# Gateway profiles connecting CANopen and Ethernet

Holger Zeltwanger, CAN in Automation e. V.

**In the past, CANopen networks were used mainly as stand-alone networks embedded in machines and deeply embedded in sub-systems. If required, non-standardized gateway devices realized the connection to higher-level Ethernet-based networks. In future applications, Ethernet-based networks also may be connected to higher-level CANopen integration networks. Besides hierarchical network architectures, there may be required non-hierarchical network architectures, too. The paper presents the current status of standardized CANopen gateways to Ethernet-based networks, and discusses the requirements on further standardization activities.**

## Introduction

There is an increasing demand to standardize the communication between CANopen and Ethernet-based networks. Several manufacturer-specific gateway solutions have been developed. In most of the applications the Ethernet gateway will be regarded as the higher hierarchical network, meaning that an Ethernet-connected controller accesses the CANopen devices via the gateway. This approach has been standardized by the CiA DSP 309 interface profiles for Modbus/TCP and generic Ethernet solutions (ASCII-based commands). In addition, there may be applications, in which the Ethernet-based network is the sub-network to CANopen. In those applications, a CANopen device accesses the Ethernet devices. Implementing both interface profiles will allow a non-hierarchical communication between CANopen and Ethernet networks.

## Homogenous and heterogeneous networks

Communication in homogenous networks using the same application, transport, and data link layer protocols is for several network approaches standardized; e.g. for TCP/IP and Ethernet and for LON.

Heterogeneous networks integrating different network technologies are not that well standardized. Gateway devices connecting TCP/IP-based networks to CANopen provide proprietary interfaces. These interfaces implement proprietary commands and replies. In particular, the CANopen side does not implement any standardized routing functionality.

## Hierarchical and non-hierarchical networks

There are different network architectures possible. A hierarchical network architecture approach provides a determined access from the higher hierarchical to the lower hierarchical network and vice versa. There are pure master/slave networks, in which only the master has the permission to request communication services. In other hierarchical networks peer-to-peer communication is supported by means of client/server communication. However, there is still a hierarchy meaning that there ma y be a bottleneck on one of the higher-hierarchical networks. There is no by-pass possible.

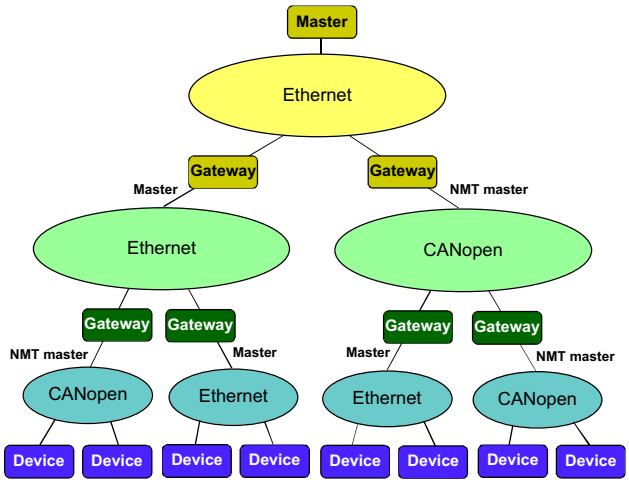


*Figure 1: Hierarchical and heterogeneous network for machine control systems*

In non-hierarchical network architectures there is no master network. Any network has the capability to request services from any other network. This allows using a by-pass.

### Transparent gateways

In homogenous CANopen network architecture, the gateway devices may be transparent to the application. The gateway device may represent up to eight generic logical devices. In the CANopen dictionary the first logical device may use the objects in the range of $6000_h$ to $67FF_h$. The following seven logical devices use the same entries with an offset of $800_h$ and so on.

In CANopen application profiles there may be defined virtual devices allowing a higher granularity. In the CANopen application profile for lift control systems, there are defined panel and display virtual devices for up to 127 floors plus several other virtual devices. The profile also supports logical devices. Each logical device represents a single lift application. In total, one object dictionary may implement all virtual devices of eight lift control systems. This means, a gateway device may provide half of the virtual device on one of the interface and the other half on the other interface. Or it may represent all virtual devices minus one on one interface and only one virtual device on the other.

### Cascaded network architecture

CANopen control system may be implemented on cascaded networks. This is very easy to implement in gateway devices, if standardized CANopen application profiles are used. For example, a lift control systems based on the above mentioned CANopen application profile may be distributed to several physical networks and connected by means of transparent gateway devices.

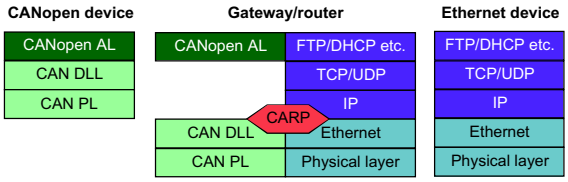


*Figure 2: Gateway protocol structure*

If using different network technologies, the gateway devices of cascaded networks require a routing function. This routing function may tunnel CANopen communication services to another CANopen network via Ethernet. In addition, the router may send or request data from any TCP/IP network connected to the network system. This requires a logical addressing scheme mapped in each network to physical addressing scheme.

In Ethernet networks, this mapping function is implemented by the (R)ARP (reverse address resolution protocol). This protocol is defined in the RFC (request for comments) 826. This mapping function may be adopted for CANopen. One approach is to use the IP function to address logically the network and than to map the IP address to the CANopen node-ID. The (R)ARP message comprises the requested/confirmed CANopen service. In case of an SDO, the message contains the index and sub-index to be accessed. In case of a PDO, the message provides the number of PDO to be received or to be transmitted. In case of a NMT command, the command specifier is transmitted within the (R)ARP message.

### Requesting and confirming CANopen services

Recently, CiA members have standardized simple gateways with limited router functionality (CiA DSP 309 – Interfacing CANopen with TCP/IP). This framework defines generic communication services on the Ethernet interface.

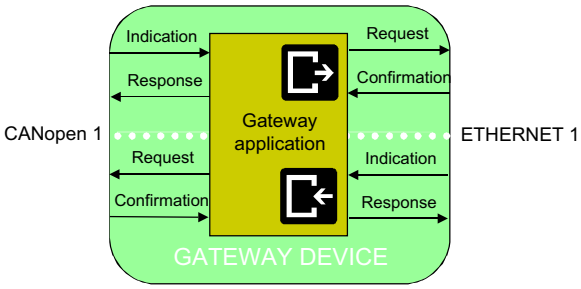


*Figure 3: Service primitives*

The generic communication services are based on the following service primitives:

*Request*

This instance requires a communication service.

*Indication*

This instance is informed on event occurred on the Ethernet network.

*Response*

This instance answers the event.

*Confirmation*

This instance receives answer on request.

The following communication services are specified:

*SDO access services*

Upload SDO, Download SDO, Configure SDO timeout

*PDO access services*

Configure RPDO, Configure TPDO, Read PDO data, Write PDO data, *RPDO received*

*CANopen NMT services*

Start node, Stop node, Set node to pre-operational, Reset node, Reset communication, Enable node guarding, Disable node guarding, Start heartbeat consumer, Disable heartbeat consumer, *Error control event received*

*Device failure management services*

Read device error, *Emergency event received*

*CANopen interface configuration services*

Initialize gateway, Store configuration, Restore configuration, Set heartbeat producer,Set node-ID, Start emergency consumer, Stop emergency consumer

*Gateway management services*

Set default network, Set default node-ID, Get version

*Controller management services*

Reset controller, Start controller, Stop controller

The CANopen gateway may support one or more of the following classes:

*Class 1:* The gateway is a device, acting as network slave (NMT slave functionality) within the CANopen network. The device shall provide SDO client functionality.

*Class 2:* The gateway is a device implementing the functionality of a class1 device, which additionally implements SDO requesting device (SRD) functionality.

*Class 3:* The gateway is a device within the CANopen network acting as the CANopen manager.

The gateway device may support more than one CANopen network. If multiple CANopen network interfaces are implemented, the CANopen networks shall be numbered uniquely (e.g. CANopen Net 1, CANopen Net 2, etc.). In each CANopen network the connected CANopen devices are uniquely addressed by the node-ID. In each CANopen device the objects are addressed uniquely by the 16-bit index and the 8-bit sub-index.

**TCP/Modbus protocol**

The above-mentioned services are mapped to a Modbus function code known as the *CANopen general reference command.* This command is an encapsulation of the services that is used to access (read from or write to) the entries of a CANopen object dictionary as well as controlling/monitoring the gateway device, and other CANopen devices. The TCP/Modbus protocols are defined in CiA DSP 309-2.

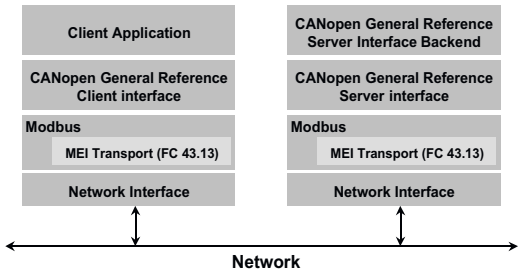


*Figure 4: Device view of both client and server modules*

The networked system is intended to work within the limitations of existing Modbus networks. Therefore, the information needed to query or modify the object dictionaries in the CANopen devices is mapped into the format of a Modbus message. The command has the 253-byte limitation in both the request and the response message. The shown figure illustrates how the CANopen general reference MEI type is incorporated into the Modbus encapsulated interface function code.

**ASCII protocol**

The standardized ASCII mapping of the communication services are specified in CiA DSP 309-3. The principle communication is based on non-case-sensitive ASCII

strings (ISO/IEC 646) instead of architecture and CPU/compiler depending binary structures. Due to this, no application handles with things like endianness, data size and byte alignment. In all cases where numbers are used, the typical representation is like in international standards (ISO/IEC 9899).

The CANopen gateway is controlled by commands. A command is composed of tokens, which are separated by any number of white-spaces and is closed with a CRLF. All commands are confirmed. Commands start with a sequence number which is enclosed by square brackets [ ]. The sequence number is a 4-byte value. It is not used for event-triggered messages. According to the addressing principle, a network number and a node number follow the sequence number. Network number and node number are optional, when the CANopen gateway only provides one Ethernet/CAN interface or when a client presets them. Commands that affect only the server not a remote node but a net and node are given net and node are ignored.

Net numbers are starting with 1. Node numbers are starting with 1. The value 0 for net or node is used to address all networks or all nodes. The token <value> designates a value of the possible CANopen data types. Within the description of the commands the sequence number is omitted for reasons of readability.

The CANopen gateway shall respond with the same sequence number at the first position as given by the request. This number shall be given in decimal format. There shall be only one response to a request. Messages due to errors in the CANopen network or the occurrence of communication objects using the producer-consumer principle shall not use a sequence number. The content of event-triggered messages is described within the command description that enables the specific service.

### Requesting and confirming Ethernet services

In order to request and confirm services from Ethernet devices by a CANopen device, corresponding communication services are required. There are no such services standardized yet. The following functions should be considered:

*Configuration services*

Upload configuration data, Download configuration data

*Real-time services*

Read real-time data, Write real-time data

*Network management services*

Start device, Stop device, Set device to pre-operational, Reset device

*Device failure management services*

Read device error

*TCP/IP interface configuration services*

Initialize gateway, Store configuration, Restore configuration

*Gateway management services*

Set default network, Set default device-ID, Get version

*Controller management services*

Reset controller, Start controller, Stop controller

These services may be implemented by dedicated Ethernet-based protocols such as generic TCP/IP, EtherCAT, Ethernet-Powerlink, etc.

CiA has not yet standardized CANopen protocols for accessing Ethernet devices by means of a gateway. There may be a need for generic TCP/IP networks as well as for dedicated EtherCAT and Ethernet Powerlink networks. The above proposed C(R)ARP may be considered as a candidate for generic TCP/IP networks.

### CANopen-Ethernet-CANopen bridges

Network architectures as shown in Figure 5 may require real-time capability, in particular in machine control systems. The real-time performance of the single CANopen sub-networks is predictable by worst-case scenarios as described in the literature. The same can be done for Ethernet networks with real-time capability (e.g. EtherCAT or Ethernet Powerlink). Additionally, the real-time capability and performance of the gateway has to be considered. The real-time performance of

the gateway is higher if the Ethernet network uses a CANopen-compliant object and data model.

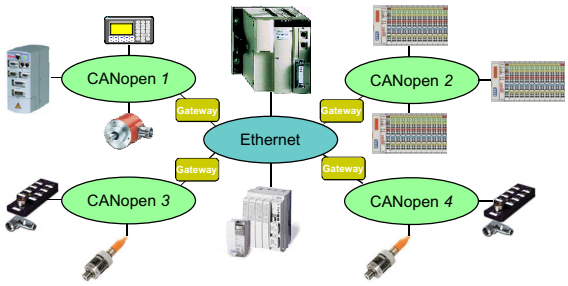


*Figure 5: Ethernet as integration platform*

Implementing the above introduced C(R)ARP, the gateway stores in the CARP cache the often used connection parameters. This increases the real-time performance. The CARP cache avoids the transmission of Ethernet broadcast messages to locate the destination device. Another approach could be pre-defined and fixed communication connections. However, such a static gateway implementation is suitable for smaller control systems without network reconfiguration requirements.

If the Ethernet network uses the CANopen object dictionary and the CANopen profile specifications, it is possible to design transparent gateways. This is that the user does not see in which network technology is used.

The simple example in Figure 6 shows that the Ethernet-connected controller communicates only with the gateway device that represents the drive and the input devices. On the CANopen side the gateway represents the controller. For devices it looks like there is only one network. However, the network performance is lower because you have to consider the delay times in both networks as well as in the gateway device. Nevertheless, in more complex control systems transparent gateway may be used to increase the number of connectable devices, to achieve longer network distances, or to separate sub-systems in order to reduce impacts between the sub-systems.

In non real-time applications, the Ethernet-based integration platform may be an Intranet or even the Internet. There have been already presented several papers on this topic in previous iCC conferences. In particular, the configuration, diagnostics, and software download may be performed via Intranet or Internet. In those applications, there are normally no real-time requirements.

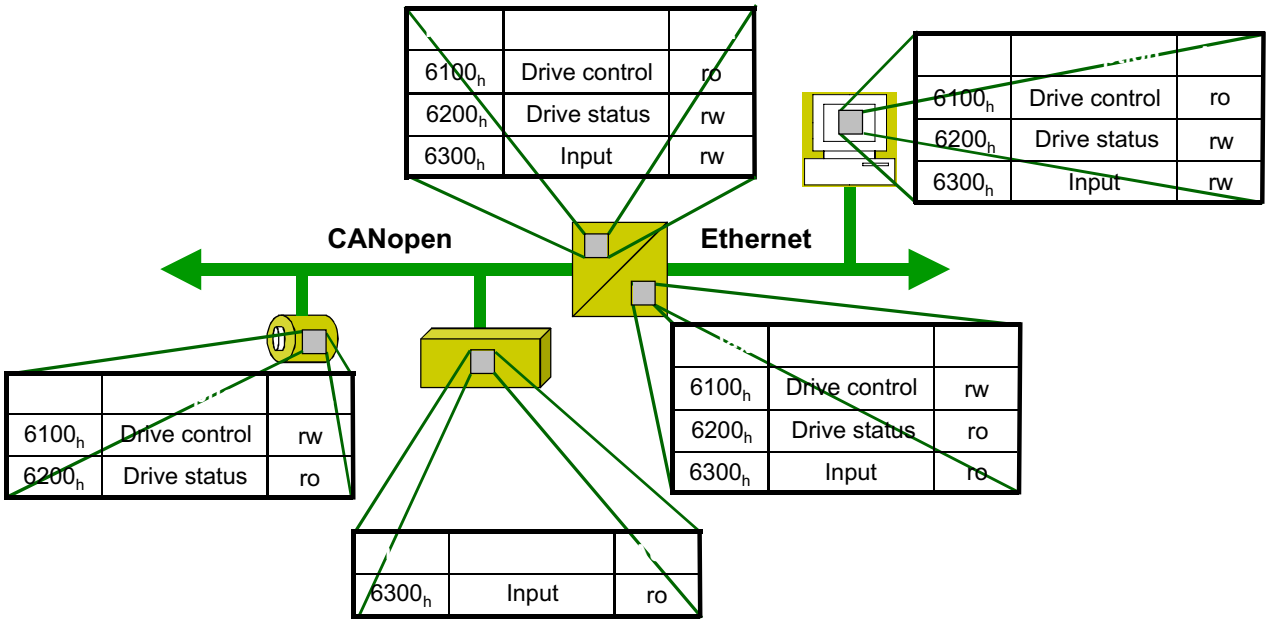


*Figure 6: Transparent CANopen/Ethernet gateway using CANopen object dictionary definitions and CANopen profile specifications*

## Literature

*CiA DSP 309-1:* Interfacing CANopen to TCP/IP – Part 1: General principles and services. Erlangen, 2004.

*CiA DSP 309-2:* Interfacing CANopen to TCP/IP – Part 2: TCP/Modbus mapping. Erlangen, 2004.

*CiA DSP 309-3:* Interfacing CANopen to TCP/IP – Part 3: ASCII mapping. Erlangen, 2004.

*RFC 826* – An Ethernet address resolution protocol (converting network protocol addresses to 48-bit Ethernet address for transmission on Ethernet hardware. 1982.

*H. Ekiz, A. Kutlu, M. D. Baba, E. T. Powner:* Design and implementation of a CAN/Ethernet bridge. iCC proceedings 1996.

*Dr. M. Wollschläger, St. Wehrmann:* Intranet-based management of CAN devices. iCC proceedings 1998.

*J. Randhahn, Prof. Dr. H. Beikirch:* CANeye – controlling and monitoring CAN via web interface. iCC proceedings 2003.

*Prof. G. Gruhler:* Remote control of CAN-based industrial equipment using Internet technologies. iCC proceedings 2003.

*J. V. C. Hernandez, J. J. Serrano, J. C. Campello, A. Bonastre, R. Ors, P. Bradbury:* A new control system for citric fruits conservation and muturation based on CAN and Internet networks. iCC proceedings 2003.

*Chr. F. Manga, Dr. H. B. Keller;* Kopplung verteilter Feldbus-Segmente über Ethernet – Simulation einer zeit- und sicherheitskritischen Kopplung. Elektronik 23/2004