

CANopen Tests automatisch generieren

Mit zunehmender Komplexität heutiger Systemarchitekturen steigt auch der Aufwand, der bei der Entwicklung solcher Systeme bzw. Systemkomponenten in die Testspezifikation, -erstellung und -durchführung investiert werden muss. Testspezifikationen sollten bereits in frühen Phasen des Entwicklungsprozesses, beispielsweise nach der Erstellung der Systemarchitektur oder während des Komponentendesigns, verfügbar sein. Fehler können somit früh erkannt und kostengünstig beseitigt werden.

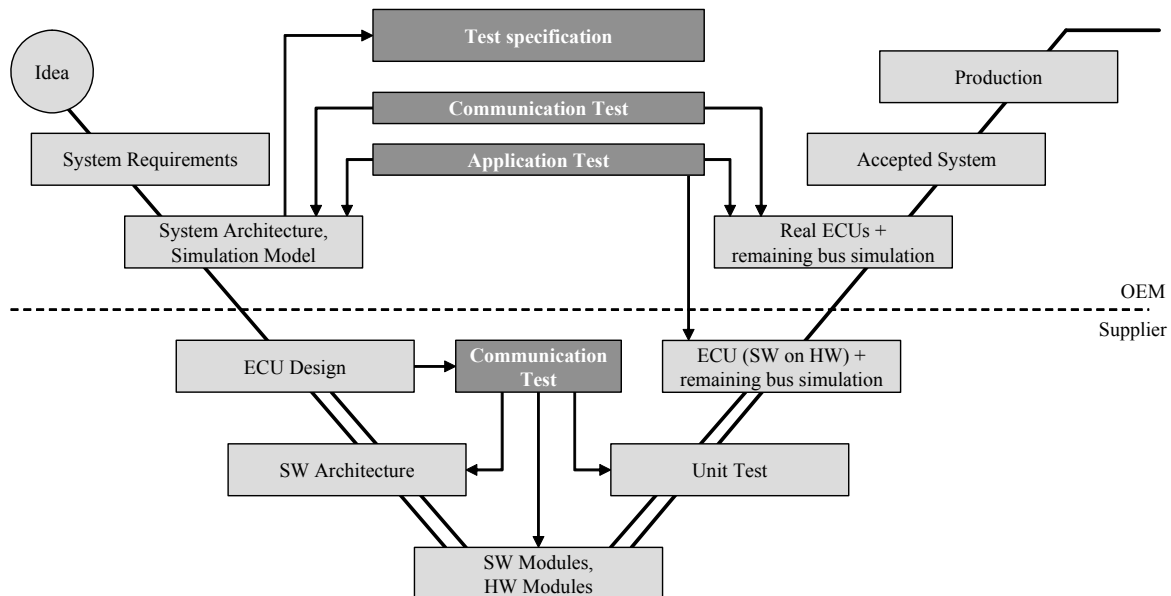
Gerätebeschreibungen können für CANopen-Systeme bereits nach der Definition der Komponentenarchitektur angefertigt werden. Gemeinsam mit der Systemdefinition bilden die Gerätebeschreibungen die Basis für die Erstellung von Testspezifikationen. Ausführbare Testsequenzen können hieraus abgeleitet und in einer entsprechenden Ablaufumgebung ausgeführt werden.

Definition des Gesamtsystems

Der Entwicklungsprozess eines CANopen-Systems kann anhand des V-Modells beschrieben werden (Bild 1). In der ersten Phase erfolgt die Festlegung der Systemanforderungen, die hauptsächlich die Definition einzelner „Use Cases“ beinhaltet [1]. Diese Informationen bilden den Input für Phase 2, wo bereits erste Aussagen über die Systemarchitektur getroffen werden können. Funktionen werden den einzelnen ECUs zugewiesen und Gerätebeschreibungen können in Form von EDS-Dateien (das Format der EDS-Dateien wurde vom CiA standardisiert und wird von dieser Organisation in Zusammenarbeit mit der Industrie weiterentwickelt) für alle Geräte erstellt werden. Außerdem können Kommunikationsbeziehungen zwischen den ECUs ebenso konfiguriert werden wie das Netzwerkmanagement und die

Mechanismen zur Fehlererkennung. Diese Festlegungen bilden die Grundlage für:

1. die Simulation des Gesamtsystems
2. die Erstellung von Testspezifikationen
3. das Pflichtenheft für Zulieferer



[Bild 1: Entwicklungsprozess eines CANopen-Systems]

Eine erste Testspezifikation

EDS-Dateien beschreiben wesentliche Teile des Funktionsumfangs eines CANopen-Gerätes. Diese Gerätebeschreibungen bilden die Grundlage für die Realisierung der Simulation und die Erstellung von Testspezifikationen. Kommunikationsspezifische Tests können direkt aus den Gerätebeschreibungen abgeleitet werden. Als Beispiel wäre hier ein Test zu nennen, der über SDO-Zugriffe die Zugriffstypen aller Objekte im Objektverzeichnis prüft und die Ergebnisse protokolliert. Neben den kommunikationsspezifischen Tests können außerdem applikative Tests spezifiziert werden. Ein solcher Test könnte beispielsweise die Übertragung des digitalen Eingangs eines I/O-Gerätes stimulieren. Geprüft würde anschließend, ob der entsprechende Signalwert am Ausgang des Empfängers

vorhanden ist. Beide Tests könnten bereits auf das simulierte Gesamtsystem angewendet werden. Sobald die Stabilität des Gesamtsystems gewährleistet ist, kann die Entwicklung der Einzelkomponenten in Auftrag gegeben werden (Phase 3). Die EDS-Dateien können – mit Ausnahme des applikativen Verhaltens – als Pflichtenheft für die Zulieferer betrachtet werden. Die parallele Entwicklung der ECUs bei den Zulieferern wird durch das simulierte Gesamtsystem begleitet. Die applikativen Tests können auch beim Zulieferer herangezogen werden, um das Verhalten des zu entwickelnden Gerätes im Gesamtsystem zu prüfen. Die Anzahl kostenintensiver Änderungswünsche seitens des OEMs – die in der Regel im Rahmen der Integrationsphase auftreten – können dadurch deutlich reduziert werden. Kommunikationsspezifische Tests können beim Zulieferer in ähnlicher Art und Weise erstellt werden wie beim OEM.

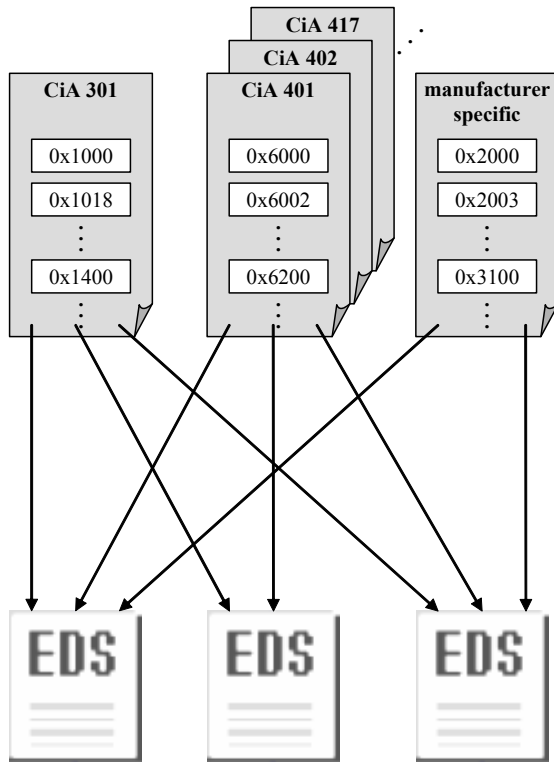
Integration der Komponenten

Nach der Fertigstellung und der Abnahme der Komponenten gehen diese in Phase 4 beim OEM ein. Hier werden die Komponenten sukzessive in das simulierte Gesamtsystem integriert. Die zuvor erstellten Kommunikations- und applikativen Tests können nun auf das System, bestehend aus den physikalischen Komponenten und der Restbussimulation, angewendet werden. Wurden alle Komponenten geliefert, erfolgt in Phase 5 der abschließende Test des realen Gesamtsystems.

EDS-Dateien als Generierungsbasis

Der Entwicklungsprozess sollte die Erstellung einer zum Gerät passenden EDS-Datei beinhalten. Leider zeigt die Praxis, dass Gerätehersteller diesen Arbeitsschritt oftmals vernachlässigen. Fehlerhafte oder unvollständige EDS-Dateien sind die Folge, im schlechtesten Fall existiert für ein Gerät gar keine

EDS-Datei. Der oben beschriebene Entwicklungsprozess zeigt, dass sich nicht nur die Gerätehersteller sondern auch die Systemdesigner mit der Erstellung von EDS-Dateien befassen müssen.



[Bild 2: Verteilung der Geräte-Funktionalitäten]

Aufgabe des Systemdesigners ist hierbei die Verteilung der Funktionalitäten auf die einzelnen Komponenten. Dabei kann es sich um standardisierte Funktionalitäten wie beispielsweise Mechanismen zur Prozessdatenkommunikation aber auch um herstellerspezifische Funktionalitäten handeln. Beide werden über Objekte im Objektverzeichnis abgebildet. Standardisierte Funktionalitäten werden vom CiA in Standardisierungsdokumenten beschrieben. Sowohl die in diesen Dokumenten beschriebenen Objekte als auch die herstellerspezifischen Objekte können in einem ebenfalls standardisierten Datenbankformat abgelegt werden. Aus dem somit entstandenen Objektpool können die notwendigen Objekte selektiert und zu einem Objektverzeichnis zusammengestellt werden.

Flexibilität der Simulation

Die Gerätebeschreibungen beinhalten alle zur Simulation der CANopen-Geräte notwendigen Informationen. Das Gesamtsystem, bestehend aus den einzelnen Gerätebeschreibungen, wird unter Verwendung eines entsprechenden Konfigurationswerkzeuges parametrisiert und man erhält eine erste Systembeschreibung in Form von Gerätekonfigurationsdateien (DCF). Deren Format wurde ebenfalls vom CiA standardisiert. Basierend auf dieser Konfiguration können Simulationsmodelle generiert und in einer entsprechenden Ablaufumgebung ausgeführt werden. Somit lassen sich bereits zu einem frühen Zeitpunkt des Projekts Aussagen über das zeitliche Verhalten des Gesamtsystems treffen. Treten beispielsweise zu hohe Buslasten auf, können umgehend Maßnahmen zur Problembeseitigung eingeleitet werden, da die Zulieferer in dieser Phase noch nicht in den Entwicklungsprozess einbezogen wurden. Das simulierte Gesamtsystem bietet demnach ein hohes Maß an Flexibilität. Es kann iterativ verfeinert werden, bis es den definierten Anforderungen genügt. Änderungen am simulierten System können kostengünstig umgesetzt und sofort geprüft werden.

Ableiten von Testsequenzen

Neben der Simulation können erste Tests auf Protokoll- und Kommunikationsebene aus den Gerätebeschreibungen abgeleitet werden. Der Protokolltest umfasst dabei beispielsweise die Überprüfung des SDO-Protokolls. Die Kommunikationstests prüfen nicht nur die Richtigkeit des Protokolls sondern auch die korrekte Abfolge von Botschaftssequenzen. Beispielsweise kann geprüft werden, ob die in CiA 301 spezifizierte Konfigurationsreihenfolge für Prozessdatenobjekte eingehalten wird [2]. Für ein CANopen-Gerät können folgende allgemein gültige Testvorlagen definiert werden:

- SDO Download Test
- SDO Upload Test
- Heartbeat Producer Test
- Heartbeat Consumer Test
- Transmit PDO Test
- EMCY Test

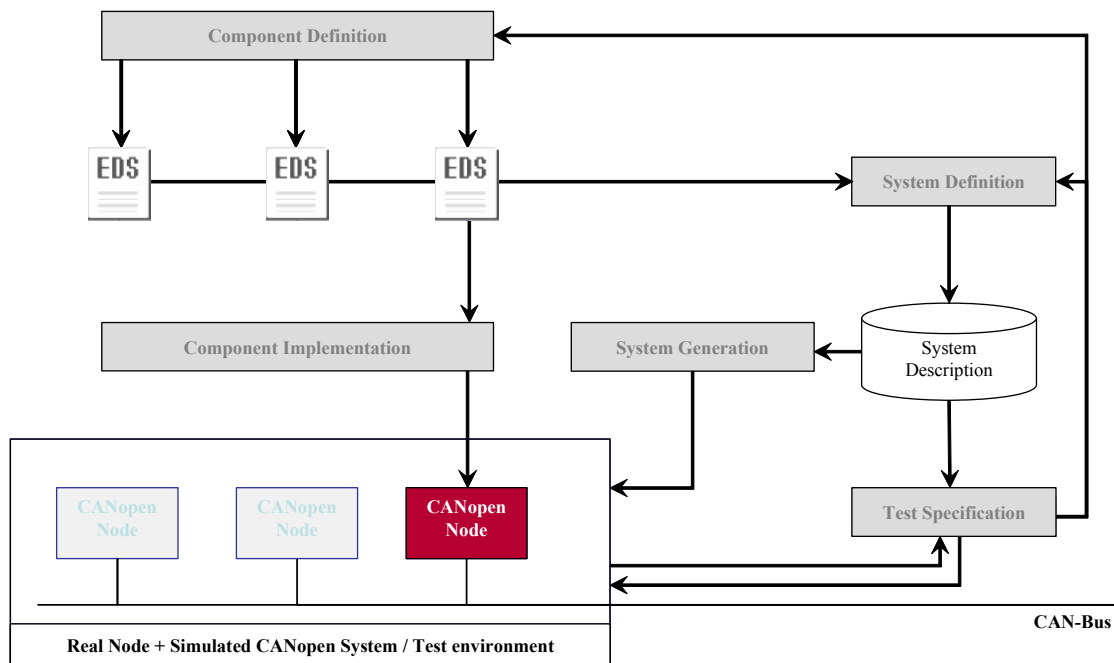


Bild 3: Entwicklungsprozess eines CANopen-Systems

Über einfache Generierungsvorlagen können gerätespezifische Testsequenzen erstellt werden. Testfunktionen werden dabei für jedes im Objektverzeichnis enthaltene Objekt angelegt. Parametriert werden die Testfunktionen anhand der in den Konfigurationsdateien der Geräte enthaltenen Daten. Unter Anderem können Testsequenzen zur Überprüfung:

- der PDO-Konfiguration,
- der Default-Werte,
- des Objektverzeichnisses,
- der NMT-Zustandsmaschine
- und des SDO-Protokolls

generiert werden. Die generierten Tests können umgehend in einer dafür geeigneten Ablaufumgebung durchgeführt werden. Im Rahmen der Integration werden genau diese Tests verwendet, um die gelieferten Komponenten zu überprüfen. Die Zulieferer wiederum können ähnliche, entwicklungsbegleitende Testsequenzen generieren. Diese können umgehend auf die Prototypen angewendet werden. Prinzipiell lassen sich somit die Testabläufe des Conformance-Test (CiA 310) knotenspezifisch generieren. Ziel des Systems soll jedoch nicht sein, den Conformance-Test des CiA zu ersetzen. Das System soll entwicklungsbegleitend fungieren und Entwicklern die Möglichkeit bieten, die Geräte im Vorfeld des eigentlichen Testes zu prüfen. Die abschließende Zertifizierung wird weiterhin nur vom CiA vorgenommen.

Generierungsvorlagen

Generierungsvorlagen sind für jeden Test einmalig zu erstellen, können aber auf jedes zu testende Gerät angewendet werden. Eine Generierungsvorlage, die die Erstellung eines Tests zur Überprüfung des Objektverzeichnisses beschreibt, würde wie folgt aussehen:

```
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
.
.
}
```

Die generierte Testsequenz, die auf Basis dieser Testvorlage erstellt wird, enthält eine Reihe von parametrisierten (Angabe von Objektindex usw.) Schreib- und Leseroutinen. Diese werden bei der Testdurchführung sequenziell abgearbeitet.

Iterativer Entwicklungsprozess

Da bei der Entwicklung eines Gerätes durchaus iterative Prozesse angewendet werden, muss der Prozess zur Generierung der Testsequenzen beliebig oft wiederholbar sein. Änderungen im Gerätedesign können sich auf die Gerätebeschreibungen auswirken. Der ursprünglich generierte Test würde dann mit hoher Wahrscheinlichkeit fehlschlagen. Trotzdem besteht die Anforderung, Testsequenzen nach der Generierung manuell zu erweitern, beispielsweise um anwendungsspezifische Ergänzungen einzubauen. Diese Erweiterungen müssen bei erneuter Generierung der Sequenz zurück gelesen werden. Erweiterungen sind in Schnittstellenfunktionen umzusetzen, deren Aufruf- und Funktionsdefinitionen bereits in die Testsequenz hineingeneriert werden:

```
Test start

// check object 0x1000
PreSdoUpload_1();
SDOUpload_1(0x1000,...);
PostSDOUpload_1();

// check object 0x1017
PreSdoUpload_2();
```

```

SDOUpload_2(0x1017,...);
PostSDOUpload_2();
PreSdoDownload_2;
SDODownload_2(0x1017,...);
PostSDODownload_2();
.
.
// check object k
PreSdoDownload_n();
SDODownload_n(0x1017,...);
PostSDODownload_n();
    
```

Test stop

Applikative Tests

Das applikative Verhalten der Geräte kann nicht in den Gerätebeschreibungen abgebildet werden. Außerdem möchte sich der Tester auf Applikationsebene nicht mit der CANopen-spezifischen Begriffswelt und deren Definitionen auseinandersetzen. Auf dieser Ebene ist es völlig uninteressant, in welchem Objekt welches Signal an welcher Stelle abgebildet ist. Interessant ist vielmehr die Information, welche Signale existieren und von welchen Geräten sie empfangen bzw. gesendet werden. Unter Anderem müssen genau diese Aspekte getestet werden. Am Beispiel des Applikationsprofils CiA 447 (Application profile for special-purpose car add-on devices) kann dieser Sachverhalt verdeutlicht werden:

Der Standard definiert ein Objekt „GPS date“. In diesem Objekt sind die Signale „GPS date year“, „GPS date month“ und „GPS date day“ abgebildet.

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	

[Bild 4: Beschreibung des Objektes „GPS date“]

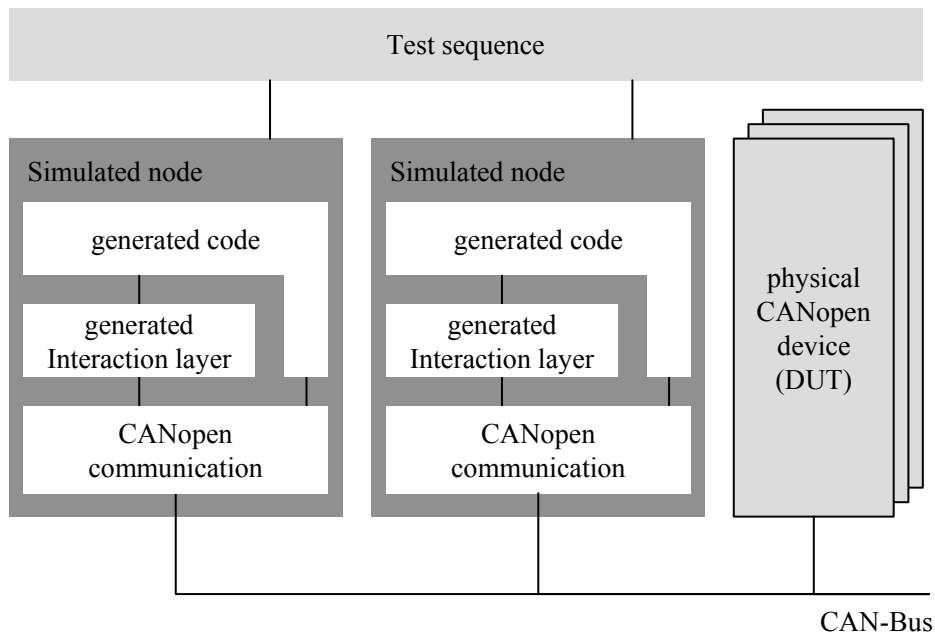
Das CiA 447 Profil legt neben der Signalbelegung der Objekte auch die Übertragungsart fest. Der Standard definiert, dass der Objektwert „GPS data“ per SDO-Protokoll übertragen wird. Für die Übertragung eines Signals im Rahmen des Tests werden folgende Informationen benötigt:

- Index + Subindex des Objektes
- Signallänge
- Startposition des Signals innerhalb des Objekts

Das Format heutiger EDS-Dateien bietet keine Möglichkeit zur Beschreibung der Signalbelegung von Objektwerten. Demnach werden auch Informationen wie Signallänge und Startposition des Signals nicht unterstützt. Auch wenn diese Anforderungen realisiert werden, ist die Generierung applikativer Testabläufe nicht automatisierbar, da das Verhalten des Systems nicht beschrieben ist.

Eine generierte Testumgebung

Dennoch kann der Entwickler durch die Generierung eines „interaction layer“ bei der Testerstellung unterstützt werden. Wenn diese Erweiterung in das simulierte Gesamtsystem eingebunden werden kann, können applikative Testfälle einfach erstellt werden.



[Bild 5: Generierte Testumgebung]

Das Testsystem besteht aus den simulierten Knoten, die um einen „interaction layer“ erweitert werden. Getestet werden ein oder mehrere physikalische Geräte. Über generierte Schnittstellenfunktionen werden die simulierten Geräte stimuliert: Signalwerte werden auf Objektwerte abgebildet und die CAN-Botschaften werden gesendet. Im aufgeführten Beispiel würde der Signalwert „GPS date month“ an die entsprechende Stelle im Objektwert (Startbit 16, Länge 4 Bit) abgebildet. Ohne diese Abbildung würde der Testschritt zum Senden des Signals „GPS data month“ = 12 (Dezember) wie folgt aussehen:

```
// write signal "GPS data month"
SDO Download(src,
             dst,
             0x60B3,
             0xC0000);
```

Parameter eins und zwei legen den Sende- bzw. Empfangsknoten fest. Im Parameter drei wird das entsprechende Objekt angegeben. Parameter vier beschreibt den Objektwert, in den das Signal (Signalwert = 12) abgebildet wurde. Die Parametrierung der Testfunktionen setzt voraus, dass die Positionen und die Länge der Signale bekannt sind. Außerdem muss die Übertragungsart berücksichtigt werden. Diese Informationen sind ausschließlich im Standard beschrieben und müssen bei der Testerstellung berücksichtigt werden. Die Verwendung eines „interaction layer“ ermöglicht eine signalorientierte Testerstellung. Folgende Testfunktion wird zum Senden des Signals „GPS data month“ verwendet:

```
Send_GPS_month(src,dst,12);
```

Die Funktionsdefinition „Send_GPS_month“ und deren Implementierung kann basierend auf der CiA 447 Spezifikation generiert werden, wenn diese zukünftig im XML-Format vorliegt. Das heutige Format der Spezifikation erfordert eine Umsetzung der Spezifikation in ein lesbares Format (XML oder Excel). Die Realisierung dieser Umsetzung kann von einem Generator übernommen werden. Die generierten Funktionen beinhalten die Abbildung des Signals auf den Objektwert und die Routine zum Senden der CAN-Botschaften. Der Testingenieur befasst sich während der Testerstellung nicht mit Signalpositionen, Indizes oder Übertragungsarten. Für ihn sind lediglich der Signalname, Sender, Empfänger und der Signalwert interessant.

Zusammenfassung

Immer kürzer werdende Entwicklungszyklen setzen einen immer effektiver werdenden Entwicklungsprozess voraus. Gravierende Fehler, die erst während der Integrationsphase lokalisiert werden, treiben die Projektkosten in die Höhe. Entwicklungsbegleitende Testszenarien können dazu beitragen, Fehler

möglichst früh zu erkennen und kostengünstig zu beheben. Basierend auf EDS-Dateien können Testszenarien zur Überprüfung des Kommunikationsverhaltens gerätespezifisch automatisch generiert werden. Die Generierung applikativer Tests ist nicht möglich, da der Beschreibung des Applikationsverhaltens im Normalfall keine standardisierte Syntax zugrunde liegt. Dennoch können Testingenieure durch die Generierung eines „interaction layer“ bei der Testerstellung unterstützt werden. Grundlage hierfür bilden die vom CiA standardisierten Applikationsprofile. Die Verwendung eines „interaction layer“ ermöglicht eine einfache, schnelle und fehlerfreie Erstellung beliebig komplexer Testszenarien.

Literaturverzeichnis:

- [1] Jürgen Klüser, Vector Informatik GmbH: Test Requirements in Networked Systems
- [2] Mirko Tischer, Vector Informatik GmbH: Prototyping and testing CANopen systems
- [3] CiA 447, Application profile for special-purpose car add-on devices

Autor:

Kai Schmidt
Vector Informatik GmbH
Ingersheimer Str. 24
D-70499 Stuttgart
www.vector-informatik.de

Tel. +49 711 / 80670-0
Fax +49 711 / 80670-249
E-Mail: kai.schmidt@vector-informatik.de
www.canopen-solutions.de