

Automatic testing of CANopen devices

Kai Schmidt (Vector Informatik)

The growing complexity of today's system architectures is associated with an increase in the effort that must be invested in test specification, test creation and test execution during the development of such systems and system components. Test specifications should be available in early phases of the development process, e.g. after the system architecture has been created or during component design. This makes it possible to detect errors early and correct them cost-effectively.

The development process for a CANopen system can be described based on the V-model. In the first phase, system requirements are defined, which for the most part contain the definitions of individual "use cases". This information represents the input for the next step, where initial assessments can already be made of the system architecture. Functions are assigned to

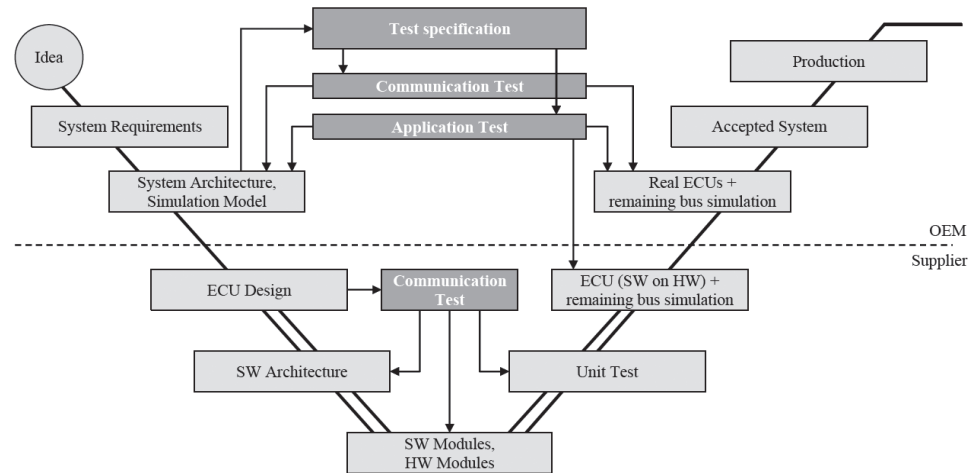


Fig. 1: Development process of a CANopen system

the individual ECUs, and device descriptions can be created for all devices in the form of EDS files (the format of the EDS files was standardized by CiA and is being further developed by this organization in cooperation with industry). In addition, communication relationships between the ECUs can be configured, as network management and error detection mechanisms. EDS files describe significant parts of the functional scope of a CANopen device. These device descriptions form the foundation for executing the simulation and creating test specifications. Communication-specific tests can be derived directly from the device descriptions. An example of this would be a test that checks all

objects in the object dictionary by SDO accesses and records the results. Besides communication-specific tests, application-related tests can also be specified. An example of such a test would be to stimulate the transmission of the digital input of an I/O device. Afterwards, a check is made to verify that the signal value exists at the output. Both tests could be used early in the simulated overall system. As soon as the stability of the overall system has been achieved, development of the individual components can be sub-contracted. The EDS files can – with the exception of application-related behavior – be considered as a requirements specification for the supplier. Parallel development of the ECUs at the suppliers is accompanied by the simulated overall system. Application-related tests can also be utilized at the supplier to test the behavior of the device to be developed within the overall system. This can signifi-

cantly reduce the number of cost-intensive changes desired by the OEMs – which generally occur within the integration phase. Communication-specific tests can be created at the supplier in a similar way as at the OEM.

After completion and acceptance of the components, they are successively integrated into the simulated overall system. The previously created communication and application-related tests can now be applied to the system, consisting of the physical components and the rest-of-bus simulation. As soon as all of the components have been delivered the concluding test of the real overall system follows.

EDS files as the basis for tests

The development process should include the creation of an EDS file appropriate to the device. Unfortunately, practice shows that device producers often ne-

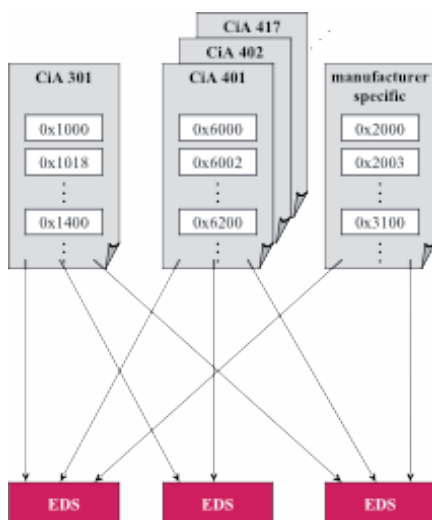


Fig. 2: Device functionality

glect this work step. Faulty or incomplete EDS files are the result; in the worst case there is no EDS file at all for a device. The development process described above shows that it is not just device producers who need to be concerned with creation of EDS files, but system designers too.

The task of the system designer here is to distribute functionalities to the individual components. These could be standardized functionalities such as mechanisms for process data communication, but they might also be manufacturer-specific functionalities. Both of these are mapped via objects in the object dictionary. CiA specifications describe standardized functions. Standardized parameters (process data, configuration and diagnostic data) as well as manufacturer-specific data can be stored in a database format that is also standardized. The necessary objects can be selected from the object pool that is created in this way, and be assembled into an object dictionary.

The device descriptions contain all of the information necessary for simulation of the CANopen device. The overall system, consisting of the individual device descriptions, is parameterized utilizing a suit-

able configuration tool, and an initial system description is obtained in the form of device configuration files (DCF), whose format has also been standardized by CiA. Based on this configuration, simulation models can be generated and executed in a suitable runtime environment. At an early point in the project, this already enables conclusions about the time behavior of the overall system. If excessive bus loads occur, for example, actions can immediately be initiated to correct the problem, since suppliers have not been involved in the development process yet. Accordingly, the simulated overall system offers a high degree of flexibility. It can be refined iteratively until it satisfies the defined requirements. Changes to the simulated system can be implemented cost-effectively and be checked immediately.

Derivation of test sequences

Besides the simulation, it is also possible to derive initial tests on the protocol and communication levels from the device descriptions. The protocol test includes checking of the SDO protocol, for example. The communication tests do not check for correctness of the

protocol, but instead for correct flow of message sequences. For example, it is possible to test whether the configuration sequence for process data objects conforms to the sequence specified in CiA 301.

The following test templates with general application can be defined for a CANopen device:

- ◆ SDO download test
- ◆ SDO upload test
- ◆ Heartbeat producer test
- ◆ Heartbeat consumer test
- ◆ Transmit PDO test
- ◆ EMCY test

Test functions are created for each object contained in the object dictionary here. The test functions are parameterized based on the data contained in the configuration files for the devices. Among other things, test sequences can be generated to check the:

- ◆ PDO configuration
- ◆ Default values
- ◆ Object dictionary
- ◆ NMT state machine, and
- ◆ SDO protocol

The generated tests may be executed right away in a suitable runtime environment. In the framework of integration work, it is precisely such tests that are used to check the delivered components. In turn, suppliers can generate similar test sequences to assist in development. They can immediately be applied to the

prototypes. Essentially, this is a way to generate test sequences of the conformance test (CiA 310) device-specifically. However, the goal of the system should not be to replace the CiA conformance test altogether. The system should accompany the development and give developers a way to test devices in advance of the actual tests. The final certification is only performed by CiA.

Generation template for each version

Generation templates must be created for each test, but they are applied to each device to be tested. A generation template that describes the creation of a test for checking the object dictionary would appear as follows:

```

for all objects
{
  get access type
  if(access == read
only){
    add test function
SDO Upload
    to test sequence
  } // if
  else if (access ==
read write){
    add test function
SDO Upload
    to test sequence

    add test function
SDO Download
    to test sequence
  } // else if
  .
  .
}
    
```

The generated test sequence created based on this test template contains a number of parameterized (by entry of object index, etc.) write and read routines. They are processed sequentially in test execution.

Iterative development process

Since iterative processes are applied throughout a device's development, the ▶

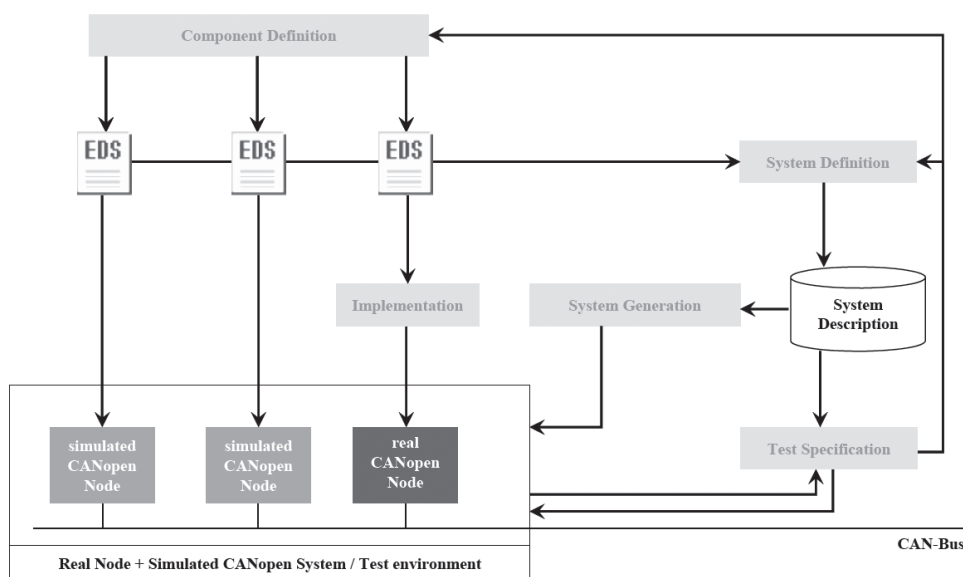


Fig. 3: Test sequences of a CANopen system

process for generating test sequences must be repeatable as often as needed. Changes to the device design can affect the device

es receive or send them. Among other things, precisely these aspects must be tested. This subject matter can be explained by the

Attribute	Value
Index	60B3 _h
Name	GPS date
Object code	Variable
Data type	Unsigned32
Category	See /CiA447-2/

31	26 25	20 19	16 15	0
<i>r</i>	<i>GPS date day</i>	<i>GPS date month</i>	<i>GPS date year</i>	

Fields	Value	Definition	Unit
<i>GPS date year</i>	0000 _h FFFD _h FFFE _h FFFF _h	Minimum value Maximum value Failure Signal not available	Years
<i>GPS date month</i>	00 _h 01 _h 0C _h 0D _h 0E _h 0F _h	Reserved Minimum value (January) Maximum value (December) Reserved Failure Signal not available	Months
<i>GPS date day</i>	00 _h 01 _h 1F _h 20 _h to 3D _h 3E _h 3F _h	Reserved Minimum value (1 st) Maximum value (31 st) Reserved Failure Signal not available	Days
<i>r</i>	11 1111 _b	Reserved	

Fig. 4: "GPS date" object description

descriptions. The test that was originally generated would then likely fail. Nonetheless, it is still necessary to be able to manually extend test sequences after generation, e.g. to incorporate application-specific supplements. These extensions must be read back when the sequence is re-generated.

Application-related tests

The application-related behavior of the devices can not be represented in the device descriptions. Furthermore, the tester does not want to have to deal with the CANopen-specific conceptual world and its definitions on the application level. On this level, it is entirely unimportant to know which signal is mapped to which object at which position. More important is information about which signals exist and which devic-

example of the CiA 447 application profile (application profile for special-purpose car add-on devices):

The standard defines an object "GPS date". Mapped to this object are the signals "GPS date year", "GPS date month" and "GPS date day".

The CiA 447 profile, besides defining signal allocations in the objects, also defines the transmission type. The standard specifies that the object value "GPS data" is transmitted by SDO protocol. The following information is needed to transmit a signal as part of a test:

- ◆ Index + Sub-Index of the object
- ◆ Signal length
- ◆ Start position of signal within the object

The format of today's EDS files is unable to describe the signal allocations of object values. Accordingly, information such as the signal length and start position of

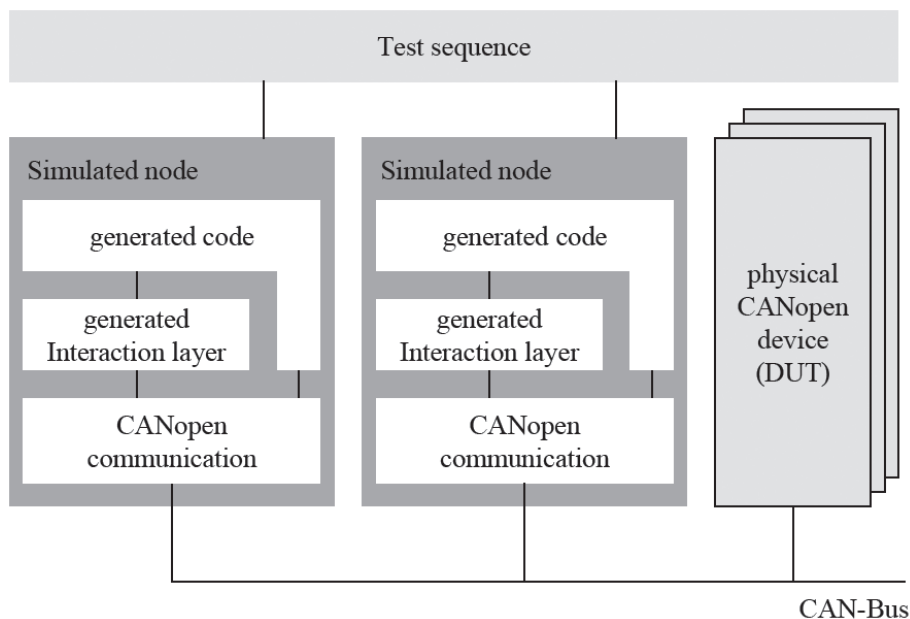


Fig. 5: Generated test environment

the signal is also unsupported. Even if these requirements could be implemented, it is not possible to automate generation of application-related test sequences, since the behavior of the system is not described.

Generated test environment

Nonetheless, the developer can be supported by gener-

ation of an “interaction layer” in test creation. If this extension can be integrated in the simulated overall system, then it is easy to create application-related test cases.

The test system consists of the simulated nodes that are extended to include an “interaction layer”. One or more physical devices are tested. The simulated devices are stimulat-

ed via generated interface functions. Signal values are mapped to object values and the CAN messages are sent. In the example depicted, the signal value “GPS date month” would be mapped to the relevant position in the object value (startbit 16, length 4 bit).

Parameterization of the test functions assumes that the positions and length of the signals are

known. Moreover, the transmission type must be considered. This information is described exclusively in the standard and must be considered in test creation. Use of an “interaction layer” enables signal-oriented test creation. It will be possible to define the function “Send_GPS_month” and generate its implementation based on the CiA 447 specification, if it exists in XML format in the future. Today’s format of the specification requires converting the specification to a readable format (XML or Excel).

This conversion task can be assumed by a generator. The generated functions contain a mapping of the signal to the object value and a routine for sending the CAN messages. During test creation, the test engineer need not be concerned about signal positions, indices or transmission types. All the test engineer is interested in are the signal name, sender, receiver and signal value.

kai.schmidt@
vector-informatik.de

The Codix 538 display by Kübler provides a CAN/CANopen interface in order to display locally any user-defined value. Numerical values can be directly scaled using a factor or offset from the display device. The display has a floating decimal point that can be inserted in any position. Via the CAN network encoders can be read out directly, thanks to the Automatic Operational Mode. The display is provides automatic bit-rate detection and up to 16 node-IDs that can be set via rotary switches. The CAN port supports base and extended frame formats with maximum data-rate of 1 Mbit/s. The display comes equipped with a 6-digit, 8-mm high-red 7-segment LED. Each segment can be directly written to, allowing not only val-

Display communicates with encoder

ues to be displayed but text messages also. There will always be room for the display, thanks to its DIN housing that measures 48 mm x 24 mm, with an installation depth of just 59 mm. Reverse polarity protection likewise facilitates installation. On account of its high IP65 protection rating the display can be easily used in industrial environments without the need for an additional protective housing. The company offers several encoders with CANopen connectivity since many years. The Sendix absolute encoder features now an additional incremental track. Parallel to the CANopen output this encoder outputs an additional TTL compatible signal with 2048 puls-

es per revolution. As well as the CANopen output, the encoder is also fitted with an RS-422 interface, which outputs the signals A₁/A₂, B₁/B₂. “This means it is possible to implement simultaneously both positioning via the CAN network and direct feedback of the speed – all in just one encoder,” said Pierre Brucker, Marketing Director at Kübler. “This saves on the costs and installation space that a second encoder would have entailed.” Using the variable PDO Mapping in the memory the user can decide, which information is to be available in real-time. Functions, such as the transmission of speed, acceleration or exiting the work area, ensure fast data availabil-

ity while reducing the load on the bus and controller. The real-time position acquisition of the expanded CANopen Sync function enables genuine time-synchronous position detection of several axes. Kübler is also launching slip rings (IST-SR085 and IST-SR060) that can be used to transmit bus signals between a stationary and rotating platform. Typical applications include cranes, rotary tables, etc. The transmission between stator and rotor units takes place via sliding contacts (for current up to 40 A). Thanks to their fully encapsulated housing (in high-grade glass-reinforced plastic), high protection rating (up to IP 64) and resistance to vibration, these rugged products are suited to industrial use.

www.kuebler.com