iCC 1995

2nd international CAN Conference

in London (United Kingdom)

Sponsored by

Motorola Semiconductor
National Semiconductor
Philips Semiconductors

Organized by

CAN in Automation (CiA)
international users and manufacturers group
Am Weichselgarten 26
D-91058 Erlangen
Phone +49-9131-69086-0
Fax +49-9131-69086-79
Email:headquarters@can-cia.de
URL: http://www.can-cia.de

Steinbeis-Transferzentrum Prozeßautomatisierung, STZP
Weingarten, Germany

Dr.-Ing. K. Etschberger

Open Communication and Interoperability of Devices within CAL-based Networks

Abstract

Based on CAL, the Standard CAN Application Layer of the CiA, two general approaches for implementation of open networks which provide interoperability and interchangeability of devices are presented. In the first approach the application processes have direct access to the services and protocols of CAL. Application depended specifications are provided and additional standardized or nonstandardized "Application Profiles". With the second approach ("CANopen") a standard application is specified which provides the environment for the usage of standard devices in different applications.

1 Introduction

The CAN-layer-2-protocol, as specified in ISO 11898 [1] is the common basis of a wide range of network solutions for which CAN can be applied. Besides the originally intended usage within cars, vans and mobile systems, CAN has found wide acceptance as a versatile communication system within all kind of distributed automation systems. Since the layer two only provides services for transmitting and requesting of short data packages, every practical distributed application needs additional functionality and rules to be fully functionally. A special aspect which in every CAN-application has to be decided is the method of allocation of communication object identifiers. The CAN-protocol leaves this open to the layer-2-user. Soon after the availability of CAN chips many application specific CAN-based network solutions were developing and it was recognized that there was an urgent need for further standardization to make data communication open, devices interoperable and interchangeable. This was the starting point of the CAN-in-Automation (CiA) association to specify a commonly accepted layer-7-protocol for CAN-based networks. The CAL (CAN Application Layer [2]) Standard has been available since 1993 and has become in the mean time widely accepted in a variety of different applications.

2 Models of Implementation of CAN Systems

In figure 2-1 three different models of implementation with respect to the interface between application and communication system are shown.

In model 1, the application (user) directly interacts with the Data-Link-Layer which is frequently extended by individual protocols such as for network management and fragmented data transfer. This approach was common until a higher protocol for CAN-based systems was available. The main disadvantages of this model is the closed system, non-standardized approach which provide a means for interoperabiltiy with devices of different manufactures and with the requirement to develop application specific communication functions.

With the model 2 approach the application uses the services of a standard application layer. One of the main advantages of this approach is that all communication specific aspects are transparent to the application and a user has access to the full functionality of the application layer, which offers him extended communication functionality in form of standarized services and protocols. Since an application layer shall be independent of any application, it does not cover applications specific items. This means that the kind of usage of application layer services and the interpretation of the transmitted data has to be specified in form of additional application dependent specifications ("Application Profiles"). This is a common approach with CAL-based CAN-networks.

In the model 3 approach the usage of application layer services and device functionality is standardized in the form of a "standard application". As figure 2-1 shows, in this approach the application has no direct access to the services of the underlying communication layer. Instead, the standard application provides a predefined model of communication (e.g. a master-slave relationship) and completely hides the functionality of the underlaying communication system. To support interchangeability of devices, the functionality of a device may be described in a standardized form ("device profile") and standard devices may be defined. This type of approach is taken by the CiA-SIGs[1)] "Motion Control", and "Distributed I/O" when specifying CANopen, and the "CAL-based Profile for Industrial Automation" [3]. DeviceNet (Allen-Bradley) and SDS (Honeywell) also follow this kind of approach.

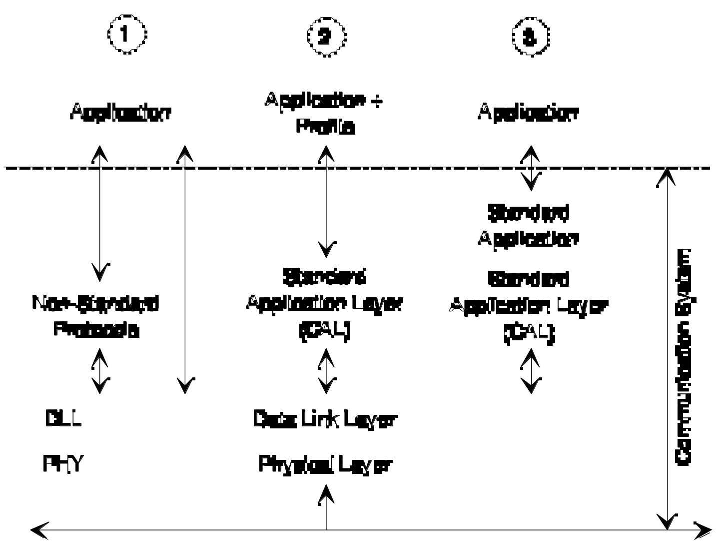

Figure 2-1: Models of Implementation of CAN Systems

In chapter 4 and 5 the two principal approaches of CAL-based network solutions according to model 2 and 3 are presented.

3 CAL, the Platform for Open Communication

In Figure. 3-1, the four service elements of CAL are shown:

- CMS, CAN Message Specification
- NMT, Network Management
- DBT, Identifier Distributor
- LMT, Layer Management

CMS offers an open, object oriented environment for the description of distributed applications and provides the following three types of objects with related services and protocols:

- CMS-Variables allow the reading and writing of data via the network

- CMS-Events allow the signalling of remote node events

- CMS-Domains allow the down- and/or uploading of data blocks of more than 8 byte

Each CMS-object is described by a set of object-specific attributes which completely specify the communicational items. Thus, CMS may be considered as a standard language for the specification of the communication interface to CAN-devices. CAL does not introduce any protocol overhead with the basic services and protocols with respect to a direct access to layer 2.

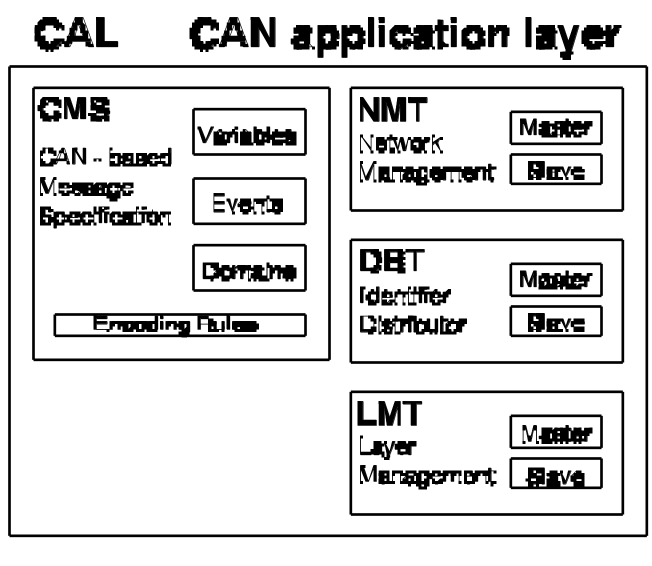

Figure 3-1: Service Elements of CAL

Besides of the communication functionality, a distributed application also needs further facilities (e.g. to control the initialization of a network) and for monitoring its correct functioning during operation. For this purpose, CAL provides a standardized **Network Management (NMT)** functionality. To reduce the required resources to a minimum, CAL-NMT performs its functions via a logical master-slave model. One module in the network acts as the master, all other modules are slaves with respect to network management. The NMT **Module Control Services** offer the user a set of services to perform initialization, starting and stopping, or identification of NMT-Slave nodes. **Error Control Services** support the detection of failures in the network or at a particular NMT Slave. **Configuration Services** support the up- and downloading of configuration data. To perform the NMT services some communication identifiers are reserved by CAL.

Of particular interest is the dynamic identifier distribution capability which CAL provides by means of its **Distributor (DBT**) facility. With this functionality it is possible to make the allocation of identifiers to CAN messages transparent to the application process and open to the system integrator. This allows open system configurations, using modules of different suppliers.

CAL distinguishes 8 object priority groups. Identifiers are distributed by the DBT-Master on request of the NMT-Slave modules. The identifier distribution process is controlled by the NMT and DBT services and protocols during the system initialization process. In figure 3-2, the principle of the CAL identifier distribution process is shown. The DBT concept of CAL also allows the coexistence of nodes which do not support the DBT protocol. This is done by reservation of identifiers in the DBT data base before dynamic distribution is started.

As a further functionality, CAL offers **Layer Management** (LMT) services which allow to set the module identification number and baud rate, via the CAN network, to be set. The LMT protocols require that only one LMT Slave is selected (e.g. by a unique serial number) in a network.

A general introduction into the concepts of CAL is given in [4] and [5].

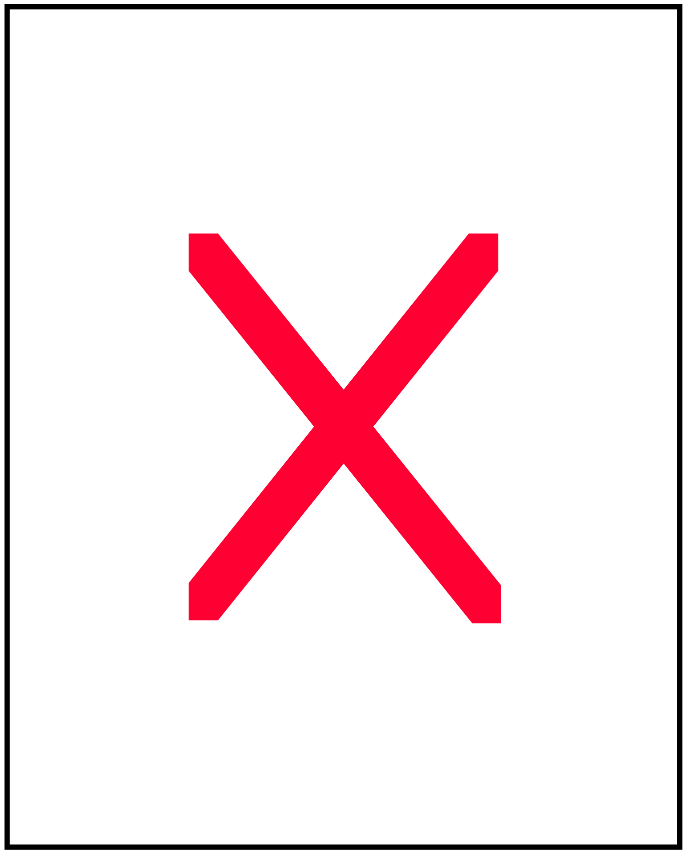

Figure 3-2: Principle of the CAL Identifier Distribution Process

4 CAL-based Network Solutions with Standard Application Layer Interface and Application Specific Profiles

With network solutions of this type the application processes have direct access to the services of CAL according to model 2 (Figure 2-1). The usage of CAL objects and services, the interpretation of transmitted data and the network initialization process takes place according to additional,. application-dependent specifications. Many applications will do this in an individual, application-oriented manner (**Non-Standard Profiles**). The usage of standardized profiles (**Standard Profiles**) is also possible to simplify the interoperability and interchangeability of devices. The two approaches are discussed in the following paragraphs.

Non-Standard Application Profiles

In figure 4-1, the general form of the module data sheet for a non-standard-profile CAL-based application is shown [2)]. Typical examples of this kind of applications are dedicated solutions within medical systems, traffic control systems, production control or machine tools machines.

The data sheet of a non-standard-profile CAN module has to include all the information which is necessary to access the modules functionality via the network. The provided information may be grouped as follows:

- **Module Identification**

- **NMT-, DBT- and LMT-Functionality**
  This is expressed in terms of the NMT/DBT/LMT capability classes according to the CAL specification.

- **Supported CMS Objects**
  By means of the CMS object attributes (e.g. object name, data type, priority group, user type) the necessary information is given to access the CMS object via the network. Beside the true functional CMS objects, module configuration and module control by CMS objects may also be provided.

- **Device Functionality**
  This is provided by the data structure of the CMS objects, extended by a description how the data has to be interpreted. It is proposed, to provide this information by an extension of the CMS object attributes in form of "application specific attributes".

1.	Module Identification	
1.1	LMT Identification − Manufacturer Name − Product Name − Serial Number	according to LMT naming conventions
1.2	NMT Identification − Module-Name − Module-ID	according to NMT naming convention
1.3	Module Function	Verbal description of Module function
2.	General Interface Parameters	
	Timing Parameters	List of supported baud rates (timing parameters)
3.	NMT-, DBT- and LMT Capabilities	
3.1	NMT node class	0 - 4, according to NMT service specification
3.2	DBT slave class	0 - 3, according to DBT service specification
3.3	LMT slave class	0 - 2, according to LMT service specification
4.	Specification of the supported CMS Objects	
4.1	Function Group	Group Name
4.2	Function	general description of function
4.3	CMS-Attributes	- Name - User Type - Priority Group - Inhibit Time - Class - Data Type - Object specific attributes according to CMS standard
4.4	Application specific-Attributes	- Usage - Meaning - Default Value

Figure 4-1: Data Sheet of a CAL-Module with Non-Standard Profile

An examples of this kind of solution is given in [6].

Standard Application Profiles

With respect to device interoperability it is advantageous to make further agreements concerning the required module capabilities and system environment. These have to be formulated according to the particularity of a distinct application area.

This approach was taken by the CiA-SIG "Mobile Applications" when defining a profile for applying CAN in mobile applications. Typical examples of this type of application are data communication structures within tanker-, silo-, communal-, sweeping- or rolling tipper vehicles, construction machines or lifting and conveyor vehicles. These types of applications are characterized by high EMC requirements, special environmental and maintenance requirements. The actual status of the **Mobile Application Profile** is characterized by the following specifications [7]:

- Specification of the Physical Layer parameters, transceiver and busline
- Availability of all CMS services
- Definition of four "**Module Groups**" according to the supported NMT-and DBT slave capability
- Definition of **Standard Parameters** according to the SAE J1939 standard. The purpose of this is to provide rules for the interpretation of transmitted data and to simplify the exchange of data between CAL based and SAE J1939 based networks[3)].
- The definition of a **standard module data sheet** comprises the specification of the general module attributes (module name, module group) as well as providing functionality in terms of CMS objects (CMS attributes, assigned standard parameters)
- Definition of possible **module setup procedures** for the adjustment of baudrate, module address, module configuration and the type of identifier distribution
- Definition of a **system startup procedure**

5 Standard Application CAL-based Network Solutions (CANopen)

A further degree of standardization is possible, if a standard application according to model 3 of figure 2-1 is provided. The main advantage of this approach is that the interfacing between communication system and user application is further simplified and the implementing of user processes reduces to a parameterization of the provided standard application. A disadvantage of this model is that the functionality and performance is determined by the provided features of the underlying standard application process.

Index	Objectrange
0000	reserved
0001-005F	Definition of static, complex, and manufacturer-specific datatypes
0060-0FFF	reserved
1000-1FFF	Communication-profile
2000-5FFF	manufacturer-specific profiles
6000-9FFF	Standard-device-profiles
A000-FFFF	reserved

Figure 5-1: General Layout of the CANopen Object Directory

This approach is taken by the CiA SIGs "Motion Control" and "Distributed I/O" when specifying CANopen, and is a standard application for distributed I/O and motion control systems ("CAL based Profile for Industrial Systems"). This specification is divided in the parts "Communication Profile" [3] and "Device Profiles".

In CANopen, the functionality of a device respresented by its objects (parameters, data, functions) is specified in form of entries to a Device Object Directory. In figure 5-1 the general layout of the CANopen Object Directory is shown. Access to an object description entry is given by an 16-Bit index and by the 8-bit subindices of the object element, if the entry is composed of more than one element. (fig. 5-2). Every object is specified by its function and some global attributes like name, data structure and number of components. For each object element the following attributes are specified: Value or function, data type, range of values, permitted range and supported features.

A certain degree of interchangeability of devices is provided since standard devices with Common basic functionality may be specified in index range 6000 to 9FFF of the CANopen object directory. Standard device profiles are already available for simple standard I/O devices [8].

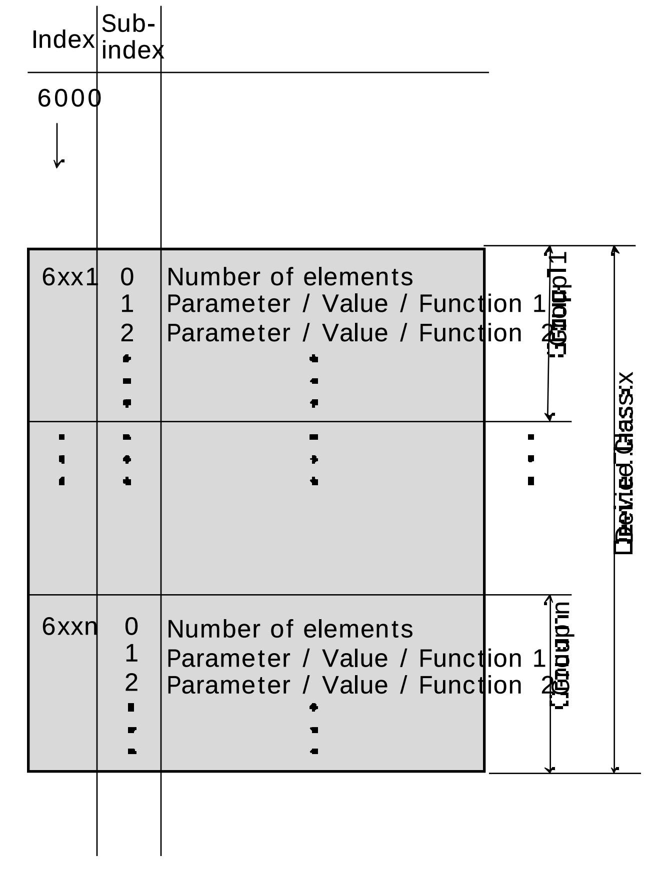

Figure 5-2: CANopen Device Description Font

CANopen supports at event-driven communication model, extended by a synchronization mechanism to support the synchronous operation of devices. The communication model is characterized by the following types of communication objects:

- Objects which provide access to all entries of a Device Object Directory. These objects are called "**Service Data Objects (SDOs)**". The main purpose of these objects is the configuration of devices during system initialization. By means of a SDO a peer-to-peer communication channel may be established between two devices. SDOs also support the transfer of fragmented data by using the CMS-Multiplexed-Domain protocols. A device may establish more than one SDOs. The transfer of SDOs normally takes place with low priority.

- Objects by which process data ("**Process Data Objects (PDOs)**" are broadcasted without any protocol overhead. The data field of a PDO is limited to a length of 8 bytes. PDOs are modelled by CMS objects of type "Stored-Event" and "Basic-Variable" with the related protocols. Objects of the "Stored-Event" CMS type support the event-controlled data generation, and also the reading of data via the network, if necessary. Objects of the "Basic Variable" CMS type model the transmission of commands.

- **Management Objects** for execution of LMT-, NMT- and DBT-services. These objects use reserved identifiers according to the CAL standard.

- **System Control Objects** for synchronization, provision of a system wide time reference or predefined emergency messages

Within the range of the Communication Profile (indices 1000 to 1FFF) all communication specific parameters are described. These are:

- Device identification
- Predefined error codes
- Number of supported SDOs and PDOs
- Parameters of the synchronization mechanism
- Description of the supported PDOs

  - PDO-status (enabled/disabled)
  - PDO-specification (length, type)
  - Default COB-ID
  - Allocated COB-ID
  - Type of transmission (synchron/asynchron, cyclic/acyclic, cycle rate)
  - List of mapped application objects
  - Default PDO-mapping

The CANopen standard application optionally assumes the availability of a central synchronization instance, which periodically transmits a high priority synchronization message. By specifying a PDO of type "synchronous", a synchronized actualization and transmitting, or strobing of received PDO data is performed. In figure 5-3 the principle of synchronization is shown. The synchronous operation of PDOs is of primary interest in connection with motion control systems which have to perform the synchronous motion of several axes.

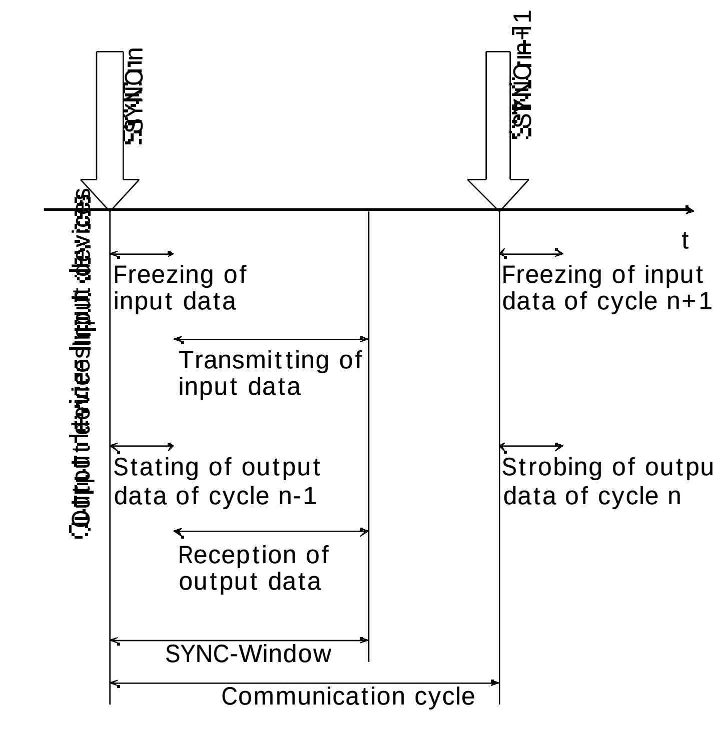

Figure 5-3: Principe of Synchronization

PDOs, which are not connected to the SYNC-message are transmitted asynchronously.

A further remarkable feature of CANopen is the possibility to configure the usage of the PDO data field during system initialization. This feature is called "PDO-mapping" and describes how the application objects of a device (specified by the index/subindex of an object directory entry) are mapped into the maximum 64 bits of a PDO. The desired mapping structure may be downloaded to a device via SDO services during system configuration.

The initialization and monitoring of a CANopen network is controlled by NMT services according to the CAL specification. To allow the configuration of PDOs via SDO an additional NMT node state is introduced ("preoperational"). During this state only SDOs are available for communication. To allow very simple devices to be included into a CANopen network, mimimum device capabilities with respect to the support of NMT services and a predefined identifier set are specified.

6 Conclusion

The presented approaches allow the construction of optimized, application-dependent solutions of open networks based on the same platform (CAL).
Whereas the direct access of CAL services, combined with additional application-specific profiles provides a high degree of optimization and performance, the approach of a standard CAL application provides a higher degree of standardization with a completely specified platform for the interoperabiltiy and interchangeability of standard devices.

Literature

[1] ISO-IS 11898, Road vehicles - Interchange of digital information - Controller Area Network (CAN) for high speed communication

[2] CiA-DS 201-207, CAN application layer for industrial applications, May 1994

[3] CiA-DSP 301, CAL-based communication profile for industrial systems, Aug 1995

[4] Etschberger, K., Suters, T.: Offene Kommunikation auf CAN-Netzwerken - Schicht-7-Protokoll entkoppelt Applikation und Kommunikation, Elektronik 1993, Heft 7/9

[5] Etschberger, K. (Hrsg.): CAN, Controller-Area-Network, Grundlagen, Protokolle, Bausteine, Anwendungen. Hanser-Verlag München Wien. ISBN 3-446-17592-2, 1994

[6] Etschberger, K., Schlegel, C.: Der Schicht-7-Standard für CAN-Netzwerke, Beschreibung von Geräten durch CAL-basierende Profile; Elektronik 1995, Heft 7, Seite 78-86

[7] CiA-SIG-MA 110-0, Profiles for Mobile Applications, Nov 1994

[8] CiA-DSP 401, Device Profile for I/O Devices