

# **A modular approach to a flexible CANopen service tool**

In recent years CANopen devices became increasingly advanced both in terms of the provided process I/O functionality and the implemented configuration mechanisms. Traditionally the device commissioning process was limited to the parameterization of the I/O and the communication functionality via the CANopen network, using either a stand-alone network configuration tool or a CANopen configuration manager integrated with the NMT (Network Management) master.

Today local layer parameters like node-ID and bit rate may be set using Layer Setting Services (LSS) as specified in CiA 305, and firmware updates are handled using the program download mechanism as described in CiA 302. Both these additional device commissioning tasks are not commonly addressed by CANopen network configuration tools.

In particular for firmware updates many device manufacturers therefore frequently opt for the implementation of proprietary solutions as the market offers only a very limited selection of off-the-shelf CANopen tools that could provide the framework for a dedicated device commissioning application. An important requirement is that such a framework has to present the underlying CANopen services at a very high level of abstraction as the end user is typically not proficient in the details of the CANopen protocols.

## **Requirements**

In addition to the generally available option to download device configuration data generated by a stand-alone CANopen network configuration tool and provided in form of standardized device configuration files (DCF), a generic device commissioning and service tool should thus include LSS master functionality and an implementation of the program download mechanism.

A second requirement to a dedicated device tool is that it should also allow to perform simple device-orientated diagnostics tasks. Here a full-blown CANopen analyzer is only of limited value as service staff typically does not have the CANopen background knowledge to be able to interpret CAN or CANopen trace files.

Beyond these more technical aspects the implemented functionality should be available in a very compact and highly mobile solution, preferably in form of a PDA.

Flexibility, expandability, and a user interface that is economical with the required screen dimensions were thus the main objectives during the development of the new generation of the IXXAT CANopen Device Manager.

To allow for a straightforward migration to mobile devices, Microsofts .Net Framework V2.0 and C# as programming language were selected as the principal development environments. Another advantage of using the .Net environment is the convenience of code and user interface generation from within UML based modeling tools.

## Modular approach

The discussion above suggests a modular approach based on a plug-in concept, in which the main executable component of the tool implements all the required CANopen communication mechanisms and services. The main module furthermore exports clearly defined interfaces to these CANopen services that are then used by dynamically installed plug-in modules. The specific configuration and diagnostics functionality, and the corresponding user interface controls, are provided by the individual plug-ins [Figure 1]. Plug-in modules are implemented as independent .Net assemblies, allowing for future functionality extensions without the need to modify the main framework component.

The advantage of this approach is that the tool can both provide the most commonly required CANopen services like network management, PDO, SDO device access, LSS and download of configuration data, but also be extended by additional functionality specifically orientated towards particular devices and applications.

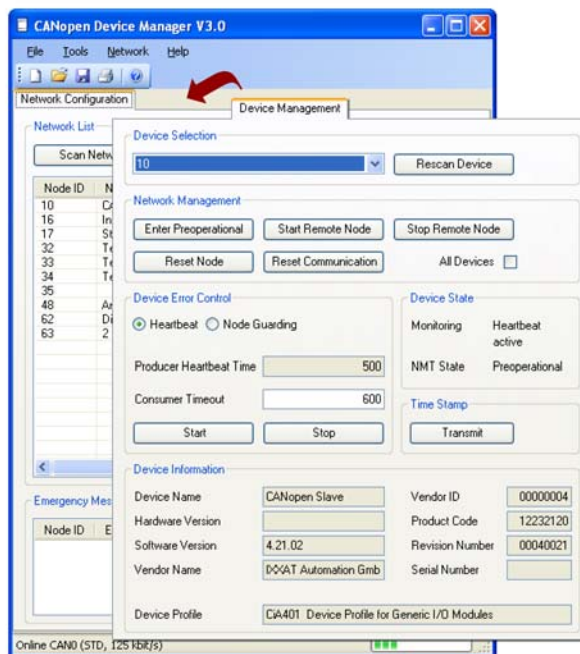


Figure 1: Dedicated plug-in modules complement the basic CANopen functionality covered by the primary CANopen Device Manager application.

## Applications

Currently plug-in modules are available for generic device management, device access [Figure 2], program and concise DCF download as well as LSS master functionality. While device management and access cover the traditional NMT, SDO (supported by a device object dictionary browser), PDO functionality, the download plug-in allows to update the firmware on correspondingly implemented CANopen devices using the program download mechanism specified in CiA 302 (requires the availability of objects 1F50<sub>h</sub> Program data, and 1F51<sub>h</sub> Program control).

The integrated LSS master capability complies with the recently published version 2.0 of the CiA 305 layer setting services and protocols specification.

For engineers implementing new devices the included functionality serves as an easy-to-use test tool to verify the correct operation of basic CANopen services, or as the framework for a complete end-of-line test application. Other possible applications are the implementation of device specific field-test functionality that may be used by service technicians to verify correct device operation in the field.

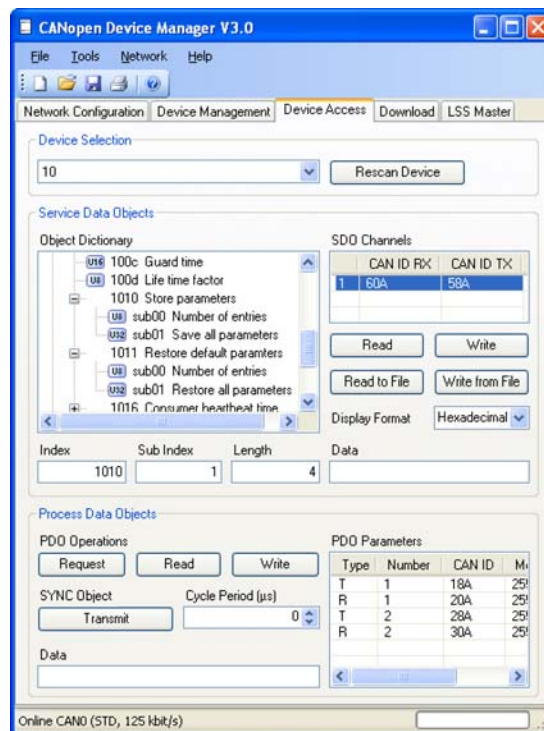


Figure 2: Device Access plug-in module providing both SDO and PDO access functionality.

## **Outlook**

The advantage of a modular, plug-in based approach is that that the customer receives a flexible framework that can be dynamically extended to meet current and future requirements to commissioning and servicing of CANopen devices. In addition to the already available plug-ins, IXXAT will provide dedicated add-on modules that cover the specific functionality of the most important CANopen device profiles, initially focusing on CiA 401, generic I/O devices, and CiA 402, drives and motion control. Further on the specification of the internal interfaces can be made available to OEM customers interested in the implementation of proprietary plug-in modules specifically targeted at their own devices.