAL/CANopen 系统的标识符分配方案。图 3.1-1b 和图 3.1-2b 是关于最小配置的 CANopen 系统的标识符分配和预定义的一组信息。一般可以使用 1760 个信息标识符。由于在基

于 CAL 的系统中可能要访问的节点多达 256 个，所以保留了 256 个信息作为节点保护；在 CANopen 的系统中，可能访问的节点为 128 个，所以只保留 128 个信息作为节点保护。

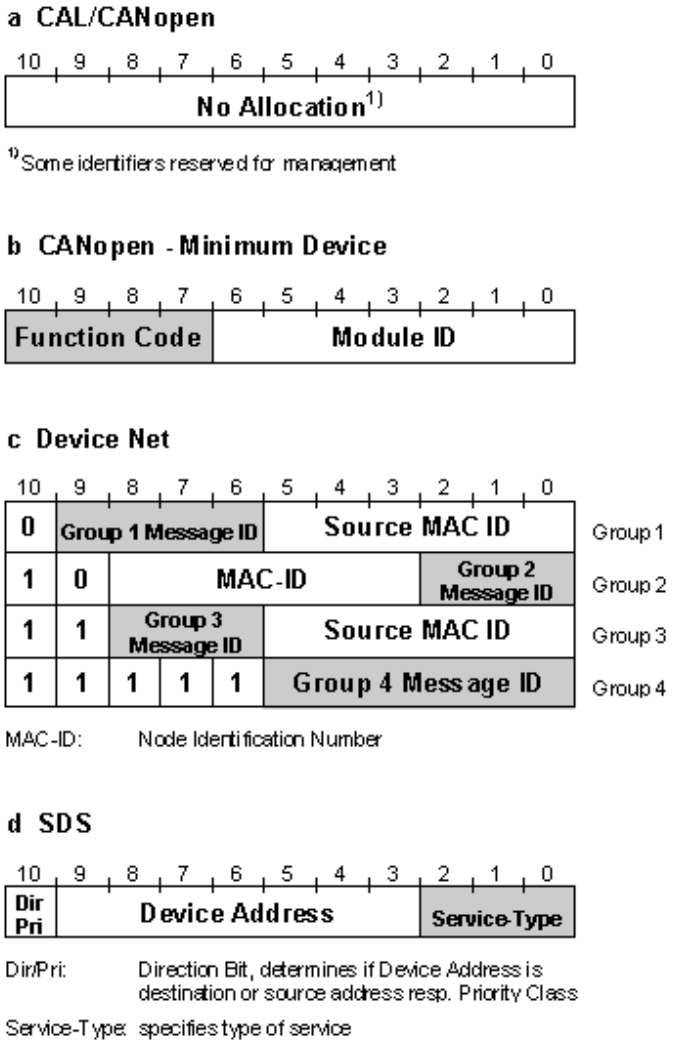


图 3.1-1

在最小系统配置中，CANopen 规定了一个面向设备的标识符分配方案，这种方法可以供 127 台设备默认连接到一台主设备上。通过 4 位的功能代码可以区分某些情况的 16 种基本功能。这些情况分别是是：2 个数据过程信道的接收和发送、一个点对点信道、节点状态控制、节点保护、紧急情况通报以及接收同步信息和有时间标记的信息。由于信息的优先级由它的功能决定，所以功能代码位于信息标识符的最高几位。（图 3.1-2）

CANopen 最小系统的配置使 1:n 通讯结构在每一默认里得到支持。 依靠不被预定义连接组使用的标识符，可以建立设备间直接的连接。

其中一个基本的 DeviceNet 标识符分配方案是面向节点的信息标识符的所有权。对于 DeviceNet 系统最多数量为 64 个的节点，其每一个节点拥有一组出自于 3 个信息组的标识符（图 3.1-1c）。信息组 1 为每个设备的 16 个信息提供一高优先信息组；信息组 3 提供每个设备 5 个低优先级标识符。这组的标识符 / 优先权被平均分配到网络上的所有设备（图 3.1-2c）。最大数量的设备保留信息标识符，这就是说：对于少于 64 个节点的网络，不使无用节点的标识符不可用于系统中。

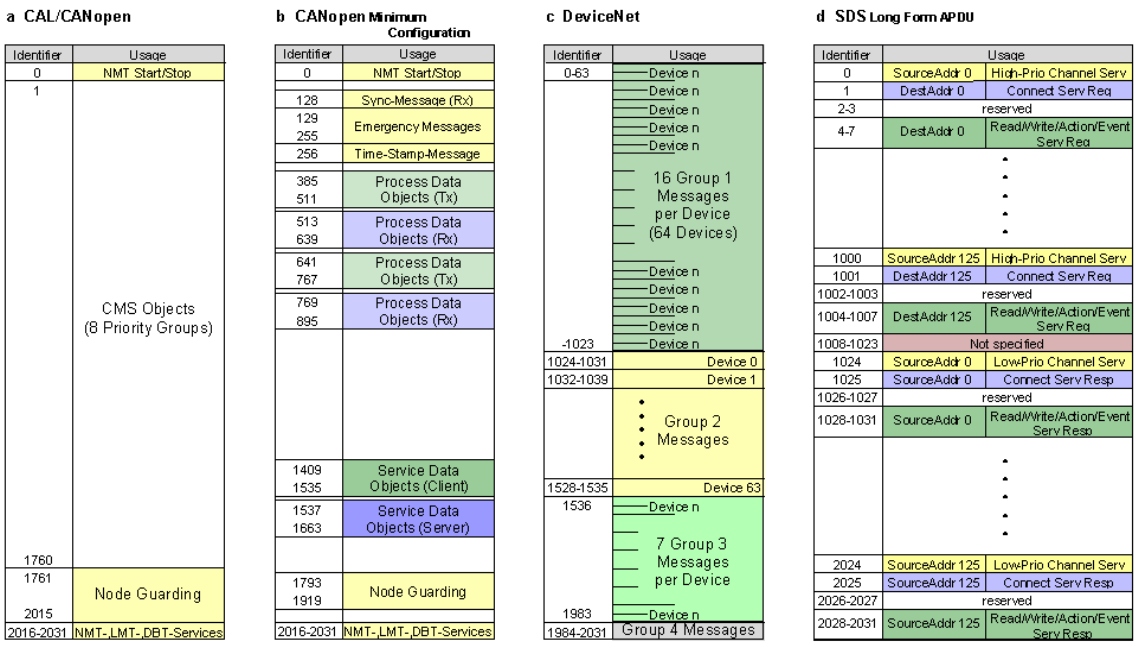


图 3.1-2

由于采用了基本 CAN 类型的控制器，信息滤波的潜能受到限制。。我们希望信息组 2 以这种受到限制的潜能去支持多个设备。因此选择了根据节点号码的滤波（功能）。这就是说，信息 2 的信息优先权主要取决于节点的号码。信息组 2 的两个信息被保留用作为管理任务（预定义的连接组的分配、复制的 MAC-ID 号）。组 2 信息的 MAC-ID 可以是目标或源地址。

IDENTIFIER BITS											IDENTITFIER USAGE
10	9	8	7	6	5	4	3	2	1	0	
0	**Group 1 Message ID**				**Source MAC-ID**						**Group 1 Messages**
0	1	1	0	1	Source MAC-ID						Slave's I/O Change of State or Cyclic Message
0	1	1	1	0	Source MAC-ID						Slave's I/O Bit-Strobe Response Message
0	1	1	1	1	Source MAC-ID						Slave's I/O Poll Response or Change of State / Cyclic Acknowledge Message
1	0	**MAC-ID**						**Group 2 Message ID**			**Group 2 Messages**
1	0	Source MAC-ID						0	0	0	Master's I/O Bit-Strobe Command Message
1	0	Source MAC-ID						0	0	1	Reserved for Master's Use - Use is TBD
1	0	Destination MAC-ID						0	1	0	Master's Change of State or Cyclic Acknowledge Message
1	0	Source MAC-ID						0	1	1	Slave's Explicit Response Messages
1	0	Destination MAC-ID						1	0	0	Master's Explicit Request Messages
1	0	Destination MAC-ID						1	0	1	Master's I/O Poll / Change of State / Cyclic Message
1	0	Destination MAC-ID						1	1	0	Group 2 Only Unconnected Explicit Request Messages
1	0	Destination MAC-ID						1	1	1	Duplicate MAC-ID Check Messages

图 3.1-3

DeviceNet 规定了一个“预定义主 / 从连接组”，以便于观察主-从系统配置的通讯。图 3.1-3 为该组的标识符分配情况。下面的信道功能是为了向基于预定义连接组的主从设备间的 I/O 与显式信息的交换提供支持：

- 显式信息信道
- 主机查询 / 改变信道的状态 / 循环
- 从机 I/O 改变信道的状态 / 循环

- 位选通信道

显式信息主要服务于设备的配置。**主机查询 / 改变状态信道**使得主机可以向设备请求 I/O 数据并把输出数据发送到从机。借助于**改变状态 / 循环**或**从机 I/O**（通过改变状态循环地触发，或通过从应用触发），从机将输入数据发送到主机。通过位选通指令，主机可以从最多为 64 个的从机中请求接收一个输入数据。由于所有的这些信息都是被响应的，所以对这些不同的功能性分配了 8 个信息标识符。如果请求获取数据的位选通没有使用一高效的标识符，则通过目的地址场在从机上（对信息进行）滤波。

图 3.1-1d 显示了 SDS 的标识符分配方法。这个方法和 DeviceNet 的组 2 信息很相似，并且完整提供了不同目的的信息预定义组。SDS 有两种信息的格式：短格式和长格式。长格式 APDU 可以是不分段或是分段的。短格式信息不包括任何数据字节，且用于一高效率的接口，是一个与单个二进制 I/O 设备进行通讯的接口。

SDS 信息的含义由下面 3 个元素决定：

- Dir/Pri 位
- 逻辑地址（7 位）
- 服务类型（3 位）

Dir/Pri 位的含义取决于信息的类型。对于短格式信息，这一位表示信息的方向（Dir=0：逻辑地址＝目的地址；Dir=1：逻辑地址＝源地址）。

对于长格式信息，Die/Pri 位是由服务类型区来定义。对于服务类型区中的读、写、行为、事件和连接，Die/Pri 位就像短格式 ADPU 一样，它根据逻辑地址区内容决定方向 。为了支持经由简单 CAN 控制器滤波的简单标识符，优先权由设备号码决定。因此，对于这些类型的服务，访问信息的所有目的地址具有比访问信息的源地址更高的优先权。

对于“信道”（多播）服务类型，逻辑地址通常是源地址，并且 Dir/Pri 位规定了两个优先级（高 / 低）。

逻辑地址区允许的逻辑地址为从 0 到 125。对于信道服务，逻辑地址总是包括源地址。

由于地址区位于信息识标识符的高位部分，所以低地址设备总是比高地址设备具有更高的优先权。

3 位的服务类型区将短格式信息和长格式信息划分为 8 种不同的服务区。长格式信息服务包括：信道、连接、写、读、行为和事件。其读、写、事件和行为服务用以读或修改一个嵌入对象的属性，报告嵌入对象发生事件，或执行对嵌入对象的行为。短格式服务信息是专门的事件和写服务，其两个服务以一种高效率的方式报告或命令一位的事件 / 行为，如“改变关闭状态”或“打开写状态”。SDS 规定 V2.0 也使连接和信道服务得到了提供。连接服务是用以在一个初始设备和一个响应设备之间建立连接。信道服务则是为请求者提供了多播和双向点对点通讯信道。

由于 SDS 用标识符说明设备的地址和服务，而不是作为唯一的数据标签，所以要用一个额外的信息来唯一标识发送的数据。这减低了协议的数据发送效率。SDS 的完整数据标签是包括逻辑设备地址、嵌入对象 ID 和属性 / 事件或行为 ID。由于除了信道服务外的所有服务都由应用确定，所以需要一个服务指定符信息区来指出这个信息是一个请求、成功、错误还是一等待响应的信息。这个额外的信息是在 CAN 数据区的头两个字节里发送（图 3.2-3）。

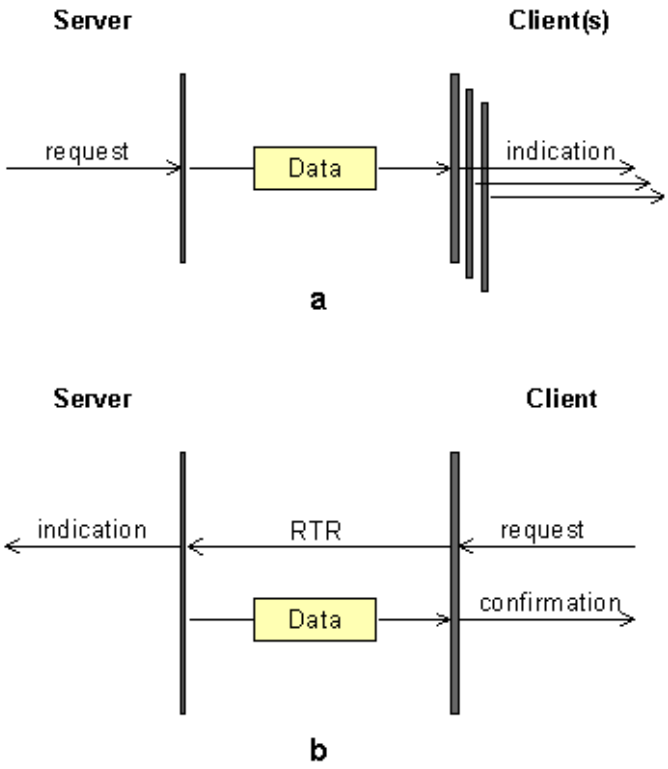


图 3.2-3

3.2 交换过程数据

在一分布式自动化系统的设备之间发送过程数据是 CAN 通讯系统的目的，它应以最高效率的方式完成 。因此，对于应用规定的数据（过程数据、I/O 数据），其传送应该根据产生者━使用者的模型来执行，其意思就是说-----通过相关的信息 ID 判断（推断）所发送的数据。信息的产生者和使用者在这种情况下被假设拥有使用数据的知识或明白发送数据的含义。

以下介绍 CAL、CANopen、DeviceNet 和 SDS 在过程数据交换中不同解决方案的主要特点。

我们想要 CAN 向分布系统的的实现提供标准的、不依赖于应用的通讯设施。CAN 以“变量”、“事件”和“域”的形式提供了通讯对象（CMS 对象）。CMS 对象由一组属性规定，并由符号名字识别。用户程序可以直接访问 CAL 的对象和 服务。变量可以是“基本”或“复合”型。基本变量和事件使得多达 8 字节数据的多播传送得以提供，无需使用任何额外的协议。复合变量包括第一个数据字节里的一复合器，用以区别每一个信息标示符的 128 个“复合型变量”，并允许 7 字节数据的传送。基本变量也有不同的访问类型（只读、只写、读写）。两设备之间的数据应答发送是由读-写变量支持的。客户（端）初始化变量的发送，事件的发送由相应对象的服务器初始化。图 3.2-3a 为“保存和立即通报事件”协议，图 3.2-3b 为“读-事件”协议，通过这些协议客户也可以读以前保存的数据。

基本和复合型的域通过流控制的分段协议支持响应发送大于 8 字节的数据。由于每一个数据段都响应，所以提供了一个由收发器控制的流控制。通过使用 3 个字节的域-多路复合器，就可以识别每个信息的标识符各种不同的域。数据块的传送由一“初始化”-请求／响应程序初始化，接着的数据段通过一个数据段请求／响应程序发送。。图 3.2-1 显示了关于“初始化复合型域”请求和“下载段”请求的数据区的原理结构。控制字

节中显示了不具有数据的信息类型（初始化 / 下载段 / 上载段），发送类型（加速 / 不加速），切换位和字节数。有了加速的（非分段的）协议，就可以发送多达 4 字节的数据。

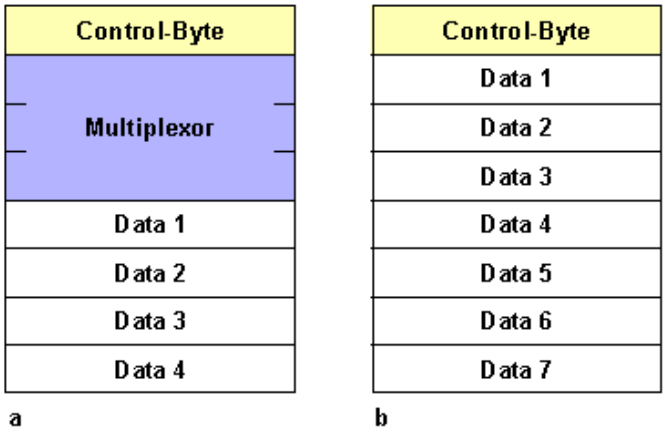


图 3.2-1

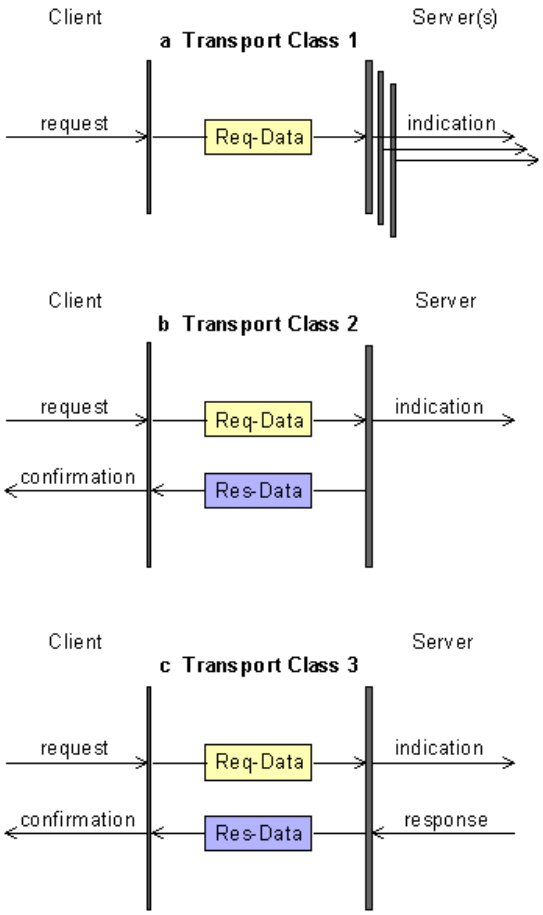


图 3.2-4

CANopen 和 DeviceNet 给人第一印象是它们为发送过程和服务/设置数据所提供的通讯机制十分的相同。对于 CANopen，过程数据通过所谓的“过程数据对象(PDOs)”传送。对于 DeviceNet，过程数据通过所谓的“I/O 信息” 传送。

表 3.2-1 总结了 CANopen，DeviceNet 和 SDS 过程数据交换的主要特点。其中一个主要的不同点是在

DeviceNet 和 SDS 中有一个不响应的分段协议的条款，它使得系统能够发送或处理长度大于 8 字节的数据。系统支持三种有关不响应分段（运输类型）的协议（图 3.2-4）并决定连接端点的“运输类型”。运输类型 2 或 3 可以用作为高效的设备的“查询”，为此，主设备将有关每一查询命令的通讯资源作为客户运输类型 2 或 3 来执行。每一从设备执行服务运输类型 2 或 3 的连接对象，以便接收查询命令以及发送有关的响应数据。

SDS 原本的设计意图是为单个二进制的分布式系统提供一个优化的解决方案。因此使用了短格式的信息，即，采用信息标示符的 3 位部分以便报告或修改单个位设备的状态。

如果要发送更多信息，则要使用长格式信息。读和写服务对一特别设备已定义的对象属性产生作用。行为指的是设备要执行一个特定的功能。事件报警用作为设备里自发事件的通知，或作为设备检测到的事件的通知。每一个这些服务都希望得到远程设备的应用程序的确认 / 响应。

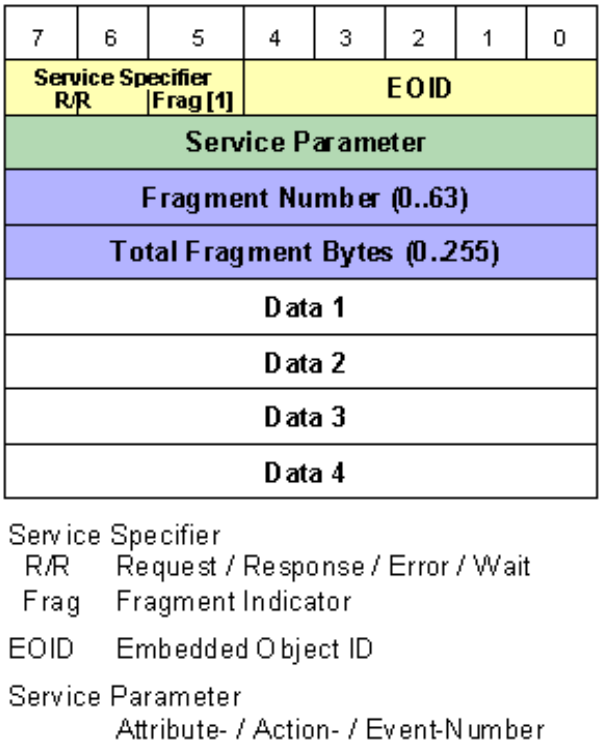


图 3.2-2

图 3.2-2 显示了长格式数据区的结构，它是一个分段的 SDS APDU。在第一个数据字节中，服务区分符用以决定这个信息是一个请求，正确还是错误信息，段指示器位指示信息是否分段，而嵌入对象 ID 区显示了在逻辑设备 32 个嵌入式对象中哪一个可能被访问。属性的数量、编址嵌入对象的行为或事件均规定在下一个数据字节。其余的 4 个字节用以发送数据。在不分段信息中，每个信息可以发送 6 字节数据。

在 SDS 规范 V2.0 中，SDS 通过“信道 APDUs”引入了过程数据的多播。信道 APDUs 总是伴随着预置器的地址（源地址）和服务类型（＝信息标识符中的信道）的发送而发送。信道 APDUs 的发送可以为不分段和分段。不分段的 APDUs 不响应并在第二个数据字节中包含一个信道号码（0～31）和信道代码（多播，点对点）。信道号码通常在配置过程中建立。CAN 数据区中剩下的 6 个字节可以用于发送过程数据。

分段的多播 APDU 的结构和读（Read）、写（Write）、行为（Action）和事件（Event）一样（图 3.2-2）。由于段以巡回（反馈）的方式发送并且只对完整数据发送做出应答，因此不含有流控制的意思。因此，如果允许其他设备访问总线，建议在相继的段之间加有适当的延迟。

信息触发

所有这里涉及到的协议都提供了不同的方式的信息触发，由应用层支持。

DeviceNet 支持下列的触发模式：循环、改变状态和应用对象触发。通过循环触发的模式，当指定信息的**传送触发定时器**终止时，信息的发送就会被启动。通过状态改变的模式，当检测到应用对象状态改变时，信息发送就会开始。 当一特定间隔时间结束而没有发送信息，信息也会被发送。通过应用对象触发模式，应用对象可以决定何时触发信息的发送。当一特定间隔时间结束而没有发送信息时，信息也会被发送。

CANopen 的触发模式划分为事件、应用请求、或预定义同步信息接收后的触发模式。

事件触发可以在子协议（Profile-）或应用指定的事件（“异步 PDO”）中产生。PDO 的发送也可以通过接收一远程请求信息（远程请求）而被触发。“同步 PDOs”是通过接收同步信息的一指定号码而被循环触发的。

同步信息也可以被用作为整个网络中数据获得的同步以及输出数据滤波的同步。

SDS 中，通过配置嵌入对象的相关属性，也提供有不同的触发模式（如循环、状态 / 值的改变）。

映射应用对象

网络设备会产生以及消耗（产生或者消耗）一个以上的应用对象，并且，在一 PDO 分别的 I/O 信息里，一个以上的应用对象的集合会是合适的。

在基于 CAL 的应用中，应用数据的映射是由程序员在定义通讯对象（如 CMS 变量或事件）时完成的。

SDS 通过所谓的“网络数据描述符”将嵌入式对象的 I/O 数据映射到 APDU 的数据区。。这个信息可以由其他设备读出。

CANopen 和 DeviceNet 为灵活地将应用数据映射到通讯对象而提供了非常完善的方法。

CANopen 规定了有关应用对象映射。应用对象通过一个叫“PDO 映射记录”的数据结构映射到 PDO 中。这个结构以对象标识符列表（对象目录索引 / 子索引）和数据长度的形式规定了映射的应用对象数据。由于 PDO 映射可通过 SDOs 访问，所以 PDO 映射可通过配置工具配置。

DeviceNet 中，应用数据的分组是由“集合”对象的实例（instance）规定的。这个“集合”对象的实例定义了发送的应用对象数据格式。一个设备可能包含多于一个 I/O 集合，，而且其相应集合（使用 / 产生_连接_路径）的选择也是一个可配置的设备选项。

表 3.2-1 CANopen、DeviceNet 和 SDS（多播）中的过程数据交换

	CANopen	DeviceNet	SDS（V2.0）
通讯对象的名字	过程数据对象	I/O 信息	多播信道 APDU
每个设备最大的通讯对象数量	512 个发送 PDO 512 个接收 PDO	27 个 I/O 发送信息 1701 个 I/O 接收信息（每设备）	32 个多播信道（每个设备最多 32 个嵌入对象）
数据区最大长度	8 字节	8 字节 分段的： 任意长度	6 字节 分段的： 64*4 字节
协议	不分段： 没有附加字节，通知/读“保存-事件”协议（CAL/CMS） 不响应	不分段： 没有附加字节，支持 3 个“传送类型”： ● 不响应 ● 服务器连接对象响应 ● 应用响应	不分段： 每段前有 2 个字节协议 不响应
		分段： 不响应 分段协议 每帧前面有一个字节协议	分段： 响应 在接收完整个信息块之后分段协议会带有响应 每段前面有 4 个字节协议
信息产生触发模式	● 在本地或远程应用的请求下 ● 循环/非循环同步	● 循环 ● 状态改变 ● 应用指定	由对象模型指定
映射应用对象	可映射的最大应用对象/PDO 数目由对象的数据大小（1 位对象：64 个可映射应用对象）决定	任意数目的应用对象 用分段协议映射	网络数据标识符定义嵌入对象（V1.8）I/O 数据的大小，间隔（granularity）和数据类型
	通过“映射参数记录”（可配置）定义应用对象 支持动态映射	通过集合对象（可能是几个集合对象）定义应用对象 支持动态映射	

3.3 点对点通讯信道

为了通过配置工具配置设备，需要有特殊的设备功能或程序载入的多用途通讯信道。这些非时间极限的通讯信道总是存在于两设备间，如，在一配置工具和要配置的设备之间。数据的发送必须通过一个响应的分段协议来执行。任何支持某种设备配置的不同更高层协议提供这种点对点通讯功能。

为此，从预定义的管理信道到每个设备，CAL 均提供有“配置服务”，作为 CAL 网络管理服务元素的一部分。因此保留了两个标示符，节点 ID 规定了信息第一分段数据区里的编址设备。

CANopen 根据 CAL 复合型域协议提供了“服务信道”。通过“服务信道” ，“服务数据对象”（SDO）可以在任何两台设备之间进行交换。这个协议给任何发送的帧提供响应信息。在初始化域请求数据区的头 3 个字节中，对象目录项的地址由 16 位的索引和 8 位的子索引规定。对象目录项的索引 / 子索引使得计划执行的功能明确地得到了规定。

少于 5 字节的数据可以只使用初始化域请求帧（“加速协议”）来发送。如果要发送多于 4 字节的数据，就必须要使用响应的分段协议，这个协议中每一段有 7 个数据字节。根据预定义的连接组，每一个 CANopen 设备必须向一个默认的服务器 SDO 连接提供两个预定义信息标识符。通过这个默认的服务器 SDO 连接，设备可以被配置工具访问。

由于应用要求 SDO 连接（如在测试工具和设备之间）的动态建立，因此引入了“SDO 管理器”实例（instance）[10]。SDO 管理器拥有预定义 SDO 连接组，因此可以访问网络上的任何设备。询问设备的 SDO 连接首先必须访问 SDO 管理器，及请求建立询问的连接。

DeviceNet 提供了多用途的面向设备的信道和服务。对象属性的读和写、对象的控制、保存 / 类型的保存 / 对象属性，等，均由“显式信息”执行。它们通过“显式信息连接”交换（信息）。“显式信息”的含义/（想使用的“显式信息连接”）陈述于 CAN 数据区。图 3.3-1 显示了分段显式信息的数据区格式。对于服务请求，通常规定有（服务规定的参数（argument））访问对象属性的访问路径（类号码，实例（instance）号码，属性号码）

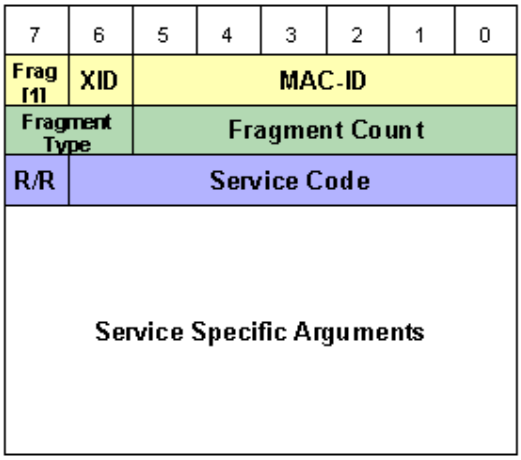


图 3.3-1

显式信息的连接是通过“不连接的信息管理器（Unconnected Messager Manager，UCMM）”建立的。UCMM 提供了两种服务以便打开和关闭显式信息连接。每个支持 UCMM 的设备保留有信息标示符以便发送 UCMM 请求以及响应信息。对于“只有组 2（Group 2 Only）”的设备（不支持 UCMM 口的设备），主机首先要分配设备预定义连接组的显式信息。分配**只有组 2（Group 2 Only）**设备的请求是作为只有组 2（Group 2 Only）的不连接显式请求（Unconnected Explicit Request）方式发送的，它伴随着保留信息标示符的发送。

在不同的逻辑设备上的任意的嵌入式对象之间，SDS 信道服务提供有直接的点对点通讯信道。这样就可以在任意两个点对点连接设备间执行读、写、行为、事件发信。对于每一嵌入对象，可以规定 4 个信道作为通

常的用途和配置。不分段的点对点信道 APDU 可以传送 5 字节的数据，而分段的 APDU 只能发送 3 字节。

在点对点信道开通及成功地用于通讯之前，必须通过连接服务建立与响应设备的连接。由于是在设备之间建立连接，因此网络上需要一个连接管理器。一个初始的设备发送一个连接请求信息以便对连接的建立过程进行初始化。连接管理器服务于所有的连接请求。当需要把一连接请求（信息）发送到相应的设备时，连接管理器作为枢纽中转这个请求。相应的设备用一响应信息应答这个请求。这个应答信息也是由连接管理器服务的。最后，连接管理器继续作为枢纽将连接响应信息中转给初始的设备，这样，相应的设备就完成了连接建立。

在表 3.3-1 中，总结了 CANopen、DeviceNet 和 SDS 的点对点通讯信道主要特征。

表 3.3-1 点对点通讯信道的主要特征

	CANopen	DeviceNet	SDS（V2.0）
名字	服务数据信道	显式信息	点对点信道
最大信道数目	每个设备 128 个客户 SDO， 128 个服务器 SDO	每个设备 27 个显式发送信息 1701 个显式接收信息	每个嵌入对象 4 个信道 每个逻辑设备 32 个嵌入对象
协议	<5 字节： 响应 不分段 >4 字节 分段发送（每段 7 字节） 每帧都响应 任何长度 （CAL 复合域协议）	<7 字节： 响应 不分段 >6 字节： 分段发送（每段 6 字节） 每帧都响应 任何长度	<6 字节： 响应 不分段 >5 字节： 分段发送（每段 3 字节） 接收到整个数据区后响应 最大 255 字节
建立连接	● 由 SDO 动态建立 ● 默认的预定义连接	● 由不连接信息管理器动态建立 ● 只有组 2 设备：由预定义的连接组分配显式信息	● 由连接管理器动态建立 ● 连接组的主 / 从组
连接服务和参数	初始化，退出 上载 / 下载 段 / 域	打开 / 关闭 对象的创建，配置，启动，停止，复位等	打开 / 关闭 读，写，事件，行为
	已编址对象目录项的索引和子索引	对象属性访问路径，服务参数	信道号码，属性（Attribute）/ 行为（Action）/ 事件标识符（Event Identifier）

3.4 建立过程数据信息连接

在 CAN 网络里，信息生产者发送信息以及信息消耗者接收信息，其各自标示符的分配（定位）建立了通讯路径。通过已经分配（定位）的信息标示符的预定义信息可以建立信息的连接，或通过信息的各种标示符分配（定位）来达到建立信息的连接目的。

在一预定义信息的系统中，信息的“功能”和信息的标示符已经被定义。比如，如果一个 SDS 主设备要读从设备中一个嵌入对象的属性值，主设备可以使用一个方向位为 0 的信息标示符，其被定义的信息“功能”为----从设备的地址位于逻辑地址区，读服务代码位于服务类型区。对象和属性 ID 的规定（specification）要在信息的数据区完成。因此，SDS 主-从模式不需要任何的方式来分配标识符。借助于信道服务中扩展的 SDS，每个设备会额外拥有一个高和一个低优先级的多播信息。由于有可能通过连接服务建立任意的多播连接，因此这些信息也被预定义，所不同的是，使用它们更加灵活了。

DeviceNet 和 CANopen 也使用 1:n 系统结构的一预定义连接组方法。比如，根据预定义组，一个已经分配（定位）了从设备预定义查询连接的 DeviceNet 主机，已经“知道” 关于发送查询请求以及期待查询响应信息的信息 ID，因为它们来源于从机的 MAC-ID。相同地，在 CANopen 中，除了其他的预定义信息之外，默认的预定义连接组提供了 2 个预定义的接收和发送 PDO。默认 PDO 的用法和含义是由设备的类型决定。

非预定义标识符分配的主要优点是：建立任何类型的通讯结构的可能性；根据应用需求的最大数量信息标示符的有效性以及面向控制而设计的信息标示符分配有效性。

CAL/CANopen 基于一变量标示符分配方案（此方案基于中央信息标示符库），而 DeviceNet 通过最大为 64 个设备的 DeviceNet 系统发布有效的标示符。

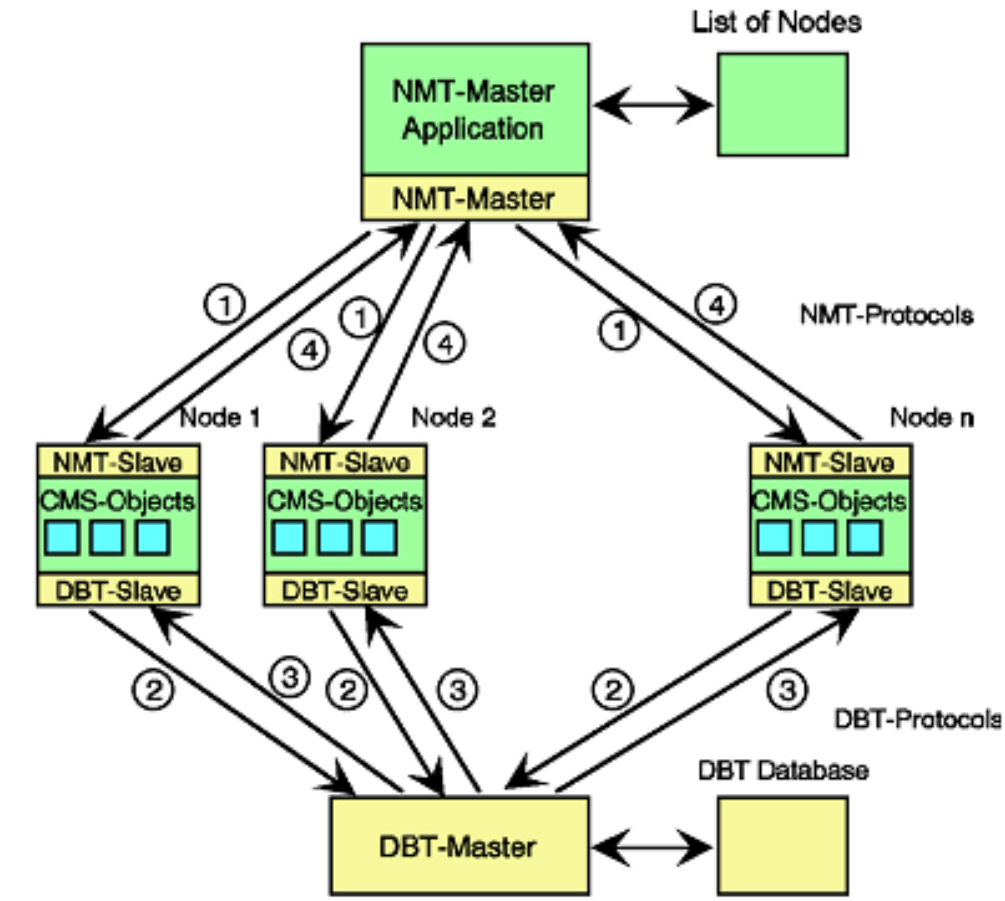
通用标识符库的标识符分配是由一个特殊的网络实例（CAL 中的分配器）或通过配置工具（CANopen）来控制。通过系统管理员，这种配置工具支持信息连接的建立。

CAL 和 CANopen 基于具有通用标示符的一变量标示符分配。通过“分配器”服务元素（图 3.4-1）可以执行 CAL 系统的标识符分配。分配器主机实例（instance）根据设备所有通讯对象的请求（优先权组，名字和类型（发送 / 接收））从信息标识符的中央库（central pool）分配信息 ID。通过根据对象名和类型连接请求，可以连接信息的客户端和服务器。网络主机在设备的网络管理连接中控制分布的过程。

如果不使用 CAL 分配器，在基于 CANopen 的系统中，信息标识符的配置是通过 SDO 信道设置设备对象目录（Object Directory）中相应的 PDO 标识符实现的。

一般的 DeviceNet 标识符分配方法是通过设备拥有信息标示符库这一情况而决定的。因此，I/O 信息的连接首先需要一个标示符的分配，这个标示符出自于信息发送设备的标示符库。

图 3.4-2 说明了两设备之间通过配置工具建立一个 I/O 信息的连接的过程。通过在一已经建立的显式信息连接中访问连接类型，可以建立 I/O 的连接。这包括 I/O 连接对象的创建和在连接的端点配置连接实例。在这个过程中，一个产生模块的信息从它的信息 ID 库中分配一个空闲的信息 ID，并将此空闲的信息 ID 与它的**源 MAC ID**（Source MAC ID）结合在一起，产生一个所谓的“连接 ID”。出自于信息组的一标示符，其“自动”分配可以通过连接 ID 属性的一直接的修改而被取消。



① Calling Node to request identifiers for all of its CMS-Objects

② Node requests an identifier for CMS-Object of name x priority group y

③ DBT-Master allocated identifier to CMS-Object

④ Node confirms end of identifier requesting

② and ③ are repeated for the COB´s of all CMS-Objects of a node

图 3.4-1

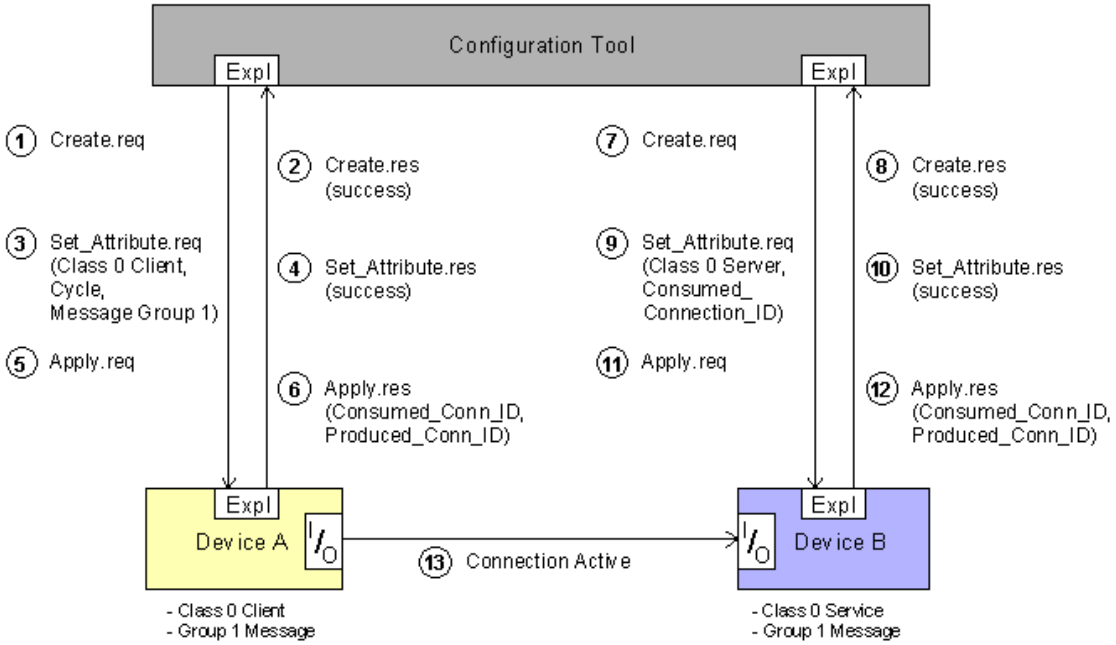


图 3.4-2

3.5 网络管理

由于应用是分布的，所以必须处理某些事件（如，应用部件的故障或节点的故障）。如果同一应用还没有被分配（分类、发布），则这些事件就不会出现。因此，对于一个正确的网络管理，其主要任务是检测和显示网络中的错误，并通过服务以一协调的方式控制分布节点的通讯状态。取决于系统的解决方案，网络的功能性可以通过显式网络管理设施提供，或通过其他方法提供。

CANopen 网络管理是基于 CAL NMT 服务元素，这些元素应用了“节点保护”原则来检测节点故障。为了这个目的，一个 NMT 主机应用通过一个远程请求帧对网络的每个节点（NMT 从机）循环发送一个保护请求。被访问的从机用它的实际通讯状态响应每个请求。如果 NMT 主机检测到节点状态改变或被访问的节点没有响应，就会有一个保护错误告示 NMT 主机应用。当节点连接到网络上后，节点保护就启动。每一个节点也监督到达节点的保护请求信息。节点的“生存时间”过期后如果没有进一步的保护请求信息，则此节点的应用被告之有网络错误。

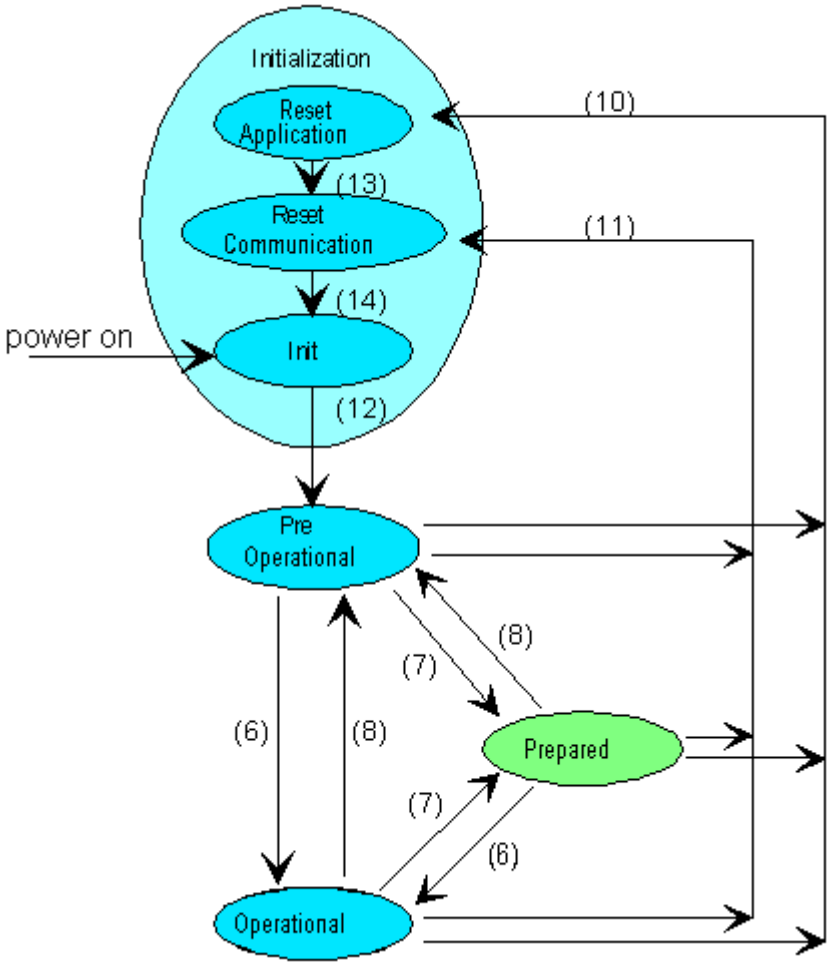


图 3.5-1

节点通讯状态的协调也由 NMT 主机实例来支持。图 3.5-1 显示了 CANopen 节点的节点状态发送方框图。上电之后，节点初始化并转换到“试运行状态”。在这个状态中，SDO 信道上的通讯可以用于节点配置，但不经过 PDO。NMT 信息“启动远程节点”可使被选中的或网络上的其他节点设置到“工作模式”。这种状态中，也可以通过 PDO 交换数据。如果网络所有节点同时工作，就可以确保通讯系统的协调操作。

根据面向连接的设计，DeviceNet 中的每个连接都受到监控。因此，根据配置的“期望信息包速率（expected packet rate）”，每个接收的连接端点都有“静止 / 看门狗定时器（Inactivity/Watchdog-Timer）”，用于监控到达节点的信息。如果定时器溢出，连接就会执行专有的“超时行动”。图 3.5-2 显示了 I/O 连接对象的状态转换方框图。在接收到创建服务（Create Service）（显式信息）后，可以通过使用专有的显式信息服务的顺序来配置连接，并且，要在整个连接完成了配置以后才允许连接。

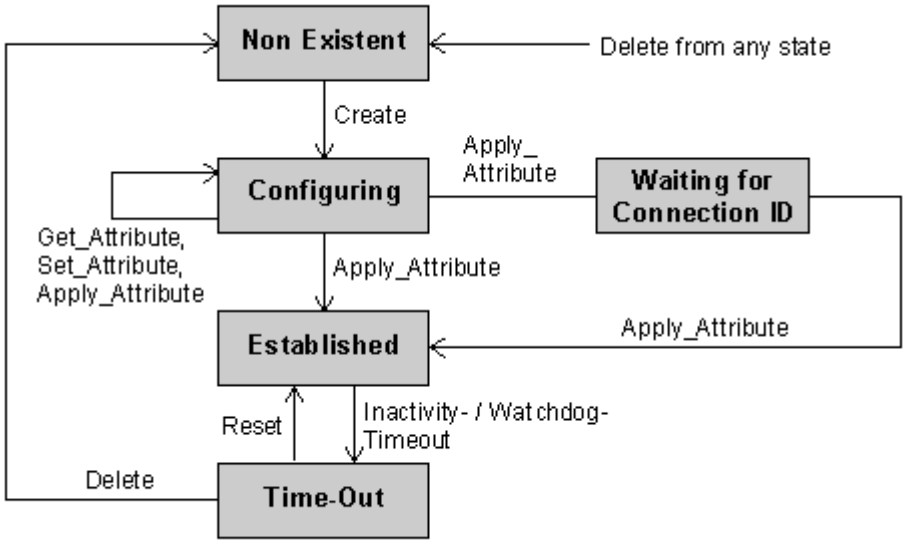


图 3.5-2

在访问每个 DeviceNet 节点之前，每个 DeviceNet 节点必须执行所谓的“重复 MAC ID 检查（Duplicate MAC ID Check）”。使用这个特殊的协议顺序，可以确保设备 MAC ID 的唯一性。所有的 DeviceNet 模块都要求参与这个 MAC ID 的检测算法。

通过“心跳（Heartbeat）-信息”可以提供设备监控的可选方法。“心跳-信息”是借助于非连接响应信息的手段，通过 UCMM 以非连接响应信息或**只有组 2** 的方式，由设备广播的。在这个信息的数据区，设备的状态得以发送。“心跳-信息”通过识别对象（Idendity Object）触发。离线之前，节点可以可选地广播错误信息。

SDS（来源于 V2.2 版）使得每一嵌入对象的每一输出提供有一个连接-看门狗（connection-watchdog），对于非信道的设备，每个逻辑设备有 1 个看门狗。通过所谓的“心跳算法”，主机设备执行了设备的循环请求（以非工作行为方式）。重复设备地址是由“登记-逻辑-设备（Enroll-Logical-Device）”行为支持的。

在任何开放式系统中，系统上电后，通过扫描网络，系统设置的协调在主机实例的支配下得以发挥。

3.6 设备建模和设备子协议（Decive Profiles）

除了标准的通讯外，还要求相似设备在开放式自动化系统中应具有额外的互用性和可交换性。因此，开放式系统的较高层协议如 DeviceNet，SDS 和 CANopen 以“设备模型”的形式描述网络上“所看见的”设备的功能性。为了提高相似设备的可交换性，必须规定工业自动化中主要设备类型的“设备子协议”，以确保不同厂家器件的相同基本（“标准”）行为。

除了设备的功能性描述外，设备模型还必须提供关于设备的身份特点、版本号、状态、诊断信息、通讯设施和配置参数的描述。

图 3.6-1 中，显示了 DeviceNet 的节点模型。它包括了几个对象，其中一些是由 DeviceNet 要求的，其他则是由产品的应用功能要求的。对象提供有抽象的描绘，是关于设备中特殊部件的描绘。对象还描绘了相关的数据（属性），以及数据的过程（服务）。表 3.6-1 简要说明 DvecieNet 例子节点对象的主要功能。

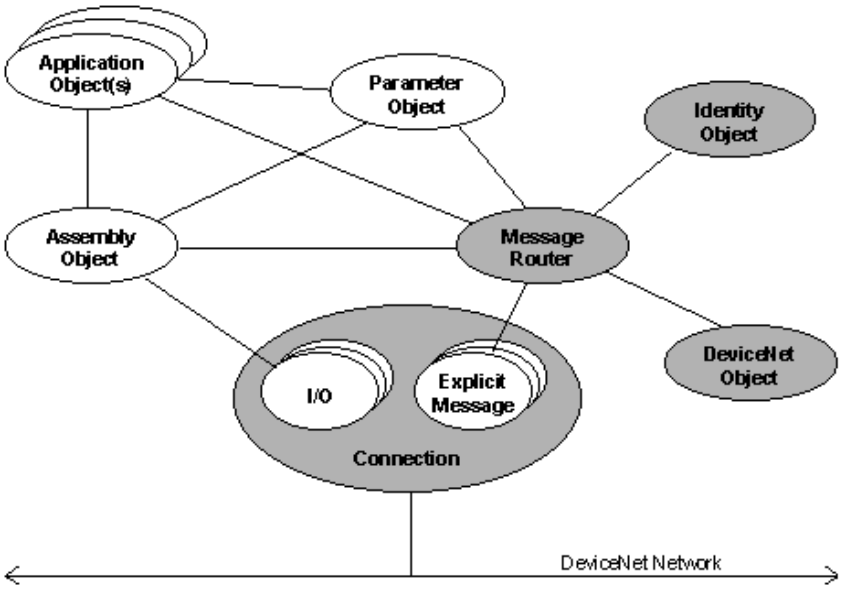


图 3.6-1

DeviceNet 中的对象访问是基于一个具有等级划分的访问方案。对象访问包括 MAC-ID（媒体访问控制标识符），类型标识符（Class ID），实例标示符（Instance ID）以及属性标识符。MAC-ID 用以将一个节点从同一网络中的所有其他节点中区分开来；类型标识符（Class ID）用以识别对象类型；实例标示符用以在相同类型的所有实例中识别一个实例；属性标示符用以识别一个类型或实例里的属性。

DeviceNet 设备子协议（Device Profile）必须包含下面的信息：

- 设备类型的对象模型
- 设备类型的 I/O 数据格式
- 配置数据和数据的公共接口

表 3.6-1 DeviceNet 节点的对象

对象	功能
连接	示例（Instantiates）连接（I/O 或显式信息）
DeviceNet	维持 DeviceNet 物理连接的配置和状态
信息路由器	将接收到的显式信息路由到正确的目标对象
集合	将多个对象的属性集合到一个单个数据块中，这个数据块可以通过单个连接发送和接收
参数	给设备配置和属性访问提供标准的方法
特性	提供设备特性的总体信息
应用	提供应用指定的行为和数据

在 SDS 中，一个物理元件可以包含高达 126 个逻辑设备，而一个逻辑设备可以包含高达 32 个嵌入式对象。SDS 以带有等级的、组织过的部件模型形式描述设备的功能性（图 3.6-2）。SDS 部件模型表示了部件的可视结构和行为。在 SDS 模型中，层次的最高等级定义了物理部件和逻辑设备的结构和行为。每个 SDS 产品必须包括这个最较高等级。功能性的不同等级由 SDS 层次分门别类记录在案，并被举例于对象类型属性里。这个层次在层次的最高层有一定义的一组对象，包括 I/O 设备，SDS 接口，IEC 1131-1 定义的功能和功能块。

SDS 层次对象的基本工作原则是继承。所有模型继承了属性、行为、和模型的事件。由 SDS 模型继承的通用结构和行为等级定义了最小的一组属性、行为，并且 SDS 部件必须具有这些属性和行为。

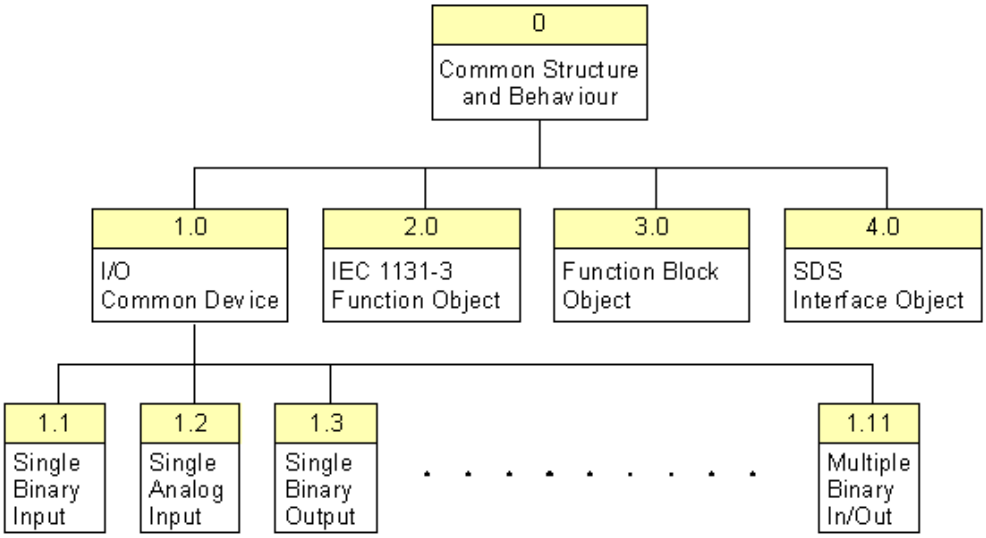


图 3.6-2

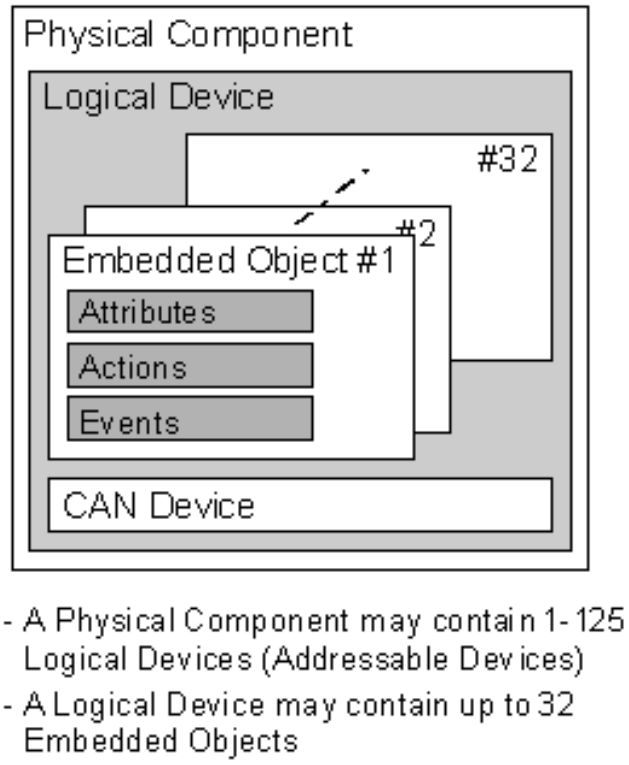


图 3.6-3

一个设备可能包含高达 32 个嵌入对象，而且这些对象都是总线可寻址的实体（图 3.6-3）。根据嵌入对象的级别，可以确定属性（Attributes，如网络数据描述符、波特率，对象类型，…，软件版本或序列号，输入数据，输出数据），行为（Action，如改变地址，自我测试，强制状态或清除错误）和事件（Event，如诊断错误，定时器结束，值的改变）。

通过“对象目录”，CANopen 基于设备功能性的描述。对象目录项是由一个 16 位索引和一个 8 位子索引号码识别，表项目（数据、参数、或功能）的功能被规定其中。除了用于定义数据类型的区段外，还有 3 个主要的区段（图 3.6-4），分别是：通讯子协议区段（Communication Profile Section），标准化的设备子协议区段（Standardised Device Profile Section）和厂商细节区段（Manufacturer Specific Section）。

Index (hex)	Object
0000	Not used
0001-009F	Definition of static, complex, manufacturer-specific and device profile-specific data types
00A0-0FFF	Reserved
1000-1FFF	Communication Profile Area
2000-5FFF	Manufacturer Specific Profile Area
6000-9FFF	Standardised Device Profile Area
A000-FFFF	Reserved

图 3.6-4

通讯子协议区段（Communication Profile Section）信息对于任何 CANopen 设备类型都是一致的，并包含与信息、参数和功能相关的设备。这些信息、参数和功能又与设备的鉴别、错误管理、包括将应用对象映射到过程数据对象的设备通讯信道的定义（图 3.6-5）有关。与 DeviceNet 有关的 CANopen 通讯子协议区段会与 DeviceNet，身份特点，连接和集合对象的功能性进行比较

CANopen 设备子协议区段（Device Profile Section）为特殊类别的基本（“标准”）设备的功能性提供了一个接口。其中有些条目是强制的，有些是可选的。强制的、共有的条目要确保设备在一个定义的基本方式中运转。主要工业设备（如 I/O 模块[11]或驱动器[12]）不同的设备子协议用以提高设备的可交换性。

厂商特殊的或不标准的设备的功能性可以通过厂商特殊子协议区段（Manufacturer Specific Profile Section）来提供。

4．结论

由于在汽车和其他任何类型的工业应用中广泛地使用了 CAN 协议，从而发展出几个类别的较高层协议。在这份论文里，我们只是讨论可以被认为是工业标准的“开放式”解决方案。论文着重讨论了工业自动化中的开放式系统解决方案。它的目的不是作为一个指南供读者选者最合适的协议，而主要是便于读者更好地理解较

高层协议。

在重型汽车、卡车及相关应用中，美国广泛使用了 SAE J1939 标准，因为这个标准延续了旧标准 J1708、J1587 和 J1922。

OSEK/VDX 规范可以被认为是汽车分布式控制中最有前途的通用软件平台。

Index	Object Class	Object
1000h	VAR	Device Type
1001h	VAR	Error Register Device Specification Data Device Global Communication Parameters Number of supported PDOs and SDOs
1200h	Record	1st Server SDO Parameter ↓ 128th Server SDO Parameter
1280h	Record	1st Client SDO Parameter ↓ 128th Client SDO Parameter
1400h	Record	1st Receive PDO Parameter ↓ 512th Receive PDO Parameter
1600h	Record	1st Receive PDO Mapping ↓ 512th Receive PDO Mapping
1800h	Record	1st Transmit PDO Parameter ↓ 512th Transmit PDO Parameter
1A00h	Record	1st Transmit PDO Mapping ↓ 512th Transmit PDO Mapping

图 3.6-5

由于 CAN 也在非汽车自动化中被广泛使用，几乎同时又产生了几个较高层协议和开放式系统方法。

由于有了 CAL，应用层协议标准在任何的应用中被被广泛接受和认可。任何的应用必须满足固定配置的环境中的特殊要求。由于 CAL 与 CANopen 兼容，所以 CANopen 模块可以在 CAL 系统中使用。

今天， DeviceNet，CANopen 和 SDS，这三种基于 CAN 的开放式系统已经成熟，其标准已经形成。这三个标准给人留下一个印象，如果把 SDS V2.2 考虑进去的话，那就是它们具有相同的功能性，虽然这三个标准采用的方法不尽相同。最明显的不同是信息标示符的使用。信息标示符的使用在 SDS 里是基于完整的预定义方式，而在 CANopen 里，信息标示符的使用是向系统设计者或综合者保持开放。DeviceNet 的信息标示符的使用也自由，但相对有些限制。DeviceNet 和 SDS 基于面向连接的观点，CANopen 基于面向信息的观点。每一个系统都随着 DeviceNet 提供大部分的协议而采用不同的数据传输协议。由 DeviceNet 使用、一致的面向对象的模型，其设备建模的解决方案也十分不同。

但是，除了系统解决方案的功能特征外，还有更进一步、非常重要的许多方面决定市场对这个解决方案的接受。这些方面有这么几项：提供支持（文档，培训），产品（控制器、设备、接口、软件、工具）已经到位，谁最后（不是至少）是系统解决方案的提供者/发源者。最后一项说明了为什么 DeviceNet 和 SDS 的应用在美国这么普遍，而 CANopen 却盛行于欧洲。这就意味着，对于一个欧洲的设备制造商来说，如果它想打开市场，就必须在它的产品上提供一个或两个美国标准的接口解决方案。幸运的是，所有的这些解决方案都基于相同的物理层和数据链路层。这就是说，当应用专有的总线接口时，只需要在软件方面下工夫。

5．参考书目

- ISO-IS 11898, Road vehicles - Interchange of digital information - Controller Area Network (CAN) for high speed communication, 1993
- Bosch, CAN specification, Version 2.0, 1991, Robert Bosch GmbH
- CAN Application Layer for Industrial Applications, CiA DS 201-207, Version 1.1, 1996
- OSEK/VDX Operating System, Version 2.0, 1997
- Etschberger, Modelling Distributed Application Processes with CAL, Proceedings of 1. ICC, 1994
- OSEK Communication Specification 1.2
- OSEK Network Management, Version 2.0, Draft 1.1, 1997
- CANopen, Communication Profile for Industrial Systems based on CAL
- CiA Draft Standard 301, Version 3.0, 96
- DeviceNet Specifications, Release 2.0 1997, Vol. I: Communication Model and Protocol, Vol. II: Device Profiles and Object Library
- Micro Switch Specification: Application Layer Protocol Specification Version 2.0, 1996, SDS Component Modelling Specification, 1995
- SAE: Recommended Practice for a Serial Control and Communication Vehicle Network, J1939 Committee Draft, 1996
- CiA Draft Standard Proposal DSP 302, Framework for Programmable Devices, 1997
- CiA Draft Standard Proposal DSP 401, Version 1.4, Device Profile for I/O Modules, 1996
- CiA Draft Standard Proposal DSP 402, Version 1.0, Device Profiles Drives and Motion Control, 1997

——译自 IXXAT/stzp 的 CAN-based Higher Layer Protocols and Prifiles 网页