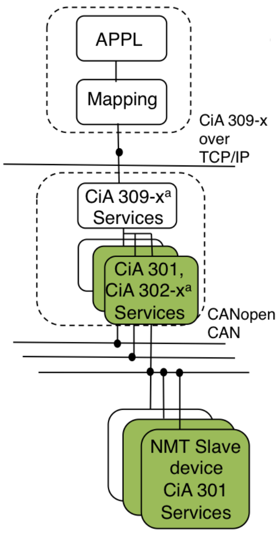
Enriching the Internet with CANopen

CANopen FD provides all attributes required for future embedded networking with the workflow defined by the CiA SIG CANopen IoT group and with the USDO functionality introduced by the CiA SIG application layer group.

Many application fields are faced with the fact that details of a traditional fieldbus system need to be accessible, e.g. for remote control via the end user's cell phone or for remote diagnostics. In addition, shortened development cycles make static, classic communication schema not feasible. To enable system maintainers to easily enhance their systems with modern devices and/or functionalities, a high degree of flexibility is required on embedded and deeply embedded system levels.

Because of tablets and smartphones, end users and system maintainers are used to a high degree of connectivity. Consequently, they would like use these devices to connect to their applications as well as for status inquires, triggering various system activities or doing system maintenance. The CiA working group SIG CANopen IoT is currently specifying a workflow that enables exactly such an accesses to a CANopen-based embedded system (see Figure 1).



Linking CANopen to the IoT:

Client application: Uses CANopen data objects and functionality addressed by reference designators.

Mapping between reference designators <-> MUX (MainIdx, SubIdx): The mapping function requires access to the nodelist.graphml and DCF files, located in the CANopen manager entity, using CiA 309-x[a] access commands. The actual CiA309-x[a] specification seems to be sufficient to meet the communication requirements.

CANopen gateway: CANopen manager within the gateway discovers available networks and devices and then stores the actual configuration to the objects 1F8x^{b}_{h} (NMT slave assignment, device type identification LSS address) and 1F2x^{b}_{h} (EDS/DCF files with reference designator extension as proposed before). The gateway generates the nodelist.graphml file based on the CANopen manager objects. The nodelist.graphml file has to be accessible using the object dictionary entry. The gateway behaves as a class 2 device according to CiA 309-1 (provides SRD functionality as specified in CiA 302-5) if the CANopen manager and the CANopen gateway are located in separate devices.

CANopen devices: The CANopen manager is able to integrate the devices during run time if the CANopen devices provide the appropriate reference designators themselves.

Figure 1: a: x is a specification parts number; b: x is a hex number

The main focus of this workflow is to integrate as many of the already existing CANopen solutions as possible. For an external access to CANopen embedded systems, the first step is to identify the embedded system. Any embedded device can be identified if it supports a correct and consistent electronic data sheet (EDS) and/or device configuration file (DCF). These EDSs and DCFs are enhanced by a reference designator (RefDesignator). The RefDesignator assigns an application-specific, logical name to the device or even to the device's parameters. A gateway between the embedded network and the Cloud gets system knowledge via EDSs/DCFs that are enhanced by RefDesignators. Along with the so-called nodelist information, GraphML-capable tools can provide the application-related visualization of the entire control system. An IoT application accesses the IoT-to-CANopen gateway using its application server functionality (web server). The IoT application then becomes capable of discovering the sub-layered system based on the nodelist.graphml files. As soon as the system is known, the IoT application can use the remote CANopen access services to reach any sub-layered CANopen device. The gateway can provide this knowledge at any interface; either to web clients or CANopen managers (see Figure 2).

As soon as the web client has knowledge of the system architecture, the web client can provide the user interface to the end user. With it the user is able to stimulate the embedded CANopen network. This covers e.g. reading and writing CANopen device parameters; typically for diagnostic or configuration purposes. To avoid reinventing the wheel, the CiA SIG CANopen IoT relies on the CANopen network access services, provided in CiA 309 specification series. With the objective of using the CiA 309 services in existing web applications, the CiA SIG CANopen IoT is translating the CiA 309-3 ASCII-protocol to URI queries. In addition, the group provides translation guidelines for the CiA 309-3 ASCII protocol to an XML-schema. Using these mappings in existing web services frees the end user from any CANopen-specifics. The intended functional addressing allows accessing embedded systems by means of logical commands. Based on the RefDesignators, e.g. a gateway can translate these commands to typical CANopen services.

To make full use of IoT applications, typically lots of data has to be communicated on the embedded level as well. CAN-based parts of IoT applications benefit from the high data throughput provided by CAN FD. In addition, the applications benefit from the high flexibility provided by CANopen FD's universal service data object (USDO). In contrast to classical SDOs, there is no need to preconfigure USDOs. The communication partner can be chosen per access. CANopen FD devices are enabled to exchange any amount of data with one, several or all CANopen devices in the network. In addition, the inherent routing capability ▷

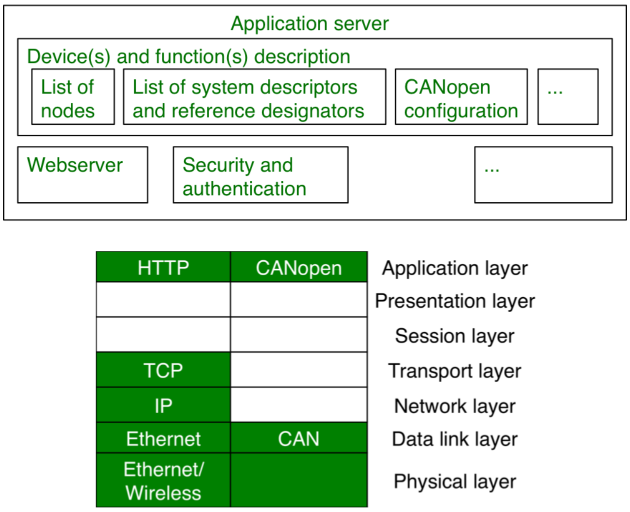


Figure 2: The IoT application can reach any sub-layered CANopen device

of USDOs is beneficial to IoT applications as well. Independently of the access point to a CANopen-based control architecture, any device can be reached as soon as the topology of the embedded network architecture is known. The CiA SIG application layer is currently specifying CANopen FD. It intends to publish the CiA 301 V5.0 by the end of this year.

In IoT applications often third parties require access to the system. To avoid any unauthorized access to the system, security issues have to be considered. CANopen already provides several measures with regard to security. Several CANopen application profiles offer parameters which let end users reveal themselves to the system as a member of a specific user group, with specific system access rights. For systems where these basic measures are not sufficient, the CiA working group TF Security develops more comprehensive security measures for authentication and encryption, suitable for CAN- and CANopen-based embedded systems.

CAN in Automation is currently merging and enhancing existing CANopen functionalities, to allow adding CANopen-based embedded systems in modern control architectures. All CiA working groups appreciate any contribution and support that enables the group to provide the CAN community with comprehensive CiA specifications and implementation guidelines. Contacts to other working group members, established during the CiA specification maintenance, can be beneficial, not only in the currently running CAN-based project but during an engineer's entire working life. ◀

Author

Reiner Zitzmann
CAN in Automation
headquarters@can-cia.org
www.can-cia.org