COPA-DATA
do it your way

zenon Treiber Handbuch

IEC850

v.7.10

© 2013 Ing. Punzenberger COPA-DATA GmbH

Alle Rechte vorbehalten.

Die Weitergabe und Vervielfältigung dieses Dokuments ist - gleich in welcher Art und Weise – nur mit schriftlicher Genehmigung der Firma COPA-DATA gestattet. Technische Daten dienen nur der Produktbeschreibung und sind keine zugesicherten Eigenschaften im Rechtssinn. Änderungen – auch in technischer Hinsicht - vorbehalten.

Inhalt

1. Willkommen bei der COPA-DATA Hilfe 5

2. IEC850 5

3. IEC850 - Datenblatt 6

4. Treiberhistorie 7

5. Voraussetzungen 10

5.1 PC 11

6. Konfiguration 11

6.1 Anlegen eines Treibers 11

6.2 Einstellungen im Treiberdialog 13

6.2.1 Allgemein 13

6.2.2 Treiberdialog Basic settings 16

6.2.3 Connections 19

7. Variablen anlegen 31

7.1 Variablen im Editor anlegen 31

7.2 Adressierung 34

7.3 Treiberobjekte und Datentypen 37

7.3.1 Treiberobjekte 38

7.3.2 Zuordnung von Datentypen 39

7.4 Variablen anlegen durch Import 42

7.4.1 Online-Import 42

7.4.2 Offline-Import 44

7.4.3 XML Import 45

7.4.4 DBF Import/Export 46

7.5 Treibervariablen 52

8. Treiberspezifische Funktionen 57

9. Reporting 67

9.1 Unbuffered Reporting 69

9.2 Buffered Reporting ... 71
9.3 Dynamische Datasets ... 73

10. Treiberkommandos ... 75

11. Fehleranalyse ... 77

11.1 Analysetool ... 77
11.2 Checkliste ... 78

12. Anhang A - Beschreibung der Functional Constraints (FCs): ... 80

13. Anhang B - Abkürzungen für data object/data attribute ... 81

14. Anhang C - Conformance statement ... 94

14.1 Protocol implementation conformance statement (PICS) ... 94
14.2 Model implementation conformance statement (MICS) ... 102

15. Anhang D - PIXIT for COPA-DATA IEC61850 Client ... 102

1. Willkommen bei der COPA-DATA Hilfe

ALLGEMEINE HILFE

Falls Sie in diesem Hilfekapitel Informationen vermissen oder Wünsche für Ergänzungen haben, wenden Sie sich bitte per E-Mail an documentation@copadata.com (mailto:documentation@copadata.com).

PROJEKTUNTERSTÜTZUNG

Unterstützung bei Fragen zu konkreten eigenen Projekten erhalten Sie vom Support-Team, das Sie per E-Mail an support@copadata.com (mailto:support@copadata.com) erreichen.

LIZENZEN UND MODULE

Sollten Sie feststellen, dass Sie weitere Module oder Lizenzen benötigen, sind unsere Mitarbeiter unter sales@copadata.com (mailto:sales@copadata.com) gerne für Sie da.

2. IEC850

Die Kommunikation zwischen dem Treiber und der SPS basiert auf dem IEC 61850 Protokoll mit Client/Server Diensten via TCP/IP (Profile A1/T1). Bei der Kommunikation verhält sich der Treiber wie ein Client (Master).

Achtung

Konfigurationen, die mit einem Treiber der Version 3400 oder höher geändert wurde, können mit älteren Treiber nicht mehr geöffnet werden.

3. IEC850 - Datenblatt

Allgemein:	
Treiberdateiname	IEC850.exe
Treiberbezeichnung	IEC61850
Steuerungs-Typen	IEC 61850 kompatible Steuerungen
Steuerungs-Hersteller	ABB; Siemens; IEC; SAT; Sprecher Automation; Kalki; GE Multilin;

Treiber unterstützt:	
Protokoll	IEC 61850;
Adressierung: Adress-basiert	-
Adressierung: Namens-basiert	x
Kommunikation spontan	x
Kommunikation pollend	x
Online Browsing	x
Offline Browsing	x
Echtzeitfähig	x

Blockwrite	-
Modemfähig	-
Serielles Logging	-
RDA numerisch	-
RDA String	-

Voraussetzungen:	
Hardware PC	Standard Netzwerkkarte
Software PC	XML-Lite wird benötigt, ist Bestandteil von Microsoft Internet Explorer 7.0; xmllite.dll, kann auch explizit von der Microsoft Homepage heruntergeladen werden.
Hardware Steuerung	-
Software Steuerung	-
Benötigt v-dll	-

Plattformen:	
Betriebssysteme	Windows CE 5.0, CE 6.0; Windows Vista, 7, 8, Server 2008/R2, Server 2012;
CE Plattformen	x86; ARM;

4. Treiberhistorie

Eine Liste der Hauptänderungen, die in der Treiberdokumentation vorgenommen wurden. Die Treiberversion (Treiber exe-Datei version) entspricht Prä-Compilations für Test und Dokumentationszwecke und nicht der freigegebenen Version des Treibers mit den beschriebenen

Funktionen.
Wenn Sie konkrete Frage zu Ihrem Projekt haben, kontaktieren Sie bitten das Consulting Team via E-Mail. support@copadata.com (mailto:support@copadata.com).

Datum	Treiberversion	Änderung
31.08.06	100	Treiber und Dokumentation erstellt
11.04.07	400	Unterstützung aller Modelle (SBO) des Control Model
04.10.07	700	Unbuffered Reporting
09.01.08	1100	Version für Windows CE
10.01.08	1100	File Transfer
28.01.08	1200	Buffered Reporting
04.02.08	1300	Auto Reset des ctlNum
29.05.08	1900	Alternative Verwendung der Kennung als externer Name
11.09.08	2300	Check condition by a command from Check attribute
12.12.08	2500	reedited driver documentation
23.12.08	2500	recognition of duplicated datasets in Reports; log messages for reported or polled variables
29.12.08	2500	configurable names of BRCBs
12.05.09	2900	bitstrings also as STRING variables
03.11. 09	3300	dynamic Data Sets option: Use preconfigured TrgOps
30.12.09	3600	option: Do not purge BRCB buffer at start
07.01.10	3600	Reports not only from LLN0
01.02.10	3700	off-line import also from *.cid and *.scd files
17.02.10	3700	Origin is sent to RT before all other attributes of a data object from Report
24.03.10	3800	Additional Cause Diagnosis
24.03.10	3900	possibility to split up Select and Operate for Command Processing module
24.03.10	3900	mapping of Time quality bits ClockFailure and ClockNotSynchronized
08.11.10	4000	Off-line Import: LNs embedded directly into IEDs without LDs are ignored
12.11.10	4100	support of data type “Check” (Tissue 10); PICS reviewed;

		PIXIT enclosed (KEMA Certification)
04.05.11	4200	UNICODE STRING in UTF-8 character encoding; consideration of all possible TimeAccuracy values; support of Authentication (ISO 8650-1)
06.05.11	4300	support of alternative IP address
19.05.11	4400	local time stamp for variables with FC not ST or MX
31.10.11	4500	individual TrgOps bits setting for Unbuffered Reports
02.11.11	4600	driver setting for Origin orCat
08.11.11	4700	individual TrgOps bits setting for Buffered Reports

Mit zenon 7.10 wurde die Versionierung der Treiber verändert. AB dieser Version gibt es eine versionsübergreifende Build-Nummer. Das ist die Nummer an der 4. Stelle der Dateiversion, z.B.: `7.10.0.4228` bedeutet, dass der Treiber für die Version 7.10 Servicepack 0 ist, und die Build Nummer `4228` besitzt. Erweiterungen oder Fehlerbehebungen werden zukünftig in einem Build eingebaut und sind dann ab der nächsthöheren Buildnummer verfügbar.

Beispiel

Eine Treibererweiterung wurde in Build ***4228*** *implementiert. Der Treiber, den Sie im Einsatz haben, verfügt über die Build-Nummer* ***8322****. Da die Build-Nummer Ihres Treibers höher ist als die Build-Nummer der Erweiterung, ist die Erweiterung enthalten. Die Versionsnummer des Treiber (die ersten drei Stellen der Dateiversion) spielen dabei keine Rolle. Die Treiber sind versionsunabsabhängig*

Datum	Build Nummer	Änderung
16.07.12	4230	support of ARRAY OF

5. Voraussetzungen

Dieses Kapitel enthält Informationen zu den Voraussetzungen, die für die Verwendung des Treibers erforderlich sind.

5.1 PC

XML Lite, das ein Bestandteil des Microsoft Internet Explorer 7.0 ist, ist erforderlich; xmllite.dll kann auch separat von der Microsoft Website heruntergeladen werden.

Dieser Treiber unterstützt eine Verbindung über die Standard Netzwerkkarte des PCs. Achten Sie darauf, dass die Steuerung und der PC im selben Netzwerkbereich liegen und die Subnet-Masken auf beiden Geräten dementsprechend konfiguriert sind.

6. Konfiguration

In diesem Kapitel lesen Sie, wie Sie den Treiber im Projekt anlegen und welche Einstellungen beim Treiber möglich sind.

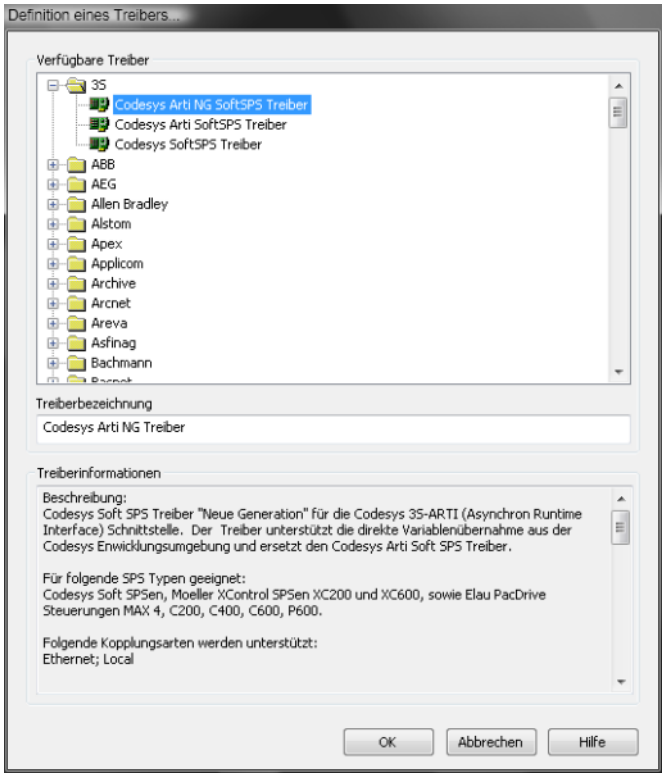
> **Info**
>
> *Weitere Einstellungen, die Sie für Variablen in zenon vornehmen können, finden Sie im Kapitel Variablen (main.chm::/15247.htm) der Online-Hilfe.*

6.1 Anlegen eines Treibers

Um einen neuen Treiber anzulegen:

1. Klicken Sie mit der rechten Maustaste im Projektmanager auf **`Treiber`** und selektieren Sie im Kontextmenü **`Treiber neu`**.

2. In der folgenden Dialogbox bietet Ihnen das Programm eine Auflistung aller verfügbaren Treiber an.



3. Selektieren Sie den gewünschten Treiber und vergeben Sie eine Bezeichnung für diesen:
   - Die Treiberbezeichnung muss eindeutig sein, d.h. wird ein und derselbe Treiber mehrmals im Projekt verwendet, so muss jeweils eine neue Bezeichnung vergeben werden.
   - Die Treiberbezeichnung ist Bestandteil des Dateinamens. Daher darf Sie nur Zeichen enthalten, die vom Betriebssystem unterstützt werden. Nicht gültige Zeichen werden durch einen Unterstrich (_) ersetzt.
   - Achtung: Die Bezeichnung kann später nicht mehr geändert werden.
4. Bestätigen Sie den Dialog mit **ok**. Im folgenden Dialog werden die einzelnen Konfigurationen der jeweiligen Treiber eingestellt.

Für ein Projekt müssen nur die jeweils notwendigen Treiber eingebunden werden. Späteres Einbinden eines weiteren Treibers ist problemlos möglich.

Info

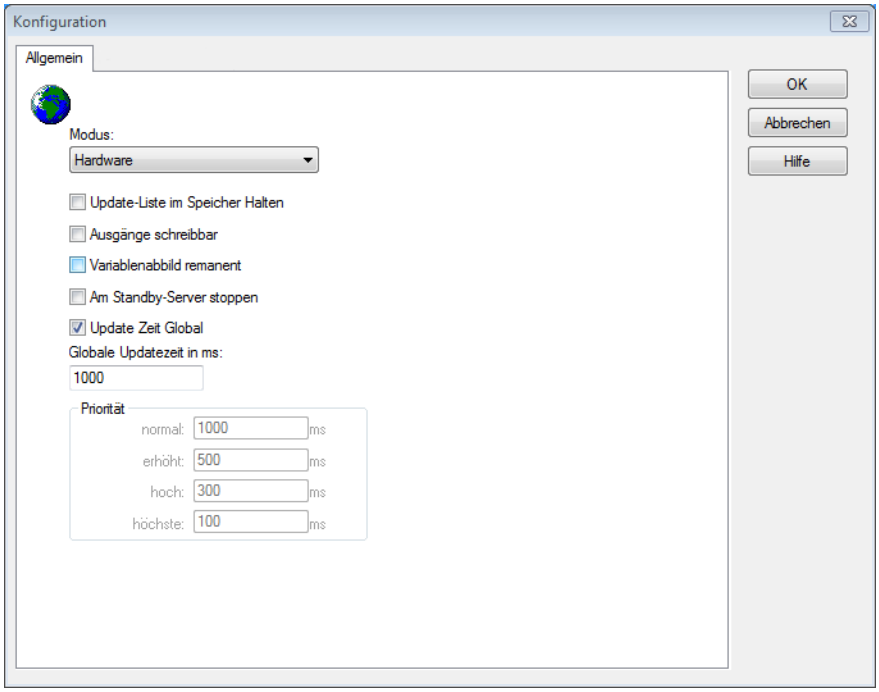
Bei neu angelegten Projekten oder bei bestehenden Projekten, die in eine Version ab 6.21 konvertiert werden, werden die folgenden Treiber automatisch angelegt:

- Intern
- MathDr32
- SysDrv.

6.2 Einstellungen im Treiberdialog

Folgende Einstellungen können Sie beim Treiber vornehmen:

6.2.1 Allgemein



Parameter	Beschreibung
Modus	Ermöglicht ein Umschalten zwischen Hardware und Simulationsmodus ▸ Hardware: Die Verbindung zur Steuerung wird hergestellt. ▸ Simulation - statisch Es wird keine Kommunikation zur Steuerung aufgebaut, die Werte werden vom Treiber simuliert. In diesem Modus bleiben die Werte konstant bzw. die Variablen behalten die über zenon Logic gesetzen Werte. Jede Variable hat seinen eigenen Speicherbereich. Zum Beispiel zwei Variablen vom Typ Merker mit Offset 79, können zur Runtime unterschiedliche Werte haben und beeinflussen sich gegenseitig nicht. Ausnahme: Der Simulatortreiber. ▸ Simulation - zählend Es wird keine Kommunikation zur Steuerung aufgebaut, die Werte werden vom Treiber simuliert. In diesem Modus zählt der Treiber die Werte innerhalb ihres Wertebereichs automatisch hoch. ▸ Simulation - programmiert Es wird keine Kommunikation zur Steuerung aufgebaut, die Werte werden von einem frei programmierbaren Simulationsprojekt berechnet. Das Simulationsprojekt wird mit der zenon Logic Workbench erstellt und läuft in einer in den Treiber integrierten zenon Logic Runtime ab. Details siehe Kapitel Treibersimulation. (main.chm::/25206.htm)
Update-Liste im Speicher Halten	Einmal angeforderte Variablen werden weiterhin von der Steuerung angefordert, auch wenn diese aktuell nicht mehr benötigt werden. Dies hat den Vorteil, dass z B. mehrmalige Bildumschaltungen nach dem erstmaligen Aufschalten beschleunigt werden, da die Variablen nicht neu angefordert werden müssen. Der Nachteil ist eine erhöhte Belastung der Kommunikation zur Steuerung.
Ausgänge schreibbar	Aktiv: Ausgänge können beschrieben werden. Inaktiv: Das Beschreiben der Ausgänge wird unterbunden.

	Hinweis: Steht nicht für jeden Treiber zur Verfügungen.
Variablenabbild remanent	Diese Option speichert und restauriert den aktuellen Wert, den Zeitstempel und die Status eines Datenpunkts. Grundvoraussetzung: Die Variable muss einen gültigen Wert und Zeitstempel besitzen. Das Variablenabbild wird im Modus Hardware gespeichert wenn: ▸ einer der Status S_MERKER_1(0) bis S_MERKER8(7), REVISION(9), AUS(20) oder ERSATZWERT(27) aktiv ist Das Variablenabbild wird immer gespeichert wenn: ▸ die Variable vom Objekttyp Treibervariable ist ▸ der Treiber im Simulationsmodus läuft. (nicht programmierte Simulation) Folgende Status werden beim Start der Runtime nicht restauriert: ▸ SELECT(8) ▸ WR-ACK(40) ▸ WR-SUC(41) Der Modus **Simulation - programmiert** beim Treiberstart ist kein Kriterium, um das remanente Variablenabbild zu restaurieren.
Am Standby-Server stoppen	Einstellung für Redundanz bei Treibern, die nur eine Kommunikationsverbindung erlauben. Dazu wird der Treiber am Standby-Server gestoppt und erst beim Hochstufen wieder gestartet. Achtung: Ist diese Option aktiv, ist die lückenlose Archivierung nicht mehr gewährleistet. Aktiv: Versetzt den Treiber am nicht-prozessführenden Server automatisch in einen Stopp-ähnlichen Zustand. Im Unterschied zum Stoppen über Treiberkommando erhält die Variable nicht den Status **abgeschaltet** (**statusverarbeitung.chm::/24150.htm**), sondern einen leeren Wert. Damit wird verhindert, dass beim Hochstufen zum Server nicht relevante Werte in AML , CEL und Archiv erzeugt werden.
Update Zeit Global	Aktiv: Die eingestellte Globale Update Zeit in ms wird für alle Variablen im Projekt verwendet. Die bei den Variablen eingestellte Priorität

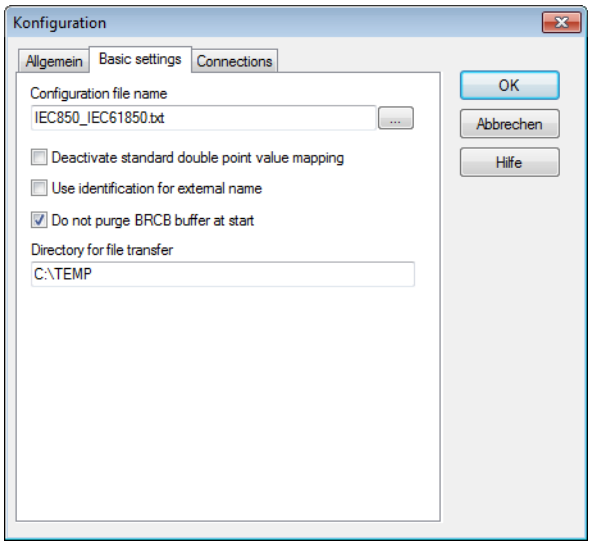
	wird nicht verwendet. `Inaktiv`: Die eingestellten Prioritäten werden für die einzelnen Variablen verwendet.
`Priorität`	Hier werden die Pollingzeiten der einzelnen Prioritäten eingestellt. Alle Variablen mit der entsprechenden Priorität werden in der eingestellten Zeit gepollt. Die Zuordnung zu den Variablen erfolgt separat bei jeder Variablen über die Einstellungen in den Variableneigenschaften. Mit den Prioritäten kann die Kommunikation der einzelnen Variablen auf die Wichtigkeit bzw. benötigte Aktualität abgestuft werden. Daraus ergibt sich eine verbesserte Verteilung der Kommunikationslast.
`OK`	Übernimmt Einstellungen aus allen Registerkarten und schließt den Dialog.
`Abbrechen`	Verwirft alle Änderungen und schließt den Dialog.
`Hilfe`	Öffnet die Online-Hilfe.

UPDATE ZEIT ZYKLISCHE TREIBER

Für zyklische Treiber gilt:

Beim **`Sollwert Setzen`**, **`Advicen`** von Variablen und bei **`Requests`** wird sofort ein Lesezyklus für alle Treiber ausgelöst - unabhängig von der eingestellten Update Zeit. Damit wird sicher gestellt, dass der Wert nach dem Schreiben in der Visualisierung sofort zur Verfügung steht. Update-Zeiten können damit für zyklische Treiber kürzer ausfallen als eingestellt.

6.2.2 Treiberdialog Basic settings



Parameter	Beschreibung
Configuration file name	Name der Datei, in der die treiberspezifischen Konfigurationen gespeichert sind. Klick auf die Schaltfläche ... öffnet Dialog zur Auswahl eines Ablageortes.
Deactivate standard double point value mapping	Inactive : Die Werte von double point values werden den Betriebselementen von zenon angepasst. Benutzen Sie diese Konfiguration, wenn Sie die Module von zenon Energy verwenden möchten. Details finden sie in der **Tabelle Double Point Values**. Aktiv: Die Werte der double point values werden so wie sie sind an zenon weitergeleitet. In diesem Fall können Sie z. B. die Befehlsgabe-Funktionen von zenon Energy nicht verwenden. Default: Inactive
Use identification for external name:	Inactive : Der Variablenname wird für die Adressierung verwendet. Active: Der Treiber verwendet für die Adressierung die Einträge der Variableneigenschaft Kennung. Default: Inactive
Do not purge BRCB buffer at start	Active: Die Puffer der BRCBs werden bei der ersten Verbindung nach dem Start der Runtime nicht gelöscht. Voraussetzungen: Für diesen BRCB wurde die zuletzt empfangene Entry-ID gespeichert.
Originator category (orCat)	Der Treiber sendet konfigurierte orCat Werte als Kommandos. Mögliche Einstellungen: ▸ 1 - Bay-control ▸ 2 - Station-control ▸ 3 - Remote-control ▸ 4 - Automatic-bay

	▸ 5 - Automatic-station ▸ 6- Automatic-remote ▸ 7 - Maintenance Default: 2- Station-control.
Directory for file transfer	Zielordner des Dateitransfers. In diesem Ordner werden alle hinaufgeladenen Dateien gespeichert.

TABELLE DOUBLE POINT VALUES

Für die Einstellung Inactive von **Deactivate standard double point value mapping** gilt:
Der Treiber bildet die Double Point Values für die Runtime entsprechend der Spalte zenon ab (stVal : CODED ENUM in DPS und DPC Classes, IEC 61850-7-3).

Position	Positionswert decimal	Positionswert binär-string	Übertragener Dbpos Wert (ungemappt)	zenon (gemappt)
intermediate state	0	00	0x00	2
off	1	01	0x40	0
on	2	10	0x80	1
bad state	3	11	0xC0	3

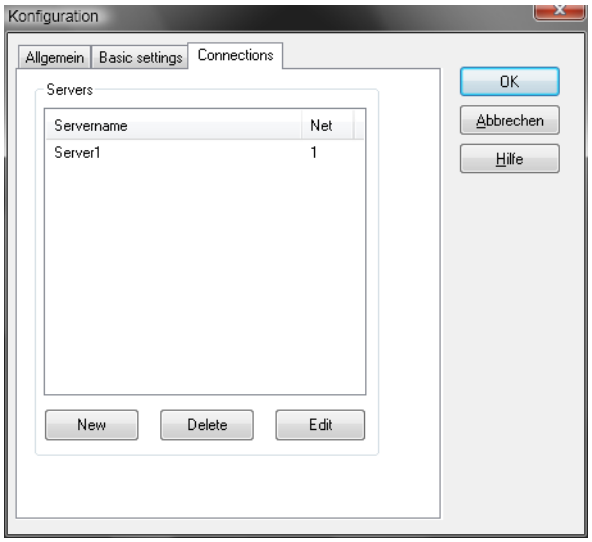
Zum Beispiel bedeutet in der Runtime:

Parameter	gemappt	unegmappt
off	0	64
on	1	128

Mehr Information finden Sie im Kapitel: Treiberspezifische Funktionen (auf Seite 57) -> **Mapping**.

6.2.3 Connections

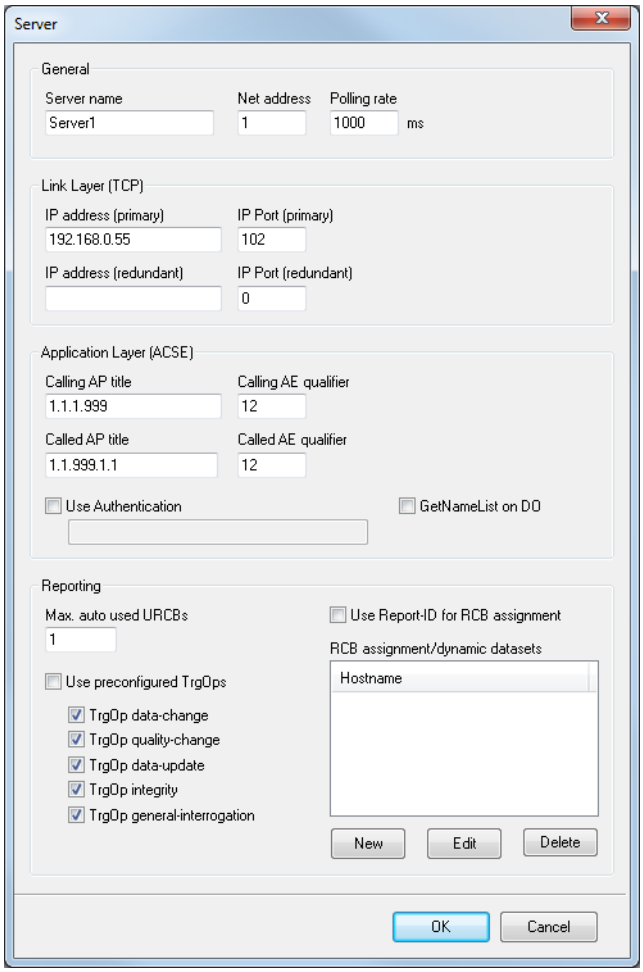
Die Verbindungen des IEC 850 Treibers werden in der Registerkarte Connections des Treiberdialogs eingestellt.



Parameter	Beschreibung
Server	Liste der bereits konfigurierten Verbindungen des IEC 61850 Servers (zu den Control Objects).
New	Erstellt eine neue Verbindung zu einem IEC 61850 Server. Öffnet den Dialog 'Server'.
Delete	Entfernt die ausgewählte Verbindung aus der Liste.
Edit	Öffnet den Dialog 'Server', um die ausgewählte Verbindung zu editieren

Server

Ein Klick auf **New** oder **Edit** im Dialog Connection (auf Seite 19) öffnet den Dialog für die Konfiguration einer Verbindung zu einem IEC 61850 Server:



Parameter	Beschreibung
`Net address`	Entspricht der Eigenschaft Netzadresse bei der Variablenkonfiguration. Maximalwert: `65535`
`Server name`	Frei wählbarer Name. Wird bei Variablenimport für den Namen der Variablen verwendet.
`Polling rate`	Definiert die Aktualisierungsfrequenz in Millisekunden. Der Treiber benützt diese Frequenz, um Daten, die nicht in aktiven `Reports` vorhanden sind, zu pollen. Dadurch wird auch ein eventueller Ausfall des IEC61850 Servers erkannt. `Maximalwert: 4294967295` **(2^32-1)**

	Wird ein höherer Wert eingegeben, wird er beim Speichern automatisch auf 1 geändert. Default: 1000 Hinweis: Wenigstens eine Variable soll immer gepollt werden auch wenn alle Daten des Projekts im Reports vorhanden sind. Sonst kann der Ausfall des IEC61850 Servers nicht erkannt werden, wenn die TCP/IP-Verbindung noch besteht. Legen Sie dafür z.B. eine ***LLN0/Mod/cltModel[CF]** Variable an und sorgen Sie dafür, dass die Variable beim Treiber immer angemeldet ist. Achtung: Der Datenaustausch mit der Runtime hängt weder von dieser Einstellung noch von der globalen Updatezeit ab. Die ist fest auf 100 ms eingestellt.
Link Layer (TCP)	Verbindungseinstellungen
IP adress (primary)	IP-Adresse des Servers, mit dem eine Verbindung hergestellt werden soll.
IP Port (primary)	Port, über den die Verbindung hergestellt werden soll. Default: 102
IP adress (redundant)	Alternative IP-Adresse. Schlägt der Verbindungsversuch zur ersten IP Adresse fehl, so wird nach der Netz-Fehlerwartezeit (je nach Netzwerk 20–30 s) umgeschaltet auf die alternative IP Adresse. Diese wird beibehalten bis entweder der Treiber neu gestartet wird (über Treiberfunktionen oder Neustart der 7.10 Runtime) oder die Verbindung fehlschlägt. Dann wird wieder versucht, mit der ersten IP-Adresse zu verbinden. Achtung: Um einen Verbindungsausfall zu erkennen, sollte wenigstens eine Variable, z. B. ***LLN0/Mod/stVal[ST]**, immer gepollt werden.
IP Port (redundant)	Alternativer IP-Port.
Application Layer (ACSE)	Einstellungen für ACSE.
Calling AP title	Einstellungen nach ISO 8650-1(ACSE), der Wert für OSI ACSE AP Title des Clients in der Universal Identifier Notierung. Soll im Standardfall nicht geändert werden.
Calling AE qualifier	Einstellungen nach ISO 8650-1(ACSE). Soll im Standardfall nicht geändert werden.

Called AP title	Einstellungen nach ISO 8650-1(ACSE), OSI-AP-Title des Servers im Format für Universal Identifier. Soll im Standardfall nicht geändert werden.
Called AE qualifier	Einstellungen nach ISO 8650-1(ACSE). Soll im Standardfall nicht geändert werden.
Use Authentication	Aktivieren Sie diese Checkbox, wenn Sie die Authentifikation nach ISO 8650-1 benutzen möchten. Aktiv: Der Treiber sendet beim Verbindungsaufbau den Authentication String mit.
Authentication String	Wenn Use Authentication aktiv ist, können Sie hier Ihren Authentication String eintragen. Beachten Sie die Längenbeschränkung auf 55 Zeichen laut ISO 8327-1 OSI Session Protocol.
GetNameList on DO	Aktiv: Der Treiber liest das Objekt Model mittels Abfrage von Data Objekten (DO) für jeden im Server vorhandenen Logical Node pro im IEC61850 Standard definiertem Functional Constraint (FC). Damit können Objekt Models auch dann ausgelesen werden, wenn manche Logical Nodes größer als die vom Server unterstützte PDU-Größe sind. Achtung: Ist diese Option aktiv, kann der Verbindungsaufbau um ein Vielfaches länger dauern. Aktivieren Sie diese Option nur, wenn das Einlesen aufgrund von Server-Einschränkungen nicht vollständig durchgeführt werden kann. Default: inaktiv
Reporting	Einstellung für die Reports (auf Seite 67).
Max. auto used URCBs	Maximal Anzahl an Unbuffered Reports, die der Treiber bei automatische Zuweisung aktiviert, also zusätzlich zu RCBs, die im Dialog **RCB assignment/dynamic datasets** konfiguriert wurden. Default: 1 Eintrag ist nur für Unbuffered Reports gültig. Mehr Information zur automatischen Zuwesiung finden Sie im Kapitel

	Unbuffered Report (auf Seite 69).
Use preconfigured TrgOps	Aktiv: die Trigger Optionen, die im IEC61850 Server bereits vorkonfiguriert sind, werden benutzt. In diesem Fall schreibt der Treiber das TrgOps Attribut nicht in den Report Control Block bei der Zuweisung. Inaktiv: der Treiber schreibt das TrgOps Attribut während der Aktivierung jedes RCB. Sie können die folgenden Trigger Optionen getrennt aktivieren/deaktivieren: ▸ TrgOp: data-change ▸ TrgOp: quality-change ▸ TrgOp: data-update ▸ TrgOp: integrity ▸ TrgOp: general-interrogation Default: alle aktiviert Hinweis: Nicht alle Server unterstützen TrgOps data-change und data-update zusammen. Auch TrgOp intergity kann zu unnötiger Überlastung der Kommunikation führen wenn für RCB eine zu kurze IntgPd (Integrity Period) im Server definiert wurde. Im Zweifelsfall setzten Sie TrgOps: data-change + quality-change + general-interrogation.
Use Report-ID for RCB assignment	Aktiv: Die Report ID wird anstatt des Reportnamens im Dialog für statische RCB Zuweisung in der **Client configuration** verwendet. Inaktiv: Die RCB Instanzen des Servers werden per Name identifiziert. Default: inaktiv Achtung: Wenn diese Option später geändert wird, müssen alle RCB Zuweisungen in der **Client configuration** gelöscht und wieder erstellt werden.

RCB assignment/dynamic datasets	Statische Report Control Block Zuweisungen und Einstellungen für Dynamic Data Sets . Hinweis: Wenn Sie in Ihrem zenon Projekt Redundanz verwenden (Eigenschaft Netzwerk aktiv aktiviert sowie Server und Standby konfiguriert), müssen Sie hier zwei Einträge erstellen: einen für den Server und einen für den Standby.
New	Erzeugt einen neuen Eintrag in der Liste. Öffnet den Dialog **Client configuraton.**
Del	Entfernt den ausgewählte Eintrag aus der Liste.
Edit	Öffnet den Dialog **Client configuration**, um den ausgewählten Eintrag zu bearbeiten.

Info

OSI-Werte (ACSE)

Der Treiber benutzt folgende festen OSI Selector Werte:

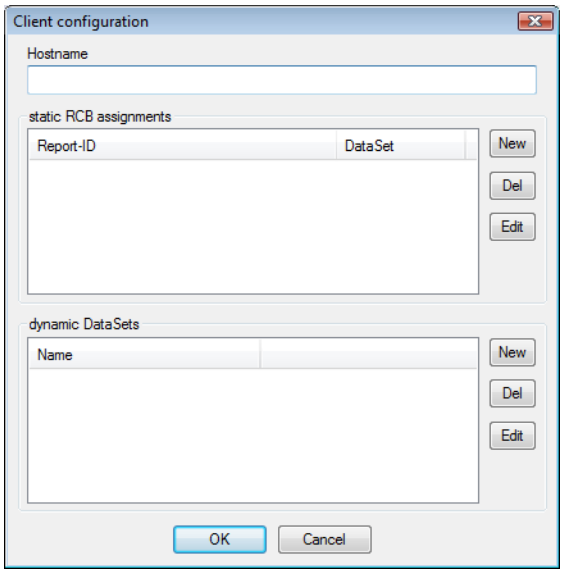
- Presentation Selector (OSI-PSEL) = "00000001"
- Session Selector (OSI-SSEL) = "0001"
- Transport Selector (OSI-TSEL) = "0001"

In einer Server SCL-Datei entspricht es folgenden Einträgen:

- <P type="OSI-AP-Title">41,999,1,1</P>
  - kodiert nach Basic Encoding Rules (IEC 8825) für Object Identifier - entspricht Standardwerten des Universal Identifiers in der `Called AP title`.
- <P type="OSI-AE-Qualifier">12</P>
  - entspricht Standardwerten in der `Called AE qualifier`.
- <P type="OSI-PSEL">00000001</P>
  - fest eingestellt im Treiber
- <P type="OSI-SSEL">0001</P>
  - fest eingestellt im Treiber
- <P type="OSI-TSEL">0001</P>
  - fest eingestellt im Treiber

Client Konfiguration

Ein Klick auf `New` oder `Edit` in der `RCB assignments/dynamic datasets` Liste des Server dialogs (auf Seite 20) öffnet den Dialog für die Einstellungen der Client-Konfiguration:

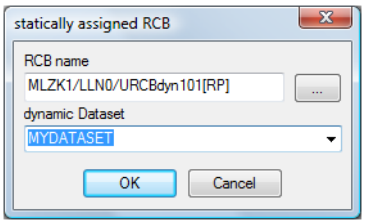


Parameter	Beschreibung
Hostname	Name des Computers auf dem der Treiber läuft, der die Reports empfängt: ▸ für standalone Projekte: der Computername des Computers auf dem die zenon Runtime läuft ▸ im zenon Netzwerk: der Computername des Projekt-Server oder Projekt-Standby Achtung: Benutzen Sie nicht "localhost" sondern den Computernamen aus dem Betriebssystem.
static RCB assignment	Einstellungen für die statische Zuweisung von RCBs (gleich für Unbuffered und Buffered RCBs). Hinweis: Buffered Reports müssen in der Treiberkonfiguration konfiguriert werden. Wenn sie nicht konfiguriert sind, werden sie nicht verwendet. Für mehr Information siehe treiberspezifische Funktionen (auf Seite 57).
Report-ID	Zeigt die bereits eingegebenen Report-IDs.
DataSet	Zeigt das dynamic Data Set (wenn der RCB eins zugewiesen wurde).
New	Erzeugt einen neuen Eintrag in der Liste. Öffnet den Dialog 'statically assigned RCB', um einen neuen Eintrag mit einer Report-ID (oder einem Name) anzulegen - Sie können entweder manuell eingeben oder den Server browsen und wenn nötig mit Dynamic Data Set (wird aus der unteren Liste genommen).
Del	Entfernt den ausgewählte Eintrag aus der Liste.
Edit	Öffnet den Dialog 'statically assigned RCB', um den ausgewählten Eintrag zu bearbeiten.
dynamic DataSets	Einstellungen für Dynamic Data Sets. Hinweis: Dynamische Datasets sind keine Voraussetzung für Buffered oder Unbuffered Reports. Es sind optionale Einstellungen, wenn der IEC61850 Server, der verwendet wird, dies unterstützt. Für mehr Information siehe Dynamische Datasets (auf Seite 73).

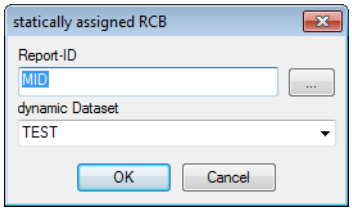
Name	Zeigt die bereits eingegebenen Data Sets - die sind in der oberen Liste vorhanden.
New	Öffnet den Dialog zur Erstellung eines Datasets.
Del	Entfernt den ausgewählte Eintrag aus der Liste.
Edit	Öffnet den Dialog 'dynamic DataSet configuration', um den ausgewählten Eintrag zu bearbeiten.

statically assigned RCB

Ein Klick auf New oder Edit in der **static RCB assignment** Liste des **Client configuration** (auf Seite 25) Dialogs öffnet den Dialog, um ein RCB zu erstellen und/oder einem RCB ein Dynamic Data Set zuzuweisen. RCBs sind entweder über den RCB-Namen definiert:

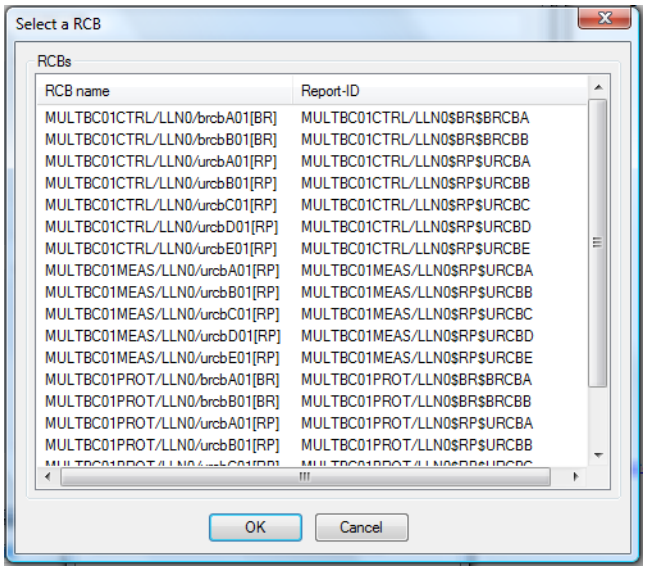


oder über die Report-ID:



Parameter	Beschreibung
RCB name/Report-ID	Der Name oder die ID des Reports (buffered oder unbuffered). Muss genau dem Namen/ReportID in der Steuerung (IEC61850 Server) entsprechen. Ein Klick auf . . . startet das online Browsing vom konfigurierten IEC 61850 Server und zeigt, wenn der Server läuft, alle Reporting Control Blocks, die zurzeit auf dem Server vorhanden sind. Hinweis: Per Default wird hier RCB name erwartet. Die Benuztzung des RCB name kann mithilfe der Eigenschaft Use Report-ID for RCB assignment in den **Server** (auf Seite 20) Eigenschaften auf Report-ID geändert werden. Report-ID: Benutzen Sie die Report-ID, die im Server konfiguriert wurde. Abhängig von der Steuerung ist die ID vielleicht schon vordefiniert oder sie kann vom Benutzer selbst konfiguriert werden z. B. über eine SCL-Datei. Hinweis: Wenn die Runtime zur selben Zeit auf dem selben Computer läuft, könne die RCBs nicht online gebrowst werden.
dynamic DataSet	Optional können Sie ein Data Set aus der Dropdownliste der bereits im Dialog 'dynamic DataSet configuration (auf Seite 29)' eingegebenen Data Sets auswählen. Wichtig: Wenn Sie hier keine Data Set definieren, wird das Data Set, das für diese RCB in der Steuerung (IEC61850 Server) vordefiniert ist, verwendet.

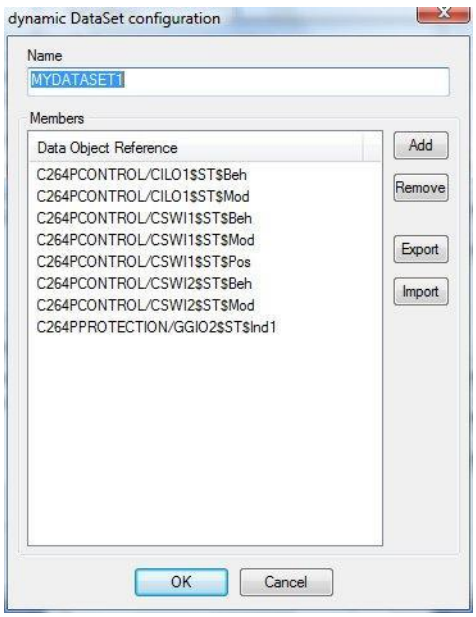
Nachdem Sie RCBs online auf dem IEC 61850 Server gebrowst haben, wird diese Liste angezeigt:



Wählen Sie den gewünschten Reporting Control Block aus der Liste.

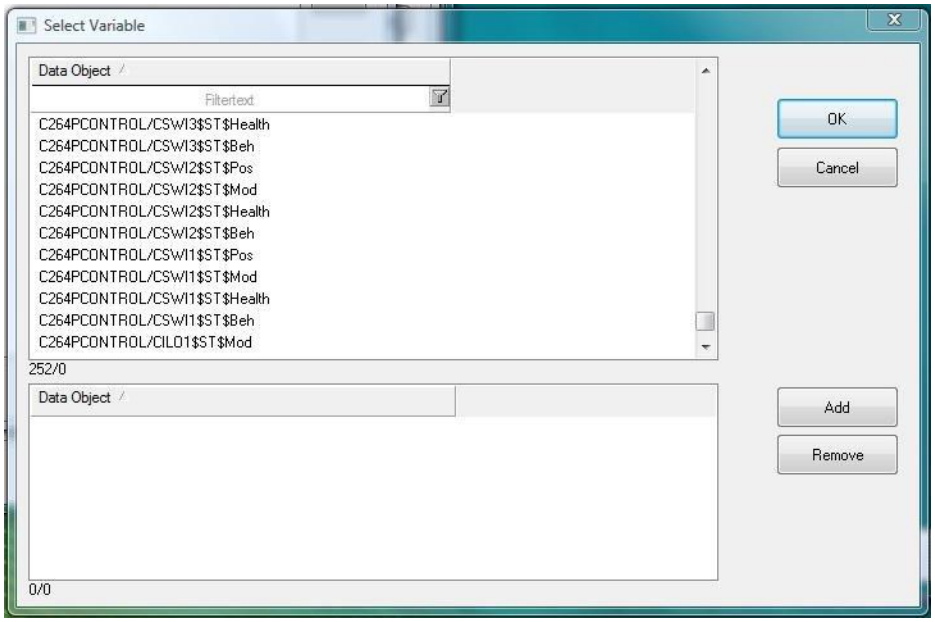
dynamic DataSet configuration

Ein Klick auf `New` oder `Edit` in der **`dynamic DataSets`** Liste des **`Client configuration`** (auf Seite 25) Dialogs öffnet den Dialog, um die Erstellung eines dynamischen Data Set zu definieren.



Parameter	Beschreibung
Name	Name des Data Set.
Members	Liste der bereits zugewiesenen Datenobjekte.
Add	Startet das Browsen von Data Objects aus der Variablenliste des zenon Projekts und zeigt alle Datenobjekte für die im Projekt eine Variable bereits erstellt wurde. Hinweis: Der Treiber browst die Datenobjekte nicht vom IEC61850 Server, aber aus der Variablenliste im Editor. Wenn aber die Runtime zur selben Zeit auf dem selben Computer läuft, könne die Variablen nicht gebrowst werden.
Remove	Entfernt den ausgewählte Eintrag aus der Liste.
Export	Exportiert Member in eine TXT-Datei.
Import	Importiert Member in eine TXT-Datei. im offenen Dialog: ▸ wählen Sie die entsprechende Datei aus ▸ wählen Sie die gewünschten Variablen aus ▸ bestätigen Sie die Einstellungen durch einen Klick auf OK

Nachdem Sie passende Projektvariablen gebrowst haben, wird Ihnen die Liste angezeigt:



Der Treiber zeigt alle - für das Reporting passenden - Datenobjekte aus der Variablenliste des zenon Projekts außer denen, die schon für das Data Set ausgewählt wurden. In der oberen Liste können Sie die gebrowsten Variablen filtern, selektieren (Mehrfachauswahl ist auch möglich) Sie sie und fügen Sie sie der unteren Liste hinzu. Bereits ausgewählte Objekte bleiben erhalten. Mithilfe der Schaltfläche **OK** fügen Sie die ausgewählte Variablen zu den **Members** des dynamischen Data Sets hinzu.

7. Variablen anlegen

So werden Variablen im zenon Editor angelegt:

7.1 Variablen im Editor anlegen

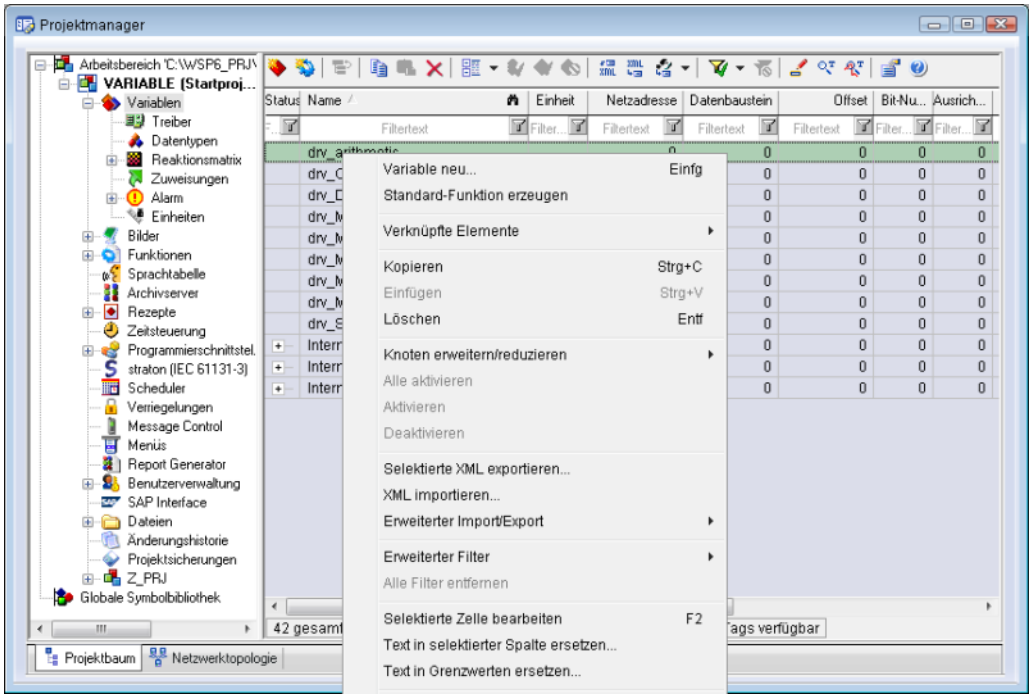
Variablen können angelegt werden:

- als einfache Variable
- in Arrays (main.chm::/15262.htm)
- als Struktur-Variablen (main.chm::/15278.htm)

DIALOG VARIABLE

Um eine neue Variable zu erstellen, gleich welchen Typs:

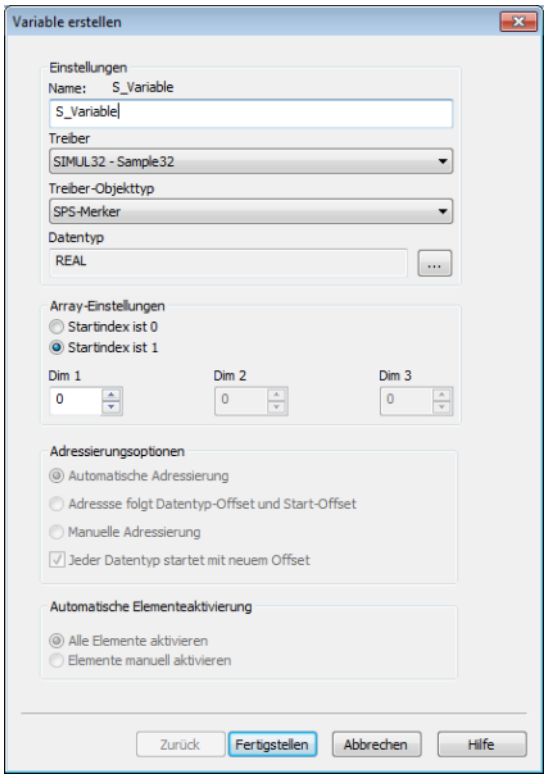
1. wählen Sie im Knoten **`Variablen`** im Kontextmenü den Befehl **`Variable neu`**



2. der Dialog zur Konfiguration der Variable wird geöffnet

3. konfigurieren Sie die Variable

4. welche Einstellungen möglich sind, hängt ab vom Typ der Variablen



Eigenschaft	Beschreibung
Name	Eindeutiger Name der Variablen. Ist eine Variable mit gleichem Namen im Projekt bereits vorhanden, kann keine weitere Variable mit diesem Namen angelegt werden. Maximale Länge: 128 Zeichen Achtung: Das Zeichen **#** ist im Variablennamen nicht zugelassen. Bei Verwendung nicht zugelassener Zeichen kann die Variablenerstellung nicht abgeschlossen werden, die Schaltfläche **Fertigstellen** bleibt inaktiv.
Treiber	Wählen Sie aus der Combobox den gewünschten Treiber. Hinweis: Sollte im Projekt noch kein Treiber angelegt sein, wird automatisch der Treiber für interne Variable (Intern.exe (Main.chm::/Intern.chm::/Intern.htm)) geladen.
Treiber-Objekttyp (cti.chm::/28685.htm)	Wählen Sie aus der Dropdownliste den passenden Treiber-Objekttyp aus.

Datentyp	Wählen Sie den gewünschten Datentyp. Klick auf die Schaltfläche ... öffnet den Auswahl-Dialog.
Array-Einstellungen	Erweiterte Einstellungen für Array-Variablen. Details dazu lesen Sie im Abschnitt Arrays.
Adressierungsoptionen	Erweiterte Einstellungen für Arrays und Struktur-Variablen. Details dazu lesen Sie im jeweiligen Abschnitt.
Automatische Elementeaktivierung	Erweiterte Einstellungen für Arrays und Struktur-Variablen. Details dazu lesen Sie im jeweiligen Abschnitt.

ABLEITUNG VOM DATENTYP

Messbereich, Signalbereich und Sollwert Setzen werden immer:

- vom Datentyp abgeleitet
- beim Ändern des Datentyps automatisch angepasst

Hinweis Signalbereich: Bei einem Wechsel auf einen Datentyp, der den eingestellten Signalbereich nicht unterstützt, wird der Signalbereich automatisch angepasst. Zum Beispiel wird bei einem Wechsel von **INT** auf **SINT** der Signalbereich auf 127 geändert. Die Anpassung erfolgt auch dann, wenn der Signalbereich nicht vom Datentyp abgeleitet wurde. In diesem Fall muss der Messbereich manuell angepasst werden.

7.2 Adressierung

Die Adressierung basiert auf den Variablennamen.

Name	Beschreibung
Name	Frei wählbarer Name Achtung: Der Name wird zur Adressierung der Variablen benutzt und muss in jedem Steuersystemprojekt eindeutig sein. Dadurch wird auch der Servername berücksichtigt, auch wenn der Treiber die Verbindung mithilfe der Eigenschaft Net address identifiziert.
Kennung	Geben Sie einen beliebigen Text ein; z. B. Ressourcenbezeichnung, Kommentar, ... Kann für die Adressierung verwendet werden (optional).
Net address	Netzadresse der Variable. Diese Adresse bezieht sich auf die Netzadresse in der Verbindungsliste der Treiberkonfiguration. Sie definiert den IEC 61850 Server auf dem die Variablen abgelegt sind.
Datenbaustein	für diesen Treiber nicht verwendet
Offset	für diesen Treiber nicht verwendet
Ausrichtung	für diesen Treiber nicht verwendet
Bitnummer	für diesen Treiber nicht verwendet
Stringlänge	Nur für String-Variablen verfügbar: Maximale Anzahl an Zeichen, aus der eine Variable bestehen kann.
Treiberobjekttyp	Abhängig von der Art der Variablen wird der Objekttyp während der Erstellung der Variablen selektiert; Der Typ kann hier später geändert werden. Für mehr Informationen siehe auch Treiberobjekte (auf Seite 38).
Datentyp	Datentyp der Variablen, die während der Erstellung der Variablen selektiert wurde; Der Typ kann hier später geändert werden. ACHTUNG: Wenn Sie den Datentyp später ändern, müssen Sie alle Eigenschaften der Variablen überprüfen und gegebenenfalls anpassen; z. B. Wertebereich.

SYNTAX NAMENSKONVENTION

Die Namenskonvention baut auf der folgenden Syntax auf: **SERVER!LD/LN/DATA/DataAttr[FC]**

Dabei gilt:

- Der Servername (auf Seite 20) wird durch ein Rufzeichen (**!**) von den folgenden Bezeichnern getrennt.

- Alle weiteren Bezeichner bis **DataAttr** werden durch einen Slash (/) voneinander getrennt.
- Nur wenn ein Bezeichner eine Struktur ist, werden deren Elemente durch einen Punkt (.) getrennt, zum Beispiel: **SERVER!LD/LN/DATA/DataAttr.item[FC]**.

Die Bezeichnungen sind:

Parameter	Beschreibung
SERVER	Frei definierbarer Name des Servers. Name soll sicher stellen, dass alle Variablennamen eindeutig sind. Empfehlung: Benutzen Sie die gleichen Namen, wie im Treiberdialog Dialog Server (auf Seite 20) definiert. Der Treiber erkennt allerdings die zu verwendende Verbindung nicht über den Server name, sondern über die Einstellung der Net address.
LD	Name des logischen Geräts.
LN	Name des logischen Knoten.
DATA	Name des 'Datenobjekts', siehe Anhang B - Datenobjekte/Datenattribute
DataAttr	Name des 'Datenattributs', siehe Anhang B - Datenobjekte/Datenattribute
FC	'Functional Constraint' des 'Datenattributes', siehe Anhang A - Beschreibung der Functional Constraints (FCs)

Achtung

*Die Namenskonvention für Variablen entspricht nicht der im IEC61850-7-1 Standard definierten **ObjectReference**.*

*Im Standard wird ein Slash (/) nur zwischen **LD** und **LD** als Seperator benutzt und ansonsten mit Punktern (.) getrennt. Wenn **DATA** oder **DataAttr** eine Struktur sind, machen es Punkte allerdings unmöglich zu unterscheiden, was **DATA** und was eine Struktur ist. Daher wird für die Variablenadressierung eine eigene Namenskonvention benutzt.*

BEISPIEL:

Der logische Knoten 'XCBR1' (Schutzschalter 1) hat unter anderem ein Datenobjekt 'Pos' welches ein CDC (Common Data Class) 'Controllable Double Point' (DPC). Dieses Datenobjekt 'Pos' wiederrum hat

Attribute 'stVal', 'q' und 't', deren Name und Semantik durch ihr CDC spezifisch, das in IEC61850-7-3 definiert ist.

Datenattribute	Name	IEC 61850 Standard
XCBR1/Pos	Schaltposition	gemäß LN Definition XCBR in 61850-7-4
XCBR1/Pos/stVal[ST]	Positionswert	gemäß CDC Definition DPC in 61850-7-3
XCBR1/Pos/q[ST]	Quality	gemäß CDC Definition DPC in 61850-7-3
XCBR1/Pos/t[ST]	Zeitstempel	gemäß CDC Definition DPC in 61850-7-3
XCBR1/Pos/Oper.ctlVal[CO]	Wert des Kommandos	gemäß Oper Typ Definition in 61850-8-1, Table E.9

Alle Objekte im Variablenmodell sind auf diese Art aufgebaut. Deshalb weiß der Treiber zum Beispiel, dass die Attribute 'q' und 't' Qualität und Zeitstempel für die Variable in 'stVal' enthalten. Dies erlaubt den Zugriff auf alle Attribute jedes Objekts im Variablenmodell. Existierende Werte für Qualität und Zeitstempel werden automatisch den Istwerten der Variablen beim Treiber zugeordnet.

Operate Kommando: Dank der spezifischen Variablennomenklatur bemerkt der Treiber automatisch, dass ein Operate Kommando erstellt werden soll, wenn auf eine Variable geschrieben wird, z. B. mit dem Namen '*/Oper.ctlVal[CO]' (Operation.ControlValue). Dann sendet er automatisch die benötigte Sequenz an Telegrammen zum Kontrollobjekt.

ZEITSTEMPEL UND QUALITÄT

Zeitstempel- und Qualitätsattribut sind immer Teile von Variablen in zenon. Variablen mit Attributen 'q' und 't' müssen nicht getrennt erstellt werden. Der Treiber benutzt automatisch den Zeitstempel und die Qualität des Datenobjekts für seine Variablen mit **FC = ST und MX** (FC steht für Functional Constraint (auf Seite 80)). Ohne Rücksicht darauf ob eine Variable für Attribut 't' erstellt wurde, wird der empfangene Zeitstempel verwendet für alle **ST** und **MX** Variablen, zum Beispiel Variable **stVal** bei CEL, AML oder Archivserver. Auch werden ausgewählte Qualitätsbits, die in Attribut 'q' empfangen wurden, den Statusbits der **stVal** Variable zugeordnet (siehe auch Treiberspezifische Funktionen (auf Seite 57)).

7.3 Treiberobjekte und Datentypen

Treiberobjekte sind in der Steuerung verfügbare Bereiche wie z. B. Merker, Datenbausteine usw. Hier lesen Sie, welche Treiberobjekte vom Treiber zur Verfügung gestellt werden und welche IEC-Datentypen dem jeweiligen Treiberobjekt zugeordnet werden können.

7.3.1 Treiberobjekte

Folgende Objekttypen stehen in diesem Treiber zur Verfügung:

Treiberobjekttyp	Kanaltyp	Lesen / Schreiben	Unterstützte Datentypen	Kommentar
PLC marker	8	L/S	REAL, LREAL, WSTRING, DINT, UDINT, BOOL, INT, UINT, STRING, USINT, SINT	Data Attributes Variablen
File Transfer	9	L/S	STRING	Befehls- und Rückgabevariablen für den Dateitransfer. Hinweis: Die Stringlänge sollte maximaler Länge des Dateinamen (inklusive des Pfads) in der Steuerung entsprechen, z. B. für *!Command variable max. 260 Zeichen (aber für *!Directory mehr). Mehr Informationen finden Sie im Kapitel über treiberspezifische Funktionen (auf Seite 57)
State Info	11	L	UINT	Interne Variablen zeigen den aktuellen Status des Befehls. Mehr Informationen finden Sie im Kapitel über treiberspezifische Funktionen (auf Seite 57)
Treibervariable	35	L/S	BOOL, SINT, USINT, INT, UINT, DINT, UDINT, REAL, STRING	Variablen für die statische Analyse der Kommunikation; wird zwischen Treiber und Runtime übertragen (nicht zur SPS). Hinweis: Die Adressierung und das Verhalten ist gleich bei den meisten zenon Treiber. Mehr Information finden Sie im Kapitel über die **Treibervariablen** (auf Seite 52)

7.3.2 Zuordnung von Datentypen

Alle Variablen in zenon werden von IEC-Datentypen abgeleitet. In folgender Tabelle werden zur besseren Übersicht die IEC-Datentypen den Datentypen der Steuerung gegenübergestellt.

ZUORDNUNG VON DATENTYPEN VON DER SPS ZU ZENON DATENTYPEN

Steuerung	zenon	Datenart	Anmerkung
BOOLEAN	BOOL	8	
INT8	SINT	10	
INT8U	USINT	9	
INT16	INT	1	
INT16U	UINT	2	
INT24	DINT	3	
INT24U	UDINT	4	
INT32	DINT	3	
INT32U	UDINT	4	
INT128	DINT	3	
FLOAT32	REAL	5	
FLOAT64	LREAL	6	
ENUMERATED	INT	1	
CODED ENUM	UDINT, STRING	4, 12	MMS Bitstring
OCTET STRING	STRING	12	
VISIBLE STRING	STRING	12	
UNICODE STRING	WSTRING	21	Kodierung erfolgt in UTF-8
PACKED LIST	UDINT, STRING	4, 12	MMS Bitstring, umgedrehte Bitreihenfolge
TIMESTAMP	LREAL	6	nur Oktets 0..7 (siehe auch TimeQuality Bits mapping (auf Seite 57))
EntryTime	LREAL	6	
TriggerConditions	UDINT, STRING	4, 12	MMS Bitstring, umgedrehte Bitreihenfolge

Datenart: Die Eigenschaft `Datenart` ist die interne numerische Bezeichnung des Datentyps. Diese wird auch für den erweiterten DBF Import/Export der Variablen verwendet.

MMS Bitstring: Die Datenattributtypen, die gemäß dem Standard (siehe IEC61850-8-1) dem `MMS Bitstring` zugeordnet werden, z. B. `CODED ENUM`, `PACKED LIST` und `TrigerConditions`, entsprechen in zenon per default Datentyp UDINT. Vobei im Fall von `PACKED LIST` und `TrigerConditions` das Bit 0 is höchstwertig in seinem Oktet. Der Treiber ermöglicht Bitstring Variablen auch mit Datentyp STRING anstatt UDINT manuell einlegen. Der Wert des String enthält nur die Zeichen "0" und "1" in der gleichen Reihenfolge wie die Bits im Standard definiert sind und nur in der Bitstring-Länge, z. B. der Wert des TrgOps Attribute - Bitstring(6) - als String "011111" (in UDINT=124) bedeutet, dass alle TriggerCondition-Bits außer Bit 0 - `'Reserved'` gesetzt sind und der Wert "000010" (in UDINT=8) - nur Bit 4 - `'intergity'`.

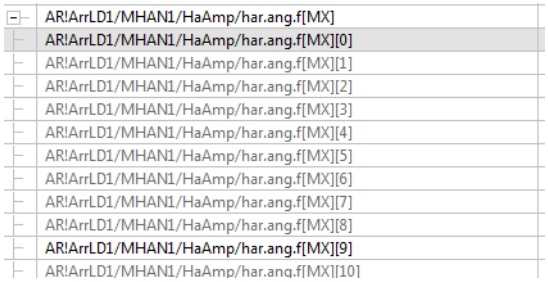
DATEN ATTRIBUTE DES TYPES "ARRAY OF"

Kommt in der IEC61850 Data Classs eine Array-Ebene vor, so wird diese Ebene für die Darstellung in zenon ans Ende der Variablenstruktur gezogen. Das heißt, aus einem **`ARRAY OF`** mit möglicherweise verschachtelten Strukturen entstehen im zenon Variablenmodell ein bzw. mehrere Arrays von skalaren Werten. Dieses Remapping erfolgt zur Projektierungszeit beim Onlineimport und zur Laufzeit bei der Zuordnung zwischen zenon Variablenliste und dem IEC61850 Objektmodell.

Dadurch besteht die Einschränkung, dass nur eine Array-Ebene unterstützt wird. Auch ist die Unterstützung von Arrays auf die Data Attribute beschränkt.

Beispiel

Das Data Attribut **`HaAmp.har`** der Common Data Class (CDC, -7-3) **`MHAN`** - im IEC61850 Standard definiert: `ARRAY[0..numHar] of Vector` - wird in zenon zu folgendem Array:



Hinzu kommen natürlich auch die Arrays für **`ang.i`**, **`mag.f`** und **`mag.i`**.

7.4 Variablen anlegen durch Import

Variablen können auch mittels Variablenimport angelegt werden. Für jeden Treiber stehen XML- und DBF-Import zur Verfügung.

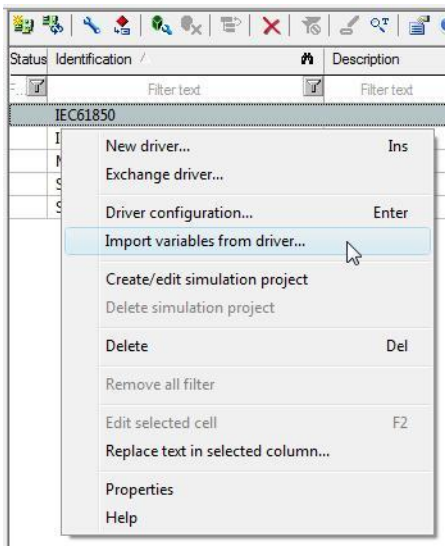
> **Info**
>
> *Details zu Import und Export von Variablen finden Sie im Handbuch Import-Export (main.chm::/13028.htm) im Abschnitt Variablen (main.chm::/13045.htm).*

7.4.1 Online-Import

Variablen in einem zenon Projekt können mit dem Import des Treibers erstellt werden. Der Treiber kann die Variablenlist des IEC 61850 Server browsen.

> **Achtung**
>
> *Wenn zur gleichen Zeit die Runtime auf dem gleichen Computer läuft, können die Variablen im Editor nicht von der Steuerung importiert werden, da der Treiber in der Runtime die gegebenen Einstellungen zur Kommunikation benutzt.*

Sie rufen den Import über das Kontextmenü des Treibers in der Treiberliste auf.



Im sich öffnenden Dialog wählen Sie die Option SPS Browsing und wählen Sie einen Server, von dem die Variablen gebrowst werden sollen.

Die Verbindung zum Server wird aufgebaut und die Liste der vorhandenen Datenpunkte (LDs, LNs, Datenobjekte, Datenattribute) wird gelesen.

Wenn das Einlesen beendet ist, öffnet sich ein wichtiger Dialog, in dem Sie die gebrowsten Variablen filtern (Groß- /Kleinschreibung, z. B. *XCBR1*stVal*), auswählen (auch Mehrfachauswahl) und von der oberen zur unteren Liste hinzufügen können.

Mit der Schaltfläche OK erstellen Sie die ausgewählten Variablen im Projekt.

Achtung

Der Treiber erstellt Variablen mit default Wertebereichen, z. B. ein UDINT mit <0.. 4294967295> und ein LREAL mit <-1000..1000>, die eventuell nicht den Wertebereichen der Attribute entsprechen, z. B. stVal-Attribut UDINT <0..3> oder t-Attribut LREAL<0.. 2147483647> mit drei Dezimalstellen für Millisekunden. Stellen Sie darum nach dem Import sicher, dass alle Variablen die richtigen Wertebereich-Eigenschaften haben und die String die richtigen Länge haben.

"ARRAY OF" IMPORT

Durch die Einschränkung, dass nur eine Array-Ebene unterstützt (auf Seite 39) wird, ist auch die Unterstützung von Arrays auf die Data-Attribute beschränkt.

Beim Onlineimport werden diese **ARRAY OF** Variablen als zenon Arrays angelegt.

Beispiel

Das Data Attribut **HaAmp.har** der Common Data Class **MHAN** - im IEC61850 Standard definiert: `ARRAY[0..numHar] of Vector` - wird in zenon zu bis zu vier Arrays, mit jeweils gleicher Anzahl an Elementen:

- ***/HaAmp.har.ang.f[MX][0..count-1]**
- ***/HaAmp.har.ang.i[MX][0..count-1]**
- ***/HaAmp.har.mag.f[MX][0..count-1]**
- ***/HaAmp.har.mag.i[MX][0..count-1]**

7.4.2 Offline-Import

Variablen können auch offline importiert werden. Die folgenden Dateiendungen werden akzeptiert:

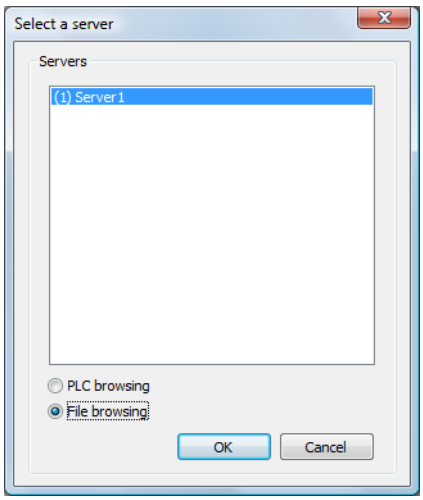
- `*.xml`
- `*.scl`
- `*.icd`
- `*.cid`
- `*.scd`

Der Treiber importiert offline nur DOI-Elemente. Die Eigenschaften `ReportControl` (Report Control Block) und `SettingControl` (Setting Control Block) werden beim Offline-Import nicht unterstützt. Sie können die Variablen für diese Attribute über Online Import oder manuell erstellen.

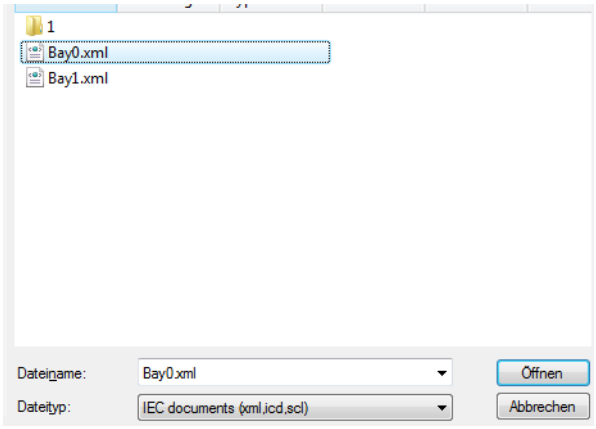
> **Achtung**
>
> *Beim Offline-Import werden vom IED direkt verlinkte Logical Nodes ignoriert. Das gilt nicht für Logical Devices. Der Treiber kann vom Objektmodell nur Attribute importieren, die mit dem IEC61850 Standard völlig konform sind.*

Um Variablen aus einer Datei zu importieren:

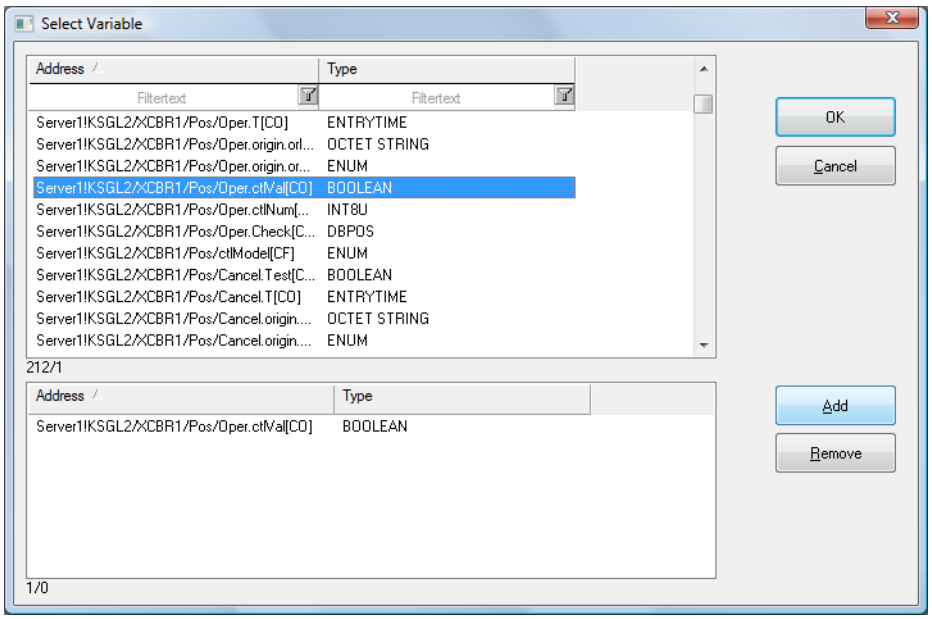
1. wählen Sie bei der Serverauswahl die Option `File browsing`



2. wählen Sie dann die Import-Datei aus.



3. die folgenden Schritte für die Auswahl der Variablen sind analog zum Online-Import.



7.4.3 XML Import

Für den Import/Export von Variablen gilt:

- Der Import/Export darf nicht aus dem Globalprojekt gestartet werden.
- Der Start erfolgt über:
  - Kontextmenü zu Variablen bzw. Datentyp im Projektbaum

- oder Kontextmenü einer Variablen bzw. eines Datentyps
- oder Symbol in der Symbolleiste Variablen

> **Achtung**
>
> *Beim Import/Überschreiben von existierenden Datentypen werden alle Variablen geändert, die auf diesem existierenden Datentyp basieren.*
>
> *Beispiel:*
>
> *Es existiert ein Datentyp XYZ abgeleitet vom Typ `INT` mit Variablen, die auf diesem Datentyp basieren. Ihre zu importierende XML-Datei enthält ebenfalls einen Datentyp mit Namen XYZ, allerdings abgeleitet vom Typ `STRING`. Wird dieser Datentyp importiert, so wird der existierende Datentyp überschrieben und bei allen auf ihm basierenden Variablen der Typ angepasst. D.h. die Variablen sind jetzt `STRING`- und keine `INT`-Variablen mehr.*

7.4.4 DBF Import/Export

Daten können nach dBase exportiert und aus dBase importiert werden.

> **Info**
>
> *Import und Export über CSV oder dBase unterstützt keine treiberspezifischen Variableneinstellungen wie z. B. Formeln. Nutzen Sie dafür den Export/Import über XML.*

IMPORT DBF-DATEI

Um den Import zu starten:

1. führen Sie einen Rechtsklick auf die Variablenliste aus
2. wählen Sie im Dropdownliste von **`Erweiterter Export/Import`** ... den Befehl **`dBase importieren`**
3. folgen Sie dem Importassistenten

Das Format der Datei ist im Kapitel Dateiaufbau beschrieben.

> **Info**
>
> *Beachten Sie:*
>
> - Treiberobjekttyp und Datentyp müssen in der DBF-Datei an den Zieltreiber angepasst werden, damit Variablen importiert werden.
> - dBase unterstützt beim Import keine Strukturen oder Arrays (komplexe Variablen).

EXPORT DBF-DATEI

Um den Export zu starten:

1. führen Sie einen Rechtsklick auf die Variablenliste aus
2. wählen Sie im Dropdownliste von **Erweiterter Export/Import** ... den Befehl **dBase exportieren**
3. folgen Sie dem Exportassistenten

> **Achtung**
>
> DBF-Dateien:
>
> - müssen in der Benennung dem 8.3 DOS Format für Dateinamen entsprechen (8 alphanumerische Zeichen für Name, 3 Zeichen Erweiterung, keine Leerzeichen)
> - dürfen im Pfadnamen keinen Punkt (.) enthalten.
>   Z. B. ist der Pfad `C:\users\Max.Mustermann\test.dbf` ungültig.
>   Gültig wäre: `C:\users\MaxMustermann\test.dbf`
> - müssen nahe am Stammverzeichnis (Root) abgelegt werden, um die eventuelle Beschränkungen für Dateinamenlänge inklusive Pfad zu erfüllen: maximal 255 Zeichen

Das Format der Datei ist im Kapitel Dateiaufbau beschrieben.

> **Info**
>
> *dBase unterstützt beim Export keine Strukturen oder Arrays (komplexe Variablen).*

Dateiaufbau der dBase Exportdatei

Für den Variablenimport und -export muss die dBaseIV-Datei folgende Struktur und Inhalte besitzen.

Achtung

dBase unterstützt keine Strukturen oder Arrays (komplexe Variablen).

DBF-Dateien müssen:

- in der Benennung dem 8.3 DOS Format für Dateinamen entsprechen (8 alphanumerische Zeichen für Name, 3 Zeichen Erweiterung, keine Leerzeichen)
- nahe am Stammverzeichnis (Root) abgelegt werden

STRUKTUR

Bezeichnung	Typ	Feldgröße	Bemerkung
KANALNAME	Char	128	Variablenname. Länge kann über den Eintrag MAX_LAENGE in der **project.ini** eingeschränkt werden.
KANAL_R	C	128	Ursprünglicher Name einer Variablen, der durch den Eintrag unter VARIABLENNAME ersetzt werden soll (Feld/Spalte muss manuell angelegt werden). Länge kann über den Eintrag MAX_LAENGE in der **project.ini** eingeschränkt werden.
KANAL_D	Log	1	Variable wird bei Eintrag 1 gelöscht (Feld/Spalte muss manuell angelegt werden).
TAGNR	C	128	Kennung. Länge kann über den Eintrag MAX_LAENGE in der **project.ini** eingeschränkt werden.
EINHEIT	C	11	Technische Maßeinheit
DATENART	C	3	Datentyp (z. B. Bit, Byte, Wort, ...) entspricht dem Datentyp.
KANALTYP	C	3	Speicherbereich in der SPS (z. B. Merkerbereich, Datenbereich, ...) entspricht Treiber-Objekttyp.
HWKANAL	Num	3	Bus-Adresse
BAUSTEIN	N	3	Datenbaustein-Adresse (nur bei Variablen aus den Datenbereich der SPS)
ADRESSE	N	5	Offset

BITADR	N	2	Für Bit-Variablen: Bitadresse Für Byte-Variablen: 0=niederwertig, 8=höherwertig Für String-Variablen: Stringlänge (max. 63 Zeichen)
ARRAYSIZE	N	16	Anzahl der Variablen im Array für Index-Variablen ACHTUNG: Nur die erste Variable steht voll zur Verfügung. Alle folgenden sind nur über VBA oder den Rezeptgruppen Manager zugänglich
LES_SCHR	L	1	Lese-Schreib-Berechtigung 0: Sollwert setzen ist nicht erlaubt 1: Sollwert setzen ist erlaubt
MIT_ZEIT	L	1	Zeitstempelung in zenon (nur wenn vom Treiber unterstützt)
OBJEKT	N	2	Treiberspezifische ID-Nummer des Primitivobjekts setzt sich zusammen aus TREIBER-OBJEKTTYP und DATENTYP
SIGMIN	Float	16	Rohwertsignal minimal (Signalauflösung)
SIGMAX	F	16	Rohwertsignal maximal (Signalauflösung)
ANZMIN	F	16	technischer Wert minimal (Messbereich)
ANZMAX	F	16	technischer Wert maximal (Messbereich)
ANZKOMMA	N	1	Anzahl der Nachkommastellen für die Darstellung der Werte (Messbereich)
UPDATERATE	F	19	Updaterate für Mathematikvariablen (in sec, eine Dezimalstelle möglich) bei allen anderen Variablen nicht verwendet
MEMTIEFE	N	7	Nur aus Kompatibilitätsgründen vorhanden
HDRATE	F	19	HD-Updaterate für hist. Werte (in sec, eine Dezimalstelle möglich)
HDTIEFE	N	7	HD-Eintragtiefe für hist. Werte (Anzahl)
NACHSORT	L	1	HD-Werte als nachsortierte Werte
DRRATE	F	19	Aktualisierung an die Ausgabe (für zenon DDE-Server, in sec, eine Kommastelle möglich)
HYST_PLUS	F	16	Positive Hysterese; ausgehend vom Messbereich
HYST_MINUS	F	16	Negative Hyterese; ausgehend vom Messbereich
PRIOR	N	16	Priorität der Variable

REAMATRIZE	C	32	Name der zugeordnete Reaktionsmatrix
ERSATZWERT	F	16	Ersatzwert; ausgehend vom Messbereich
SOLLMIN	F	16	Sollwertgrenze Minimum; ausgehend vom Messbereich
SOLLMAX	F	16	Sollwertgrenze Maximum; ausgehend vom Messbereich
VOMSTANDBY	L	1	Variable vom Standby-Server anfordern; der Wert der Variable wird im redundanten Netzwerkbetrieb nicht vom Server sondern vom Standby-Server angefordert
RESOURCE	C	128	Betriebsmittelkennung. Freier String für Export und Anzeige in Listen. Länge kann über den Eintrag MAX_LAENGE in der **project.ini** eingeschränkt werden.
ADJWVBA	L	1	Nichtlineare Wertanpassung: 0: Nichtlineare Wertanpassung wird verwendet 1: Nichtlineare Wertanpassung wird nicht verwendet
ADJZENON	C	128	Verknüpftes VBA-Makro zum Lesen der Variablenwerte für die nichtlineare Wertanpassung.
ADJWVBA	C	128	Verknüpftes VBA-Makro zum Schreiben der Variablenwerte für die nichtlineare Wertanpassung.
ZWREMA	N	16	Verknüpfte Zählwert-Rema.
MAXGRAD	N	16	Maximaler Gradient für die Zählwert-Rema.

Achtung

Beim Import müssen Treiberobjekttyp und Datentyp in der DBF-Datei an den Zieltreiber angepasst werden, damit Variablen importiert werden.

GRENZWERTDEFINITION

Grenzwertdefinition für Grenzwert 1 bis 4, bzw. Zustand 1 bis 4:

Bezeichnung	Typ	Feldgröße	Bemerkung
AKTIV1	L	1	Grenzwert aktiv (pro Grenzwert vorhanden)
GRENZWERT1	F	20	technischer Wert oder ID-Nummer der verknüpften Variable für einen dynamischen Grenzwert (siehe VARIABLEx) (wenn unter VARIABLEx 1 steht und hier −1, wird die bestehende Variablenzuordnung nicht überschrieben)
SCHWWERT1	F	16	Schwellwert für den Grenzwert
HYSTERESE1	F	14	wird nicht verwendet
BLINKEN1	L	1	Blinkattribut setzen
BTB1	L	1	Protokollierung in CEL
ALARM1	L	1	Alarm
DRUCKEN1	L	1	Druckerausgabe (bei CEL oder Alarm)
QUITTIER1	L	1	quittierpflichtig
LOESCHE1	L	1	löschpflichtig
VARIABLE1	L	1	dyn. Grenzwertverknüpfung der Grenzwert wird nicht durch einen absoluten Wert (siehe Feld GRENZWERTx) festgelegt.
FUNC1	L	1	Funktionsverknüpfung
ASK_FUNC1	L	1	Ausführung über die Alarmmeldeliste
FUNC_NR1	N	10	ID-Nummer der verknüpften Funktion (steht hier -1, so wird die bestehende Funktion beim Import nicht überschrieben)
A_GRUPPE1	N	10	Alarm/Ereignis-Gruppe
A_KLASSE1	N	10	Alarm/Ereignis-Klasse
MIN_MAX1	C	3	Minimum, Maximum
FARBE1	N	10	Farbe als Windowskodierung
GRENZTXT1	C	66	Grenzwerttext
A_DELAY1	N	10	Zeitverzögerung
INVISIBLE1	L	1	Unsichtbar

BEZEICHNUNGEN IN DER SPALTE BEMERKUNG BEZIEHEN SICH AUF DIE IN DEN DIALOGBOXEN ZUR DEFINITION VON VARIABLEN VERWENDETEN BEGRIFFE. BEI UNKLARHEITEN, SIEHE KAPITEL VARIABLENDEFINITION.

7.5 Treibervariablen

Das Treiberkit implementiert eine Reihe von Treibervariablen. Diese sind unterteilt in:

- Information
- Konfiguration
- Statistik und
- Fehlermeldungen

Die Definitionen der im Treiberkit implementierten Variablen sind in der Importdatei **drvvar.dbf** (auf der CD im Verzeichnis: CD_Laufwerk:/Predefined/Variables) verfügbar und können von dort importiert werden.

Hinweis: Variablennamen müssen in zenon einzigartig sein. Soll nach einem Import der Treibervariablen aus **drvvar.dbf** ein erneuter Import durchgeführt werden, müssen die zuvor importierten Variablen umbenannt werden.

Info

Nicht jeder Treiber unterstützt alle Treibervariablen.

Zum Beispiel werden:

- Variablen für Modem-Informationen nur von modemfähigen Treibern unterstützt
- Treibervariablen für den Polling-Zyklus nur für rein pollenden Treibern
- verbindungsbezogene Informationen wie ErrorMSG nur von Treibern, die zu einem Zeitpunkt nur eine Verbindung bearbeiten

INFORMATION

Name aus Import	Typ	Offset	Erklärung
MainVersion	UINT	0	Haupt-Versionsnummer des Treibers.
SubVersion	UINT	1	Sub-Versionsnummer des Treibers.
BuildVersion	UINT	29	Build-Versionsnummer des Treibers.
RTMajor	UINT	49	zenon Hauptversionsnummer
RTMinor	UINT	50	zenon Sub-Versionsnummer
RTSp	UINT	51	zenon Servicepack-Nummer
RTBuild	UINT	52	zenon Buildnummer
LineStateIdle	BOOL	24.0	TRUE, wenn die Modemleitung belegt ist.
LineStateOffering	BOOL	24.1	TRUE, wenn ein Anruf rein kommt.
LineStateAccepted	BOOL	24.2	Der Anruf wird angenommen.
LineStateDialtone	BOOL	24.3	Rufton wurde erkannt.
LineStateDialing	BOOL	24.4	Wahl aktiv.
LineStateRingBack	BOOL	24.5	Während Verbindungsaufbau.
LineStateBusy	BOOL	24.6	Zielstation besetzt.
LineStateSpecialInfo	BOOL	24.7	Spezielle Statusinformation empfangen.
LineStateConnected	BOOL	24.8	Verbindung hergestellt.
LineStateProceeding	BOOL	24.9	Wahl ausgeführt.
LineStateOnHold	BOOL	24.10	Verbindung in Halten.
LineStateConferenced	BOOL	24.11	Verbindung im Konferenzmodus.
LineStateOnHoldPendConf	BOOL	24.12	Verbindung in Halten für Konferenz.
LineStateOnHoldPendTransfer	BOOL	24.13	Verbindung in Halten für Transfer.
LineStateDisconnected	BOOL	24.14	Verbindung beendet.
LineStateUnknow	BOOL	24.15	Verbindungszustand nicht bekannt.
ModemStatus	UDINT	24	Aktueller Modemstatus.
TreiberStop	BOOL	28	Treiber gestoppt

			Bei `Treiberstop`, hat die Variable den Wert `TRUE` und ein **OFF**-Bit. Nach dem Treiberstart, hat die Variable den Wert `FALSE` und kein **OFF**-Bit.
SimulRTState	UDINT	60	Informiert über Status der Runtime bei Treibersimulation.

KONFIGURATION

Name aus Import	Typ	Offset	Erklärung
ReconnectInRead	BOOL	27	Wenn TRUE, dann wird beim Lesen automatisch ein Neuaufbau der Verbindung durchgeführt.
ApplyCom	BOOL	36	Änderungen an den Einstellungen der seriellen Schnittstelle zuweisen. Das Schreiben auf diese Variable hat unmittelbar den Aufruf der Methode SrvDrvVarApplyCom zur Folge (aktuell ohne weitere Funktion).
ApplyModem	BOOL	37	Änderungen an den Modemeinstellungen zuweisen. Das Schreiben auf diese Variable hat unmittelbar den Aufruf der Methode SrvDrvVarApplyModem zur Folge. Diese schließt die aktuelle Verbindung und öffnet eine neue entsprechend den Einstellungen **`PhoneNumberSet`** und **`ModemHwAdrSet`**.
PhoneNumberSet	STRING	38	Telefonnummer, welche verwendet werden soll.
ModemHwAdrSet	DINT	39	Hardwareadresse, welche zu der Telefonnummer gehört.
GlobalUpdate	UDINT	3	Updatezeit in Millisekunden (ms).
BGlobalUpdaten	BOOL	4	TRUE, wenn die Updatezeit global ist.
TreiberSimul	BOOL	5	TRUE, wenn der Treiber in Simulation ist.

TreiberProzab	BOOL	6	TRUE, wenn das Prozessabbild gehalten werden soll.
ModemActive	BOOL	7	TRUE, wenn das Modem bei diesem Treiber aktiv ist.
Device	STRING	8	Name der seriellen Schnittstelle oder Name des Modem.
ComPort	UINT	9	Nummer der seriellen Schnittstelle.
Baudrate	UDINT	10	Baudrate der seriellen Schnittstelle.
Parity	SINT	11	Parität der seriellen Schnittstelle.
ByteSize	USINT	14	Bitanzahl pro Zeichen der seriellen Schnittstelle. Wert = 0, wenn der Treiber keine serielle Kommunikation herstellen kann.
StopBit	USINT	13	Anzahl der Stoppbits der seriellen Schnittstelle.
Autoconnect	BOOL	16	TRUE, wenn die Modemverbindung automatisch beim Lesen/Schreiben aufgebaut werden soll.
PhoneNumber	STRING	17	Aktuelle Telefonnummer.
ModemHwAdr	DINT	21	Hardwareadresse zur aktuellen Telefonnummer.
RxIdleTime	UINT	18	Wenn länger als diese Zeit in Sekunden (s) erfolgreich kein Datenverkehr stattfindet, wird die Modemverbindung beendet.
WriteTimeout	UDINT	19	Maximale Schreibdauer bei einer Modemverbindung in Millisekunden (ms).
RingCountSet	UDINT	20	So oft läutet ein hereinkommender Anruf, bevor dieser angenommen wird.

ReCallIdleTime	UINT	53	Wartezeit zwischen Anrufen in Sekunden (s).
ConnectTimeout	UINT	54	Zeit in Sekunden (s) für Verbindungsaufbau.

STATISTIK

Name aus Import	Typ	Offset	Erklärung
MaxWriteTime	UDINT	31	Längste Zeit in Millisekunden (ms), die zum Schreiben benötigt wird.
MinWriteTime	UDINT	32	Kürzeste Zeit in Millisekunden (ms), die zum Schreiben benötigt wird.
MaxBlkReadTime	UDINT	40	Längste Zeit in Millisekunden (ms), die zum Lesen eines Datenblocks benötigt wird.
MinBlkReadTime	UDINT	41	Kürzeste Zeit in Millisekunden (ms), die zum Lesen eines Datenblocks benötigt wird.
WriteErrorCount	UDINT	33	Anzahl der Schreibfehler.
ReadSucceedCount	UDINT	35	Anzahl der erfolgreichen Leseversuche.
MaxCycleTime	UDINT	22	Längste Zeit in Millisekunden (ms), die zum Lesen aller angeforderten Daten benötigt wurde.
MinCycleTime	UDINT	23	Kürzeste Zeit in Millisekunden (ms), die zum Lesen aller angeforderten Daten benötigt wurde.
WriteCount	UDINT	26	Anzahl der Schreibversuche.
ReadErrorCount	UDINT	34	Anzahl der fehlerhaften Leseversuche.
MaxUpdateTimeNormal	UDINT	56	Zeit seit letzter Aktualisierung der Prioritätsgruppe **`Normal`** in Millisekunden (ms).
MaxUpdateTimeHigher	UDINT	57	Zeit seit letzter Aktualisierung der Prioritätsgruppe **`Höher`** in Millisekunden (ms).
MaxUpdateTimeHigh	UDINT	58	Zeit seit letzter Aktualisierung der Prioritätsgruppe **`Hoch`** in Millisekunden (ms).

MaxUpdateTimeHighest	UDINT	59	Zeit seit letzter Aktualisierung der Prioritätsgruppe **Höchste** in Millisekunden (ms).
PokeFinish	BOOL	55	Geht für eine Abfrage auf 1, wenn alle anstehenden Pokes ausgeführt wurden.

FEHLERMELDUNGEN

Name aus Import	Typ	Offset	Erklärung
ErrorTimeDW	UDINT	2	Zeit (in Sekunden seit 1.1.1970), wann der letzte Fehler auftrat.
ErrorTimeS	STRING	2	Zeit (in Sekunden seit 1.1.1970), wann der letzte Fehler als String auftrat.
RdErrPrimObj	UDINT	42	Nummer des PrimObjektes, als der letzte Lesefehler verursacht wurde.
RdErrStationsName	STRING	43	Name der Station, als der letzte Lesefehler verursacht wurde.
RdErrBlockCount	UINT	44	Anzahl der zu lesenden Blöcke, als der letzte Lesefehler verursacht wurde.
RdErrHwAdresse	DINT	45	Hardwareadresse, als der letzte Lesefehler verursacht wurde.
RdErrDatablockNo	UDINT	46	Bausteinnummer, als der letzte Lesefehler verursacht wurde.
RdErrMarkerNo	UDINT	47	Merkernummer, als der letzte Lesefehler verursacht wurde.
RdErrSize	UDINT	48	Blockgröße, als der letzte Lesefehler verursacht wurde.
DrvError	USINT	25	Fehlermeldung als Nummer.
DrvErrorMsg	STRING	30	Fehlermeldung als Klartext.
ErrorFile	STRING	15	Name der Fehlerprotokolldatei.

8. Treiberspezifische Funktionen

Spezielle Spezifikationen des IEC 61850 Standard wurden als spezifische Funktionen im IEC 850 Treiber implementiert.

ASSOCIATION

Der Treiber öffnet eine Verbindung zu einem Logical Device nur, wenn mindestens eine Variable aus dem LD im Projekt angelegt ist.

Nach einen Abbruch der Kommunikation (z. B. Netzwerkausfall) wird die Verbindung zum Logical Device nur dann automatisch wiederhergestellt, wenn wenigstens eine Variable des LD gerade gepollt wird.

Empfehlung: Erstellen Sie für jedes LD wenigstens eine Variable (z. B. **`*/LLN0/Mod/stVal[ST]`** oder **`*ctlModel[CF]`**) für ein obligatorisches Attribut und aktivieren Sie für die Variable die Eigenschaft Variable ständig lesen (Gruppe Zusätzliche Einstellungen), damit die Variable immer beim Treiber angemeldet bleibt. Dieses Attribut darf nicht über Reports kommuniziert werden sondern sollte pollende Kommunikation verwenden.

COMMANDS (CONTROL MODEL)

Der Treiber unterstützt die Befehle 'Direct Operate' und 'Select Before Operate' (SBO) mit normaler und erhöhter Sicherheit. Der Treiber überprüft automatisch den tatsächlichen Control Model und führt das Kommando entsprechend aus.

Wenn ein Wert auf eine Projektvariable gesetzt ist, die dem Item ctlVal in einem Dataattribut mit dem Namen 'Oper' entspricht und es die Functional Constraint CO oder SP (*/Oper.ctlVal[CO|SP]) hat, überprüft der Treiber automatisch welches Control Model für einen Befehl bei diesem Dataobjekt benötigt wird.
Dafür wird zuerst das Attribut */ctlModel[CF] dieses Data Object ausgelesen. Abhängig vom Wert dieses Attributs, wird der entsprechende Befehl ausgeführt und der jeweilige Control Model benutzt:

Wert 0: keine Aktion, Ausführung des Befehls nicht möglich (nur Status)
Wert 1: ein 'Operate' Dienst (direct control with normal security) wird ausgeführt.
Wert 2: 'Select' und 'Operate' Dienste (select before operate with normal security) werden ausgeführt.
Wert 3: ein 'Operate' Dienst (direct control with enhanced security) wird ausgeführt.
Wert 4: 'SelectWithVariable' und 'Operate' Dienste (select before operate with enhanced security) werden ausgeführt.

Info

*Der Treiber liest das **`ctlModel`** Attribut vom Server, um den Befehl auszuführen, aber der Treiber überträgt den **`ctlModel`** Wert nicht automatisch zur Runtime. Auch dann nicht, wenn für das **`ctlModel`** eine Variable erstellt wurde. Wenn eine Variable erstellt wurde, wird sie gemäß der Polling-Rate, die in der Treiberkonfiguration definiert ist, aktualisiert.*

DER BEFEHL IM DETAIL

`Operate` Kommando: Der Wert wird direkt auf `*/Oper[CO]` gesetzt.

`Select` Kommandos werden sequentiell verarbeitet.

- `Select` Kommando mit normaler Sicherheit: lesen von `*/SBO[CO]`, schreiben auf `*/Oper[CO]`.
- `Select` Kommando mit erhöhter Sicherheit: schreiben auf `*/SBOw[CO]`, schreiben auf `*/Oper[CO]`.

Die erforderliche Kommandosequenzen werden vom Treiber automatisch in der richtigen Reihenfolge ausgeführt. Keine weiteren Konfigurationsschritte sind erforderlich.

MANUELLES SELECT UND CANCEL

Zusätzlich unterstützt der Treiber manuelles Select und Cancel, aber nur über das Modul **`Befehlsgabe`** in zenon.
Wenn die Befehlsgabe einen Befehl sendet und die Eigenschaft `Select Before Operate` aktiv ist, führt der Treiber - gemäß dem **`ctlModel`** des selben Objekts - ein Select aus. Das gilt nur für`Model=2` oder `4`. Die Werte des `ctlModel=0`, `1` oder `3` werden als negative Antwort des Select behandelt.

Hinweis: `Select Before Operate` meint die Eigenschaft bei der **`Oper.ctlVal`** Variable in zenon und darf nicht mit dem Attribut `SBO` verwechseln),

Wenn der IEC 61850 Server auf das Select positiv antwortet, speichert der Treiber die Information über das ausgewählte Dataobjekt und im folgenden Befehl (aus der zweiten Stufe der Befehlsgabe) führt der Treiber nur noch das Operate aus.
Wenn die Befehlsgabe auf der zweiten Stufe abgebrochen wird, sendet der Treiber ein Cancel.

Der Treiber ignoriert die Eigenschaft `Select Before Operate` der **`Oper.ctlVal`** Variablen, wenn der Wert der `ctlVal` direkt gesetzt wurde und nicht durch eine Aktion der Befehlsgabe.

CTLNUM (CONTROL SEQUENCE NUMBER)

Der Treiber erstellt automatisch die Werte des `ctlNum` Attributes. Wenn ein Wert auf die Projektvariable gesetzt wurde, entsprechend dem Attribut `*/Oper.ctlVal[CO|SP]`, erhöht der Treiber automatisch den Wert des `*/Oper.ctlNum[CO]`. Der Treiber fängt beim Inkrementieren des `ctlNum` wieder bei `0` an, wenn der Wert `127` erreicht hat.
Der Treiber zählt auch `ctlModel=0` (keine Aktion). Bei diesem Model wird die **`ctlNum`** Variable in der

Runtime nicht aktualisiert, da eigentlich - gemäß dem Standard (IEC61850-7-3) - das Attribut `ctlVal` nicht existieren sollte.

ADDITIONAL CAUSE DIAGNOSIS UND 'STATE INFO' VARIABLEN

Für negative Antworten eines Befehls, hält der IEC61850 Server das AdditionalCauseDiagnosis bereit (AddCause in LastApplError Struktur). Die möglichen Werte der AddCause Service-Parameter sind in der IEC 61850-7-2 gelistet. Der Treiber kann den empfangenen AddCause an zenon weitergeben und auch den aktuellen Zustand des laufenden Befehls wiedergeben (z. B. Select/Operate/Cancel) und Auskunft darüber geben, ob der Befehl erfolgreich war.

Der Treiber stellt diese Information in Variablen des Treiberobjekttyps `State Info` zur Verfügung (siehe Treiberobjekte (auf Seite 38)), wenn diese erstellt wurden. Variablen, die die Stufe des aktuellen Befehls wiedergeben, können für jede **`*/Oper`**-Struktur erstellt werden.

- `*/Oper.AddCause`: Nach einem gescheiterten Befehl, zeigt der Wert Fehlerinformation, die er von der Steuerung erhalten hat.
- `*/Oper.ControlRun`: Der Zustand oder das Ergebnis des aktuellen Befehls:
  - `1` = Befehl läuft
  - `0` = Befehl erfolgreich
  - `-1` = Befehl gescheitert
- `*/Oper.ControlState`: Der Zustand des gerade laufenden Befehls (sequentiell verarbeitet):
  - `-1` = Befehl beendet
  - `0` = ctlModel lesen
  - `1` = 'Select' gesendet
  - `2` = 'Operate' gesendet
  - `3` = 'Cancel' gesendet

Beispiel:

Für die Variable 'PLC-Marker'-BOOL 'GE650!GEDeviceF650/XCBR1/Pos/Oper.ctlVal[CO]' können Sie die folgenden Zustandvariablen erstellen:

State Info'-SINT 'GE650!GEDeviceF650/XCBR1/Pos/Oper.AddCause

State Info'-SINT 'GE650!GEDeviceF650/XCBR1/Pos/Oper.ControlRun

State Info'-SINT 'GE650!GEDeviceF650/XCBR1/Pos/Oper.ControlState

Fall 1:

Wenn Sie den Wert direkt auf `*/Oper.ctlVal` setzen oder die Variable die Eigenschaft `SBO` nicht besitzt, ist die Reihenfolge der Zustände die folgende:

1. */Oper.ControlRun = 1 und */Oper.ControlState = 0
2. */Oper.ControlState = 1 und dann */Oper.ControlState = 2
3. */Oper.ControlRun = 0 und */Oper.ControlState = -1

Fall 2:

Wenn Sie die zweistufige Befehlsgabe für `*/Oper.ctlVal` mit Eigenschaft `SBO` verwenden, ist die Reihenfolge der Zustände für jeden Befehlsdienst (für Select, Operate und Cancel) die folgende:

1. */Oper.ControlRun = 1 und */Oper.ControlState = 0
2. */Oper.ControlState = x
3. */Oper.ControlRun = 0 und */Oper.ControlState = -1

Dabei gilt: x=1 oder 2 oder 3 ist, je nach Befehlsdienst.

Achtung

*Die **`State Info`** Variablen haben keinen Wert so lange kein Befehl gegeben wurde. Um Statusvariablen auszuwerten, müssen Sie zuerst einen Wert auf die **`*/Oper.ctlVal[CO|SP]`** Variable setzen.*

SERVICEPARAMETER 'CHECK' FÜR BEFEHLE

Es gibt zwei Methoden, um Check-Bedingungen für die folgenden Operate oder SelectWithValue Befehle zu setzen:

1. Check-Bedingung kann durch das Setzen eines Werts für */Oper.Check[CO] vordefiniert werden. Der aktuelle Wert dieses Attributs wird für Befehlsservice-Parameter verwendet.
   Achtung: Check Attribut ist eine PACKED LIST (MMS Bitstring) mit 2 Bit. Gemäß IEC61850-8-1 8.1.3.5 soll das erste Member der PACKED LIST auf Bit(0) gemappt werden und dieses Bit soll das höchstwertige (most significant) Bit des Oktet sein. Das heißt, dass die gültigen Sollwerte für das Check Attribut 0x80 (synchrocheck), 0x40 (interlock-check), 0xC0 (beide Check-Bits) und 0x00 sind.

2. In zenon durch Setzen in der Befehlsgabe: Aktion - Eigenschaft `Qualifier of Command`. Die gültigen Sollwerte sind 2 - synchrocheck (umgedreht 01b), 1 - interlock-check (umgedreht 10b), 3 (beide Bits) und 0.

ORIGINATOR

Die Originator Attribute orCat (Category) und orIdent (Identification) zeigen, welcher Originator (z. B. ein Client der sich auf bay level befindet) die letze Änderung der Daten (z. B. ein Kommando ausgegeben) verursacht hat.

Die Origin Attribute sind obligatorischer Teil der Oper, SBOw und Cancel Dienste und optional z. B. beim Datenobjekt 'Pos'. Um in der Runtime den Originator der empfangenen Werte der z. B. */Pos/stVal[ST] zu analysieren, können Sie in zenon Projektvariablen mit entsprechenden Namen anlegen: */Pos/origin.orCat[ST] und */Pos/origin.orIdent[ST].
Wichtig: Um das Origin eines Datenobjekts zu analysieren, muss in einem IEC61850 Server implementiert sein, dass der Server das Origin des empfangenen Befehls in diese Attribute schreibt, z. B. **`von * /Oper.origin.*[CO]`**.

Beim Empfang eines Reports leitet der Treiber den Originator an die Runtime weiter bevor er die anderen Attribute des empfangenen Datenobjekts weiterleitet (wenn nur ein Origin mit Functional Constraint ST im berichteten Datenobjekt existiert).

Hinweis: Sie können **`orCat`** oder **`orIdent`** Variablen benutzen, um einen dynamischen Grenzwerttext für eine berichtete **`stVal`** Variable zusammenzustellen.

> **Info**
>
> *Beim Ausführen eines Befehls setzt der Treiber für die Originator Attribute in SBOw, Oper und Cancel den Wert:*
> `orCat = 2` *(station control)*
>
> *Den Wert für* `orCat` *können Sie in den Basic Settings (auf Seite 16) manuell anpassen.*
> `orIdent` *= 'zenon: <Computername>' (String zwischen Anführungszeichen - CHAR(27)) , e.g. 'zenon: MyPC'*

MAPPING VON DOUBLE POINT VALUES

Gemäß des IEC61850-7-3 Standard hat in gemeinsamen Datenklassen Double Point Status (DPS) und Controllable Double Point (DPC) das Attribut stVal den Datentyp `CODED ENUM` mit Wertebereich: intermediate-state (00) | off (01) | on (10) | bad-state (11). Daher wartet der Treiber, dass ein Double Point Attribut des Datentyp `CODED ENUM` auf dem Kommunikationsprotokoll als ein MMS des Datentyps

Bitstring gemappt ist und somit die folgenden Werte haben kann: 0x00, 0x40, 0x80 und 0xC0. Dieser Basis-Datentyp ist in der SCL Sprache mit dem Namen 'Dbpos' zulässig (IEC61850-6 Kapitel 9.5.4.2 Attribute Basistypen). In zenon entspricht dem eine Variable mit Datentyp UDINT (siehe Zuordnung der Datentypen (auf Seite 39)).

Wie in der Treiberkonfiguration bleibt die Option 'Deactivate standard double point value mapping' als default auf inaktiv. Um die Module aus zenon Energy zu verwenden, z. B. die Funktionalitäten der **Befehlsgabe**, ordnet der Treiber Werte der stVal Attribute von DPS und DPC gemäß der folgenden Tabelle zu:

Positionswert	zenon Dbpos Wert - STRING	zenon Dbpos Wert - UDINT ungemappt	zenon Wert - UDINT gemappt
Differenzstellung (00)	'00'	0	2
Aus (01)	'01'	64 (0x40)	0
Ein (10)	'10'	128 (0x80)	1
Störung (11)	'11'	192 (0xC0)	3

Der Treiber konvertiert nur Werte für Variablen mit Namen ***/stVal** und Datentyp UDINT.

QUALITY BITS, ZEITSTEMPEL UND TIMEQUALITY BITS MAPPING

Der Treiber kann empfangene Attribute 'q' und '**t**' in entsprechenden Projektvariablen unterstützen, wenn diese erstellt wurden.
Aber der Treiber ordnet immer noch ausgewählte Quality Bits (des Attributs '**q**') und TimeQuality Bits (des Attributs '**t**') entsprechenden Statusbits jeder zenon Variable zu, die für ein Attribut des Datenobjekts erstellt wurde und eine **FC (auf Seite 80)=ST** oder **FC=MX** hat, z. B. für Statusbits von **stVa**l. Der Treiber ordnet auch der Wert des Attributs 't' dem Zeitstempel aller **ST/MX** Variablen des Datenobjekts zu.

Zum Beispiel: Der Zeitstempel, empfangen als '**t**' Attribut, wird für die entsprechende **stVal** Variable in der **CEL** verwendet. Der Treiber benutzt automatisch den Zeitstempel und die Qualität des Datenobjekts für alle seine Variablen, die im Projekt erstellt wurden. Deshalb müssen keine Variablen für die Attribute 'q' und 't' separat erstellt werden, außer wenn Sie die Qualitätsbits analysieren möchten, die keinen Statusbits zugeordnet sind.

Info

Der Treiber berücksichtigt den von TimeQuality erhaltenen Genauigkeitswert. Ist dieser Wert nicht unspezifisch (alle Bits), kürzt der Treiber den Zeitwert entsprechend.

Zum Beispiel: Genauigkeitswert < 10 Bit: der Treiber kürzt den erhaltenen Zeitwert bis auf die Millisekunden.

Die ausgewählten Quality und TimeQuality Bits werden den Statusbits der Variablen der antworteten Datenobjekte wie folgt zugeordnet:

IEC61850 Quality oder TimeQuality Bits	Statusbit in zenon	Anmerkungen
Validity = Invalid oder Reserved oder Questionable	INVALID	Gekennzeichnet durch ein rotes Quadrat am verknüpften Bildelement.
Overflow	OV_870 (Overflow)	auch ein IEC 60870 Status
Source = Substituted	SB_870 (Substituted)	auch ein IEC 60870 Status
Test	TEST (Test bit)	auch ein IEC 60870 Status
OperatorBlocked	BL_870 (Blocked)	auch ein IEC 60870 Status
ClockFailure	T_INVAL (Zeit ungültig) und T_INTERN (Echtzeit intern)	Der Wert des '**t**' Attributs wird ignoriert - **stVal** hat den Zeitstempel der lokalen PC-Zeit.
ClockNotSynchronized	T_INVAL (Zeit ungültig) und T_EXTERN (Echtzeit extern)	

Das TimeQuality Bit LeapSecondsKnown wird nicht T_INVAL zugeordnet.

Die Variablen des Datenobjekts ohne '**t**' Attribut (oder mit '**t**' Wert 0) und die Variablen mit anderen FC (auf Seite 80) als **ST/MX** erhalten den Zeitstempel der lokalen PC-Zeit.

Info

Sie können Statusbits (z. B. `T_INVAL`) auswerten, indem Sie Combi-Element oder Reaktionsmatrix (vom Typ Multi) verwenden und Sie können sie auch in CEL oder AML filtern.

FILETRANSFER

Für den Filetransfer sind drei Funktionen implementiert:

1. Ordnerinformation anfordern
2. Datei vom Server holen
3. Datei löschen

1. ORDNERINFORMATION ANFORDERN

Um Ordnerinformationen anzufordern:

1. erzeugen Sie in Ihrem zenon Projekt zwei Stringvariablen des `Treiberobjekttyps File Transfer` mit der entsprechenden Netzadresse:
   a) Die erste Variable, im weiteren Verlauf als Kommandovariable bezeichnet, mit dem Namen "**`SERVER!Command`**" kann auch dazu benutzt werden, um Dateien zu holen und zu löschen.
   b) Die zweite Variable mit dem Namen "**`SERVER!Directory`**" wird nur für das Ergebnis der Ordneranfrage benutzt. Sie empfängt den Ordnerinhalt als lesbaren Text. Deshalb sollte deren Größe der maximalen Größe des Dateinamen (inklusive des Odners) in der Steuerung entsprechen.
2. setzen Sie für die Kommandovariable den Wert "`DIR`" (für das Root-Directory) bzw. "`DIR <file_spec>`".
   <file_spec> ist ein String, der als Übergabeparameter an den Server gesendet wird.

Wenn der Ordner erfolgreich empfangen wurde:

- ändert die Kommandovariable ihren Wert auf "`DIR OK`"
- enthält die Ordnervariable den empfangenen Ordnerinhalt als lesbaren Text. Eine Zeile dieses Texts hat das folgende Format:

<Dateiname>;<Dateilänge>;<Zeitstempel>.

2. DATEI VOM SERVER HOLEN

Um eine Datei vom Server zu holen, setzen Sie bei der Kommandovariable den Wert "GET <Dateiname>".

Wenn die Datei erfolgreich empfangen wurde:

- wird sie in dem Ordner gespeichert, der als Directory for file transfer im Treiberdialog Basic settings (auf Seite 16) definiert wurde.
  Auch Dateien aus einem Unterordner werden im Hauptordner abgelegt.
- ändert die Kommandovariable ihren Wert auf "GET OK"

3. DATEI LÖSCHEN

Um eine Datei vom Server zu löschen, setzen Sie bei der Kommandovariablen den Wert "DEL <Dateiname>".

Wenn die Datei erfolgreich gelöscht wurde, ändert die Kommandovariable ihren Wert auf "DEL OK"

FEHLERBEHANDLUNG

Wenn bei der Ausführung des Dateientransfers Fehler auftreten, z. B. der Server antwortet negativ, ändert die Kommandovariable ihren Wert auf "XXX ERROR" (XXX = DIR, GET oder DEL) und der Treiber schreibt optional einen Eintrag in die Log-Datei (siehe auch Fehleranalyse (auf Seite 77)).

EINSCHRÄNKUNGEN

Es wird nur eine Array-Ebene unterstützt. Die Unterstützung von Arrays ist auf die Data-Attribute beschränkt. Das Beschreiben von Array-Elementen ist nicht möglich. Beim Lesen via Polling oder Reporting wird, wie bei allen anderen Data-Attributen auch, grundsätzlich immer das gesamte Data Object gelesen, das das Array-Attribut beinhaltet.

Die Informationen über unterstützte oder benötigte Dienste erhalten Sie im Anhang C (auf Seite 94).

9. Reporting

Das Reporting wurde durch den IEC 61850 Standard für spontane Kommunikation eingeführt. Die Variablen, die mit einem erfolgreich aktivierten Report verknüpft sind, werden nicht länger gepollt, sondern werden von der Steuerung gemeldet, wenn eine Wertänderung auftritt. Welche Variablen gemeldet werden, wenn ein Client einen spezielle Report Control Block (RCB) aktiviert, wird im DatSet (Data Set) Attribut des RCB gespeichert. Das DatSet Attribut ist ein sichtbarer String für den Namen des Datenobjekts, dessen Wertänderung vom IEC 61850 Server spontan gemeldet wird.

Der IEC850 Treiber unterstützt Buffered und Unbuffered Reporting mit in der Steuerung vordefinierten Data Sets und mit Dynamic Data Sets - mit DatSet Attribut, das vom Treiber während der Aktivierung des RCB dynamisch definiert wurde.

Wie vom IEC 61850 Standard vorgesehen, sind für einen Report Datenobjekte mit Functional Constraints (FC) verfügbar:

- Status (ST)
- Measurands (MX)

In einem Data Set:

- Datenobjekte mit unterschiedlichen FCs können zusammen verwendet werden, wenn der IEC 61850 Server dies unterstützt.
- Datenobjekte aus unterschiedlichen Logical Devices können enthalten sein.

Um die Aktivierung des Reporting zu ermöglichen, sollten sie folgenden Bedingungen erfüllt sein:

- Das Data Set und der Reporting Control Block, der das Data Set benutzt, müssen im selben Logical Device sein.
- Mindestens eine Variable des Logical Node, in dem der RCB inkludiert ist, muss im Projekt erstellt worden sein. Anderenfalls berücksichtigt der Treiber diesen Logical Node erst gar nicht. Ausnahme: LLN0. Der Logical Node LLN0 wird für jedes Logical Device automatisch berücksichtigt, wenn das LD wenigsten eine Variable hat, die in der Projekt-Variablenliste erstellt wurde.

> **Achtung**
>
> Wenn Sie die Data Sets in der Steuerung projektieren, dürfen Sie keine Datenattribute sondern nur ganze Datenobjekte hinzufügen. Andernfalls können Zeitstempel und Quality Informationen im Treiber nicht verarbeitet werden, da sie nicht von der Steuerung gesendet werden (siehe auch Standard 61850-7-2 14.2.3.2.2.9, Abbildung 23)
> Der Treiber aktiviert nur Reports, in denen Data Sets ganze Datenobjekte enthalten.

Die Basiskonfiguration des Reporting erfolgt in der Treiberkonfiguration - Server (auf Seite 20). Die konfigurierten Buffered Reports in der Treiberkonfiguration haben eine höhere Priorität als die Unbuffered Reports. Wenn die Aktivierung der Buffered Reports misslingt, wird der Treiber stattdessen die Unbuffered Reports aktivieren, die auch die Data Sets enthalten, die die benötigten Objekte enthalten. Wenn Sie nicht möchten, dass der Treiber das Unbuffered Reporting aktiviert, setzen Sie die Eigenschaft `max. auto used URCBs` in der Treiberkonfiguration auf 0.

Wenn der Treiber aus irgendeinem Grund die Reports nicht aktivieren kann, wird stattdessen Polling benutzt.

> **Info**
>
> *Bei der Aktivierung des Reporting schreibt der Treiber (optionale) Log-Einträge aus denen hervorgeht, welche Variablen werden nicht mehr gepollt, sondern wurden erforgreich im Reporting aktiviert (in Spalte 'general text').*

TRIGGER OPTIONS

Der Treiber unterstützt alle `Trigger Conditions`, die benutzt werden, um das Versenden von Report zu triggern. Die Trigger Options (TrgOps) definieren, welche Wertänderung einer Dateninstanz berichtet wird. Bei der Aktivierung des Reports versucht der Treiber, alle konfigurierten TrgOps zu aktivieren, ignoriert aber Schreibfehler. Wenn das Schreiben misslingt, werden die Trigger Options benutzt, die schon in der SCL Datei (oder vom IED Programm) des IEC61850 Server konfiguriert wurden. Sie können die einzelnen Trigger Options per Checkbox aktivieren/deaktivieren, wenn Sie die Checkbox `Use preconfigured TrgOps` deaktivieren. Diese Einstellung gilt für alle RCBs.

Der `TrigerConditions` Datenattributtyp, der gemäß dem Standard dem `MMS Bitstring(6)` zugeordnet wird, falls eine entsprechende *TrgOps[RP|BR] Variable beim Import angelegt wurde, entspricht in zenon per default dem Datentyp UDINT (motorola-format, Bit 0 höchstwertig in seinem Oktet). Zusätzlich ermöglicht der Treiber solche Variablen mit Datentyp STRING anstatt UDINT. Der Wert des STRING hat die Länge 6 und enthält nur die Zeichen "0" und "1" in der gleichen Reihenfolge wie die Bits im Standard definiert sind, z. B. der Wert als String "011111" (als UDINT=124=01111100b) bedeutet,

dass alle TriggerCondition-Bits außer Bit 0 - 'Reserved' gesetzt sind und der Wert "000010" (als UDINT=8=00001000b) - nur Bit 4 - 'intergity'.

9.1 Unbuffered Reporting

Der IEC850 Treiber unterstützt zwei Varianten der Unbuffered Reports:

1. automatische Zuweisung: alle Vorhandenen Reporting Control Blocks werden auf ihren Inhalt (Data Sets) überprüft. Reports welche die meisten Variablen des Projektes enthalten werden zuerst angemeldet;
2. statische Zuweisung: der Treiber verwendet die Zuweisung von Reporting Control Blocks zum Client Rechner wie es in Treiberkonfiguration - Client configuration (auf Seite 25) definiert ist.

AUTOMATISCHE ZUWEISUNG

Die automatische Zuweisung arbeitet mit vordefinierten Data Sets und vorkonfigurierten Zuweisungen von Data Sets zu Reporting Control Blocks in der Steuerung.

Während des Treiberstarts werden alle Data Sets, die im Server vorhanden sind, ausgelesen und gemäß ihrer Usability geordnet (Nummer der inkludierten Variablen des Projekts). Diese Sortierung gibt die Reihenfolge an, in der der Treiber versucht freie URCBs fürs Reporting zu aktivieren. Es werden höchstens so viele URCBs aktiviert wie in der Treiberkonfiguration - Server (auf Seite 20) in Eigenschaft max. auto used URCBs definiert sind. Alle Datenobjekte des Data Sets, die erfolgreich mit einem aktiven RCB verknüpft sind, werden nicht länger gepollt, sondern werden bei einer Wertänderung gemeldet.

REGELN FÜR AUFTRETENDE KONFLIKTE

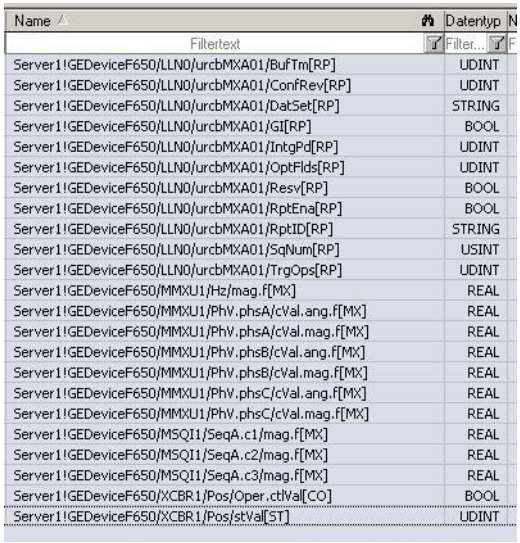
Bei der Aktivierung des Reporting, erkennt der Treiber auch Reports mit gleichem Inhalt; die Namen dieser Reports kann unterschiedlich sein. Bevor die RCBs gemäß ihrer Usability für das zenon Projekt sortiert werden, sortiert der Treiber zuerst die Datenobjekte der Data Sets von jedem RCB und vergleicht sie, um Dubletten zu verweiden. Das folgende gilt für Konflikte bei der Aktivierung des Reporting:

- Für gleiche Reports - mit identischer oder unterschiedlicher Reihenfolge der gleichen Datenobjekte: nur ein Report wird aktiviert.

- Wenn ein Report schon von einem anderen Client belegt ist, wird der nächste freie Report aktiviert.
- Wenn alle Reports schon von anderen Clients besetzt sind, werden die Datenpunkte gepollt.

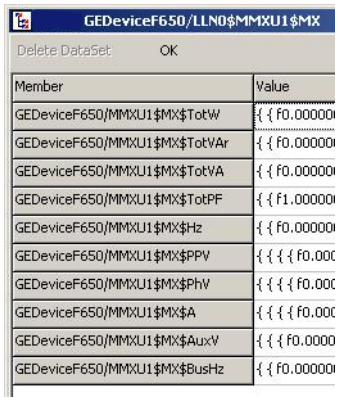
BEISPIEL FÜR AUTOMATISCHE ZUWEISUNG

Die folgenden Variablen sind in ihrem Projekt verfügbar



Name	Datentyp
Filtertext	Filter...
Server1!GEDeviceF650/LLN0/urcbMXA01/BufTm[RP]	UDINT
Server1!GEDeviceF650/LLN0/urcbMXA01/ConfRev[RP]	UDINT
Server1!GEDeviceF650/LLN0/urcbMXA01/DatSet[RP]	STRING
Server1!GEDeviceF650/LLN0/urcbMXA01/GI[RP]	BOOL
Server1!GEDeviceF650/LLN0/urcbMXA01/IntgPd[RP]	UDINT
Server1!GEDeviceF650/LLN0/urcbMXA01/OptFlds[RP]	UDINT
Server1!GEDeviceF650/LLN0/urcbMXA01/Resv[RP]	BOOL
Server1!GEDeviceF650/LLN0/urcbMXA01/RptEna[RP]	BOOL
Server1!GEDeviceF650/LLN0/urcbMXA01/RptID[RP]	STRING
Server1!GEDeviceF650/LLN0/urcbMXA01/SqNum[RP]	USINT
Server1!GEDeviceF650/LLN0/urcbMXA01/TrgOps[RP]	UDINT
Server1!GEDeviceF650/MMXU1/Hz/mag.f[MX]	REAL
Server1!GEDeviceF650/MMXU1/PhV.phsA/cVal.ang.f[MX]	REAL
Server1!GEDeviceF650/MMXU1/PhV.phsA/cVal.mag.f[MX]	REAL
Server1!GEDeviceF650/MMXU1/PhV.phsB/cVal.ang.f[MX]	REAL
Server1!GEDeviceF650/MMXU1/PhV.phsB/cVal.mag.f[MX]	REAL
Server1!GEDeviceF650/MMXU1/PhV.phsC/cVal.ang.f[MX]	REAL
Server1!GEDeviceF650/MMXU1/PhV.phsC/cVal.mag.f[MX]	REAL
Server1!GEDeviceF650/MSQI1/SeqA.c1/mag.f[MX]	REAL
Server1!GEDeviceF650/MSQI1/SeqA.c2/mag.f[MX]	REAL
Server1!GEDeviceF650/MSQI1/SeqA.c3/mag.f[MX]	REAL
Server1!GEDeviceF650/XCBR1/Pos/Oper.ctlVal[CO]	BOOL
Server1!GEDeviceF650/XCBR1/Pos/stVal[ST]	UDINT

Die */urcbMXA01/* Variablen haben rein informativen Zweck. Damit Reporting funktioniert, müssen die Variablen für die RCB Attribute nicht zwingend im zenon Projekt existieren. Die Steuerung enthält auch ein Data Set GEDeviceF650/LLN0$MMXU1$MX mit dem folgenden Inhalt.

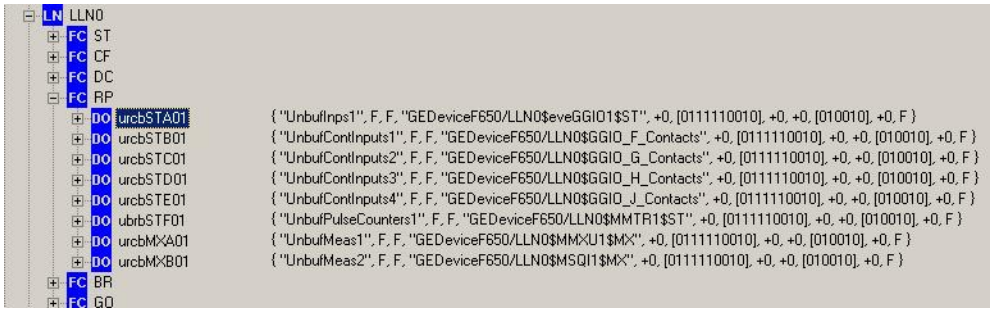


GEDeviceF650/LLN0$MMXU1$MX

Delete DataSet OK

Member	Value
GEDeviceF650/MMXU1MXTotW	{ { f0.00000
GEDeviceF650/MMXU1MXTotVAr	{ { f0.00000
GEDeviceF650/MMXU1MXTotVA	{ { f0.00000
GEDeviceF650/MMXU1MXTotPF	{ { f1.00000
GEDeviceF650/MMXU1MXHz	{ { f0.00000
GEDeviceF650/MMXU1MXPPV	{ { { { f0.000
GEDeviceF650/MMXU1MXPhV	{ { { { f0.000
GEDeviceF650/MMXU1MXA	{ { { { f0.000
GEDeviceF650/MMXU1MXAuxV	{ { { f0.0000
GEDeviceF650/MMXU1MXBusHz	{ { f0.00000

In der Steuerung ist dieses Data Set dem RCB LLN0RPurcbMXA01 zugewiesen. Dafür geben Sie den Name des Data Set im Attribut DatSet dieses RCBs ein.



Beim Treiberstart werden die RCBs LLN0RPurcbMXA01 und LLN0RPurcbMXB01 aktiviert (wenn sie noch frei sind).
Da es keine weiteren geeigneten RCBs gibt, die Variablen des Projekts in ihren Data Sets haben, bleiben die übrigen RCBs ungenutzt. Wenn das Projekt auch eine Variable enthält, die im Data Set GEDeviceF650/LLN0$eveGGIO1$ST inkludiert ist, würde das bedeuten, dass LLN0RPurcbSTA01 auch aktiviert wird.

STATISCHE ZUWEISUNG

Für statischer Zuweisung ist in der Treiberkonfiguration - statically assigned RCB (auf Seite 27) fest definiert, welche Reporting Control Blocks benutzt werden. Der Treiber versucht, alle gelisteten Reporting Control Blocks, unabhängig deren Inhalt, zu aktivieren.
Wenn in der Treiberkonfiguration auch ein Dynamic Data Set eingeben wurde, weißt der Treiber in der Steuerung diese Data Set dem RCB zu, indem er den Wert des DatSet Attribut dieses Blocks ändert bevor der Report aktiviert wird.
Wenn kein Dynamic Data Set eingegeben wurde, ist das in der Steuerung vordefiniert Data Set gültig (genauso wie bei der automatischen Zuweisung).

Wenn ein Block erfolgreich aktiviert wurde, werden alle Datenobjekte, die in seinem Data Set enthalten sind, nicht länger gepollt, sondern werden bei einer Wertänderung gemeldet.

9.2 Buffered Reporting

Bei Buffered Reports speichert ein IEC61850 Server die Änderungen in der Zeit, in der ein angeschlossener Client unerreichbar ist. Wenn der Client (der IEC850 Treiber) wieder online ist (z. B. Netzwerkverbindung wiederhergestellt), liest der Treiber alle Änderungen und verarbeitet sie, z. B. erzeugt entfallene Alarme.

Bei Buffered Reporting gibt es keine automatische Zuweisung. Die Zuweisung von BRCBs zum Treiber ist immer statisch:

- entweder über die Treiberkonfiguration so wie bei der statischen Zuweisung bei Unbuffered Reporting beschrieben;
- oder aus Kompatibilitätsgründen zu älteren Versionen des Treibers, über die `Report ID` der BRCBs die in der SCL Datei projektiert wurden (die Eigenschaft `Use Report-ID for RCB assignment` in der Treiberkonfiguration sollte aktiviert sein).

Wenn die Zuweisungen in der SCL Datei projektiert werden sollen, muss die zugewiesene Report ID (RptID) der BRCB einen String mit folgender Nomenklatur enthalten 'zenon_Computername_*'; z. B.: zenon_MyPC_Measurands1. Dann benutzt der Treiber das Data Set of RCB, das fest mit der SCL Datei verknüpft ist. Ein Buffered Report kann nur vom einen Host aktiviert werden. Um den Treiber davon abzuhalten alle Buffered Reports zu aktivieren und den Zugriff für andere IEC61850 Clients zu blockieren, müssen sie explizit in der Treiberkonfiguration (auf Seite 11) definiert werden.

Abgesehen davon ist das grundlegende Verhalten gleich dem bei Unbuffered Reporting.

BRCB BUFFER

- Der Treiber löscht automatisch den Inhalt des Puffers der BRCBs, wenn die erst Verbindung nach dem Start der Runtime hergestellt wurde.
- Wenn der Puffer erhalten bleiben soll,
  - müssen Sie in der Treiberkonfiguration die Option `Do not purge BRCB buffer at start` im Basic settings (auf Seite 16) aktivieren;
  - die zuletzt erhaltene Entry ID für das BRCB muss vorhanden sein.
- Wenn die Verbindung abbricht und wiederaufgebaut wird, ohne dass die Runtime neu gestartet wird, bleibt der Puffer erhalten und die Daten werden gesammelt.

Der Treiber speichert die zuletzt erhaltene Entry ID für jedes BRCB jede Minute oder wenn der Treiber geschlossen wird in einer Datei im Verzeichnis der Runtime-Daten.

BUFFERED REPORTS UND ZENON REDUNDANZ

Da zenon Redundanz unterstützt (siehe zenon Netzwerk) und ein BRCB nur einmal von einem PC aktiviert werden kann, müssen Sie für den Projekt-Sever und den Standby-Server getrennt festlegen, welcher PC welchen Report aktivieren soll. Der Treiber wird sowohl auf dem Server als auch auf dem

Standby gestartet und überprüft die Treiberkonfiguration (Eigenschaft `Hostname`), ob er eine Konfiguration findet, die dem Computernamen entspricht, auf dem die Runtime läuft.

Darum benötigen Sie in der Steuerung zwei Kopien des Buffered Report im Fall einer redundanten Konfiguration; eine für den Treiber, der auf dem Projekt-Server läuft und eine für den Treiber, der auf dem Standby-Server läuft. Und in der Treiberkonfiguration erstellen Sie zwei Einträge in der Liste RCB assignment (auf Seite 27). Einen mit dem Computernamen des Projekt-Servers und einen mit dem Computernamen des Standby-Servers. Beim Konfigurationseintrag des Projekt-Servers können Sie die erste Kopie des BRCB eintragen und beim Konfigurationseintrag des Standby-Servers die zweite Kopie.

Achtung

Auch wenn Sie keine Redundanz verwenden, müssen Sie in der Treiberkonfiguration (Eigenschaft `Hostname`)dennoch festlegen welche Reports für den PC verwendet werden, auf dem die zenon Runtime läuft.

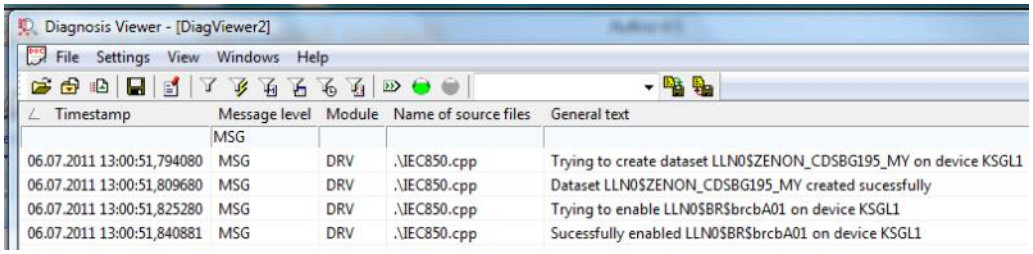
9.3 Dynamische Datasets

Um Dynamic Data Sets zu benutzen, müssen die beiden Dienste `CreateDataSet()` und `DeleteDataSet()` vom IEC 61850 Server unterstützt werden und das Attribut DatSet muss in dem Reporting Control Blocks schreibbar sein (SCL: ReportSettings.datSet="Dyn"). Beim Projektieren müssen Sie mögliche Einschränkungen des Servers berücksichtigen wie zum Beispiel maximale Anzahl von Elementen im Data Set (SCL: DynDataSet.max und maxAttributes).

Dynamic Data Sets listen Datenobjekte auf und werden im Dialog dynamic DataSet configuration (auf Seite 29) konfiguriert. Sie können für Unbuffered Reporting und für Buffered Reporting verwendet werden, aber nur, wenn in der Treiberkonfiguration - Client-Konfiguration (auf Seite 25) statische Zuweisungen mit RCBs projektiert wurden.

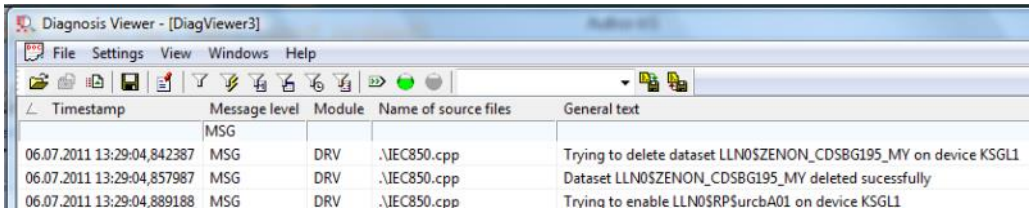
Wenn in der Treiberkonfiguration ein Dynamic Data Set für ein RCB eingeben wurde, weist der Treiber in der Steuerung dieses Data Set dem RCB zu, indem er den Wert des DatSet Attribut dieses Blocks ändert bevor der Report aktiviert wird.

Im Diagnosis Viewer (auf Seite 77) kann das Anlegen eines dynamischen Datasets überprüft werden:



Bevor die statisch zugewiesenen Reporting Control Blocks aktiviert sind, werden alle dynamischen Data Sets, die mit ihnen via Treiberkonfiguration - Client-Konfiguration (auf Seite 25) verknüpft sind, mit denen auf dem IEC 61850 Server verglichen. Alle beim Server verfügbaren Dynamic Data Sets, die nicht (mehr) benötigt werden, werden gelöscht. Alle beim Server nicht verfügbaren Dynamic Data Sets werden erstellt und alle Data Sets die auf beiden Seiten verfügbar sind, werden auf selben Inhalt überprüft. Wenn der Inhalt abweicht, werden sie gelöscht und neu erstellt. Der Treiber löscht aber nur die Data Sets die er selbst mal erstellt hat. Wenn der Treiber ein Data Set nicht löschen kann wird der zugewiesene RCB aktiviert mit bereits vorhandenen Data Set.

Im Diagnosis Viewer (auf Seite 77) kann das Löschen eines dynamischen Datasets überprüft werden:



Die Dynamic Data Sets, die in der Runtime erstellt werden (via `CreateDataSet()` Dienst) erhalten einen Client-abhängigen Namen am Server, um sie leicht wieder zu finden. Die Nomenklatur des Dynamic Data Set Namens ist wie folgt:

`LLN0$ZENON_hostname_datasetname`. Hierbei:

`Hostname` ist der Computername des Clients so wie er in der Treiberkonfiguration definiert ist - Client-Konfiguration (auf Seite 25) unter `Hostname` und
`datasetname` ist der Name, der in der Treiberkonfiguration definiert wurde - Dynamic DataSet configuration (auf Seite 27).

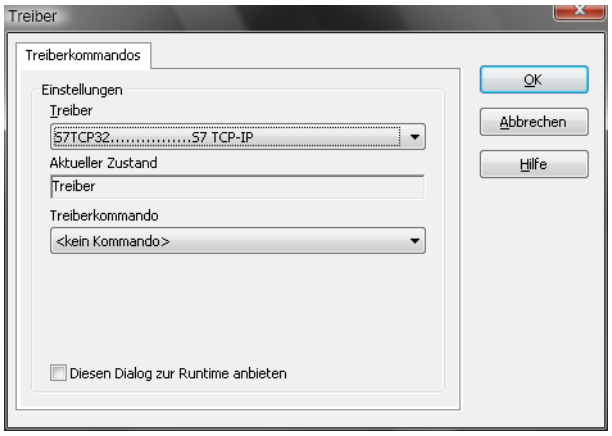
Wenn kein Dynamic Data Set in der RCB Konfiguration des Treibers eingegeben wurde, ist das in der Steuerung vordefiniert Data Set gültig.

10. Treiberkommandos

Dieses Kapitel beschreibt Standardfunktionalitäten, die für die meisten zenon Treiber gültig sind. Nicht alle hier beschriebenen Funktionalitäten stehen für jeden Treiber zur Verfügung. Zum Beispiel enthält ein Treiber, der laut Datenblatt keine Modemverbindung unterstützt, auch keine Modem-Funktionalitäten.

Treiberkommandos dienen dazu, Treiber über zenon zu beeinflussen, z. B. starten und stoppen. Die Projektierung erfolgt über die Funktion **`Treiber Kommandos`**. Dazu:

- legen Sie eine neue Funktion an
- wählen Sie Variablen -> Treiberkommandos
- der Dialog zur Konfiguration wird geöffnet



Parameter	Beschreibung
Treiber	Dropdownliste mit allen im Projekt geladenen Treibern.
Aktueller Zustand	Fixer Eintrag, in aktuellen Versionen ohne Funktion.
Treiberkommando	Dropdownliste zur Auswahl des Kommandos.
▸ Treiber starten (Online-Modus)	Treiber wird neu initialisiert und gestartet.
▸ Treiber stoppen (Offline-Modus)	Treiber wird angehalten, es werden keine neuen Daten angenommen. Hinweis: Ist der Treiber im Offline-Modus, erhalten alle Variablen, die für diesem Treiber angelegt wurden, den Status Abgeschaltet (OFF; Bit 20).

Treiber in Simulationsmodus	Treiber wird in den Simulationsmodus gesetzt. Die Werte aller Variablen des Treibers werden vom Treiber simuliert. Es werden keine Werte von der angeschlossenen Hardware (z. B. SPS, Bussystem, ...) angezeigt.
Treiber in Hardwaremodus	Treiber wird in den Hardwaremodus gesetzt. Für die Variablen des Treibers werden die Werte von der angeschlossenen Hardware (z. B. SPS, Bussystem, ...) angezeigt.
Treiberspezifisches Kommando	Eingabe treiberspezifischer Kommandos. Öffnet Eingabefeld für die Eingabe eines Kommandos.
Treiber Sollwertsetzen aktivieren	Sollwert setzen auf Treiber ist erlaubt.
Treiber Sollwertsetzen deaktivieren	Sollwert setzen auf Treiber wird verhindert.
Verbindung mit Modem aufbauen	Verbindung aufbauen (für Modem-Treiber). Öffnet Eingabefelder für Hardware-Adresse und Eingabe der zu wählenden Nummer.
Verbindung mit Modem trennen	Verbindung beenden (für Modem-Treiber).
Diesen Dialog zur Runtime anbieten	Dialog wird zur Runtime für Änderungen angeboten.

TREIBERKOMMANDOS IM NETZWERK

Wenn sich der Rechner, auf dem die Funktion **Treiberkommandos** ausgeführt wird, im zenon Netzwerk befindet, werden zusätzliche Aktionen ausgeführt. Ein spezielles Netzwerkkommando wird vom Rechner zum Server des Projekts gesendet, der dann die gewünschte Aktion auf seinem Treiber durchführt. Zusätzlich sendet der Server das gleiche Treiberkommando zum Standby des Projekts. Der Standby führt die Aktion auch auf seinem Treiber aus.

Dadurch ist gewährleistet, dass Server und Standby synchronisiert sind. Dies funktioniert nur, wenn Server und Standby jeweils eine funktionierende und unabhängige Verbindung zur Hardware haben.

11. Fehleranalyse

Sollte es zu Kommunikationsproblemen kommen, bietet dieses Kapitel Hilfen, um den Fehler zu finden.

11.1 Analysetool

Alle zenon Module wie z. B. Editor, Runtime, Treiber, usw. schreiben Meldungen in eine gemeinsame Log-Datei. Um sie korrekt und übersichtlich anzuzeigen, benutzen Sie das Programm Diagnose Viewer (main.chm::/12464.htm), das mit zenon mitinstalliert wird. Sie finden es unter Start/Alle Programme/zenon/Tools 7.10 -> Diagviewer.

zenon Treiber protokollieren alle Fehler in Log-Dateien. Der Standardordner für die Log-Dateien ist der Ordner **LOG** unterhalb des Ordners `ProgramData`, zum Beispiel: `C:\ProgramData\zenon\zenon7.10\LOG` für die zenon Version 7.10. Log-Dateien sind Textdateien mit einer speziellen Struktur.

Achtung: Mit den Standardeinstellungen zeichnet ein Treiber nur Fehlerinformationen auf. Mit dem **Diagnose Viewer** kann bei den meisten Treibern die Diagnose-Ebene auf „Debug" und „Deep Debug" erweitert werden. Damit protokolliert der Treiber auch alle anderen wesentlichen Aufgaben und Ereignisse.

Im Diagnose Viewer kann man auch:

- eben erstellte Einträge live mitverfolgen
- die Aufzeichnungseinstellungen anpassen
- den Ordner, in dem die Log-Dateien gespeichert werden, ändern

Hinweise:

1. Unter Windows CE werden aus Ressourcegründen auch Fehler standardmäßig nicht protokolliert.
2. Der Diagnose Viewer zeigt alle Einträge in UTC (Koordinierter Weltzeit) an und nicht in der lokalen Zeit.
3. Der Diagnose Viewer zeigt in seiner Standardeinstellung nicht alle Spalten einer Log-Datei an. Um mehr Spalten anzuzeigen, aktivieren Sie die Eigenschaft **Add all columns with entry** im Kontextmenü der Spaltentitel.

4. Bei Verwendung von reinem **Error-Logging** befindet sich eine Problembeschreibung in der Spalte **Error text**. In anderen Diagnose-Ebenen befindet sich diese Beschreibung in der Spalte **General text**.
5. Viele Treiber zeichnen bei Kommunikationsprobleme auch Fehlernummern auf, die die SPS ihnen zuweist. Diese werden in **Error text** und/oder **Error code** und/oder **Driver error parameter (1** und **2)** angezeigt. Hinweise zur Bedeutung der Fehlercodes erhalten Sie in der Treiberdokumentation und der Protokoll/SPS-Beschreibung.
6. Stellen Sie am Ende Ihrer Tests den Diagnose-Level von **Debug** oder **Deep Debug** wieder zurück. Bei **Debug** und **Deep Debug** fallen beim Protokollieren sehr viele Daten an, die auf der Festplatte gespeichert werden und die Leistung Ihres Systems beeinflussen können. Diese werden auch nach dem Schließen des **Diagnose Viewers** weiter aufgezeichnet.

Info

Weitere Informationen zum Diagnose Viewer finden Sie im Kapitel Diagnose Viewer (main.chm::/12464.htm).

11.2 Checkliste

Überprüfen Sie bei Fehlern Ihr System anhand der Treiberdokumentation auf korrekte Treiberkonfiguration, Adressierung der Variablen in zenon Projekten, Reporteinstellungen in der Steuerung etc.

Überprüfen Sie vor allem:

- Ist das Gerät (IEC 61850 Server) für die Kommunikation mit dem Stromnetz verbunden und bereit für den Einsatz?
  - Kann eine TCP-Verbindung zum Gerät aufgebaut werden (überprüfen Sie das via 'ping') über den definierten Port (überprüfen Sie das via Telnet)?

- Ist eine Kommunikation zwischen diesem Gerät und einem anderen IEC 61850 Client möglich?
- Befindet sich der Treiber im Hardwaremodus und nicht im Simulationsmodus?

- Haben Sie die Fehler, die in der Log-Datei der Treiber protokolliert werden, ausgewertet?
  - Welche Fehler sind aufgetreten?
  - Wurde das Diagnose-Logging: für den richtigen Treiber, wenigstens für das Error-Level und für alle Treibermodule aktiviert?
  - Haben Sie berücksichtigt, dass die Zeiten im Diagnose-Viewer immer UTC sind?
  - Ist der Diagnose-Viewer so eingestellt, dass alle registrierten Einträge und alle Spalten mit dem Inhalt (Filtern) dargestellt werden?
- Variablenadressierung:
  - Haben alle Variablen die richtigen Netzadressen?
  - Sind die Namen (oder Kennungen) der Variablen gültig (SERVER!LD/LN/DO/DA[FC])?
  - Haben die Variablen den richtigen Treiberobjekttyp, Datentyp und Wertebereich?
- Allgemein und Reporting:
  - Wird wenigstens eine Variable vom Logical Device gepollt?
  - Beinhaltet die RCB die ganzen Datenobjekte (nicht nur die Datenattribute */stVal aber auch */q und */t)?
  - Hat die Eigenschaft `max. auto used URCBs` den richtigen Wert?
  - Ist im Gerät Reporting verfügbar?
  - Wenn in der SCL-Datei definiert: enthält die Report ID des Buffered RCB den String "zenon_Computername*"?
  - Für Dynamic Data Set: ist das DataSet-Attribut im Reporting Control Block beschreibbar?
  - Haben Sie Einschränkungen des Geräts wie zum Beispiel die maximale Anzahl der Elemente im Data Set etc. berücksichtigt?

12. Anhang A - Beschreibung der Functional Constraints (FCs):

FC	Semantics	Beschreibung
ST	Status information	steht für eine Statusinformation, deren Wert gelesen, ersetzt, berichtet und geloggt aber nicht geschrieben werden kann.
MX	Measurands (analogue values)	steht für eine Messgröße, deren Wert gelesen, ersetzt, berichtet und geloggt aber nicht geschrieben werden kann.
CO	Control	represents a control information whose value may be operated (control model) and read.
SP	Setpoint	represent a set-point information whose value may be controlled (control model) and read.
SV	Substitution	represents a substitution information whose value may be written to substitute the value attribute and read.
CF	Konfiguration	represents a configuration information whose value may be written and read.
DC	Beschreibung	represents a description information whose value may be written and read
EX	Extended definition	represents an extension information providing a reference to a name space. Extensions are used in conjunction with extended definitions of LNs, DATA, and DataAttributes in 61850-7-3 and IEC 61850- 7-4. Value may be read.

13. Anhang B - Abkürzungen für data object/data attribute

Term	Beschreibung
f	floating point (analog value)
i	integer (analog value)
d	Visible String(255)
q	Quality
t	Timestamp

A	Current
Acs	Access
ACSI	Abstract Communication Service Interface
Acu	Acoustic
Age	Ageing
Alm	Alarm
Amp	Current non phase related
To	Analogue
Ang	Angle
Auth	Authorization
Auto	Automatic
Aux	Auxiliary
Av	Average
B	Bushing
Bat	Battery
Beh	Behaviour
Bin	Binary
Blk	Block, blocked
Bnd	Band
Bo	Bottom
Cancel	Cancel
Cap	Capability
Capac	Capacitance
Car	Carrier

CB	Circuit Breaker
CDC	Common Data Class
CE	Cooling Equipment
Cf	Crest factor
Cfg	Konfiguration
CG	Core Ground
Ch	Channel
Cha	Charger
Chg	Change
Chk	Check
Chr	Characteristic
Cir	Circulating
Clc	Calculate
Clk	Clock, clockwise
Cls	Close
Cnt	Counter
Col	Coil
Cor	Correction
Crd	Coordination
Crv	Curve
CT	Current Transducer
Ctl	Control
Ctr	Center
Cyc	Cycle

Dea	Dead
Den	Density
Det	Detected
DExt	De-excitation
Diag	Diagnostics
Dif	Differential, difference
Dir	Richtung
Dis	Distance
Dl	Delay
Dlt	Delete
Dmd	Demand
Dn	Down
DPCSO	Double point controllable status output
DQ0	Direct, Quadrature, and zero axis quantities
Drag	Drag hand
Drv	Drive
DS	Device State
Dsch	Discharge
Dur	Duration

EC	Earth Coil
EE	External Equipment
EF	Earth Fault
Ena	Enabled
Eq	Equalization, Equal
Ev	Evaluation
Ex	External
Exc	Exceeded
Excl	Exclusion
Ext	Excitation
FA	Fault Arc
Fact	Factor
Fan	Fan
Flt	Fault
Flw	Flow
FPF	Forward Power Flow
Fu	Fuse
Fwd	Forward
Gen	General
Gn	Generator
Gnd	Ground
Gr	Group
Grd	Guard
Gri	Grid

H	Harmonics (phase related)
H2	Hydrogen
H2O	Water
Ha	Harmonics (non phase related)
Hi	High, highest
HP	Hot point
Hz	Frequency
IEEE	Institute of Electrical and Electronic Engineers
Imb	Imbalance
Imp	Impedance non phase related
In	Input
Ina	Inactivity
Incr	Increment
Ind	Indication
Inh	Inhibit
Ins	Insulation
Int	Integer
ISCSO	Integer status controllable status output

Km	Kilometer
L	Lower
LD	Logical Device
LDC	Line Drop Compensation
LDCR	Line Drop Compensation Resistance
LDCX	Line Drop Compensation Reactance
LDCZ	Line Drop Compensation Impedance
LED	Light Emitting Diode
Len	Length
Lev	Level
Lg	Lag
Lim	Limit
Lin	Line
Liv	Live
Lo	Low
LO	Lockout
Loc	Local
Lod	Load, loading
Lok	Locked
Los	Loss
Lst	List
LTC	Load Tap Changer

M	minutes
M/O	Data Object is Mandatory or Optional
Max	Maximum
Mem	Memory
Min	Minimum
Mod	Mode
Mot	Motor
Ms	Milliseconds
Mst	Moisture
MT	Main Tank
N	Neutral
Nam	Name
Net	Net sum
Ng	Negative
Nom	Nominal, Normalizing
Num	Number
Ofs	Offset
Op	Operate, Operating
Oper	Operate*
Opn	Open
Out	Output
Ov	Over, Override, Overflow
Pa	Partial

Par	Parallel
Pct	Percent
Per	Periodic
PF	Power Factor
Ph	Phase
Phy	Physical
Pls	Pulse
Plt	Plate
Pmp	Pump
Po	Polar
Pol	Polarizing
Pos	Position
POW	Point on wave switching
PP	Phase to phase
PPV	Phase to phase voltage
Pres	Pressure
Prg	Progress, in progress
Pri	Primary
Pro	Protection
Ps	Positive
Pst	Post
Pwr	Power
Qty	Quantity
R	Raise

R0	Zero sequence resistance
R1	Positive sequence resistance
Rat	Winding ratio
Rcd	Record, recording
Rch	Reach
Rcl	Reclaim
Re	Retry
React	Reactance; Reactive
Rec	Reclose
Red	Reduction
Rel	Release
Rem	Remote
Res	Residual
Ris	Resistance
Rl	Relation, relative
Rms	Root mean square
Rot	Rotation, Rotor
Rs	Reset, Resetable
Rsl	Result
Rst	Restraint
Rsv	Reserve
Rte	Rate
Rtg	Rating
Rv	Reverse
Rx	Receive, received

S1	Step one
S2	Step two
SBO	Select before operate*
SBOw	Select with value*
Sch	Scheme
SCO	Supply change over
SCSM	Specific Communication Service Mapping
Sec	Security
Seq	Sequence
Set	Setting
Sh	Shunt
Spd	Speed
SPl	Single Pole
SPCSO	Single point controllable status output
Src	Source
St	Status
Stat	Statistics
Stop	Stop
Std	Standard
Str	Start
Sup	Supply
Svc	Service
Sw	Switch
Swg	Swing
Syn	Synchronisation

Tap	Tap
Td	Total distortion
Tdf	Transformer derating factor
Test	Test
Thd	Total Harmonic Distortion
Thm	Thermal
TiF	Telephone influence factor
Tm	Time
	• Tmh = Time in h
	• Tmm = Time in min
	• Tms = Time in s
	• Tmms = Time in ms
Tmp	Temperature (°C)
To	Top
Tot	Total
TP	Three pole
Tr	Trip
Trg	Trigger
Ts	Total signed
Tu	Total unsigned
Tx	Transmit, transmitted
Typ	Type
Un	Under

V	Voltage
VA	Volt Amperes
Vac	Vacuum
Val	Value
VAr	Volt Amperes Reactive
Vlv	Valve
Vol	Voltage non phase related
VT	Voltage Transducer
W	Active Power
Wac	Watchdog
Watt	Active Power non phase related
Wei	Weak End Infeed
Wh	Watt hours
Wid	Width
Win	Window
Wrm	Warm
X0	Zero sequence reactance
X1	Positive sequence reactance

Z	Impedance
Z0	Zero sequence impedance
Z1	Positive sequence impedance
Zer	Zero
Zn	Zone
Zro	Zero sequence method

* SBO,SBOw,Oper, und Cancel sind in IEC 61850-8-1 definiert:

Data attribute name	Semantics
SBO	Select – returns ACSI name of control
SBOw	SelectWithValue – receives service parameters
Oper	Operate – receives service parameters and control values
Cancel	Cancel – receives service parameters and control values

14. Anhang C - Conformance statement

Das Confirmance Statement des iec850 Treibers zu dem Protokoll IEC61850 liegt nur in englischer Sprache vor.

14.1 Protocol implementation conformance statement (PICS)

A.1 - BASIC ACSI CONFORMANCE STATEMENT

	Client / subscriber	Server / publisher	Value / comments

Client-Server roles				
B11	Server side (of TWO-PARTY-APPLICATION-ASSOCIATION)	-	N	for Server see zenon Logic
B12	Client side of (TWO-PARTY-APPLICATION-ASSOCIATION)	Y	-	
SCSMs supported				
B21	SCSM: IEC 61850-8-1 used	Y		
B22	SCSM: IEC 61850-9-1 used			not supported
B23	SCSM: IEC 61850-9-2 used			not supported
B24	SCSM: other			not supported
Generic substation event model (GSE)				
B31	Publisher side	-		
B32	Subscriber side	N		
Transmission of sampled value model (SVC)				
B41	Publisher side	-		
B42	Subscriber side	N		
- = not applicable Y = supported N or empty = not supported				

A.2 - ACSI MODELS CONFORMANCE STATEMENT

	Client / subscriber	Server / publisher	Value / comments

If Server or Client side (B11/12) is supported				
M1	Logisches Gerät	Y		
M2	Logical node	Y		
M3	Data	Y		
M4	Data set	Y		
M5	Substitution	Y		
M6	Setting group control	Y		not tested
Reporting				
M7	Buffered report control	Y		
M7-1	sequence-number	Y		
M7-2	report-time-stamp	Y		
M7-3	reason-for-inclusion	Y		
M7-4	data-set-name	Y		
M7-5	data-reference	Y		
M7-6	buffer-overflow	Y		
M7-7	entryID	Y		
M7-8	BufIm	Y		
M7-9	IntgPd	Y		
M7-10	GI	Y		
M8	Unbuffered report control	Y		
M8-1	sequence-number	Y		
M8-2	report-time-stamp	Y		
M8-3	reason-for-inclusion	Y		
M8-4	data-set-name	Y		
M8-5	data-reference	Y		
M8-6	BufIm	Y		
M8-7	IntgPd	Y		

M8-8	GI	Y		
Logging				
M9	Log control	N		
M9-1	IntgPd	N		
M10	Log	N		
M11	Control	N		
If GSE (B31/B23) is supported				
M12	GOOSE	N		
M13	GSSE	N		
If SVC (B41/B42) is supported				
M14	Multicast SVC	N		
M15	Unicast SVC	N		
If Server or Client side (B11/12) is supported				
M16	Time	N		Time of Operating System must be available with required accuracy
M17	File Transfer	Y		
Y = service is supported N or empty = service is not supported				

A.3 - ACSI SERVICE CONFORMANCE STATEMENT

ACSI Service Conformance		AA: TP/MC	Client (C)	Server (S)	Comments
Server					
S1	GetServerDirectory	TP	Y		
Application association					

S2	Associate		Y		
S3	Abort		Y		supported on TCP/IP
S4	Release		N		
Logical device					
S5	GetLogicalDeviceDirectory	TP	Y		
Logical node					
S6	GetLogicalNodeDirectory	TP	Y		
S7	GetAllDataValues	TP	N		not supported
Data					
S8	GetDataValues	TP	Y		
S9	SetDataValues	TP	Y		
S10	GetDataDirectory	TP	Y		
S11	GetDataDefinition	TP	Y		= GetDataDirectory
Data set					
S12	GetDataSetValues	TP	N		not supported
S13	SetDataSetValues	TP	N		not supported
S14	CreateDataSet	TP	Y		
S15	DeleteDataSet	TP	Y		
S16	GetDataSetDirectory	TP	Y		
Substitution					
S17	SetDataValues	TP	Y		
Setting group control					
S18	SelectActiveSG	TP	Y		not tested
S19	SelectEditSG	TP	Y		not tested
S20	SetSGValues	TP	Y		not tested

S21	ConfirmEditSGValues	TP	Y		not tested
S22	GetSGValues	TP	Y		not tested
S23	GetSGCBValues	TP	Y		not tested
Reporting					
Buffered report control block (BRCB)					
S24	Report	TP	Y		
S24-1	data-change (dchg)		Y		
S24-2	quality-change (qchg)		Y		
S24-3	data-update (dupd)		Y		
S25	GetBRCBValues	TP	Y		
S26	SetBRCBValues	TP	Y		
Unbuffered report control block (URCB)					
S27	Report	TP	Y		
S27-1	data-change (dchg)		Y		
S27-2	quality-change (qchg)		Y		
S27-3	data-update (dupd)		Y		
S28	GetURCBValues	TP	Y		
S29	SetURBCValues	TP	Y		
Logging					
Log control block					
S30	GetLCBValues	TP	Y		not tested
S31	SetLCBValues	TP	Y		not tested
Log					
S32	QueryLogByTime	TP	N		not supported
S33	QueryLogAfter	TP	N		not supported
S34	GetLogStatusValues	TP	Y		not tested

ACSI Service Conformance		AA: TP/MC	Client/ subscriber	Server/ publisher	Comments
Generic substation event model (GSE)					
GOOSE-CONTROL-BLOCK					
S35	SendGOOSEMessage	MC	C8	C8	not supported
S36	GetGoReference	TP	O	C9	not supported
S37	GetGOOSEElementNumber	TP	O	C9	not supported
S38	GetGoCBValues	TP	Y	O	
S39	SetGoCBValues	TP	Y	O	
GSSE-CONTROL-BLOCK					
S40	SendGSSEMessage	MC	C8	C8	not supported
S41	GetGsReference	TP	O	C9	not supported
S42	GetGSSEDataOffset	TP	O	C9	not supported
S43	GetGsCBValues	TP	Y	O	not tested
S44	SetGsCBValues	TP	Y	O	not tested
C-8 shall declare support for at least one (SendGOOSEMessage or SendGSSEMessage) C-9 shall declare support if TP association is available					
Transmission of sampled value model (SVC)					
Multicast SVC					
S45	SendMSVMessage	MC	N	C10	not supported
S46	GetMSVCBValues	TP	N	O	not supported
S47	SetMSVCBValues	TP	N	O	not supported
Unicast SVC					
S48	SendUSVMessage	TP	N	C10	not supported
S49	GetUSVCBValues	TP	N	O	not supported
S50	SetUSVCBValues	TP	N	O	not supported
C10 – shall declare support for at least one (SendMSVMessage or SendUSVMessage)					

ACSI Service Conformance		AA: TP/MC	Client (C)	Server (S)	Comments
Control					
S51	Select		Y		
S52	SelectWithValue	TP	Y		
S53	Cancel	TP	Y		
S54	Operate	TP	Y		
S55	Command-Termination	TP	Y		
S56	TimeActivated-Operate	TP	N		not supported
File Transfer					
S57	GetFile	TP	Y		
S58	SetFile	TP	N		not supported
S59	DeleteFile	TP	Y		
S60	GetFileAttributeValues	TP	Y		

ACSI Service Conformance		Client (C)	Server (S)	Comments
Time				
T1	Time resolution of internal clock	Operating System		
T2	Time accuracy of internal clock	Operating System	T1	T0 (10ms), T1 (1ms), T2 (100µs), T3 (25µs), T4 (4µs), T5 (1µs)
T3	Supported TimeStamp resolution	T1 (1ms)		or LREAL decimal place when t-attribute created as variable (T3)

14.2 Model implementation conformance statement (MICS)

All logical nodes, common data classes and data attribute types - as defined in IEC 61850-7-3 and IEC 61850-7-4 (Standard Edition 1) - are supported.

Introduced by Edition 2 for common data class APC the data attribute ctlVal as AnalogValue (structure) with FC=CO is supported in read-only Services; ctlVal[SP] in APC from Edition1 is supported in all Services.

15. Anhang D - PIXIT for COPA-DATA IEC61850 Client

> **Info**
>
> *Das PIXIT Dokument zum Zertifizierungstest des IEC850 Treibers liegt nur in englischer Sprache vor.*
>
> *PIXIT basiert auf einer von der Firma KEMA gelieferten Vorlage.*

INTRODUCTION

This document specifies the protocol implementation extra information for testing (PIXIT) of the IEC 61850 interface in the client system: “COPA-DATA IEC850 - Driver for IEC61850 protocol”, further referred to as “Client”.

Together with the PICS and the MICS the PIXIT forms the basis for a conformance test according to IEC 61850-10.

The following chapters specify the PIXIT for each applicable ACSI service model as structured in IEC 61850-10 and the “Conformance Test Procedures for Client System with IEC 61850-8-1 interface” by KEMA.

PIXIT FOR CONFIGURATION

Description	Value / Clarification
Describe how the client handles nameplate configuration revision mismatches	The Client reads out data model at each associate and therefore does not need to check configuration revision.
Describe how the client handles report control block configuration revision mismatches	RCB configuration is read out at each associate.
<additional items>	The Client does not use items of pre-defined data model until reading them from data model from an online server. The SCL file is used only by offline import of variables to HMI project in the Editor for pre-configuration. Please refer Client documentation for more details.

PIXIT FOR ASSOCIATION MODEL

Description	Value / Clarification
Garanteed number of servers that can set-up an association simultaneously (one association per server)	Per design: 65535 Tested: 50
Lost connection detection time range (default range of TCP_KEEPALIVE is 1 – 20 seconds)	The current TCP/IP timeout of Operating System The connection loss to LD is only detected if at least one its attribute is currently polled. Please refer Client documentation for more details
Lost (abort) connection retry time	30 seconds
Is authentication supported	Y
What is the maximum and minimum MMS PDU size	Max MMS PDU size 65535 Min MMS PDU size dependent of max. DO
What is the typical startup time after a power supply interrupt	The Client works only together with the zenon Runtime and both are Windows applications. 1. Windows does not guarantee the files will be not corrupted after power failure. Please use an UPS to prevent loss of data stored on HDD. 2. There is no typical start-up time for the Client

	as the start-up time of the Runtime depends of HMI project design and PC performance. In practice, the Client opens TCP/IP session to a server after few (1-4) seconds since application start.

PIXIT FOR SERVER MODEL

Description	Value / Clarification
Maximum object identification length	LD: 128 chars/octets LN: 16 chars/octets DO: 128 chars/octets DA: 128 chars/octets FC: 2 chars/octets
Does client support autodescription	In the Runtime the Client reads data model only via MMS. Then the Client does not uses data not responded via MMS by association. In the HMI Editor (by pre-configuration of the HMI project) the Client can read the data model from SCL file (off-line import) or from the server via MMS (on-line import) or data can be configured manually (variable creation). Please refer HMI documentation for more details.
What analogue value (MX) quality bits are used in the client	Y Good, Y Invalid, Y Reserved, Y Questionable Y Overflow Y OutofRange Y BadReference Y Oscillatory Y Failure Y OldData Y Inconsistent Y Inaccurate Y Process Y Substituted Y Test Y OperatorBlocked
Which status value (ST) quality bits are used in the client	Y Good, Y Invalid, Y Reserved,

	Y Questionable Y BadReference Y Oscillatory Y Failure Y OldData Y Inconsistent Y Inaccurate Y Process Y Substituted Y Test Y OperatorBlocked
Describe how to view/display quality values	1.zenon Status Bits by stVal variable in HMI (only selected bits. Please refer Client documentation for more details.), 2.the value of */q variable in HMI, if created
Describe how to force a SetDataValues request	Set value to the variable created in HMI. Please refer HMI documentation for more details.
Describe how to force a GetAllDataValues request	not used
Describe how the client behaves in case of: ▸ GetDataDefinition response- ▸ GetLogicalDeviceDirectory response- ▸ GetAllDataValues response- ▸ GetDataValues response- ▸ SetDataValues response-	▸ I-Bit* + log error entry ▸ I-Bit* + log error entry ▸ not supported ▸ I-Bit* + log error entry ▸ log error entry *Status bit INVALID of variables mirroring DO/DA. Please refer HMI documentation for more details.

PIXIT FOR DATA SET MODEL

Description	Value / Clarification
Describe how to force a GetDataSetValues request	not used

Describe how to force a SetDataSetValues request	not used
Describe how to force a DeletaDataSet request	would be done automatically according Client configuration for Dynamic Data Sets. Please refer Client documentation for more details.
Describe how the client handles following dataset mismatches between the SCL and the data sets exposed via MMS: 1. new dataset element 2. missing dataset element 3. Reordered dataset members in a dataset of a different data type 4. Reordered dataset members in a dataset of the same data type	No mismatch possible - SCL file is not used in the Client during communication; the Client only uses data sets exposed via MMS.
Describe how the client behaves in case of: ▸ GetLogicalNodeDirectory(DATA-SET) response- ▸ GetDataSetDirectory response-	log entry + data will be polled (no reporting)
Does the client support the creation of: ▸ persistent datasets ▸ non-persisten datasets	 ▸ Y ▸ N
Describe how the client behaves in case of: ▸ CreateDataSetDirectory response- ▸ DeleteDataSet response-	 ▸ log entry + data will be polled (not reported) ▸ log entry + old dataset used for reporting

PIXIT FOR SUBSTITUTION MODEL

Description	Value / Clarification
Describe how to substitute a value	For attributes with FC = SV (*/subEna, subQ, subID and e.g. subVal) create variables in the HMI project. Then first set values to be substituted (*/subVal, subQ, subID) and then set value TRUE to variable */subEna.

	Please refer HMI documentation for more details about setting variable values
<additional items>	There is a second method to make substitution only in HMI - e.g. when an IEC 61850 server does not support substitution. Any variable created in the HMI project can be turn to alternative value and that makes the HMI to simulate it. Please refer description of status bit ALT_VAL in the HMI manual for more details.

PIXIT FOR SETTING GROUP CONTROL MODEL

Description	Value / Clarification
Describe how to change the active setting group	set value of adequate variable in HMI project
Describe how to get the actual setting group values	create variables in the HMI project for attributes with these values
Describe how to edit setting group values	set value of adequate variables in HMI project
Describe how the client behaves in case of: ▸ GetSGCBValues response- ▸ The configured SG is different then the actual setting group	 ▸ log entry ▸ not tested
<additional items>	The Client was never tested with an IED supporting Setting Group

PIXIT FOR REPORTING MODEL

Description	Value / Clarification
Does the client search for RCB in all logical nodes? when not specify the logical nodes	Y All logical nodes* or Y The following logical nodes**: ▸ LLN0 ▸ RCBs assigned in the Client configuration ▸ BRCBs with RptID=zenon_computername_* * for automatic allocation of URCBs ** for static allocation of URCBs and BRCBs

	Please refer Client documentation for more details.
Which dynamic RCB attributes are/can be configured by the client	N RptID Y DataSet Y (fix)* Optional fields Y Trigger conditions N Buffer time N Integrity period *are pre-configured; cannot be changed dynamically. For attributes with FC = BR|RP variables in the HMI project can be created, read, written.
Does the client supports IED's with indexed and non-indexed report control blocks (RCB)	Y Buffered RCB indexed Y Buffered RCB not indexed Y Unbuffered RCB indexed Y Unbuffered RCB not indexed
The supported trigger conditions are	Y integrity Y data change Y quality change Y data update Y general interrogation
The minimum required optional fields are	N sequence-number N report-time-stamp N reason-for-inclusion Y* data-set-name N data-reference N buffer-overflow Y* entryID N conf-rev The Client can receive reports with any additional optional field, not only the minimum; the optional fields not required are then ignored.
Does the client support segmented reports	Y
Does the client support pre-assigned RCB	Y
Does the client support reported data set containing structured data objects or data attributes?	Y reporting of data objects N* reporting of data attributes *the Client process report but ignores single data attributes while they do not have time and quality. Skipped daName items of data set will

	be polled.
Describe how the client does respond when an URCB is already reserved	log entry + trying to use another one if dynamically allocated, fallback polling
Describe how the client does respond when a BRCB is already reserved	log entry, data will be polled
Describe how the client does respond on a SetBRCBValues(EntryID) respond-	log entry
Describe how the client does respond when a report has an unknown: dataset, RptId, unexpexted number of dataset entries, and/or unexpexted data type format entries	mismatches are not possible - dataset directory and RCB configuration is read out via MMS at each associate
Describe how the client detect reporting configuration changes (mismatches). Does it check the "configuration revision" attributes and/or does it check the dataset members?	RCB configuration is read out at each associate, "configuration revision" doesn't care
Describe how to force the client to change the RCB buffertime	no setting in the Client configuration For attributes with FC = BR|RP variables in the HMI project can be created, read, written.

PIXIT FOR LOGGING MODEL

Description	Value / Clarification
Does the client search for LCB in all logical nodes? when not specify the logical nodes	not supported
Describe how to change LOG and LCB attributes	not supported

PIXIT FOR GENERIC SUBSTATION EVENTS MODEL

Description	Value / Clarification
What elements of a subscribed GOOSE header are checked to decide the message is valid and the allData values are accepted? Ignored = element value is not checked, message will be accepted SCL match = element value should match with the configuration, otherwise the GOOSE message will	not supported

be ignored	
For the checked GOOSE header elements describe the checking conditions in more detail when necessary	not supported
What is the behavior when one subscribed GOOSE message isn't received or syntactically incorrect (missing GOOSE)	not supported
What is the behavior when one subscribed GOOSE message exceeds the previous time Allowed to Live (TAL)	not supported
What is the behavior when a subscribed GOOSE message is out-of-order	not supported
What is the behavior when a subscribed GOOSE message is duplicated	not supported
May the GOOSE data set contain structured data objects?	Y/N

PIXIT FOR CONTROL MODEL

Description	Value / Clarification
What control modes are supported	Y status-only Y direct-with-normal-security Y sbo-with-normal-security Y direct-with-enhanced-security Y sbo-with-enhanced-security For each use of */Oper.ctlVal[CO|SP], the Client automatically checks current */ctlModel[CF]. Depending on the value of that attribute, the corresponding control model is executed. Please refer Client documentation for more details.
Is Time activated operate (operTm) supported	N
Is "operate-many" supported	N
Can the client set the test flag?	N
What check conditions can be set	Y synchrocheck Y interlock-check

	There are two methods to set check conditions for Operate and SelectWithValue: 1. Check condition can be predefined by setting a value to */Oper.Check[CO] variable. The current variable value is used for command service parameter; 2. in the IEC 61850 interface (HMI): by setting in Command processing: Action its property Qualifier of Command (Please refer HMI Help)
Which originator categories are supported and what is the originator identification?	orCat =2 (station control)* orIdent ='zenon: nnn' where: nnn - computer name ' - apostrophe (CHAR(27)) *per default orCal=2 - setting in the Client configuration
Describe if and how the client sets/increments the ctlNum	for each use of any ctlVal[CO|SP] the Client increments* ctlNum from 0 to 127 and next turns to 0. * also by ctlModel=0 (no control possible)
What does the client when its receives a LastApplicationError and describe how to view the additional cause?	The Client gives possibility to create (pro */Oper structure) an internal "State Info" variable for received AddCause item value and variables ControlRun and ControlState showing current state and success/fail of command. These internal variables can be display and store in the HMI project. Please refer Client documentation for more details.
What does the client when its receives a Select, SelectWithValue or Operate respond negative ?	The Client sets corresponding values to intern variables ControlRun (-1=error) and AddCause. Optionally writes a log entry. Please refer Client documentation for more details.
Can the client change the control model via online services?	Y
What does the client when the ctlModel is not initialized in the SCL?	out of scope The Client does not read ctlModel from SCL, only via MMS
<additional items>	

PIXIT FOR TIME AND TIME SYNCHRONIZATION MODEL

Description	Value / Clarification
Described how to view the internal time & quality or how to expose the timestamp and timestamp quality via the IEC 61850 interface	The Client uses Windows time (local PC clock), thus the Client does not cooperate directly with any SNTP application. The Client does not send timestamp quality to control object.
What time quality bits are supported	N LeapSecondsKnown Y* ClockFailure Y* ClockNotSynchronized *only time quality bit received in t attribute
What is the behavior when the time synchronization signal/messages are lost	Not applicable
When is the quality bit "Clock failure" set?	The Client does not set any time quality bits, it supports reception of ClockFailure
When is the quality bit "Clock not synchronised" set?	The Client does not set any time quality bits, it supports reception of ClockNotSynchronised
<additional items>	The Client maps time quality bits received in t attribute into Status bits of variables responding same DO in the HMI project: - ClockNotSynchronised - status bits T_INVAL & T_EXTERN - ClockFailure - status bits T_INVAL & T_INTERN (and changes time attribute to HMI time). Please refer Client documentation for more details.

PIXIT FOR FILE TRANSFER MODEL

Description	Value / Clarification
Describe when or how to force the client to request GetServerDirectory(FILE) and what it does with the responded filenames	The Client gives possibility to create two internal string variables: a Command and a Directory variable. Setting value of Command to: "DIR <fileSpec>" results with the name or list of files in Directory variable. Please refer Client documentation for more details.

Does the client uses a wildcard in the GetServerDirectory(FILE) request	No
Does the client support IED's that include the path in the file name in the GetServerDirectory(FILE) respond?	Y path included Y path not included Y list of files with a char separator
Does the client support IED's that use the fileseparator	N "/" Y "\"
What is the maximum file name size including path	OS limit, e.g. Windows Vista - 256 chars
Can the client read a file with size 0	Y
Are directory/file name case sensitive	Not case sensitive only Command value is case sensitive: "DIR <fileSpec>" or "GET <fileSpec>", the <fileSpec> is not case sensitive
Maximum file size	per design 4 GB practical* 25 MB * transfer of 25 MB takes about 40 seconds, bigger transfers wasn't tested
Describe how the client behaves in case of: GetFileAttributes response-	Command Variable value changes to "DIR ERROR" (or "GET ERROR" for GetFile responce-)
<additional items>	For DeleteFile set Command variable value to "DEL <fileSpec>"