BECKHOFF New Automation Technology

Manual | EN

# TS6509

## TwinCAT 2 | PLC IEC 61400-25 Server

## Supplement | Communication

2022-11-09 | Version: 1.1

# Table of contents

# 1 Foreword

## 1.1 Notes on the documentation

This description is only intended for the use of trained specialists in control and automation engineering who are familiar with applicable national standards.
It is essential that the documentation and the following notes and explanations are followed when installing and commissioning the components.
It is the duty of the technical personnel to use the documentation published at the respective time of each installation and commissioning.

The responsible staff must ensure that the application or use of the products described satisfy all the requirements for safety, including all the relevant laws, regulations, guidelines and standards.

### Disclaimer

The documentation has been prepared with care. The products described are, however, constantly under development.
We reserve the right to revise and change the documentation at any time and without prior announcement.
No claims for the modification of products that have already been supplied may be made on the basis of the data, diagrams and descriptions in this documentation.

### Trademarks

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® and XPlanar® are registered trademarks of and licensed by Beckhoff Automation GmbH.
Other designations used in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owners.

### Patent Pending

The EtherCAT Technology is covered, including but not limited to the following patent applications and patents:
EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702
with corresponding applications or registrations in various other countries.


EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany

### Copyright

## 1.2 Safety instructions

### Safety regulations

Please note the following safety instructions and explanations!
Product-specific safety instructions can be found on following pages or in the areas mounting, wiring, commissioning etc.

### Exclusion of liability

All the components are supplied in particular hardware and software configurations appropriate for the application. Modifications to hardware or software configurations other than those described in the documentation are not permitted, and nullify the liability of Beckhoff Automation GmbH & Co. KG.

### Personnel qualification

This description is only intended for trained specialists in control, automation and drive engineering who are familiar with the applicable national standards.

### Description of symbols

In this documentation the following symbols are used with an accompanying safety instruction or note. The safety instructions must be read carefully and followed without fail!

**⚠ DANGER**

**Serious risk of injury!**

Failure to follow the safety instructions associated with this symbol directly endangers the life and health of persons.

**⚠ WARNING**

**Risk of injury!**

Failure to follow the safety instructions associated with this symbol endangers the life and health of persons.

**⚠ CAUTION**

**Personal injuries!**

Failure to follow the safety instructions associated with this symbol can lead to injuries to persons.

***NOTE***

**Damage to the environment or devices**

Failure to follow the instructions associated with this symbol can lead to damage to the environment or equipment.

**Tip or pointer**

This symbol indicates information that contributes to better understanding.

## 1.3 Notes on information security

The products of Beckhoff Automation GmbH & Co. KG (Beckhoff), insofar as they can be accessed online, are equipped with security functions that support the secure operation of plants, systems, machines and networks. Despite the security functions, the creation, implementation and constant updating of a holistic security concept for the operation are necessary to protect the respective plant, system, machine and networks against cyber threats. The products sold by Beckhoff are only part of the overall security concept. The customer is responsible for preventing unauthorized access by third parties to its equipment, systems, machines and networks. The latter should be connected to the corporate network or the Internet only if appropriate protective measures have been set up.

In addition, the recommendations from Beckhoff regarding appropriate protective measures should be observed. Further information regarding information security and industrial security can be found in our https://www.beckhoff.com/secguide.

Beckhoff products and solutions undergo continuous further development. This also applies to security functions. In light of this continuous further development, Beckhoff expressly recommends that the products are kept up to date at all times and that updates are installed for the products once they have been made available. Using outdated or unsupported product versions can increase the risk of cyber threats.

To stay informed about information security for Beckhoff products, subscribe to the RSS feed at https://www.beckhoff.com/secinfo.

# 2 Product Overview

The IEC 61400-25 Server realizes a communication between a wind turbine and one or several control stations. This TwinCAT Supplement product consists of two basic components. For the configuration of the data model you have the so called "TwinCAT Telecontrol Configurator". This configurator generates PLC Code which can be integrated into a TwinCAT PLC Project. Here the second component, the TcIEC61850Server.lib comes into operation. This PLC library integrates the MMS (Manufacturing Message Specification) based communication stack and realises the communication according to IEC 61400-25. The IEC 61850 is the base standard and the IEC 61400-25 is a specialisation for wind turbine communication.

This documentation is devided into these two components:

- TwinCAT Telecontrol Configurator [▶ 21]
- PLC communication stack [▶ 43]

### System requirements

- Programming environment:
  - XP, XPe;
  - TwinCAT Installation Level: TwinCAT PLC or higher;
  - TwinCAT System Version 2.10.0 Build >= 1340 or higher;
  - .NET Framework 3.5 for the TwinCAT Telecontrol Configurator
- Target system:
  - PC or CX (x86): XP, XPe, CE 6 (Image v3.06e or higher);
  - CX (ARM): CE 5 (Image v2.20e or higher);
  - TwinCAT PLC runtime system version 2.10.0 or higher;

### Product components

- TwinCAT Telecontrol Configurator for the configuration of the standardized data model. Logical Devices, Logical Nodes, Data Object and Data Attributes can be configurated.
- TcIEC61850Server.Lib (PLC library with the server block, this library needs to be integrated into the PLC project, all of the other libraries will automatically be added to the PLC project);
- TcACSI.Lib (Basic ACSI-Data types for the device modelling according to IEC 61850-7-2);
- TcIEC61850_8_1.Lib (SCSM: Mapping on MMS-services accoding to IEC 61850-8-1);
- TcMMS.Lib (MMS-Basic functions, function blocks, data types);
- TcULOSI.Lib (Implementation of the Session-, Presentation- and ACSE-layer);
- TcBER.Lib (BER-Decoding/Encoding);
- TcRFC1006.Lib (Implementation of the transport layer);
- TcTPKT.Lib (Transport-Packetizer);
- TcCollections.Lib (Collection-Classes);
- TcSocketHelper.Lib (TCP/IP auxiliary functions);
- TcpIp.Lib (TCP/IP basic functions);
- TwinCAT TCP/IP Connection Server;

### Installation

### Demo version

If you did not get a valid product key, you have the possibility to use a limited demo version of the IEC 61400-25 Server. Therefore, you must insert **Demo** as a product key. Now there are maximum three logical Nodes available and after an hour the connection to the Client will cut off. But this hour is enough for a basic communication test. After this hour you can do a restart of the server.

### Windows XP, XPe

The PLC libraries will be copied into the ..\TwinCAT\Plc\Lib-folder during the installation. The TwinCAT TCP/IP Connection Server is going to be listed in the TwinCAT Server list. At the TwinCAT Start the TCP/IP Connection Server starts automatically and at a TwinCAT Stop the Server stops.

### Windows CE

The product version for the runtime system under Windows CE is available as a separate product. Follow the following steps if you have a product version for Windows CE:

- Install the product on your programming-pc as usual. The PLC libraries will be copied in the .. \TwinCAT\Plc\Lib-folder during the installation.
- X86 CPU (CX1000, CX1020, IPC, ... ):
  - After the installation you will find a Cabinet-file for the CE-runtime system in this folder: **... \TwinCAT\CE\TwinCAT PLC IEC 61850 Server CE**
  - Copy the data file: **TcPlclec61850Svr_CE.I586.CAB** in a folder on the CE-runtime system.
- ARM CPU (CX9000, CX9010, ... ), TwinCAT PLC Library: IEC 60870-5-104 Understation (slave) **v2.0.4** and higher:
  - After the installation you will find a Cabinet-file for the CE-runtime system in this folder: **... \TwinCAT\CE\TwinCAT PLC IEC 61850 Server CE**
  - Copy the data file: **TcPlclec61850Svr_CE.ARMV4I.CAB** in a folder on the CE-runtime system.
- On the CE-System: Install the CE-components (by double-click on the Cabinet-file).
- Reboot the CE device. The TCP/IP Connection Server is going to be started automatically with the CE-operating system.

During the installation you will be asked if you want to work "case sensitive" or not. When you choose this option, this applies for all TwinCAT projects that are on the installation device. If you do not use this option, the intended differentiation of case sensitive from IEC 61400-25 will be ignored and everything will be communicated in capital letters.

### Introduction/Tutorial

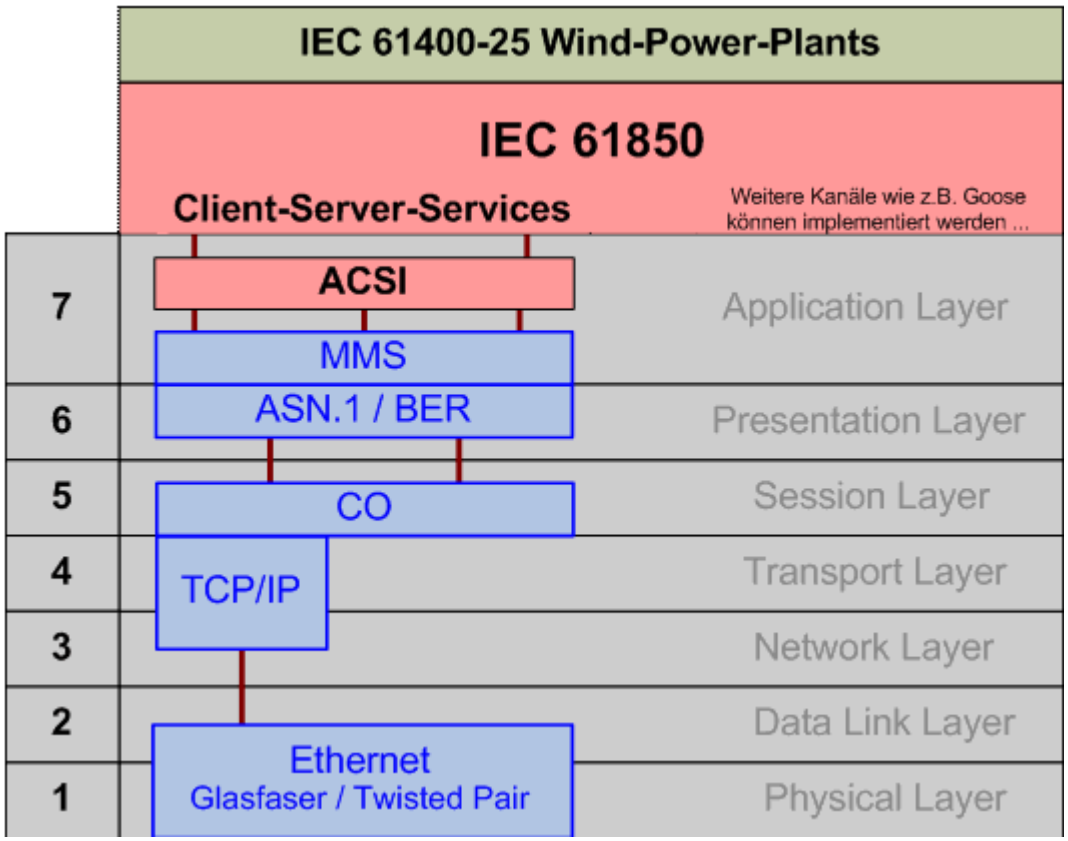
For the TwinCAT IEC 61850 Server there is a detailed step-by-step instruction available. A programming code for a fast communication test you will find under examples in the attachment of this documentation.

### Further documentation

- Standard IEC 61850-7-1:2003
- Standard IEC 61850-7-2:2003
- Standard IEC 61850-7-3:2003
- Standard IEC 61850-7-4:2003
- Standard IEC 61850-8-1:2004
- Standard IEC 61400-25:2003
- Standard ISO/FDIS 9506-1
- Standard ISO/FDIS 9506-2
- TwinCAT TCP/IP Connection Server

# 3 Protocol

The IEC 61850 is a general transmission protocol for protective and control equipment in medium and high voltage electrical substations. Physically this communication is based on the Ethernet Technology. At present the Server-Client-communication over MMS (Manufacturing Message Specification) and TCP/IP (Transmission Control Protocol/ Internet Protocol) is supported. The communication stack for the Server which is developed by Beckhoff (see graphic below) is integrated step-by-step into the TwinCAT PLC in form of libraries. In the following graphic you can also see the IEC 61400-25 which has been defined especially for the Wind turbines and therefore is a specialization of the IEC 61850.




The most important libraries which are responsible for the mapping of the IEC 61850 will be described here:

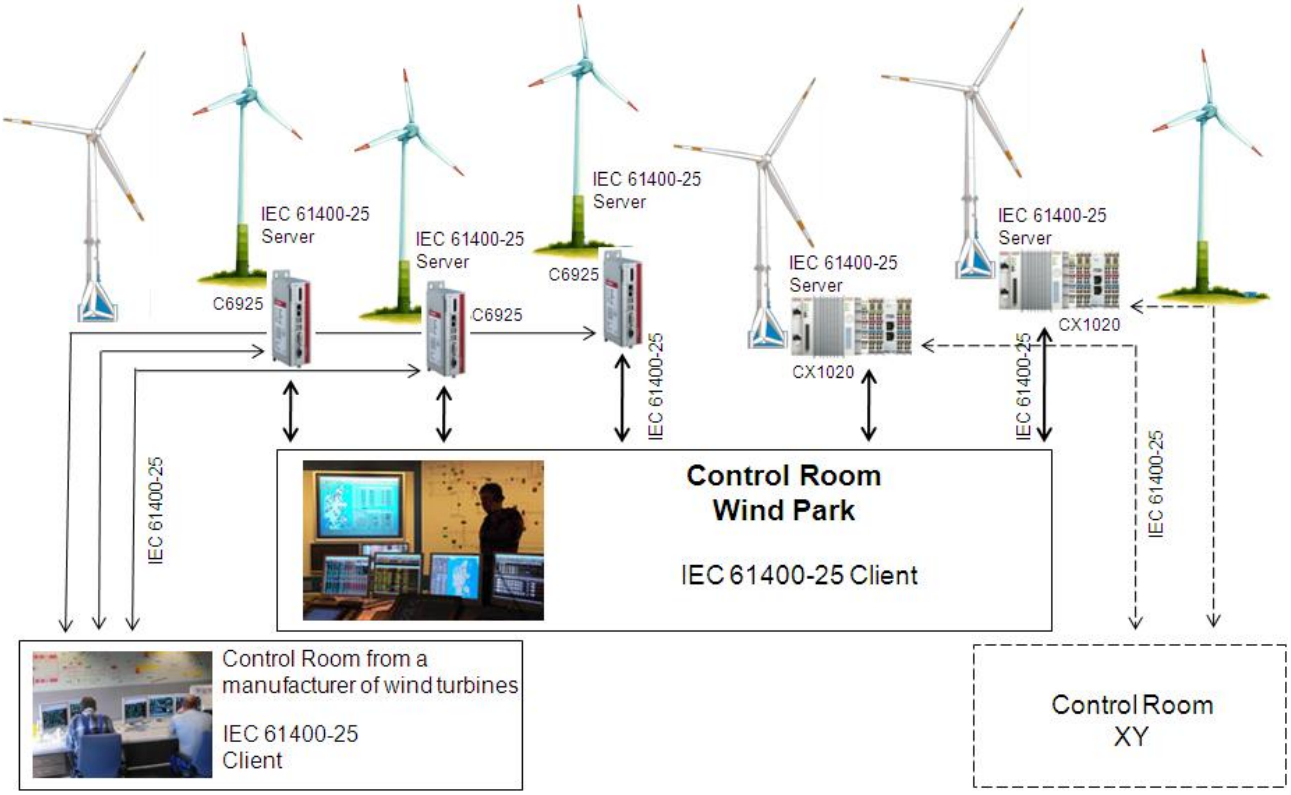
| Library | Description |
|---|---|
| **TcACSI.Lib** | This library is responsible for a clean interface to the subordinate layers. The therefore realized "Abstract Communication Service Interface" contains all basic-data types of the above named standards and the corresponding blocks to browse through the hierarchical structures of the IEC 61850 or IEC 61400-25. This ACSI-interface allows a universal and future-proofed connection of the norm to the subordinated protocols. |
| **TcMMS.Lib** | In the TcMMS.Lib the necessary services for the communication have been implemented. |
| **TcASN1.Lib** | In this library the user has all of the coding- and decoding function blocks for the in BER (Basic Encoding Rules for ASN.1) coded telegrams. |

To realize the TCP/IP communication the TwinCAT TCP/IP Connection Server is used underneath the known libraries. This Supplement product is responsible for the realization of one or more TCP/IP-Server/-Clients in the TwinCAT PLC.

The in 2004 released IEC 61850 is strictly object-oriented and its data model is build up hierarchically. There are 5 hierarchical levels which have been implemented into the TwinCAT PLC through the corresponding libraries of the standards. Explanations and examples how to apply such a structure are defined under Protocol Overview [▶ 11].

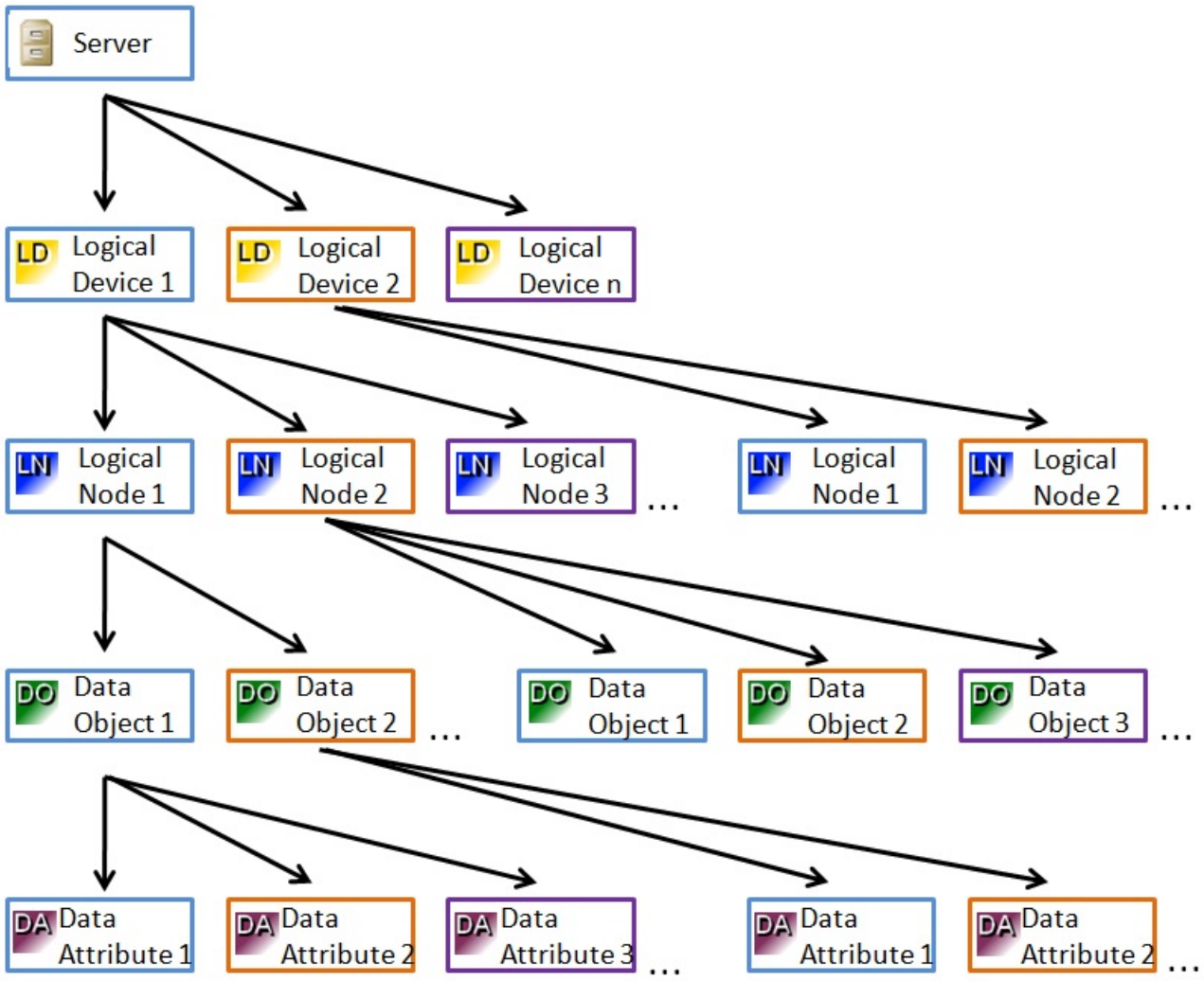
**Example of use**

In this case the IEC 61400-25 allows that wind turbines from different manufacturers can communicate with a central control station. Because of the standardization the expectations are that manufacturer-specific protocols will be avoided. The company Beckhoff supports this way by the implementation of IEC 61850 and IEC 61400-25 into the TwinCAT Automationsuite.




## 3.1 Data model

The IEC 61850 is the basic norm for different application-specializations (et.al. for the IEC 61400-25). Thats why all data- and object models for the standards that are based on this norm are equal. This data model is structured into five hierarchy levels. Thereby the corresponding standards provide the necessary Logical Nodes, Data Objects and Data Attributes. The user only needs to make sure that his Server is connected to the communication system and apply his own Logical Devices.




| | |
|---|---|
| **Server** | The first hierarchy level in the data model of the IEC 61850 or the IEC 61400-25 forms the server. The server offers a connection point for a device, whereby communication systems that are based on Ethernet can be linked. In this case TwinCAT would display such a server with the libraries that are included to the protocol on an Industrial PC or Embedded PC. |
| **Logical Device** | The Logical Devices form the second hierarchy level in the data model of the IEC 61850 and IEC 61400-25. This level separates a single physical device into several separate components, the so called Logical Devices. The advantage of this separation is that functions or objects that belong together will stand together because of their similarities. And they also can be switched into another operating mode. According to the device different Logical Nodes can be implemented out of the third hierarchy level. |
| **Logical Node** | The Logical Nodes form the third hierarchy level in the data model of the IEC 61850 or the IEC 61400-25. They represent the information of all the sub-functions that can occur in the digital station control technology and in wind energy plants. These are all the protective functions but also automation functions and functions to process measurement- and meter data. For all the normed Logical Nodes of the IEC 61850 and IEC 61400-25 there are defined identifiers that always consist of a shortcut with four letters. For example, the shortcut *XSWI* stands for *Circuit Switch*. A Logical Device can consist out of several of these nodes. |
| **Data Object** | A Logical Node can consist of several Data Objects that form the fourth hierarchy level of the IEC 61850 and IEC 61400-25 data model. Characteristic of this hierarchy level is, that data objects can be interlaced hierarchically. A data object can consist of several sub data objects before it will access on a basic- or complex data type in a Common Data Class (CDC). |

| | |
|---|---|
| DA **Data Attribute** | The lowest hierarchy level of the IEC 61850 and IEC 61400-25 are forming the data attributes. They display the detailed information or values of the data objects. Because the same scale of data attributes is defined for several data objects, the attributes of an object are combined into the so called Common Data Classes (CDC). Thereby the corresponding CDC is named for every data object. You can also say that the data objects are specializations of the CDC. |

## 3.2 Interoperability Check List

**Contents**

**1. Profile compliance**



**2. MMS conformance**



### *1. Profile compliance*

**N***in development

#### A profile support

| Profile | | Client | Server | Comments |
|---|---|---|---|---|
| A1 | Client/Server | N | Y | |
| A2 | GOOSE/GSE Management | N | N | Only GOOSE, not GSE Management |
| A3 | GSSE | N | N | |
| A4 | Time sync | N | N* | Time accuracy: 1ms (performance class T1)<br>Time resolution: approx. 0.9ms |

### T profile support

| Profile | | Client | Server | Comments |
|---|---|---|---|---|
| T1 | TCP/IP profile | N | Y | |
| T2 | OSI T profile | N | N | |
| T3 | GOOSE/GSE T profile | N | N | Only GOOSE, not GSE |
| T4 | GSSE T profile | N | N | |
| T5 | Time sync T profile | N | N | |

### MMS InitiateRequest general parameters

| InitiateRequest | Client-CR | Server-CR |
|---|---|---|
| InitiateRequest | | |
| localDetailCalling | N | Y |
| proposedMaxServOutstandingCalling | N | Y |
| proposedMaxServOutstandingCalled | N | Y |
| initRequestDetail | N | Y |
| | | |
| InitiateRequestDetail | | |
| proposedVersionNumber | N | Y |
| proposedParameterCBB | N | Y |
| servicesSupportedCalling | N | Y |

### MMS InitiateResponse general parameters

| InitiateResponse | Client-CR | Server-CR |
|---|---|---|
| InitiateResponse | | |
| localDetailCalled | N | Y |
| negotiatedMaxServOutstandingCalling | N | Y |
| negotiatedMaxServOutstandingCalled | N | Y |
| initResponseDetail | N | Y |
| | | |
| InitiateResponseDetail | | |
| negotiatedVersionNumber | N | Y |
| negotiatedParamterCBB | N | Y |
| servicesSupportedCalled | N | Y |

### Client/server services supported

| IEC 61850-7-2 model | IEC 61850-7-2 services | Implemented (Y/N) |
|---|---|---|
| Server | GetServerDirectory | Y |
| Association | Associate | Y |
| | Abort | Y |
| | Release | Y |
| Logical Device | GetLogicalDeviceDirectory | Y |
| Logical Node | GetLogicalNodeDirectory | Y |
| | GetAllDataValues | Y |

| IEC 61850-7-2 model | IEC 61850-7-2 services | Implemented (Y/N) |
|---|---|---|
| Data | GetDataValues | Y |
| | SetDataValues | Y |
| | GetDataDirectory | Y |
| | GetDataDefinition | Y |
| Data Set | GetDataSetValues | N* |
| | SetDataSetValues | N* |
| | CreateDataSet | Y |
| | DeleteDataSet | Y |
| | GetDataSetDirectory | Y |
| Substitution | GetDataValues | N |
| | SetDataValues | N |
| Setting Group Control Block | SelectActiveSG | N |
| | SelectEditSG | N |
| | SetSGValues | N |
| | ConfirmEditSGValues | N |
| | GetSGValues | N |
| | GetSGCBValues | N |
| Report Control Block | Report | N* |
| | GetBRCBValues | N* |
| | SetBRCBValues | N* |
| | GetURCBValues | N* |
| | SetURCBValues | N* |
| LOG Control Block | GETLCBValues | N |
| | SETLCBValues | N |
| | GetLogStatusValues | N |
| | QueryLogByTime | N |
| | QueryLogAfter | N |
| GOOSE | GetCoCBValues | N |
| | SetGoCBValues | N |
| GSSE | GetGsCBValues | N |
| | SetGsCBValues | N |
| Control | Select | N |
| | SelectWithValue | N |
| | Cancel | N |
| | Operate | N |
| | CommandTermination | N |
| | TimeActivatedOperate | N |
| FILE transfer | GetFile | N |
| | SetFile | N |
| | DeleteFile | N |
| | GetFileAttributeValues | N |

### *2. MMS conformance*

**MMS Initiate request general Parameters**

| MMS Service Supported CBB | Client-CR | | Server-CR | |
|---|---|---|---|---|
| | realized | value/ range | realized | value/ range |
| status | | | N | |

| | | | | |
|---|---|---|---|---|
| getNameList | | | Y | |
| identify | | | Y | |
| rename | | | N | |
| read | | | Y | |
| write | | | Y | |
| getVariableAccessAttributes | | | Y | |
| defineNamedVariable | | | N | |
| defineScatteredAccess | | | N | |
| getScatteredAccessAttributes | | | N | |
| deleteVariableAccess | | | N | |
| defineNamedVariableList | | | Y | |
| getNamedVariableListAttributes | | | Y | |
| deleteNamedVariableList | | | Y | |
| defineNamedType | | | N | |
| getNamedTypeAttributes | | | N | |
| deleteNamedType | | | N | |
| input | | | N | |
| output | | | N | |
| takeControl | | | N | |
| relinquishControl | | | N | |
| defineSemaphore | | | N | |
| deleteSemaphore | | | N | |
| reportPoolSemaphoreStatus | | | N | |
| reportSemaphoreStatus | | | N | |
| initiateDownloadSequence | | | N | |
| downloadSegment | | | N | |
| terminateDownloadSequence | | | N | |
| initiateUploadSequence | | | N | |
| uploadSegment | | | N | |
| terminateUploadSequence | | | N | |
| requestDomainDownload | | | N | |
| requestDomainUpload | | | N | |
| loadDomainContent | | | N | |
| storeDomainContent | | | N | |
| deleteDomain | | | N | |
| getDomainAttributes | | | N | |
| createProgramInvocation | | | N | |
| deleteProgramInvocation | | | N | |
| start | | | N | |
| stop | | | N | |
| resume | | | N | |
| reset | | | N | |
| kill | | | N | |
| getProgramInvocationAttributes | | | N | |
| obtailFile | | | N | |
| devineEventCondition | | | N | |
| deleteEventCondition | | | N | |
| getEventConditionAttributes | | | N | |
| getEventConditionStatus | | | N | |

| | | | | |
|---|---|---|---|---|
| getEventConditionMonitoring | | | N | |
| triggerEvent | | | N | |
| defineEventAction | | | N | |
| deleteEventAction | | | N | |
| alterEventEnrollment | | | N | |
| reportEventEnrollmentStatus | | | N | |
| getEventEnrollmentAttributes | | | N | |
| acknowledgeEventNotification | | | N | |
| getAlarmSummary | | | N | |
| getAlarmEnrollmentSummary | | | N | |
| readJournal | | | N | |
| writeJournal | | | N | |
| initializeJournal | | | N | |
| reportJournalStatus | | | N | |
| createJournal | | | N | |
| deleteJournal | | | N | |
| fileOpen | | | N | |
| fileRead | | | N | |
| fileClose | | | N | |
| fileRename | | | N | |
| fileDelete | | | N | |
| fileDirectory | | | N | |
| unsolicitedStatus | | | N | |
| informationReport | | | N* | |
| eventNotification | | | N | |
| attachToEventCondition | | | N | |
| attachToSemaphore | | | N | |
| conclude | | | Y | |
| cancel | | | N* | |
| getDataExchangeAttributes | | | N | |
| exchangeData | | | N | |
| defineAccessControlList | | | N | |
| getAccessControlListAttributes | | | N | |
| reportAccessControlledObjects | | | N | |
| deleteAccessControlList | | | N | |
| alterAccessControl | | | N | |
| reconfigureProgramInvocation | | | N | |

**MMS Parameter Conformance Building Block (CBB)**

| MMS Service Supported CBB | Client-CR | | Server-CR | |
|---|---|---|---|---|
| | realized | value/ ranged | realized | value/ ranged |
| STR1 | | | Y | |
| STR2 | | | Y | |
| VNAM | | | Y | |
| VALT | | | Y | |
| VADR | | | N | |
| VSCA | | | N | |
| TPY | | | Y | |
| VLIS | | | Y | |

| | | | | |
|---|---|---|---|---|
| REAL | | | N | |
| CEI | | | N | |

**Alternate AccessSelection Conformance Statement**

| AlternateAccessSelection | Client-CR | | Server-CR | |
|---|---|---|---|---|
| | realized | value/ ranged | realized | value/ ranged |
| accessSelection | | | N | |
| component | | | N | |
| index | | | N | |
| indexRange | | | N | |
| allElements | | | N | |
| alternateAccess | | | Y | |
| selectAccess | | | N | |
| component | | | N | |
| index | | | N | |
| indexRange | | | N | |
| allElements | | | N | |

**Variable Access Conformance Statement**

| VariableAccessSpecification | Client-CR | | Server-CR | |
|---|---|---|---|---|
| | realized | value/ range | realized | value/ range |
| listOfVariable | | | Y | |
| variableSpecification | | | N* | |
| alternateAccess | | | N* | |
| variableListName | | | N* | |

**Variable Conformance Statement**

| VariableSpecification | Client-CR | | Server-CR | |
|---|---|---|---|---|
| | realized | value/ range | realized | value/ range |
| name | | | Y | |
| address | | | N | |
| variableDescription | | | N | |
| scatteredAccessDescription | | | N | |
| invalidated | | | N | |

**Read Conformance Statement**

| Read | Client-CR | | Server-CR | |
|---|---|---|---|---|
| | realized | value/ range | realized | value/ range |
| Request | | | | |
| specificationWithResult | | | N | |
| variableAccessSpecification | | | Y | |
| Response | | | | |
| variableAccessSpecification | | | N | |
| listOfAccessResult | | | Y | |

**GetVariableAccessAttributes Conformance Statement**

| GetVariableAccessAttributes | Client-CR | Server-CR |
|---|---|---|

| | realized | value/ range | realized | value/ range |
|---|---|---|---|---|
| Request | | | | |
| name | | | Y | |
| address | | | N | |
| Response | | | | |
| mmsDeletable | | | N* | |
| address | | | N | |
| typeSpecification | | | Y | |

**DefineVariableAccessAttributes Conformance Statement**

| DefineVariableAccessAttributes | Client-CR | | Server-CR | |
|---|---|---|---|---|
| | realized | value/ range | realized | value/ range |
| Request | | | | |
| variableListName | | | N | |
| listOfVariable | | | N | |
| variableSpecification | | | N | |
| alternateAccess | | | N | |
| Response | | | N | |

**GetNamedVariableList Conformance Statement**

| GetNamedVariableListAttributes | Client-CR | | Server-CR | |
|---|---|---|---|---|
| | realized | value/ range | realized | value/ range |
| Request | | | | |
| objectName | | | Y | |
| Response | | | | |
| mmsDeletable | | | N* | |
| listOfVariable | | | Y | |
| variableSpecification | | | Y | |
| alternateAccess | | | N | |

**DeleteNamedVariableList Conformance Statement**

| DeleteNamedVariableList | Client-CR | | Server-CR | |
|---|---|---|---|---|
| | realized | value/ range | realized | value/ range |
| Request | | | | |
| Scope | | | N | |
| listOfVariableListName | | | Y | |
| domainName | | | Y | |
| Response | | | | |
| numberMatched | | | Y | |
| numberDeleted | | | Y | |
| DeleteNamedVariableList-Error | | | Y | |

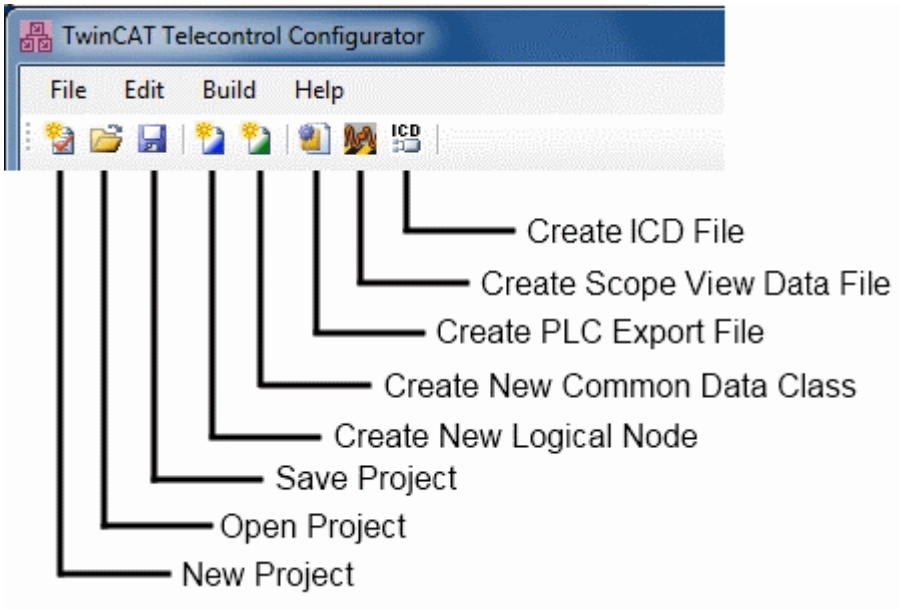
**GOOSE Conformance Statement**

| GOOSE | Subscriber | Publisher | Value/Comment |
|---|---|---|---|
| GOOSE Services | N | N | |
| SendGOOSEMessage | N | N | |
| GetGoReference | N | N | |
| GetGOOSEElementNumber | N | N | |

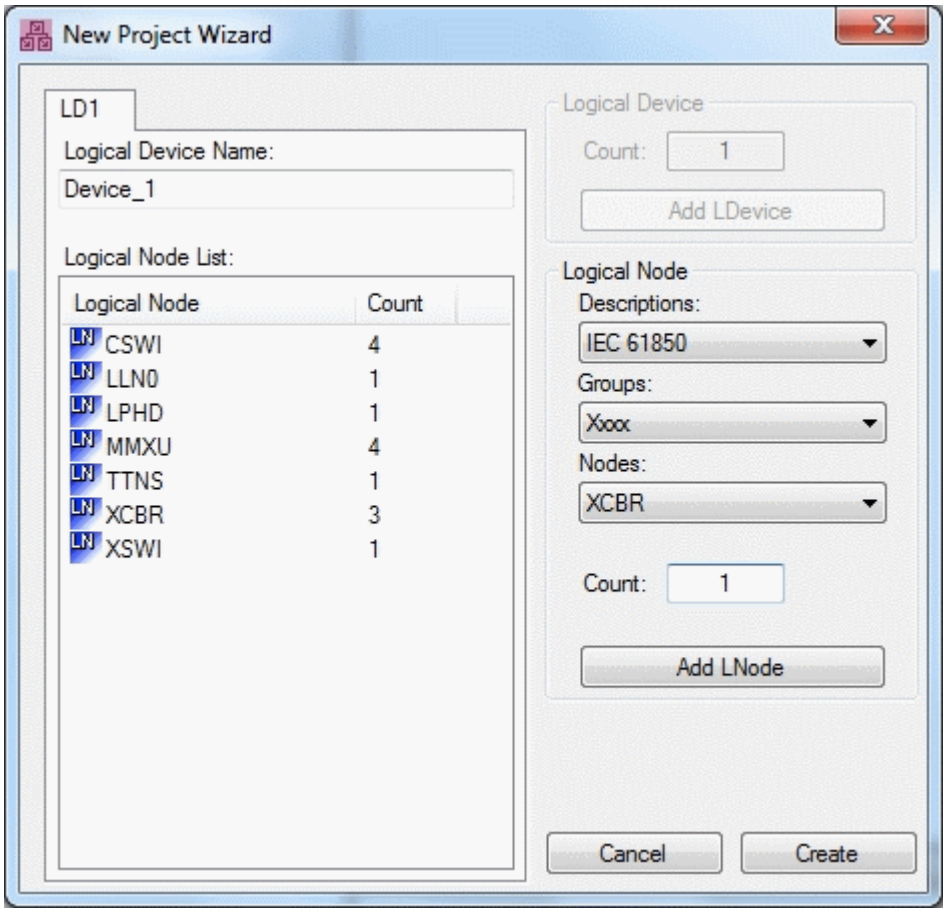
| GOOSE | Subscriber | Publisher | Value/Comment |
|---|---|---|---|
| GetGoCBValue | N | N | |
| SetGoCBValue | N | N | |
| GSENotSupported | N | N | |
| GOOSE Control Block | N | N | ReadOnly |

# 4 Configurator

The TwinCAT Telecontrol Configurator enables a perfect separation between the configuration of the standardized data model and the programming in the PLC. The PLC code which is generated after the configuration can easily be integrated and used as an export-file into the PLC Control.



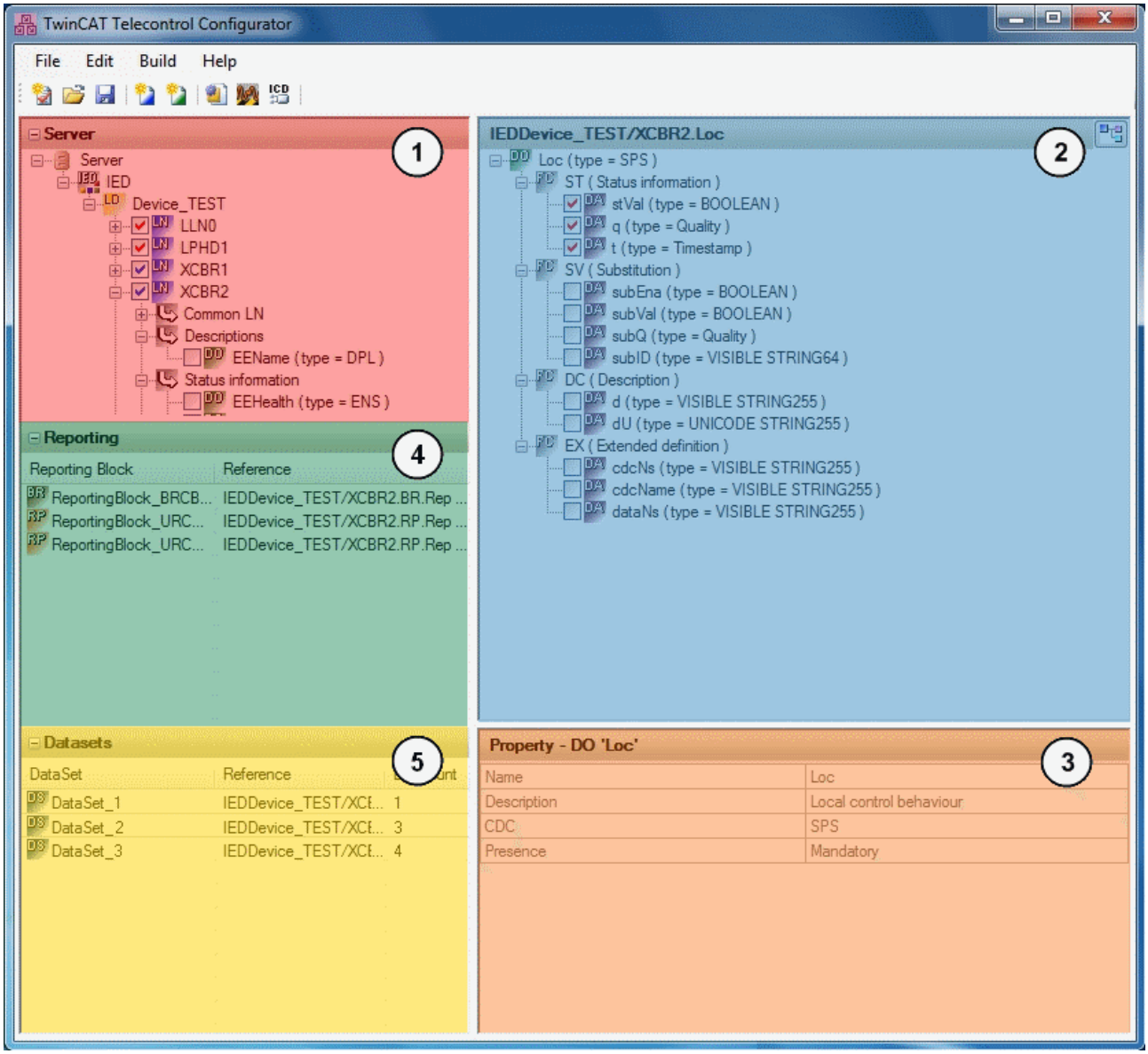

By pressing the *New Project* button in the toolbar of the configurator a Wizard-Window opens as it is shown in the next picture. This wizard helps to create a first start configuration. You can enter a Logical Device name and to select the Logical Nodes out of the different standards. The mandatory nodes *LLN0* and *LPHD* do not have to be selected especially because they will enter automatically. Another advantage is that you can enter the number of instances for a Logical Node. This will reduce the time of configuration.



Next to the standard functions *Open* and *Save* the TwinCAT Telecontrol Configurator offers the possibility to define own, private Logical Nodes and Common Data Classes. Therefore, the user needs to use the given data types of the IEC61850 to maintain conformance. To see how the creation of custom specific extensions will work, read here: Private Logical Nodes [▶ 35] and Private Common Data Classes [▶ 37].

The most important functionality of this configurator is, that you can build a PLC export data file out of the created configuration. Thereby only the data types and objects that are really needed in the configuration will be exported. Resources will be saved, which makes this Supplement product much more scalable. Details of the content of the Export data file and how to include these into a TwinCAT PLC project will be explained here [▶ 38].

You can find all the toolbar functionalities also in the menu bar of the configurator. Under *File* you can also find the function *Save As...*. Furthermore, the surface of the configurator is divided into five important parts.

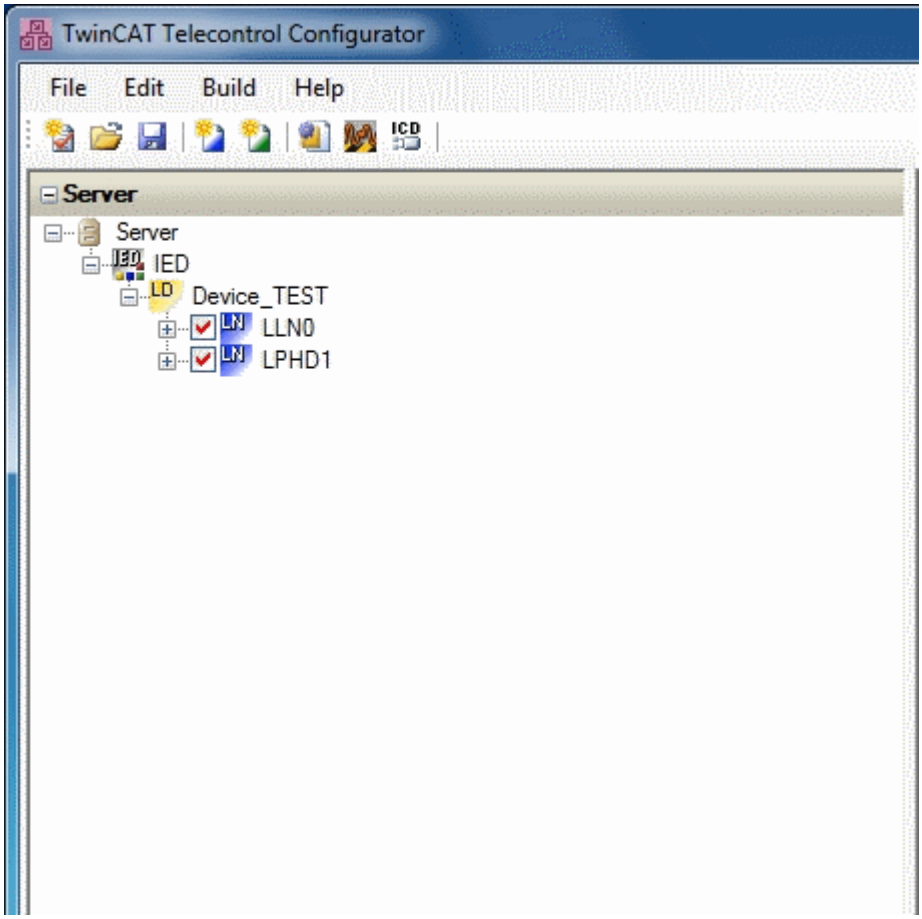



| | |
|---|---|
| [▶ 23]<br>1 | **Server Configuration:**<br><br>The Server overview contains all of the configurated Logical Devices and also offers the Logical Nodes: Server Configuration [▶ 23] |

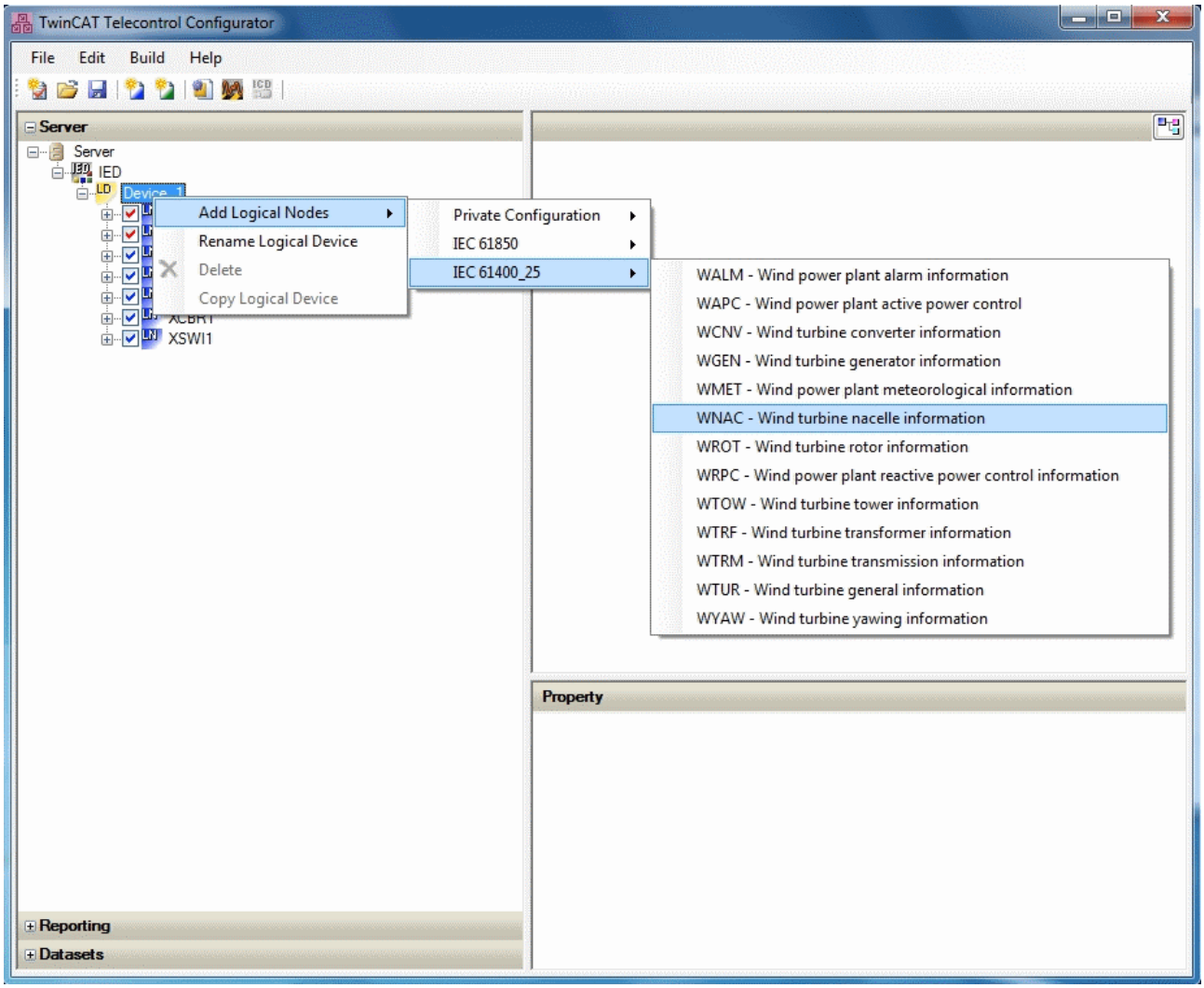
| | |
|---|---|
| [▶ 24] 2 | **Data Object Configuration:**<br><br>This overview contains all data attributes of the data object which have been selected in the server configuration: Data Object Configuration [▶ 24] |
| [▶ 29] 3 | **Object properties:**<br><br>In this window you can see the corresponding properties of the selected objects: Object properties [▶ 29] |
| [▶ 31] 4 | **Reporting:**<br><br>In this window all of the generated reporting blocks will be listed: Reporting [▶ 31] |
| [▶ 33] 5 | **DataSets:**<br><br>In this windows all of the generated DataSets will be listed: DataSets [▶ 33] |

## 4.1 Server Configuration

In the server overview of the TwinCAT Telecontrol Configurator you can see all the configurated Logical Devices and the corresponding Logical Nodes. In this window you have the possibility to add Logical Nodes to the existing configuration and to activate the optional Data Objects of a node.



If you open a new configuration the mandatory Logical Nodes LLN0 and LPHD will automatically be applied. You can identify that these are mandatory because of the red marks in the checkboxes of the nodes. This differentiating factor applies to all objects in the configurator. If optional objects will be activated, they will get a blue mark. You also have the possibility to change the name of the Logical Device by click on the name.

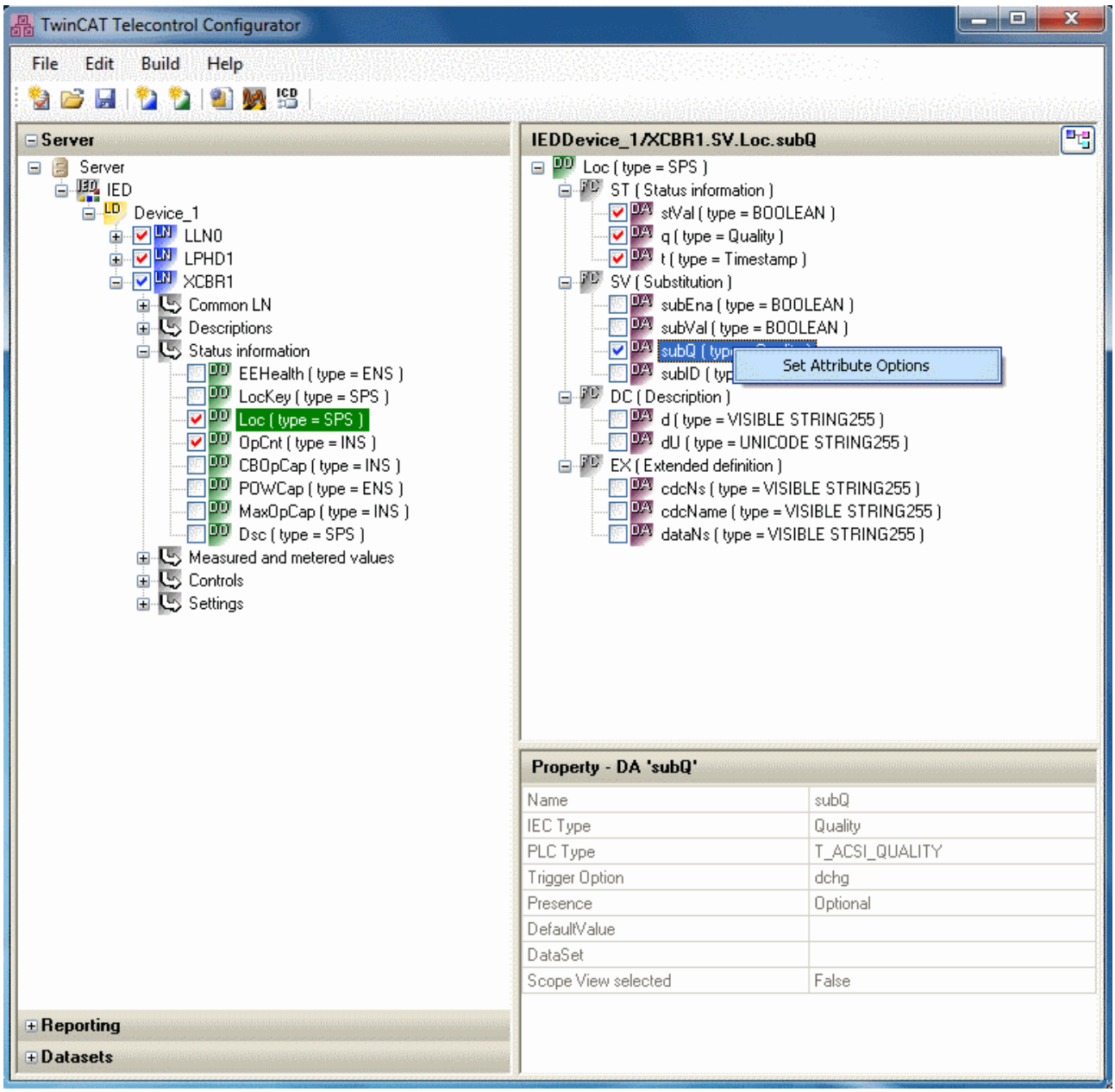



With a right-click on the respective Logical Device the context menu opens. There you have the option *Add Logical Node*. In another window the user has the possibility to choose a Logical Node out of different databases. There is always a private and an IEC 61850 database. Depending on what product license you have got, other databases will be activated. In this example the database of the Wind Turbine standard IEC 61400-25 is also available.

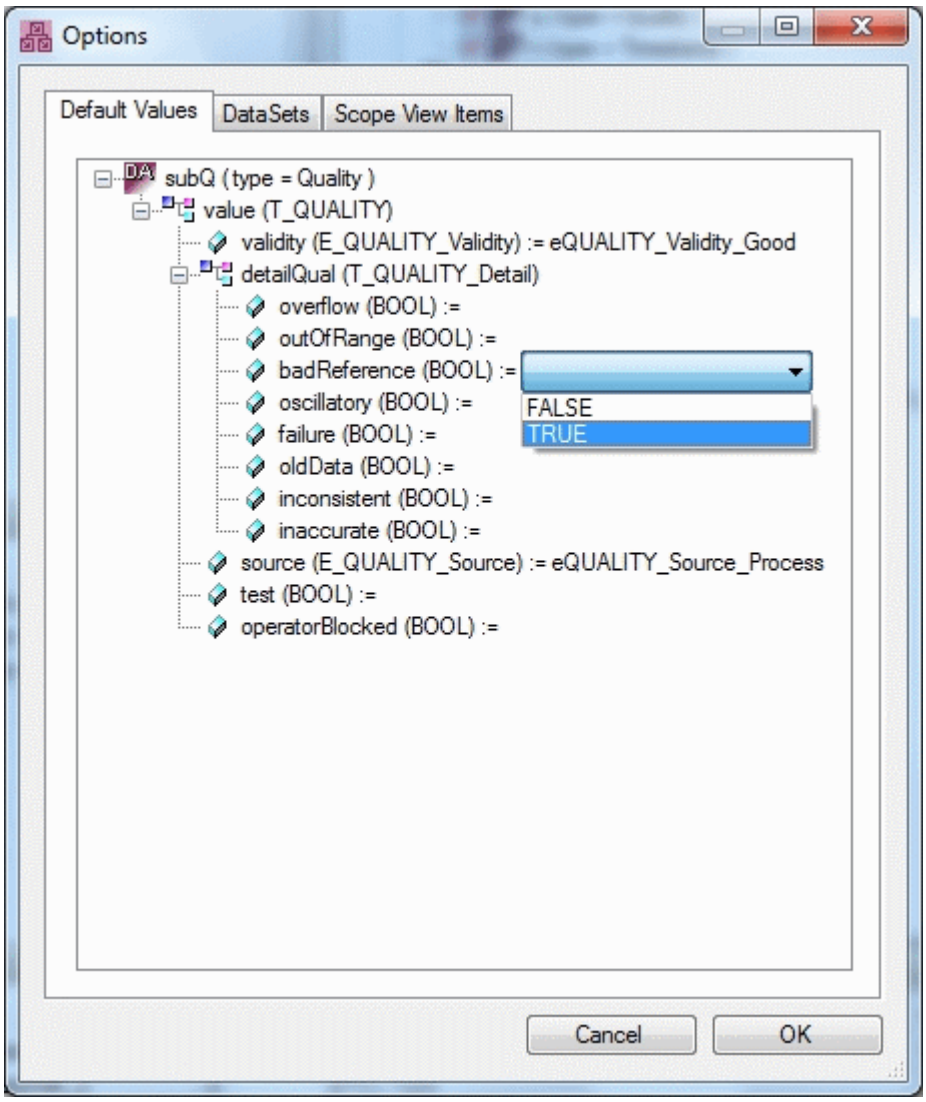
The configuration of the individual data objects will be carried on the right side of the configurator in the Data Object Configuration [▶ 24]

## 4.2 Data Object Configuration

In the data object overview on the right side of the Telecontrol Configurator the data attributes of the different data objects can be configurated.



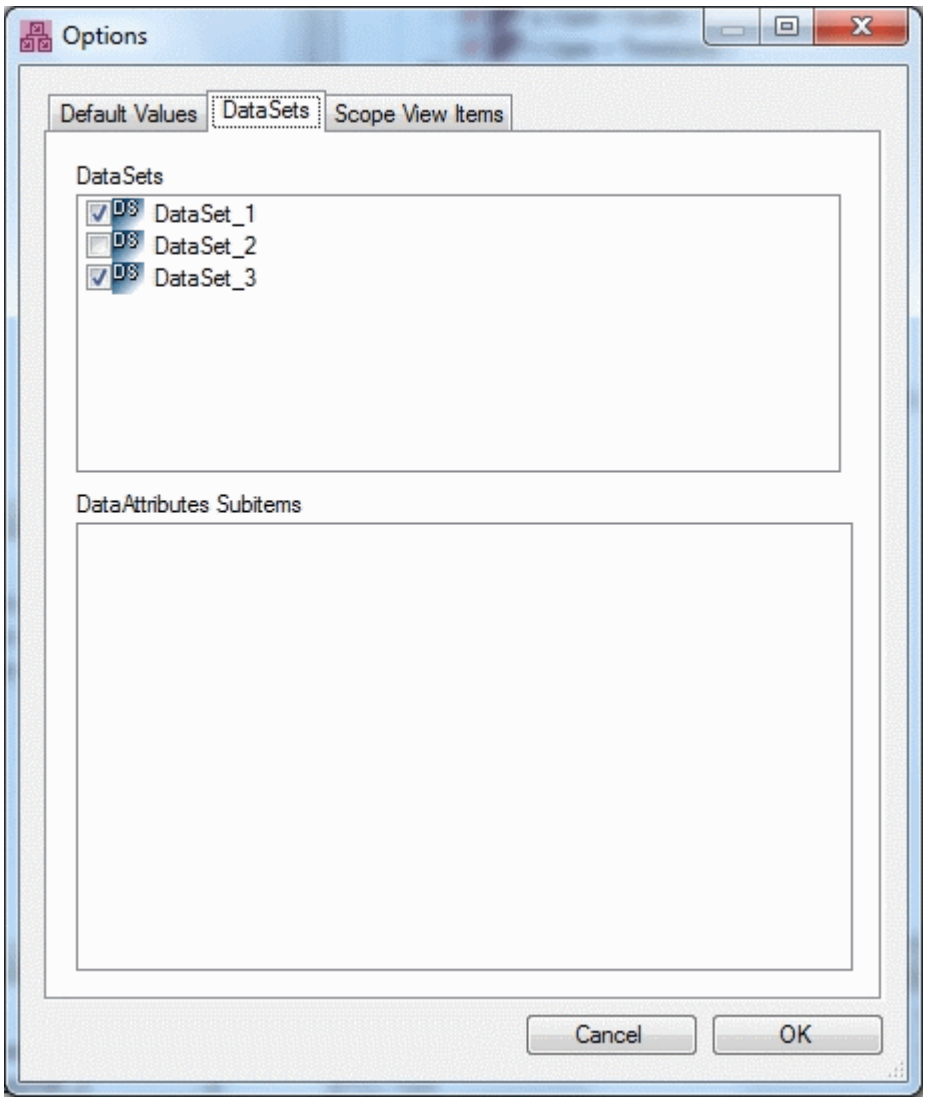

As in the server configuration [▶ 23] you can activate the optional data attributes by the corresponding checkboxes here. Furthermore, it is possible to get in the *Set default Value* dialog by double-click or with the help of the context-menu of the data attribute.

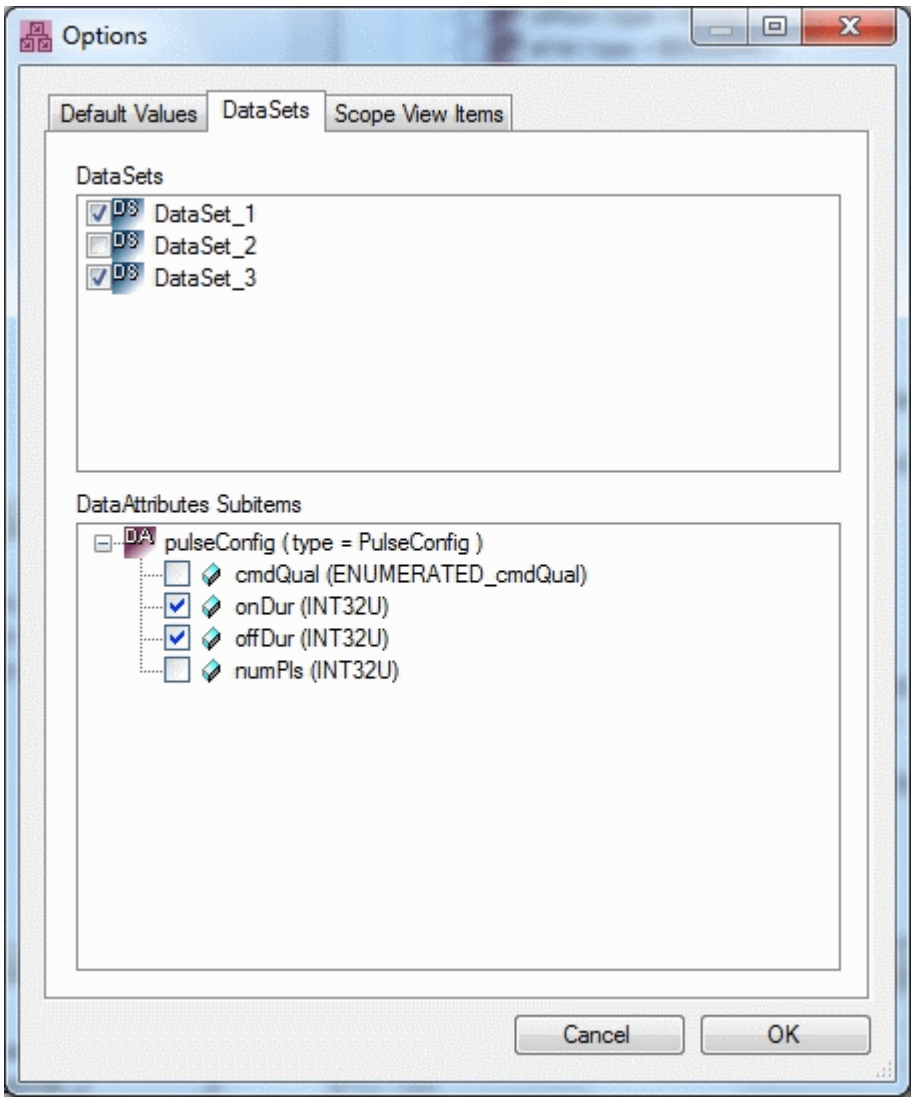



In the *Set default Value* dialog it is possible to set initial values for the data attributes. Therefore, the user gets an offer of all choices for the corresponding attribute, except of numerical values. Later, these values will be exported in the PLC code.

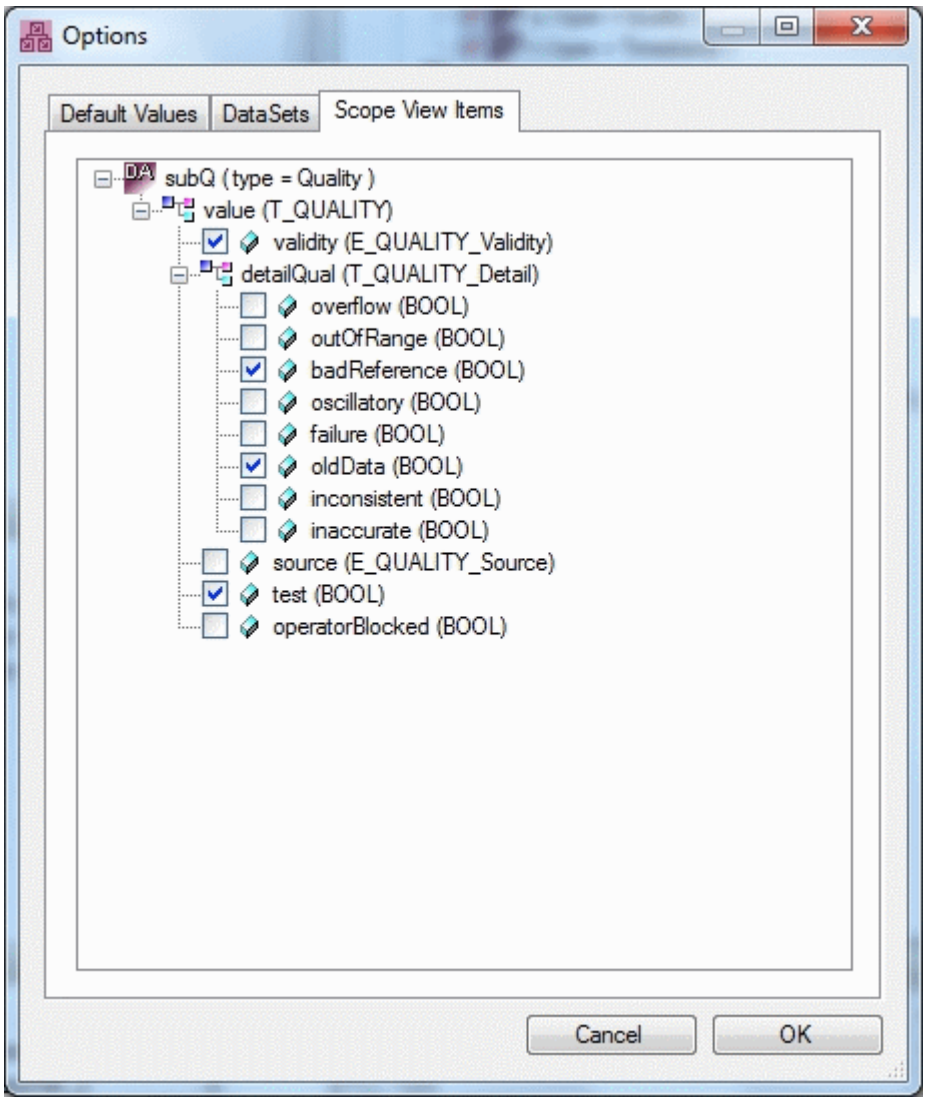
If you select an attribute as it is shown in the first picture you will see the information for the adjusted initial value in the overview of the object settings [▶ 29].

Under the tab "DataSets" it is possible to assign the selected attribute to a DataSet. If the attributes contains sub-elements these will be shown in a tree as you can see in the following pictures.


Options
Default Values
DataSets
Scope View Items
DataSets
DataSet_1
DataSet_2
DataSet_3
DataAttributes Subitems
Cancel
OK




Under the tab "Scope View Items" you can select all the elements of the selected attributes for a Scope View configuration.




## 4.3 Object properties

In the object properties you will find useful information for the selected object.

**Logical Node:**

| Property - LN 'XCBR' | |
|---|---|
| Name | XCBR |
| Description | Circuit breaker |
| Group | Switchgear |

Name: Name of the Logical Node, as it is shown in the data model tree.

Description: Offers a short description what the short cut means.

Group: Names the superior group where the Logical Node (IEC 61850-7-4) is part of.

**Data Object:**

| Property - DO 'BlkOpn' | |
|---|---|
| Name | BlkOpn |
| Description | Block opening |
| CDC | SPC |
| Presence | Mandatory |

Name: Name of the Data Objects, as it is shown in the data model tree.

Description: Offers a short description, what task this data object has.

CDC: Names the Common Data Class (IEC 61850-7-3) where the selected data object belongs to.

Presence: Says if the selected data object is a mandatory or an optional object.

**Data Attribute:**

| Property - DA 'subEna' | |
|---|---|
| Name | subEna |
| IEC Type | BOOLEAN |
| PLC Type | T_ACSI_BOOLEAN |
| Trigger Option | dchg |
| Presence | Optional |
| DefaultValue | (value:=TRUE) |
| DataSet | DataSet_1 |
| Scope View selected | True |

Name: Name of the Data Attribute, as it is shown in the data model tree.

IEC Type: Names the data type of the attribute, how it is defined in the IEC 61850.

PLC Type: Names the data type of the attribute, how it is mapped in the PLC.

Trigger Option: Names the trigger option for the respective attribute. This is important for the Data Sets and for the Reporting.

Presence: Says if the selected attribute is a mandatory or optional attribute.

Default Value: Shows the set default value, as set in the PLC when there is an export.

**Functional Constrain:**


Name: Standardized shortcut of the selected Functional Constrains.

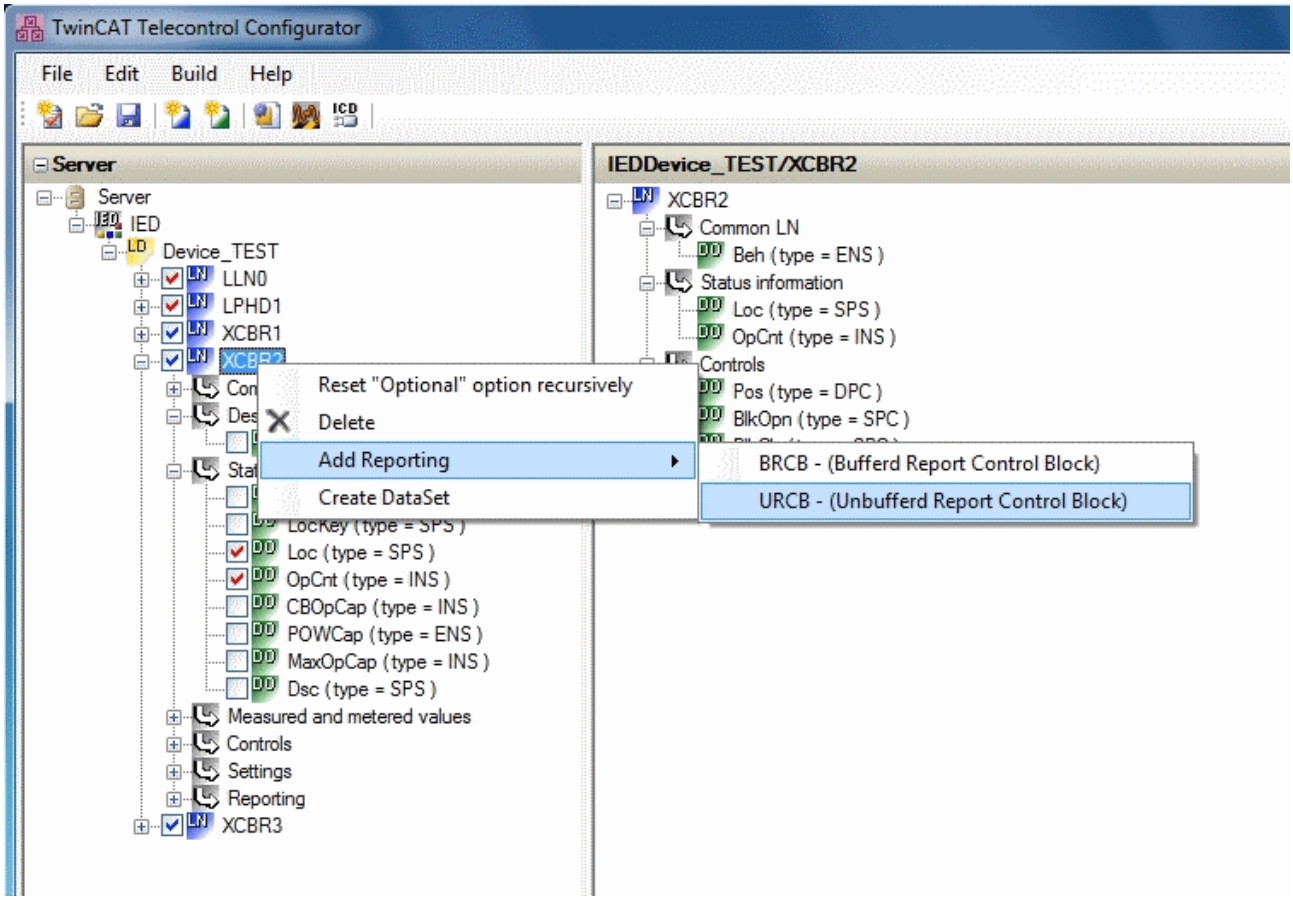
Description: Meaning of the shortened Functional Constrains.

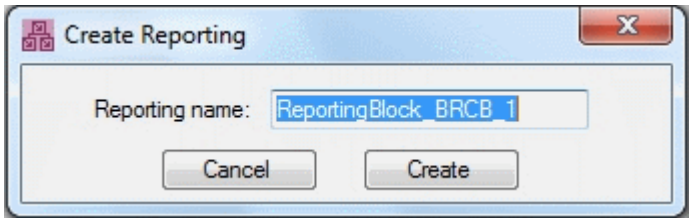
Bit Mask: Bit Mask which will be set in the PLC Export-data file.

DataSet: Lists the DataSets in which the Function Constrain is included.

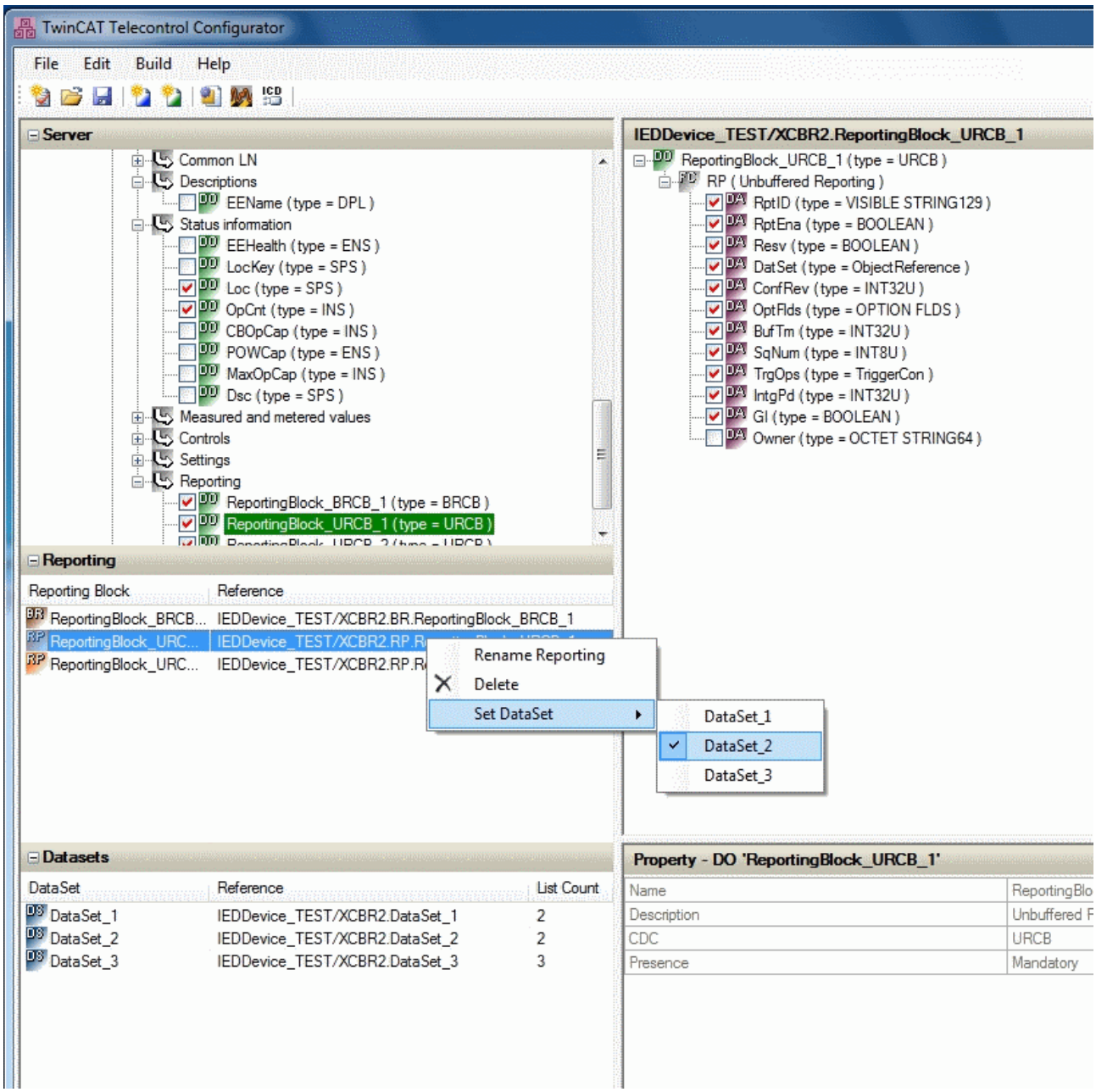
## 4.4 Reporting

Reporting blocks can be created out of the context menu as it is shown in the picture below. You have the possibility to create "bufferd" or "unbufferd" reporting blocks. Because the reporting blocks are allocated to the Logical Nodes the corresponding context menu entry only is available at the selected Logical Node. After that a dialog opens where you can enter the name of the reporting block. After you have confirmed the dialog the reporting block will be attached to the Logical Node and will be listed in the reporting list (see second picture).



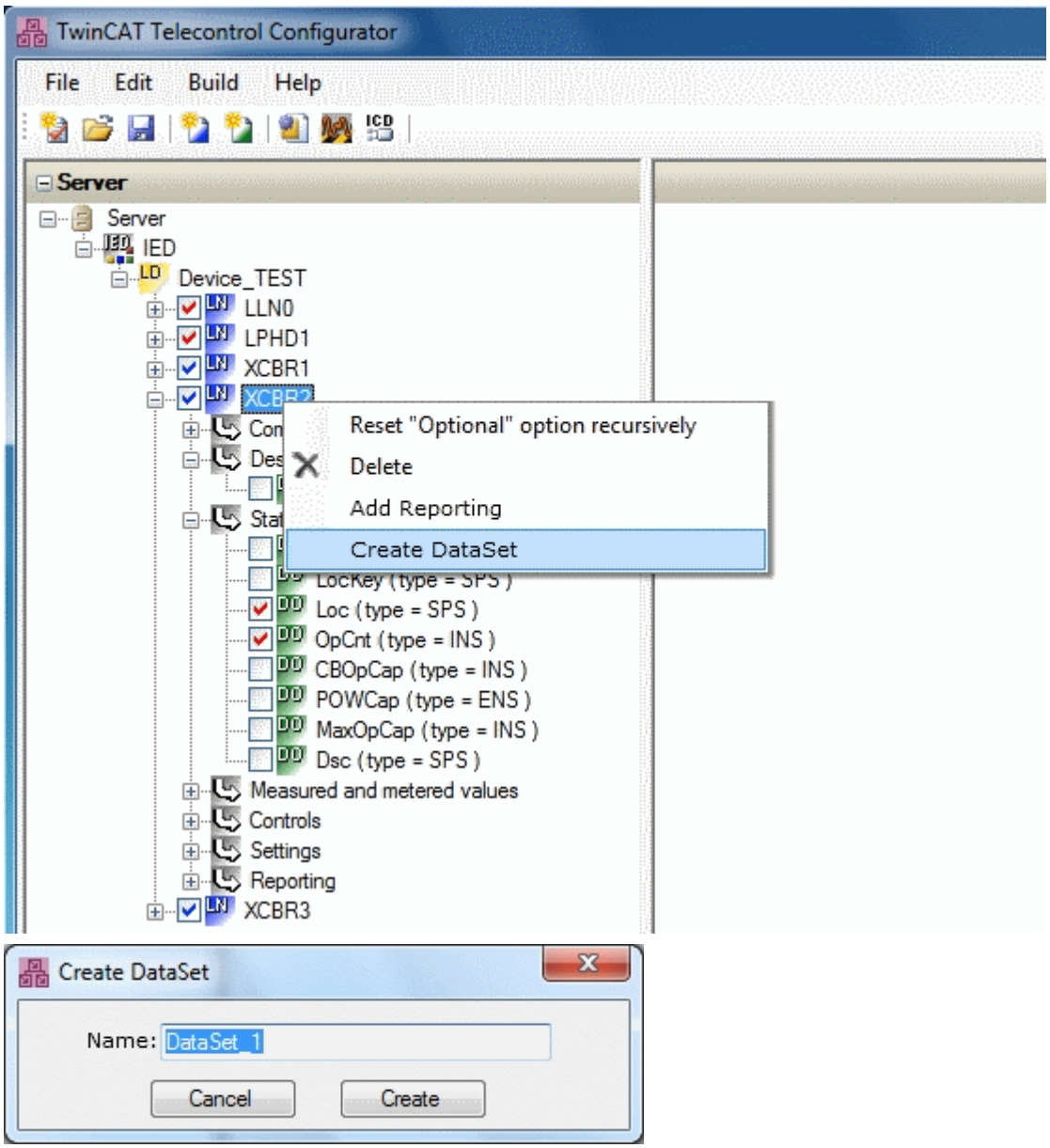

It is possible to link the reporting blocks with a DataSet in the reporting list. The reference of the DataSet will be written into the attribute "DataSet" of the reporting block. Furthermore, you have the possibility to rename or to delete the reporting block.




If you select a reporting block out of the reporting list the DataAttributes of the reporting block will be shown in the window on the right side.

## 4.5 DataSets

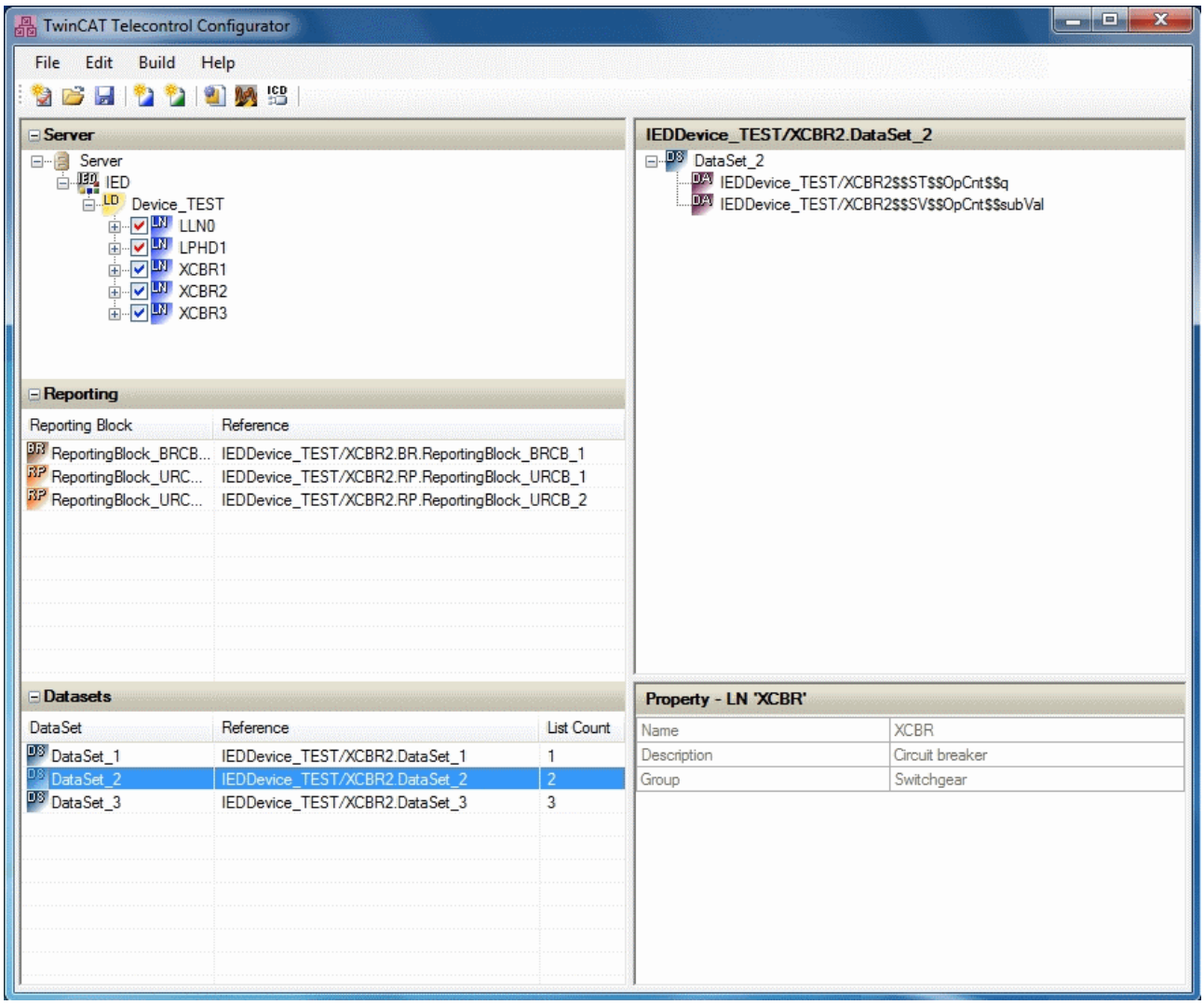
DataSets can be generated out of the context menu as you can see in the picture below. Because the DataSets will be allocated to the Logical Nodes the corresponding context menu entry is only available at the selected Logical Node. After that a dialog opens where you can enter the name of the DataSet. After you have confirmed the dialog the DataSet will be listed in the DataSetlist (see second picture).







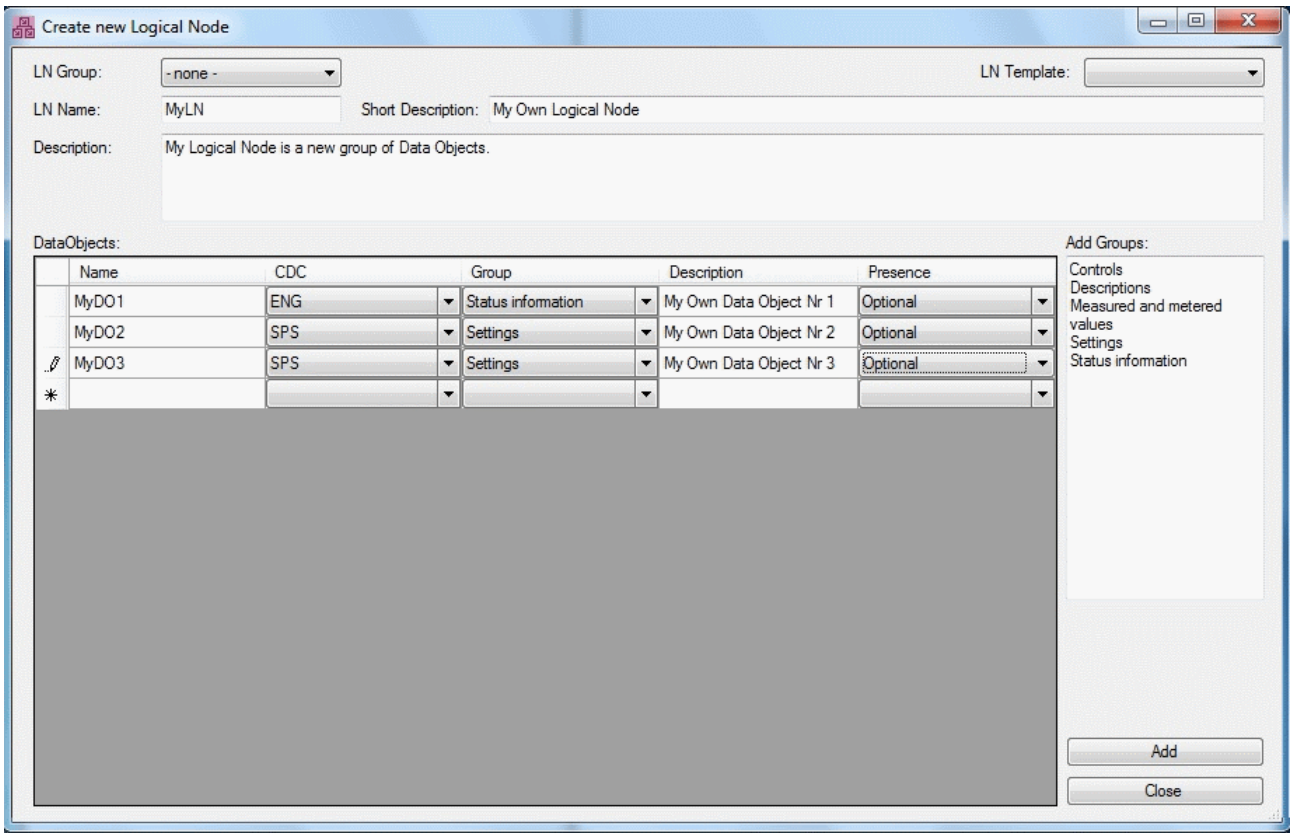
When you choose a DataSet out of the DataSetlist all the DataAttributes and Functional Constrains that are belonging to the DataSet will be shown in the window on the right side.



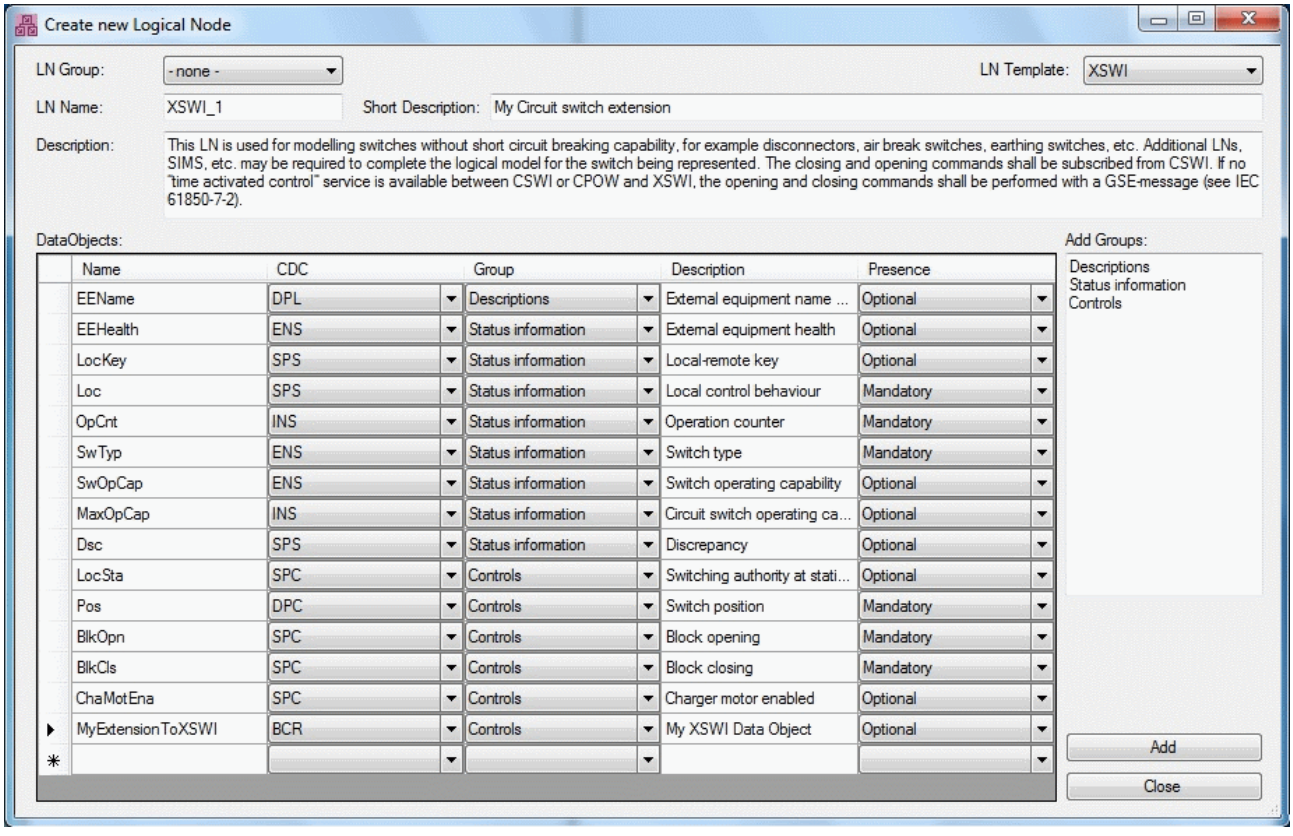

In the DataSetlist all DataSets with the corresponding reference and the amount of the attached elements will be shown. In the context menu you can change the DataSetname or to delete the DataSets.

## 4.6 Private Logical Nodes

With the function *New Logical Node* it is possible to define your own Logical Nodes or to extend existing Nodes. Thereby the new elements will be written into a private database. These elements then will be available for every new configuration.




In the shown window you can define your own Logical Nodes. Therefore, you can select a Common Data Class for each Data Object. Furthermore, you can assign the object to a group. This will lead to a clearly represented categorization in the configurator. In the column Presence you can set if the element should be activated per default or should be optional.

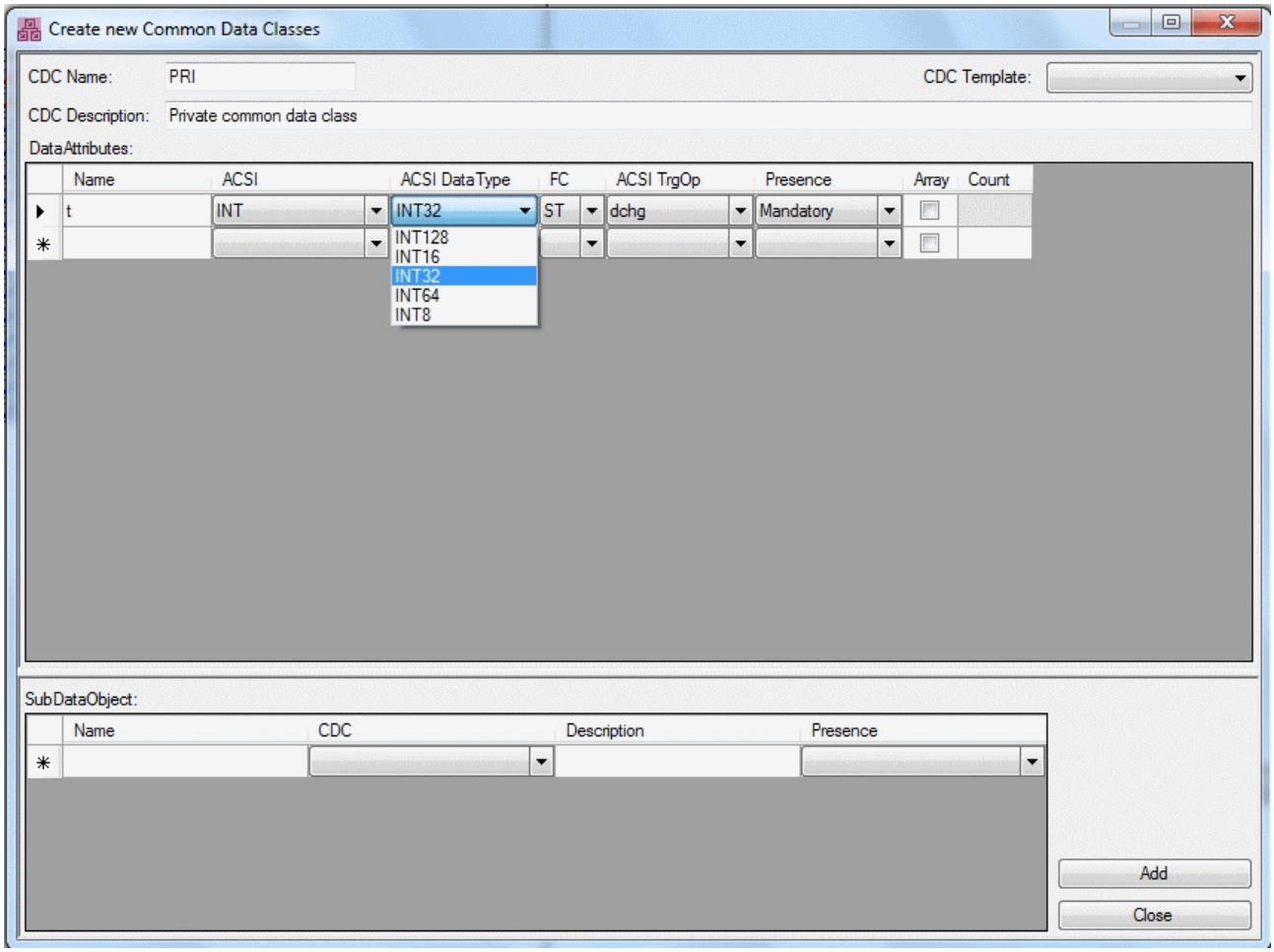



In this graphic the in the standard defined Logical Node *XSWI* (*Circuit Switch*) is extended user-specifically. Therefore, you can select a *Template* for a Logical Node at the upper right side of the window. In order to avoid a conflict, the name of the Logical Node needs to be another than the name of the template.

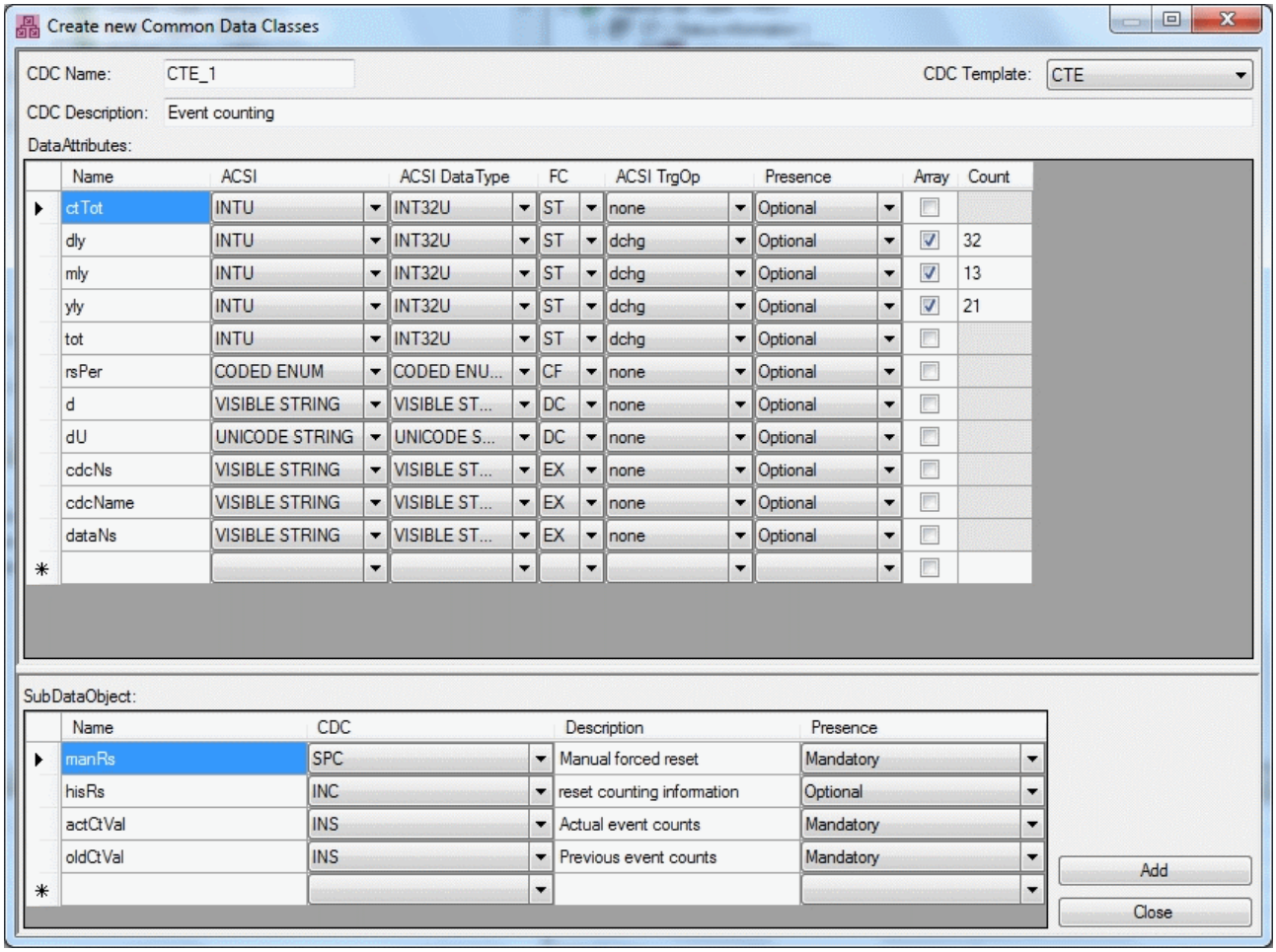
By confirmation with the *Add* Button the new Logical Node will be written into the private database and is available for the configuration.

## 4.7 Private Common Data Classes

With the function *New Common Data Class* it is possible to define own Common Data Classes or to extend existing CDCs. Thereby the new elements will be written into a private database. This elements then will be available for every new configuration.




If you apply a new Common Data Class it is possible to include further *Sub Data Objects* in addition to the attributes. Therefore, you have the possibility to select a CDC at the bottom of the window. For the attributes you can select the data types of the IEC 61850. This could be an integer for example. To describe the exact size of the integer there is a selection for the respective data type under *ACSI Data Types*. The Functional Constraints and the Trigger Option can be set as well. In the column Presence you can set if the element should be activated per default or should be optional.

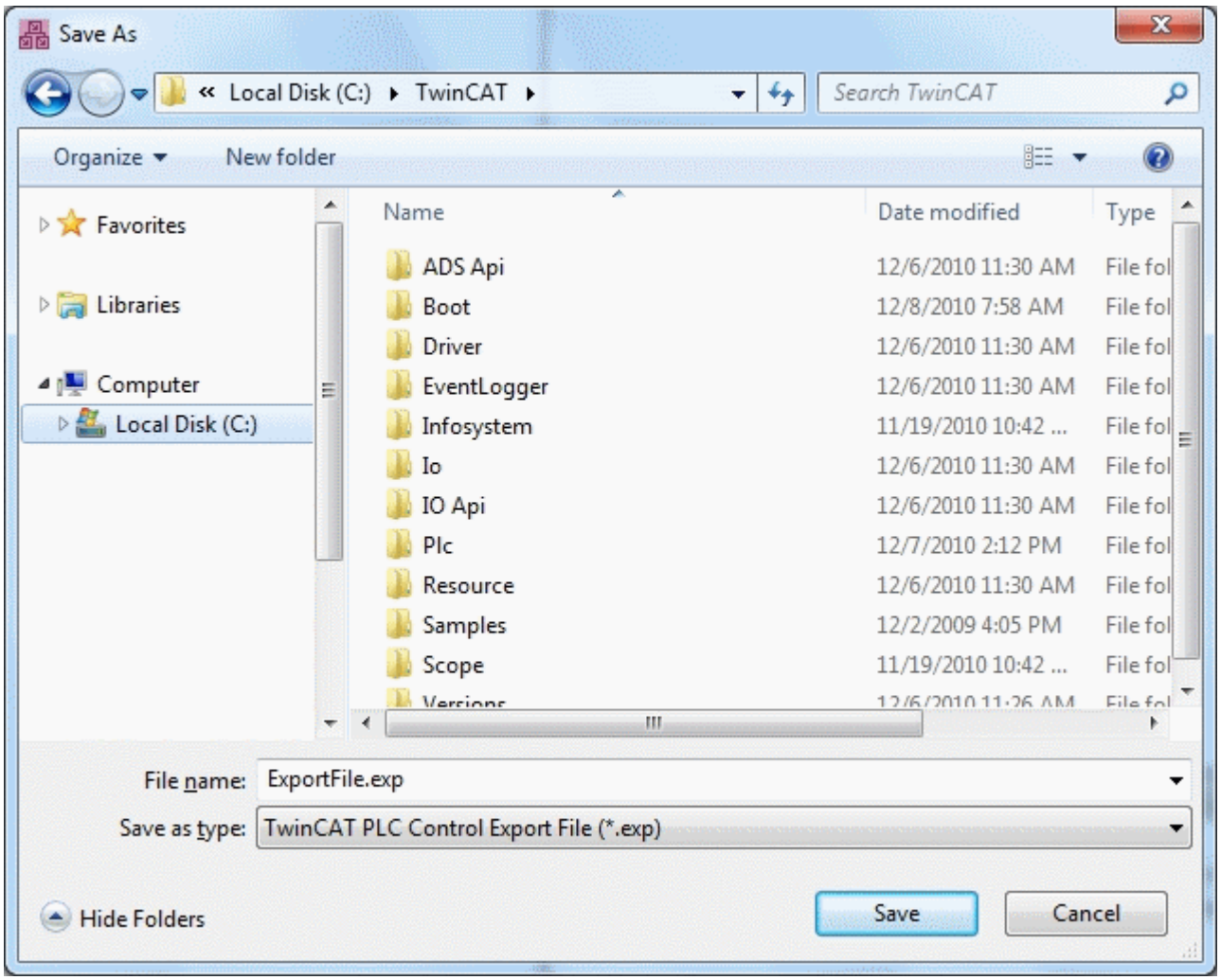



In this graphic the in the standard defined CDC *INC* (*Controllable Integer*) is extended user-specifically. Therefore, you can select a *Template* for a CDC at the upper right side of the window. In order to avoid a conflict, the name of the CDC needs to be another than the name of the template.

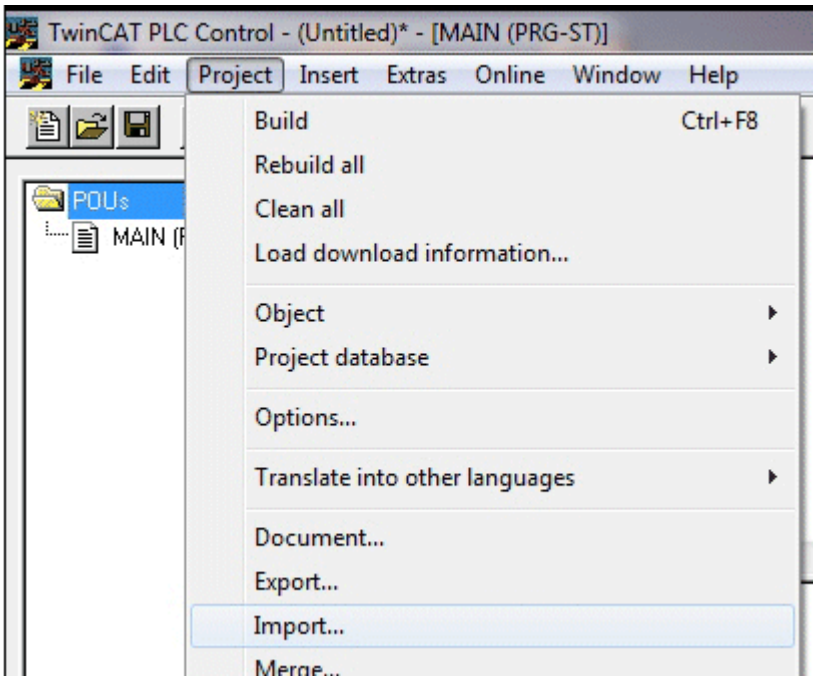
By confirmation with the *Add* Button the new Common Data Class will be written into the private database and is available for the configuration.

## 4.8 PLC Export data file

After the compilation of the configuration for the IEC 61850 Server the *Create PLC Export File* in the toolbar can be activated to generate the export data file. The later import into the TwinCAT PLC Control makes the data model of IEC 61850 / IEC 61400-25 useable in the PLC.

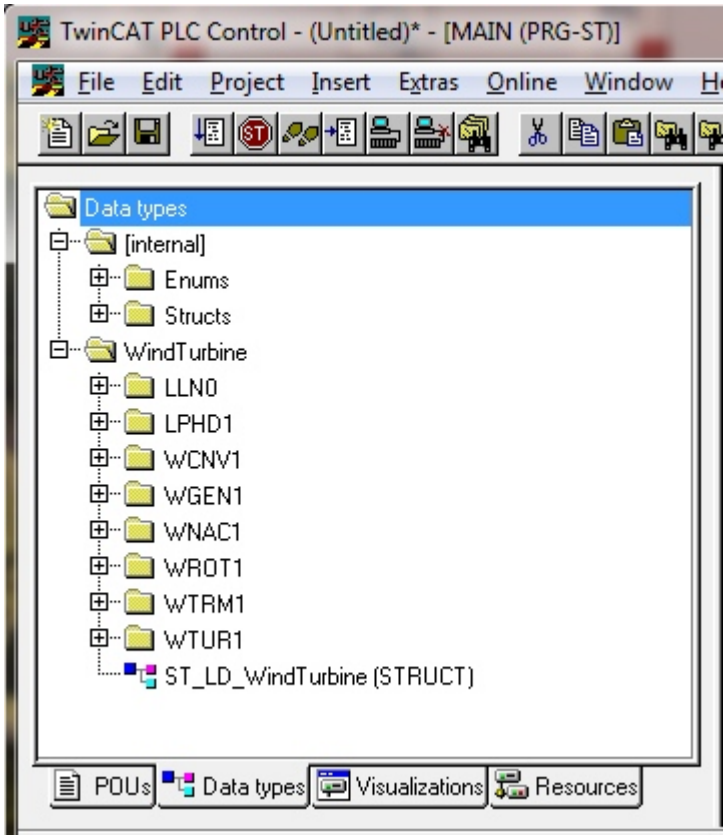



In the *Save as...* dialog the memory location can be selected. You can read the export data file in the TwinCAT PLC Control with the Import function. Therefore, you must select *Project* and then *Import* in the menu bar... as it is shown in the next picture.

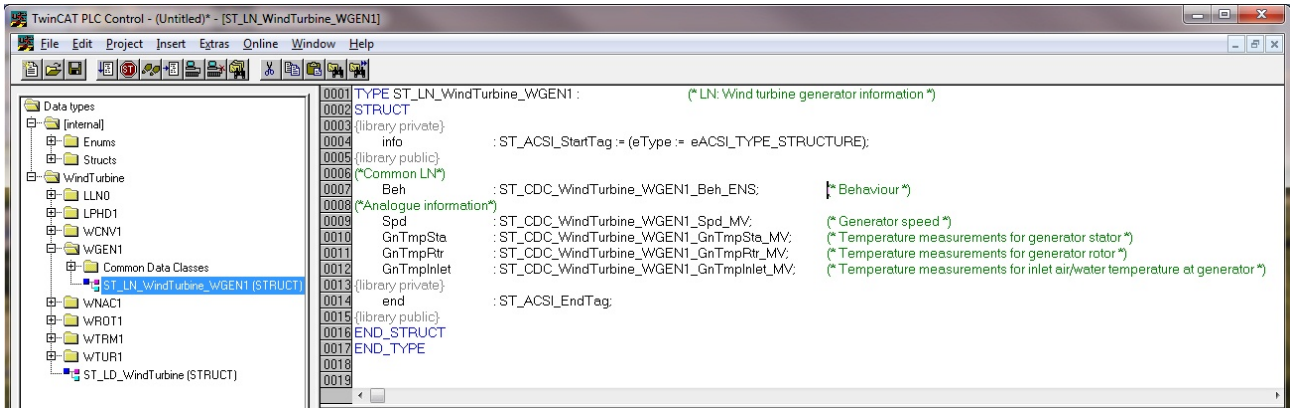



Because of the import two folders will be created in the *Data Types* tab. On the one hand *[internal]* and on the other hand a folder which has the name of the Logical Device that we created in the configurator. As you can see in the next picture the folder has the name *WindTurbine.* This folder contains the actual

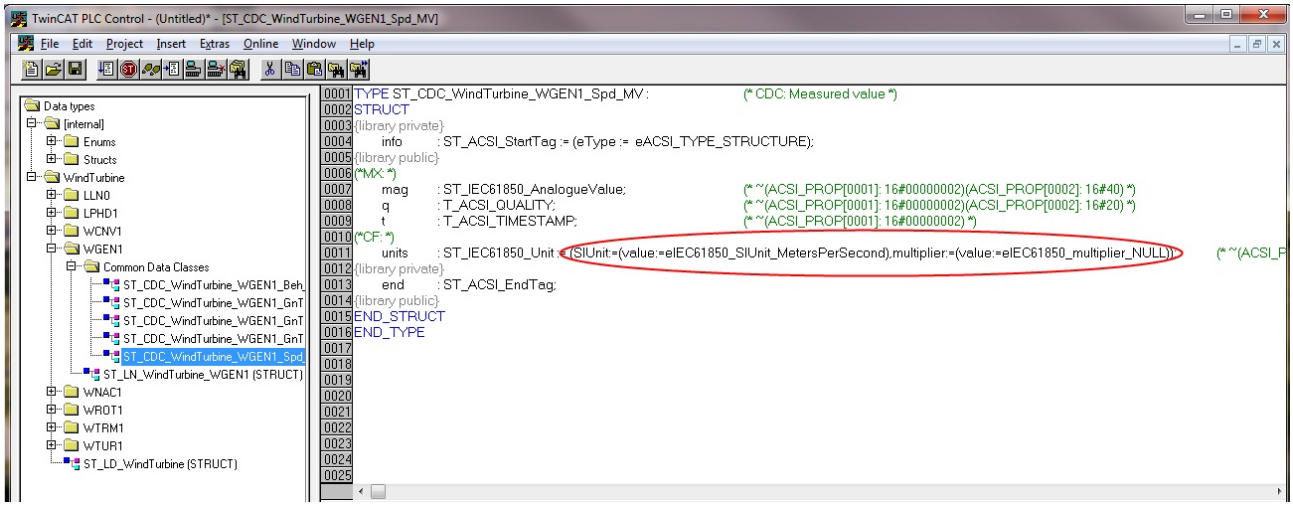
configuration with every Logical Nodes and the subordinated hierarchical levels. In the folder *[internal]* you will find some structured data types which will be used in the configuration. Normally this folder is of little importance for the user.



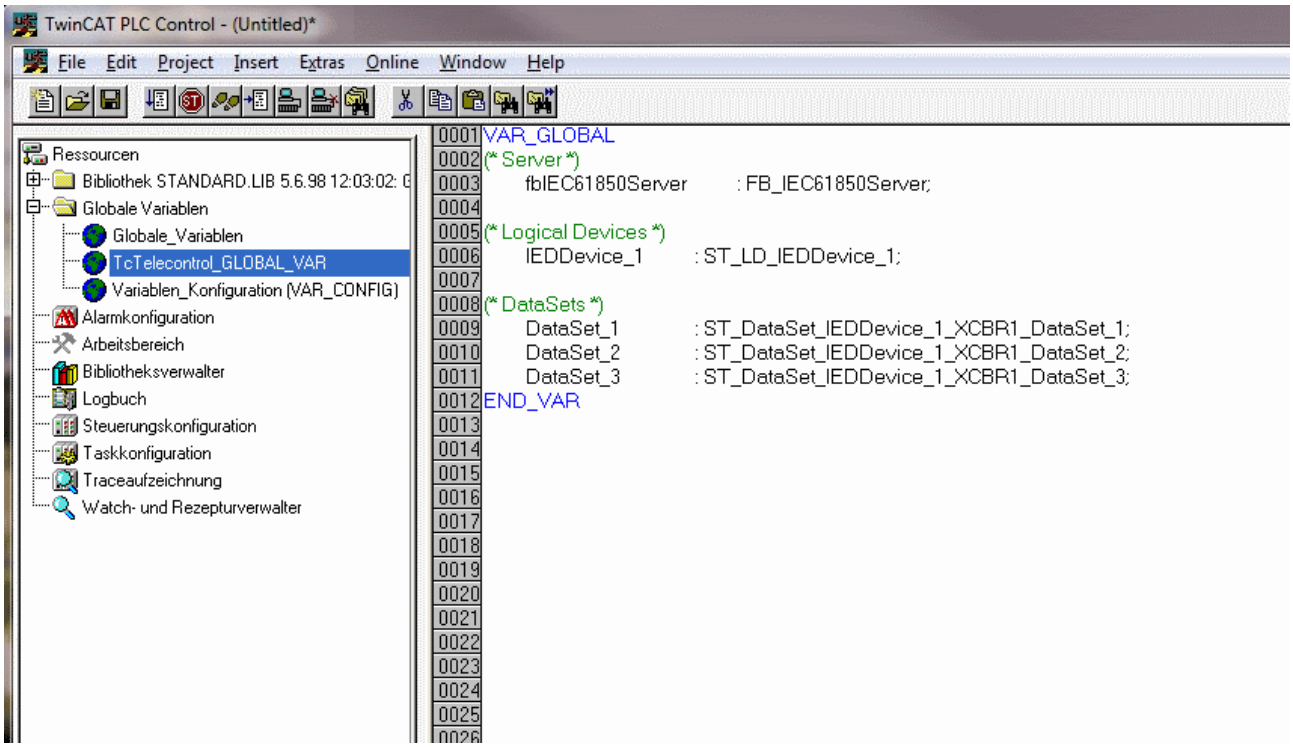

In the folder WindTurbine you can see the structure *ST_LD_WindTurbine.* This structure must be instantiated in the PLC project and needs to be committed to the function block FB_IEC6850Server. You will find detailed information about this point in the examples of this documentation.




If you open each folder of the Logical Nodes you can see the containing Data Objects in another structure. The variables *info* and *end* are for the internal communication. These are only relevant for the user if he wants to do a manual copy of the created structures. They need to exist in every structure. The folder *Common Data Classes* contains the attributes of the different objects.



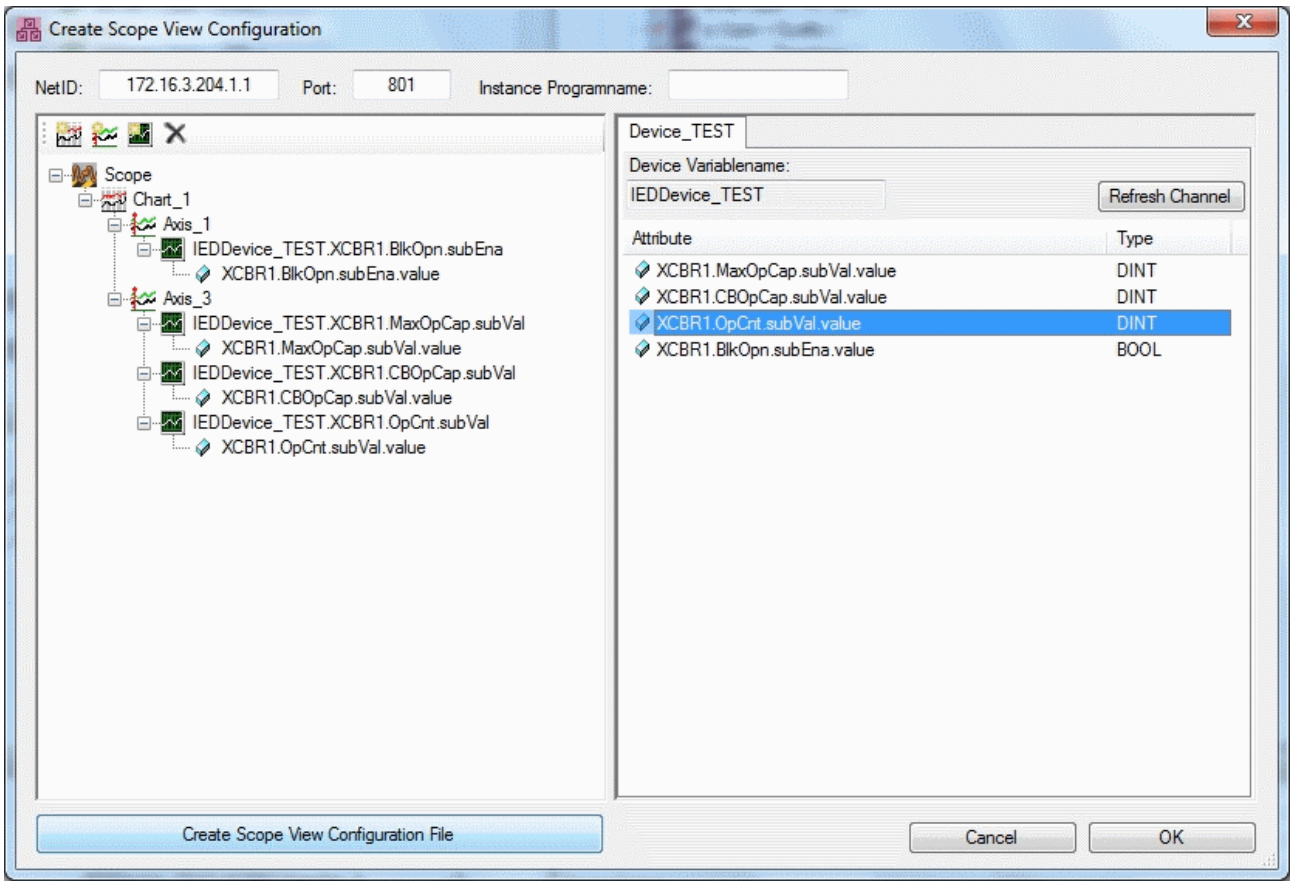

Exemplary you can see here the attributes for the Data Object *Spd* (*Wind Turbine Generator Speed*). Because of the comments the controller gets information as for example about Functional Constrain or Trigger Option. But also the *Default Values* which has been set in the configurator will be visible. So, it is possible to detect that *Meters per second* is set for the attribute*units.*




Global variables will be generated when you import the Export data file . The object "TcTelecontrol_GLOBAL_VAR" will be generated and contains an instance of the FB_IEC61850Server - on instance per Logical Device and one instance of the generated DataSets.

## 4.9 Scope View configuration data file

After the generation of the configuration for the IEC 61850 Server the *Create Scope View Data File* can be actuated in the toolbar to create a Scope View configuration data file with all the selected attributes. The allocation of the individual elements to the different channels will be carried out by the following dialog:




On the left side you can see the Scope View configuration tree with the known Scope View elements "Scope", "Chart", "Axis" and "Channel".

On the right side you can find all the selected attributes out of the Telecontrol configuration.

To connect the attributes with a channel you need to pull the attribute by drag and drop to the corresponding channel. The attribute will be added automatically to the channel and the channel name is going to be adapted to the corresponding attribute. On the other hand, the channel name also can be changed manually.

To create the Scope View configuration data file you need to apply the "Create Scope View Configuration File" button. In the dialog "save as" you can select the storage location of the data file.

# 5 PLC API

The most important PLC function blocks, functions, data types and constants of the TwinCAT IEC 61850 Server are discribed in this part of the documentation. For using this elements you have to insert the TcIEC61850Server.lib in your TwinCAT PLC Control project.

**Requirements**

| Development Environment | Target | PLC Library |
|---|---|---|
| TwinCAT v2.10.0 Build >= 1340 | PC or CX (x86, ARM) | TcIEC61850Server.Lib |

## 5.1 TcIEC61850Server

### 5.1.1 Function Blocks

#### 5.1.1.1 FB_IEC61850Server


With this function block you can realise a Server according to the IEC 61850 standard in the TwinCAT PLC. The connection parameters like IP-address and the application-specific parameters will be set over the input variables appPara and srvPara.

The Server block offers two actions:

- A_INIT; At the call of this action the server changes into the INITIALIZING condition. At this phase the internal database will be initialized (the instances of the joint data classes, attributes, logical nodes etc. will be generated). After the initialization of the internal database the Server is in the PREOP condition. At this condition furthermore application-dependent data classes can be generated in the database e.g. Datasets.
- A_OPERATE; The Server changes to the OPERATIONAL condition by calling this action. At this condition the Server can connect to the Client and exchange data actively.

**VAR_INPUT**

```
VAR_INPUT
    appPara : ST_MMS_ApplicationPara;    (* Application parameter *)
    srvPara : ST_MMS_ProtocolPara;   (* Communication parameter *)
    pLD     : DWORD := 0;          (* Pointer to logical device node *)
    cbLD    : UDINT := 0;          (* Byte size of logical device node *)
END_VAR
```

**appPara**: Application-specific configuration parameters [▶ 47] (MMS and IEC 61850 parameter).

**srvPara**: Parameters [▶ 47] that basically affect the configuration of the communication stack beneath MMS.

**pLD**: Pointer (address) on the instance of the structure with the model description of the logical device (LD). The address can be determined with the ADR operator.

**cbLD**: Size of the byte of the structure with the model description of the logical device (LD). The size of the byte can be determined with the SIZEOF operator.

### VAR_OUTPUT

```
VAR_OUTPUT
    system : ST_IED_System;
    eState : E_IED_STATE;
END_VAR
```

**system**: System [▶ 46] variable.

**eState** : Device status [▶ 45].

### Example:

You can find further examples including every source under: Examples.

```
PROGRAM MAIN
VAR
    bInit   : BOOL := TRUE;
    bOperate    : BOOL := TRUE;
    bServer     : FB_IEC61850Server := (    srvPara := (    sHost := '127.0.0.1',(* change the IP-
Address to meet your needs *)
                            debug := ( bError := TRUE, bLog := TRUE ) ) );(* Server instance *)

    fbChange : FB_ACSI_ChangePresence;(* This function block changes the mandatory/
optional presence parameter *)
 Relay   : ST_LD_Relay;(* Our logical device *)
END_VAR
```

```
IF bInit THEN
    bInit := FALSE;
    fbServer.appPara.bAccessCtrl := FALSE; (* TRUE = Use access control/
authentication, FALSE = Don't use access control/authentication *)
  (* first user *)
    fbServer.appPara.accessCtrl[1].bAuthentication := TRUE;(* TRUE = access control/
authentication for this user enabled, FALSE = Disabled *)
 fbServer.appPara.accessCtrl[1].authentMechanism := '2.2.3.1';
    fbServer.appPara.accessCtrl[1].authentPassword := 'PASSWORD';
    (* second user *)
    fbServer.appPara.accessCtrl[2].bAuthentication := TRUE;
    fbServer.appPara.accessCtrl[2].authentMechanism := '2.2.3.1';
    fbServer.appPara.accessCtrl[2].authentPassword := 'OTHER';

  (* at this point you can change the presence of some attributes... *)

    (* change presence of some attributes *)
(*      fbChange( pDO := ADR( Relay.LLN0.Loc.subEna ), cbDO := SIZEOF(  Relay.LLN0.Loc.subEna ), bPr
esence := TRUE );*)(* change to mandatory *)
(*      fbChange( pDO := ADR( Relay.LLN0.Loc.t ), cbDO := SIZEOF(  Relay.LLN0.Loc.t ), bPresence :=
FALSE );*)(* change to optional *)

    fbServer.A_INIT( pLD := ADR( Relay ), cbLD := SIZEOF( Relay ) );
END_IF

IF bOperate THEN(* switch to oparating state *)
    IF fbServer.eState = eIED_STATE_PREOP THEN
        bOperate := FALSE;

        (* at this point you can configure your data sets...*)

        fbServer.A_OPERATE( pLD := ADR( Relay ), cbLD := SIZEOF( Relay ) );
    END_IF
END_IF

fbServer( pLD   := ADR( Relay ), cbLD := SIZEOF( Relay ) );
```

### Requirements

| Development environment | Target system | PLC libraries to include |
|---|---|---|
| TwinCAT v2.10.0 Build >= 1340 | PC or CX (X86, ARM) | TcIEC61850Server.Lib |

## 5.1.2 Functions

### 5.1.2.1 F_GetVersionTcIEC61850Server


With this function it is possible to read information of the version from the PLC library.

**FUNCTION F_GetVersionTcIEC61850Server: UINT**

```
VAR_INPUT
    nVersionElement : INT;
END_VAR
```

**nVersionElement** : Version element, that should be read. Possible parameters:

- 1 : major number;
- 2 : minor number;
- 3 : revision number;

**Requirements**

| Development environment | Target system | PLC libraries to include |
|---|---|---|
| TwinCAT v2.10.0 Build >= 1340 | PC or CX (x86, ARM) | TcIEC61850Server.Lib |

## 5.1.3 Data types

### 5.1.3.1 E_IED_STATE

Operating mode of the server.

```
TYPE E_IED_STATE :
(
    eIED_STATE_IDLE := 0,
    eIED_STATE_INITIALIZING,
    eIED_STATE_PREOP,
    eIED_STATE_OPERATIONAL,
    eIED_STATE_STOPPING
);
END_TYPE
```

The server is in the IDLE state after the server-code has been downloaded onto the target device and none of the server-actions (A_INIT oder A_OPERATE) have been called.
At the call of the action A_INIT the server changes into the INITIALIZING state. In this phase the internal database will be initialized (the instances of the common data classes, attributes, logical nodes, ... will be generated). After that the server changes into the PREOP state. In this state more application-dependent data classes can be generated in the database, for example data sets. When all of the data sets have been configurated the OPERATIONAL state can be changed by calling the action A_OPERATE. In this state the server can build a connection to the client and can change data actively.

**Requirements**

| Development environment | Target system | PLC libraries to be included |
|---|---|---|
| TwinCAT v2.10.0 Build >= 1340 | PC or CX (x86, ARM) | TcIEC61850Server.Lib |

#### 5.1.3.2 ST_IED_System

```
TYPE ST_IED_System :
STRUCT
    vmd : ST_MMS_VMD;
END_STRUCT
END_TYPE
```

**vmd**: This structure [▶ 50] represents the VMD-Object (Virtual Manufacturing Device).

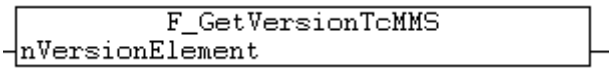
**Requirements**

| Development environment | Target system | PLC libraries to be included |
|---|---|---|
| TwinCAT v2.10.0 Build >= 1340 | PC or CX (x86, ARM) | TcIEC61850Server.Lib |

## 5.2 TcMMS

### 5.2.1 Functions

#### 5.2.1.1 F_GetVersionTcMMS




With this function versioninformation of the PLC-library can be readout.

**FUNCTION F_GetVersionTcMMS : UINT**

```
VAR_INPUT
    nVersionElement : INT;
END_VAR
```

**nVersionElement** : Versionelement, that should be readout. Possible Parameters:

- 1 : major number;
- 2 : minor number;
- 3 : revision number;

**Requirements**

| Development environment | Target system | PLC libraries to include |
|---|---|---|
| TwinCAT v2.10.0 Build >= 1340 | PC or CX (x86, ARM) | TcMMS.Lib |

### 5.2.2 Data types

#### 5.2.2.1 E_MMS_Environment

```
TYPE E_MMS_Environment:
(
    eMMS_Environment_None := 0,
    eMMS_Environment_Establishing,
    eMMS_Environment_Running,
    eMMS_Environment_Relinquishing
);
END_TYPE
```

This variable delivers the status of the MMS-environment.

**eMMS_Environment_None**: MMS-environment is in inactive status.

**eMMS_Environment_Establishing**: MMS-environment is set up right now.

**eMMS_Environment_Running**: MMS-environment is active (MMS-services can be executed)

**eMMS_Environment_Relinquishing**: MMS-environment will be break down.

### Requirements

| Development environment | Target system | PLC libraries to include |
|---|---|---|
| TwinCAT v2.10.0 Build >= 1340 | PC or CX (x86, ARM) | TcMMS.Lib |

## 5.2.2.2 ST_MMS_ApplicationPara

```
TYPE ST_MMS_ApplicationPara :
STRUCT
  (* Default mms settings *)
 maxPDUsize           : UDINT := 65000;
    localDetail       : DINT  := 65000;
    maxServOutCalling : INT := 10;
    maxServOutCalled  : INT := 10;

    services          : ST_MMS_ServiceSupportOptions;(* Default supported services *)
     parameters       : ST_MMS_ParameterSupportOptions;(* Default parameter support CBB *)
    nestingLevel      : SINT := 5;(* Default nesting level *)
  (* authentication / access control *)
    bAccessCtrl       : BOOL := FALSE;(* Access control: TRUE = enabled, FALSE = disabled *)
    accessCtrl        : ARRAY[1..MAX_MMS_ACCESS_CONTROL_USERS] OF ST_MMS_AccessControl;
END_STRUCT
END_TYPE
```

**maxPDUsize**: max. Byte length of the PDU (protocol data unit)

**localDetail**: Local detail called.

**maxServOutCalling**: max. authorised number of unconfirmed MMS-services at the calling system.

**maxServOutCalled**: max. authorised number of unconfirmed MMS-services at the called system.

**services**: Supported MMS-services [▶ 49].

**parameters**: Supported CBB-parameters [▶ 50].

**nestingLevel**: max. interlacing depth of the MMS-object data.

**bAccessCtrl**: Activates/deactivates the authentication over a password. If this variable is set at the connection establsihment the configuration settings in the array variable *"accessCtrl"* are getting proofed.

**accessCtrl**: User-access [▶ 49] and -configuration settings. Every array element equates a user, so it is possible to configurate MAX_MMS_ACCESS_CONTROL_USERS [▶ 53]

### Requirements

| Ddevelopment environment | Target system | PLC libraries to include |
|---|---|---|
| TwinCAT v2.10.0 Build >= 1340 | PC or CX (x86, ARM) | TcMMS.Lib |

## 5.2.2.3 ST_MMS_ProtocolPara

MMS communication parameters.

```
TYPE ST_MMS_ProtocolPara:
STRUCT
    sSrvNetID           : T_AmsNetID := ''; (* TwinCAT TCP/
IP Connection Server network address (default = empty string) *)
    bClient             : BOOL := FALSE;
(* TRUE => act as client, FALSE => act as server (reserved for future use) *)
```

```
    sHost               : T_IPv4Addr := '127.0.0.1'; (* If client => remote address, if server => lo
cal address. String containing an (Ipv4) internet protocol dotted address. *)
    nPort               : UDINT := 102; (* If client => remote internet protocol (IP) port, if serve
r => local listener port (default 102) *)
    nMode               : DWORD := 16#80000001;
(* Additional (optional) parameters e.g. bit 31 = CONNECT_MODE_ENABLEDBG, bit 0 = LISTEN_MODE_CLOSEA
LL *)
    eAccept     : E_SocketAcceptMode := eACCEPT_ALL; (* Connection accept flags (server only) *)
    local_AP_title      : STRING := '1.1.1.999.1';
    local_AE_qualifier      : UDINT := 12;
    local_T_selector    : STRING := '00 01';(* local-transport-selector *)
    local_S_selector    : STRING := '00 01';(* local-session-selector *)
    local_P_selector    : STRING := '00 00 00 01';(* local-presentation-selector *)

    remote_AP_title     : STRING := '1.1.1.999';
    remote_AE_qualifier : UDINT := 12;
    remote_T_selector   : STRING := '00 01';(* remote-transport-selector *)
    remote_S_selector   : STRING := '00 01';(* remote-session-selector *)
 remote_P_selector      : STRING := '00 00 00 01';(* remote-presentation-selector *)


    eTPDUsize           : E_COTP_DUsize := eCOTP_DUsize_1024;

    debug               : ST_MMS_DebugPara;
END_STRUCT
END_TYPE
```

**sSrvNetID**: The network address of TwinCAT TCP/IP Connection Server (default = empty string);

**bClient**: Reserved and not used. If set, then the device acts as client. If not set than as server (default = FALSE);

**sHost**: Local IPv4 server host address as string (default = '127.0.0.1' = loopback address);

**nPort**: Local server port address (default = 102);

**nMode**: Additional (optional) parameters (default = CONNECT_MODE_ENABLEDBG OR LISTEN_MODE_CLOSEALL):

- Bit 31: Enables/disables socket creation/release debug messages (use the constant CONNECT_MODE_ENABLEDBG to mask/set this bit);
- Bit 0: Enables/disables the automatic close of old connections/sockets on PLC start (user the constant LISTEN_MODE_CLOSEALL to mask/set this bit);
- All other bits are reserved.

**eAccept**: Reserved and not used. Server only: connection accept flags (default = eACCEPT_ALL = accept all incoming connections);

**local_AP_title**: ACSE AP-Title of this device as string (default = '1,1,1,999.1' );

**local_AE_qualifier**: ACSE AE-Qualifier of this device (default = 12);

**local_T_selector**: Local TSAP-Address as string (TSAP = Transport Service Access Point, default = '00 01');

**local_S_selector**: Local SSAP-Address as string (SSAP = Session Service Access Point, default = '00 01');

**local_P_selector**: Local PSAP-Address as string (PSAP = Presentation Service Access Point, default = '00 00 00 01');

**remote_AP_title**: ACSE AP-Title of remote device as string (default = '1,1,1,999');

**remote_AE_qualifier**: ACSE AE-Qualifier of remote device (default = 12);

**remote_T_selector**: Remote TSAP-Address as string (TSAP = Transport Service Access Point, default = '00 01');

**remote_S_selector**: Remote SSAP-Address as string (SSAP = Session Service Access Point, default = '00 01');

**remote_P_selector**: Remote PSAP-Address as string (PSAP = Presentation Service Access Point, default = '00 00 00 01');

**eTPDUSize**: Max. byte length [▶ 52] of the transport layer data segment ( default = eCOTP_DUsize_1024 = 1024 byte );

**debug**: This structure [▶ 51] configures the behavior of debug messages. The member variables are enabling/disabling logging of debug messages to the TwinCAT System Manager Logview.

### Requirements

| Development environment | Target system | PLC libraries to include |
|---|---|---|
| TwinCAT v2.10.0 Build >= 1340 | PC or CX (x86, ARM) | TcMMS.Lib |

## 5.2.2.4 ST_MMS_AccessControl

```
TYPE ST_MMS_AccessControl :
STRUCT
  (* authentication *)
    bAuthentication     : BOOL := TRUE;(* Authentication: TRUE = enabled, FALSE = disabled *)
    authentMechanism    : STRING := '2.2.3.1';(* Authentication mechanism: (joint-iso-
ccitt.2.3.1) = 52 03 01 hex *)
    authentPassword     : STRING(MAX_MMS_PASSWORD_LENGTH) := 'PASSWORD';
(* Authentication password *)

    services            : ST_MMS_ServiceSupportOptions;(* Supported services *)
    parameters          : ST_MMS_ParameterSupportOptions;(* Parameter support CBB *)
    nestingLevel        : SINT := 5;(* Nesting level *)

END_STRUCT
END_TYPE
```

**bAuthentication**: The authentication-settings are activated for this user if the parameter is set on TRUE. Over this parameter you can temporary deactivate the access for some users.

**authentMechanism**: Authentication-Mechanism (default = "2.2.3.1").

**authentPassword**: Authentication-Password [▶ 53] (default = "PASSWORD").

**services**: Supported MMS-Services [▶ 49].

**parameters**: Supported CBB-Parameter [▶ 50].

**nestingLevel**: Max. permitted interlacing-depth of the MMS-Variables (default = 5).

### Requirements

| Development environment | Target system | PLC libraries to include |
|---|---|---|
| TwinCAT v2.10.0 Build >= 1340 | PC or CX (x86, ARM) | TcMMS.Lib |

## 5.2.2.5 ST_MMS_ServiceSupportOptions

```
TYPE ST_MMS_ServiceSupportOptions:
STRUCT
    status                          : BOOL := TRUE;
    getNameList                     : BOOL := TRUE;
    identify                        : BOOL := TRUE;
    read                            : BOOL := TRUE;
    write                           : BOOL := TRUE;
    getVariableAccessAttributes     : BOOL := TRUE;
    defineNamedVariableList         : BOOL := TRUE;
    getNamedVariableListAttributes  : BOOL := TRUE;
    deleteNamedVariableList         : BOOL := TRUE;
    getDomainAttributes             : BOOL := TRUE;
    kill                            : BOOL := FALSE;
    readJournal                     : BOOL := TRUE;
    writeJournal                    : BOOL := TRUE;
    initializeJournal               : BOOL := TRUE;
    reportJournalStatus             : BOOL := TRUE;
    getCapabilityList               : BOOL := TRUE;
    FILEOPEN                        : BOOL := TRUE;
```

```
    FILEREAD                        : BOOL := TRUE;
    FILECLOSE                       : BOOL := TRUE;
    fileRename                      : BOOL := FALSE;
    fileDelete                      : BOOL := TRUE;
    fileDirectory                   : BOOL := TRUE;
    unsolicitedStatus               : BOOL := TRUE;
    informationReport               : BOOL := TRUE;
    conclude                        : BOOL := TRUE;
    cancel                          : BOOL := TRUE;
END_STRUCT
END_TYPE
```

**Requirements**

| Development environment | Target system | PLC libraries to include |
|---|---|---|
| TwinCAT v2.10.0 Build >= 1340 | PC or CX (x86, ARM) | TcMMS.Lib |

### 5.2.2.6 ST_MMS_ParameterSupportOptions

Parameters that will be negotiated between the two communication partners during the connection establishment (Association).

```
TYPE ST_MMS_ParameterSupportOptions:
STRUCT
    str1 : BOOL := TRUE;
    str2 : BOOL := TRUE;
    vnam : BOOL := TRUE;
    valt : BOOL := TRUE;
    vlis : BOOL := TRUE;
END_STRUCT
END_TYPE
```

**str1**: When this is set the device supports the Array-Type-Identifier for the type-description of the Array-Variables.

**str2**: When this is set the device supports the Structure-Type-Identifier for the type-description of the structured variable.

**vnam**: When this is set the device supports the named-access in variableSpecification.

**valt**: When this is set the device supports the so-called AlternateAccess. Indicated access to Array-Elements (INDEX or INDEX-RANGE).

**vlis**: When this is set the device supports the variableListName in variableAccessSpecification.

**Requirements**

| Development environment | Target system | PLC libraries to include |
|---|---|---|
| TwinCAT v2.10.0 Build >= 1340 | PC or CX (x86, ARM) | TcMMS.Lib |

### 5.2.2.7 ST_MMS_VMD

```
TYPE ST_MMS_VMD :
STRUCT
    vendorName      : T_MaxString := 'BECKHOFF';
    modelName       : T_MaxString := 'TwinCAT IEC 61850 Server';
    revision        : T_MaxString := '1.0.0';

    Associations    : ARRAY[1..MAX_MMS_APPLICATION_ASSOCIATIONS] OF ST_MMS_ApplicationAssociation;

    errors          : FB_MMS_ErrorFifo;

    nAllocLoad      : UDINT := 0;
END_STRUCT
END_TYPE
```

**vendorName**: VMD-Name of manufacturer.

**modelName**: VMD-Name of model.

**revision**: VMD-Version number as a String.

**Associations**: A list of the application-associations [▶ 53]. An application-association [▶ 51] describes a client-server-connection (mapping). At the setting up of a new connection an unused array-element is assigned to the new connection. The application-association-structure variable provides status information of the association.

**errors**: Error-Fifo.

**logs**: Log-Fifo.

**nAllocLoad**: Percentage filling status of the MMS-object database.

### Requirements

| Development environment | Target system | PLC libraries to include |
|---|---|---|
| TwinCAT v2.10.0 Build >= 1340 | PC or CX (x86, ARM) | TcMMS.Lib |

## 5.2.2.8 ST_MMS_ApplicationAssociation

```
TYPE ST_MMS_ApplicationAssociation :
STRUCT
    aaIdentifier    : UDINT := 0;
    authentPassword : STRING(MAX_MMS_PASSWORD_LENGTH) := '';

    services        : ST_MMS_ServiceSupportOptions;
    parameters      : ST_MMS_ParameterSupportOptions;
    nestingLevel    : SINT := 0;


    bAuthentUser    : BOOL := FALSE;
(* TRUE = User with authentication, FALSE = User without authentication *)
    eEnvironment    : E_MMS_Environment := eMMS_Environment_None;
END_STRUCT
END_TYPE
```

**aaIdentifier**: Application-Association-Identifier.

**authentPassword**: Authentication-password [▶ 53].

**services**: Supported MMS-services [▶ 49].

**parameters**: Supported MMS-CBB-parameters [▶ 50].

**nestingLevel**: max. interlacing-depth of the MMS-data structure.

**bAuthentUser**: This variable is set TRUE if you used an authentication-password at the buildup of the application-association

**eEnvironment**: Status of the MMS- runtime environment [▶ 46].

### Requirements

| Deveopment environment | Target system | PLC libraries to include |
|---|---|---|
| TwinCAT v2.10.0 Build >= 1340 | PC or CX (x86, ARM) | TcMMS.Lib |

## 5.2.2.9 ST_MMS_DebugPara

```
TYPE ST_MMS_DebugPara :
STRUCT
    bError  : BOOL := FALSE; (* Enable/disable error messages. *)
    bLog    : BOOL := FALSE; (* Enable/disable log messages. *)

    mms     : ST_ProtocolDebugPara;
    ulosi   : ST_ProtocolDebugPara;
```

```
    cotp    : ST_ProtocolDebugPara;
END_STRUCT
END_TYPE
```

**bError**: Activates / deactivates the output of error messages in the TwinCAT System Manager logger-view-window.

**bLog**: Activates / deactivates the output of log messages (information, warnings) in the TwinCAT System Manager logger-view-window.

**mms**: This structure [▶ 52] contains the configuration parameters for the debug output of the MMS-layer.

**ulosi**: This structure contains the configuration parameters for the debug outout of the upper-layer-OSI-layer (ACSI, Presentation, Session).

**cotp**: This structure contains the configuration parameters for the debug output of the transport-layer.

### Requirements

| Development environment | Target system | PLC libraries to include |
|---|---|---|
| TwinCAT v2.10.0 Build >= 1340 | PC or CX (x86, ARM) | TcMMS.Lib |

## 5.2.2.10 ST_ProtocolDebugPara

```
TYPE ST_ProtocolDebugPara:
STRUCT
    eTx   : E_DbgDirection := eDbgDirection_OFF;(* Enable/disable debug output of data-requests. *)
    eRx   : E_DbgDirection := eDbgDirection_OFF;(* Enable/disable debug output of data-
indications. *)
    eCtrl : E_DbgDirection := eDbgDirection_OFF;(* Enable/
disable debug output of connection control events/actions. *)
END_STRUCT
END_TYPE
```

**eTx** : Activates the output of the outgoing telegrams.

**eRx** : Activates the output of the incoming telegrams.

**eCtrl** :Activates the output of the control news.

### Requirements

| Development environment | Target system | PLC libraries to include |
|---|---|---|
| TwinCAT v2.10.0 Build >= 1340 | PC or CX (x86, ARM) | TcMMS.Lib |

## 5.2.2.11 E_COTP_DUsize

```
TYPE E_COTP_DUsize:
(
    eCOTP_DUsize_Unused := 0, (* *)
    eCOTP_DUsize_128 := 7, (* 128 byte (default) *)
    eCOTP_DUsize_256 := 8, (* 256 byte *)
    eCOTP_DUsize_512 := 9, (* 512 byte *)
    eCOTP_DUsize_1024 := 10, (* 1024 byte *)
    eCOTP_DUsize_2048 := 11, (* 2048 byte *)
    eCOTP_DUsize_4096 := 12 , (* 4096 byte *)
    eCOTP_DUsize_8192 := 13 , (* 8192 byte *)
    eCOTP_DUsize_16384 := 14, (* 16384 byte *)
    eCOTP_DUsize_32768 := 15 (* 32768 byte *)
);
END_TYPE
```

Larger data files will be segmented from the transport layer. This variable configures the maximum length of a data segment.

**Requirements**

| Development environment | Target system | PLC-libraries to be included |
|---|---|---|
| TwinCAT v2.10.0 Build >= 1340 | PC or CX (x86, ARM) | TcRFC1006.Lib |

## 5.2.3 Constants

### 5.2.3.1 MMS constants

| Value | Value | Description |
|---|---|---|
| MAX_MMS_APPLICATION_ASSOCIATIONS | 1 | Maximum amount of coexistent active Application-Associations. |
| MAX_MMS_PASSWORD_LENGTH | 10 | Maximum length of the authentication-password. |
| MAX_MMS_ACCESS_CONTROL_USERS | 4 | Maximum amount of the user account. |

**Requirements**

| Development environment | Target system | PLC libraries to include |
|---|---|---|
| TwinCAT v2.10.0 Build >= 1340 | PC or CX (x86, ARM) | TcMMS.Lib |

# 5.3 TcACSI

## 5.3.1 Function blocks

### 5.3.1.1 FB_ACSI_CreateDataSetFromCSV


With this function block DataSets for the IEC 61850/61400-25 can be generated. The generation of the DataSets only can be done after the completion of the initialization of all of the Logical Devices.

**VAR_IN_OUT**

```
VAR_IN_OUT
    db : T_HACSIDB; (* ACSI configuration database *)
END_VAR
```

**db**: Is the database of the IEC-Server

**VAR_INPUT**

```
VAR_INPUT
    sDSRef  : T_MaxString := ''; (* data-set reference string (data-set name)*)
    pList   : DWORD := 0; (* Adress of data-set member list (CSV-String) *)
    cbList  : UDINT := 0; (* Byte size of data-set member list (CSV-String) *)
END_VAR
```

**sDSRef**: Names all DataSet references of the corresponding DataSet-instance.

**pList**: Pointer (address) onto the list of the structure of the DataSet-instance. The address can be found out with the help of the ADR-Operator.

**cbList**: Length of the list of the DataSet-instance.

### VAR_OUTPUT

```
VAR_OUTPUT
    eError : E_ACSI_ServiceError := eACSI_ServiceError_Success;
END_VAR
```

**eError**: Give-back parameter. See E_ACSI_ServiceError [▶ 58]

### Example:

In the following example the generation of the DataSet that is applied in the TcTelecontrol for the IEC-server is shown

```
PROGRAM MAIN
VAR
    fbCreateDataSet : FB_ACSI_CreateDataSetFromCSV; DataSet_1 : ST_DataSet_IEDDevice_XCBR1_DataSet_1
; (* Your configured DataSet from TcTelecontrol *)
END_VAR
```

```
...
(* create DataSet *)
fbCreateDataSet(sDSRef := DataSet_1.sDSRef, pList := ADR(DataSet_1.sList), cbList := LEN(DataSet_1.s
List) +1, db := fbIEC61850Server.system.db, eError =>errID);
```

### Requirements

| Development environment | Target system | PLC-libraries to be included |
|---|---|---|
| TwinCAT v2.10.0 Build >= 1340 | PC or CX (x86, ARM) | TcACSI.Lib |

## 5.3.1.2 Event Detection FUNCTION_BLOCKS

This function blocks will be used to detect value changes at the variables of an IEC-server. Different events can be detected at a variable. The block can also be used to send a reporting frame to a client. Therefore the corresponding variables in a Report Control Block need to be available as a reference.

This form of function blocks exists for every IEC-data type in the TcACSI.lib. As the function blocks only differ from this data type there is an overview of the function blocks on this page where some of them are described.

Every function block offers three actions:

- **A_SetValue:** Sets a value in the IEC data structure and informs the server-block that the value of a data attribute has been described with a new or the same value (UPDATE or CHANGE). That is the only way how the server block can detect that he needs to generate a reporting for this data attribute. A reporting only can be generated if this attribute has been configurated in the corresponding dataset of the Report-Control-Block.
- **A_GetEvent:** Identifies if a client has changed or read a data attribute in the IEC-data structure. Now these events can be detected: ACSI_EVENT_WRITE, ACSI_EVENT_READ, ACSI_EVENT_UPDATE, ACSI_EVENT_CHANGE. A_GetEvent is used in combination with A_ClearEvent.
- **A_ClearEvent:** Sets back a detected event, so that the next change can be detected.

| | A_SetValue | A_GetEvent | A_ClearEvent |
|---|---|---|---|
| **value, bits, or octets** | The new attribute value that should be written | Is not used | Is not used |
| **event_ctrl** | Reserved for future applications and should be<br>zero (0) | Is not used | Defines which event should be resetted/ accepted. Multiple events can be resetted with one call. The following values can be "wired-OR": |

| | A_SetValue | A_GetEvent | A_ClearEvent |
|---|---|---|---|
| | | | ACSI_EVENT_WRITE, ACSI_EVENT_READ, ACSI_EVENT_UPDATE, ACSI_EVENT_CHANGE. To set back all the values at the same time you can use the value 0xFFFFFFFF. |
| **event_ind** | This variable is a bitmask. Every bit is equivalent to an event. An event applies when the corresponding bit will be set. | This variable is a bitmask. Every bit is equivalent to an event. An event applies when the corresponding bit will be set. | This variable is a bitmask. Every bit is equivalent to an event. An event applies when the corresponding bit will be set. |
| **event_cnf** | Reserved for future applications and is not used now | Reserved for future applications and is not used now | Reserved for future applications and is not used now |

### Function blocks with interface "value":

```
FB_ACSI_BOOLEAN   ⇒ BOOLEAN

FB_ACSI_INT8              ⇒ INT8
FB_ACSI_INT16             ⇒ INT16
FB_ACSI_INT32             ⇒ INT32
FB_ACSI_INT64             ⇒ INT64
FB_ACSI_INT128            ⇒ INT128

FB_ACSI_UINT8             ⇒ UINT8
FB_ACSI_UINT16            ⇒ UINT16
FB_ACSI_UINT32            ⇒ UINT32
FB_ACSI_UINT64            ⇒ UINT64
FB_ACSI_UINT128           ⇒ UINT128

FB_ACSI_FLOAT32           ⇒ FLOAT32
FB_ACSI_FLOAT64           ⇒ FLOAT64

FB_ACSI_ENUMERATED        ⇒ ENUMERATED

FB_ACSI_VISIBLESTRING255 ⇒ VISIBLESTRING255
FB_ACSI_UNICODESTRING255 ⇒ UNICODESTRING64

FB_ACSI_ENTRYTIME         ⇒ ENTRYTIME
FB_ACSI_TIMESTAMP         ⇒ TIMESTAMP
FB_ACSI_ENTRYID           ⇒ ENTRYID
FB_ACSI_QUALITY           ⇒ QUALITY
FB_ACSI_TRIGGERCON        ⇒ TRIGGERCON
FB_ACSI_OPTIONFLDS        ⇒ OPTIONFLDS
```

### Function block FB_ACSI_BOOLEAN:

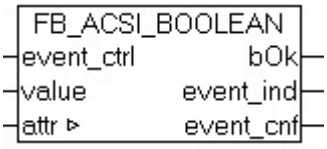



Fig. 1: TcPlcLibACSI_FB_ACSI_BOOLEAN

### VAR_IN_OUT

```
VAR_IN_OUT
    attr        : T_ACSI_BOOLEAN; (* ACSI attribute; It depends on the data type *)END_VAR
```

**attr**: Is the attribute of the IEC-standard from the corresponding data type.

### VAR_INPUT

```
VAR_INPUT
    event_ctrl  : DWORD := 16#FFFFFFFF;
(* Event control mask (used by A_ClearEvent), default: clear all events *)
    value       : BOOL;(* New value to set; The data type depends on the Function Block *)
END_VAR
```

**event_ctrl**: See chart above.

**value**: See chart above.

### VAR_OUTPUT

```
VAR_OUTPUT
    bOk         : BOOL;(* TRUE = Success, FALSE = Access failed *)
    event_ind   : DWORD := 16#00000000;
(* Event indication bits, bit set => event indication, bit reset => no event indication *)
    event_cnf   : DWORD := 16#FFFFFFFF;
(* Event confirmation bits, bit set => event confirmed, bit reset => waiting for event confirmation
*)
END_VAR
```

**bOk**: If this value is TRUE, no errors appeared, and the editing can go on. If it is FALSE an error exists.

**event_ind**: See chart above.

**event_cnf**: See chart above.

### Example:

In the following example the use of A_SetValue, A_GetEvent and A_ClearEvent will be shown.

```
PROGRAM MAIN
VAR
    dataCtrl : ST_ACSI_DATACTRL; bValue  : T_ACSI_BOOLEAN; (* Value from the data type BOOLEAN *)
END_VAR
---------------------(* Read Value *)
ctrl.BOOLEAN.A_GetEvent(attr := IEDRelay.XCBR1.BlkCls.stVal, event_ctrl := 0);
IF ctrl.BOOLEAN.bOk THEN  (* Detect when value is written (check the event_ind bits )*)IF (ctrl.BOOL
EAN.event_ind AND ACSI_EVENT_WRITE) = ACSI_EVENT_WRITE THEN
        write_bool_counter := write_bool_counter + 1;
    END_IF(* Detect when value is read (check the event_ind bits) *)IF (ctrl.BOOLEAN.event_ind AND A
CSI_EVENT_READ) = ACSI_EVENT_READ THEN
        read_bool_counter := read_bool_counter + 1;
    END_IF(* clear events (is necessary to detect the next event)*)
    ctrl.BOOLEAN.A_ClearEvent(attr := IEDRelay.XCBR1.BlkCls.stVal, event_ctrl := ACSI_EVENT_WRITE OR
 ACSI_EVENT_READ);(* clear read and write events *)END_IF
---------------------(* Write value *)
dataCtrl.BOOLEAN.A_SetValue(attr :=IEDDevice.XCBR1.SumSwARs.actVal, value := bValue.value, event_ctr
l := 0);
```

### Function block with Interface "bits":

```
FB_ACSI_CODEDENUM8      ⇒ CODEDENUM8
FB_ACSI_CODEDENUM16     ⇒ CODEDENUM16
FB_ACSI_CODEDENUM32     ⇒ CODEDENUM32
FB_ACSI_CODEDENUM64     ⇒ CODEDENUM64
FB_ACSI_CODEDENUM128    ⇒ CODEDENUM128
```

### Function block FB_ACSI_CODEDENUM8:

### VAR_IN_OUT

```
VAR_IN_OUT
    attr        : T_ACSI_CODEDENUM8; (* ACSI attribute; It depends on the data type *)END_VAR
```

**attr**: Is the attribute of the IEC-standard from the corresponding data type.

### VAR_INPUT

```
VAR_INPUT
    event_ctrl  : DWORD := 16#FFFFFFFF;
(* Event control mask (used by A_ClearEvent), default: clear all events *)
    bits        : ARRAY[0..0] OF BYTE := 1(0); (* New value to set; The data type depends on the Fun
ction Block *)
END_VAR
```

**event_ctrl**: See chart above.

**bits**: See chart above.

### VAR_OUTPUT

```
VAR_OUTPUT
    bOk         : BOOL;(* TRUE = Success, FALSE = Access failed *)
    event_ind   : DWORD := 16#00000000;
(* Event indication bits, bit set => event indication, bit reset => no event indication *)
    event_cnf   : DWORD := 16#FFFFFFFF;
(* Event confirmation bits, bit set => event confirmed, bit reset => waiting for event confirmation
*)
END_VAR
```

**bOk**: If this value is TRUE, no errors appeared, and the editing can go on. If it is FALSE an error exists.

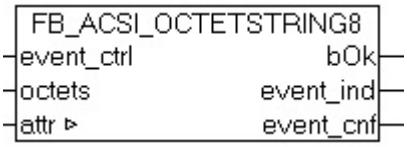
**event_ind**: See chart above.

**event_cnf**: See chart above.

### Function block with Interface "octets":

```
FB_ACSI_OCTETSTRING8    ⇒ OCTETSTRING8
FB_ACSI_OCTETSTRING16   ⇒ OCTETSTRING16
FB_ACSI_OCTETSTRING32   ⇒ OCTETSTRING32
FB_ACSI_OCTETSTRING64   ⇒ OCTETSTRING64
```

### Function block FB_ACSI_OCTETSTRING8:




### VAR_IN_OUT

```
VAR_IN_OUT
    attr        : T_ACSI_OCTETSTRING8; (* ACSI attribute; It depends on the data type *)END_VAR
```

**attr**: Is the attribute of the IEC-standard from the corresponding data type.

### VAR_INPUT

```
VAR_INPUT
    event_ctrl  : DWORD := 16#FFFFFFFF;
(* Event control mask (used by A_ClearEvent), default: clear all events *)
    octets      : ARRAY[0..7] OF BYTE := 8(0);
(* OCTETS: 8 byte; New value to set; The data type depends on the Function Block *)
END_VAR
```

**event_ctrl**: See chart above.

**bits**: See chart above.

### VAR_OUTPUT

```
VAR_OUTPUT
    bOk         : BOOL;(* TRUE = Success, FALSE = Access failed *)
    event_ind   : DWORD := 16#00000000;
(* Event indication bits, bit set => event indication, bit reset => no event indication *)
    event_cnf   : DWORD := 16#FFFFFFFF;
(* Event confirmation bits, bit set => event confirmed, bit reset => waiting for event confirmation
*)
END_VAR
```

**bOk**: If this value is TRUE, no errors appeared, and the editing can go on. If it is FALSE an error exists.

**event_ind**: See chart above.

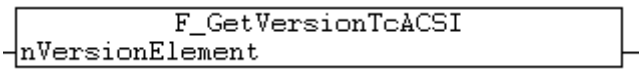
**event_cnf**: See chart above.

### Requirements

| Development environment | Target system | PLC libraries to be included |
|---|---|---|
| TwinCAT v2.10.0 Build >= 1340 | PC or CX (x86, ARM) | TcACSI.Lib |

## 5.3.2 Functions

### 5.3.2.1 F_GetVersionTcACSI




With this function versioninformation of the PLC-library can be readout.

### FUNCTION F_GetVersionTcACSI : UINT

```
VAR_INPUT
    nVersionElement : INT;
END_VAR
```

**nVersionElement** : Version element, that should be readout. Possible parameter:

- 1 : major number;
- 2 : minor number;
- 3 : revision number;

### Requirements

| Development environment | Target system | PLC libraries to include |
|---|---|---|
| TwinCAT v2.10.0 Build >= 1340 | PC or CX (x86, ARM) | TcACSI.Lib |

## 5.3.3 Data types

### 5.3.3.1 E_ACSI_ServiceError

ACSI service errors

| Constant | Value | Description |
|---|---|---|
| eACSI_ServiceError_Success | 0x0000 | No error |
| eACSI_ServiceError_InstanceNotAvailable | 0x8500 | Instance not available |

| Constant | Value | Description |
|---|---|---|
| eACSI_ServiceError_InstanceInUse | 0x8501 | Instance in use |
| eACSI_ServiceError_AccessViolation | 0x8502 | Access violation |
| eACSI_ServiceError_AccessNotAllowedInCurrentState | 0x8503 | Access is not allowed in the current state |
| eACSI_ServiceError_ParameterValueInappropriate | 0x8504 | Parameter value is not qualified |
| eACSI_ServiceError_ParameterValueInconsistent | 0x8505 | Parameter value is inconsequent |
| eACSI_ServiceError_ClassNotSupported | 0x8506 | Class is not supported |
| eACSI_ServiceError_InstanceLockedByOtherClient | 0x8507 | Instance is blocked by another client |
| eACSI_ServiceError_ControlMustBeSelected | 0x8508 | Service must be selected |
| eACSI_ServiceError_TypeConflict | 0x8509 | Type conflict |
| eACSI_ServiceError_FailedDueToCommunicationsConstraint | 0x850A | Failed because of limitation in the communication |
| eACSI_ServiceError_FailedDueToServerConstraint | 0x850B | Failed because of limitation in the server |

**Requirements**

| Development environment | Target system | PLC-libraries to include |
|---|---|---|
| TwinCAT v2.10.0 Build >= 1340 | PC or CX (x86, ARM) | TcACSI.Lib |

### 5.3.3.2 ST_ACSI_DATACTRL

The structure DATACTRL contains function blocks for all IEC-data types to accomplish a control of value changes. These can be used for the realization of the reporting functionality.

```
TYPE ST_ACSI_StartTag:(* ACSI-Type start delimiter *)
STRUCT
    BOOLEAN          : FB_ACSI_BOOLEAN;

    INT8             : FB_ACSI_INT8;
    INT16            : FB_ACSI_INT16;
    INT32            : FB_ACSI_INT32;
    INT64            : FB_ACSI_INT64;
    INT128           : FB_ACSI_INT128;

    UINT8            : FB_ACSI_UINT8;
    UINT16           : FB_ACSI_UINT16;
    UINT32           : FB_ACSI_UINT32;
    UINT64           : FB_ACSI_UINT64;
    UINT128          : FB_ACSI_UINT128;

    FLOAT32          : FB_ACSI_FLOAT32;
    FLOAT64          : FB_ACSI_FLOAT64;

    ENUMERATED       : FB_ACSI_ENUMERATED;

    CODEDENUM8       : FB_ACSI_CODEDENUM8;
    CODEDENUM16      : FB_ACSI_CODEDENUM16;
    CODEDENUM32      : FB_ACSI_CODEDENUM32;
    CODEDENUM64      : FB_ACSI_CODEDENUM64;
    CODEDENUM128     : FB_ACSI_CODEDENUM128;

    OCTETSTRING8     : FB_ACSI_OCTETSTRING8;
    OCTETSTRING16    : FB_ACSI_OCTETSTRING16;
    OCTETSTRING32    : FB_ACSI_OCTETSTRING32;
    OCTETSTRING64    : FB_ACSI_OCTETSTRING64;

    VISIBLESTRING255 : FB_ACSI_VISIBLESTRING255;
    UNICODESTRING64  : FB_ACSI_UNICODESTRING255;

    ENTRYTIME        : FB_ACSI_ENTRYTIME;
    TIMESTAMP        : FB_ACSI_TIMESTAMP;
    ENTRYID          : FB_ACSI_ENTRYID;
    QUALITY          : FB_ACSI_QUALITY;
    TRIGGERCON       : FB_ACSI_TRIGGERCON;
```

```
    OPTIONFLDS       : FB_ACSI_OPTIONFLDS;
END_STRUCT
END_TYPE
```

### Requirements

| Development environment | Target system | PLC libraries to be included |
|---|---|---|
| TwinCAT v2.10.0 Build >= 1340 | PC or CX (x86, ARM) | TcACSI.Lib |

## 5.3.4 Constants

### 5.3.4.1 ACSI constants

#### ACSI-Objectdatabase-Parameter

| Constant | Value | Description |
|---|---|---|
| MAX_ACSI_OBJECTNODE_ALLOC | 30000 | Max. number of the objects that are available |
| MAX_ACSI_OBJECTNAME_LENGTH | 31 | Max. Object name-String length (symbols) |

#### Functional Constraints bit mask

All functional constraints of the basic norm IEC 61850 were implemented in the TwinCAT PLC as a constant of the ACSI Library.

```
ACSI_FC_ST :DWORD := 16#00000001; (* Status information (use of FC in the definition of DATA) *)
ACSI_FC_MX :DWORD := 16#00000002; (* Measurands (analogue values) (use of FC in the definition of DA
TA) *)
ACSI_FC_CO :DWORD := 16#00000004; (* Control (use of FC in the definition of DATA) *)
ACSI_FC_SP :DWORD := 16#00000008; (* Setpoint (use of FC in the definition of DATA and control block
s) *)
ACSI_FC_SV :DWORD := 16#00000010; (* Substitution (use of FC in the definition of DATA) *)
ACSI_FC_CF :DWORD := 16#00000020; (* Configuration (use of FC in the definition of DATA) *)
ACSI_FC_DC :DWORD := 16#00000040; (* Description (use of FC in the definition of DATA) *)
ACSI_FC_SG :DWORD := 16#00000080; (* Setting group (use of FC in the definition of DATA) *)
ACSI_FC_SE :DWORD := 16#00000100; (* Setting group editable (use of FC in the definition of DATA) *)
ACSI_FC_EX :DWORD := 16#00000200; (* Extended definition (use of FC in the definition of DATA) *)
ACSI_FC_BR :DWORD := 16#00000400; (* Buffered report (-
> Reserved for control classes and use of the FC in the definition of control blocks) *)
ACSI_FC_RP :DWORD := 16#00000800; (* Unbuffered report (-
> Reserved for control classes and use of the FC in the definition of control blocks) *)
ACSI_FC_LG :DWORD := 16#00001000; (* Logging (-
> Reserved for control classes and use of the FC in the definition of control blocks) *)
ACSI_FC_GO :DWORD := 16#00002000; (* Goose control (-
> Reserved for control classes and use of the FC in the definition of control blocks) *)
ACSI_FC_GS :DWORD := 16#00004000; (* Gsse control (-
> Reserved FOR control classes AND use OF the FC in the definition OF control blocks) *)
ACSI_FC_MS :DWORD := 16#00008000; (* Multicast sampled value control (-
> Reserved for control classes and use of the FC in the definition of control blocks) *)
ACSI_FC_US :DWORD := 16#00010000; (* Unicast sampled value control (-
> Reserved for control classes and use of the FC in the definition of control blocks) *)
ACSI_FC_XX :DWORD := 16#00020000; (* Shall represent all DataAttributes of a Data of any FC. "XX" sh
all not used as value in DataAttributes *)
```

#### Trigger Options bit mask

```
(* ACSI_TrgOp_rsv : DWORD := 16#80; reserved *)
ACSI_TrgOp_dchg : DWORD := 16#40;(* data-change *)
ACSI_TrgOp_qchg : DWORD := 16#20;(* quality-change *)
ACSI_TrgOp_dupd : DWORD := 16#10;(* data-update *)
ACSI_TrgOp_intg : DWORD := 16#08;(* integrity *)
ACSI_TrgOp_gi : DWORD := 16#04;(* general-interrogation *)
```

### Requirements

| Develoment environment | Target system | PLC libraries to include |
|---|---|---|
| TwinCAT v2.10.0 Build >= 1340 | PC or CX (x86, ARM) | TcACSI.Lib |

# 6 Examples

The examples use the following Host-IP-address: "127.0.0.1" (Loopback-Adapter). The Host-address in the example project needs to be consistent with the address of the target system where the server should run. For example, if you have a CX10xx with the IP-address "172.16.11.83" that should be the Server you need to configurate the IP-address in the example project of the CX10xx and download the project on this device. In the Client (control station) the address of the CX10xx needs to be entered as well as a host-address.

The maximum number of nodes at an evaluation version is limited on 3.

| Example | Description |
|---|---|
| https://infosys.beckhoff.com/content/1033/TcPlcLibIEC61400_25Server/Resources/11681717259/.zip | An IEC 61400-25 Demo-Server with the following nodes: LLN0, LPHD, WCNV, WGEN, WNAC, WROT, WTUR, WYAW |

# 7 Appendix

## 7.1 TCP Keep-Alive Messages

A Keep-Alive Message is a confirmation message or an "acknowledge". Therewith you can proof in the backround if a communication partner who established a connection, is still active and therefore is participating in the communication. If the communication partner should not be active anymore, the communication channel will be closed clearly and regular, to be free for a new attendant.

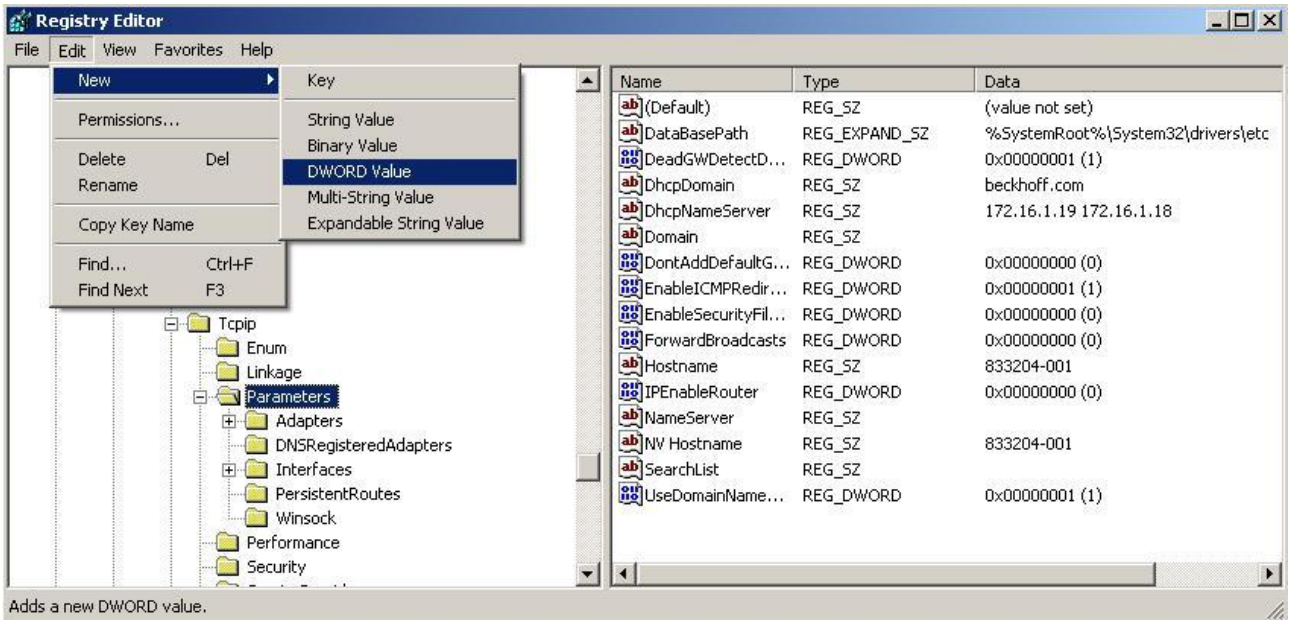
Keep-Alive can be configurated in the registration by the keys Keep-Alive-Time and Keep-Alive-Interval. The default value for the Keep-Alive-Time is set on two hours. The interval-time is set by default to one second. This is the time until there is a repetition of a not-answered Alive-request. **All values are stated in milliseconds.** If the corresponding key is not listed in the registration they need to be applied as follows. These are no TwinCAT-settings but specific settings of the operating system. Therefore you will find further information in the operating system documentation, e.g. at the Microsoft homepage.

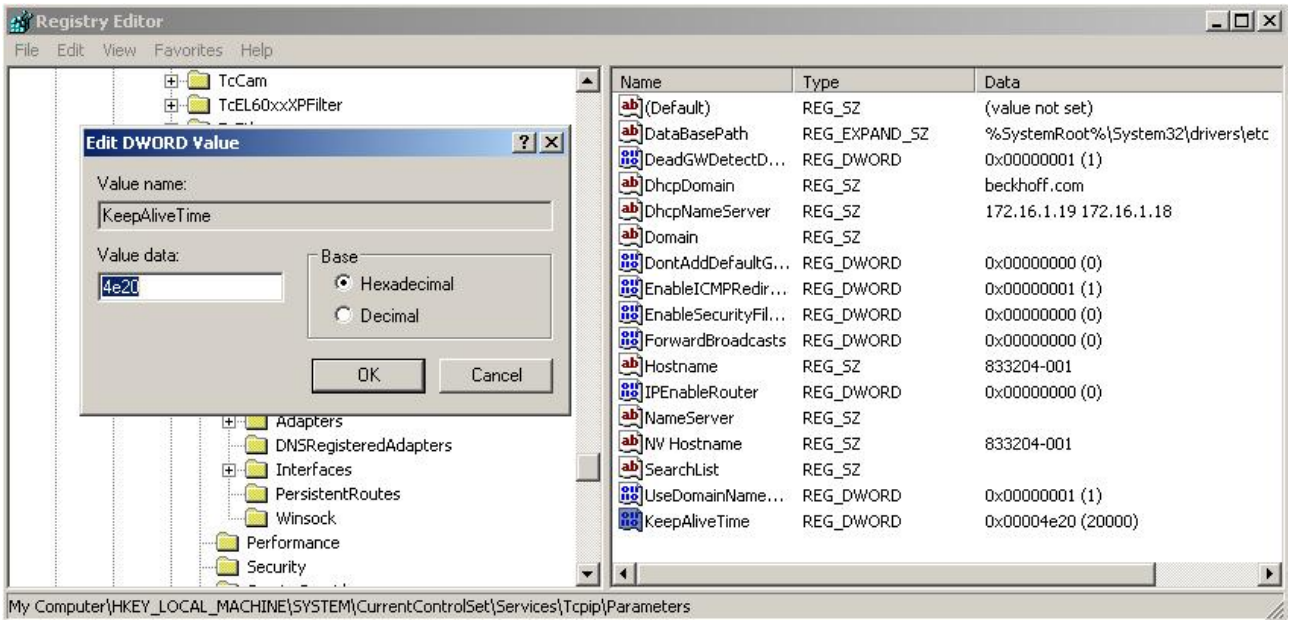
**Under Windows W2K, XP, Windows Embedded Standard:**

- In the folder HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\ an object with the name KeepAliveTime needs to be applied as DWORD. Therefore you need to insert under Edit -> New -> DWORD Value. If you set it to the value 0x4E20, this equates 20 seconds.
- In the folder HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\ an object with the name KeepAliveInterval needs to be applied as DWORD. Therefore you need to insert under Edit -> New -> DWORD Value. If you set it to the value 0x1388, this equates 5 seconds.

**Under Windows CE:**

- In the folder HKEY_LOCAL_MACHINE\Comm\TcpIp\Parms\ an object with the name KeepAliveTime needs to be applied as DWORD. Therefore you need to insert under Edit -> New -> DWORD Value. If you set it to the value 0x4E20, this equates 20 seconds.
- In the folder HKEY_LOCAL_MACHINE\Comm\TcpIp\Parms\ an object with the name KeepAliveInterval needs to be applied as DWORD. Therefore you need to insert under Edit -> New -> DWORD Value. If you set it to the value 0x1388, this equates 5 seconds.






## 7.2 Firewall Settings

The TwinCAT IEC 61850/61400-25 server uses during the client/server communication (MMS mapping) the TCP/IP as transport protocol (T profile). You must regard, that the correspondent TCP port is enabled by using a firewall. The table below lists the ports, that must be considered by using a firewall.

| Description | Type | Protocol | Port |
|---|---|---|---|
| Connection Oriented Transport Protocol (COTP, ISO 8073) | Protocol | TCP | 102 |

The configuration of the windows firewall is done via a dialog in the system control. Further information can be found in the windows resp. firewall documentation.

Please note, if you use an embedded controller without display and USB, you have to enable the port for remote display (Windows CE) or remote desktop (Windows XP / Windows Vista) in the firewall. Otherwise you have no possibility to administrate the computer via network.

More Information:
**www.beckhoff.com/ts6509**

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Germany
Phone: +49 5246 9630
info@beckhoff.com
www.beckhoff.com