## Table of contents

1 Foreword .................................................................................................................. 5  
   1.1 Notes on the documentation ............................................................................. 5  
   1.2 Safety instructions ......................................................................................... 6  
   1.3 Notes on information security ........................................................................ 7  
2 Overview .................................................................................................................. 8  
3 Installation ................................................................................................................ 9  
   3.1 System requirements ...................................................................................... 9  
   3.2 Installation ...................................................................................................... 9  
   3.3 Licensing ....................................................................................................... 12  
4 Analytics Workflow - First Steps ....................................................................... 15  
   4.1 Recording data from the machine .................................................................. 15  
   4.2 Communication ............................................................................................ 18  
   4.3 Historicize data ............................................................................................. 19  
   4.4 Import Analytics Files .................................................................................. 24  
   4.5 Analyse data ................................................................................................ 25  
   4.6 24h Analytics application ............................................................................. 30  
5 Technical introduction ......................................................................................... 40  
6 Configuration ......................................................................................................... 42  
   6.1 Service .......................................................................................................... 42  
   6.2 Databases/Stores ........................................................................................... 44  
      6.2.1 Analytics Binary File ............................................................................. 44  
      6.2.2 Microsoft SQL ...................................................................................... 45  
      6.2.3 Microsoft Azure Blob .......................................................................... 47  
   6.3 Recorder ......................................................................................................... 49  
   6.4 Working with Historical Data ........................................................................ 53  
7 PLC API .................................................................................................................. 58  
   7.1 Function blocks ............................................................................................... 58  
      7.1.1 Topic Architecture ............................................................................... 58  
      7.1.2 FB_ALY_StorageProvider .................................................................. 64  
   7.2 Data types ....................................................................................................... 67  
      7.2.1 E_CancelType ....................................................................................... 67  
      7.2.2 E_RawDataFormat .............................................................................. 68  
      7.2.3 E_SymbolMode .................................................................................... 68  
      7.2.4 ST_ALY_SP_Config ............................................................................. 68  
      7.2.5 ST_ConnectionSettings ...................................................................... 69  
      7.2.6 E_RecordMode ...................................................................................... 69  
      7.2.7 E_RingBufferMode .............................................................................. 70  
8 Samples .................................................................................................................. 71  
   8.1 PLC Client ...................................................................................................... 71  
9 Appendix ............................................................................................................... 76  
   9.1 FAQ - frequently asked questions and answers ............................................ 76  
   9.2 Third-party components ................................................................................. 76
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:
with corresponding applications or registrations in various other countries.

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

Copyright

© Beckhoff Automation GmbH & Co. KG, Germany.
The reproduction, distribution and utilization of this document as well as the communication of its contents to others without express authorization are prohibited.
Offenders will be held liable for the payment of damages. All rights reserved in the event of the grant of a patent, utility model or design.
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!

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="DANGER" /></td>
<td>Serious risk of injury! Failure to follow the safety instructions associated with this symbol directly endangers the life and health of persons.</td>
</tr>
<tr>
<td><img src="image" alt="WARNING" /></td>
<td>Risk of injury! Failure to follow the safety instructions associated with this symbol endangers the life and health of persons.</td>
</tr>
<tr>
<td><img src="image" alt="CAUTION" /></td>
<td>Personal injuries! Failure to follow the safety instructions associated with this symbol can lead to injuries to persons.</td>
</tr>
<tr>
<td><img src="image" alt="NOTE" /></td>
<td>Damage to the environment or devices Failure to follow the instructions associated with this symbol can lead to damage to the environment or equipment.</td>
</tr>
</tbody>
</table>

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 Overview

Beckhoff offers with the TwinCAT Analytics Storage Provider a possibility to store your high resolution data into a database or store by the usage of the TwinCAT Analytics Binary data format. The key is that the user does not have to worry about the structure of the data or tables. This is done automatically by the Storage Provider, which puts all that has been configured into a database or store and also searches the desired variables for the post-analysis.

Components

- TwinCAT Analytics Storage Provider Service: Is a Windows Service who manages the communication.
- TwinCAT Analytics Storage Provider PLC Library: Is a TwinCAT 3 PLC library with functions to control the Storage Provider from a PLC application.

Principle of operation

The Analytics Storage Provider receives and sends data by MQTT. Therefore, he is connected to a native MQTT Message Broker somewhere in the network.

Supported databases/stores

- TwinCAT Analytics Binary File
- Microsoft SQL
- Microsoft Azure Blob
3 Installation

3.1 System requirements

See the requirements of the Service and the PLC library of the Analytics Storage Provider in the following tables. It is also possible to install both on one system as well.

<table>
<thead>
<tr>
<th>Technical data Service</th>
<th>TF3520 TC 3 Analytics Storage Provider</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target system</td>
<td>Windows 7, Windows 8, Windows 10</td>
</tr>
<tr>
<td>.NET Framework</td>
<td>.Net 4.5.1 or higher</td>
</tr>
<tr>
<td>Min. TwinCAT version</td>
<td>3.1.4022.25</td>
</tr>
<tr>
<td>Min. TwinCAT level</td>
<td>TC1000 TC3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Technical data Library</th>
<th>TF3520 TC 3 Analytics Storage Provider</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target system</td>
<td>Windows 7, Windows 8, Windows 10</td>
</tr>
<tr>
<td>Min. TwinCAT version</td>
<td>3.1.4022.29</td>
</tr>
<tr>
<td>Min. TwinCAT level</td>
<td>TC1200 TC3</td>
</tr>
</tbody>
</table>

3.2 Installation

The following section describes how to install the TwinCAT 3 Function for Windows-based operating systems.

- The TwinCAT 3 Function setup file was downloaded from the Beckhoff website.
1. Run the setup file as administrator. To do this, select the command **Run as administrator** in the context menu of the file.
   - The installation dialog opens.
2. Accept the end user licensing agreement and click **Next**.
3. Enter your user data.

4. If you want to install the full version of the TwinCAT 3 Function, select Complete as installation type. If you want to install the TwinCAT 3 Function components separately, select Custom.
5. Select **Next**, then **Install** to start the installation.

![TwinCAT Server Installation dialog box]

- A dialog box informs you that the TwinCAT system must be stopped to proceed with the installation.

6. Confirm the dialog with **Yes**.
7. Select Finish to exit the setup.

The TwinCAT 3 Function has been successfully installed and can be licensed (see Licensing [12]).

3.3 Licensing

The TwinCAT 3 function can be activated as a full version or as a 7-day test version. Both license types can be activated via the TwinCAT 3 development environment (XAE).

Licensing the full version of a TwinCAT 3 Function

A description of the procedure to license a full version can be found in the Beckhoff Information System in the documentation "TwinCAT 3 Licensing".

Licensing the 7-day test version of a TwinCAT 3 Function

A 7-day test version cannot be enabled for a TwinCAT 3 license dongle.

1. Start the TwinCAT 3 development environment (XAE).
2. Open an existing TwinCAT 3 project or create a new project.
3. If you want to activate the license for a remote device, set the desired target system. To do this, select the target system from the Choose Target System drop-down list in the toolbar.

The licensing settings always refer to the selected target system. When the project is activated on the target system, the corresponding TwinCAT 3 licenses are automatically copied to this system.
4. In the **Solution Explorer**, double-click **License** in the **SYSTEM** subtree.

   ![Solution Explorer](image1.png)

   - The TwinCAT 3 license manager opens.

5. Open the **Manage Licenses** tab. In the **Add License** column, check the check box for the license you want to add to your project (e.g. "TF4100 TC3 Controller Toolbox").

   ![Manage Licenses](image2.png)

6. Open the **Order Information (Runtime)** tab.

   - In the tabular overview of licenses, the previously selected license is displayed with the status "missing".

   ![Order Information (Runtime)](image3.png)
7. Click **7-Day Trial License**... to activate the 7-day trial license.

   ![License Activation Screen]

   A dialog box opens, prompting you to enter the security code displayed in the dialog.

   ![Enter Security Code]

   8. Enter the code exactly as it is displayed and confirm the entry.

   9. Confirm the subsequent dialog, which indicates the successful activation.

   ![License Status]

   In the tabular overview of licenses, the license status now indicates the expiry date of the license.

10. Restart the TwinCAT system.

   The 7-day trial version is enabled.
4 Analytics Workflow - First Steps

This step by step documentation presents the complete TwinCAT Analytics workflow. From the data acquisition over the communication and historizing up to the evaluation and analysis of the data and to the presentation of the data in web-based dashboard.

4.1 Recording data from the machine

On the machine side is the Analytics Logger the recorder of process data from the machine image, PLC, NC and so on. The Logger is working in the real-time context of TwinCAT.

The TwinCAT Analytics Logger is installed with TwinCAT XAE and XAR. The Logger can act as MQTT Client to communicate the recorded data to a native MQTT Message Broker or store the data in the same data format in a local binary file. By the usage as MQTT Client the Logger is able to bypass short disconnects to the Message Broker with a ring buffer functionality. You can configure a ring buffer as well for the local binary file storage.

- To configure the Analytics Logger you have to navigate in your existing TwinCAT Project to the Analytics tree node.
- Right click on this node and click on “Add Data Logger” to add one new instance to your configuration

You can make your specific Analytics Logger settings

- Data Format: Binary file or MQTT stream
  - FILE format: Analytics Logger stores the data in local binary files and all other settings are not necessary anymore. The files will be stored in \C\TwinCAT\3.1\Boot\Analytics.
  - BINARY: Data will be sent to the configured MQTT Message Broker. You can have multiple Logger in one TwinCAT project to communicate data to different MQTT Message Broker.
  - Data Compression: on (default) or off
- Max Compression: mode of the compression
- MQTT host name
- MQTT Tcp port
- MQTT main topic for own hierarchical levels to keep the identification easy
- MQTT Client ID should be unique in the network
- MQTT username
- MQTT password to make authentication at the message broker

- At the TLS (Transport Layer Security) tab, security settings can be configured. TLS is a secure communication channel between client and server. By the usage of certificates, the TCP port 8883 is exclusively reserved for MQTT over TLS. Analytics Logger is supporting the modes CA Certificates, CA Certificates & Client Certificate and Preshared Key (PSK) mode.

  - If variables in your PLC application are marked in the declaration with the attribute 'TcAnalytics' they will be shown automatically as a stream below the Data Logger tree node.

An additional device stream will be shown if your configuration provides an EtherCAT Process Image.
• In the stream a Selection tab is available to choose the variables that should be recorded.

• Finally it is possible to change the package size for the frames or to configure the ring buffer for disconnects and file in the Data Handling tab.

4.2 Communication

Currently, the Analytics workflow is fully mappable via MQTT. The engineering tools can also access the data of the machines via ADS and carry out analyzes.
If you choose for the IoT communication protocol MQTT you have to setup a native MQTT Message Broker somewhere in the network (VM in a cloud system is also possible). This Message Broker provides a decoupling of the different applications in the Analytics Workflow.

4.3 Historicize data

After installation of the TwinCAT Analytics Storage Provider you are able to configure the service running in the background. You will find therefore the TcAnalyticsStorageProvider_Config application in the folder C:\TwinCAT\Functions\TF3520-Analytics-StorageProvider\WinService.
The main part of the topic can be set in the configuration as well as the comment which will be used for identification if more than one Storage Provider is registered at the Message Broker.

You are able to provide the Message Broker settings and to decide for storage type:

- Analytics File (binary file)
- Microsoft SQL
- Microsoft Azure Blob (Azure Cloud necessary)
Finally, you can save the config and start the service. The next step is the configuration of the specific record. Therefore you should choose in your development environment the Storage Provider Recorder.

The recorder has also to connect to the Message Broker. So you have to provide the same settings as for the background service.

After this you can click to icon with the small cloud to search Storage Providers at the configured Message Broker. Here you will find also the comment you gave already by the service configuration to identify your Storage Provider.
The configuration of the record is very easy. You have just to choose your target in the Target Browser. Click on Live data and choose one or more variables by multiselect and put them by drag and drop to the recorder window.

The recorder will ask you if you like to add just the chose variables or the complete source process image of the variables.
You can also configure record names and a duration (otherwise endless until manual stop). A ringbuffer can be set by memory or time.

Click the Start button to start the record. After this you can also disconnect the recorder, because the background service do the work. It is also possible that someone else connect to this Storage Provider service and control the running record.
After and also during the record you can choose the historical data as input for your analysis in the Target Browser. In the Target Browser you will find for historical data a new control on the right hand side. There you can choose the time span for your data.

4.4 Import Analytics Files

In the following it is assumed that you have installed TwinCAT under "C:/TwinCAT". Otherwise, you must adjust the specified paths accordingly.

You can import recordings from the Analytics Logger stored in Analytics File Format (Analytics.tas, Analytics-<Date>.tay) into the Storage Provider.

To do this, perform the following steps:

1. Place the folder with the Analytics files in your Storage Provider location
   By default, here: C:\TwinCAT\Functions\TF3520-Analytics-StorageProvider\Storage\Analytics
   StorageProvider (create the folders manually if they do not exist)

2. Open the TcAnalyticsSPAlyFileStoreMigration.exe. The program can be found under the path C:\TwinCAT\Functions\TF3520-Analytics-StorageProvider\WinService.

3. Specify the path to your Storage Provider location
   By default, here: C:\TwinCAT\Functions\TF3520-Analytics-StorageProvider\Storage.

4. Press Scan file structure:
5. Press **Migrate**.

6. Now you can see your imported data in the **TwinCAT Target Browser** [53].

You may have to wait a short time or restart your Storage Provider.

### 4.5 Analyse data

Open your TwinCAT Engineering environment to start the data analysis.

Open **Visual Studio® > File > New > Project…**

Select the **Analytics project template** from **TwinCAT Measurement**.
The new project is displayed in the Solution Explorer. After clicking the Analytics Project tree node element a start window opens where you can select your first action. From here you can add a network, open the Toolbox, open the Target Browser or open the Analytics Storage Provider Recorder. In the following steps you will perform all these actions.

It makes sense to open the Toolbox of Visual Studio® first. There you will find all the algorithms supported by TwinCAT Analytics. Algorithms need to be grouped and organized into networks. Right-click Analytics Project to add a new network, or add a network using the start page. The first network is always generated by default.
When you click on the network, an editor opens. Now you can drag and drop the desired algorithm into the editor interface. After selecting the algorithm, you need to connect input variables to the modules (algorithm). To do this, open the **Target Browser**.

TwinCAT > Target Browser > Target Browser

Now select the **TcAnalytics** or **TcAnalyticsFile** tab in the Target Browser. We continue with the tab **TcAnalytics** (MQTT). Click the icon highlighted in green in the toolbar of this Analytics extension. A window opens in which you can specify the connectivity data of your message broker.
Select your MQTT Analytics client (TwinCAT Analytics Logger, TwinCAT IoT Data Agent or Beckhoff EK9160). Each controller has a unique ID. This ID is displayed in the Target Browser. Clicking the gear icon will take you to the Machine Administration page. Here you can assign a system alias name that will be displayed in the Target Browser instead of the ID.

In the next step, you can choose between live data and historical data for each MQTT Analytics client. In this case, the historical data is provided by the TwinCAT Analytics Storage Provider.
You can drag and drop the variables into the inputs of the specific algorithm. In most algorithms, conditions such as thresholds, time intervals, logical operators etc. can be specified. These settings are made in the middle of each module.

Finally, your first Analytics Project is complete. To start the analysis, click **Start Analytics**. To stop the analysis, click **Stop Analytics**.

Before starting the analysis or during runtime, you can click the **Add Reference Scope** button. This will automatically create a scope configuration that matches your Analytics project.
The analysis results can be displayed in the Scope View graphs using drag-and-drop. For example, a mean value can be displayed as a new channel in the view. Timestamps as markers on the x-axes show significant values.

### 4.6 24h Analytics application

The last big step in the TwinCAT Analytics workflow is the continuous 24h machine analysis. It runs parallel to the machine applications in the field. To do this in a very easy way the TwinCAT Analytics Workbench is able to generate automatically a PLC code and a HTML5-based Dashboard of your Analytics configuration. Both can be downloaded to a TwinCAT Analytics Runtime (TC3 PLC and HMI Server) and provide the same analysis results as the configurator tool in engineering environment.

At first save your configuration and open the Analytics Deploy Runtime wizard. You can do this via the context menu at the Analytics Project tree item or by the Start Page.
When the wizard is open, there are some tabs available for you to click through. First one is called Solution. Here it is possible to decide how you like to use your Analytics project in PLC code: As...

- completely new solution
- part of an existing solution
- update of an existing Analytics solution

In the TwinCAT PLC Target tab you can choose the ADS target system which runs the TwinCAT Analytics Runtime (TF3550). The generated project is immediately executable. Therefore, you can set the option Activate PLC Runtime. Also, that directly a boot project is created.
Specially for Virtual Machines it is important to run the project on isolated cores. Also, an option in this tab. The next tab Results is only necessary if you have chosen the Stream Results option in the algorithm properties. If you like to send results you can decide here in which way (local in a file/ by MQTT) and format (binary/JSON). Also, this is generated automatically and starts running after activation.
A down sampling of the results is possible by setting a cycle time. The next tab is reserved for the HMI Dashboard. The prerequisite for the automatic Dashboard generation is the selection of HMI controls for the corresponding algorithms whose results are to be displayed in the dashboard.
You can choose for your Analytics Dashboard different options like start page with a map, layouts, sorting algorithm, own colors and logos. If you choose more languages for the Analytics controls a language switch menu will be generated as well.
Choose one of the installed Visual Studio versions. And whether the generation should start the instance visibly or whether it should only be set up and activated in the background.
Finally you find a summery.
Now you can click the Deploy button to start the generation process. The PLC project and the HMI Dashboard will be generated now.
After the message Deploy Runtime succeeded you can find a new Visual Studio®/XAE Shell instance on your desktop. The new Solution and both Projects are generated.
5 Technical introduction

The basic idea of the TwinCAT Analytics Storage Provider is to have a gateway, which takes the user from the configuration of the databases and stores behind. The user does not need to setup his own table structure in a database. He needs just to configure which of the supported data sinks and variables from the data source he likes to use.

Service management

The Analytics Storage Provider Service can run somewhere in the network. He is implemented as Windows Service. The service can run on hardware devices like Industrial PCs or Embedded PCs in the local network and also on virtual machines in the same network or e.g. in a Cloud System.

Value management

The Storage Provider speaks the TwinCAT Analytics binary format. This enables him to receive and save streams from a MQTT Message Broker, as well as to create and send new streams himself. The user is managing that just by the Recorder integrated in his own engineering system with TwinCAT Analytics Workbench or Service Tool. The variables itself are shown into the TwinCAT Target Browser. For the Analytics binary format it is separated into Live and Historical data. Live data can be used as input for the Analytics Storage Provider. Historical data are the values from the database/store provided by the Storage Provider.

Topologies

The many degrees of freedom of the IoT technologies enable a variety of topologies. The following picture shows the most significant constellations.

1. Each SW package is running on own HW device or virtual machine
2. The Windows Service of the Analytics Storage Provider is running on the same device as the database/store
3. Analytics Engineering, Analytics Storage Provider and database or store are on the same device. Only the Message Broker and the Analytics Logger (data source) are on other devices
4. In this topology view is only the Analytics Logger on an own PC. Maybe in a machine application. All other tools of the Analytics tool chain are on one device include the MQTT Message Broker.
6 Configuration

The configuration of the Analytics Storage Provider is divided into two main parts. The user has to configure the Service of the Storage Provider and variables by a Recorder. You will also find the supported databases and stores in this chapter.

6.1 Service

The Windows service of the Analytics Storage Provider requires at least a TwinCAT 3 ADS as a basis. The setup automatically detects whether a TwinCAT 3 XAE is present or not. If available, also install the PLC library, otherwise only the Windows service.

After the installation you will find everything you need under C:\TwinCAT\Functions\TF3520-Analytics-StorageProvider. One hierarchical level down you will see the WinService folder, where you will find the TcAnalyticsStorageProvider_config.exe file.

Open this executable file and you will see the configuration interface.

- Analytics Storage Provider
  - **Main Topic:** set your own main topic here to identify a stream with historical data.
  - **Comment:** you can enter a comment that will be displayed later in the Recorder window in the Analytics Workbench or the Service Tool.

- Message broker

- Settings:
Enter here your message broker data such as URL, user name, password or certificates.

The settings can be checked. The result is displayed in the following window:

- **Storage**
  - **Type**: select the type of your data sink here, such as Analytics File or Microsoft SQL
  - **Connection String**: you can configure your Connection String manually or automatically through the given settings mask.
  
  For Microsoft SQL see:

- **Additional Properties**
  Two additional settings can be made in the Additional Properties.

  Max Write Length:
  For Analytics File: the amount of data that is written to the .tay file in one call is specified here
  For MsSQL: the amount of data that is stored in a tbl_Data data set is specified here

  Max Duration:
  For Analytics File: here, the time is specified, after which a new .tay file is created
With MsSQL: here, this setting is not relevant

- Analytics Storage Provider Windows Service
  **Status:** View the status and default settings for starting the Windows service

Of course, the created configuration can be saved. If you click **Save Config**, the settings in TcAlySPWinService.exe.config will be saved in `C:\TwinCAT\Functions\TF3520-Analytics-StorageProvider\WinService`.

## 6.2 Databases/Stores

### 6.2.1 Analytics Binary File

The TwinCAT Analytics Binary File is and TwinCAT specific data storage. Therefore no external software is necessary. You can use this kind of store directly after the installation of the Analytics Storage Provider. It is the same file what the TwinCAT Analytics Logger is providing in his "offline" configuration without MQTT Message Broker.
For the configuration it is enough that you choose your favorite folder on the local device where the Storage Provider is running.

![Folder Browser]

As confirmation you will see the used folder in the Connection String window.

### 6.2.2 Microsoft SQL

With the support of Microsoft SQL Server (Azure SQL as well) you have an additional on-premises solution to store Analytics binary data. You have just to provide the connection string for your MsSQL Server.
To make that simple you are able to click on the “SQL ConnString” button to open the input mask. There you can provide the configuration settings. Also of remote databases which are reachable by network connections.

After starting the WinService and finishing the very first configuration of the Recorder tool we start the communication to the database. In this moment the Storage Provider is creating the four necessary tables by himself. As example see the following screenshot of Microsoft SQL Server Management Studio.
6.2.3 Microsoft Azure Blob

To use the Microsoft Azure Blob store you need to have a Microsoft Azure Cloud account. There you get also your individual connection string for the configuration of the TwinCAT Analytics Storage Provider.
You have to copy the connection string into the description field. In Azure itself, the storage is to be generated. Choose “Storage accounts (classic)”. 

After generating the storage you will find under “Access keys” the secondary connection string. This string is to be use in the Analytics Storage Provider configuration.
6.3 Recorder

The Analytics Storage Provider Recorder is part of the Analytics Engineering setups. So you will find the Recorder in the installation of TwinCAT Analytics Workbench and TwinCAT Analytics Service Tool.

Toolbar

(1) Connection Settings

Enter here your message broker data such as URL, user name, password or certificates.

Connection Settings

Broker: MyMessageBroker.cloudapp.net
User: TwinCAT Pwd: ••••• Port: 8883

Certificate

CA: C:\Certificates\MyCA.crt
Cert: C:\Certificates\MyClientCert.crt
Key: C:\Certificates\MyClientCert.key

The settings can be checked. The result is displayed in the following window:
(2) Find TwinCAT Analytics StorageProvider

With the given connection settings, the recorder will connect to the message broker and search for every connected StorageProvider.

(3) TwinCAT Analytics StorageProvider Instances

This DropDown field contains all connected StorageProvider instances from the connected message broker.

(4) Info

Shows the description and other settings of the selected StorageProvider instance.

(5) Filter

It is possible to filter the view of the live running recordings. You can see all recordings or you can see only the recordings which were started from your recorder.

(6) Messages

Shows all Logs from the StorageProvider instance.

(7) Clean Recordings

It is possible to delete recordings from the StorageProvider instance.
(8) Download Recordings

You can download recordings which are stored in the cloud or on other systems from the StorageProvider into a local AnalyticsFile.
(9) RecorderAlias

Insert an alias name for the Recorder to get a better overview of all running recordings. Who has start the recording?

Start a recording

If you drag some symbols from the TwinCAT TargetBrowser into the Recorder Window the following Dialog will start up.
All needed settings like record duration, record name, can be made. It is also possible to set the ringbuffer options. Here are the different options:

### 6.4 Working with Historical Data

Historical Data can be analysed with the Analytics Workbench or the Analytics Service Tool. To see your recorded data, you need the TwinCAT Target Browser.

**Select data from TwinCAT Target Browser**

The Historical Data can directly be dragged from the Target Browser to an input of an analytics algorithm. First of all you have to click in the left corner of the Target Browser on TcAnalytics. There you can see your configured broker which lists the live and historical data of your different devices. This should look like in the following figure.
Go to the historical stream that you have created and select the record you want to analyse. All of your records are listed to the right in the Record Window. By default the last record is selected.

When you are live recording, the time range of the record will be updated every few seconds. By default the whole time range of a record is used. Additionally you can edit the start and end time to analyse your desired data range. This can be done with a range slider, textboxes or in a graphical calendar view. When you click on the icon right next to the textboxes, the calendar view will be displayed.

After these steps you can drag and drop a symbol to an input of an algorithm as well as you can do with the symbols of the live data.

After you have done this, a new input source is generated for your historical stream and can be seen in the Solution Explorer of your Visual Studio. Firstly, the dragged symbol and a timestamp of the current device time are listed below this stream. New dragged symbols of this stream will be listed as well there.
Analyse your historical data in the Analytics Configurator

To analyse your historical data press on the Start Analytics button. In contrast to analysing live data, a green progress bar appears. The speed of your analysis depends on your record length, the amount and size of your symbols as well as on your broadband speed to the broker. The analysis stops automatically when the progress bar ends. The results will remain visible.

Analysis of your historical data in your Analytics Runtime

You can provide the configuration with your historical data to an Analytics Runtime (PLC). In addition to the historical data, the live data is also analyzed. This allows you to switch between them and not lose live data by streaming historical data. The reason for this is that they are separated into two different tasks. The start of the analysis of historical data must be triggered.

- Computing time for historical data

  Unlike the Analytics Configurator, the analysis of historical data in the PLC takes a similar amount of time as the original recording of the data. Depending on the packet size and the set sampling rate, the processing of the data can be shortened compared to the recording. However, cycle over-runs due to excessively large packets must be taken into account.

Main differences of the folder structure in the created PLC project:
Implementation of the logic in your TwinCAT HMI

The preparation and writing of values in your PLC are for testing purposes. It is recommended to implement this and other logic in the PLC code with interactions from your TwinCAT HMI application if required.

You can start historical data analysis by triggering bGetHistoricalData in stCtrl_T1. The cancellation takes place by triggering bCancelHistoricalData.

This can be done in the MAIN_Analytics_Historical file as shown in the following figure:
To switch between live and historical data results for your HMI dashboard, you can set the `bHistorical` symbol in the **AnalyticsHMI** GVL. With this option, you do not need any additional controls to display historical data (of course, you can also use your own controls for historical data). The analysis of the live data is not interrupted by calling up some historical data. After viewing the historical data, you can switch back to the current live results. This change only affects the variables in your GVL.
7 PLC API

7.1 Function blocks

7.1.1 Topic Architecture

7.1.1.1 T_ALY_SPGetHistorical_Cmd

Syntax

Definition:

```
FUNCTION_BLOCK T_ALY_SPGetHistorical_Cmd EXTENDS T_ALY_JsonPayload
VAR_INPUT
    sTopic : T_MaxString;
    sLayout : GUID;
    eMode : E_SymbolMode := E_SymbolMode.All;
    eOutputFormat : E_RawDataFormat := E_RawDataFormat.Bin;
    nMaxSampleCount : UDINT := 3000;
    nUserSampleTime : DINT := -1;
    nRecordID : DINT;
    nStartTimestamp : LINT;
    nEndTimestamp : LINT;
    sResultTopic : T_MaxString;
    arrSymbol : ARRAY [0..255] OF T_ALY_Symbol
END_VAR
```

Inheritance hierarchy

T_ALY_JsonPayload [63]

T_ALY_SPGetHistorical_Cmd
Inputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sTopic</td>
<td>T_MaxString</td>
<td>Topic name of the recorded Live Stream</td>
</tr>
<tr>
<td>sLayout</td>
<td>GUID</td>
<td>Layout GUID of the recording</td>
</tr>
<tr>
<td>eMode</td>
<td>E_SymbolMode</td>
<td>Get all symbols or only a subset</td>
</tr>
<tr>
<td>eOutputFormat</td>
<td>E_RawDataFormat</td>
<td>Format of the returned data (actually only “Bin” supported)</td>
</tr>
<tr>
<td>nMaxSampleCount</td>
<td>UDINT</td>
<td>Max count of samples in one payload packet</td>
</tr>
<tr>
<td>nUserSampleTime</td>
<td>DINT</td>
<td>Sampletime in milliseconds of the returned stream. (-1 use the recorded sampletime)</td>
</tr>
<tr>
<td>nRecordID</td>
<td>DINT</td>
<td>Number of the record</td>
</tr>
<tr>
<td>nStartTimestamp</td>
<td>LINT</td>
<td>StartTime</td>
</tr>
<tr>
<td>nEndTimestamp</td>
<td>LINT</td>
<td>EndTime</td>
</tr>
<tr>
<td>sResultTopic</td>
<td>T_MaxString</td>
<td>Topic name of the result stream</td>
</tr>
<tr>
<td>arrSymbol</td>
<td>ARRAY [0..255] OF T_ALY_Symbol</td>
<td>If SymbolMode is Subset, only the list of this symbols will be returned</td>
</tr>
</tbody>
</table>

Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Definition location</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset</td>
<td></td>
<td>Reset all values inside of the payload FB</td>
</tr>
<tr>
<td>Init_JsonValue</td>
<td>Inherited from T_ALY_JsonPayload</td>
<td>Initialize FB with Json object</td>
</tr>
<tr>
<td>Init_String</td>
<td>Inherited from T_ALY_JsonPayload</td>
<td>Initialize FB with Json string</td>
</tr>
<tr>
<td>GetJsonLength</td>
<td>Inherited from T_ALY_JsonPayload</td>
<td>Get Length of Json payload</td>
</tr>
<tr>
<td>GetJsonString</td>
<td>Inherited from T_ALY_JsonPayload</td>
<td>Get Json payload as String</td>
</tr>
</tbody>
</table>

Development environment | Target platform | PLC libraries to include |
--------------------------|-----------------|--------------------------|
TwinCAT v3.1.4022.25     | PC or CX (x64, x86, ARM) | Tc3_AnalyticsStorageProvider |
7.1.1.2 T_ALY_SPRecordData_Cmd

Syntax

Definition:

FUNCTION BLOCK T_ALY_SPRecordData_Cmd EXTENDS T_ALY_JsonPayload
VAR_INPUT
  sAlias : T_MaxString;
  sRecordName : T_MaxString;
  eRecording : E_RecordMode;
  sRecorder : GUID;
  sRecorderAlias : T_MaxString;
  sTopic : T_MaxString;
  eDataFormat : E_RawDataFormat;
  nDuration : DINT;
  eRingBufferMode : E_RingBufferMode;
  nRingBufferParameter : DINT;
  eMode : E_SymbolMode;
  sSymbolLayout : GUID;
  arrSymbols : ARRAY [0..255] OF T_ALY_Symbol
END_VAR

Inheritance hierarchy

T_ALY_JsonPayload [63]
  T_ALY_SPRecordData_Cmd
## Inputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sAlias</td>
<td>T_MaxString</td>
<td>Alias name for the Recording</td>
</tr>
<tr>
<td>sRecordName</td>
<td>T_MaxString</td>
<td>Name for this record</td>
</tr>
<tr>
<td>eRecording</td>
<td>E_RecordMode</td>
<td>Start or Stop the recording</td>
</tr>
<tr>
<td>sRecorder</td>
<td>GUID</td>
<td>Individual GUID of the recorder</td>
</tr>
<tr>
<td>sRecorderAlias</td>
<td>T_MaxString</td>
<td>Alias name for the recorder</td>
</tr>
<tr>
<td>sTopic</td>
<td>T_MaxString</td>
<td>Topic name of the live stream</td>
</tr>
<tr>
<td>eDataFormat</td>
<td>E_RawDataFormat</td>
<td>Store data format. (actually only Binary format is supported)</td>
</tr>
<tr>
<td>nDuration</td>
<td>DINT</td>
<td>Duration in minutes of the recording. (-1 unlimited)</td>
</tr>
<tr>
<td>eRingBufferMode</td>
<td>E_RingBufferMode</td>
<td>Ringbuffer modus</td>
</tr>
<tr>
<td>nRingBufferParameter</td>
<td>DINT</td>
<td>TimeBased =&gt; Parameter in minutes DataBased =&gt; Parameter in Megabytes</td>
</tr>
<tr>
<td>eMode</td>
<td>E_SymbolMode</td>
<td>Record all symbols or only a subset</td>
</tr>
<tr>
<td>sSymbolLayout</td>
<td>GUID</td>
<td></td>
</tr>
<tr>
<td>arrSymbols</td>
<td>ARRAY [0..255] OF T_ALY_Symbol</td>
<td>If SymbolMode is Subset, only the list of this symbols will be recorded</td>
</tr>
</tbody>
</table>

## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Definition location</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset</td>
<td></td>
<td>Reset all values inside of the payload FB</td>
</tr>
<tr>
<td>Init_JsonValue</td>
<td>Inherited from T_ALY_JsonPayload</td>
<td>Initialize FB with Json object</td>
</tr>
<tr>
<td>Init_String</td>
<td>Inherited from T_ALY_JsonPayload</td>
<td>Initialize FB with Json string</td>
</tr>
<tr>
<td>GetJsonLength</td>
<td>Inherited from T_ALY_JsonPayload</td>
<td>Get Length of Json payload</td>
</tr>
<tr>
<td>GetJsonString</td>
<td>Inherited from T_ALY_JsonPayload</td>
<td>Get Json payload as String</td>
</tr>
</tbody>
</table>

## Development environment

<table>
<thead>
<tr>
<th>Target platform</th>
<th>PLC libraries to include</th>
</tr>
</thead>
<tbody>
<tr>
<td>PC or CX (x64, x86, ARM)</td>
<td>Tc3_AnalyticsStorageProvider</td>
</tr>
</tbody>
</table>

### 7.1.1.3 T_ALY_SPCancel_Cmd

**Syntax**

```
T_ALY_SPCancel_Cmd
  _eCancelType E_CancelType
  _arrParameter ARRAY[0..99] OF T_MaxString
```

**Definition:**


FUNCTION_BLOCK T_ALY_SPCancel_Cmd EXTENDS T_ALY_JsonPayload
VAR_INPUT
  eCancelType : E_CancelType;
  arrParameter : ARRAY [0..99] OF T_MaxString;
END_VAR

Inheritance hierarchy

T_ALY_JsonPayload [63]
  T_ALY_SPCancel_Cmd

Inputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eCancelType</td>
<td>E_CancelType [67]</td>
<td></td>
</tr>
<tr>
<td>arrParameter</td>
<td>ARRAY [0..99] OF T_MaxString</td>
<td></td>
</tr>
</tbody>
</table>

Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Definition location</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset</td>
<td></td>
<td>Reset all values inside of the payload FB</td>
</tr>
<tr>
<td>Init_JsonValue</td>
<td>Inherited from T_ALY_JsonPayload [63]</td>
<td>Initialize FB with Json object</td>
</tr>
<tr>
<td>Init_String</td>
<td>Inherited from T_ALY_JsonPayload [63]</td>
<td>Initialize FB with Json string</td>
</tr>
<tr>
<td>GetJsonLength</td>
<td>Inherited from T_ALY_JsonPayload [63]</td>
<td>Get Length of Json payload</td>
</tr>
<tr>
<td>GetJsonString</td>
<td>Inherited from T_ALY_JsonPayload [63]</td>
<td>Get Json payload as String</td>
</tr>
</tbody>
</table>

Development environment

<table>
<thead>
<tr>
<th>Target platform</th>
<th>PLC libraries to include</th>
</tr>
</thead>
<tbody>
<tr>
<td>PC or CX (x64, x86, ARM)</td>
<td>Tc3_AnalyticsStorageProvider</td>
</tr>
</tbody>
</table>

7.1.1.4 SubTypes

7.1.1.4.1 T_ALY_Symbol

Syntax

Definition:
FUNCTION_BLOCK T_ALY_Symbol EXTENDS T_ALY_JsonPayload
VAR_INPUT
  sName : T_MaxString;
  sBaseType : T_MaxString;
  nBitOffset : UDINT;
  nBitSize : UDINT;
END_VAR
Inheritance hierarchy

T_ALY_JsonPayload

T_ALY_Symbol

Inputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sName</td>
<td>T_MaxString</td>
<td>Name of the symbol</td>
</tr>
<tr>
<td>sBaseType</td>
<td>T_MaxString</td>
<td>DataType of the symbol</td>
</tr>
<tr>
<td>nBitOffset</td>
<td>UDINT</td>
<td>BitOffset of the symbol</td>
</tr>
<tr>
<td>nBitSize</td>
<td>UDINT</td>
<td>BitSize of the symbol</td>
</tr>
</tbody>
</table>

Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Definition location</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset</td>
<td></td>
<td>Reset all values inside of the payload FB</td>
</tr>
<tr>
<td>InitJsonValue</td>
<td>Inherited from T_ALY_JsonPayload[63]</td>
<td>Initialize FB with Json object</td>
</tr>
<tr>
<td>InitString</td>
<td>Inherited from T_ALY_JsonPayload[63]</td>
<td>Initialize FB with Json string</td>
</tr>
<tr>
<td>GetJsonLength</td>
<td>Inherited from T_ALY_JsonPayload[63]</td>
<td>Get Length of Json payload</td>
</tr>
<tr>
<td>GetJsonString</td>
<td>Inherited from T_ALY_JsonPayload[63]</td>
<td>Get Json payload as String</td>
</tr>
</tbody>
</table>

Development environment

| TwinCAT v3.1.4022.25 | PC or CX (x64, x86, ARM) | Tc3_AnalyticsStorageProvider |

7.1.1.5 [Internal]

7.1.1.5.1 T_ALY_JsonPayload

Syntax

Definition:

```plaintext
FUNCTION_BLOCK INTERNAL T_ALY_JsonPayload
```
## Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Definition location</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reset</td>
<td></td>
<td>Reset all values inside of the payload FB</td>
</tr>
<tr>
<td>InitJsonValue</td>
<td>Inherited from T_ALY_JsonPayload [63]</td>
<td>Initialize FB with Json object</td>
</tr>
<tr>
<td>InitString</td>
<td>Inherited from T_ALY_JsonPayload [63]</td>
<td>Initialize FB with Json string</td>
</tr>
<tr>
<td>GetJsonLength</td>
<td>Inherited from T_ALY_JsonPayload [63]</td>
<td>Get Length of Json payload</td>
</tr>
<tr>
<td>GetJsonString</td>
<td>Inherited from T_ALY_JsonPayload [63]</td>
<td>Get Json payload as String</td>
</tr>
</tbody>
</table>

### Development environment

<table>
<thead>
<tr>
<th>Target platform</th>
<th>PLC libraries to include</th>
</tr>
</thead>
<tbody>
<tr>
<td>PC or CX (x64, x86, ARM)</td>
<td>Tc3_AnalyticsStorageProvider</td>
</tr>
</tbody>
</table>

### Syntax

**Definition:**

```plaintext
FUNCTION_BLOCK FB_ALY_StorageProvider
VAR_INPUT
    stConfig : ST_ALY_SP_Config;
END_VAR
VAR_OUTPUT
    bBusy : BOOL;
    bError : BOOL;
    ipResultMessage : I_TcMessage;
    eConnectionState : ETcIotMqttClientState;
END_VAR
```

### Inputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stConfig</td>
<td>ST_ALY_SP_Config [68]</td>
<td>Struct to configure the FB</td>
</tr>
</tbody>
</table>
### Outputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bBusy</td>
<td>BOOL</td>
<td>TRUE as soon as a method of the function block is active.</td>
</tr>
<tr>
<td>bError</td>
<td>BOOL</td>
<td>Becomes TRUE as soon as an error situation occurs.</td>
</tr>
<tr>
<td>ipResultMessage</td>
<td>I_TcMessage</td>
<td>Message interface from the TwinCAT 3 EventLogger, which provides details on the return value.</td>
</tr>
<tr>
<td>eConnectionState</td>
<td>ETcIotMqttClientState</td>
<td>Indicates the state of the connection between client and broker as enumeration ETcIotMqttClientState.</td>
</tr>
</tbody>
</table>

### Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Definition location</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Call</td>
<td>Local</td>
<td>Method for background communication with the TwinCAT driver. The method must be called cyclically.</td>
</tr>
<tr>
<td>Cancel</td>
<td>Local</td>
<td>Method to cancel activities from the TwinCAT Analytics StorageProvider</td>
</tr>
<tr>
<td>GetHistoricalData</td>
<td>Local</td>
<td>Method to order historical Data</td>
</tr>
<tr>
<td>StartRecord</td>
<td>Local</td>
<td>Starts the recording of a live MQTT binary stream</td>
</tr>
<tr>
<td>StopRecord</td>
<td>Local</td>
<td>Stops the selected recording</td>
</tr>
</tbody>
</table>

### Development environment

**Target platform**: PC or CX (x64, x86, ARM)

**PLC libraries to include**: Tc3_AnalyticsStorageProvider

### 7.1.2.1 Cancel

Syntax

```plaintext
METHOD Cancel : BOOL
VAR_INPUT
    stCmd : REFERENCE TO T_ALY_SPCancel_Cmd;
END_VAR
```

### Inputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stCmd</td>
<td>REFERENCE TO T_ALY_SPCancel_Cmd</td>
<td>Json command to cancel operations of the TwinCAT Analytics Storage Provider</td>
</tr>
</tbody>
</table>

### Return value

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cancel</td>
<td>BOOL</td>
<td>Is TRUE if done</td>
</tr>
</tbody>
</table>
7.1.2.2 GetHistoricalData

Syntax

METHOD GetHistoricalData : BOOL
VAR_INPUT
  stCmd : REFERENCE TO T_ALY_SPGetHistorical_Cmd;
END_VAR

Inputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stCmd</td>
<td>REFERENCE TO T_ALY_SPGetHistorical_Cmd</td>
<td>Json command to get historical data from TwinCAT Analytics Storage Provider</td>
</tr>
</tbody>
</table>

Return value

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetHistoricalData</td>
<td>BOOL</td>
<td>Is TRUE if done</td>
</tr>
</tbody>
</table>

7.1.2.3 StartRecord

Syntax

METHOD StartRecord : BOOL
VAR_INPUT
  stCmd : REFERENCE TO T_ALY_SPRecordData_Cmd;
END_VAR

Inputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stCmd</td>
<td>REFERENCE TO T_ALY_SPRecordData_Cmd</td>
<td>Json command to start recording a live stream</td>
</tr>
</tbody>
</table>

Return value

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>StartRecord</td>
<td>BOOL</td>
<td>Is TRUE if done</td>
</tr>
</tbody>
</table>

7.1.2.4 StopRecord

Syntax

METHOD StopRecord : BOOL
VAR_INPUT
  stCmd : REFERENCE TO T_ALY_SPRecordData_Cmd;
END_VAR

Return value

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>StartRecord</td>
<td>BOOL</td>
<td>Is TRUE if done</td>
</tr>
</tbody>
</table>
Syntax
METHOD StopRecord : BOOL
VAR_INPUT
  stCmd : REFERENCE TO T_ALY_SPRecordData_Cmd;
END_VAR

Inputs

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stCmd</td>
<td>REFERENCE TO T_ALY_SPRecordData_Cmd [60]</td>
<td>Json command to stop recording of a live stream</td>
</tr>
</tbody>
</table>

Return value

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>StopRecord</td>
<td>BOOL</td>
<td>Is TRUE if done</td>
</tr>
</tbody>
</table>

7.1.2.5 Call

Syntax
METHOD Call : BOOL

Return value

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Call</td>
<td>BOOL</td>
<td></td>
</tr>
</tbody>
</table>

7.2 Data types

7.2.1 E_CancelType

Syntax

Definition:

```
TYPE E_CancelType :
  HistoricalData := 0,
  AllRecordData
) INT;
END_TYPE
```

Parameter

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HistoricalData</td>
<td>Canceled the selected historical data stream</td>
</tr>
<tr>
<td>AllRecordData</td>
<td>Canceled all running recordings</td>
</tr>
</tbody>
</table>
7.2.2 E_RawDataFormat

Syntax

Definition:

```plaintext
TYPE E_RawDataFormat :
  (    Bin := 0,
       Json
  )INT;
END_TYPE
```

Parameter

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bin</td>
<td>Analytics Binary Stream Format</td>
</tr>
<tr>
<td>Json</td>
<td>TwinCAT Json Format (actually not supported)</td>
</tr>
</tbody>
</table>

7.2.3 E_SymbolMode

Syntax

Definition:

```plaintext
TYPE E_SymbolMode :
  (    All := 0,
       Subset
  )INT;
END_TYPE
```

Parameter

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>All symbols of the stream will be used</td>
</tr>
<tr>
<td>Subset</td>
<td>Only a subset of symbols will be used</td>
</tr>
</tbody>
</table>

7.2.4 ST_ALY_SP_Config

Syntax

Definition:

```plaintext
TYPE ST_Msg :
  STRUCT
    sMainTopic     : T_MaxString;
    sProviderGuid  : GUID;
    stConnSettings : ST_ConnectionSettings
  END_STRUCT
END_TYPE
```
<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sMainTopic</td>
<td>T_MaxString</td>
<td>The main topic where the TwinCAT Analytics Storage Provider is located on the message broker.</td>
</tr>
<tr>
<td>sProviderGuid</td>
<td>GUID</td>
<td>The individual GUID of the TwinCAT Analytics Storage Provider Instance.</td>
</tr>
<tr>
<td>stConnSettings</td>
<td>ST_ConnectionSettings</td>
<td>MQTT connection settings to connect with the message broker.</td>
</tr>
</tbody>
</table>

### 7.2.5 ST_ConnectionSettings

**Syntax**

**Definition:**

```plaintext
TYPE ST_ConnectionSettings :
  STRUCT
    sHostName : T_MaxString;
    nHostPort : UINT := 1883;
    sUserId : T_MaxString;
    sPassword : T_MaxString;
    bWithCertificate : BOOL := BOOL;
    sCA : T_MaxString;
    sCert : T_MaxString;
    sKey : T_MaxString;
    sKeyPwd : T_MaxString;
  END_STRUCT
END_TYPE
```

**Parameter**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sHostName</td>
<td>T_MaxString</td>
<td>sHostName can be specified as name or as IP address. If no information is provided, the local host is used.</td>
</tr>
<tr>
<td>nHostPort</td>
<td>UINT</td>
<td>The host port can be specified here. The default is 1883.</td>
</tr>
<tr>
<td>sUserId</td>
<td>T_MaxString</td>
<td>Optionally, a user name can be specified.</td>
</tr>
<tr>
<td>sPassword</td>
<td>T_MaxString</td>
<td>A password for the user name can be entered here.</td>
</tr>
<tr>
<td>bWithCertificate</td>
<td>BOOL</td>
<td>If TRUE the certificates will be used for communication.</td>
</tr>
<tr>
<td>sCA</td>
<td>T_MaxString</td>
<td>Certificate of the certificate authority (CA)</td>
</tr>
<tr>
<td>sCert</td>
<td>T_MaxString</td>
<td>Client certificate to be used for authentication at the broker.</td>
</tr>
<tr>
<td>sKey</td>
<td>T_MaxString</td>
<td>Private key of the client</td>
</tr>
<tr>
<td>sKeyPwd</td>
<td>T_MaxString</td>
<td>Password of the private key, if applicable</td>
</tr>
</tbody>
</table>

### 7.2.6 E_RecordMode

**Syntax**

**Definition:**

```plaintext
TYPE E_RecordMode :
  {
    Start := 0,
  }
```

Parameter

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start</td>
<td>Starts the recording of the configured record</td>
</tr>
<tr>
<td>Stop</td>
<td>Stops the recording</td>
</tr>
</tbody>
</table>

### 7.2.7 E_RingBufferMode

**Syntax**

**Definition:**

```plaintext
definition:

TYPE E_RingBufferMode:
{
    None := 0,
    TimeBased,
    DataBased
} INT;
END_TYPE
```

**Parameter**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>Recording without ringbuffer mode</td>
</tr>
<tr>
<td>TimeBased</td>
<td>Ringbuffer based on a given time period</td>
</tr>
<tr>
<td>DataBased</td>
<td>Ringbuffer based on a given max data size</td>
</tr>
</tbody>
</table>
8 Samples

8.1 PLC Client

This PLC sample shows the use of the TwinCAT Analytics Storage Provider library. The sample code shows reading and writing. For the sample to work coherently, both the use of the Analytics Logger for sending measured data to an MQTT Message Broker and the import of historical data via the Analytics Stream Helper are shown.

The basis is an appropriately set up native MQTT Message Broker and an Analytics Storage Provider service.

The PLC sample shows the following steps:

2. Read Stream Description: reception of the stream description from the MQTT Message Broker. Evaluation of the JSON description for the use of the necessary IDs.
3. Analytics Storage Provider: starting and stopping recordings, as well as reading recordings and historical data.
4. Analytics Stream Helper: receiving the historical data from the Analytics Storage Provider and mapping the data into a Global Variable List for the historical data.

To use the Storage Provider in the PLC, different GUIDs are necessary for the identification of services and data. The following screenshots show where the corresponding GUIDs can be found.

- TwinCAT System ID
- Symbol Info GUID
- Storage Provider Recorder GUID
- Storage Provider Service GUID

**TwinCAT System ID**

![TwinCAT System ID Screenshot](image-url)
Symbol Info GUID
Identification ID for the symbols from the TwinCAT Analytics binary format.

Storage Provider Recorder GUID
Identification GUID for the Storage Provider Recorder in Visual Studio® - this is installed with the TwinCAT Measurement setup.

Storage Provider Service GUID
GUID of the individually used Storage Provider Service. If you are connected to a Storage Provider in the Recorder window, you can click the information icon for this GUID.
Sample code architecture

All relevant parts of the configuration and the program code are marked in the following picture:

Stream Helper

For receiving the historical data sent by the Analytics Storage Provider via MQTT.

Variable Live/Historical

The GVL is for the live data and the GVL_Hist is for the historical data.

Storage Provider Command Helper Functions

These Helper Functions generate the commands for communication with the Storage Provider Service in JSON format.
Description Helper Function Block

This function block receives the information from the Description Topic, including the information about the SystemID.

MAIN Programs

The Main program invokes communication to the Analytics Storage Provider. The Main Historical program implements the mapping of historical data from the Stream Helper into the GVL_Hist.

Analytics Logger

Sends the variables of the GVL to an MQTT Message Broker.

Sample Start

Before the sample can be started, you must set the MQTT Message Broker you are using in three different places.

Analytics Stream Helper:

<table>
<thead>
<tr>
<th>Object</th>
<th>Context</th>
<th>Parameter (Int)</th>
<th>Parameter (Online)</th>
<th>Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Value</td>
<td>CS</td>
<td>Type</td>
<td>PTCID</td>
</tr>
<tr>
<td>MQTT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Host Name</td>
<td>172.17.62.145</td>
<td>□</td>
<td>STRING(80)</td>
<td>0x020201...</td>
</tr>
<tr>
<td>TcpPort</td>
<td>1883</td>
<td>□</td>
<td>UINT</td>
<td>0x020201...</td>
</tr>
<tr>
<td>Topic Prefix</td>
<td>_AlySPTest/ResultValues</td>
<td>□</td>
<td>STRING(255)</td>
<td>0x020300...</td>
</tr>
<tr>
<td>Client ID</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Username</td>
<td></td>
<td></td>
<td>STRING(80)</td>
<td>0x020201...</td>
</tr>
<tr>
<td>Password</td>
<td></td>
<td></td>
<td>STRING(80)</td>
<td>0x020201...</td>
</tr>
<tr>
<td>Max Handles</td>
<td>100</td>
<td>□</td>
<td>UDINT</td>
<td>0x020300...</td>
</tr>
<tr>
<td>Fifo Exponent</td>
<td>8</td>
<td>□</td>
<td>UDINT</td>
<td>0x020300...</td>
</tr>
<tr>
<td>Sender System Id</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Tls

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>CS</th>
<th>Type</th>
<th>PTCID</th>
</tr>
</thead>
<tbody>
<tr>
<td>CaFile</td>
<td></td>
<td>□</td>
<td>STRING(255)</td>
<td>0x020201...</td>
</tr>
<tr>
<td>CertFile</td>
<td></td>
<td>□</td>
<td>STRING(255)</td>
<td>0x020201...</td>
</tr>
<tr>
<td>KeyFile</td>
<td></td>
<td>□</td>
<td>STRING(255)</td>
<td>0x020201...</td>
</tr>
<tr>
<td>KeyPw</td>
<td></td>
<td>□</td>
<td>STRING(255)</td>
<td>0x020201...</td>
</tr>
<tr>
<td>CtrlFile</td>
<td></td>
<td>□</td>
<td>STRING(255)</td>
<td>0x020201...</td>
</tr>
<tr>
<td>Pskld</td>
<td></td>
<td>□</td>
<td>STRING(255)</td>
<td>0x020201...</td>
</tr>
<tr>
<td>PskPwd</td>
<td></td>
<td>□</td>
<td>STRING(255)</td>
<td>0x020201...</td>
</tr>
<tr>
<td>Insecure</td>
<td>FALSE</td>
<td>□</td>
<td>BOOL</td>
<td>0x020201...</td>
</tr>
</tbody>
</table>

Analytics Logger:
MAIN program:

You must then change the sProviderGUID for the FB_ALY_StorageProvider. This can be found as described above in this document.

Now go to the MAIN program to control the sample. With the enum eCtrl you can set the action you would like to perform. The available options are:

- StartRecord
- StopRecord
- ReadRecords
- GetHistorical

With a rising edge at the variable bExecute the action selected in the enum is executed. If you have made more than one record, you can see this in the array aRecordInfo. With the index it is then possible to select the different records. The timespans are also displayed, you could theoretically still adjust these within the timespan. To do this, you would need to modify the logic of the sample in the helper function F_CreateAlySPGetHistCmd accordingly.

Documents about this

- TcALYStorageProviderSample (Resources/zip/11270100747.zip)
9 Appendix

9.1 FAQ - frequently asked questions and answers

In this section frequently asked questions are answered, in order to facilitate your work with the TwinCAT Analytics Storage Provider (ASP). If you have any further questions, please contact our support team at support@beckhoff.com.

1. How can I manage the table schema of MS SQL with ASP? [76]
2. Can I control the Storage Provider in a programmable way? [76]
3. Is it also possible to save results from the Analytics Runtime? [76]
4. Are open source software components used in TwinCAT Measurement products? [76]

How can I manage the table schema of MS SQL with ASP?

You don't have to worry about the table schema. This is done completely by the Analytics Storage Provider. You only have to specify on which database server the data should be stored. If you want to see data in your own table structure, you have to stream the data into a TwinCAT Analytics Runtime and have the TwinCAT Database Server write the data in your structure.

Can I control the Storage Provider in a programmable way?

Yes, via the PLC interface for the TwinCAT Storage Provider. You can start/stop recordings or retrieve historical data (raw data or result data).

Is it also possible to save results from the Analytics Runtime?

Yes, this is possible. For this purpose, you can choose to send the results to an MQTT Message Broker when generating the Analytics Runtime from the Analytics Workbench configurator. This data stream can be captured by the Storage Provider.

Are open source software components used in TwinCAT Measurement products?

Yes, various open source components are used.

Please see the information on the page Third-party components [76].

9.2 Third-party components

This software contains third-party components. Please refer to the license file provided in the following folder for further information:

…\TwinCAT\Functions\TwinCAT Measurement\Legal