



DRIVECOM

DriveServer-Leitfaden CANopen

Entwicklung eines DriveServer-kompatiblen OPC-
Busservers für CANopen-Anschaltbaugruppen

Version 1.0, 23. April 2003

Herausgeber: DRIVECOM Nutzergruppe e.V.
Postfach 1102
D-32817 Blomberg
Telefon : 0 52 35 / 3-4 18 64
Fax : 0 52 35 / 3-4 18 62
Internet : <http://www.DRIVECOM.org>

Alle Rechte, auch die der Übersetzung, vorbehalten. Kein Teil dieser Information darf in irgendeiner Form (Druck, Fotokopie, Mikrofilm oder einem anderen Verfahren) ohne schriftliche Genehmigung der DRIVECOM Nutzergruppe e.V., reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.

Änderungen vorbehalten



Verfasser:

Hersteller von Antriebstechnik:

| | | |
|-----------|------------------|---------------|
| Dunker | SEW-Eurodrive | Bruchsal |
| Leurs | Rexroth Indramat | Lohr am Main |
| Mirbach | Lenze | Hameln |
| Pollmeier | ESR Pollmeier | Ober-Ramstadt |
| Senneka | Stöber | Pforzheim |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

Hersteller von Feldbus-Anschaltbaugruppen:

| | | |
|---------|------------------|-------------|
| Hadlich | ifak system GmbH | Magdeburg |
| Lange | Softing | Haar |
| Wagner | Hilscher | Hattersheim |

Anmerkungen und Kommentare sind an Herrn Stefan Pollmeier gl@esr-pollmeier.de zu richten.

Änderungshistorie:

| Version | Name | Firma | Kommentar |
|---------|---------|-------|---------------------------------|
| 0.1 | Mirbach | Lenze | Erstellung |
| 0.2 | Mirbach | Lenze | Ergänzung um das Zustandsmodell |
| 1.0 | Mirbach | Lenze | Freigabe erteilt |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |



Inhalt

| | | |
|-------|--|----|
| 1 | Einleitung | 4 |
| 1.1 | Ziele dieses Dokumentes..... | 4 |
| 1.2 | Bezugsquellen..... | 4 |
| 2 | Architektur..... | 5 |
| 3 | Anforderungen an CANopen-Busserver..... | 6 |
| 3.1 | Automatisches Erfassen der am Bus angeschlossenen Geräte | 6 |
| 3.2 | DRIVECOM-OPC-Items | 6 |
| 3.3 | Registrierung..... | 7 |
| 3.4 | Zustandsmodell | 7 |
| 3.5 | Parameterdaten..... | 8 |
| 3.5.1 | Einfache SDO-Dienste versus Domain-Dienste | 8 |
| 3.5.2 | Datentypen..... | 8 |
| 3.6 | Prozeßdaten | 9 |
| 3.6.1 | Prozeßdatenlänge..... | 9 |
| 3.6.2 | Sync-Message..... | 9 |
| 3.6.3 | Offene Fragen..... | 9 |
| 4 | Glossar..... | 10 |
| 5 | Literatur | 11 |



1 Einleitung

Im Rahmen der DRIVECOM-Nutzergruppe wurde die DriveServer-Spezifikation /3/ erstellt. Sie basiert auf der OPC-Spezifikation /1/ und unterscheidet zwischen den Busservern, die feldbuspezifisch sind, und den DriveServern, die gerätespezifisch, aber feldbusunabhängig sind. Dieses Dokument erläutert, welche Anforderungen speziell an einen CANopen-OPC-Server gestellt werden, um kompatibel zur DriveServer-Spezifikation der DRIVECOM zu sein.

1.1 Ziele dieses Dokumentes

Die DriveServer-Spezifikation deckt nur diejenigen Teile ab, die feldbusunabhängig sind. Um sicherzustellen, daß Busserver-Implementierungen vollständig kompatibel und damit austauschbar sind, werden ergänzend zur DriveServer-Spezifikation in diesem Dokument die besonderen Belange für das Feldbussystem CANopen aufgeführt. Zielgruppe dieses Dokumentes sind also die Hersteller von Felbuskarten und OPC-Servern, die CANopen unterstützen.

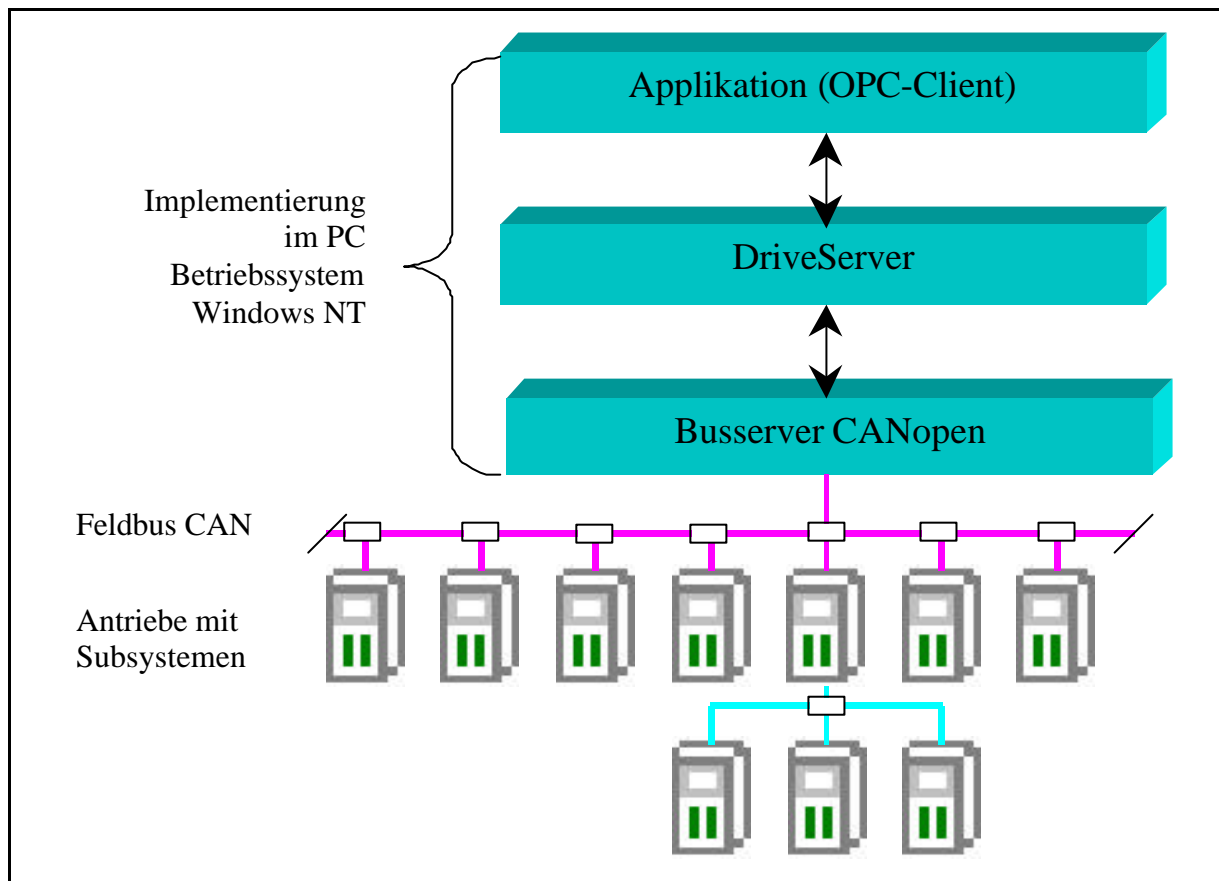
1.2 Bezugsquellen

Es liegt im Interesse der Antriebshersteller, die Unterstützung der DriveServer-Spezifikation bei möglichst vielen Herstellern von Felbusprodukten zu erlangen. Es besteht daher die Möglichkeit, Testmaterialien und Dokumentationen von den beteiligten Firmen zu erhalten. Im einzelnen handelt es sich um folgende Materialien und Bezugsquellen:

| Material | Bezugsquelle |
|---|--|
| DriveServer im Objectcode | kann vom jeweiligen Antriebshersteller bezogen werden |
| Generischer DRIVECOM-DriveServer im Quellcode | kann durch Beitritt zur DRIVECOM-Nutzergruppe erworben werden |
| Busserver und Anschaltbaugruppen | wird durch die Hersteller der Felbuskarten vertrieben |
| Testgeräte | werden von den Antriebsherstellern bei Bedarf zur Verfügung gestellt |

2 Architektur

Die hierarchische DriveServer Architektur kapselt die zwei verschiedenen Ebenen im Kommunikationssystem: die Antriebsebene und die Feldbusebene. Der DriveServer ist zunächst selbst Client eines Kommunikationsservers, im folgenden Busserver genannt, da die Kommunikationsschnittstellen typischerweise zu einem Feldbus bestehen, in diesem Fall zum CAN-Bus mit Kommunikationsprofil CANopen.



3 Anforderungen an CANopen-Busserver

3.1 Automatisches Erfassen der am Bus angeschlossenen Geräte

Beim Start des Busserver ermittelt dieser die angeschlossenen Geräte. Dazu versucht er, auf allen denkbaren Geräteaddresses das CANopen-Objekt 1000hex zu lesen. Abhängig davon, ob eine Antwort vom Gerät empfangen wird, kann die Information entnommen werden, auf welchen Busadressen Teilnehmer angeschlossen sind. Dieses Verfahren funktioniert selbst dann, wenn das Gerät das Objekt 1000 nicht unterstützt und ein NACK zurückliefert!

Alternativ besteht auch die Möglichkeit, daß der Busserver konfiguriert wird. In diesem Fall wird die Erfassung der Geräte nur bei der Erstellung der Konfiguration durchgeführt und nicht bei jedem Start. Nachteil ist allerdings, daß bei einem häufig wechselnden Umfeld stets an mehreren Stellen konfiguriert werden muß.

3.2 DRIVECOM-OPC-Items

Die DriveServer-Spezifikation sieht vor, daß für jedes am Bus angeschaltete Gerät einige OPC-Items im Namensraum anzulegen sind (vgl. /3/ Kapitel 3.3.2.1) Das Anlegen dieser Items übernimmt der Busserver automatisch während der Start-Phase. Es handelt sich dabei um die folgenden Items:

| Name des Items | Beschreibung | M/O |
|-------------------|---|-----|
| DS_Vendorname | Enthält die Herstellerkennung, die von der CiA festgelegt werden und im Internet unter http://www.can-cia.com/ einsehbar sind. Die Information kann aus Objekt 1018hex Subindex 1 entnommen werden. Unterstützt das Gerät nicht das Profil DS301 /2/ und kennt dieses Objekt folglich nicht, so bleibt dieses Item leer, ist aber trotzdem im Namensraum vorhanden. | M |
| DS_DeviceName | Enthält den Gerätetyp. Die Information kann aus Objekt 1008hex Subindex 0 entnommen werden. | M |
| DS_DeviceID | Enthält die Geräteadresse, die beim Erfassen der Geräte ermittelt wurde. | M |
| DS_OperatingState | gibt den Betriebszustand des Busses wider; vgl. 3.4 | O |

Unterstützt ein OPC-Server neben CANopen noch einen anderen Feldbus, so muß er zusätzlich das Item DS_BusSystem implementieren. Ein Client kann dann ermitteln, ob das fragliche Gerät am CANopen oder einem anderen Bus angeschlossen ist.

Kann ein OPC-Server mehrere voneinander unabhängige Busstränge (z.B. 2 PC-Karten oder Karten mit mehreren Kanälen) gleichzeitig ansprechen, so muß zusätzlich das Item DS_BusPort implementiert werden.

Ferner kann der Busserver dem Client noch mitteilen, welches Profil das Gerät unterstützt. Dazu kann optional das Item DS_ProfileID verwendet werden.

3.3 Registrierung

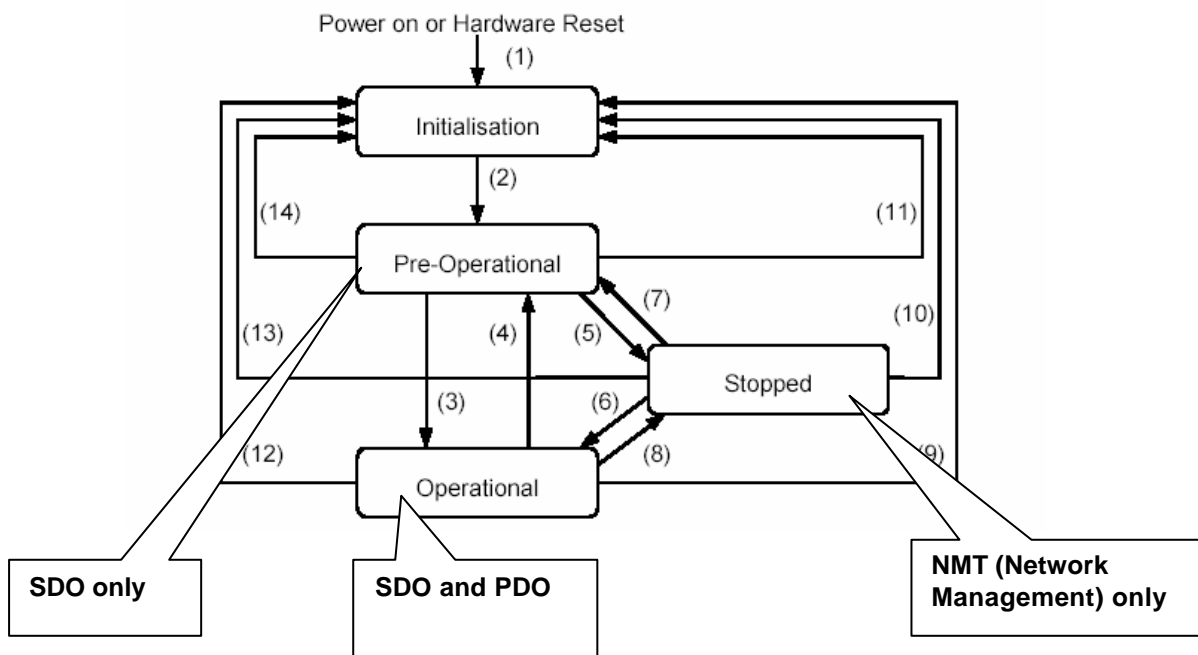
Ein CANopen-Busserver muß beim Installieren die folgenden Component Categories in die Registry eintragen:

- CATID_BusServer10
- CATID_CANOPEN

Die Werte der oben genannten Component Categories sind aus /3/ Kapitel 4 zu entnehmen.

3.4 Zustandsmodell

Das CANopen-Zustandsmodell umfaßt 4 Phasen. Da es für einen OPC-Client von Interesse sein kann, in welchem Zustand sich ein CANopen-Gerät befindet, kann dieser über das OPC-Item DS_OperatingState abgefragt werden.



Als Wert des Items DS_OperatingState sind die folgenden Textkonstanten zulässig:

Initialisation, Pre-Operational, Operational, Stopped



3.5 Parameterdaten

Die DriveServer-Spezifikation definiert den Zugriff auf Parameter über eine Index / Subindex-Adressierung. Diese Adressierung läßt sich direkt auf CANopen-SDO-Objekte abbilden.

3.5.1 Einfache SDO-Dienste versus Domain-Dienste

Die einfachen Datentypen, die bis zu 4 Byte lang sind, können mit einem CANopen-Protokoll übertragen werden. Lediglich für Strings oder Felder sind Domain-Dienste erforderlich. Legt der OPC-Client beim Busserver ein Item mit dem Datentypen VT_BSTR oder VT_ARRAY an, so weiß der Busserver, daß zur Übertragung Domaindienste benötigt werden und benutzt diese beim Schreiben. Beim Lesen erkennt der Busserver auch an der Antwort des Gerätes, daß es sich um einen Domain-Dienst handelt.

3.5.2 Datentypen

Für die Typumwandlung zwischen den OPC-Datentypen und den CANopen-Datentypen gilt folgende Konvention:

| Variant-Typ | CANopen-Datentyp |
|-------------------|--|
| VT_I1 | INTEGER8 |
| VT_I2 | INTEGER16 |
| VT_I4 | INTEGER32 |
| VT_UI1 | UNSIGNED8 |
| VT_UI2 | UNSIGNED16 |
| VT_UI4 | UNSIGNED32 |
| VT_R4 | REAL32 |
| VT_R8 | REAL64 |
| VT_BSTR | VISIBLE_STRING |
| VT_ARRAY VT_UI1 | all types (raw format), OCTETT_STRING, ... |

Wenn ein Typ nicht direkt abgebildet werden kann, da z.B. der angeforderte Variant-Datentyp kürzer als der CANopen-Datentyp ist, so sollten die ersten Bytes des CANopen-Datentyps verwendet werden. Ist der Variant-Datentyp länger als der CANopen-Typ, so sind die nicht benutzten Bytes mit Null aufzufüllen.

Die Länge des Variant-Datentyps VT_ARRAY | VT_UI1 kann dynamisch angepaßt werden. Wird beim Anlegen eines OPC-Items mit diesem Datentyp keine Länge explizit angegeben, so wird diese aus dem Datenstrom des Feldbusses ermittelt und ein entsprechend großes Feld angelegt. Bei jedem Lese-/Schreibvorgang kann die Länge variieren.



3.6 Prozeßdaten

Für den Zugriff auf Prozeßdaten kann auf die Verwendung der PDO-Mappings verzichtet werden. Stattdessen erfolgt der Zugriff über die DriveServer-Syntax.

Beispiel (Prozeßdatenlänge 8 Byte):

INI1S2D3

xxxxxxx XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX xxxxxxxx xxxxxxxx xxxxxxxx

Byte 1 Byte 2 Byte 3 Byte 4 Byte 5 Byte 6 Byte 7 Byte 8

Eingangsdaten auf Prozeßdatenkanal 1, beginnend mit Byte 2, als VT_I4 darstellen.

3.6.1 Prozeßdatenlänge

Die Länge der Prozeßdaten ist nicht konstant 8 Byte, wie in obigem Beispiel.

Offene Fragen: Soll die tatsächliche Länge über die Konfigurations-SDO's ermittelt werden?

3.6.2 Sync-Message

Der Busserver kann bei Bedarf eine Sync-Message erzeugen. Der dafür verwendete Identifier und die Zykluszeit sind zu konfigurieren. Das Senden der Sync-Nachricht kann nicht über OPC gesteuert werden.

3.6.3 Offene Fragen

- Wie soll mit TPDOs umgegangen werden?
- Welche NMT-Dienste sollten vom Busserver unterstützt werden und eventuell über OPC-Items bedienbar sein?



4 Glossar

| | |
|------------------|---|
| Bus-Server | OPC-Server, der eine Kommunikation mit Geräten ermöglicht. Der Namensraum und die verfügbaren Items des Servers sind kommunikations-spezifisch |
| DriveServer | OPC-Server, der eine Kommunikation mit Geräten (Antrieben) ermöglicht. Der Namensraum und die verfügbaren Items des Servers sind gerätespezifisch |
| Namensraum | Übersicht über die dem OPC-Server bekannten Datenpunkte. Ein Namensraum kann hierarchisch strukturiert oder auch <code>flat</code> formatiert sein, d.h. alle verfügbaren Datenpunkte werden in einer Liste dargestellt |
| Index / Subindex | Adressierung der Parameter im Antriebsregler. Dieser Index ist verschieden vom DPV1-Index! |
| SDO | Service Data Object |
| PDO | Process Data Object |

5 Literatur

- /1/ OLE for Process Control, Data Access Custom Interface Standard, Version 2.0
<http://www.opcfoundation.org/>
- /2/ CANopen, Application Layer and Communication Profile, CiA Draft Standard 301,
Version 4.01
<http://www.can-cia.com/>
- /3/ Spezifikation DriveServer, Version 1.1
<http://www.drivecom.org/>