zenon driver manual

IEC850

v.7.50

Contents

1. Welcome to COPA-DATA help 6

2. IEC850 6

3. IEC850 - Data sheet 7

4. Requirements 8

4.1 PC 8

5. Configuration 9

5.1 Creating a driver 9

5.2 Settings in the driver dialog 11

5.2.1 General 11

5.2.2 Driver dialog basic settings 14

5.2.3 Connections 20

6. Creating variables 39

6.1 Creating variables in the Editor 39

6.2 Addressing 42

6.3 Driver objects and datatypes 45

6.3.1 Driver objects 45

6.3.2 Assignment of data types 47

6.4 Creating variables by importing 49

6.4.1 Online import 50

6.4.2 Offline import 54

6.4.3 XML import 56

6.4.4 DBF Import/Export 57

6.5 Driver variables 62

7. IEC850 functions 69

7.1 Establishment of a connection and detection of a connection failure 70

7.2 Commands (Control Model) 72

7.2.1 Select and cancel 74

7.2.2 Additional Cause Diagnosis 75

7.2.3 Service parameters of the command ... 76
7.3 Service Tracking ... 80
7.4 Mapping of double point values ... 82
7.5 Quality, time stamp and status bits of the variable ... 84
7.6 Filetransfer ... 87
7.6.1 Request folder information ... 88
7.6.2 Get file from server ... 88
7.6.3 Delete file ... 89

8. Reporting ... 89
8.1 Unbuffered Reporting ... 93
8.2 Buffered Reporting ... 95
8.3 RCB activation - verification possibilities ... 98
8.4 Dynamic Data Sets ... 100

9. Changing the driver mode in Runtime ... 101

10. Error analysis ... 104
10.1 Analysis tool ... 104
10.2 Check list ... 105

11. Appendix A - Description of the 'Functional Constraints' (FCs): ... 106

12. Appendix B - Abbreviations for data object/data attribute ... 107

13. Appendix C - Conformance statement ... 117
13.1 Protocol implementation conformance statement (PICS) ... 117
13.2 Model implementation conformance statement (MICS) ... 124

14. Appendix D - PIXIT ... 125

15. Anhang E - TICS ... 141
15.1 Part 6 ... 142
15.2 Part 7-1 ... 143
15.3 Part 7-2 ... 144
15.4 Part 7-3 ... 145
15.5 Part 7-4 ... 146
15.6 Part 8-1 ... 148

1. Welcome to COPA-DATA help

GENERAL HELP

If you cannot find any information you require in this help chapter or can think of anything that you would like added, please send an email to documentation@copadata.com (mailto:documentation@copadata.com).

PROJECT SUPPORT

You can receive support for any real project you may have from our Support Team, who you can contact via email at support@copadata.com (mailto:support@copadata.com).

LICENSES AND MODULES

If you find that you need other modules or licenses, our staff will be happy to help you. Email sales@copadata.com (mailto:sales@copadata.com).

2. IEC850

Communication between the driver and the PLC is based on the IEC 61850 protocol with client/server services via TCP/IP (A1/T1 profile). The driver acts as a client (master) when communicating.

> **Attention**
>
> Configurations that have been edited with a driver from version 3400 cannot be opened with older drivers anymore.

3. IEC850 - Data sheet

General:	
Driver file name	IEC850.exe
Driver name	IEC 61850 driver
PLC types	IEC 61850 compatible PLCs
PLC manufacturer	ABB; Siemens; IEC; SAT; Sprecher Automation; Kalki; GE Multilin; Schweitzer Engineering Laboratories; SEL; NR Electric; NARI;

Driver supports:	
Protocol	IEC 61850;
Addressing: Address-based	--
Addressing: Name-based	X
Spontaneous communication	X
Polling communication	X
Online browsing	X
Offline browsing	X
Real-time capable	X
Blockwrite	--
Modem capable	--
Serial logging	--
RDA numerical	--
RDA String	--

Requirements:	
Hardware PC	standard Network Adapter
Software PC	XML-Lite is required, it is included in Microsoft Internet Explorer 7.0; xmllite.dll, can also be downloaded from Microsoft Homepage.
Hardware PLC	--
Software PLC	--
Requires v-dll	--

Platforms:	
Operating systems	Windows CE 6.0, Embedded Compact 7; Windows 7, 8, 8.1, 10, Server 2008R2, Server 2012, Server 2012R2;
CE platforms	x86; ARM;

4. Requirements

This chapter contains information on the requirements that are necessary for use of this driver.

4.1 PC

XML Lite, which is part of Microsoft Internet Explorer 7.0, is necessary; xmllite.dll can also be downloaded separately from the Microsoft website.

This driver supports a connection via the standard network card of the PC. Make sure that the PLC and the PC are in the same network range and that the subnet masks are set accordingly on both devices.

5. Configuration

In this chapter you will learn how to use the driver in a project and which settings you can change.

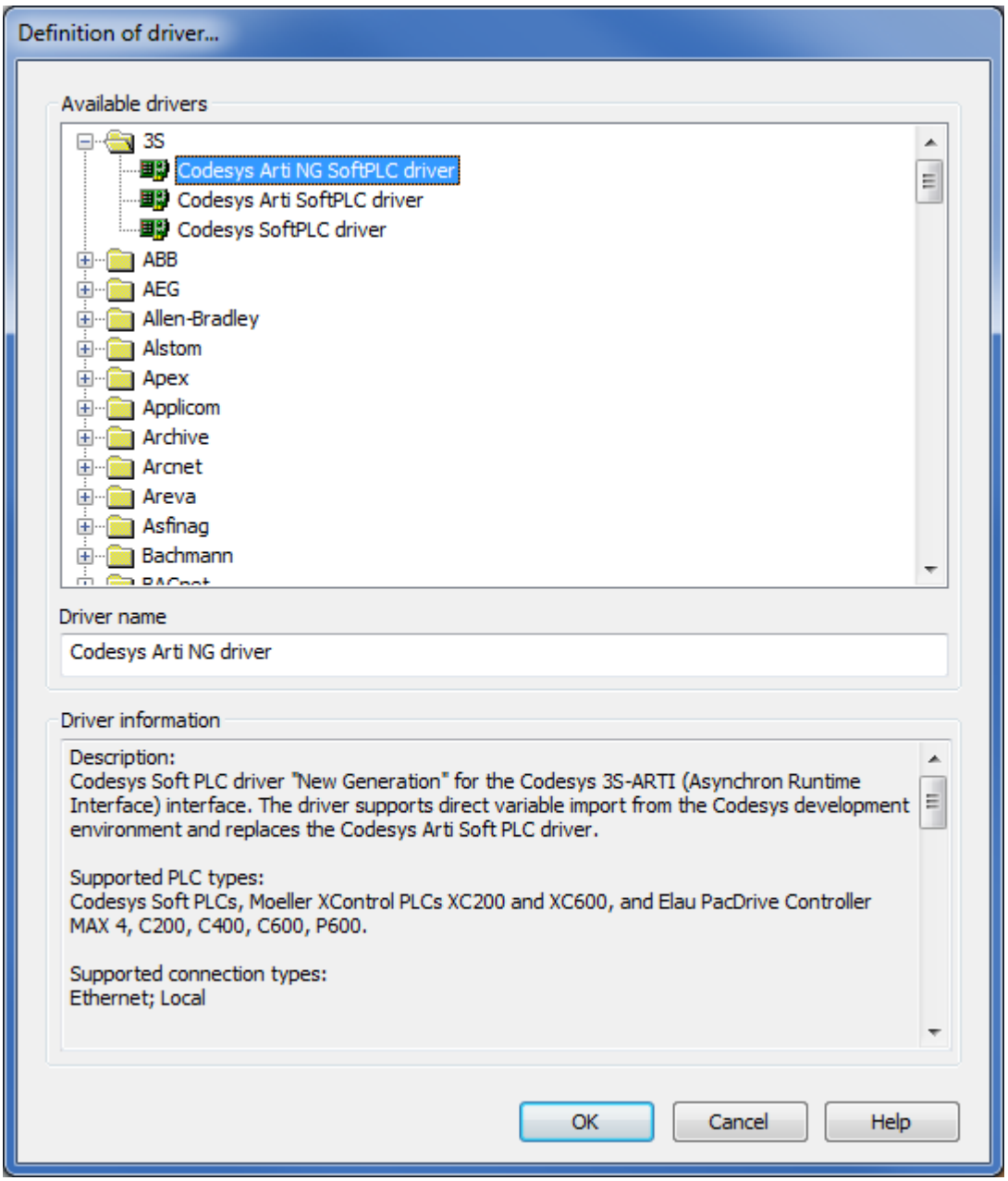
> **Information**
>
> *Find out more about further settings for zenon variables in the chapter Variables (main.chm::/15247.htm) of the online manual.*

5.1 Creating a driver

In order to create a new driver:

1. Right-click on **Driver** in the Project Manage and select **Driver new** in the context menu.

2. In the following dialog the control system offers a list of all available drivers.



3. Select the desired driver and give it a name:
   - The driver name has to be unique, i.e. if one and the same driver is to be used several times in one project, a new name has to be given each time.
   - The driver name is part of the file name. Therefore it may only contain characters which are supported by the operating system. Invalid characters are replaced by an underscore (_).
   - Attention: This name cannot be changed later on.

4. Confirm the dialog with **OK**. In the following dialog the single configurations of the drivers are defined.

Only the respective required drivers need to be loaded for a project. Later loading of an additional driver is possible without problems.

Information

For new projects and for existing projects which are converted to version 6.21 or higher, the following drivers are created automatically:

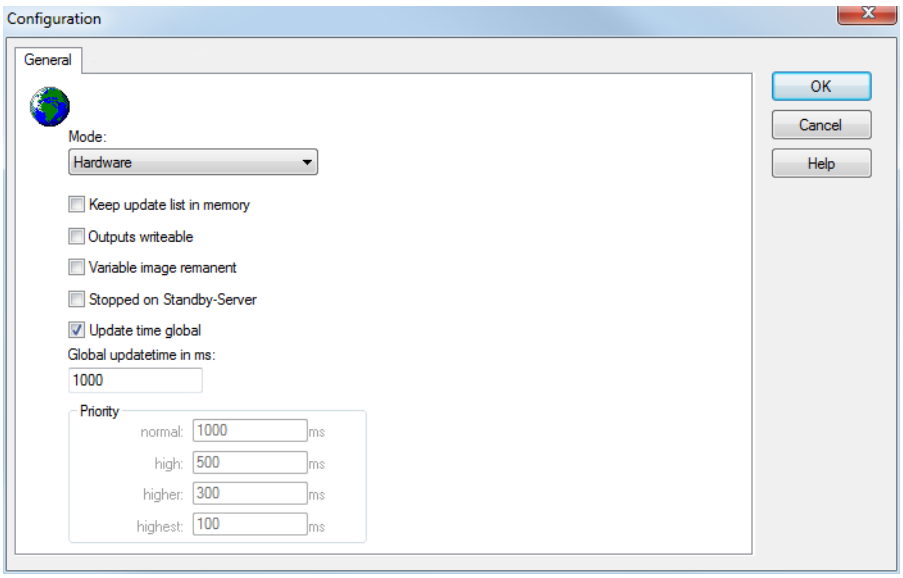
- Internal
- MathDr32
- SysDrv.

5.2 Settings in the driver dialog

You can change the following settings of the driver:

5.2.1 General

The configuration dialog is opened when a driver is created. In order to be able to open the dialog later for editing, double click on the driver in the list or click on the **Configuration** property.



Parameters	Description
Mode	Allows to switch between hardware mode and simulation mode ▸ Hardware: A connection to the control is established. ▸ Simulation static No communication between to the control is established, the values are simulated by the driver. In this modus the values remain constant or the variables keep the values which were set by zenon Logic. Each variable has its own memory area. E.g. two variables of the type marker with offset 79 can have different values in the Runtime and do not influence each other. Exception: The simulator driver. ▸ Simulation - counting No communication between to the control is established, the values are simulated by the driver. In this modus the driver increments the values within a value range automatically. ▸ Simulation - programmed N communication is established to the PLC. The values are calculated by a freely programmable simulation project. The simulation project is created with the help of the zenon Logic Workbench and runs in a zenon Logic Runtime which is integrated in the driver. For details see chapter Driver simulation (main.chm::/25206.htm).
Keep update list in the memory	Variables which were requested once are still requested from the control even if they are currently not needed. This has the advantage that e.g. multiple screen switches after the screen was opened for the first time are executed faster because the variables need not be requested again. The disadvantage is a higher load for the communication to the control.
Output can be written	`Active`: Outputs can be written. `Inactive`: Writing of outputs is prevented. Note: Not available for every driver.
Variable image remanent	This option saves and restores the current value, time stamp and the states of a data point. Fundamental requirement: The variable must have a valid value and time stamp.

	The variable image is saved in mode hardware if: ▸ one of the states S_MERKER_1(0) up to S_MERKER8(7), REVISION(9), AUS(20) or ERSATZWERT(27) is active The variable image is always saved if: ▸ the variable is of the object type **Driver variable** ▸ the driver runs in simulation mode. (not programmed simulation) The following states are not restored at the start of the Runtime: ▸ SELECT(8) ▸ WR-ACK(40) ▸ WR-SUC(41) The mode **Simulation - programmed** at the driver start is not a criterion in order to restore the remanent variable image.
Stop on Standby Server	Setting for redundancy at drivers which allow only on communication connection. For this the driver is stopped at the Standby Server and only started at the upgrade. Attention: If this option is active, the gapless archiving is no longer guaranteed. `Active`: Sets the driver at the not-process-leading Server automatically in a stop-like state. In contrast to stopping via driver command, the variable does not receive status **switched off** (**statusverarbeitung.chm::/24150.htm**) but an empty value. This prevents that at the upgrade to the Server irrelevant values are created in the AML, CEL and Historian. Note: Not available if the CE terminal serves as a data server. You can find further information in the zenon Operator manual in the CE terminal as a data server chapter.
Global Update time	`Active`: The set **Global update time** in `ms` is used for all variables in the project. The priority set at the variables is not used. `Inactive`: The set priorities are used for the individual variables.
Priority	The polling times for the individual priority classes are set here. All variables with the according priority are polled in the set time. The allocation to the variables takes place separately in the settings of the variable properties. The communication of the individual variables are graduated in respect of importance or necessary topicality using the priorities.

	Thus the communication load is distributed better. Attention: Priority classes are not supported by each driver For example, drivers that communicate spontaneously do not support it.

CLOSE DIALOG

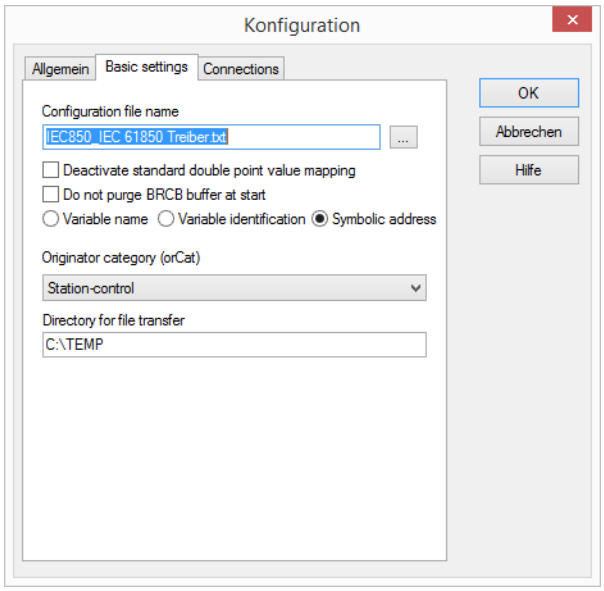
Parameters	Description
OK	Applies all changes in all tabs and closes the dialog.
Cancel	Discards all changes in all tabs and closes the dialog.
Help	Opens online help.

UPDATE TIME FOR CYCLICAL DRIVERS

The following applies for cyclical drivers:

For **Set value**, **Advising** of variables and **Requests**, a read cycle is immediately triggered for all drivers - regardless of the set update time. This ensures that the value is immediately available for visualization after writing. Update times can therefore be shorter than pre-set for cyclical drivers.

5.2.2 Driver dialog basic settings



Parameters	Description
Configuration file name	Name of the file in which the driver-specific configurations are saved. Click on the **...** button to open the dialog for selecting a drop folder. Note: You can read about the structure of this configuration file in the Driver configuration file (on page 16) chapter.
Deactivate standard double point value mapping	▸ `Inactive`: The values of Double Point Values are adjusted to the operating elements of zenon. Use this configuration if you want to use the modules of the zenon Energy Edition. You can find details in the Double Point Value Mapping (on page 82) chapter. ▸ `Active`: The values of the Double Point Values are forwarded to zenon as they are. In this case, you cannot use the command processing function of zenon Energy or ALC for example. Default: `Inactive`
Do not purge BRCB buffer at start	▸ `Active`: The buffer of the BRCBs are not deleted on the IED when the connection is first established after Runtime has started. Requirement: For this BRCB , the last EntryID received has been saved. If no EntryID has been saved yet, the property is ignored and the driver sets BRCB.PurgeBuf=TRUE. Only has an effect after the first attempt to activate a BRCB after the start. Is ignored with repeated attempts to set a BRCB.RptEna=TRUE - if the **RCBs enable retries** time has expired. More in the **Buffered reporting** (on page 95) chapter.
Addressing:	States which variable property is used in the driver for the addressing (ObjectReferenz). ▸ **Variable name** (**Name**) ▸ Variable identification (**Identification**) ▸ Symbolic address (**Symbolic address**) Default: **Symbolic address** Note: When importing the variables, the driver fills all three variable properties with valid references. You can rename the properties that the driver does not then use for addressing.
Originator category (orCat)	The driver sends the configured **orCat** value in commands - in SBOw and Oper structures - that it writes to the IED .

	Possible settings: ▸ `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	Directory for file transfer. All files loaded are saved in this folder. More in the Filetransfer (on page 87) chapter.

In the Editor, the driver saves its configuration in the TXT file as defined in **Configuration file name**. In Runtime, the driver gets its configuration from the copy of the file that the editor has provided.

STRUCTURE OF A CONFIGURATION FILE OF THE DRIVER

Line	Description
1	Number of configured servers
2 – (m-1)	Server configuration (see server configuration)
m – n	Possible further server configuration(s)

SERVER CONFIGURATION

Line	Description	Example
1	Start of a server configuration (= `*** SERVER ***`)	*** SERVER ***
2	Number of items in this server configuration, abbreviated: `CNTSRVITEMS`	44
3	Net address	1
4	Server name	RELAY1
5	Primary IP address	192.168.250.22
6	Primary IP port	102
7	**Polling rate** Read interval in milliseconds	1000
8	Calling AE qualifier	12
9	Called AE qualifier	12
10	Calling AP title[0]	1
11	Called AP title[0]	1
12	Calling AP title[1]	1
13	Called AP title[1]	1
14	Calling AP title[2]	1
15	Called AP title[2]	999
16	Calling AP title[3]	999
17	Called AP title[3]	1
18	Calling AP title[4]	-1
19	Called AP title[4]	1
20	Calling AP title[5]	-12851
21	Called AP title[5]	-1
22	Calling AP title[6]	-12851

23	Called AP title[6]	-12851
24	Calling AP title[7]	-12851
25	Called AP title[7]	-12851
26	Calling AP title[8]	-12851
27	Called AP title[8]	-12851
28	Calling AP title[9]	-12851
29	Called AP title[9]	-12851
30	**Max. auto used URCBs** Maximum number of Unbuffered Reports (URCBs) that the driver activates with automatic assignment per Logical Device	10
31	* - in newer configurations With old configuration: names of the assigned Buffered Reports (BRCBs), separated with commas	*
32	**Use preconfigured (SCL) options** 0 = subsequently configured RCB settings (TrgOps, OptFlds, IntgPd, BufTm) are used 1 = the RCB settings that have already been preconfigured in the IEC61850 server - in its SCL file - are used	0
33	**Use Report-ID for RCB assignment** 0 = The RCB instances of the server are identified by name. 1 = Report ID that is used instead of the report name in the dialog for RCB assignment.	0
34	**Use Authentication** 0 = no ISO-Authentication used 1 = If active, the driver sends the **Authentication String** at establishing the connection.	0
35	**Authentication String**	
36	Alternative IP address	
37	Alternative IP port	0
38	TrgOp data-change: 0 = inactive ; 1 = active	1
39	TrgOp quality-change: 0 = inactive ; 1 = active	1
40	TrgOp data-update: 0 = inactive ; 1 = active	0
41	TrgOp integrity: 0 = inactive ; 1 = active	0
42	TrgOp general-interrogation: 0 = inactive ; 1 = active	1
43	**GetNameList on DO**	0

	0 = Normal GetNameList 1 = The driver reads the object model by requesting data objects (DO) for each Logical Node available in the server and each Functional Constraint (FC) (FC) defined in the IEC61850 standard.	
44	**Integrity Period**	7000
45	**Buffer Time**	500
46	**OptFlds** Optional fields of the RCB	73
47	**RCBs enable reties** Cycle in seconds in which an attempt is made to activate RCBs that were not activated successfully again. Only present if CNTSRVITEMS >= is 45	7
48	**Automatic Watchdog**	1
49	**Data consistency scan**	300
50	**Use SCADA network orIdend**	0
51	Number of client configurations	
52 – (m-1)	Client configuration (see Client configuration)	
m – n	Possible further client configurations	

CLIENT CONFIGURATION

Line	Description	Beispiel
1	Start of a client configuration (= `*** CLIENTCFG ***`)	*** CLIENTCFG ***
2	Number of Items in this client configuration	1
3	**Hostname (RT computer name)** Name of the computer on which the driver is running that receives the reports	WKS007
4	**ClientLN.iedName**	SCADA_Server1
5	**orIdent**	
6	Number of RCB configurations	2
7 – (i-1)	RCB configuration (see RCB configuration)	
i – (j-1)	Possible further RCB configurations	
y	Number of dynamic dataset configurations	
k – (l-1)	Dynamic dataset configuration (see dynamic dataset configuration)	

l – m	Possible further dynamic dataset configuration	

RCB CONFIGURATION

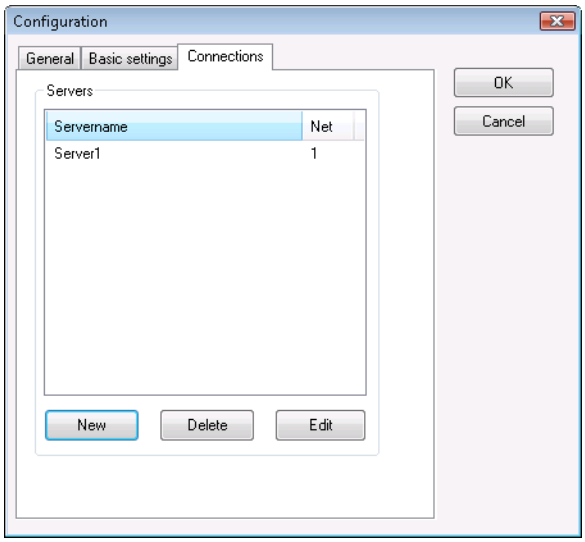
Line	Description	Example
1	Start of an RCB configuration (= `*** RCBCFG ***`)	*** RCBCFG ***
2	Number of items in this RCB configuration	2
3	RCB name or ID	UP_CTRL/LLN0/urcb_QxCSWI1_Pos02[RP]
4	Name of the dynamic data set.	NEW_DYN_DATASET

DYNAMIC DATASET CONFIGURATION

Line	Description	Example
1	Start of a dynamic dataset (= `*** DATASET ***`)	*** DATASET ***
2	Number of items in this dynamic dataset configuration	1
3	Name of the dynamic data set	NEW_DYN_DATASET
4	Number of Object References of the dynamic dataset	2
5	Object Reference	UPCTRL/Q1CSWI1CFPos
6 – n	Possible further Object Reference.	UPCTRL/Q1CSWI1STPos

5.2.3 Connections

The connections of the IEC850 driver are set in the **Connections** tab of the driver dialog.



Parameters	Description
Server	List of connectiosn to the IEC 61850 server that have already been configured (to the control objects).
New	Creates a new connection to an IEC 61850 server. Opens the 'Server' dialog.
Delete	Removes the selected connection from the list.
Edit	Opens the 'Server' dialog to edit the selected connection

CLOSE DIALOG

Parameters	Description
OK	Applies all changes in all tabs and closes the dialog.
Cancel	Discards all changes in all tabs and closes the dialog.
Help	Opens online help.

Server

Clicking on **New** or **Edit** in the dialog **Connection** (on page 20) opens the dialog for the configuration of a connection to an IEC 61850 server:

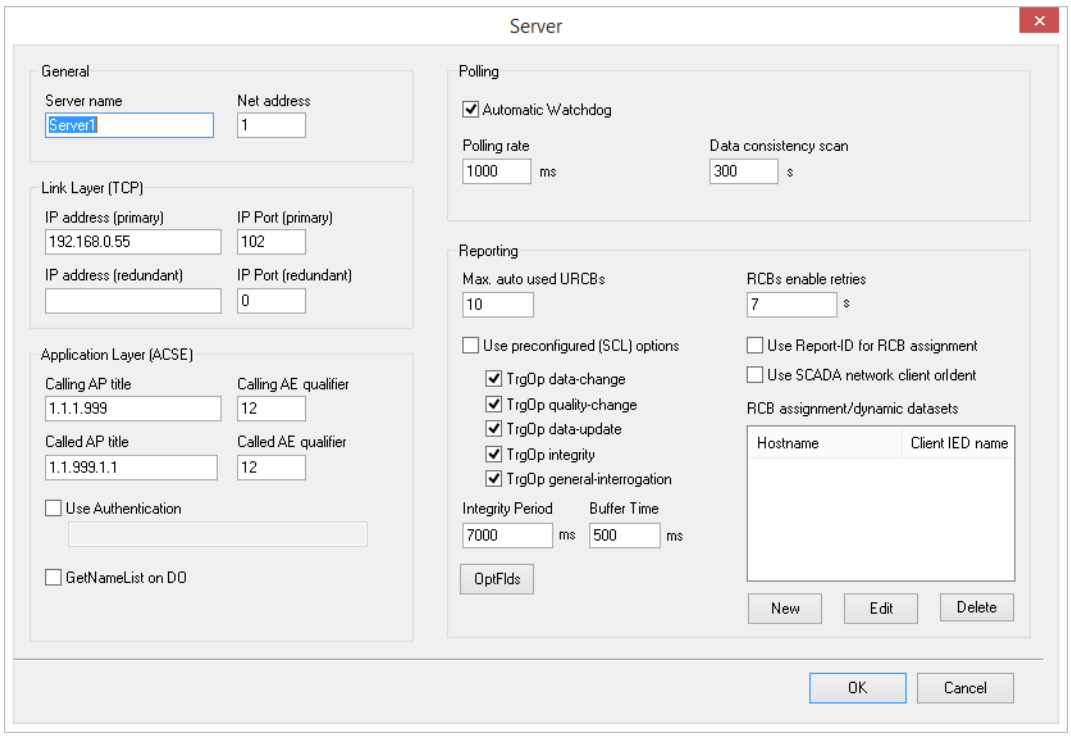
Note: This dialog is only available in English.

Note: In order to be able to establish the first connection to a server, it is sufficient if the following properties have been configured:

- Server name

- Net address
- IP address

You can therefore already communicate with the server in Runtime or create variables by means of online import (on page 49) in the zenon Editor.



GENERAL

Parameters	Description
Server name	Freely definable name. Is used for the names of variables when variables are imported. Default: `Server1`
Net address	Corresponds to the **Net address** property (property group: **Addressing**) in variable configuration. Maximum value: `65535`

LINK LAYER (TCP)

Parameters	Description
Link Layer (TCP)	Connection settings
IP adress (primary)	IP address of the server to which a connection is to be made.
IP Port (primary)	Port via which the connection will be established. Default: `102`
IP adress (redundant)	Alternative IP address. If the connection to the first IP address fails, the alternative IP address will be used after the net error waiting time has passed `(20-30 sec., depending on the network)`. The alternative address will be kept until the driver is restarted (via driver functions or restart of the zenon Runtime) or until the connection fails. Then, the first IP address will be used again. Attention: In order to detect a loss of a connection, the **Automatic watchdog** watchdog property should be activated or at least one variable, such as `*LLN0/Mod/stVal[ST]` should always be polled.
IP Port (redundant)	Alternative IP port.

POLLING

Parameters	Description
Automatic Watchdog	Checkbox to activate the watchdog. If active, the driver automatically sends, for each **Polling rate** configured, a read request for `*LLN0$ST$Mod$stVal` in order to detect a possible loss of connection Note: At least one data attribute should always be polled even if all data of the project is present in the report. Otherwise the failure of the IEC61850 server cannot be detected. Only deactivate this property if it is guaranteed in the project that at least one polled variable is always signed in.

	Default: `active`
Polling rate	Defines the update frequency in milliseconds. The driver supports this rate to poll data that is not in active **reports**. At this a possible failure of the IEC61850 server is also considered. Maximum value: `4294967295` (2^32-1) If a higher value is entered, it is automatically changed to 1 when saved. Default: `1000 ms` Attention: Exchange of data in Runtime does not depend on this setting or on the global update time. This is set to `100ms` and fixed.
Data consistency scan	Defines the cycle in which the driver checks the data model to see that it is consistent. Is only used if, in configured reports, the datasets contain individual data attributes instead of data objects. Default: `300 seconds` Input range: `0 – 999999` (higher inputs are not possible)

REPORTING

Parameters	Description
Reporting	Settings for the reports (on page 89).
Max. auto used URCBs	Maximum number of **Unbuffered Reports** which the driver activates at automatic allocation per Logical Device, i.e. in addition to RCBs that were configured in the **RCB assignment/dynamic datasets** dialog. Default: `10` Entry is only valid for **Unbuffered Reports**. you can find more information on automatic allocation in the **Unbuffered Report** (on page 93) chapter.
RCBs enable retries	Configuration of the time in which the driver reattempts writing of the RCBs that were not registered successfully - an RCB.RptEna=TRUE. Entry of the time in seconds. Entry range: `0 – 999999` Default: `7` Note: ▸ Should be greater than the **Polling rate**, otherwise it is not guaranteed the the variables affected have the initial values. ▸ With an entry of 0, there is no attempt to register an RCB again. ▸ These settings only concern RCBs that are listed in the **RCB assignment** section.

Use preconfigured (SCL) options	`Active`: The driver activates a report without overwriting the data attributes of the RCB. The content of the SCL file of the server is defined as a result of this. The following data attributes are affected by this: ▶ IntgPd ▶ BufTime ▶ TrgOps ▶ OptFields `Inactive`: The driver writes the data attributes of the RCB during activation. You can activate/deactivate the following trigger options regardless of one another. ▸ `TrgOp: data-change` ▸ `TrgOp: quality-change` ▸ `TrgOp: data-update` ▸ `TrgOp: integrity` ▸ `TrgOp: general-interrogation` Default: all `active` Note: Not all servers support TrgOps data-change and data-update together. TrgOp intergity can also lead to an unnecessary overload of communication if a an IntgPd (Integrity Period) that is too short was defined in the server for RCB. In case of doubt, set TrgOps: data-change + quality-change + general-interrogation.
Integrity Period	Time interval (IntgPd) in milliseconds in which the server sends an Integrity Report. Default: `7000` ms Note: not active if `TrgOp integrity` is deactivated or **Use preconfigured (SCL) options** is activated. Because an Integrity Report does not normally contain value changes, it is expressly recommended that only one single report on the server is activated with `TrgOp: integrity`. With an activated integrity report, the server can detect a connection failure more quickly. However zenon does not need this report.
Buffer Time	Time interval (BufTime) in milliseconds in which the server collects the data for a report. Default: `500 ms` Note: not active if **Use preconfigured (SCL) options** is activated.

OptFlds	Opens the dialog to configure the Optional Fields (on page 28) of the report. Note: not active if **Use preconfigured (SCL) options** is activated.
Use Report-ID for RCB assignment	`Active`: The report ID is used instead of the report name in the **Client configuration** dialog for static RCB assignment. Note: The IEC61850 standard allows several RCBs to have the same value in RptID . In Runtime, the driver will search for all RCBs with the respective Report ID and register them. `Inactive`: The RCB instances of the server are identified by name. Default: `inactive` Attention: If this option is subsequently changed, all RCB assignments in the **client configuration** must be deleted and created again.
Use SCADA network orIdend	If this checkbox is activated, `orIdent` is used in commands, based on the name of the computer on which the user has triggered the command. The computer name is resolved from the list of host names. This configuration is carried out in the **Client configuration** dialog (on page 29). If the checkbox is not activated, the orIdent of the Primary Server is used. Default: `Inactive` Note: The default origin.orIdent `'zenon: <computer_name>'` is used in commands if: ▸ In the client configuration for the corresponding computer (= **Hostname**), no orIdent is configured. ▸ in the client configuration for the corresponding computer, no entry with **Hostname** has been configured.
RCB assignment/dynamic datasets	Static Report Control Block allocations and settings for Dynamic Data Sets. ▸ `Hostname` ▸ `Client IED name` ▸ `Client orIdent` As configured in the **Client configuration** (on page 29) dialog. The width of the column display can be increased or reduced by clicking

	the mouse. The section can be moved with the scroll bar. Note: If you have activated 'Use redundancy' in your zenon project, (**Network active** property activated and **Server 1** and **Server 2** configured), you must create two entries here: one for the server and one for the standby.
New	Creates a new entry in the list. Opens the **Client configuraton** (on page 29) dialog**.**
Del	Removes the selected entry from the list.
Edit	Opens the **Client configuration** (on page 29) dialog to edit the selected entry.

APPLICATION LAYER (ACSE)

Parameters	Description
Application Layer (ACSE)	Settings for the ACSE.
Calling AP title	Settings according to ISO 8650-1(ACSE), the value for the `OSI ACSE AP Title` of the client in the Universal Identifier notation. Should not normally be amended.
Calling AE qualifier	Settings according to ISO 8650-1(ACSE). Should not be used in standard cases.
Called AP title	Settings according to ISO 8650-1(ACSE), `OSI-AP-Title` of the server in the format for Universal Identifier. Should not normally be amended.
Called AE qualifier	Settings according to ISO 8650-1(ACSE). Should not be used in standard cases.
Use Authentication	Activate this checkbox if you want to use authentication according to ISO 8650-1. `Active`: If active, the driver sends the Authentication String at establishing the connection.
`[Input field for Authentication String]`	Input field for `Authentication String`. Note the length limitation of 55 characters in accordance with ISO 8327-1 OSI session protocol. Note: Only active if **Use Authentication** is active.
GetNameList on DO	`Active`: The driver reads the Objekt Model by requesting data objects (DO) for each available Logical Node pro in every server in the IEC61850 standard defined Functional Constraint (FC).

	Therefore Objekt Models can then be read when some Logical Nodes are larger than the PDU size supported by the server. Attention: If this option is active, it can take much longer to establish the connection. Only activate this option if reading in cannot be carried out in full due to server limitations. Default: `inactive`

CLOSE DIALOG

Parameters	Description
OK	Applies settings and closes the dialog.
Cancel	Discards all changes and closes the dialog.

Information

OSI value (ACSE)

The driver uses the following OSI selector values:

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

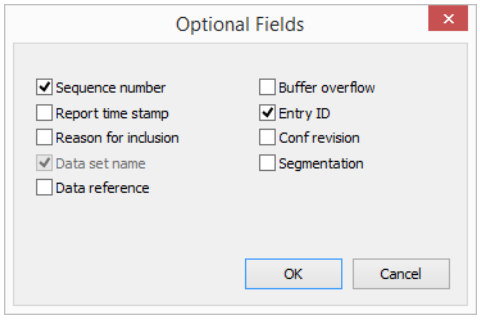
In a server SCL file, it corresponds to the following entries:

- <P type="OSI-AP-Title">41,999,1,1</P>
  - coded in accordance with Basic Encoding Rules (IEC 8825) for Object Identifier - corresponds to standard values of the Universal Identifiers in the **Called AP title**.
- <P type="OSI-AE-Qualifier">12</P>
  - corresponds to standard values in the **Called AE qualifier**.
- <P type="OSI-PSEL">00000001</P>
  - set in the driver; cannot be changed
- <P type="OSI-SSEL">0001</P>
  - set in the driver; cannot be changed
- <P type="OSI-TSEL">0001</P>
  - set in the driver; cannot be changed

Optional Fields

The Optional Fields are written on the server when a report is activated. These correspond to the bits in the OptFlds data attribute of the RCB.

"**Sequence number**", "**Data set name**" and "**Entry ID**" are activated by default. "**Data set name**" cannot be changed, because without this option the driver cannot evaluate the reports received.



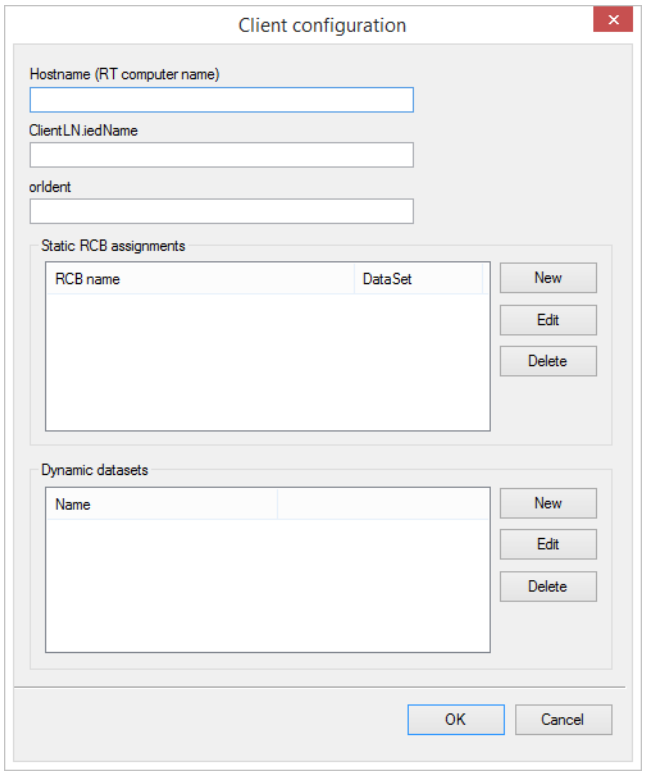
CLOSE DIALOG

Parameters	Description
OK	Applies settings and closes the dialog.
Cancel	Discards all changes and closes the dialog.

Client configuration

Clicking on **New** or **Edit** in the **RCB assignments/dynamic datasets** list in the server dialogs (on page 21) opens the dialog for the client configuration settings:

Note: This dialog is only available in English.



Parameters	Description
Hostname (RT computer name)	Name of the computer on which the driver is running that receives the reports: ▸ For standalone projects: Computer name of the computer on which zenon Runtime is running ▸ In the zenon network: Computer name of the project server or project standby server Note: Do not use "`localhost`"; use the computer names from the operating system instead. The **Hostname** must not be empty if at least one of the following configuration settings has been configured. Attention: If the current **Runtime computer name** and the configured **host name** do not correspond, the reports defined under S**tatic RCB assignment** (on page 33) are ignored.
ClientLN.iedName	IED name of the IEC61850 clients as stated in the SCD file and given there under RCB.RptEnabled.ClientLN. Permitted characters: A to Z, a to z and 0 to 9 and underscore _ maximum 64 characters. Default: `empty` Hint: In order to easily configure the driver with a SCD file, use the **IEC850 Driver Configuration Wizard**. You can find further details in the Wizard documentation.
orIdent	Input field for the configuration of the orIdent. For each computer in the zenon network, a freely-definable orIdent entry can be configured. You can use up to 32 characters. Default: `empty (= 'zenon: <computer_name>')` Note: The originator that is used in commands is also dependent on the configuration in the Basic **Settings** dialog (**orCat** property) and in the **Server** dialog (**Use Client orIdent** checkbox). If the **Use Client orIdent** checkbox is not active, the `orIdent` of the Primary Server is always used in commands. If the **Use Client orIdent** checkbox is active: The default origin.orIdent `'zenon: <computer_name>'` is used in commands if: ▸ In the client configuration for the corresponding computer (= **Hostname**), no orIdent is configured.

	▸ in the client configuration for the corresponding computer, no entry with **Hostname** has been configured.

STATIC RCB ASSIGNMENTS

Settings for the static assignment of RCBs (the same for unbuffered and buffered RCBs).

Note: Buffered reports must be configured in the driver configuration. Otherwise they are not used.

For more information, see driver-specfic functions (on page 69).

Parameters	Description
RCB name	Name of the configured RCBs. RCBs are configured in the Statically assigned RCB dialog. This configuration is started by clicking on the corresponding **New** button.
DataSet	Displays the dynamic Data Set (if the RCB was allocated one). Allocation: 1. Configure, in the **Dynamic datasets** area with the attendant **New** button, a corresponding Dynamic dataset. 2. This dynamic dataset can be selected in the Statically assigned RCB dialog in the Dynamic dataset drop-down list.
New	Creates a new entry in the list. Opens the '**"Statically assigned RCB** (on page 33)' dialog to create a new entry with a report ID (or a name) - you can either enter it manually or browse the server and if necessary with **Dynamic Data Set** (is taken from the lower list).
Edit	Opens the '**Statically assigned RCB** (on page 33)' dialog to edit the selected entry.
Delete	Removes the selected entry from the list.

DYNAMIC DATASETS

Settings for dynamic data sets.

Note: Dynamic data sets are not a requirement for buffered or unbuffered reports. They are optional settings if the IEC61850 server that is used supports this.

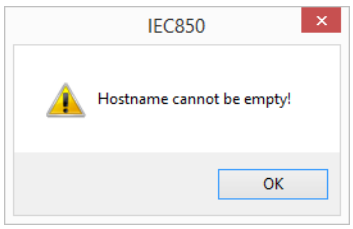
For more information, see dynamic data sets (on page 100).

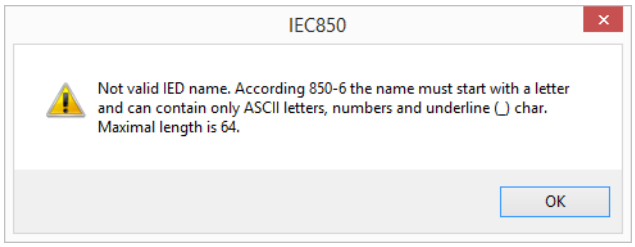
Parameters	Description
Name	Displays the data sets that have already been enterd - that are already present in the upper list.

New	Opens the dialog for creating a data set.
Edit	Opens the '**dynamic DataSet configuration** (on page 37)' dialog to edit the selected entry.
Delete	Removes the selected entry from the list.

WARNING NOTICES

If the configuration is not correct, you are notified of this with a corresponding warning dialog. Confirm this dialog by clicking on the OK button to return to the configuration dialog. Then correct the error.

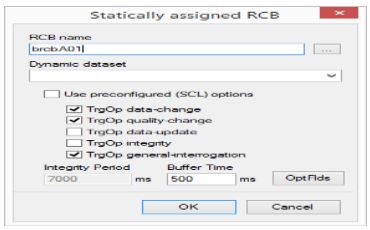




Note: The warning dialog is only available in English.

Statically assigned RCB

Clicking on **New** or **Edit** in the **static RCB assignment** list of the **Client configuration** (on page 29) dialog opens the dialog to create an RCB or to assign an RCB a Dynamic Data Set. RCBs are defined by means of the RCB name:



Parameters	Description
RCB name/Report-ID	The name (unique) or the RptID (possibly not unique) of the report - buffered or unbuffered - that is to register the driver (i.e. use it). Must correspond exactly to the name/report ID in the PLC (IEC61850 server). A click on **...** starts online browsing of the configured IEC 850 server and shows if the server is running, all Report Control Blocks that are currently in the data model of the server. Note: **RCB name** is expected here by default. The RCB name is saved with the full `ObjectReference` and is thus unique. The use of the **RCB name** can be changed to report ID using the **Use Report-ID for RCB assignment** property in the **Server** (on page 21) properties. **Report-ID**: the report ID is saved as it is received from the 850 server; the same ID can thus affect several RCBs. The standard allows several RCBs to have the same value in RptID . Attention: In Runtime, the driver will search for all RCBs with the respective Report ID and register them. Note: If Runtime is also running at the same time on the same computer, some 850 servers can refuse the second connection from the same IP. The driver can then not read out the RCBs in the Editor.
dynamic DataSet	You can also optionally select a data set from the drop-down list of the data sets that have already been entered into the 'dynamic DataSet configuration (on page 37)' dialog. Important: If you do not define a data set here, the data set that is predefined for this RCB in the PLC (IEC61850 server) is used.
Use preconfigured (SCL) options	`Active`: The driver activates a report without overwriting the data attributes of the RCB. The content of the SCL file of the server is defined as a result of this. The following data attributes are affected by this: ▶ IntgPd ▶ BufTime

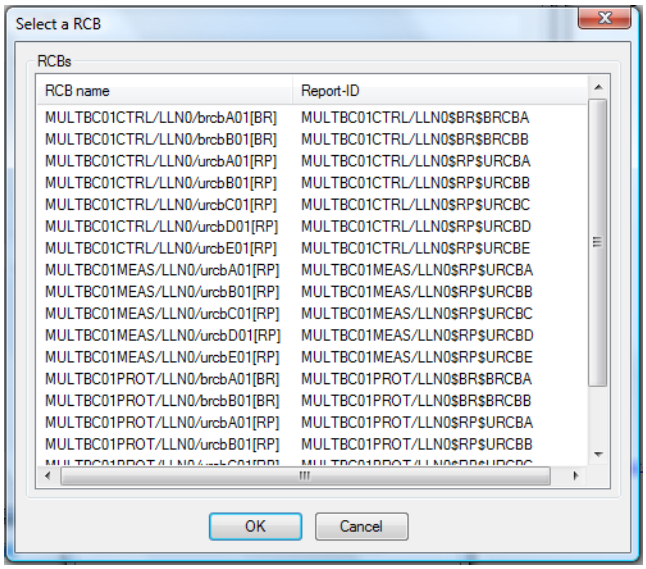
	▶ TrgOps ▶ OptFields `Inactive`: The driver writes the data attributes of the RCB during activation. You can activate/deactivate the following trigger options regardless of one another. ▸ `TrgOp: data-change` ▸ `TrgOp: quality-change` ▸ `TrgOp: data-update` ▸ `TrgOp: integrity` ▸ `TrgOp: general-interrogation` Default: all `active` Note: Not all servers support TrgOps data-change and data-update together. TrgOp intergity can also lead to an unnecessary overload of communication if a an IntgPd (Integrity Period) that is too short was defined in the server for RCB. In case of doubt, set TrgOps: data-change + quality-change + general-interrogation.

Integrity Period	Time interval (IntgPd) in milliseconds in which the server sends an Integrity Report. Default: `7000` ms Note: not active if `TrgOp integrity` is deactivated or **Use preconfigured (SCL) options** is activated. Because an Integrity Report does not normally contain value changes, it is expressly recommended that only one single report on the server is activated with `TrgOp: integrity`. With an activated integrity report, the server can detect a connection failure more quickly. However zenon does not need this report.
Buffer Time	Time interval (BufTime) in milliseconds in which the server collects the data for a report. Default: `500 ms` Note: not active if **Use preconfigured (SCL) options** is activated.
OptFlds	Opens the dialog to configure the Optional Fields (on page 28) of the report. Note: not active if **Use preconfigured (SCL) options** is activated.

CLOSE DIALOG

Parameters	Description
OK	Applies settings and closes the dialog.
Cancel	Discards all changes and closes the dialog.

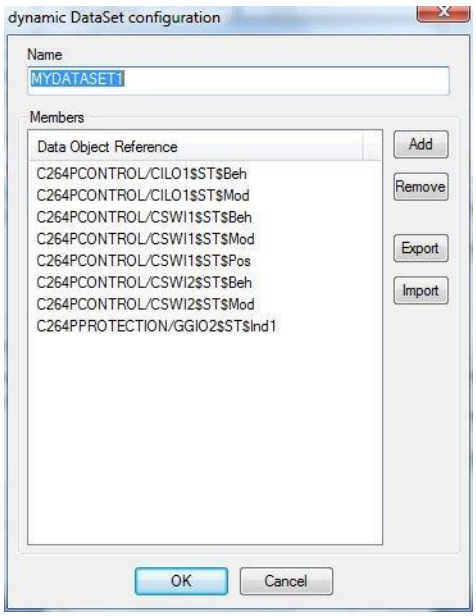
This list is displayed after you have browsed RCBs online on the IEC 61850 server:



Select the desired Report Control Block from the list.

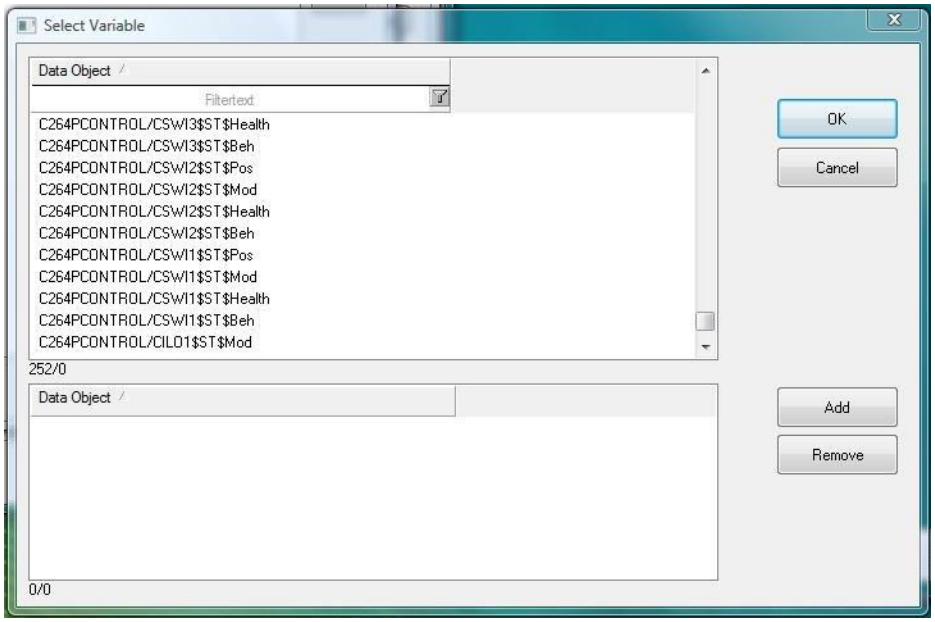
Dynamic DataSet configuration

Clicking on **New** or **Edit** in the **dynamic DataSets** list of the **Client configuration** (on page 29) dialog opens the dialog to define a dynamic data set.



Parameters	Description
Name	Name of the data set.
Members	List of pre-configured data objects.
Add	Starts the browsing of Data Objects from the variable list of the zenon project and displays all data objects for which a variable has already been created in the project. Note: The driver does not browse the data objects from the IEC61850 server, but from the variable list in the Editor. If the Runtime is also running at the same time on the same computer, the variables cannot be browsed.
Remove	Removes the selected entry from the list.
Export	Exports members to a TXT file.
Import	Imports members to a TXT fle. In the open dialog: ▸ Select the corresponding file. ▸ Select the desired variables. ▸ Confirm the settings by clicking on OK.

The following list is displayed once you have browsed suitable project variables:



The driver displays all data objects - that are appropriate for data reporting - from the variable list of the zenon project, except those that have already been selected for the data set. In the upper list, you can filter and select the listed variables. Multiple selection is possible (with the Alt or Shift key held down).

Clicking on the **Add** button adds your selection to the lower list. Objects that have already been selected are kept. Use the **OK** button to add the selected variables to the **members** of the dynamic data set.

6. Creating variables

This is how you can create variables in the zenon Editor:

6.1 Creating variables in the Editor

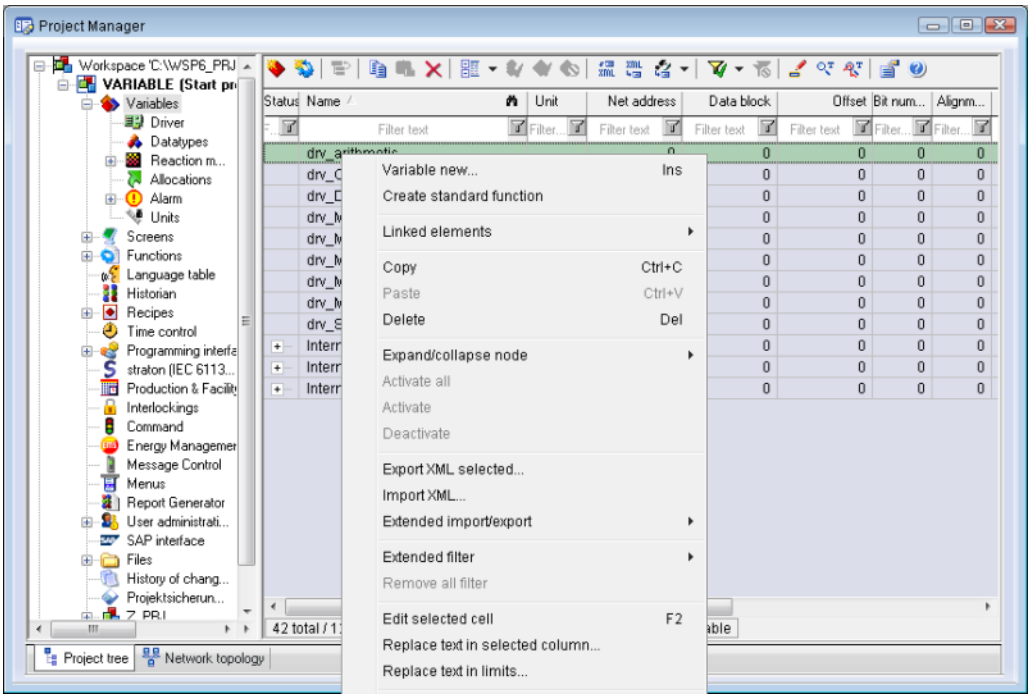
Variables can be created:

- as simple variables
- in arrays (main.chm::/15262.htm)
- as structure variables (main.chm::/15278.htm)

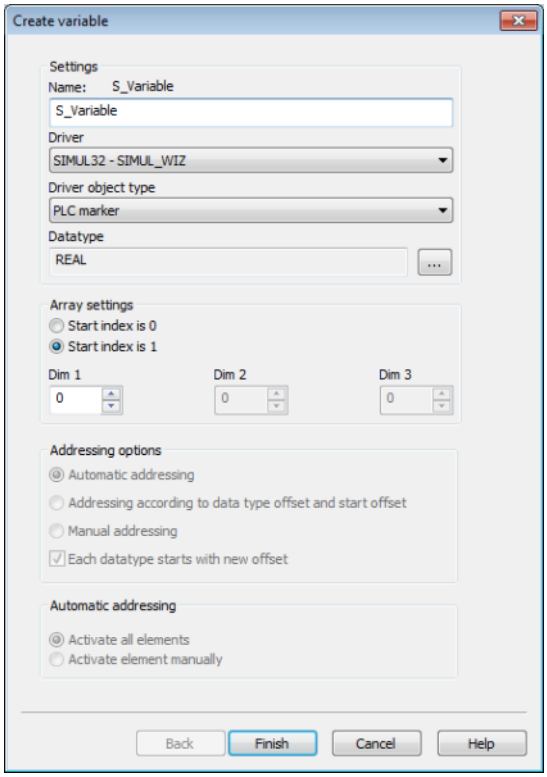
VARIABLE DIALOG

To create a new variable, regardless of which type:

1. Select the **New variable** command in the **Variables** node in the context menu



2. The dialog for configuring variables is opened
3. configure the variable
4. The settings that are possible depends on the type of variables



Property	Description
Name	Distinct name of the variable. If a variable with the same name already exists in the project, no additional variable can be created with this name. Maximum length: 128 character Attention: The characters **#** and **@** are not permitted in variable names. If non-permitted characters are used, creation of variables cannot be completed and the **Finish** button remains inactive. Note: For some drivers, the addressing is possible over the property **Symbolic address**, as well.
Drivers	Select the desired driver from the drop-down list. Note: If no driver has been opened in the project, the driver for internal variables (**Intern.exe** (**Main.chm::/Intern.chm::/Intern.htm**)) is automatically loaded.
Driver object type (cti.chm::/28685.htm)	Select the appropriate driver object type from the drop-down list.
Data type	Select the desired data type. Click on the ... button to open the selection dialog.
Array settings	Expanded settings for array variables. You can find details in the Arrays chapter.
Addressing options	Expanded settings for arrays and structure variables. You can find details in the respective section.
Automatic element activation	Expanded settings for arrays and structure variables. You can find details in the respective section.

SYMBOLIC ADDRESS

The **Symbolic address** property can be used for addressing as an alternative to the **Name** or **Identification** of the variables. Selection is made in the driver dialog; configuration is carried out in the variable property. When importing variables of supported drivers, the property is entered automatically.

Maximum length: `1024` characters.

INHERITANCE FROM DATA TYPE

Measuring range, **Signal range** and **Set value** are always:

- derived from the datatype
- Automatically adapted if the data type is changed

Note for signal range: If a change is made to a data type that does not support the set **signal range**, the **signal range** is amended automatically. For example, for a change from **INT** to **SINT**, the **signal range** is changed to `127`. The amendment is also carried out if the **signal range** was not inherited from the data type. In this case, the **measuring range** must be adapted manually.

6.2 Addressing

The addressing is based on the variable names.

Name	Description
Name	Freely definable name Attention: The name is used for addressing the variables and must be clear in each control system project. As a result, the server name is also taken into account if the driver identifies the connection with the help of the **Net Address** property. Addressing is symbolic by means of name. The name fields are automatically set during online import.
Identification	Enter any desired text; for example resource name, comments, ... Can be used for addressing (optional). Addressing is symbolic by means of identification. The identification fields are automatically set during online import.
Net address	Network address of variable. This address refers to the network address in the connection list of the driver configuration. It defines the IEC 61850 server on which variables are saved.
Data block	Not used for this driver
Offset	Not used for this driver
Alignment	Not used for this driver
Bit number	Not used for this driver
String length	Only available for String variables: Maximum number of characters from which a variable can exist.
Driver connection/Driver Object Type	Dependent on the type of variable, the object type is selected when the variable is created; the type can be changed here later. For more information, see also the driver objects (on page 45).
Driver connection/Data Type	Data type of variables that were selected when the variable was created; the type can be changed here later. Attention: If you subsequently change the data type here, you must check all variable properties, such as value range for example, and change them if necessary.
Driver connection/Priority	not used for this driver The driver does not support cyclically-poling communication in priority classes.
Symbolic address	Addressing is symbolic by means of symbolic address. The symbolic address fields are automatically set during online import.

SYNTAX NAMING CONVENTION

The naming convention is based on the following syntax: `SERVER!LD/LN/DATA/DataAttr[FC]`

For this, the following applies:

- The server name (on page 21) is separated by the exclamation mark (**!**) from the following identifiers.
- All additional identifiers to `DataAttr` are separated by a slash (/).
- Only if an identifier is a structure, their elements are separated by a dot (**.**); for example: `SERVER!LD/LN/DATA/DataAttr.item[FC].`

The designations are:

Parameters	Description
SERVER	Freely definable name of the server. Name should make sure that all variable names are identical. Recommendation: Use the same names as defined in driver dialog Dialog Server (on page 21). The driver however does not recognize the used connection via the **Server name** but via settings the **Net address**.
LD	Name of the logical device.
LN	Name of the logical node.
DATA	Name of the 'data object', see Appendix B - data objects / data attributes
DataAttr	Name of the 'data attribute', see Appendix B - data objects / data attributes
FC	"functional constraint" of the "data attribute", see Appendix A - Description of the Functional Constraints (FCs)

Attention

*The naming convention for variables complies with the **ObjectReference** defined in standard IEC61850-7-1.*

*In the standard a slash (/) is only used as separator between **LD** and **LD** and otherwise separated by dots (**.**). If **DATA** or **DataAttr** is a structure, dots make it impossible to differentiate between **DATA** and structure. Therefore an own naming conversion is used for the variable addressing.*

EXAMPLE:

The logical node "XCBR1" (circuit breaker 1) has, among other things, a data object "Pos", which is a CDC (Common Data Class) "Controllable Double Point" (DPC). This data point 'Pos' has, in turn, attributes 'stVal', 'q' and 't', the name and semantics of which is specific due to its CDC, which is defined in IEC61850-7-3.

Data attribute	Name	IEC 61850 Standard
XCBR1/Pos	Switching position	In accordance with LN definition XCBR in 61850-7-4
XCBR1/Pos/stVal[ST]	position value	In accordance with CDC definition DPC in 61850-7-3
XCBR1/Pos/q[ST]	Quality	In accordance with CDC definition DPC in 61850-7-3
XCBR1/Pos/t[ST]	Timestamp	In accordance with CDC definition DPC in 61850-7-3
XCBR1/Pos/Oper.ctlVal[CO]	Value of the command	In accordance with 'Oper' type definition in 61850-8-1, Table E.9

All objects in the variable model are constructed in this way. For this reason, the driver knows, for example, that the attributes 'q' and 't' contain quality and timestamps for the variables in 'stVal'. This allows access to all attributes of the each object in the variable model. Existing values for quality and time stamp are automatically assigned to the actual values of the driver.

Operate command: Thanks to the specific variable nomenclature, the driver automatically notices that an operate command is to be created if a variable is written to, for example with the name '*/Oper.ctlVal[CO]' (Operation.ControlValue). It the automatically sends the required sequence of telegrams to the control object.

TIME STAMP AND QUALITY

Time stamp and quality attribute are always parts of variables in zenon. Variables with attributes 'q' and 't' do not need to be created separately. The driver automatically uses the time stamp and the quality of the data object for its variables with **FC = ST and MX** (FC stands for Functional Constraint (on page 106)). Without taking into account whether a variable for attribute 't' was created, the time stamp received is used for all **ST** and **MX** variables, for example the **stVal**variable for CEL, AML or Historian. Selected quality bits that are received in attribute 'q' are also assigned the status bits of the **stVal** variable (see also Driver-specific functions (on page 69)).

6.3 Driver objects and datatypes

Driver objects are areas available in the PLC, such as markers, data blocks etc. Here you can find out which driver objects are provided by the driver and which IEC data types can be assigned to the respective driver objects.

6.3.1 Driver objects

The following object types are available in this driver:

Driver object type	Channel type	Read	Write	Supported data types	Comment
PLC marker	8	X	X	`REAL, LREAL, WSTRING, DINT, UDINT, BOOL, INT, UINT, STRING, USINT, SINT`	Data attribute variables
File Transfer	9	X	X	`STRING`	Command and return variables for the file transfer. Note: The string length should correspond to the maximum length of the file names (including the path) in the PLC; a maximum of 260 characters for `*!Command` variable for example, (but more for `*!Directory`). You can find more information in the chapter File transfer (on page 87)
Connection state	36	X	--	`UDINT`	Internal variables of this object type show the status of the connection to the 850 server. The variable must have `ConnectionState` as a reference and the correct **Net address**. Example: **Name** or **Symbolic address**: `*!ConnectionState`. You can find more information in the chapter Establishment of a connection and detection of a connection failure (on page 70).
Command Info	11	X	--	`UINT`	Internal variables show the current status of the command; transferred from the driver to Runtime (not by the PLC).

					You can find more information in the chapter Additional Cause Diagnosis (on page 75).
Service tracking	64	X	--	`REAL, LREAL, DINT, UDINT, BOOL, INT, UINT, STRING, USINT, SINT`	Additional Service tracking (*[SR]) variables. You can find more information in the chapter Service tracking (Main.chm::/IEC850.chm::/117281.htm).
Settings	65	X	X	`USINT`	In accordance with Standard 61850-8-1, 8.1.3.7: ▶ `TimeQuality` ▶ `TimeAccuracy` For the T-attribute in `SBOw`, `Oper` and `Cancel`. You can find more information in the chapter TimeQuality und TimeAccuracy.
Driver variable	35	X	X	`BOOL, SINT, USINT, INT, UINT, DINT, UDINT, REAL, STRING`	Variables for the static analysis of the communication; is transferred between driver and Runtime (not to the PLC). Note: The addressing and the behavior is the same for most zenon drivers. You can find more information in the chapter on **driver variables (on page 62).**

6.3.2 Assignment of data types

All variables in zenon are derived from IEC data types. The following table compares the IEC datatypes with the datatypes of the PLC.

ASSIGNMENT OF DATA TYPES FROM THE PLC TO ZENON DATA TYPES

PLC	zenon	Data type	Remark
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	
INT64	DINT	3	Attention: In the variable value, only the lowest 32-bit is applied. The other bits are cut off and not taken into account.
INT128	DINT	3	Attention: In the variable value, only the lowest 32-bit is applied. The other bits are cut off and not taken into account.
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	Coding is carried out in UTF-8
PACKED LIST	UDINT, STRING	4, 12	`MMS Bitstring`, inverted bit order
TIMESTAMP	LREAL	6	Only octets 0..7 (see also TimeQuality Bit mapping (on page 69))
EntryTime	LREAL	6	
TriggerConditions	UDINT, STRING	4, 12	`MMS Bitstring`, inverted bit order

Data type: The property **Data type** is the internal numerical name of the data type. It is also used for the extended DBF import/export of the variables.

MMS Bitstring: The data attribute types that were assigned to the `MMS Bitstring` in accordance with the standard (see IEC61850-8-1), `CODED ENUM`, `PACKED LIST` und `TrigerConditions`for example, correspond to the UDINT data type in zenon. Whereby in the case of `PACKED LIST` and `TrigerConditions`, the bit 0 has the highest value in its octet. The driver also allows bit string variables with the data type STRING instead of UDINT to be inserted manually. The value of the string only contains the characters "0" and "1" in a sequence that is the same as that of the bits defined in the standard and only in the length of the bit string, for example the value of the TrgOps Attribute - Bitstring(6) - as String "011111" (in UDINT=124) means that all trigger condition bits except bit 0 - `'Reserved'` are set and the value "000010" (in UDINT=8) - only bit 4 - `'intergity'`.

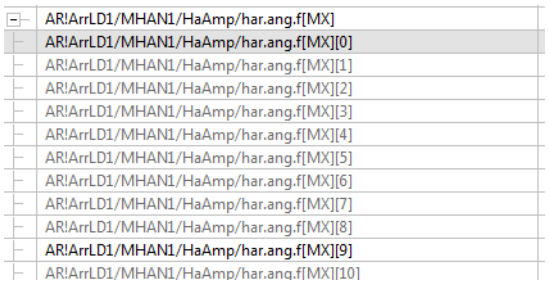
"ARRAY OF" TYPE DATA ATTRIBUTES

If there is an array level in the IEC61850 Data Classs, this level is placed at the end of the variable structure for display in zenon. This means that, from an **ARRAY OF** that may have nested structures, there are one or several arrays of scaled values in the zenon variable model. This remapping is carried out at the start of the configuration time during online import and in the runtime environment during the allocation between the zenon variable list and the IEC61850 object model.

As a result of this, there is the restriction that only one array level is supported. The support for arrays is also limited to the data attribute.

Example

The data attribute **HaAmp.har** of the Common Data Class (CDC, -7-3) **MHAN** - as defined in the IEC61850 standard: `ARRAY[0..numHar] of Vector` – becomes the following array in zenon:



In addition, there are of course also the arrays for **ang.i**, **mag.f** and **mag.i**.

6.4 Creating variables by importing

Variables can also be imported by importing them. The XML and DBF import is available for every driver.

Information

You can find details on the import and export of variables in the Import-Export (main.chm::/13028.htm) manual in the Variables (main.chm::/13045.htm) section.

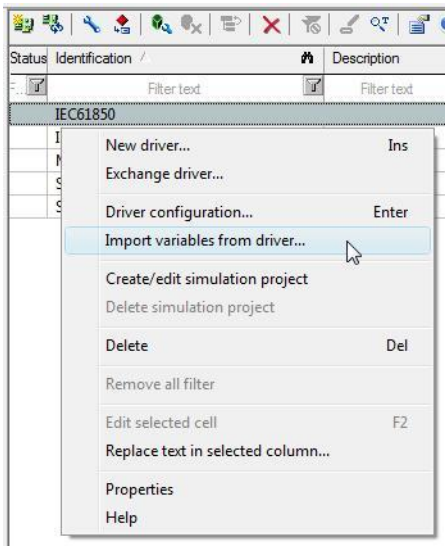
6.4.1 Online import

Variables in a zenon project can be created with the import of the driver. The driver can browse the variable list of the IEC 61850 server.

Attention

If Runtime is running on the same computer at the same time, the variables in the editor cannot be imported by the PLC, because the driver in Runtime uses the given settings for communication.

You call up the import using the context menu of the driver in the driver list.



A dialog is opened.

SELECT A SERVER

Parameters	Description
Server	List of the connections to IEC61850 servers created in the driver. You can read further information about the configuration of server connections in the Connections (on page 20) chapter.
Source	Source of the data model: ▶ PLC Browsing Variables are imported from an IEC61850 server that is available online. ▶ File Browsing Variables are imported from an SCL file. You can read further information about the SCL file in the IEC61850-6 standard.
OK	Applies settings and closes the dialog.
Cancel	Discards all changes and closes the dialog.

PROCEDURE FOR PLC BROWSING

- In the dialog that opens, select the PLC Browsing option and select a server from which the variables can be imported.
- The connection to the server is established and the list of existing data points (LDs, LNs, data objects, data attributes) is read.
- If the reading in has ended, an important dialog opens in which you can filter the browsed variables (capitalization, for example *XCBR1*stVal*), select them (including multiple selection) and add them from the upper to lower list.
- You create the selected variables in the project using the OK button.

Attention

The driver creates variables with default value ranges, for example a UDINT with <0... 4294967295> and a LREAL with <-1000..1000>, which may not correspond to the value ranges of the attributes, for example stVal-attribute UDINT <0..3> or t-attribute LREAL<0.. 2147483647> with three decimal places for milliseconds. Therefore ensure after import after that all variables have the correct value range properties and the string is the right length.

IMPORT OF ADDRESSES WITH MORE THAN 128 CHARACTERS

When reading (browsing) the data model, the driver generates the addresses of the variables. This is displayed in the '`Symbolic address`' column. The generated address is Server!ObjectReference[FC].

If this address is longer than the 128 characters permitted for the variable name, it is shortened to 124 characters and supplemented with `_###`. `###` stands for a serial number. This number counts up from `001` to `999`.

If new variables are created, whose generated name was longer than 128 characters, this shortened number supplemented with figures is used as variable **Name** and **Identification**. The unshortened address is used as **Symbolic address**.

Attention

*In Runtime, more than 128 characters is only supported if, in the driver configuration, in the Basic settings (on page 14) dialog, **Symbolic address** is selected for the addressing.*

RECOMMENDATION

RENAME THE VARIABLES AFTER THE IMPORT. OTHERWISE NAMING CONFLICTS COULD OCCUR AFTER ANOTHER IMPORT."ARRAY OF" IMPORT

As a result of the limitation of only one array level being supported (on page 47), the support of arrays is also limited to the data attributes.

These **ARRAY OF** variables are created as zenon arrays when imported.

Example

The data attribute `HaAmp.har` of the Common Data Class **MHAN** - as defined in the IEC61850 standard: `ARRAY[0..numHar] of Vector` –becomes up to four arrays in zenon, with the same respective number of elements:

- ***/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]**

SERVICE TRACKING IMPORT

Service tracking (**Main.chm::/IEC850.chm::/117281.htm**) is only offered if one or more elements with FC = SR are present on the Logical Device. For example, one or more Logical Nodes LTRK. For import, only Service tracking is automatically offered for Control Services. Control Services means: only Service tracking command execution information.

If there is Service tracking on the server, a proposal for the creation of additional service tracking variables is created for each data object that contains the data attributes with FC = CO.

Example

Data object with command execution: S1!Device/Node/Pos/Oper.ctlVal[CO]

These additional variables no longer exist on the server but are additionally created in the respective data object with a possible command execution.

The variables that are actually created as additional service tracking variables are selected from this list of proposals.
The mane of this proposal comprises the following:
`Datenobjektreferenz_Servicetracking-Datenattributname[SR]`. As a service tracking data attribute name, the data attributes that are present in the Logical Node LRTK for Control Services are proposed.

Example

Name: S1!Device/Node/Pos_errorCode[SR].

The data type of the additional service tracking variable corresponds to the data type of the SR data attribute that is present in the Logical Node LRTK.
The driver object type is Service tracking.

In addition, a string variable *_allCTS[SR] is also offered. This string variable contains the complete Service tracking (Main.chm::/IEC850.chm::/117281.htm) Information and can be evaluated with the help of zenon Logic or VBA/VSTA.

6.4.2 Offline import

Variables can also be imported offline. The following file endings are accepted:

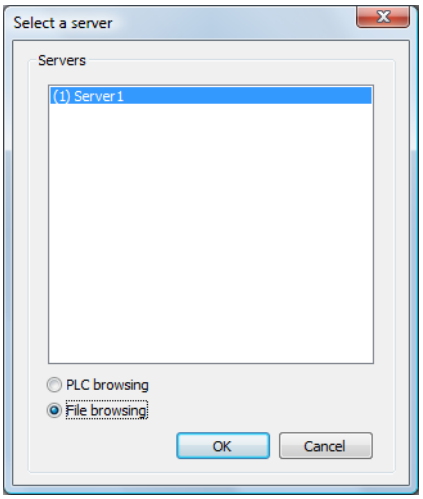
- `*.xml`
- `*.scl`
- `*.icd`
- `*.cid`
- `*.scd`

The driver only imports DOI elements online. The **ReportControl** properties (report control block) and **SettingControl** (Setting Control Block) are not supported during offline import. You can create the variables for these attributes by means of online import or manually.

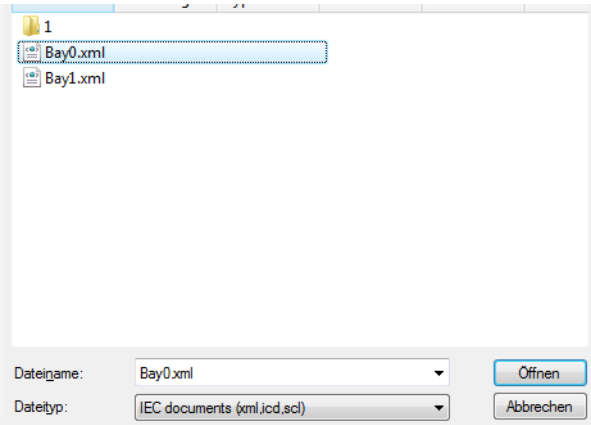
> **Attention**
>
> *The driver can then only import from the data model of the IED if the data model is compliant with the IEC61850 standard. All non-compliant elements are then ignored and cannot be selected in the import dialog. For example, the driver ignores the Logical Nodes that are linked directly to the IED nodes, instead of to the Logical Device nodes.*

To import variables from a file:

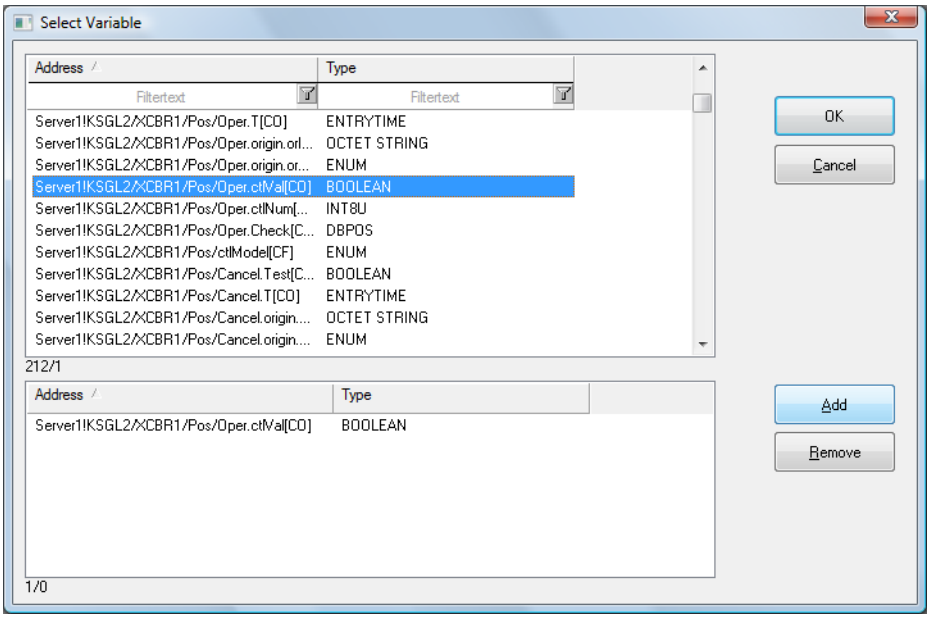
1. Select the **File browsing** option in server selection



2. Select the import file.



3. The following steps for the selection of variables are similar to online import.



IMPORT OF ADDRESSES WITH MORE THAN 128 CHARACTERS

When reading (browsing) the data model, the driver generates the addresses of the variables. This is displayed in the '`Symbolic address`' column. The generated address is Server!ObjectReference[FC].

If this address is longer than the 128 characters permitted for the variable name, it is shortened to 124 characters and supplemented with `_###`. `###` stands for a serial number. This number counts up from `001` to `999`.

If new variables are created, whose generated name was longer than 128 characters, this shortened number supplemented with figures is used as variable **Name** and **Identification**. The unshortened address is used as **Symbolic address**.

Attention

In Runtime, more than 128 characters is only supported if, in the driver configuration, in the Basic settings (on page 14) dialog, ***Symbolic address*** *is selected for the addressing.*

RECOMMENDATION

RENAME THE VARIABLES AFTER THE IMPORT. OTHERWISE NAMING CONFLICTS COULD OCCUR AFTER ANOTHER IMPORT.SERVICE TRACKING IMPORT

Service tracking is only offered if one or more elements with FC = SR are present on the Logical Device. For example, one or more Logical Nodes LTRK. For import, only Service tracking is automatically offered for Control Services. Control Services means: only Service tracking command execution information.

If there is Service tracking in the SCL file, a proposal for the creation of additional service tracking variables is created for each data object that contains the data attributes with FC = CO.

Example

Data object with command execution: S1!Device/Node/Pos/Oper.ctlVal[CO]

These additional variables no longer exist on the server but are additionally created in the respective data object with a possible command execution.

The variables that are actually created as additional service tracking variables are selected from this list of proposals.
The mane of this proposal comprises the following:
Datenobjektreferenz_Servicetracking-Datenattributname[SR]. As a service tracking data attribute name, the data attributes that are present in the Logical Node LRTK for Control Services are proposed.

Example

Name: S1!Device/Node/Pos_errorCode[SR]

The data type of an additional service tracking variable corresponds to the data type of the SR data attribute, that is in the Logical Node LRTK.
The driver object type is Service tracking.

In addition, a string variable `*_allCTS[SR]` is also offered. This string variable contains the complete service tracking Information and can be evaluated with the help of zenon Logic or VBA/VSTA.

6.4.3 XML import

For the import/export of variables the following is true:

- The import/export must not be started from the global project.
- The start takes place via:
  - Context menu of variables or data typ in the project tree
  - or context menu of a variable or a data type
  - or symbol in the symbol bar variables

> **Attention**
>
> *When importing/overwriting an existing data type, all variables based on the existing data type are changed.*
>
> *Example:*
>
> *There is a data type XYZ derived from the type `INT`with variables based on this data type. The XML file to be imported also contains a data type with the name XYZ but derived from type `STRING`. If this data type is imported, the existing data type is overwritten and the type of all variables based on it is adjusted. I.e. the variables are now no longer `INT` variables, but `STRING` variables.*

6.4.4 DBF Import/Export

Data can be exported to and imported from dBase.

> **Information**
>
> *Import and Export via CSV or dBase supported; no driver specific variable settings, such as formulas. Use export/import via XML for this.*

IMPORT DBF FILE

To start the import:

1. right-click on the variable list
2. in the drop-down list of **Extended export/import...** select the **Import dBase** command
3. follow the import assistant

The format of the file is described in the chapter File structure.

Information

Note:

- Driver object type and data type must be amended to the target driver in the DBF file in order for variables to be imported.
- dBase does not support structures or arrays (complex variables) at import.

EXPORT DBF FILE

To start the export:

1. right-click on the variable list
2. in the drop-down list of **Extended export/import...** select the **Export dBase...** command
3. follow the export assistant

Attention

DBF files:

- must correspond to the 8.3 DOS format for filenames (8 alphanumeric characters for name, 3 character suffix, no spaces)
- must not have dots (.) in the path name.
  e.g. the path `C:\users\John.Smith\test.dbf` is invalid.
  Valid: `C:\users\JohnSmith\test.dbf`
- must be stored close to the root directory in order to fulfill the limit for file name length including path: maximum 255 characters

The format of the file is described in the chapter File structure.

Information

dBase does not support structures or arrays (complex variables) at export.

File structure of the dBase export file

The dBaseIV file must have the following structure and contents for variable import and export:

Attention

dBase does not support structures or arrays (complex variables) at export.

DBF files must:

- conform with their name to the 8.3 DOS format (8 alphanumeric characters for name, 3 characters for extension, no space)
- Be stored close to the root directory (Root)

STRUCTURE

Identification	Type	Field size	Comment
KANALNAME	Char	128	Variable name. The length can be limited using the MAX_LAENGE entry in **project.ini**.
KANAL_R	C	128	The original name of a variable that is to be replaced by the new name entered under "VARIABLENNAME" (field/column must be entered manually). The length can be limited using the MAX_LAENGE entry in **project.ini**.
KANAL_D	Log	1	The variable is deleted with the 1 entry (field/column has to be created by hand).
TAGNR	C	128	Identification. The length can be limited using the MAX_LAENGE entry in **project.ini**.
EINHEIT	C	11	Technical unit
DATENART	C	3	Data type (e.g. bit, byte, word, ...) corresponds to the data type.
KANALTYP	C	3	Memory area in the PLC (e.g. marker area, data area, ...) corresponds to the driver object type.
HWKANAL	Num	3	Bus address
BAUSTEIN	N	3	Datablock address (only for variables from the data area of the PLC)
ADRESSE	N	5	Offset
BITADR	N	2	For bit variables: bit address For byte variables: 0=lower, 8=higher byte For string variables: Length of string (max. 63 characters)
ARRAYSIZE	N	16	Number of variables in the array for index variables ATTENTION: Only the first variable is fully available. All others are only available for VBA or the Recipegroup Manager

LES_SCHR	L	1	Write-Read-Authorization 0: Not allowed to set value. 1: Allowed to set value.
MIT_ZEIT	L	1	time stamp in zenon (only if supported by the driver)
OBJEKT	N	2	Driver-specific ID number of the primitive object comprises TREIBER-OBJEKTTYP and DATENTYP
SIGMIN	Float	16	Non-linearized signal - minimum (signal resolution)
SIGMAX	F	16	Non-linearized signal - maximum (signal resolution)
ANZMIN	F	16	Technical value - minimum (measuring range)
ANZMAX	F	16	Technical value - maximum (measuring range)
ANZKOMMA	N	1	Number of decimal places for the display of the values (measuring range)
UPDATERATE	F	19	Update rate for mathematics variables (in sec, one decimal possible) not used for all other variables
MEMTIEFE	N	7	Only for compatibility reasons
HDRATE	F	19	HD update rate for historical values (in sec, one decimal possible)
HDTIEFE	N	7	HD entry depth for historical values (number)
NACHSORT	L	1	HD data as postsorted values
DRRATE	F	19	Updating to the output (for zenon DDE server, in [s], one decimal possible)
HYST_PLUS	F	16	Positive hysteresis, from measuring range
HYST_MINUS	F	16	Negative hysteresis, from measuring range
PRIOR	N	16	Priority of the variable
REAMATRIZE	C	32	Allocated reaction matrix
ERSATZWERT	F	16	Substitute value, from measuring range
SOLLMIN	F	16	Minimum for set value actions, from measuring range
SOLLMAX	F	16	Maximum for set value actions, from measuring range
VOMSTANDBY	L	1	Get value from standby server; the value of the variable is not requested from the server but from the Standby Server in redundant networks
RESOURCE	C	128	Resources label. Free string for export and display in lists. The length can be limited using the MAX_LAENGE entry in **project.ini**.
ADJWVBA	L	1	Non-linear value adaption: `0`: Non-linear value adaption is used `1`: Non-linear value adaption is not used

ADJZENON	C	128	Linked VBA macro for reading the variable value for non-linear value adjustment.
ADJWVBA	C	128	ed VBA macro for writing the variable value for non-linear value adjustment.
ZWREMA	N	16	Linked counter REMA.
MAXGRAD	N	16	Gradient overflow for counter REMA.

Attention

When importing, the driver object type and data type must be amended to the target driver in the DBF file in order for variables to be imported.

LIMIT VALUE DEFINITION

Limit definition for limit values 1 to 4, or status 1 to 4:

Identification	Type	Field size	Comment
AKTIV1	L	1	Limit value active (per limit value available)
GRENZWERT1	F	20	technical value or ID number of a linked variable for a dynamic limit value (see VARIABLEx) (if VARIABLEx is 1 and here it is −1, the existing variable linkage is not overwritten)
SCHWWERT1	F	16	Threshold value for limit value
HYSTERESE1	F	14	Is not used
BLINKEN1	L	1	Set blink attribute
BTB1	L	1	Logging in CEL
ALARM1	L	1	Alarm
DRUCKEN1	L	1	Printer output (for CEL or Alarm)
QUITTIER1	L	1	Must be acknowledged
LOESCHE1	L	1	Must be deleted
VARIABLE1	L	1	Dyn. limit value linking the limit is defined by an absolute value (see field GRENZWERTx).
FUNC1	L	1	Functions linking
ASK_FUNC1	L	1	Execution via Alarm Message List
FUNC_NR1	N	10	ID number of the linked function (if “-1” is entered here, the existing function is not overwritten during import)
A_GRUPPE1	N	10	Alarm/Event Group
A_KLASSE1	N	10	Alarm/Event Class
MIN_MAX1	C	3	Minimum, Maximum
FARBE1	N	10	Color as Windows coding
GRENZTXT1	C	66	Limit value text
A_DELAY1	N	10	Time delay
INVISIBLE1	L	1	Invisible

Expressions in the column "Comment" refer to the expressions used in the dialog boxes for the definition of variables. For more information, see chapter Variable definition.

6.5 Driver variables

The driver kit implements a number of driver variables. These are divided into:

- Information
- Configuration
- Statistics and
- Error message

The definitions of the variables implemented in the driver kit are available in the import file **drvvar.dbf** (on the installation medium in the `\Predefined\Variables` folder) and can be imported from there.

Note: Variable names must be unique in zenon. If driver variables are to be imported from **drvvar.dbf** again, the variables that were imported beforehand must be renamed.

Information

Not every driver supports all driver variants.

For example:

- Variables for modem information are only supported by modem-compatible drivers
- Driver variables for the polling cycle only for pure polling drivers
- Connection-related information such as ErrorMSG only for drivers that only edit one connection at a a time

INFORMATION

Name from import	Type	Offset	Description
MainVersion	UINT	0	Main version number of the driver.
SubVersion	UINT	1	Sub version number of the driver.
BuildVersion	UINT	29	Build version number of the driver.
RTMajor	UINT	49	zenon main version number
RTMinor	UINT	50	zenon sub version number
RTSp	UINT	51	zenon Service Pack number
RTBuild	UINT	52	zenon build number
LineStateIdle	BOOL	24.0	TRUE, if the modem connection is idle
LineStateOffering	BOOL	24.1	TRUE, if a call is received
LineStateAccepted	BOOL	24.2	The call is accepted
LineStateDialtone	BOOL	24.3	Dialtone recognized
LineStateDialing	BOOL	24.4	Dialing active
LineStateRingBack	BOOL	24.5	While establishing the connection
LineStateBusy	BOOL	24.6	Target station is busy

LineStateSpecialInfo	BOOL	24.7	Special status information received
LineStateConnected	BOOL	24.8	Connection established
LineStateProceeding	BOOL	24.9	Dialing completed
LineStateOnHold	BOOL	24.10	Connection in hold
LineStateConferenced	BOOL	24.11	Connection in conference mode.
LineStateOnHoldPendConf	BOOL	24.12	Connection in hold for conference
LineStateOnHoldPendTransfer	BOOL	24.13	Connection in hold for transfer
LineStateDisconnected	BOOL	24.14	Connection terminated.
LineStateUnknow	BOOL	24.15	Connection status unknown
ModemStatus	UDINT	24	Current modem status
TreiberStop	BOOL	28	Driver stopped For `driver stop`, the variable has the value `TRUE` and an **OFF** bit. After the driver has started, the variable has the value `FALSE` and no **OFF** bit.
SimulRTState	UDINT	60	Informs the status of Runtime for driver simulation.

CONFIGURATION

Name from import	Type	Offset	Description
ReconnectInRead	BOOL	27	If TRUE, the modem is automatically reconnected for reading
ApplyCom	BOOL	36	Apply changes in the settings of the serial interface. Writing to this variable immediately results in the method SrvDrvVarApplyCom being called (which currently has no further function).
ApplyModem	BOOL	37	Apply changes in the settings of the modem. Writing this variable immediately calls the method SrvDrvVarApplyModem. This closes the current connection and opens a new one according to the settings **PhoneNumberSet** and **ModemHwAdrSet**.

PhoneNumberSet	STRING	38	Telephone number, that should be used
ModemHwAdrSet	DINT	39	Hardware address for the telephone number
GlobalUpdate	UDINT	3	Update time in milliseconds (ms).
BGlobalUpdaten	BOOL	4	TRUE, if update time is global
TreiberSimul	BOOL	5	TRUE, if driver in sin simulation mode
TreiberProzab	BOOL	6	TRUE, if the variables update list should be kept in the memory
ModemActive	BOOL	7	TRUE, if the modem is active for the driver
Device	STRING	8	Name of the serial interface or name of the modem
ComPort	UINT	9	Number of the serial interface.
Baudrate	UDINT	10	Baud rate of the serial interface.
Parity	SINT	11	Parity of the serial interface
ByteSize	USINT	14	Number of bits per character of the serial interface Value = 0 if the driver cannot establish any serial connection.
StopBit	USINT	13	Number of stop bits of the serial interface.
Autoconnect	BOOL	16	TRUE, if the modem connection should be established automatically for reading/writing
PhoneNumber	STRING	17	Current telephone number
ModemHwAdr	DINT	21	Hardware address of current telephone number
RxIdleTime	UINT	18	Modem is disconnected, if no data transfer occurs for this time in seconds (s)

WriteTimeout	UDINT	19	Maximum write duration for a modem connection in milliseconds (ms).
RingCountSet	UDINT	20	Number of ringing tones before a call is accepted
ReCallIdleTime	UINT	53	Waiting time between calls in seconds (s).
ConnectTimeout	UINT	54	Time in seconds (s) to establish a connection.

STATISTICS

Name from import	Type	Offset	Description
MaxWriteTime	UDINT	31	The longest time in milliseconds (ms) that is required for writing.
MinWriteTime	UDINT	32	The shortest time in milliseconds (ms) that is required for writing.
MaxBlkReadTime	UDINT	40	Longest time in milliseconds (ms) that is required to read a data block.
MinBlkReadTime	UDINT	41	Shortest time in milliseconds (ms) that is required to read a data block.
WriteErrorCount	UDINT	33	Number of writing errors
ReadSucceedCount	UDINT	35	Number of successful reading attempts

MaxCycleTime	UDINT	22	Longest time in milliseconds (ms) required to read all requested data.
MinCycleTime	UDINT	23	Shortest time in milliseconds (ms) required to read all requested data.
WriteCount	UDINT	26	Number of writing attempts
ReadErrorCount	UDINT	34	Number of reading errors
MaxUpdateTimeNormal	UDINT	56	Time since the last update of the priority group **Normal** in milliseconds (ms).
MaxUpdateTimeHigher	UDINT	57	Time since the last update of the priority group **Higher** in milliseconds (ms).
MaxUpdateTimeHigh	UDINT	58	Time since the last update of the priority group **High** in milliseconds (ms).
MaxUpdateTimeHighest	UDINT	59	Time since the last update of the priority group **Highest** in milliseconds (ms).
PokeFinish	BOOL	55	Goes to 1 for a query, if all current pokes were executed

ERROR MESSAGE

Name from import	Type	Offset	Description
ErrorTimeDW	UDINT	2	Time (in seconds since 1.1.1970), when the last error occurred.
ErrorTimeS	STRING	2	Time (in seconds since 1.1.1970), when the last error occurred.
RdErrPrimObj	UDINT	42	Number of the PrimObject, when the last reading error occurred.
RdErrStationsName	STRING	43	Name of the station, when the last reading error occurred.
RdErrBlockCount	UINT	44	Number of blocks to read when the last reading error occurred.

RdErrHwAdresse	DINT	45	Hardware address when the last reading error occurred.
RdErrDatablockNo	UDINT	46	Block number when the last reading error occurred.
RdErrMarkerNo	UDINT	47	Marker number when the last reading error occurred.
RdErrSize	UDINT	48	Block size when the last reading error occurred.
DrvError	USINT	25	Error message as number
DrvErrorMsg	STRING	30	Error message as text
ErrorFile	STRING	15	Name of error log file

7. IEC850 functions

Special specifications of the IEC 61850 standard were implemented as specific functions in the IEC 850 driver.

You can find detailed information in the respective

- Verbindungaufbau und Verbindungsausfallerkennung (on page 70)
- Commands (Control Model) (on page 72)

  commands - overview

  - Select und Cancel (on page 74)
  - Additional Cause Diagnosis (on page 75)
  - Service parameters of the command (on page 76)
- Service Tracking (Main.chm::/IEC850.chm::/117281.htm)
- Mapping of double point values (on page 82)
- Quality bits, time stamp and status bits of the variable (on page 84)
- TimeQuality and TimeAccuracy
- Filetransfer (on page 87)

  There are three functions implemented for the file transfer:

  - Request folder information

- Get file from server
- Delete file

LIMITATIONS

- Data type: **ARRAY OF**
  Only one array level is supported. The support for arrays is limited to the data attributes. It is not possible to write array elements. When reading by means of polling or reporting, as with other data attributes too, the complete Data Object (FCD reference), which contains the array attribute, is always read.
- Data types **INT64, INT128**:
  In the variable value, only the lowest 32-bit is applied.
  The other bits are cut off and not taken into account.
- For reports with Service Tracking, only datasets with references are supported.
- Conformance statement (PICS/MICS):
  You can find the information on supported or required services and data types in Appendix C (on page 117).

7.1 Establishment of a connection and detection of a connection failure

The driver only opens a connection to a Logical Device if at least one variable from the LD is created in the project.

Attention: Connection breakdowns cannot be detected if all variables are received by means of Reports . After a communication breakdown (due to a network failure, for example), the connection to the Logical Device is then only reestablished automatically if at least one variable of the LD is being polled.

Recommendation: To detect a connection failure, activate the **Automatic Watchdog** property in the driver configuration.

If **Automatic Watchdog** has been activated: the driver automatically starts to poll the `*/LLN0/Mod/stVal[ST]` data attribute as soon as all other data of the Logical Device only comes via reports, i.e. whilst all polled variables are not signed in. The read queries that are executed cyclically, in the cycle of the **Polling rate** driver setting force the operating system to check the TCP connection again and to report to the driver if the connection is disconnected.

CONNECTION STATE VARIABLE

A `Connection State` variable, if created, provides information on whether the driver has a TCP and then MMS connection to the server and whether the connection consists of primary or secondary IP addresses. Furthermore, you receive information on whether all configured static RCB assignments have been successfully registered.

This variable must be created with the `Connection state` **Driver object type** (on page 45). It must contain the correct network address of the connection, have the reference `*!ConnectionState` and the data type has to be **UDINT**.

The bits of the variable value mean the following:

For the primary TCP/IP connection to the 850 server:

Bit	Meaning	Value (hex)
1	`TCP_CONNECTED`	0x02
2	`TCP_CONNECTING`	0x04
3	`TCP_CONNECT_FAILED`	0x08
16	`MMS_ASSOCIATED`	0x10000
17	`MMS_RCB_ENABLE_FAILED`	0x20000

For the secondary TCP/IP connection:

Bit	Meaning	Value (hex)
5	`TCP_CONNECTED`	0x20
6	`TCP_CONNECTING`	0x40
7	`TCP_CONNECT_FAILED`	0x80
24	`MMS_ASSOCIATED`	0x1000000
25	`MMS_RCB_ENABLE_FAILED`	0x2000000

The `MMS_RCB_ENABLE_FAILED` bit is only set if one (or more) RCBs that have been configured in the driver configuration in **RCB assignment** (on page 33) could not be activated. This happens for example if the IEC61850 server does not write to RCB data attributes because another client is already using this report.

The `Connection State` variable does not provide an evaluation of whether it was possible to activate the URCBs via '**max. auto used URCBs** (on page 21)'.

Example

The standard process for the value change of the variable is:

TCP_CONNECT_FAILED => TCP_CONNECTING => TCP_CONNECTED => TCP_CONNECTED + MMS_ASSOCIATED

Or, in the event of a TCP error:

TCP_CONNECT_FAILED => TCP_CONNECTING => TCP_CONNECT_FAILED

Or, in the event of an error only once the RCBs are activated:

=> TCP_CONNECTED + MMS_ ASSOCIATED + MMS_RCB_ENABLE_FAILED

7.2 Commands (Control Model)

The driver supports the 'Direct Operate' and 'Select Before Operate' commands (SBO) with normal and increased security. To trigger a command, set a value to the `Oper.ctlVal` variable. The driver then automatically checks the actual Control Model and executes a complete command sequence accordingly. For the pending command, the required command sequences are automatically executed by the driver in the correct sequence accordingly.

Example: The driver first carries out a Select and then an Operate. No additional configuration steps are necessary for this.

Attention

*Do not create any variables that correspond to the data attributes */SBO, */SBOw* or */Cancel*.*

*To send a command to the server, only set the value of the */Oper.ctlVal* variable. The driver will automatically recognize that the command sequence must also contain a Select. A Select is then executed automatically.*

The variables with with Functional Constraint CO are 'write-only'. That means:

- These only have a value once it has been set in Runtime.
- These variables still do not have a value after starting in Runtime.
- They are reset to the value 0 after a connection has been lost and is established again.
- In Runtime, only the values of the CO variables `Oper.ctlVal`, `Oper.Check` and `Oper.Test` are displayed, other zenon variables with Functional Constraint CO, even if configured, are ignored by the driver.

- The driver supports remaining CO data attributes for communication with an 850 server, not for the exchange of data with Runtime.

WRITE SET VALUE TO OPER.CTLVAL

If a value is set to a project variable that corresponds to the `ctlVal` item in a data attribute with the name 'Oper' and it has the Functional Constraint CO or SP (`*/Oper.ctlVal[CO|SP]`, `*/Oper.ctlVal.i[CO]` or `*/Oper.ctlVal.f[CO]`), the driver automatically checks which Control Model is required for a command in this Data Object.
To do this, the data attribute `*/ctlModel[CF]` of this Data Object is read off first. Depending on the value of this data attribute, the respective Control Model is used and the command is executed in the corresponding procedure:

- `Value 0` – status only:
  No action, execution of the command not possible
- `Value 1` – direct control with normal security:
  An 'Operate' service is executed - the value is written to `*/Oper[CO]`.
- `Value 2` – select before operate with normal security:
  The 'Select' and 'Operate' services are executed - read of `*/SBO[CO]`, write to `*/Oper[CO]`.
- `Value 3` – direct control with enhanced security:
  An 'Operate' service is executed - the value is written to `*/Oper[CO]`.
- `Value 4` – select before operate with enhanced security:
  The 'Select' and 'Operate' services are executed - write to `*/SBOw[CO]`, write to `*/Oper[CO]`.

> **Info**
>
> *The driver always reads the `ctlModel` data attribute from the server again in order to execute the command correctly. However the driver does not transfer the **ctlModel** to the Runtime automatically. This includes situations where a variable was created for the **ctlModel**. If a variable was created, it is updated in accordance with the **Polling Rate** that is defined in the driver configuration.*

COMMANDS FOR ANALOG VALUES

If it is a command to an analog value (`*/Oper.ctlVal.i[CO]` or `*/Oper.ctlVal.f[CO]`), the driver checks to see whether the respective counterpart (`f` or `i`) exists for the set value on the object model of the server. If both values are present in the object model, the other respective value is calculated automatically using the previously-read configuration of the data object with the formula `f = ((i * scaleFactor) + offset) / 10^unitsMultiplier` (Note: `10` high `unitsMultiplier`) automatically calculated. Then both values (`i` and `f`) are sent as a structure in a command. No configuration steps are necessary for this.

COMMANDS - OVERVIEW

- Select and Cancel (on page 74)
- Additional Cause Diagnosis (on page 75)
- Service parameters of the command (on page 76)

7.2.1 Select and cancel

During a command, the driver only supports a Cancel with the zenon **Command**.
A Select can thus be separated from an Operate in command input. After the Select, instead of an Operate it is possible for a Cancel to follow.

CANCEL OF SELECT

If the command Processing is canceled from the second stage, the driver sends a Cancel.

PROCEDURE FOR CANCEL

The driver always sends sends a Select automatically if this is required according to the Control Model.

If the command sends a two-stage command and the **Select Before Operate** property of the action variable is active, the driver then - in accordance with the `ctlModel` of the data object - sends a Select during the first stage of the command action. This only applies for `ctlModel=2` or `4`. The values of the `ctlModel=0`, `1` or `3` are treated as a negative response from the Select .

Note: **Select Before Operate** means the property for the `Oper.ctlVal[CO]` variable in zenon and must not be confused with the data attribute `SBO`.

If the IEC 61850 server responds to the Select positively, the driver saves the information using the selected data object. In the following command (from the second stage of the command) the driver only executes the Operate .

The **Select Before Operate** property of the `ctlVal` variables has no influence on the execution of commands if the value of the `ctlVal` has been set directly and not by an action of the Command Processing.

CANCEL OF OPERATE

The driver supports Cancel operate only for the **Command Processing** module in zenon.
If the **Cancel operate** property is activated for the command variable, an Operate that has not yet been canceled can be canceled with Control Model with enhanced safety (3 or 4). If an ongoing Operate is canceled in the command input, the driver sends a Cancel to the server.

The **Cancel operate** property of the `ctlVal` variables has no influence on the execution of commands if the value of the `ctlVal` has been set directly and not by an action of the Command Processing.

Note: not all IEC 61850 servers support a Cancel after an Operate has been started.

7.2.2 Additional Cause Diagnosis

In accordance with the IEC61850 standard, an IEC61850 server can also send an additional service with `AdditionalCauseDiagnosis` to the driver (`AddCause` in `LastApplError` structure) for negative responses to a command. The possible values of the `AddCause` service parameter are listed in the IEC 61850-7-2. The driver can forward the `AddCause` received to zenon.

The driver also informs Runtime of the current status of the ongoing command (for example Select/Operate/Cancel) and provides information on whether the command was successful.

'COMMAND INFO' VARIABLES

Whilst already running, the driver provides the additional information in variables of the `Command Info` driver object type (see driver objects (on page 45)) as soon as these have been created. Variables that relay the stage of the current command can be created for each `*/Oper` structure.

- `*/Oper.AddCause`:
  After a command has failed, the value shows the error information - AddCause. This information can then only be used if this has also been sent by the server. Not every server supports this service. In this case, the value of the variable remains `0`.
- `*/Oper.ControlRun`:
  The status or the result of the current command:
  - `1` = command is running
  - `0` = command successful
  - `-1` = command failed
- `*/Oper.ControlState`:
  The state of the command that is currently running (processed sequentially):
  - `-1` = command ended
  - `0` = read ctlModel
  - `1` = 'Select' sent
  - `2` = 'Operate' sent
  - `3` = 'Cancel of Selectsent
  - 4 = 'Operate response' received (only in ctlModel=3 or 4)
  - 5 = 'Cancel of Operate' received (only in ctlModel=3 or 4)

Example:

You can create the following status variables for the variable 'PLC-Marker'-BOOL 'GE650!GEDeviceF650/XCBR1/Pos/Oper.ctlVal[CO]'

Command Info - SINT: S1!GEDeviceF650/XCBR1/Pos/Oper.AddCause

Command Info - SINT: S1!GEDeviceF650/XCBR1/Pos/Oper.ControlRun

Command Info - SINT: S1!GEDeviceF650/XCBR1/Pos/Oper.ControlState

Case 1:

If you set the value to `*/Oper.ctlVal` directly or the variable does not have the `Select Before Operate` property, the sequence of states is as follows:

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

Case 2:

If you use the two-stage command processing for `*/Oper.ctlVal` with the `Select Before Operate` property, the sequence of states for each command serve (for Select, Operate and Cancel) is the following:

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

For this, the following applies: x=1 or 2 or 3, depending on the command service.

> **Attention**
>
> *The **Command Info** variables do not have any value if no command was given. To evaluate these variables, you must first set a value to the `*/Oper.ctlVal[CO|SP]` variable.*

7.2.3 Service parameters of the command

The command services (for example Operate) that the driver sends to the 850 server consist of defined structures with additional parameters. The structure is defined in the IEC 61850 standard.

These parameters are handled as follows.

CHECK PARAMETERS

The driver sends all commands with `Check` = 0 by default. There are two methods for setting the `Check` value for the following Operate or SelectWithValue service:

1. The `Check` parameter can be pre-defined by setting a value for the `*/Oper.Check[CO]` variable. The current value of the variable is used for the command service parameter.
   Attention: The `Check` data attribute is a PACKED LIST (MMS bit string) with 2 bits. In accordance with IEC61850-8-1 8.1.3.5, the first member of the PACKED LIST is mapped to bit(0) and this bit should be the highest value (most significant) bit of the octet. This means that the valid set values for the `Check` data attribute 0x80 (synchrocheck), 0x40 (interlock-check), are 0xC0 (both check bits) and 0x00.
2. In zenon by setting in the command processing: Action - property **Qualifier of Command**. The valid set values are 2 - synchrocheck (inverted 01b), 1 - interlock-check (inverted 10b), 3 (both bits) and 0.

TEST PARAMETER

The driver sends all commands with `Test` = 0 by default. The `Test` flag for the following Operate, SelectWithValue and Cancel services can be predefined by setting a value for the `*/Oper.Test[CO]` variable. The current value of the variable is used for the command service parameter.

Attention

If you set the new values to the *[CO] variables in Runtime, these are forwarded to the driver. The values are then set to the next Operate or SelectWithValue in the driver.

The new values are initially written to the server in the subsequent command.

A loss of the connection sets the values, including the values of all CO variables, to the status `INVALID`. The reestablishment of the connection then resets the values to 0.

ORIGINATOR

The `Originator` data attributes `orCat` (Category) and `orIdent` (Identification) show which Originator (for example a client that is at bay level) has caused the last change of data (issued a command for example).

The `Origin` attributes are an obligatory part of the Oper, SBOw and Cancel services and optional for the upper node of a data object, e.g. 'Pos'. In order to analyze the `Originator` of the value changes `*/Pos/stVal[ST]` in the Runtime, you can create project variables with corresponding names in zenon: `*/Pos/origin.orCat[ST]` and `*/Pos/origin.orIdent[ST]`. If the 850 server does not support `Service Tracking`, then it is evident in these attributes which client was the last to give a

successful command. The attributes with Functional Constraint `CO` are not suitable for this because, according to the standard, it is not permitted to read them from an 850 server.

The driver compiles the Originator when sending a command, consisting of:

- **orCat**
  Dialog **Basic Settings**
- **orIdent**
  Dialog **Client configuration**

The orIdent can be used, based on the computer name of the computer on which the user has triggered the command. To do this, activate the **Use Client orIdent** property in the **Server** dialog. Then also create, in the **Client configuration** driver dialog, a **Hostname** for each computer in the zenon network.

The default origin.orIdent `'zenon: <computer_name>'` is used in commands if:

- In the client configuration for the corresponding computer (= **Hostname**), no orIdent is configured.
- in the client configuration for the corresponding computer, no entry with **Hostname** has been configured.

If the **Use Client orIdent** property is not active, the `orIdent` of the Primary Server is used.

Example

When a command is executed, the driver sets the originator for the `Originator` data attributes SBOw, Oper and Cancel as follows:

Example 1 - default settings:

- **orCat** = 2 (station control)
- **orIdent** = empty

Result: `2,'zenon: <computer_name>'` (string between quotation marks - CHAR(27)) , e.g. 'zenon: MyPC'

Example 2 - "Use Client orIdent" checkbox deactivated:

- **orCat** = 2 (station control)
- **orIdent** = "SCADA_850_Client"
  Configured in the **Hostname** for **Server 1** and **Server 2** in the **Client configuration** dialog

Result: `2,'SCADA_850_Client'`

Example 3 - "Use Client orIdent" checkbox activated:

- **orCat** = 2 (station control)
- **orIdent** = "PC_Room01"
  Configured in the **Hostname** for all computers in the zenon network that are physically in the same room.

  Result: `2,'PC_Room01'`

Note: In order to analyze the Originator of a data object, the IEC61850 server must be programmed accordingly. The server must accept the `Origin` of the received command from `*/Oper.origin.*[CO]` in these attributes.

If a Reports is received, the driver forwards the `Originator` to Runtime with priority before it forwards the other data attributes of the received data object (if only one `Origin` with Functional Constraint ST exists in the reported data object). The originator is always forwarded first.

Note: You can use `orCat` or `orIdent` variables to compile a `dynamic limit value text` for value changes to the `stVal` variables.

CTLNUM (CONTROL SEQUENCE NUMBER)

The driver automatically creates the values of the `ctlNum` data attribute that are given to the 850 server during the command. Each time a value is set to the `*/Oper.ctlVal[CO]` variable, the driver

automatically increases the value of the `*/Oper.ctlNum[CO]`. The driver restarts at `0` if the value `127` has been reached. A zenon variable `ctlNum[CO]`, even if configured, is not however updated in Runtime. The `ctlNum` value is only for correct execution of commands, not for the information of users. The driver also counts **write set value** to `ctlVal` if `ctlModel=0` (no action).

7.3 Service Tracking

Service tracking was introduced with edition 2 of the IEC61850 standard. As a basic requirement, the IEC61850 server must contain at least one logical node LTRKin the data model.

In order to use this functionality, corresponding additional service tracking variables must be created.

You can find detailed information on creating these variables in the Online import of an IEC 61850 server (on page 50) and Offline import (on page 54) chapters.

Attention

The additional service tracking variables - "Service tracking" driver object type - only get the values in Runtime if at least one *[SR] "PLC marker" driver object type variable of Logical Node LTRK in this Logical Device has been created in the project.

A PLC marker of LTRK need only exist and does not need to be requested nor shown in a screen.

If a PLC marker is created, the driver automatically requests all relevant data attributes of the following data objects with Service tracking from the controller: SpcTrk, DpcTrk, IncTrk, EncTrk, ApcFTrk, ApcIntTrk, BscTrk, IscTrk, BacTrk. These data objects all belong to the CTS common data class.

In order to be able to receive further data objects with service tracking information - that belongs to a different common data class (different than CTS):

- Create the LTRK*[SR] PLC marker driver object variables.
  By means of online or offline import.
- Create additional "service tracking" driver object type service tracking variables.
  This is to be carried out manually.

EXAMPLE:

To receive service tracking by means of value changes of the RptEna[BR] attribute in report control blocks ‘brcbA01‘ and ‘brcbB02‘, create the following variables:

- A PLC marker variable *LTRK*BrcbTrk/rptEna[SR]
- The "service tracking" variables: *brcbA01_rptEna[SR] and *brcbB02_rptEna[SR]

A service tracking object with FC = SR can be received from polling or by means of a report. The driver also supports data models with more than one instance of the LTRK node per logical device.

Note: If a screen contains service tracking variables that have not previously been registered, the variables remain empty until service tracking information is received by the Server.

ERRORCODE[SR] VARIABLES - ACTION FOR STATUS BIT M1

If additional service tracking variables are created for a data object, there is no pre-defined sequence in which the values are transferred from the driver to Runtime.

The consistency of data, for example for the creation of dynamic limit texts for the CEL, is however ensured by the following process:

- Each time service tracking is received, the driver sets the status bit **M1** with the service tracking variable `*_errorCode[SR]`.
- After all service tracking values have been transferred to Runtime, the driver resets the status bit M1 of the `*_errorCode[SR]` variable (`M1 = 0`).
- This resetting of the M1 status bit is a suitable trigger for the creation of a CEL/AML entry.

> **Attention**
>
> If you evaluate the status bit M1 with a reaction matrix and the service tracking values are received from polling and not from a report, the initial value also triggers a CEL/AML entry.

ALLCTS[SR] VARIABLE

A `*_allCTS[SR]` variable contains the received service tracking information, consisting of the data attributes of the Common Data Class CTS.

- This variable is a string variable.
- The value of the variable is a text, consisting of data attributes of the Common Data Class CTS in the following sequence: `objRef, serviceType, errorCode, originatorID, t, ctlVal, origin.orCat, origin.orIdent, ctlNum, T, Test, Check, respAddCause`.
- The individual data attributes are each separated by a `;`.
- The status bits of the variable result from the sum of all status bits of the data attributes.
- The time stamp results from the time stamp of the last attribute of the service tracking object. If there is no time stamp, the current time is used.

If the service tracking object received does not contain a Control Service (CTS), but instead Common (CST) or Control Blocks (UTS, BTS etc.), then the *_allCTS variable only consists of data attributes of the Common Data Classe CST: `objRef, serviceType, errorCode, originatorID, t`.

LOG ENTRIES

LOG entry	Debug Level	Description
Variable %s has wrong format for a service tracking variable.	Error	The variable addressing used does not correspond to the expected format. „`Server!Device/Node/Object_Attribute`" is expected
Variable %s cannot be advised since no Service Tracking Variable from Driver Object Typ SPS-Merker was found.	Error	The driver cannot request the variable from the PLC because no "PLC marker" [SR] variables were created in the Editor. Please create at least one corresponding variable.

7.4 Mapping of double point values

Double Point Value Mapping is a standard function of the zenon Energy driver. It only influences zenon Runtime and has no effect on the driver communication with a device. Configuration is carried out in the driver settings in the **Basic Settings** tab.

Note: It is recommended that you leave the **Deactivate standard double point value mapping** option in the driver configuration as the default, inactive.

The driver uses Double Point Value Mapping to convert values so that they are displayed in a user-friendly manner. However this only applies to the HMI.

The driver always communicates with one device with values for Double Points with 2-bit information. This corresponds to the definitions of the energy standard. That means:

Parameters	Double Point	Value	Meaning
Intermediate	`00b`	`0`	Switches are neither open nor closed, for example the End-Position has not yet been reached
Off	`01b`	`1`	Switch open
On	`10b`	`2`	Close switch/switch closed
Fault	`11b`	`3`	Error

Double Points are coded with 2-bits in the energy sector for historical reasons: The transmission of a telegram to a serial connection (RS232) with a series of values that only contain `0` was not safeguarded against transmission errors. In order to increase the certainty, it was decided in the first standards that the value for `OFF` is not to be sent as `0` but as `01b`, which corresponds to decimal `1`. These Double Point Values also precisely reflect the type of how two sensors record the physical position of a switch.

However, the values sent this way may be confusing for people:

- OFF = 1
- ON = 2

Humans are used to all other devices and systems:

- OFF = 0
- ON = 1

At the same time, Single Point Values are also defined with OFF = 0 in the same standard.

The user must always be aware of the technical level on which they are acting and receiving or sending information. In stressful situations, this can very easily lead to serious mistakes. For example, if ON is sent instead of OFF.

In order to avoid this dangerous error, the zenon Energy driver offers its own Double Point Value Mapping.

MAPPING VOR HMI

With the Double Point Value Mapping, all Double Points in zenon have the following values:

- Intermediate = 2
- Off = 0
- On = 1
- Fault = 3

> **Information**
>
> This function can be deactivated in the driver settings. However some features such as Command Processing or ALC can no longer be used then.
>
> Recommendation: Do not use numerical elements and numerical values to display OFF/ON or OPEN/CLOSE. Use combined elements with graphic symbols or text elements instead.

DPI MAPPING IN IEC61850 CLIENT DRIVER

In accordance with the IEC61850-7-3 standard, in Common Data Classes, Double Point Status (DPS) and Controllable Double Point (DPC), the stVal attribute has the data type CODED ENUM with the value range: intermediate-state (00) | off (01) | on (10) | bad-state (11). The driver therefore waits until a Double Point attribute of the data type CODED ENUM is mapped to the communication protocol as an MMS of the bit string data type. The stVal data attribute can thus have the following values: 0x00, 0x40, 0x80 and 0xC0. This basic data type is permitted in the SCL language with the name

`Dbpos` (IEC61850-6). In zenon, this corresponds to a variable with data type UDINT (default) or STRING (if changed manually) - see assignment of datatypes (on page 47).

In the driver configuration, the **Deactivate standard double point value mapping** option should remain inactive by default in order to use the modules from the zenon Energy Edition, for example the functions of **Command Processing** and **ALC**.

With the default settings, the driver assigns values of the `stVal` data attributes of DPS and DPC in accordance with the following table:

Position value in the end device	zenon Dbpos value - STRING	zenon Dbpos value - UDINT unmapped	zenon Wert - value mapped
Intermediate (00b)	'00'	0	2
Off (01b)	'01'	64 (0x40)	0
On (10b)	'10'	128 (0x80)	1
Fault (11b)	'11'	192 (0xC0)	3

The driver only converts values for variables with the names `*/stVal` and numerical data type that corresponds to bType `'Dbpos'` in the SCL language (IEC61850-6). This means that the values of the `*/stVal` variable with a data type set to STRING manually are excluded from mapping.

This means in Runtime (for example):

Position of the switch	UDINT mapped	UDINT unmapped	STRING
off	0	64	'01'
on	1	128	'10'

7.5 Quality, time stamp and status bits of the variable

Variables in zenon with **FC (on page 106)=ST** or **FC=MX** (for example `*/Pos/stVal[ST]`, `*/PhV.phsA/cVal.mag.f[MX]`) get a time stamp and status bits from the accompanying data attributes 'q' and 't'. This happens in the driver automatically. Additional configuration in zenon is not necessary.

The driver always assigns selected Quality Bits (the '**q**' attribute) and TimeQuality Bits (the '**t**' attribute) of the corresponding status bits to each zenon variable that has been created for an attribute of the data object and has an **FC (on page 106)=ST** or **FC=MX**, for example for status bits of the `stVal` variable. The driver also assigns the value of the '`t`' attribute to the time stamp of all **ST/MX** variables of the data object.

The driver can also support received data attributes '`q`' and '**t**' in corresponding project variables if these have been created.

Example: Receiving the time stamp as a '**t**' attribute is used for the corresponding `stVal` variable in the **CEL**.

The driver automatically uses the time stamp and the quality of the data object for all its variables that were created in the project. For this reason, no variables need to be created separately for the attributes '`q`' and '`t`', except if you want to analyze quality bits that have not been assigned any status bits.

Info

The driver takes the TimeQuality of the 't' attribute into account. In addition to 3 status bits, TimeQuality also contains 5 bits for the TimeAccuracy information, the precision value (N bits of accuracy - performance class). The theoretically-possible precision values are between 0 - 24 and value 31 is defined as "unspecified" (all bits). However only some of the precisions can use a valid 850 server. The lowest precision permitted for an 850 product is class T0 - 7 bits - 10ms.

If TimeAccuracy is not 31 "unspecified", the FractionOfSeconds field of the time value - the binary decimal places - correspond to the precision - N+1 - shortened. The time stamp is then rounded to the milliseconds.

Example:

- TimeAccuracy = 10 bits (class T1, 11 binary decimal places): the driver shortens the received time value to the precision 1/2048=0.00048828125 [s] and rounds up to the milliseconds.
- TimeAccuracy = 0 bits (no valid SBOw, Oper and Cancel in the 850-7-2 standard, 1 binary decimal place): the driver shortens the time value received down to the seconds.

The selected Quality and TimeQuality bits are assigned to the status bits of the variables in the same data object as follows:

IEC61850 Quality or TimeQuality bits	Status bit in zenon	Notes
Validity = **Invalid** or **Reserved** or **Questionable**	`INVALID`	Identified by a red square on the linked element.
Overflow	`OV_870 (Overflow)`	This is also an IEC 60870 status
OutOfRange	`OR_DRV (value outside the valid range)`	
Source = Substituted	`SB_870 (Substituted)`	This is also an IEC 60870 status
Test	`TEST (Test bit)`	This is also an IEC 60870 status
OperatorBlocked	`BL_870 (Blocked)`	This is also an IEC 60870 status
ClockFailure	`T_INVAL (invalid time)` and `T_INTERN (real time internal)`	The value of the '**t**' attribute is ignored - **stVal**has the corresponding time stamp of the local PC clock.
ClockNotSynchronized	`T_INVAL (invalid time)` and `T_EXTERN (real time external)`	

The TimeQuality bit **LeapSecondsKnown**is not assigned T_INVAL.

The variables of the data object without '**t**' attribute (or with '**t**' value of `0`) and the variables with other Functional Constrain (on page 106) as **ST**/**MX** receive the time stamp of the local PC time.

Info

*You can evaluate status bits (such as `T INVAL`) by using a combined element or `reaction matrix` (of type "Multi"). The status bits can be filtered in **CEL** or **AML**.*

CAUSE OF TRANSMISSION (COT)

For each variable, the driver provides information about whether the value of the variable was received from polling or by means of a report. The information is written to the status bits `COTx` .

COT value	Status bits	Description
1	COT0	The value was received by means of polling.
2	COT1	The value was received by means of an Integrity Report. Note: This only works if Reason-for-inclusion is activated in the optional fields (on page 28) dialog. Otherwise the COT=3
3	COT0, COT1	The value was received by means of a report Note: If Reason-for-inclusion is activated in the Optional fields (on page 28) dialog, it is only reports with the trigger option data-change, data-update or quality-change. If Reason-for-inclusion is not activated, it is all reports - including Integrity and GI.
20	COT2, COT4	The value was received by means of an General Interogation Report. Note: This only works if Reason-for-inclusion is activated in the optional fields (on page 28) dialog. Otherwise the COT=3

Info

The COT value for theIEC850 driver is orientated towards the IEC60870 standard:

1 - periodic, cyclic

2 - background scan

3 - spontaneous

20 - general interrogation

During a command execution using the **Command Processing** module, the driver updates the status bits of the */Oper.ctlVal[CO] variable as follows:

- Select, SelectWithValue: `SE_870` + `COT_act(6)`, then `SE_870` + `COT_actcon(7)` possible with `N_CONF`
- Operate: `COT_act(6)`, then `COT_actcon(7)` with possibly `N_CONF`, then possibly `COT_actterm(10)`
- Cancel: `SE_870` + `COT_deact(8)`, then `SE_870` + `COT_deactcon(9)`

The **watchdog timer** of the **Command Processing** evaluates these status bits of the **Action variable**.

7.6 Filetransfer

There are three functions implemented for the file transfer:

- Request folder information (on page 88)

- Get file from server (on page 88)
- Delete file (on page 89)

TROUBLESHOOTING

If errors occur when carrying out the file transfer, such as the server responding negatively, the command variable changes its value to "`XXX ERROR`" (`XXX = DIR`, `GET` or `DEL`) and the driver optionally writes an entry into the LOG file (see also error analysis (on page 104)).

7.6.1 Request folder information

To request folder information:

1. Create, in your zenon, two string variables of the **driver object type** `File Transfer` with the corresponding network address:
   a) The first variable, hereafter called the command variable, with the name **"SERVER!Command"** can also be used to get files and to delete them.
   b) The second variable with the name "**SERVER!Directory**" is only used for the result of the folder query. It receives the folder content as legible text. For this reason, its size should correspond to the maximum size of the file name (including the folder) in the PLC.
2. For the command variable, set the value "`DIR`" (for the root directory) or "`DIR <file_spec>`".
   <file_spec> is a string that is sent to the server as a transfer parameter.

If the folder has been successfully received:

- The command variable changes its value to "`DIR OK`"
- The folder variable contains the received folder content as legible text. One line of this text has the following format:

  <File name>;<File length>;<Time stamp>.

7.6.2 Get file from server

To get a file from the server, set the value "`GET <file name>`" for the command variable.

If the file has been successfully received:

- It is saved in the folder that was defined as **Directory for file transfer** in the driver dialog **Basic settings** (on page 14).
  Files from a subfolder are also stored in the main folder.
- The command variable changes its value to "`GET OK`"

7.6.3 Delete file

To get a file from the server, set the value "`DEL <file name>`" for the command variable.

If the file was deleted successfully, the command variable changes its value to "`DEL OK`"

8. Reporting

Reporting was introduced by the IEC 61850 standard for spontaneous communication. The variables that are linked with a successfully activated RCB enable Report are no longer polled, but a reported by the PLC (an IED) if there is a value change. The variables that are reported if an 850 client registers a certain Report Control Block (RCB) (enabled), is defined in the DatSet (Data Set) attribute of the RCB . The Data Set is defined in the SCL file of the IEDs and contains the data objects whose value changes are reported spontaneously by the IEC 61850 server.

The configuration of the reporting is carried out in driver configuration - server (on page 21). In **RCB assignement/dynamic datasets** (on page 33), Buffered and Unbuffered Reports can be configured. RCBs configured this way have a higher priority than any automatically-searched Unbuffered Reports. If the activation of an RCB is unsuccessful, the driver can instead automatically activate a **URCB**, which also contains datasets if it contains the required data. If you do not want the driver to activate the URCBs automatically, set the **Max. auto used URCBs** property in the driver configuration to 0.

If the driver cannot activate the reports for any reason, polling is used instead. In this case, in the **Connection State** variable (on page 70), the bit 17 or 25 (`MMS_RCB_ENABLE_FAILED`) is set.

You can find further information on the different possibilities for checking the activation of `RCB`s in the RCB activation - checking possibilities (on page 98) chapter.

GENERAL INTERROGATION

The driver also sends, if the RCBs is activated, a General Interrogation (RCB.GI=TRUE) command to get the most recent value of the variables. The first time the RCB is activated (i.e. when Runtime is started) and when the connection is reestablished (after a connection has been lost), it sends a GI. However the general query is not sent if it is a repeated attempt to enable a report. This happens after expiry of the time configured in the **RCBs enable retries** property. This avoids data duplicates, because the values have already been received from the polling.

DATA SETS

The IEC850 driver supports reports with the data sets predefined in the PLC (in the SCL file) or with Dynamic Data Sets - with the DatSet attribute, which was dynamically defined during activation of the RCB.

As envisaged by the IEC 61850 Standard, for a Report data object with Functional Constraints (FC) the following is available:

- Status - ST
- Measurands - MX
- Service tracking - SR
- Configuration - CF
- and further: SP, SV

If a non-standards-compliant reference has been defined in a DataSet (other than the FCs listed here), the driver nevertheless activates the RCB. However the data with invalid FCs is ignored in the reports received. The data objects and data attributes concerned continue to be polled.

In a data set:

- Data objects with different FCs can be used together if the IEC 61850 server supports this.
- Data objects from different Logical Devices can be contained.

To make activation of reporting possible, the following conditions should be met:

- The data set and the Reporting Control Block that uses the data set must be in the same Logical Device.
- At least one variable of the Logical Node, in which the RCB is included, must have been created in the project. Otherwise the driver does not take this Logical Node into account at all. Exception: LLN0. The RCBs defined in the logical node LLN0 are automatically taken into account for each logical device if the LD has at least one variable that was created in the project variable list.

DATA SETS WITH FCD OR FCDA

The dynamic data sets configured in the driver always consist of FCD references:

- Functional Constraints (FC)

- Data Object (DO)

Datasets already defined in an IED usually consist of FCD references, but they can also consist of FCDA references.

An FCDA reference consists of:

- FCD
- Data Attribute (DA)

Note: To preclude data inconsistencies, it is recommended that data sets are always used with FCD references. To do this, the SCL file of the server must be configured accordingly (no "daName" elements in the DataSet).

Note also that it is only with FCD references where the value changes are contained in the report with the current time stamp of the server in the report. With FCDA references, only the reports with GI or Integrity contain the time stamp of the server.

If the server uses FCDA references that cannot be reconfigured to FCD references (for example because the server does not accept any updates to the SCL file) the driver will however activate a report with such data sets. Exception: Reports with Service Tracking are only supported with FCD!

REPORTS WITH DATA SET WITH FCDA

For reports with a data set with FCDA , the driver works according to the following rules:

- Whilst activating the RCB, the structure of the DataSet is recognized. If there are individual data attributes present here, a check is made to see whether, in this DataSet, an attendant Quality has also been defined at the same time. If no FCDA reference for the corresponding Quality attribute is found, that is however in the data model, the driver writes a LOG entry:

LOG entry	Level	Description
Enabling scan of %s on device %s	MSG	The element given (q attribute) is queried in the cycle of the **Data consistency scan**.

  From this point in time, the q attribute missing in the report - depending on the driver configuration in the **Data consistency scan** setting - is polled automatically. This ensures that for each report, measured values received also have the quality information and thus correspond to the IEC61850 standard.

- If the t attribute is contained in a received report, this is also used. If the t attribute is missing in a report, the values of the variables contain an internal time stamp.
- If the q attribute is contained in the received report, the status bits of the variables are updated accordingly. If this q attribute is missing, the status bits of the variables reflect the q attribute as already recognized in the driver. This can be from an earlier report, polling of the registered data object or by a data scan (as configured in **Data consistency scan**).
- If a report that only contains a t attribute is received (without measured value, without q attribute), a LOG entry is created because this is not permitted in the standard. This can only

occur if the PLC has been configured or structured incorrectly. With an invalid SCL file, for example. The time stamp received is ignored.
Exception: If a t variable is created in the project and then subsequently registered like the report receipt, the value of the variable reflects the value of the t attribute received.

LOG entry	Level	Description
Received report contains only yyy$t; missing value and quality. Report will be ignored. Probable reason: mistake in IED configuration (SCL)	Warning	A report that has neither measured values nor a q attribute has been received. This indicates that the server has been configured with an invalid SCL file. Example: In SCL.DataTypeTemplates, the t data attribute was configured with its own Trigger Options.

TRIGGER OPTIONS

The driver supports all `Trigger Conditions`, that are used to trigger the sending of the report. The Trigger Options (TrgOps) define which value change of a data instance is reported.

When activating the RCB - if the **Use preconfigured (SCL) options** checkbox is not activated - the driver sends the configured Trigger Options to the server. Write errors are ignored and the RCB continues to be used.

Note: Ensure that `TrgOp: general-interrogation` is activated. Otherwise the driver cannot receive values for the variables present in the report when first started up.

The `TrigerConditions` data attribute type that was assigned to the `MMS Bitstring(6)` in accordance with the standard if the corresponding *TrgOps[RP|BR] variable was created during import, corresponds to the UDINT data type in zenon by default (Motorola format, bit 0 highest value in its octet). In addition, the driver allows such variables with the data type STRING instead of UDINT. The value of the STRING has the length 6 and only contains the characters "0" and "1"; in the same sequence as the bits are defined in the standard.

Example

Value as string "011111" (as UDINT=124=01111100b) means that all TriggerCondition bits (except bit 0) - are set as `'Reserved'`.

The value "000010" (as UDINT=8=00001000b) means that only the bit `4 – 'intergity'` is set.

8.1 Unbuffered Reporting

The IEC850 driver supports two variants of Unbuffered Reports:

1. Automatic assignment:
   propert **max. auto used URCBs**:
   All existing Unbuffered Report Control Blocks (URCB) are checked in terms of content (data sets). The URCBs that contain most variables of a project are registered first;
2. Static assignment - **statically assigned RCB** (on page 33) dialog:
   The driver uses the assignment of Report Control Blocks to the computer - **Hostname** - as it is defined in the driver configuration - client configuration (on page 29).

AUTOMATIC ASSIGNMENT

Automatic assignment works with pre-defined data sets and pre-configured assignments of data sets to Report Control Blocks in the PLC.

Whilst the driver starts, all data sets that are present in the server are read off and ordered according to their usability (number of project variables included). This sorting indicates the sequence in which the driver attempts to activate free URCBs for reporting. At most, only as many URSBs per Logical Device are activated as are defined in Driver configuration (on page 21) in the **max. auto used URCBs** property. All data objects of the data set that are successfully linked to an activated URCB are no longer polled but are reported in the event of a value change.

RULES FOR CONFLICTS THAT OCCUR

When activating reporting, the driver also recognizes URCBs with the same content; the name of these reports can be different. Before the URCBs are sorted for the zenon project according to their usability, the driver first sorts the data objects of the data sets of each URCB and compares them in order to avoid duplicates. The following applies for conflicts when activating the Reporting:

- For the same reports - with identical or different sequences of the same data objects: only one URCB is activated.
- If an URCB is already occupied by another client, the next free URCB is activated.
- If all URCBs are already occupied by other clients, the data points are polled.

EXAMPLE OF AUTOMATIC ALLOCATION

The following variables are available in your project

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

The `*/urcbMXA01/*` variables are for information only. The variables for the RCB attributes need not necessarily exist in the zenon project in order for reporting to work. The PLC also contains a data set `GEDeviceF650/LLN0$MMXU1$MX` with the following content.

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	{ { { { f0.00
GEDeviceF650/MMXU1MXPhV	{ { { { f0.00
GEDeviceF650/MMXU1MXA	{ { { { f0.00
GEDeviceF650/MMXU1MXAuxV	{ { { f0.0000
GEDeviceF650/MMXU1MXBusHz	{ { f0.00000

This data set is assigned to RCB `LLN0$RP$urcbMXA01` in the PLC. To do this, enter the name of the data set in the DatSet attribute of this RCB.

```
LN LLN0
  FC ST
  FC CF
  FC DC
  FC RP
    DO urcbSTA01   { "UnbufInps1", F, F, "GEDeviceF650/LLN0$eveGGIO1$ST", +0, [0111110010], +0, +0, [010010], +0, F }
    DO urcbSTB01   { "UnbufContInputs1", F, F, "GEDeviceF650/LLN0$GGIO_F_Contacts", +0, [0111110010], +0, +0, [010010], +0, F }
    DO urcbSTC01   { "UnbufContInputs2", F, F, "GEDeviceF650/LLN0$GGIO_G_Contacts", +0, [0111110010], +0, +0, [010010], +0, F }
    DO urcbSTD01   { "UnbufContInputs3", F, F, "GEDeviceF650/LLN0$GGIO_H_Contacts", +0, [0111110010], +0, +0, [010010], +0, F }
    DO urcbSTE01   { "UnbufContInputs4", F, F, "GEDeviceF650/LLN0$GGIO_J_Contacts", +0, [0111110010], +0, +0, [010010], +0, F }
    DO ubrbSTF01   { "UnbufPulseCounters1", F, F, "GEDeviceF650/LLN0$MMTR1$ST", +0, [0111110010], +0, +0, [010010], +0, F }
    DO urcbMXA01   { "UnbufMeas1", F, F, "GEDeviceF650/LLN0$MMXU1$MX", +0, [0111110010], +0, +0, [010010], +0, F }
    DO urcbMXB01   { "UnbufMeas2", F, F, "GEDeviceF650/LLN0$MSQI1$MX", +0, [0111110010], +0, +0, [010010], +0, F }
  FC BR
  FC GO
```

The URCBs `LLN0$RP$urcbMXA01` and `LLN0$RP$urcbMXB01` are activated when the driver starts (if they are still free).
Because there are no further suitable URCBs that have project variables in their datasets, the remaining URCBs are not used. If the project also contains a variable that is included in the data set `GEDeviceF650/LLN0$eveGGIO1$ST`, this would mean the `LLN0$RP$urcbSTA01` is also activated.

STATIC ASSIGNMENT

For static assignment, the Report Control Blocks that are used are defined and fixed in the driver configuration - **statically assigned RCB** (on page 33). The driver attempts to activate all listed RCBs, regardless of their content.
If a Dynamic Data Set was also entered in the driver configuration, the driver in the PLC of this data set allocates this set to the RCB, in that it changes the value of the DatSet attribute before the report is activated.
If no Dynamic Data Set was entered, the data set pre-defined in the PLC is valid (as with automatic allocation).

If an RCB was activated successfully, all data objects that are contained in its data set are no longer polled. In this case, the driver expects that the IED reports the value changes spontaneously.

8.2 Buffered Reporting

For Buffered Reports, an IEC61850 server saves the changes in the time in which a connected 850 client cannot be reached. If the client (the IEC850 driver) is online again (if the connection is resetablished, for example), the driver reads all changes and processes them; it creates omitted alarms, for example.

There is no automatic allocation for Buffered Reporting. The allocation of BRCBs to the driver is always static:

- Either using the driver configuration as described for static allocation with Unbuffered Reporting (on page 93);
- Or for reasons of compatibility with older drivers, via the the `Report ID` of the BRCB that was configured in the SCL file and contains certain text. To do this, the **Use Report-ID for RCB assignment** property should be activated in the driver configuration.
  If the assignments are to be configured in the SCL file, the assigned report ID (RptID) of the BRCB must contain a string with the following nomenclature 'zenon_Computername_*'; for example: zenon_MyPC_Measurands1. The driver then uses the data set of RCB that is defined in the SCL file.

A Buffered Report can only be activated by a host (a client). To prevent the driver from activating all Buffered Reports and blocking access for other IEC61850 clients, these must be explicitly defined in the driver configuration (on page 9).

Irrespective of this, the basic behavior is the same as with Unbuffered Reporting.

> **Information**
>
> *Once Runtime is started, the driver first reads the RCB attributes and only writes to enable if the RCB:*
>
> - Is free
> - Is not yet enabled (for example by another 850 client)
> - Is not reserved
>
> *In the event of repetitions - from the **RCBs enable retries** properties - the driver no longer reads the RCB. In this case, it always writes RCB.RptEna=TRUE and then checks whether the PLC confirms success or returns an error.*

GET OR DELETE BRCB BUFFER

If the RCB of the previously-used BRCB is activated again (**RCB enable**) (i.e. another establishment of the connection after a loss of the connection), the driver sends a saved **EntryID** to the 850 server to get the buffered value changes from the IED . This procedure is defined as such in the IEC61850 standard. The **Do not purge BRCB buffer at start** option has no influence on this.

The first time an attempt to enable a report is made (i.e. when Runtime is started), the driver can also send the value of the **EntryID**. However, the driver only does this if the **Do not purge BRCB buffer at start** option is activated and if the driver has saved an **EntryID** beforehand.

Hint: The driver saves the last-received **EntryID** for each `ObjectReference` to BRCB first internally and then - each minute or if Runtime is closed - to a TXT file in the directory of the Runtime data.

In the following situations however, the driver sends the **purge buffer** command to delete the buffer on the IED :

- It is the first connection after Runtime has been started and the **Do not purge BRCB buffer at start** option in the basic settings (on page 14) is deactivated.
- The driver has not received a valid **EntryID** value beforehand.
- It is a repeated attempt to enable a report - after the **RCBs enable retries** time has expired. The BRCB.PurgeBuf command is always sent with a repeated attempt. This avoids data duplicates, because the values have already been received from the polling. The **Do not purge BRCB buffer at start** option has no effect on repetitions.

Attention

An 850 server is obliged to first confirm the writing of **EntryID** and only then to send the buffered values. The reports are ignored as long as the BRCB activation has not yet been confirmed.

BUFFERED REPORTS AND ZENON REDUNDANCY

Because zenon supports redundancy (see zenon network) and a BRCB can only be activated once by a PC, you must define separately, for the project **Server 1** and the **Server 2** (Standby Server), which PC is to activate which report. The driver is started on both the Primary Server and the Standby and checks the driver configuration (**Hostname** property), to see if it finds a configuration that corresponds to the computer name that is running on Runtime.

For this reason, you need two copies of the Buffered Report in the PLC (850 server) in the event of a redundant configuration; one for the driver that runs on the project server and one for the driver that runs on the Standby Server. You also create two entries in the RCB assignment (on page 33) list in the driver configuration. One with the computer name of the project server and one with the computer name of the Standby Server.

Note: With the configuration entry for the computer name of **Server 1**, you can enter the first instance of the BRCB ("brcbA01" for example) and for the configuration entry of **Server 2** - the second instance ("brcbA02" for example).

Attention

Even if you do not use redundancy, you must nevertheless define which reports are used for the PC (**Hostname** property) in the driver configuration on which zenon Runtime runs.

BRCB.RESVTMS HANDLING

The driver only handles the data attribute `BRCB.ResvTms` if no **ClientLN.iedName** has been defined in the driver configuration. You configure this **ClientLN.iedName** in the client configuration (on page 29) driver dialog or with the **IEC850 Driver Configuration** wizard from an SCD file.

Otherwise it is assumed that the server with the same SCD file has already been preconfigured, in which the BRCBs are exclusively reserved for other clients.

If no **ClientLN.iedName** is defined in the driver configuration, when the connection to the server is established, the BRCBs that are to be enabled are checked to see if there is the optional attribute `ResvTms`. In the process, the value -1 means that this BRCB is exclusively reserved for another client in the SCD file. In this case there is an incorrect configuration and a warning is issued in the LOG.

Possible cause of the error:

- The driver was not configured with the correct `Client IED name`.

- In the driver, BRCBs were configured statically that were however reserved for another client according to the SCD file.

If no **ClientLN.iedName** was configured in the driver, the driver sets the `ResvTms` attribute to 32767 seconds in order to reserve this BRCB for the longest possible time in the event of a connection failure. This time is 2^15 - 1 seconds = 7fff hex, approximately 9 hours.

8.3 RCB activation - verification possibilities

To find out if a certain RCB has been activated, read the value of the `*RptEna[RP/BR]` data attribute.

If the value of this variable is `TRUE`, then the RCB is assigned by a client. This information is only meaningful and reliable if only one single client communicates with the server (IED).

Otherwise you do see that the report has been enabled, but there is no information about the client that activated this report. Additional information on the fact that one of the RCBs could not be activated can be read from the values of the **Connection State** variable (on page 70).

If the driver cannot activate the reports for any reason, polling is used instead. In this case, in the **Connection State** variable, the bit 17 or 25 (`MMS_RCB_ENABLE_FAILED`) is set.

DIAGNOSIS VIEWER

You can get more details if you read the LOG file with the Diagnosis Viewer. In the LOG file, the corresponding information on the enabling of the reports by the driver (IEC850 client) is logged at MSG level. The attendant LOG entry:

- "`Trying to enable xxx on device nnn`".
  Example for xxx: `LLN0$BR$brcbA01`.

Note that only the ERROR level is logged by default in the **Diagnosis Viewer**. You must configure the corresponding **Messagelevel** in the **Diagnosis Viewer**. This configuration in the **DiagViewer** is only possible if the corresponding driver tasks are currently active. This is then the case if Runtime is running or the driver configuration is open in the zenon Editor.

If the MSG level is activated in the Diagnosis Viewer, but the "`Trying to enable xxx on device nnn`" LOG entry is not there, then ensure that the "`Hostname`" property of the driver configuration has been configured correctly. This configuration must be configured with the name of the computer on which the driver is running (= zenon Runtime).
Note: You configure the **Hostname** configuration in the **Client configuration** (on page 29) driver dialog.

Ensure that there is at least one `*[ST|MX]` variable in your zenon project whose value is reported by this report.

Additional LOG entries

- Enabling the report is accepted by the server:
  - 'MSG' LOG entry with text: "`Successfully enabled xxx on device yyy`"
  - 'DEBUG' LOG entries with the list of data objects with text: "`Disabled ST polling for zzz on device nnn`".
    Example for zzz: `*/XCBR1$ST$Pos`.
    Note that these data objects are now expected via a report and are no longer polled.
- Enabling of the report is refused by the server - 'ERROR' LOG entry with the number of the MMS error code as received by the server.

> **Information**
>
> If the driver is configured for the use of certain RCBs and these RCBs have an incorrect dataset definition, the report is activated but some data objects still need to be polled. In the LOG file, you can find entries via enabling of RCB, but not the entries via deactivated polling.

CAUSE OF TRANSMISSION (COT)

For each variable, the driver provides information about whether the value of the variable was received from polling or by means of a report. The information is written to the status bits `COTx` .

COT value	Status bits	Description
1	COT0	The value was received by means of polling.
2	COT1	The value was received by means of an Integrity Report. Note: This only works if Reason-for-inclusion is activated in the optional fields (on page 28) dialog. Otherwise the COT=3
3	COT0, COT1	The value was received by means of a report Note: If Reason-for-inclusion is activated in the Optional fields (on page 28) dialog, it is only reports with the trigger option data-change, data-update or quality-change. If Reason-for-inclusion is not activated, it is all reports - including Integrity and GI.
20	COT2, COT4	The value was received by means of an General Interogation Report. Note: This only works if Reason-for-inclusion is activated in the optional fields (on page 28) dialog. Otherwise the COT=3

Info

The COT value for theIEC850 driver is orientated towards the IEC60870 standard:

1 - periodic, cyclic

2 - background scan

3 - spontaneous

20 - general interrogation

$$Trenner

Info

You can evaluate `COTx` status bits by using a combined element or reaction matrix ('multi' type). The status bits can be filtered in CEL or AML.

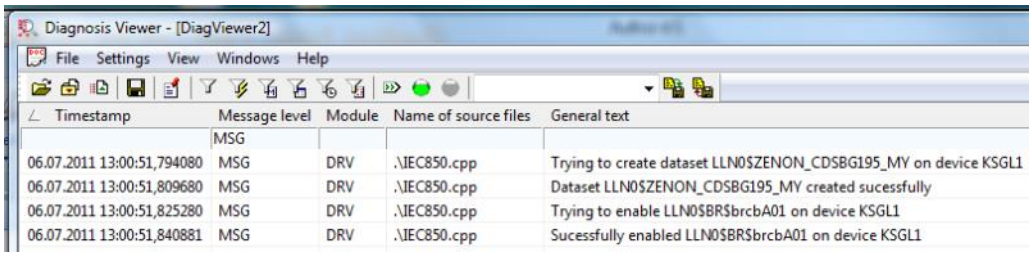
8.4 Dynamic Data Sets

To use Dynamic Data Sets, the two services `CreateDataSet()` and `DeleteDataSet()` must be supported by the IEC 61850 server and the DatSet attribute must be able to be written to in the Reporting Control Blocks (SCL: ReportSettings.datSet="Dyn"). ReportSettings.datSet="Dyn"). When configuring, you must take into account all possible server restrictions such as the maximum number of elements in the data set (SCL: DynDataSet.max and maxAttributes).

Dynamic Data Sets lists data objects and are configured in the dynamic DataSet configuration (on page 37) dialog. They can be used for Unbuffered Reporting and for Buffered Reporting, but only if static assignments with RCBs were configured in the driver configuration - client configuration (on page 29).

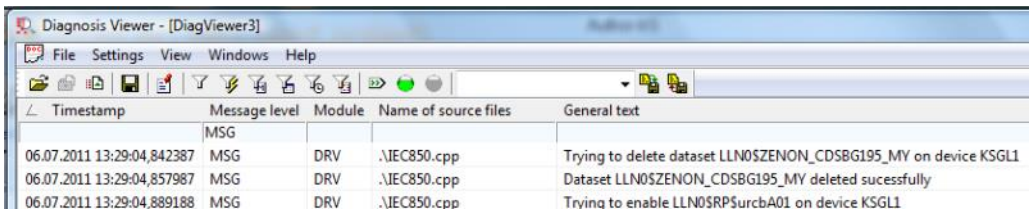
If, in the driver configuration, a Dynamic Data Set was entered for an RCB, the driver in the PLC assigns this data set to the RCB, by changing the value of the DatSet attribute of this block before the Report is activated.

The creation of a dynamic data set can be checked in the Diagnosis Viewer (on page 104):



Before the statically assigned Reporting Control Blocks is activated, all dynamic data sets that are linked to them via driver configuration - client configuration (on page 29) are compared to those on the IEC 61850 server. All Dynamic Data Sets available on the server that are not (or no longer) required are deleted. All Dynamic Data Sets not available on the server are created and all data sets that are available on both sides are checked to see that they have the same content. If the content is different, they are deleted and created again. However, the driver only deletes the data sets that it has created itself. If the driver cannot delete a data set, the asigned RCB is activated with the pre-existing data set.

The deletion of a dynamic data set can be checked in the Diagnosis Viewer (on page 104):



The dynamic data sets that were created in Runtime (using the `CreateDataSet()` service) receive a client-dependent name on the server, in order for them to be easily found again. The nomenclature of the Dynamic Data Set name is as follows:

`LLN0$ZENON_hostname_datasetname`. Here:

Hostname is the computer name of the client as it is defined in the driver configuration.
You can find this configuration in the client configuration (on page 29) under **Hostname**.
datasetname is the name that was defined in the driver configuration.
You can find this configuration in the client configuration (on page 29) in the dynamic dataset configuration (on page 33).

If no Dynamic Data Set was entered in the RCB configuration of the driver, the pre-defined data set in the PLC is valid.

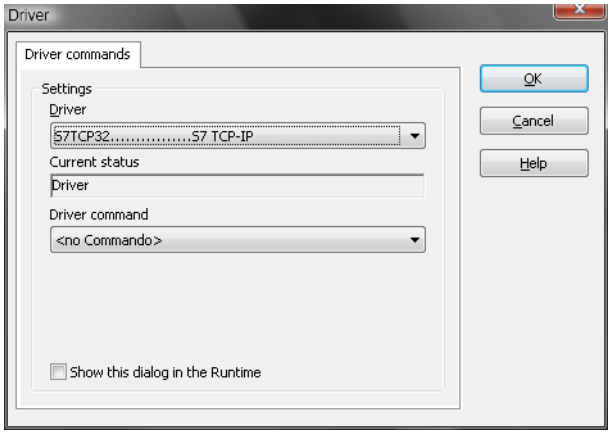
9. Changing the driver mode in Runtime

This chapter describes standard functions that are valid for most zenon drivers. Not all functions described here are available for every driver. For example, a driver that does not, according to the data sheet, support a modem connection also does not have any modem functions.

Driver commands are used to influence drivers using zenon; start and stop for example.
The engineering is implemented with the help of function **Driver commands**. To do this:

- create a new function
- select Variables -> Driver commands

- The dialog for configuration is opened



Parameter	Description
Drivers	Drop-down list with all drivers which are loaded in the project.
Current status	Fixed entry which has no function in the current version.
Driver command	Drop-down list for the selection of the command.
▸ `Start driver (online mode)`	Driver is reinitialized and started.
▸ `Stop driver (offline mode)`	Driver is stopped. No new data is accepted. Note: If the driver is in offline mode, all variables that were created for this driver receive the status `switched off` (`OFF`; Bit 20).
▸ `Driver in simulation mode`	Driver is set into simulation mode. The values of all variables of the driver are simulated by the driver. No values from the connected hardware (e.g. PLC, bus system, ...) are displayed.
▸ `Driver in hardware mode`	Driver is set into hardware mode. For the variables of the driver the values from the connected hardware (e.g. PLC, bus system, ...) are displayed.
▸ `Driver-specific command`	Enter driver-specific commands. Opens input field in order to enter a command.
▸ `Driver - activate set setpoint value`	Write set value to a driver is allowed.
▸ `Driver - deactivate set setpoint value`	Write set value to a driver is prohibited.
▸ `Establish connecton with modem`	Establish connection (for modem drivers) Opens the input fields for the hardware address and for the telephone number.
▸ `Disconnect from modem`	Terminate connection (for modem drivers)
Show this dialog in the Runtime	The dialog is shown in Runtime so that changes can be made.

DRIVER COMMANDS IN THE NETWORK

If the computer, on which the **driver command** function is executed, is part of the zenon network, additional actions are carried out. A special network command is sent from the computer to the project server, which then executes the desired action on its driver. In addition, the Server sends the same driver command to the project standby. The standby also carries out the action on its driver.

This makes sure that Server and Standby are synchronized. This only works if the Server and the Standby both have a working and independent connection to the hardware.

10. Error analysis

Should there be communication problems, this chapter will assist you in finding out the error.

10.1 Analysis tool

All zenon modules such as Editor, Runtime, drivers, etc. write messages to a joint log file. To display them correctly and clearly, use the Diagnosis Viewer (main.chm::/12464.htm) program that was also installed with zenon. You can find it under Start/All programs/zenon/Tools 7.50 -> Diagviewer.

zenon driver log all errors in the LOG files. The default folder for the LOG files is subfolder **LOG** in directory `ProgramData`, example:

`%ProgramData%\COPA-DATA\LOG`. LOG files are text files with a special structure.

Attention: With the default settings, a driver only logs error information. With the **Diagnosis Viewer** you can enhance the diagnosis level for most of the drivers to "Debug" and "Deep Debug". With this the driver also logs all other important tasks and events.

In the Diagnosis Viewer you can also:

- Follow newly-created entries in real time
- customize the logging settings
- change the folder in which the LOG files are saved

Note:

1. The Diagnosis Viewer displays all entries in UTC (coordinated world time) and not in local time.
2. The Diagnosis Viewer does not display all columns of a LOG file per default. To display more columns activate property **Add all columns with entry** in the context menu of the column header.
3. If you only use **Error-Logging**, the problem description is in the column **Error text**. For other diagnosis level the description is in the column **General text**.
4. For communication problems many drivers also log error numbers which the PLC assigns to them. They are displayed in **Error text** or **Error code** or **Driver error parameter** (**1** and **2**). Hints on the meaning of error codes can be found in the driver documentation and the protocol/PLC description.
5. At the end of your test set back the diagnosis level from **Debug** or **Deep Debug**. At **Debug** and **Deep Debug** there are a great deal of data for logging which are saved to the hard drive and which can influence your system performance. They are still logged even after you close the **Diagnosis Viewer**.

> **Attention**
>
> In Windows CE errors are not logged per default due to performance reasons.

You can find further information on the Diagnosis Viewer in the Diagnose Viewer (main.chm::/12464.htm) manual.

10.2 Check list

Check your system for errors using the driver documentation; for example correct driver configuration, addressing of the variables in zenon projects, report settings in the control unit, etc.

Check most of all:

- Is the device (IEC 61850 server) connected to the power grid and ready for use?
  - Can a TCP connection to the device be created (check this using a 'ping') via the defined port (check this via Telnet)?
  - Is communication between this device and another IEC 61850 client possible?
  - Is the driver in hardware mode and not in simulation mode?
- Have you evaluated the errors that are logged in the log file of the driver?
  - Which errors occurred?
  - Was the diagnosis logging activated, at least for the error level and for all driver modules?
  - Have you taken into account that the times in the Diagnosis Viewer are always UTC?
  - Is the Diagnose Viewer set so, that all registered and all columns are diplayed with the content (filtering)?
- Variable addressing:
  - Do all variables have the right addresses?
  - Are the names (or identifications) of the variables valid (SERVER!LD/LN/DO/DA[FC])?
  - Do the variables have the right driver object type, data type and value range?
- General information and reporting:
  - Is at least one variable from Logical Device polled?
  - Does the RCB contain the whole data objects (not just the data attributes */stVal, but also */q and */t)?
  - Does the **max. auto used URCBs** property have the correct value?
  - Is reporting available in the device?
  - If defined in the SCL file: Does the report ID of Buffered RCB contain the string "zenon_Computername*"?

- For Dynamic Data Set: Is the data set describable in the Reporting Control Block?
- Have you taken into account limitations of the device, such as the maximum number of elements in the data set etc. ?

11. Appendix A - Description of the ‘Functional Constraints’ (FCs):

FC	Semantics	Description
ST	Status information	Stands for status information, the value of which can be read, replaced, reported and logged, but not written.
MX	Measurands (analogue values)	Stands for status information, the measured size of which can be read, replaced, reported and logged, but not written.
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	Configuration	represents a configuration information whose value may be written and read.
DC	Description	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.

12. Appendix B - Abbreviations for data object/data attribute

Information

Appendix B - Data object / data attribute is only available in English.

Term	Description
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 are defined in IEC 61850-8-1:

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

13. Appendix C - Conformance statement

Information

The Confirmance Statement of the **iec850 driver** for the IEC61850 protocol is in English only.

13.1 Protocol implementation conformance statement (PICS)

This document specifies the PICS of the IEC 61850 interface in the client system: "**zenon Supervisor**/Energy Edition", Version 7.50, further referred to as "Client". The zenon Editor and Runtime are further referred to as "HMI".

A.1 - BASIC ACSI CONFORMANCE STATEMENT

Client-Server roles		**Client / subscriber**	**Server / publisher**	**Value / comments**
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		**Client / subscriber**	**Server / publisher**	**Value / comments**
B21	SCSM: IEC 61850-8-1 used	Y		
B22	SCSM: IEC 61850-9-1 used	N		deprecated
B23	SCSM: IEC 61850-9-2 used	N		
B24	SCSM: other	N		
Generic substation event model (GSE)		**Client / subscriber**	**Server / publisher**	**Value / comments**
B31	Publisher side	--	N	for GOOSE see zenon Logic
B32	Subscriber side	N	--	see zenon Logic
Transmission of sampled value model (SVC)		**Client / subscriber**	**Server / publisher**	**Value / comments**
B41	Publisher side	-	N	
B42	Subscriber side	N	-	for 90-5 see zenon Logic
- = not applicable **Y** = supported **N** or **empty** = not supported				

A.2 - ACSI MODELS CONFORMANCE STATEMENT

IF SERVER SIDE (B11) AND/OR CLIENT SIDE (B12) SUPPORTED

	Client / subscriber	**Value / comments**

M1	Logical device	Y	
M2	Logical node	Y	
M3	Data	Y	
M4	Data set	Y	
M5	Substitution	Y	
M6	Setting group control	Y	
Reporting		**Client / subscriber**	**Value / comments**
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		**Client / subscriber**	**Value / comments**
M9	Log control	N	
M9-1	IntgPd	N	
M10	Log	N	

general		Client / subscriber	Value / comments
M11	Control	Y	
M17	File Transfer	Y	
M18	Application association	Y	
M19	GOOSE Control Block	Y	
M20	Sampled Value Control Block	Y	

IF GSE (B31/B23) IS SUPPORTED

		Client / subscriber	Value / comments
M12	GOOSE	N	
M13	GSSE	N	

IF SVC (B41/B42) IS SUPPORTED

		Client / subscriber	Value / comments
M14	Multicast SVC	N	
M15	Unicast SVC	N	

FOR ALL IEDS

		Client / subscriber	Value / comments
M16	Time	Y	Time of Operating System must be available with required accuracy

Y = service is supported

N or **empty** = service is not supported

A.3 - ACSI SERVICE CONFORMANCE STATEMENT

SERVER

	AA: TP/MC	Client (C)	Comments

S1	GetServerDirectory (LOGICAL-DEVICE)	TP	Y	
Application association		**AA: TP/MC**	**Client (C)**	**Comments**
S2	Associate		Y	
S3	Abort		Y	
S4	Release		N	
Logical device		**AA: TP/MC**	**Client (C)**	**Comments**
S5	GetLogicalDeviceDirectory	TP	N	
Logical node		**AA: TP/MC**	**Client (C)**	**Comments**
S6	GetLogicalNodeDirectory	TP	Y	only for DataSets
S7	GetAllDataValues	TP	N	

DATA

		AA: TP/MC	**Client (C)**	**Comments**
S8	GetDataValues	TP	Y	▶ polling: FCD ▶ watchdog, scan, SBO: FCDA
S9	SetDataValues	TP	Y	
S10	GetDataDirectory	TP	Y	
S11	GetDataDefinition	TP	N	
Data set		**AA: TP/MC**	**Client (C)**	**Comments**
S12	GetDataSetValues	TP	N	
S13	SetDataSetValues	TP	N	
S14	CreateDataSet	TP	Y	
S15	DeleteDataSet	TP	Y	
S16	GetDataSetDirectory	TP	Y	
Substitution		**AA: TP/MC**	**Client (C)**	**Comments**
S17	SetDataValues	TP	Y	

Setting group control		AA: TP/MC	Client (C)	Comments
S18	SelectActiveSG	TP	Y	
S19	SelectEditSG	TP	Y	
S20	SetSGValues	TP	Y	
S21	ConfirmEditSGValues	TP	Y	
S22	GetSGValues	TP	Y	
S23	GetSGCBValues	TP	Y	

REPORTING

Buffered report control block (BRCB)		AA: TP/MC	Client/ subscriber	Comments
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)		**AA: TP/MC**	**Client/ subscriber**	**Comments**
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		AA: TP/MC	Client/ subscriber	Comments
S30	GetLCBValues	TP	Y	
S31	SetLCBValues	TP	Y	
Log		**AA:**	**Client/**	**Comments**

		TP/MC	subscriber	
S32	QueryLogByTime	TP	N	
S33	QueryLogAfter	TP	N	
S34	GetLogStatusValues	TP	Y	

GENERIC SUBSTATION EVENT MODEL (GSE)

GOOSE-CONTROL-BLOCK		AA: TP/MC	Client/ subscriber	Comments
S35	SendGOOSEMessage	MC	N	
S36	GetGoReference	TP	N	
S37	GetGOOSEElementNumber	TP	N	
S38	GetGoCBValues	TP	Y	
S39	SetGoCBValues	TP	Y	
GSSE-CONTROL-BLOCK		**AA: TP/MC**	**Client/ subscriber**	**Comments**
S40	SendGSSEMessage	MC	N	deprecated
S41	GetGsReference	TP	N	deprecated
S42	GetGSSEDataOffset	TP	N	deprecated
S43	GetGsCBValues	TP	Y	deprecated
S44	SetGsCBValues	TP	Y	deprecated

TRANSMISSION OF SAMPLED VALUE MODEL (SVC)

Multicast SVC		AA: TP/MC	Client (C)	Comments
S45	SendMSVMessage	MC	N	
S46	GetMSVCBValues	TP	Y	
S47	SetMSVCBValues	TP	Y	
Unicast SVC		**AA: TP/MC**	**Client (C)**	**Comments**
S48	SendUSVMessage	TP	N	
S49	GetUSVCBValues	TP	Y	
S50	SetUSVCBValues	TP	Y	

CONTROL

		AA: TP/MC	Client (C)	Comments
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	

FILE TRANSFER

		AA: TP/MC	Client (C)	Comments
S57	GetFile	TP	Y	
S58	SetFile	TP	N	
S59	DeleteFile	TP	Y	
S60	GetServerDirectory (FILE-SYSTEM)	TP	Y	

TIME

		Client (C)	Comments
T1	Time resolution of internal clock	Operating System (T1)	
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 decimal places of LREAL when t-attribute created as variable (T3)

13.2 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 - are supported.

- Data types INT64 and INT128 are fully supported on MMS, but their values in the Runtime only in range of lowest 32 bits as mapped to **DINT**.
- Data type ARRAY OF - only single-level arrays are supported; items of this data type are read-only.
- Object References in received Service Tracking data are not supported with '@' wildcard instead the IED Name.

14. Appendix D - PIXIT

Info

*The PIXIT document for the certification test of the IEC850 driver "**zenon Supervisor**/Energy Edition", Version 7.50 is only available in English.*

PIXIT based on a template provided by the UCA International Users Group.

INTRODUCTION

This document specifies the protocol implementation extra information for testing (PIXIT) of the IEC 61850 interface in the client system: "**zenon Supervisor**/Energy Edition", Version 7.50, further referred to as “Client”. The zenon Editor and Runtime are further referred to as "HMI".

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 Edition 2 interface” by UCA International Users Group.

PIXIT FOR CONFIGURATION

ID	Description	Value / Clarification
Cf1	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.
Cf2	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 if online read of data model fails. Preconfigured items are used according current online definition or omited if not available online. The SCL-file is used in Editor for pre-configuration of HMI projects, e.g. in driver configuration Wizard and in offline import of variables. Please refer client documentation for more details.

PIXIT FOR ASSOCIATION MODEL

ID	Description	Value / Clarification
As1	Garanteed number of servers that can set-up an association simultaneously (one association per server)	per design: 65535 tested: 1000
As2	Lost connection detection time range (default range of TCP_KEEPALIVE is 1 – 20 seconds)	current TCP/IP timeout of Operating System
As3	Lost (abort) connection retry time	30 seconds
As4	Is authentication supported?	Y
As5	What is the maximum and minimum MMS PDU size?	Max MMS PDU size 65535 Min MMS PDU size depends on size of largest DO in server's data model
As6	What is the typical startup time after a power supply interrupt?	The client works only together with the zenon Runtime, these are Windows applications. zenon Runtime + client may work on redundant PCs then by power loss or Hardware problem on main PC the secundary system takes (seamless) over. 1. Windows does not guarantee the files will be not corrupted after power failure. We recommend to use Redundant systems and/or an UPS to prevent loss of data stored on HDD.

		2. The start-up time for the client depends of HMI project design and PC performance. In typical HMI projects client opens TCP/IP session to a server after few (1-4) seconds since application start.
As7	How does the client disconnect from the server?	service: Abort

PIXIT FOR SERVER MODEL

ID	Description	Value / Clarification
Sr1	Maximum object identification length?	LD: 128 chars/octets FC: 2 chars/octets LN: 16 chars/octets DO: 128 chars/octets DA: 128 chars/octets in sum max 256 chars/octets (Symbolic Address)
Sr2	Does client support autodescription?	Y During association client always reads current data model of server via online services: GetServerDirectory and GetDataDirectory. During this process client skipes not present LD and reads LN of created HMI variables; then skipes not present DO and DA. After association client does not use items not present on server. The HMI variables, if created but mirroring non-existing items - have I-Bit*.
Sr3	Describe how to view data values	as value of HMI variable
Sr4	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

Sr5	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
Sr6	Describe how to view/display quality values	1. the status bits of */stVal variable in HMI, if created (only selected bits; please refer client documentation for more details) 2. the value of */q variable in HMI, if created
Sr7	Describe how to force a SetDataValues request	Set value to the variable created in HMI, e.g. via screen element or write set value function.
Sr8	Describe how to force a GetDataValues request	no need to force - client does automatically: ▸ client does GetDataValues periodically for all not reported FC$DO when at least one DA is mirrored to currently advised HMI variable. ▸ when no variables are currently advised, client uses GetDataValues(LLN0STMod$stVal) as automatic watchdog for connection loss detection.
Sr9	Describe how to force a GetAllDataValues request	service not used
Sr10	Does the client support writing blkEna values?	Y
Sr11	Describe how the client behaves in case of:	see also Sr2
▪	▸ GetDataDefinition response-	▸ service not used
▪	▸ GetDataDefinition response+ with more or less attributes as expected	▸ service not used

▪	▸ GetLogicalDeviceDirectory response-	▸ service not used
▪	▸ GetAllDataValues response-	▸ service not used
▪	▸ GetAllDataValues response+ with more or less attributes as expected	▸ service not used
▪	▸ GetDataValues response-	▸ log error entry; value of HMI variable will be not updated - have previously received value or null
▪	▸ GetDataValues response+ with more or less attributes as expected	to request process data the client uses GetDataValues on FC$DO, not DA level (see Sr8). ▸ HMI variables mirroring items not present in server's data model have I-Bit* (see Sr2) and are not requested ▸ items present in data model are requested but if not included in GetDataValue response then adequate HMI variables are not updated ▸ additional items are ignored.
▪	▸ SetDataValues response-	▸ log error entry
Sr12	Which time quality attributes from the server are used in the client	▸ N Leap Second Known, ▸ Y ClockFailure ▸ Y Clock not synchronized ▸ Y Accuracy
Sr13	Describe how to view time quality attributes	client maps time quality bits received in t-attribute to status bits of HMI variables responding the same FC$DO as following: ▸ ClockNotSynchronised - status bits T_INVAL & T_EXTERN ▸ ClockFailure - status bits T_INVAL & T_INTERN and HMI variable values have local timestamp ▸ Accuracy < 11111b - client truncates fraction of seconds of timestamp of HMI variable
	*I-Bit is status bit INVALID of HMI variables mirroring DO/DA. Please refer HMI documentation for more details.	

PIXIT FOR DATA SET MODEL

ID	Description	Value / Clarification
Ds1	Describe how to force a GetDataSetValues request	service not used

Ds2	Describe how to force a SetDataSetValues request	service not used
Ds3	Describe how to force a DeletaDataSet request	client does it automatically when in client configuration user changes name and members for Dynamic Data Sets.
Ds4	Describe how the client handles following dataset mismatches between the SCL and the data sets exposed via MMS: ▸ new dataset element ▸ missing dataset element ▸ Reordered dataset members in a dataset of a different data type ▸ Reordered dataset members in a dataset of the same data type	No mismatch possible - client uses autodescription (see Sr2)
Ds5	Describe how the client behaves in case of: ▸ GetLogicalNodeDirectory(DATA-SET) response- ▸ GetDataSetDirectory response- ▸ GetDataSetValues response- ▸ SetDataSetValues response-	LOG entry + data will be polled (no reporting)
Ds6	Maximum name length: ▸ for dataset ▸ for dataset member, including LD and FC	▸ MMS: 128/16$128, HMI: 256 ▸ MMS: 128/128, HMI: 256
Ds11	Describe how to force a CreateDataSet request: 1. persistent 2. non-persisten	1. client creates automatically new dataset before it enables the RCB where client is configured to use Dynamic Data Set; and when this dataset does not exist in IED yet. 2. not used
Ds12	Describe how the client behaves in case of: ▸ CreateDataSet response- ▸ DeleteDataSet response-	▸ log entry, and data will be polled ▸ client retries to delete the data set by each next association

PIXIT FOR SUBSTITUTION MODEL

ID	Description	Value / Clarification
Sub 1	Describe how to substitute a value	following steps: 1. create HMI variables for attributes with FC=SV (*/subEna, subQ, subID and

		e.g. subVal) 2. set values in HMI variables (*/subVal, subQ, subID) to values the IED should then use while substituting 3. set value TRUE to HMI variable */subEna. Please refer HMI documentation for more details about setting variable values
	<additional items>	Hint: when IED does not support substitution there is an alternative method in the HMI to substitute HMI variable values: a HMI variable can be switched to "alternative value" and than the HMI - and not IED - substitute the value. Please refer description of status bit ALT_VAL in the HMI documentation for more details.

PIXIT FOR SETTING GROUP CONTROL MODEL

ID	Description	Value / Clarification
Sg1	How can the client be forced to send a GetLogicalNodeDirectory(SGCB) request	service not used Note. to (re)read data model via GetDataDirectory force reconnect, e.g. using HMI "driver commands" to stop and start client
Sg2	Describe how to change the active setting group	set value of adequate variable in HMI project (ActSG or EditSG)
Sg3	Describe how to get the actual setting group values	create variables in the HMI project for adequate attributes and advise these variables
Sg4	Describe how to edit setting group values	set value of adequate variables in HMI project
Sg5	Describe how the client behaves in case of: ▸ GetSGCBValues response- ▸ SelectEditSG response- ▸ SetEditSGValue response- ▸ SelectActiveSG response- ▸ ConfirmEditSGValues response-	client uses autodescription and then requests only attributes present in server's data model. By write failure client logs error; by negative responses HMI variables are not updated.
	<additional items>	

PIXIT FOR REPORTING MODEL

ID	Description	Value / Clarification
Rp1	Does the client search for RCB in all logical nodes?	Y - All logical nodes

	If not, specify the logical nodes	▸ LLN0 always ▸ any LN when RCB name is defined in the client configuration or at least one variable in this LN exists in HMI ▸ in addition: BRCBs in all LDevices/LN - if BRCB.RptID=zenon_computername_*
Rp2	Which dynamic RCB attributes are/can be configured by the client	by client configuration: ▸ N* RptID ▸ Y DataSet ▸ Y Optional fields ▸ Y Trigger conditions ▸ Y Buffer time ▸ Y Integrity period *see also <addititonal items>
Rp3	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
Rp4	The supported trigger conditions are	Y integrity Y data change Y quality change Y data update Y general interrogation
Rp5	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 (*BRCBs only) N conf-rev client can process reports with any additional optional field, not only the minimum; the optional fields not required may be ignored.
Rp6	Does the client support segmented reports	Y
Rp7	Does the client support pre-assigned RCB	Y for more details see manual of IEC850 driver configuration wizard
Rp8	Does the client support reported data set containing structured data objects or data attributes?	Y reporting of data objects Y* reporting of data attributes *not recommended as this kind of dataset does not assure that each trigged report will contain current q and t attribute values.

Rp9	Describe how the client does respond when an URCB is already reserved	log entry + trying to use free one but only if configured to use automatic (not statical) enabling of URCBs. for staticallly assigned URCB see Rp10
Rp10	Describe how the client does respond when a BRCB is already reserved	log entry and 'connection state' indicator about failure; data will be polled optionally, per default each 7s, the client tries periodically to enable the RCB.
Rp11	Describe how the client does respond on a SetBRCBValues(EntryID) respond-	error log entry; as next client tries to enable BRCB anyway and sends GI; and process buffered data if IED reports whole contents of buffer
Rp12	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 - client uses autodescription - see Sr2
Rp13	Describe how the client detect reporting configuration changes (mismatches). Does it check the "configuration revision" attributes and/or does it check the dataset members? Is the dataset update done online or offline?	▸ N* Check ConfRev ▸ Y Check dataset members ▸ online update at each associate *optionally, an unexpected change of ConfRev value, if created adequate HMI variable, may produce e.g. an user alarm
Rp14	Describe how to force the client to change the RCB buffertime	BufTm is a setting in the client configuration see also <addititonal items>
Rp15	Does client set server TrgOps.GI prior to first issuance of GI command?	Yes, if use of client's TrgOps is configured in the client configuration
Rp16	Describe how to force the client to send the GI request	▸ client does it automatically by enabling of RCB and by reconnect. ▸ GI on user demand - see <addititonal items>
Rp17	Describe how to force the client to enable a RCB	settings in the client configuration
Rp18	Describe how the client does respond when a report control block is renamed or deleted ▸ Does it prevent reading the deleted RCB ▸ If it reads the missing RCB, how does it handle the GetURCBValues or GetBRCBValues response-	▸ no reading of deleted RCB during autodescription (see Sr2) client checks if all in client (statically) configured RCBs are existing in the IED's data model; if not: log entry and 'connection state' indicator about failure; data will be polled
Rp19	Describe how the client behaves in case of: ▸ SetURCBValues response- ▸ Unsupported optional fields	▸ if SetURCBValues(RptEna) fails -> see Rp9 and Rp10 ▸ if SetURCBValues(DataSet) fails -> error log entry; even when RCB is

	▸ Unsupported trigger condition(s)	enabled data will be polled ▸ if SetURCBValues fails for other attributes -> client uses enabled RCB (like it is). ▸ client supports all OptFlds and TrgOps

Rp20	Describe how the client behaves in case of: ▸ Buffer overflow	by reconnect client always sends GI and process received buffered data; to purge buffer is an option.
Rp21	Describe how to force the client to send SetBRCBValues request for ▸ EntryID ▸ PurgeBuf	▸ client sends by reconnect the last received EntryID auromaticaly ▸ client sends PurgeBuf automatically, but only by HMI start; this may be deactivated. ▸ PurgeBuf on user demand - see <addititonal items>
Rp22	Does the client support writing resvTms	Y
Rp23	Does the client support reading owner	N* see <additional items>; *compare Rp13
	<additional items>	In addition to settings for RCBs in the client configuration, a user may create HMI variables for any attribute with FC=BR|RP (manually or by online import). Then in HMI the current value of RCB's attribute may be read; and may be written (e.g. GI) Note: written eventually only while RptEna is set to FALSE as else IED should refuse the updates of e.g. RptId, ResvTms etc.

PIXIT FOR LOGGING MODEL

ID	Description	Value / Clarification
	<additional items>	logging services are not supported

PIXIT FOR GOOSE CONTROL BLOCK MODEL

ID	Description	Value / Clarification
	<additional items>	system user may create HMI variables for any attribute with FC=GO (manually or by online import). Then in HMI the current values of GoCB's attributes may be read; and may be written (e.g. GoEna) Note: written eventually only while GoEna is set to FALSE as else IED should refuse the updates of e.g. GoId etc.

PIXIT FOR CONTROL MODEL

ID	Description	Value / Clarification

Ctl1	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 client checks current ctlModel value automatically and then executes command in corresponding control model; no need for additional configuration steps or pre-configuration from SCL see client documentation for more details
Ctl2	Is Time activated operate (operTm) supported?	N
Ctl3	Is "operate-many" supported?	N
Ctl4	Can the client set the test flag?	Y
Ctl5	What check conditions can be set?	Y synchrocheck Y interlock-check there are two methods to set check conditions: 1. by setting a value to HMI variable */Oper.Check[CO] if created 2. by configuring in Command Processing modul of HMI property 'Qualifier of Command' see HMI documentation for details
Ctl6	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
Ctl7	Describe if and how the client sets/increments the ctlNum	by each use* of ctlVal the client increments ctlNum from 0 to 127 and next turns to 0. * also by ctlModel=0 (no control possible)
Ctl8	What does the client when its receives a LastApplicationError and describe how to view the additional cause?	HMI user may create an internal "Command Info" variable (pro */Oper structure) for received AddCause item value and variables ControlRun and ControlState showing current state and success/fail of command. These internal variables can be displayed and/or stored see client documentation for more details
Ctl9	What does the client when its receives a Select, SelectWithValue or Operate respond negative ?	client sets corresponding values to intern "Command Info" (see Ctl8) variables ControlRun (-1=error) and AddCause; optionally writes a log entry and sets by HMI variable Oper.ctlVal status bit N_CONF see client documentation for more details

Ctl10	Can the client change the control model via online services?	Y
Ctl11	What does the client when the ctlModel is not initialized in the SCL?	client does not need ctlModel value in SCL; client checks current ctlModel value automatically via online services each time before is executing command; no mismatch possible
Ctl12	What does the client when the ctlModel in SCD and in SERVER SIMULATOR is different?	see Ctl11
Ctl13	Describe how to view a ▸ CommandTermination request+ ▸ CommandTermination request- ▸ TimeActivatedOperateTermination request+ and request-	▸ CommandTermination reception is visible in 'Command Info' internal HMI variables (see Ctl8); it may ends action timeout in HMI modul Command Processing, if used ▸ service TimeActivatedOperate is not supported Note: there is no special handling for negative response, only AddCause update (if server sends some). By use of Command Processing module for final success (or failure) of commnand the change of related ST/MX attribute cares.
	<additional items>	

PIXIT FOR TIME AND TIME SYNCHRONIZATION MODEL

ID	Description	Value / Clarification
Tm1	Described how to view the internal time & quality or how to expose the timestamp and timestamp quality via the IEC 61850 interface	▸ View: client uses Windows time (local PC clock) and does not cooperate directly with a SNTP application. In HMI a user may create additional dynamical elements to view internal PC time and check clock quality using tools of own choice, e.g. VBA/VSTA or SCADA Logic. ▸ Expose: in Operate and SelectWithValue services, client uses current value of additional HMI variables 'Settings': TimeQuality and TimeAccuracy.
Tm2	What time quality bits are supported?	In Operate and SelectWithValue services variable TimeQuality may* update: ▸ Y** LeapSecondsKnown ▸ Y** ClockFailure ▸ Y ClockNotSynchronized *depending e.g. SNTP agent installed on the

		PC with HMI and user-implemented solution. **see <additional items>
Tm3	What is the behavior when the time synchronization signal/messages are lost?	▸ PC (Windows) internal clock runs not synchronized anymore ▸ local SNTP agent can detect the signal loss and this information can be processed in HMI ▸ as result in HMI the variable TimeQuality may change value, if user adequate designed HMI project
Tm4	When is the quality bit "Clock failure" set?	Optionaly HMI may force the Client to set ClockFailure (TimeQuality bit 1) when PC clock was reseted or for test purpose. see <additional items>
Tm5	When is the quality bit "Clock not synchronised" set?	TimeQuality bit 2 set if SNTP agent informs HMI that it detects the loss of time synchronization signal
	<additional items>	Note: practical use for commands given from HMI on Windows PC has only ClockNotSynchronized: ▸ PC having severe Hardware problem - resulting in ClockFailure - is probably no more able to work in general. We recommend to use Redundant systems - compare As6. ▸ LeapSecondsKnown is not commonly supported in SNTP agents and has low use in commands

PIXIT FOR FILE TRANSFER MODEL

ID	Description	Value / Clarification
Ft1	Describe when or how to force the client to request GetServerDirectory(FILE) and what it does with the responded filenames	client gives possibility to create two internal 'File transfer' 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 see client documentation for more details
Ft2	Does the client uses a wildcard in the GetServerDirectory(FILE) request	No
Ft3	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

Ft4	Does the client support IED's that use the fileseparator	▸ N "/" ▸ Y "\"
Ft5	What is the maximum file name size including path	Operating Sytem limit (Windows)
Ft6	Can the client read a file with size 0	Y
Ft7	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
Ft8	Maximum file size	per design 4 GB practically* 25 MB * transfer of 25 MB may take about 30 seconds, bigger transfer per single file wasn't tested
Ft9	Describe how the client behaves in case of: ▸ GetFile response- ▸ GetFileAttributes response- ▸ SetFile response-	▸ Command variable value changes to "GET ERROR" or "DIR ERROR" (or "DEL ERROR" by service DeleteFile) ▸ SetFile is not supported
	<additional items>	For DeleteFile set Command variable value to "DEL <fileSpec>"

PIXIT FOR SERVICE TRACKING MODEL

ID	Description	Value / Clarification
Tr1	Which tracking services are supported by the client: ▸ BrcbTrk ▸ UrcbTrk ▸ LocbTrk ▸ GocbTrk ▸ MsvcbTrk ▸ UsvcbTrk ▸ SgcbTrk ▸ SpcTrk ▸ DpcTrk ▸ IncTrk ▸ EncTrk ▸ ApcFTrk ▸ ApcIntTrk	▸ Y* - BrcbTrk ▸ Y* - UrcbTrk ▸ Y* - LocbTrk ▸ Y* - GocbTrk ▸ Y* - MsvcbTrk ▸ Y* - UsvcbTrk ▸ Y* - SgcbTrk ▸ Y - SpcTrk ▸ Y - DpcTrk ▸ Y - IncTrk ▸ Y - EncTrk ▸ Y - ApcFTrk ▸ Y - ApcIntTrk

	▸ BscTrk ▸ IscTrk ▸ BacTrk ▸ GenTrk	▸ Y - BscTrk ▸ Y - IscTrk ▸ Y - BacTrk ▸ Y - GenTrk *) per default only CST attributes, the rest of data attributes optionally. See client documentation for more details.
Tr2	Describe how to view the tracking objects or their attributes	in additional HMI variables with reference: *LN/DO_xxx[SR] and 'driver object type' = 'service tracking', where xxx is "allCTS" or name of data attribute in CTS class or in another common data class possible in LTRK. See client documentation for more details
	<additional items>	

15. Anhang E - TICS

This document specifies the TICS of the IEC 61850 interface in the client system: "**zenon Supervisor/Energy Edition**", Version 7.50, further referred to as "Client". The zenon Editor and Runtime are further referred to as "HMI".

MANDATORY EDITION 2 TISSUES

Below tables give an overview of the applicable mandatory Tissues.

SUPPORTED BY CLIENT:

- **"Y":** means that the client supports servers that have implemented the respective tissue.
- **"ni":** No impact on testing
- **"na":** not applicable if the client does not support the corresponding ACSI service(s)

15.1 Part 6

Tissue	Description	Supported by client
658	Tracking related features	Y
663	FCDA element cannot be a "functionally constrained logical node"	Y
668	Autotransformer modeling	Y
719	ConfDataSet - maxAttributes definition is confusing	na
721	Log element name	na
768	bType VisString65 is missing	Y
779	object references	na
788	SICS S56 from optional to mandatory	Y
789	ConfLdName as services applies to both server and client (supportsLdname)	Y
804	valKind and IED versus System configuration (valimport)	na
806	Max length of log name inconsistent between -6 and -7-2	ni
807	Need a way to indicate if "Owner" present in RCB	Y
822	Extension of IED capabilities	Y
823	ValKind for structured data attributes	Y
824	Short addresses on structured data attributes	Y
825	Floating point value	Y
845	SGCB ResvTms	Y
853	SBO and ProtNs	Y
855	Recursive SubFunction	na
856	VoltageLevel frequency and phases	ni
857	Function/SubFunction for ConductingEquipment	na
886	Missing 8-1 P-types	Y
901	tServices as AP or as IED element	Y
936	SupSubscription parameter usage is difficult	na
1175	IPv6 address lowercase only	na

15.2 Part 7-1

Tissue	Description	Supported by client
828	Data model namespace revision IEC 61850-7-4:2007[A]	Y
1151	simulated GOOSE disappears after 1st appearance when LPHD.Sim = TRUE	na
1196	Extensions to standardized LN classes made by third parties Note: this tissue is not final yet	na

15.3 Part 7-2

Tissue	Description	Supported by client
778	AddCause values – add value not- supported	Y
780	What are unsupported trigger option at a control block?	na
783	TimOper Resp- ; add Authorization check	na
786	AddCause values 26 and 27 are switched	Y
820	Mandatory ACSI services (use for PICS template)	Y
858	typo in enumeration ServiceType	ni
861	dchg of ConfRev attribute	Y
876	GenLogiclNodeClass and SGCB, GoCB, MsvCB, UsvCB	ni
1038	Loss of Info Detection After Resynch	Y
1062	Entrytime not used in CDC	ni
1071	Length of DO name	ni
1091	The sentence "The initial value of EditSG shall be 0", has to be stated in part 7.2 not in 8.1	Y
1127	Missing owner attribute in BTS and UTS	Y
1163	Old report in URCB	ni
1202	GI not optional	ni

15.4 Part 7-3

Tissue	Description	Supported by client
697	persistent command / PulseConfig	Y
698	Wrong case is BAC.dB attribute	Y
722	Units for 'h' and 'min' not in UnitKind enumeration.	ni
919	Presence Condition for sVC	ni
925	Presence of i or f attribute - Problem with writing	Y
926	Presence Conditions within RangeConfig	Y

15.5 Part 7-4

Tissue	Description	Supported by client
671	mistake in definition of Mod & Beh	Y
674	CDC of ZRRC.LocSta is wrong	Y
675	SIML LN	ni
676	Same data object name used with different CDC	Y
677	MotStr is used with different CDC in PMMS and SOPM LN classes	Y
679	Remove CycTrMod Enum	Y
680	SI unit for MHYD.Cndct	ni
681	Enum PIDAlg	ni
682	ANCR.ParColMod	ni
683	Enum QVVR.IntrDetMth	ni
685	Enum ParTraMod	ni
686	New annex H - enums types in XML	Y
694	Data object CmdBlk	ni
696	LSVS.St (Status of subscription)	ni
712	interpretation of quality operatorBlocked	Y
713	DO Naming of time constants in FFIL	Y
724	ANCR.Auto	Y
725	Loc in LN A-group	Y
734	LLN0.OpTmh vs. LPHD.OpTmh	Y
735	ISAF.Alm and ISAF.AlmReset	Y
736	PFSign	Y
742	GAPC.Str, GAPC.Op and GAPC.StrVal	Y
743	CCGR.PmpCtl and CCGR.FanCtl	Y
744	LN STMP, EEHealth and EEName	Y
772	LPHD.PwrUp/PwrDn shall be transient	Y
773	Loc, LocKey and LocSta YPSH and YLTC	Y

774	ITCI.LocKey	Y
775	KVLV.ClsLim and OpnLim	ni
776	LPHD.OutOv/InOv and LCCH.OutOv/InOv	ni
800	Misspelling in CSYN	Y
802	CCGR and Harmonized control authority	Y
808	Presence condition of ZMOT.DExt and new DOs	Y
831	Setting of ConfRevNum in LGOS	Y
838	Testing in Beh=Blocked	Y
844	MFLK.PhPiMax, MFLK.PhPiLoFil, MFLK.PhPiRoot DEL->WYE	Y
849	Presence conditions re-assessing in case of derived statistical calculation	Y
877	QVUB -settings should be optional	Y
909	Remove ANCR.ColOpR and ColOpL	Y
920	Resetable Counter is NOT resetable	Y
932	Rename AVCO.SptVol to AVCO.VolSpt	Y
939	Change CDC for ANCR.FixCol	Y
991	LGOS: GoCBRef (as well as LSVS.SvCBRef) should be mandatory	ni
1007	PTRC as fault indicator - Update of description required	ni
1044	TapChg in AVCO	Y
1077	Rename DOnames within LTIM	Y

Information

Tissues 675, 735, 772, 775, 776, 878 are not relevant for conformance testing

15.6 Part 8-1

Tissue	Description	Supported by client
784	Tracking of control (CTS)	Y
817	Fixed-length GOOSE float encoding	ni
834	File dir name length 64	Y
951	Encoding of Owner attribute	Y
1040	More associate error codes	Y
1178	Select Response+ is non-null value	Y